

# **BIG-IP® Controller Administrator Guide**

version 3.3







---

# Service and Support Information

## Product Version

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

## Obtaining Technical Support

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

## Contacting F5 Networks

|                        |                                                      |
|------------------------|------------------------------------------------------|
| <b>Web</b>             | www.f5.com                                           |
| <b>Toll-free phone</b> | (888) 961-7242                                       |
| <b>Corporate phone</b> | (206) 272-5555                                       |
| <b>Fax</b>             | (206) 272-5556                                       |
| <b>Email</b>           | sales@f5.com                                         |
| <b>Mailing Address</b> | 501 Elliott Avenue West<br>Seattle, Washington 98119 |

---

## Legal Notices

### Copyright

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

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

### Trademarks

F5, BIG-IP, and 3-DNS are registered trademarks of F5 Networks, Inc. SEE-IT, 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® Local Traffic 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® Local Traffic 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](http://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**

|                                                               |     |
|---------------------------------------------------------------|-----|
| Getting started .....                                         | I-1 |
| Choosing a solution .....                                     | I-1 |
| Choosing a configuration tool .....                           | I-3 |
| Using the Administrator Kit .....                             | I-4 |
| Stylistic conventions .....                                   | I-5 |
| Finding additional help and technical support resources ..... | I-7 |
| What's new in version 3.3 .....                               | I-8 |
| BIG-IP e-Commerce Controller .....                            | I-8 |
| BIG-IP Cache Load Balancer .....                              | I-8 |
| Performance enhancements .....                                | I-9 |
| Learning more about BIG-IP Controller product family .....    | I-9 |

## **Chapter 1**

### **Basic Web Site and e-Commerce Configuraton**

|                                                     |     |
|-----------------------------------------------------|-----|
| Basic web site and e-commerce configuration .....   | 1-1 |
| A basic web site and e-commerce configuration ..... | 1-1 |
| Setting up the topology .....                       | 1-1 |

## **Chapter 2**

### **A Simple Intranet Configuration**

|                                       |     |
|---------------------------------------|-----|
| A simple intranet configuration ..... | 2-1 |
| Setting up the topology .....         | 2-2 |
| Using additional features .....       | 2-3 |

## **Chapter 3**

### **Configuring an SSL Accelerator**

|                                                                   |      |
|-------------------------------------------------------------------|------|
| Introducing the SSL Accelerator .....                             | 3-1  |
| Hardware acceleration options .....                               | 3-2  |
| Configuring the SSL Accelerator .....                             | 3-3  |
| Generating a key and obtaining a certificate .....                | 3-3  |
| Installing certificates from the certification authority (CA) ... | 3-10 |
| Create an HTTP virtual server .....                               | 3-12 |
| Create an SSL gateway .....                                       | 3-13 |

|                                                                                          |      |
|------------------------------------------------------------------------------------------|------|
| Enabling, disabling, or deleting an SSL gateway .....                                    | 3-16 |
| Displaying the configuration for an SSL gateway from the<br>command line .....           | 3-17 |
| Optional SSL Accelerator configuration .....                                             | 3-18 |
| Create a last hop pool that includes additional network<br>devices .....                 | 3-19 |
| Modify the SSL gateway so that it references the last hop<br>pool .....                  | 3-20 |
| Introducing the SSL accelerator cell configuration .....                                 | 3-21 |
| Configuration tasks .....                                                                | 3-24 |
| Configuring the BIG-IP Controller which load balances the<br>SSL accelerator cells ..... | 3-25 |
| Configuring an SSL accelerator for use in a cell .....                                   | 3-30 |
| Setting the default route on each node in a cell .....                                   | 3-34 |
| Introducing the SSL accelerator half sandwich configuration ....                         | 3-35 |
| Configuration tasks .....                                                                | 3-36 |
| Configuring the BIG-IP Controllers handling inbound traffic .                            | 3-38 |
| Configuring each SSL accelerator .....                                                   | 3-41 |
| Configuring the BIG-IP Controller that load balances the<br>content servers .....        | 3-44 |
| Configuring the content servers .....                                                    | 3-50 |

## Chapter 4

### VPN Load Balancing

|                                                                        |     |
|------------------------------------------------------------------------|-----|
| VPN load balancing .....                                               | 4-1 |
| Configuring interfaces for VPN load balancing .....                    | 4-3 |
| Configuring virtual servers for VPN load balancing .....               | 4-3 |
| VPN and router load balancing .....                                    | 4-5 |
| Configuring interfaces for VPN load balancing .....                    | 4-7 |
| Configuring virtual servers for VPN and router load<br>balancing ..... | 4-7 |



## Chapter 5

### ISP Load Balancing

|                                                                         |     |
|-------------------------------------------------------------------------|-----|
| ISP load balancing .....                                                | 5-1 |
| Configuring interfaces for the additional internet connection .....     | 5-1 |
| Configuring virtual servers for an additional internet connection ..... | 5-2 |

## Chapter 6

### One IP Network Topologies

|                                                                                        |     |
|----------------------------------------------------------------------------------------|-----|
| One IP network topology with one interface .....                                       | 6-1 |
| Configuring the interface in the single interface topology .....                       | 6-1 |
| Defining a pool for the servers .....                                                  | 6-2 |
| Virtual server configuration .....                                                     | 6-2 |
| Client SNAT configuration .....                                                        | 6-2 |
| One IP network topology with two interfaces .....                                      | 6-3 |
| Configuring the interfaces in the single IP network with two interfaces topology ..... | 6-4 |
| Routing issues .....                                                                   | 6-4 |
| Defining a pool for the servers .....                                                  | 6-5 |
| Virtual server configuration .....                                                     | 6-5 |
| Client SNAT configuration .....                                                        | 6-5 |

## Chapter 7

### nPath Routing

|                                                                   |     |
|-------------------------------------------------------------------|-----|
| nPath routing .....                                               | 7-1 |
| Defining a virtual server with address translation disabled ..... | 7-1 |
| Setting the route through the BIG-IP Controller .....             | 7-4 |
| Setting the idle connection time-out .....                        | 7-4 |

## Chapter 8

### Balancing Traffic Inbound to Enterprise Servers

|                                                                                             |      |
|---------------------------------------------------------------------------------------------|------|
| Introducing firewall load balancing for inbound traffic .....                               | 8-1  |
| Configuration tasks .....                                                                   | 8-3  |
| Configuring routing to the internal network .....                                           | 8-3  |
| Creating pools for firewalls and servers .....                                              | 8-4  |
| Creating a pool for outside firewall interfaces .....                                       | 8-4  |
| Creating a pool for servers .....                                                           | 8-6  |
| Creating a pool for inside firewall interfaces .....                                        | 8-7  |
| Creating virtual servers for the firewall sandwich .....                                    | 8-8  |
| Creating a wildcard virtual server to load balance the<br>outside firewall interfaces ..... | 8-9  |
| Creating a standard virtual server to load balance the<br>enterprise servers .....          | 8-11 |
| Designating the last hop pool .....                                                         | 8-12 |
| Configuring interfaces .....                                                                | 8-14 |
| Configuring administrative routing .....                                                    | 8-15 |

## Chapter 9

### Balancing Traffic Outbound to the Internet

|                                                                |      |
|----------------------------------------------------------------|------|
| Introducing firewall load balancing for outbound traffic ..... | 9-1  |
| Configuration tasks .....                                      | 9-3  |
| Verifying user network routing .....                           | 9-3  |
| Creating a pool for the firewalls .....                        | 9-4  |
| Creating a wildcard virtual server .....                       | 9-5  |
| Enhancing security .....                                       | 9-7  |
| Configuring interfaces .....                                   | 9-8  |
| Configuring network address translation on firewalls .....     | 9-10 |
| Creating a route for return traffic .....                      | 9-10 |

## Chapter 10

### Balancing Two-Way Traffic With Destination Processing

|                                                                                        |       |
|----------------------------------------------------------------------------------------|-------|
| Introducing two-way firewall load balancing with destination processing .....          | 10-1  |
| Configuration tasks .....                                                              | 10-3  |
| Configuring routing to the internal network .....                                      | 10-3  |
| Creating pools for firewalls and servers .....                                         | 10-4  |
| Creating a pool for outside firewall interfaces .....                                  | 10-4  |
| Creating a pool for inside firewall interfaces .....                                   | 10-5  |
| Creating a pool for servers .....                                                      | 10-7  |
| Creating virtual servers for inbound traffic .....                                     | 10-8  |
| Creating a network virtual server to load balance the firewalls .....                  | 10-8  |
| Creating a standard virtual server to load balance intranet servers .....              | 10-10 |
| Designating a last hop pool for inbound traffic .....                                  | 10-12 |
| Creating virtual servers for outbound traffic .....                                    | 10-13 |
| Creating a wildcard virtual server for balancing traffic to the firewalls .....        | 10-14 |
| Creating a forwarding wildcard virtual server to forward traffic to the Internet ..... | 10-15 |
| Designating a last hop pool for outbound traffic .....                                 | 10-17 |
| Configuring interfaces .....                                                           | 10-17 |
| Configuring administrative routing .....                                               | 10-19 |

## Chapter 11

### Balancing Two-Way Traffic

|                                                               |       |
|---------------------------------------------------------------|-------|
| Introducing firewall load balancing for two-way traffic ..... | 11-1  |
| Configuration tasks .....                                     | 11-3  |
| Configuring routing to the internal network .....             | 11-3  |
| Creating pools for the firewalls .....                        | 11-4  |
| Creating a pool for outside firewall interfaces .....         | 11-4  |
| Creating a pool for inside firewall interfaces .....          | 11-5  |
| Creating virtual servers for inbound traffic .....            | 11-7  |
| Creating a network virtual server .....                       | 11-7  |
| Creating a forwarding virtual server .....                    | 11-10 |

|                                                                                             |       |
|---------------------------------------------------------------------------------------------|-------|
| Designating a last hop pool for inbound traffic .....                                       | 11-11 |
| Creating virtual servers for outbound traffic .....                                         | 11-12 |
| Creating a wildcard virtual server for balancing outbound<br>traffic to the firewalls ..... | 11-13 |
| Creating a forwarding wildcard virtual server to forward<br>traffic to the Internet .....   | 11-14 |
| Designating a last hop pool for outbound traffic .....                                      | 11-15 |
| Configuring interfaces .....                                                                | 11-16 |
| Configuring administrative routing .....                                                    | 11-18 |

## Chapter 12

### Configuring Local Server Acceleration

|                                                    |       |
|----------------------------------------------------|-------|
| Introducing local server acceleration .....        | 12-1  |
| Maximizing memory or processing power .....        | 12-2  |
| Using the configuration diagram .....              | 12-2  |
| Configuration tasks .....                          | 12-3  |
| Creating pools .....                               | 12-4  |
| Creating a pool for the cache servers .....        | 12-5  |
| Creating a pool for the origin server .....        | 12-6  |
| Creating a pool for hot content .....              | 12-7  |
| Creating a cache control rule .....                | 12-8  |
| Cacheable content expression .....                 | 12-8  |
| Content demand status .....                        | 12-11 |
| Creating a virtual server .....                    | 12-12 |
| Configuring for intelligent cache population ..... | 12-13 |
| Configuring a SNAT .....                           | 12-14 |
| Configuring interfaces .....                       | 12-15 |

## Chapter 13

### Configuring Remote Server Acceleration

|                                              |      |
|----------------------------------------------|------|
| Introducing remote server acceleration ..... | 13-1 |
| Maximizing memory or processing power .....  | 13-2 |
| Using the configuration diagram .....        | 13-3 |
| Configuration tasks .....                    | 13-4 |
| Creating pools .....                         | 13-4 |
| Creating a pool for the cache servers .....  | 13-5 |
| Creating a pool for the origin server .....  | 13-6 |

|                                                    |       |
|----------------------------------------------------|-------|
| Creating a pool for hot content .....              | 13-7  |
| Creating a cache control rule .....                | 13-8  |
| Cacheable content expression .....                 | 13-9  |
| Content demand status .....                        | 13-11 |
| Creating a virtual server .....                    | 13-12 |
| Configuring for intelligent cache population ..... | 13-14 |
| Configuring a SNAT .....                           | 13-15 |
| Configuring interfaces .....                       | 13-15 |
| Marking the origin server node as remote .....     | 13-18 |

## Chapter 14

### Configuring Forward Proxy Caching

|                                             |       |
|---------------------------------------------|-------|
| Introducing forward proxy caching .....     | 14-1  |
| Maximizing memory or processing power ..... | 14-2  |
| Using the configuration diagram .....       | 14-2  |
| Configuration tasks .....                   | 14-3  |
| Creating pools .....                        | 14-4  |
| Creating a pool for the cache servers ..... | 14-5  |
| Creating a pool for the origin server ..... | 14-6  |
| Creating a pool for hot content .....       | 14-7  |
| Creating a cache control rule .....         | 14-8  |
| Cacheable content expression .....          | 14-8  |
| Content demand status .....                 | 14-11 |
| Creating a virtual server .....             | 14-12 |

## Chapter 15

### SNAT and Virtual Servers Combined

|                                         |      |
|-----------------------------------------|------|
| SNAT and virtual servers combined ..... | 15-1 |
|-----------------------------------------|------|

## Chapter 16

### Setting up 802.1q VLAN Trunk Mode

|                                                    |      |
|----------------------------------------------------|------|
| Setting up 802.1q VLAN trunk mode .....            | 16-1 |
| Adding VLAN tag definitions to /etc/netstart ..... | 16-1 |
| Adding VLAN tag definitions to BIG/db .....        | 16-2 |
| Configuring multiple VLANs on one interface .....  | 16-2 |

|                                                                            |      |
|----------------------------------------------------------------------------|------|
| To enable or disable VLAN tags on the command line . . . . .               | 16-3 |
| Using ifconfig to add another VLAN . . . . .                               | 16-3 |
| Using netstat to view VLAN tags . . . . .                                  | 16-3 |
| Disabling and enabling VLAN tags using the Configuration utility . . . . . | 16-4 |

## Chapter 17

### Essential Configuration Tasks

|                                                                                  |       |
|----------------------------------------------------------------------------------|-------|
| Determining which configuration tasks to do . . . . .                            | 17-1  |
| Basic configuration tasks . . . . .                                              | 17-1  |
| Optional configuration tasks . . . . .                                           | 17-2  |
| Configuring a pool . . . . .                                                     | 17-5  |
| Configuring virtual servers . . . . .                                            | 17-6  |
| Using standard or wildcard virtual servers . . . . .                             | 17-6  |
| Using additional features with virtual servers . . . . .                         | 17-7  |
| Defining standard virtual servers . . . . .                                      | 17-7  |
| Defining wildcard virtual servers . . . . .                                      | 17-8  |
| Allowing access to ports and services . . . . .                                  | 17-12 |
| Configuring the timer settings . . . . .                                         | 17-13 |
| Setting the node ping timer . . . . .                                            | 17-14 |
| Setting the timer for reaping idle connections . . . . .                         | 17-15 |
| Setting the service check timer . . . . .                                        | 17-17 |
| Service checking for wildcard servers and ports . . . . .                        | 17-18 |
| Changing the global load balancing mode . . . . .                                | 17-19 |
| Using Ratio mode . . . . .                                                       | 17-20 |
| Configuring NATs and IP forwarding for nodes . . . . .                           | 17-21 |
| Defining a standard network address translation (NAT) . . . . .                  | 17-23 |
| Defining a secure network address translation (SNAT) . . . . .                   | 17-24 |
| Setting up IP forwarding . . . . .                                               | 17-27 |
| Configuring Extended Content Verification service checking . . . . .             | 17-28 |
| ECV service check properties . . . . .                                           | 17-29 |
| Writing send and receive strings for ECV service checks . . . . .                | 17-30 |
| Setting up ECV service check using the Configuration utility . . . . .           | 17-32 |
| Manually configuring and testing the /etc/bigd.conf file . . . . .               | 17-33 |
| Configuring persistence for e-commerce and other dynamic content sites . . . . . | 17-35 |
| Setting up SSL persistence . . . . .                                             | 17-37 |

|                                                        |       |
|--------------------------------------------------------|-------|
| Setting up simple persistence .....                    | 17-38 |
| Configuring and synchronizing redundant systems .....  | 17-40 |
| Preparing to use the synchronization command .....     | 17-40 |
| Synchronizing configurations between controllers ..... | 17-41 |
| Configuring fail-safe settings .....                   | 17-43 |

## Chapter 18

### Monitoring and Administration

|                                                                                         |       |
|-----------------------------------------------------------------------------------------|-------|
| Monitoring and administration utilities provided on the BIG-IP<br>Controller .....      | 18-1  |
| Using the bigpipe command utility as a monitoring tool .....                            | 18-2  |
| Monitoring the BIG-IP Controller .....                                                  | 18-2  |
| Monitoring virtual servers, virtual addresses, and<br>services .....                    | 18-10 |
| Monitoring nodes and node addresses .....                                               | 18-12 |
| Monitoring NATs .....                                                                   | 18-13 |
| Monitoring SNATs .....                                                                  | 18-13 |
| Working with the BIG/stat utility .....                                                 | 18-14 |
| Working with the BIG/top utility .....                                                  | 18-17 |
| Working with the Syslog utility .....                                                   | 18-19 |
| Removing and returning items to service .....                                           | 18-20 |
| Removing the BIG-IP Controller from service .....                                       | 18-21 |
| Removing individual virtual servers, virtual addresses, and<br>ports from service ..... | 18-22 |
| Removing individual nodes and node addresses from<br>service .....                      | 18-23 |
| Viewing the currently defined virtual servers and nodes ....                            | 18-23 |
| Viewing system statistics and log files .....                                           | 18-24 |
| Viewing system statistics .....                                                         | 18-24 |
| Viewing log files .....                                                                 | 18-25 |
| Printing the connection table .....                                                     | 18-25 |
| Changing passwords for the BIG-IP Controller .....                                      | 18-25 |
| Changing passwords and adding new user IDs for the<br>BIG-IP web server .....           | 18-26 |
| Working with the BIG/db database .....                                                  | 18-27 |
| Using bigdba .....                                                                      | 18-27 |

**Chapter 19**

**Configuring SNMP**

Working with SNMP on the BIG-IP Controller .....19-1

Configuring SNMP on the BIG-IP Controller .....19-1

    Downloading the MIBs .....19-2

Understanding configuration file requirements .....19-3

    /etc/hosts.deny .....19-3

    /etc/hosts.allow .....19-3

    /etc/snmpd.conf .....19-4

    /etc/rc.local .....19-7

    /etc/snmptrap.conf .....19-8

    Syslog .....19-9

**Glossary**

**Index**



---

# Introduction

---



## 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

The BIG-IP HA Controller supports the following types of solutions. The subsequent chapters in this guide each outline a solution and provide configuration instructions for the solution.

### Basic web site and e-commerce configuration

This chapter covers the most common application of the BIG-IP Controller. In this configuration, traffic is distributed across an array of web servers that host standard web traffic, including e-commerce traffic.

### A simple intranet configuration

This chapter describes how to set up the BIG-IP Controller to load balance a basic corporate intranet.

### Load balancing virtual private networks (VPNs)

This chapter describes two scenarios for load balancing virtual private networks (VPNs) with the BIG-IP Controller.

### ISP load balancing

This chapter describes how you can use the BIG-IP Controller to create separate connections to the internet. This functionality is useful if you need to add an additional connection to the internet.

### Topologies for single IP networks

This chapter describes two load balancing scenarios on a single IP network.

### nPath routing

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

### Per-connection routing

This chapter describes how to configure the BIG-IP Controller in situations where it accepts connections for virtual servers from more than one router. The per-connection routing feature allows you to send the return data back through the same device from which the connection originated. You can use this option to spread the load among outbound routers, or to ensure that connections go through the same device if that device is connection-oriented, such as a proxy, cache, firewall, or VPN router.

### SNAT and virtual servers combined

In some cases you may want to configure outbound transparent device load balancing and SNAT source translations. In the configuration described in this chapter, the BIG-IP Controller changes the source address of the clients to the external SNAT address. In this way, the actual IP addresses are not exposed to the internet. At the same time the BIG-IP Controller can load balance the same connection across multiple nodes. Therefore, both SNAT translation and virtual server load balancing can operate on the same connection in this configuration.

## 802.1q VLAN trunk mode

This chapter describes how to set up the BIG-IP Controller to use IEEE 802.1q VLAN Trunks. VLAN tags are not supported on the external interfaces. You can define a single VLAN tag for each IP address defined for each BIG-IP Controller internal interface. This includes node network addresses, administrative addresses, shared administrative aliases, and additional aliases.

## 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 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 more 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 <b>&lt;your name&gt;</b> , 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 conventions used in this manual*



## 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 a PDF version of this administrator guide. Note that some BIG-IP Controller upgrades replace the online Getting Started Guide, Administrator Guide, and Reference Guide with updated versions.
- 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 by sending an e-mail to the F5 Help Desk at [askf5@f5.com](mailto:askf5@f5.com).

## 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

This version of the BIG-IP Controller is available as the BIG-IP e-Commerce Controller. 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 Load Balancer

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 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.*

I

---

## Basic Web Site and e-commerce configuration

---

- Basic Web site and e-commerce configuration



## Basic web site and e-commerce configuration

The most common application of the BIG-IP Controller is to distribute traffic across an array of web servers that host standard web traffic, including e-commerce traffic.

The following sections provide you with a basic configuration example that can help you plan your installation. These examples can also help you understand how you can use some of the most popular BIG-IP Controller features to resolve specific issues or to enhance network performance in general.

### A basic web site and e-commerce configuration

First, we start with a basic configuration where a BIG-IP Controller load balances two sites: **www.MySite.com** and **store.MySite.com**. The **www.MySite.com** site provides standard web content, and the **store.MySite.com** site is the e-commerce site that sells items to **www.MySite.com** customers. In this scenario, the BIG-IP Controller provides simple load balancing for both sites.

## Setting up the topology

To set up load balancing for these sites, you need to create two pools that are referenced by two virtual servers, one for each site. Even though the sites are related and they may even share the same IP address, each requires its own virtual server because it uses a different port to support its particular protocol: port 80 for the HTTP traffic going to **www.MySite.com**, and port 443 for the SSL traffic going to **store.MySite.com**.

Figure 1.1 shows the topology for the sample configuration. Each site uses two of the three web servers to host its content. Both sites happen to share Server 2.

### ◆ Note

*Note that in this example, as in all examples in this guide, we use only non-routable IP addresses. In a real topology, the virtual server IP addresses would have to be routable on the Internet.*

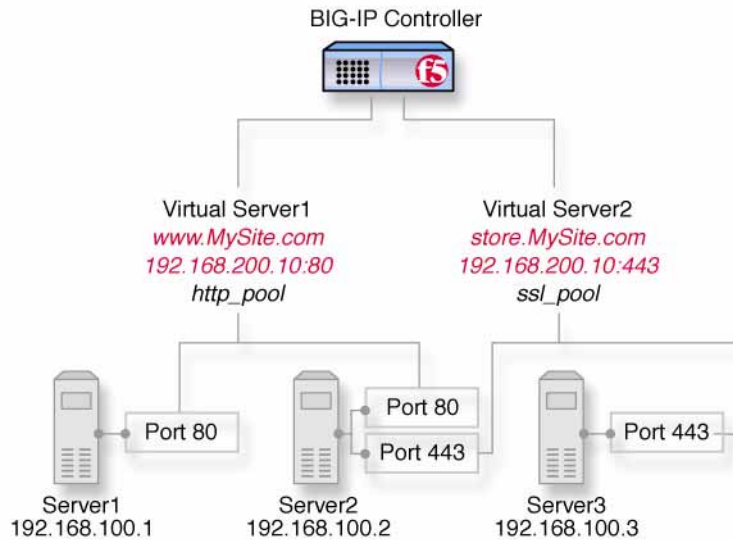


**Figure 1.1** A basic configuration

The **virtual servers** that you define always include three basic elements:

#### ❖ Virtual IP address

This is the IP address that is registered with DNS and associated with your site's domain name. In our example, both **www.MySite.com** and **store.MySite.com** use the same IP address: **192.168.200.10**. Both domain names would presumably have to be registered with DNS to resolve to that IP address.



❖ **Port**

The port that hosts the specific service supported by your site. In our example, we have two different sites that support two different ports: port 80 and port 443.

❖ **Pools of servers that host your site**

Pools contain the list of physical servers that actually host your site connections. For a given pool of servers, you list each IP address and port number pair, referred to as a **member**. Even though our example includes only three physical servers, it actually has four members:

- Virtual Server 1 references the **http\_pool** that contains two members: **192.168.100.1:80** and **192.168.100.2:80**.
- Virtual Server 2 references the **ssl\_pool** that contains two members: **192.168.100.2:443** and **192.168.100.3:443**.

The BIG-IP Controller distributes connections among the three servers according to a user-specified load balancing mode. The most common mode is Round Robin, which simply distributes each new connection to the next server in line, eventually distributing the connections equally among all the servers.

## Using additional features

In this type of configuration, you might want to take advantage of the following BIG-IP Controller features:

❖ **Extended Content Verification**

Verifies that the web servers are not only up and running, but also able to send valid content to clients. For example, you could use Extended Content Verification to make sure that **www.MySite.com** returns its home page rather than an HTTP 404 error.

❖ **Persistence**

Allows returning e-commerce customers to bypass load balancing and connect to the back-end server to which they originally connected that may contain user-specific information. In our example, **store.MySite.com** provides the ability for users to fill a shopping cart, disconnect from the site, and then return

up to 24 hours later to purchase the items. When the user returns to purchase the items, the user may need to go to the same back-end server, depending on how the e-commerce site is set up.

❖ **Network Address Translation**

Allows you to make direct administrative connections to the web servers through the BIG-IP Controller. If your administrative workstation is on the network connected to the BIG-IP Controller's external interface, and administrative workstations frequently are, this feature is essential.

❖ **Secure Network Address Translation (SNAT)**

Allows you to map internally routable IP addresses to an externally routable IP address. SNATs do not allow incoming connections.

# 2

---

## A Simple Intranet Configuration

---

- A simple intranet configuration



## A simple intranet configuration

After you have completed the most basic configuration options for the BIG-IP Controller (see ***Determining which configuration tasks to do*** on page 17-1), you may want to review this sample configuration that uses the BIG-IP Controller for traffic management. The following example describes a simple intranet load balancing configuration.

The following sections provide you with a basic intranet configuration that can help you plan your installation. This example can also help you understand how people use some of the most popular BIG-IP Controller features to resolve specific issues or to enhance network performance in general.

This example is a configuration that might be found in a large corporate intranet. In this scenario, the BIG-IP Controller performs load balancing for two different types of connection requests:

### ❖ **Connections to the company's intranet web site**

The load balancing for the company's intranet web site is similar to basic Internet web site load balancing. The BIG-IP Controller simply load balances the two web servers that host the company intranet web site.

### ❖ **Connections to hosts on the Internet**

In this example, the BIG-IP Controller provides load balancing for connections bound for the Internet. However, the example shows a somewhat sophisticated setup where the BIG-IP Controller actually intercepts HTTP traffic and directs it to a special cache server. Only clients using protocols other than HTTP, such as FTP or SMTP email, get load balanced to one of the two firewalls that lead to the Internet. This greatly reduces the number of concurrent connections that the firewalls have to maintain. Clients looking to retrieve web content get the content from the cache server itself, instead of the actual web site host. If the cache server does not have the content that the client is looking for, the cache server retrieves the content from the real web site on behalf of the client and then forwards it to the client.

## Setting up the topology

To set up load balancing for this intranet example, you need to create three pools that are referenced by three virtual servers: one that handles load balancing for the internal corporate web site, one that directs outbound HTTP traffic to the cache server, and one that handles load balancing for the firewalls.

Figure 2.1 shows the topology for the sample configuration. A standard virtual server handles the load balancing for the corporate intranet web site, **Corporate.main.net**. Wildcard Virtual Server 1 takes all of the outbound HTTP traffic and directs it to the cache server. Wildcard Virtual Server 2 handles all of the remaining traffic that actually has to go out to the Internet.

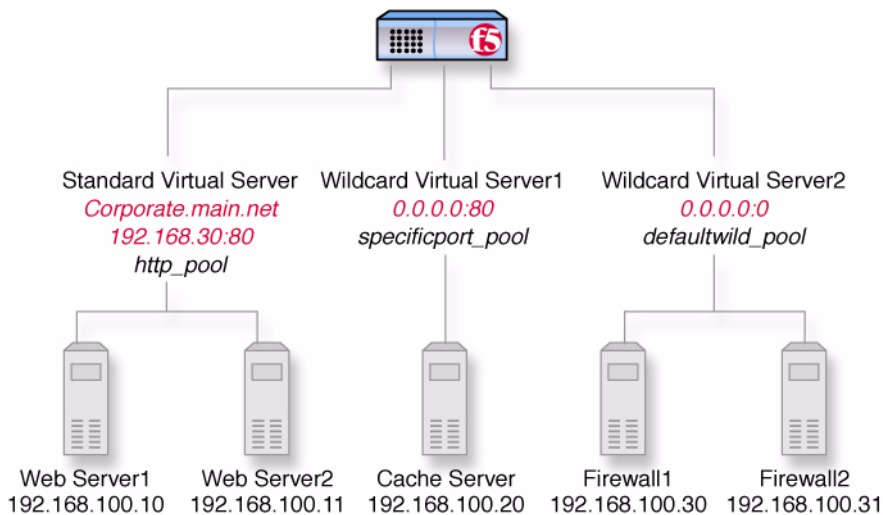


**Figure 2.1** A basic intranet configuration

The **wildcard virtual servers** are a special type of virtual server, which accept traffic going to IP addresses unknown to the BIG-IP Controller, as all outside Internet addresses would be. When the BIG-IP Controller receives a connection request, it immediately tries to match the requested IP address to one of its virtual server IP addresses. If it cannot find a match among the standard virtual

servers that it manages, it then looks for a wildcard virtual server. Wildcard virtual servers provide the default IP address of **0.0.0.0** that the BIG-IP Controller can use as a sort of catch-all IP address match.

This example contains three types of virtual servers:

❖ **Standard virtual server**

The standard virtual server references the **http\_pool** that contains two members: **192.168.100.10:80** and **192.168.100.11:80**.

❖ **Port-specific wildcard virtual servers**

A port-specific wildcard virtual server uses the default IP address, but it has a specific port number, and it only handles traffic associated with that port number. In the preceding example, the port-specific wildcard virtual server captures all outbound traffic that uses port 80 and directs it to the cache server. The port-specific wildcard virtual server references the **specificport\_pool** that contains one member: **192.168.100.20:80**.

❖ **Default wildcard virtual servers**

A default wildcard virtual server is one that uses only port **0**. Port **0**, like the **0.0.0.0** IP address, is a catch-all match for outgoing traffic that does not match any standard virtual server or any port-specific wildcard virtual server. Default wildcard virtual servers typically handle traffic only for firewalls or routers. In the preceding example, the default wildcard virtual server load balances the intranet's firewalls that connect to the Internet. The default wildcard virtual server references the **defaultwild\_pool** that contains two members: **192.168.100.30:80** and **192.168.100.31:80**.

## Using additional features

In this type of configuration, you might want to take advantage additional BIG-IP Controller features that are described both in this guide and in the ***BIG-IP Controller Reference Guide***. These features include:

### ❖ **State mirroring**

This feature is available only for redundant BIG-IP Controller systems, and it greatly enhances the reliability of your network. A redundant system runs two BIG-IP Controllers at the same time. One unit actively handles all connection requests, and the other unit acts as a standby that immediately takes over if the active unit fails and reboots. The state mirroring feature allows the standby unit to maintain all of the current connection and persistence information. If the active unit fails and the standby unit takes over, all connections continue, virtually uninterrupted. This is especially useful for long-lived connections, such as FTP connections which would otherwise have to re-establish an entire transfer session.

### ❖ **Destination address affinity**

Allows the BIG-IP Controller to cache content on specified cache servers by sending all requests for the same server to the same node. This avoids caching the same content on multiple cache servers. Because the above example includes only one cache server, you would not actually implement this feature in that example. However, the destination address affinity feature is very useful for users who work with multiple cache servers in a similar intranet scenario. Caching specific information on the same cache server saves disk space on your cache servers.

### ❖ **IP address filtering**

Allows you to deny connections going to or coming from specific IP addresses. This feature is useful if you are experiencing denial-of-service attacks from hostile sources. You can set up an IP filter to ignore traffic coming in from the hostile IP address.



# 3

---

## Configuring an SSL Accelerator

---

- Introducing the SSL Accelerator
- Hardware acceleration options
- Configuring the SSL Accelerator
- Optional SSL Accelerator configuration
- Using the SSL Accelerator cell configuration
- Introducing the SSL Accelerator half sandwich configuration



## Introducing the SSL Accelerator

The SSL Accelerator feature allows the BIG-IP Controller to accept HTTPS connections (HTTP over SSL), connect to a web server, retrieve the page, and then send the page to the client.

A key component of the SSL Accelerator feature is that the BIG-IP Controller can retrieve the web page using an unencrypted HTTP request to the content server. With the SSL Accelerator feature, you can configure an SSL gateway on the BIG-IP Controller that decrypts HTTP requests that are encrypted with SSL. Decrypting the request offloads SSL processing from the servers to the BIG-IP Controller. This also allows the BIG-IP Controller to use the header of the HTTP request to intelligently control how the request is handled.

When the SSL gateway on the BIG-IP Controller connects to the content server, it uses the original client's IP address and port as its source address and port, so that it appears to be the client (for logging purposes).

This chapter describes the following features of the BIG-IP Controller SSL Accelerator:

- ❖ Hardware accelerator options
- ❖ Configuring an SSL Accelerator
- ❖ Enabling and disabling an SSL Accelerator
- ❖ Viewing the configuration of an SSL Accelerator
- ❖ Optional SSL Accelerator configuration
- ❖ Using an SSL Accelerator cell configuration
- ❖ Introducing the SSL Accelerator half sandwich

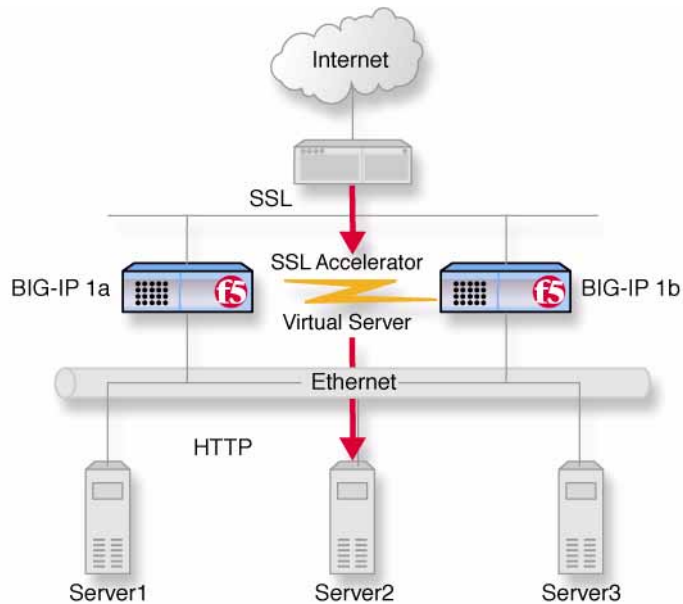


**Figure 3.1** An incoming SSL connection received by an SSL Accelerator configured on a redundant BIG-IP Controller system

## Hardware acceleration options

Because the SSL Accelerator feature is computationally intensive, you should only use this feature on a BIG-IP Controller with an encryption accelerator installed.

The BIG-IP Controller detects the accelerator card at boot up and starts the server for the card.

### ◆ Note

*Hardware acceleration greatly increases the number of SSL transactions the BIG-IP Controller can handle.*

## Configuring the SSL Accelerator

There are several steps required to set up the SSL Accelerator on the BIG-IP Controller. These steps include:

- ❖ Generating a key and obtaining a certificate
- ❖ Configuring the BIG-IP Controller with the certificate and key
- ❖ Creating an HTTP virtual server
- ❖ Creating the gateway for the SSL Accelerator

An additional configuration option you can use with the SSL Accelerator is a last hop pool. You can use this option if the SSL Accelerator accepts connections from multiple firewalls or routers. For additional information about this option, see ***Optional SSL Accelerator configuration*** on page 3-18.

### Generating a key and obtaining a certificate

In order to use the SSL Accelerator feature you must obtain a valid x509 certificate from an authorized certification authority (CA). The following list contains some companies that are certification authorities:

- ❖ Verisign (<http://www.verisign.com>)
- ❖ Digital Signature Trust Company (<http://secure.digisigtrust.com>)
- ❖ GlobalSign (<http://www.globalsign.com>)
- ❖ GTE Cybertrust (<http://www.cybertrust.gte.com>)
- ❖ Entrust (<http://www.entrust.net>)

You can generate a key, a temporary certificate, and a certificate request form with the Configuration utility or from the command line.

We recommend using the Configuration utility for this process. The certification process is generally handled through a web page. Parts of the process require you to cut and paste information from a browser window in the Configuration utility to another browser window on the web site of the CA.

### Additional information about keys and certificates

You must have a separate certificate for each domain name on each redundant pair of BIG-IP Controllers, regardless of how many non-SSL web servers are load balanced by the BIG-IP Controller.

If you are already running an SSL server you can use your existing keys to generate temporary certificates and request files. However, you must obtain new certificates if the ones you have are not for the following web server types:

- ❖ Apache + OpenSSL
- ❖ Stronghold

### Generating a key and obtaining a certificate in the Configuration utility

To obtain a valid certificate, you must have a private key. If you do not have a key, you can use the Configuration utility on the BIG-IP Controller to generate a key and a temporary certificate. You can also use the Configuration utility to create a request file you can submit to a certification authority (CA). You must complete three tasks in the Configuration utility to create a key and generate a certificate request.

- ❖ Generate a certificate request
- ❖ Submit the certificate request to a CA and generate a temporary certificate
- ❖ Install the SSL certificate from the CA

Each of these tasks is described in detail in the following section.

**Creating a new certificate request in the Configuration utility**

1. In the navigation pane, click **Proxies**.  
The Proxies screen opens.
2. On the toolbar, click **Create SSL Certificate Request**.  
The New SSL Certificate Request screen opens.
3. In the Key Information section, select a key length and key file name.

**a) Key Length**

Select the key length you want to use for the key. You can choose either **512** or **1024** bytes.

**b) Keyfile Name**

Type in the name of the key file. This should be the fully qualified domain name of the server for which you want to request a certificate. You must add the **.key** file extension to the name.

4. In the Certificate Information section, type the information specific to your company. This information includes:

- **Country**

Type the two letter ISO code for your country, or select it from the list. For example, the two-letter code for the United States is US.

- **State or Province**

Type the full name of your state or province, or select it from the list. You must enter a state or province.

- **Locality**

Type the city or town name.

- **Organization**

Type the name of your organization.

- **Organizational Unit**

Type the division name or organizational unit.

- **Domain Name**

Type the name of the domain upon which the server is installed.

- **Email Address**  
Type the email address of a person who can be contacted about this certificate.
  - **Challenge Password**  
Type the password you want to use as the challenge password for this certificate. The CA uses the challenge password to verify any changes you make to the certificate at a later date.
  - **Retype Password**  
Retype the password you entered for the challenge password.
5. Click the **Generate Certificate Request** button.  
After a short pause, the SSL Certificate Request screen opens.
  6. In the SSL Certificate Request screen, you can start the process of obtaining a certificate from a CA and you can generate and install a temporary certificate:
    - **Begin the process for obtaining a certificate from CA**  
Click on the URL of a CA to begin the process of obtaining a certificate for the server. After you select a CA, follow the directions on their web site to submit the certificate request. After your certificate request is approved, and you receive a certificate back from the CA, see *Installing certificates from the CA in the Configuration utility* on page 3-10, for information about installing it on the BIG-IP Controller.
    - **Generate and install a temporary certificate**  
Click the **Generate Self-Signed Certificate** button to create a self-signed certificate for the server. We recommend that you use the temporary certificate for testing only. You should only take your site live after you receive a properly-signed certificate from a certification authority. When you click this button, a temporary certificate is created and installed on the BIG-IP Controller. This certificate is valid for 30 days.



This temporary certificate allows you to set up an SSL gateway for the SSL Accelerator while you wait for a CA to return a permanent certificate.

## Generating a key and obtaining a certificate from the command line

To obtain a valid certificate, you must have a private key. If you do not have a key, you can use the **genconf** and **genkey** utilities on the BIG-IP Controller to generate a key and a temporary certificate. The **genkey** and **gencert** utilities automatically generate a request file you can submit to a certification authority (CA). If you have a key, you can use the **gencert** utility to generate a temporary certificate and request file. These utilities are described in the following list:

### ❖ **genconf**

This utility creates a key configuration file that contains specific information about your organization. The **genkey** utility uses this information to generate a certificate.

### ❖ **genkey**

After you run the **genconf** utility, run this utility to generate a temporary 30 day certificate for testing the SSL Accelerator on the BIG-IP Controller. This utility also creates a request file that you can submit to a certification authority (CA) to obtain a certificate.

### ❖ **gencert**

If you already have a key, run this utility to generate a temporary certificate and request file for the SSL Accelerator.

## To generate a key configuration file using the **genconf** utility

If you do not have a key, you can generate a key and certificate with the **genconf** and **genkey** utilities. First, run the **genconf** utility from the root (/) with the following commands:

```
cd /
```

```
/var/asr/gateway/bin/genconf
```

The utility prompts you for information about the organization for which you are requesting certification. This information includes:

- ❖ The fully qualified domain name (FQDN) of the server
- ❖ The two-letter ISO code for your country
- ❖ The full name of your state or province
- ❖ The city or town name
- ❖ The name of your organization
- ❖ The division name or organizational unit

For example, Figure 3.2 contains entries for the server **my.server.net**:

```
Common Name (full qualified domain name): my.server.net
Country Name (ISO 2 letter code): US
State or Province Name (full name): WASHINGTON
Locality Name (city, town, etc.): SEATTLE
Organization Name (company): MY COMPANY
Organizational Unit Name (division): WEB UNIT
```

*Figure 3.2 Example entries for the **genconf** utility*

After you run the **genconf** utility, you can run the **genkey** utility to create a temporary certificate and a request file.

### **To generate a key using the **genkey** utility**

After you run the **genconf** utility, you can generate a key with the **genkey** utility. Type the following command from the root (/) to run the **genkey** utility:

```
cd /
/var/asr/gateway/bin/genkey <server_name>
```

For the **<server\_name>**, type the FQDN of the server to which the certificate applies. After the utility starts, it prompts you to verify the information created by the **genconf** utility. After you run this utility, a certification request form is created in the following directory:

```
/var/asr/gateway/requests/<fqdn>.req
```

The **<fqdn>** is the fully qualified domain name of the server. Please contact your CA and follow their instructions for submitting this request form.

In addition to creating a request form you can submit to a certification authority, this utility also generates a temporary certificate. The temporary certificate is located in:

**/var/asr/gateway/certs/<fqdn>.cert**

The **<fqdn>** is the fully qualified domain name of the server.

Note that you must copy the key and certificate to the other controller in a redundant system.

This temporary certificate is good for thirty days, after which time you should have a valid certificate from your CA. If you do not have a certificate within 30 days, you can re-run this program.

---

 **WARNING**

*Be sure to keep your previous key if you are still undergoing certification. The certificate you receive is valid only with the key that originally generated the request.*

### **To generate a certificate with an existing key with the gencert utility**

To generate a temporary certificate and request file to submit to the certification authority with the **gencert** utility, you must first copy an existing key for a server into the following directory on the BIG-IP Controller:

**/var/asr/gateway/private/**

After you copy the key into this directory, type the following command at the command line:

**cd /**

**/var/asr/gateway/bin/gencert <server\_name>**

For the **<server\_name>**, type the FQDN of the server to which the certificate applies. After the utility starts, it prompts you for various information. After you run this utility, a certification request form is created in the following directory:

**/var/asr/gateway/requests/<fqdn>.req**

The **<fqdn>** is the fully qualified domain name of the server. Please contact your certification authority (CA) and follow their instructions for submitting this request form.

## Installing certificates from the certification authority (CA)

After you obtain a valid x509 certificate from a certification authority (CA) for the SSL Accelerator, you must copy it onto each BIG-IP Controller in the redundant configuration. You can configure the accelerator with certificates from the Configuration utility or from the command line.

### Installing certificates from the CA in the Configuration utility

1. In the navigation pane, click **Proxies**.  
The Proxies screen opens.
2. On the toolbar, click **Install SSL Certificate**.  
The Install SSL Certificate screen opens.
3. In the **Certfile Name** box, type the fully qualified domain name of the server with the file extension **.cert**. Note that if you generated a temporary certificate when you submitted a request to the CA, select the name of the certificate from the drop down list. This allows you to overwrite the temporary certificate with the certificate from the CA.
4. Paste the text of the certificate into the Install SSL Certificate window. Make sure you include the **Begin Certificate** line and the **End Certificate** line. For an example of a certificate, see Figure 3.3.
5. Click the **Write Certificate File** button.

```

-----BEGIN CERTIFICATE-----
MIIB1DCCAX4CAQAwDQYJKoZIhvcNAQEEBQAwTElMAkGA1UEBhMCVVMxCzAJBgNV
BAGTAldBMRAwDgYDVQQHEwdTZWF0dGxLMRQwEgYDVQQKEwtGNSB0ZXR3b3JrczEc
MBoGA1UECxMTUHJvZHVjdCBEZXZlbG9wbWVudDETMBEGA1UEAxMKc2VydMvyLm5l
dDAeFw0wMDA0MTkxNjMxNTlaFw0wMDA1MTkxNjMxNTlaMHUxCzAJBgNVBAYTALVT
MQswCQYDVQQIEwJXQTEQMA4GA1UEBxMHU2VhdHRsZTEUMBIGA1UEChMLRjUgTmV0
d29ya3MxHDAaBgNVBASTE1Byb2R1Y3QgRGV2ZWxvcG1lbnQxEzARBgNVBAMTCnNl
cnZlci5uZXQwXDANBgkqhkiG9w0BAQEFAANLADBIASfCFXq3Jt+FevxUqBZ9T
Z7nHx9uaF5x9V5xMZYgekjc+LrF/yazhmq4PCxrhs3gvJmgtSh50YJrhJgfs2bE
gwIDAQABMA0GCSqGSIb3DQEBAUAA0EAD1q6+u/aMaM2qdo7EjWx14TYQQGomYoq
eydlzb/3F0iJAynDXnGnSt+CVvyRXtvmG7V8xJamzkyEpZd4iLacLQ==
-----END CERTIFICATE-----

```

**Figure 3.3** An example of a certificate

After the certificate is installed, you can continue with the next step to creating an SSL gateway for the server.

### Installing certificates from the CA on the command line

Copy the certificate into the following directory on each BIG-IP Controller in a redundant system:

**/var/asr/gateway/certs/**

#### ◆ Note

*The certificate you receive from the certification authority (CA) should overwrite the temporary certificate generated by **genkey** or **gencert**.*

If you used the **genkey** or **gencert** utilities to generate the request file, a copy of the corresponding key should already be in the following directory on the BIG-IP Controller:

`/var/asr/gateway/private/`

 **WARNING**

*The keys and certificates must be in place on both controllers in a redundant system before you configure the SSL Accelerator. You must do this manually; the configuration synchronization utilities do not perform this function.*

## Create an HTTP virtual server

After you configure the BIG-IP Controller with the certificates and keys, the next step is to create a virtual server that references a pool containing the HTTP servers for which the SSL Accelerator handles connections. Note that before you create the HTTP virtual server, you can use a pool or rule that references your HTTP servers. The example in this section describes how to create a virtual server that references a pool that contains the HTTP virtual servers. For more information about creating a pool, see *Defining a pool for the servers* on page 4-8.

### Creating an HTTP virtual server in the Configuration utility

1. In the navigation pane, click **Virtual Servers**.  
The Virtual Servers screen opens.
2. On the tool bar, click **Add Virtual Server**.  
The Add Virtual Server screen opens.
3. Add the attributes you want for the virtual server such as **Address**, **Port**, **Unit ID** (active-active only), and **Interface**.
4. In the Resources section, click **Pool**.
5. In the Pool list, select the pool of HTTP servers you want to use with the virtual server.
6. Click **Apply**.

### Creating an HTTP virtual server from the command line

Note that before you create the HTTP virtual server, you must configure a pool that contains your HTTP servers. For more information about creating pools, see *Defining a pool for the servers*, on page 4-8. After you have defined a pool that contains the HTTP servers, use the following syntax to create a virtual server that references the pool:

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

For example, if you want to create a virtual server **20.1.1.1:80**, that references a pool of HTTP servers named **http\_pool**, you would type the following command:

```
bigpipe vip 20.1.1.1:80 use pool http_pool
```

After you create the virtual server that references the pool of HTTP servers, you can create an SSL gateway. The following section describes how to create an SSL gateway.

## Create an SSL gateway

After you create the HTTP virtual server for which the SSL Accelerator handles connections, the next step is to create an SSL gateway.

### Creating an SSL gateway in the Configuration utility

1. In the navigation pane, click **Proxies**.  
The Proxies screen opens.
2. On the toolbar, click **Add Proxy**.  
The Add Proxy screen opens.
3. In the **Proxy Address** box, type the IP address for the SSL gateway.
4. In the **Proxy Netmask** box, type the netmask you want to use for the SSL gateway. If you leave this setting blank, the BIG-IP Controller creates a default based on the network class of the IP address on the external (destination processing) interface. Type a user-defined netmask only if necessary.

5. In the **Proxy Broadcast** box, type the broadcast address you want to use for this SSL gateway. The BIG-IP Controller automatically generates a broadcast address if you do not type one. Type a user-defined broadcast address only if necessary.
6. In the **Proxy Port** box, type the port number that the proxy server uses, or select a service from the list box. Note that if you select a service, the Configuration utility uses the default port number associated with that service.
7. In the **Unit ID** list, select the unit number you want to assign this SSL gateway. Connections served by this SSL gateway are managed by the controller assigned to this unit ID. This only applies if this controller is running in active mode.
8. For **Interface**, select the destination processing interface on which you want to create the SSL gateway. Select **default** to allow the Configuration utility to select the interface based on the network address of the SSL gateway. If you choose **None**, the BIG-IP Controller does not create an alias and generates no ARPs for the virtual IP address.
9. In the **Destination Address** box, type the IP address or host name of the node or virtual server to which the SSL gateway maps. This should be the virtual server you created that references the pool of HTTP servers on your network that will respond to requests handled by the SSL gateway.
10. In the **Destination Port** box, type a port name or number, such as port **80** or **http**, or select the service name from the drop-down list.
11. In the **Destination Target** list, select the type of target to which the proxy sends connections:
  - **External Node**  
Select **External Node** if the SSL gateway handles connections for an IP address of a server that resides on the network instead of a virtual server.



- **Local Virtual Server**  
Select **Local Virtual Server** if the SSL accelerator gateway handles connections for a virtual server located on the BIG-IP Controller.
12. In the **SSL Certificate** box, type the name of the SSL certificate you installed on the BIG-IP Controller. You can select the certificate you want to use from the drop-down list.
  13. In the **SSL Key** box, type the name of the SSL key for the certificate you installed on the BIG-IP Controller. You can select the key from the drop-down list. It is important that you select the key used to generate the certificate you selected in the **SSL Certificate** box.
  14. In the **Last Hop Pool** list, select the last hop pool that contains other network devices from which the BIG-IP Controller receives connections. This feature is optional. You need to use this feature only if the SSL gateway is accepting connections from multiple network devices.
  15. Click **Apply**.

### Creating an SSL gateway from the command line

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

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

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

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

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

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

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

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

**Figure 3.4** An example SSL gateway configuration

## Enabling, disabling, or deleting an SSL gateway

After you have created an SSL gateway, you can enable it, disable it, or delete it using the Configuration utility or from the command line.

### Enabling or disabling an SSL gateway in the Configuration utility

1. In the navigation pane, click **Proxies**.  
The Proxies screen opens.
2. In the Proxies list, select the SSL gateway you want to enable or disable.  
The Proxy Properties screen opens.
3. In the Proxy Properties screen, clear the **Enable** box to disable the Proxy, or check the **Enable** box to enable the SSL gateway.
4. Click **Apply**.

### Deleting an SSL gateway in the Configuration utility

1. In the navigation pane, click **Proxies**.  
The Proxies screen opens.
2. In the Proxies list, select the SSL gateway you want to delete.  
The Proxy Properties screen opens.
3. On the toolbar, click **Delete**.

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

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

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

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

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

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

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

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

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

### Displaying the configuration for an SSL gateway from the command line

You can view the configuration information for an SSL gateway from the command line with the **show** keyword.

### Displaying configuration information for an SSL accelerator gateway from the command line

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

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

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

**bigpipe proxy 209.100.19.22:80 show**

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

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

*Figure 3.5 Output from the **bigpipe proxy show** command*

## Optional SSL Accelerator configuration

Depending on your network configuration, the SSL Accelerator may require additional configuration. For example, in cases where the BIG-IP Controller receives connections from several devices, such as routers or firewalls, you can configure a last hop pool for the SSL Accelerator. The last hop pool must contain the IP addresses of the routers or firewalls from which the BIG-IP Controller receives connections.

## Create a last hop pool that includes additional network devices

If the SSL gateway accepts connections from multiple firewalls or routers, you can configure a last hop pool for the SSL gateway. This last hop pool can contain any other devices, such as firewalls or routers, through which connections are received by the BIG-IP Controller.

### Creating a last hop pool with additional network devices in the Configuration utility

1. In the navigation pane, click **Pools**.  
The Pools screen opens.
2. On the toolbar, click **Add Pool**.  
The Add Pool screen opens.
3. In the **Pool Name** box, type the name you want to use for the pool.
4. From the load balancing method list, select the load balancing mode you want to use for this pool.
5. Use the resources options to add the devices from which the BIG-IP Controller receives connections. To add devices to the pool, type the IP address in the **Node Address** box, type the port number in the **Port** box, and then type in the ratio or priority for this node. Finally, to add the node to the list, click the add ( >> ) button.
  - **Node Address**  
Type the IP addresses of routers or other devices from which the BIG-IP Controller receives connections.
  - **Port**  
Type the port number of the port you want to use for this node in the pool.
  - **Ratio**  
Type a number to assign a ratio to this node within the pool. For example, if you are using the ratio load balancing method and you type a **1** in this box, the node will receive fewer connections from the load-balancing pool than a node marked **2**.

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

6. Click **Apply**.

### **Creating a last hop pool with additional network devices from the command line**

Use the following syntax to configure a last hop pool for the SSL gateway that contains the additional network devices:

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

For example, you might use the following command to create a last hop pool that contains three routers:

```
bigpipe pool sslasthop_pool {lb_mode ratio_member member  
  11.12.1.100:80 ratio 1 priority 1 member 11.12.1.101:80 ratio 1  
  priority 1 11.12.1.102:80 ratio 1 priority 1}
```

After you create the last hop pool, you must modify the SSL gateway so that it references the last hop pool. The next section describes how to do this with the Configuration utility or from the command line.

## **Modify the SSL gateway so that it references the last hop pool**

After you create the last hop pool that contains other devices, such as firewalls or routers, you can reference it from the SSL gateway using either the Configuration utility or the command line.

### Adding a last hop pool to an SSL gateway in the Configuration utility

1. In the navigation pane, click **Proxies**.  
The Proxies screen opens.
2. In the Proxies list, select the SSL gateway to which you want to assign the last hop pool.  
The Proxy Properties screen opens.
3. In the Last Hop Pool list, select the last hop pool that contains additional network devices.
4. Click **Apply**.

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

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

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

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

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

## Introducing the SSL accelerator cell configuration

This chapter explains how to set up a scalable SSL accelerator configuration. This configuration is useful for any enterprise that handles a large amount of encrypted traffic.

With this configuration, you can increase the scale of the network by adding a new **cell**. A cell consists of an SSL accelerator and one or more nodes for which it proxies SSL connections.

Figure 3.6 shows a configuration of an SSL accelerator cell. The SSL accelerator cell described in this chapter includes BIG-IP Controllers 1a and 1b, the SSL accelerator **accelerator1**, and **Node1** and **Node2**.

The following sections refer to Figure 3.6 as an example of how you can set up such a configuration.

---

◆ **Note**

*The IP addresses shown in the example configuration are fictitious. When implementing your configuration, choose IP addresses that are consistent with your network or networks.*



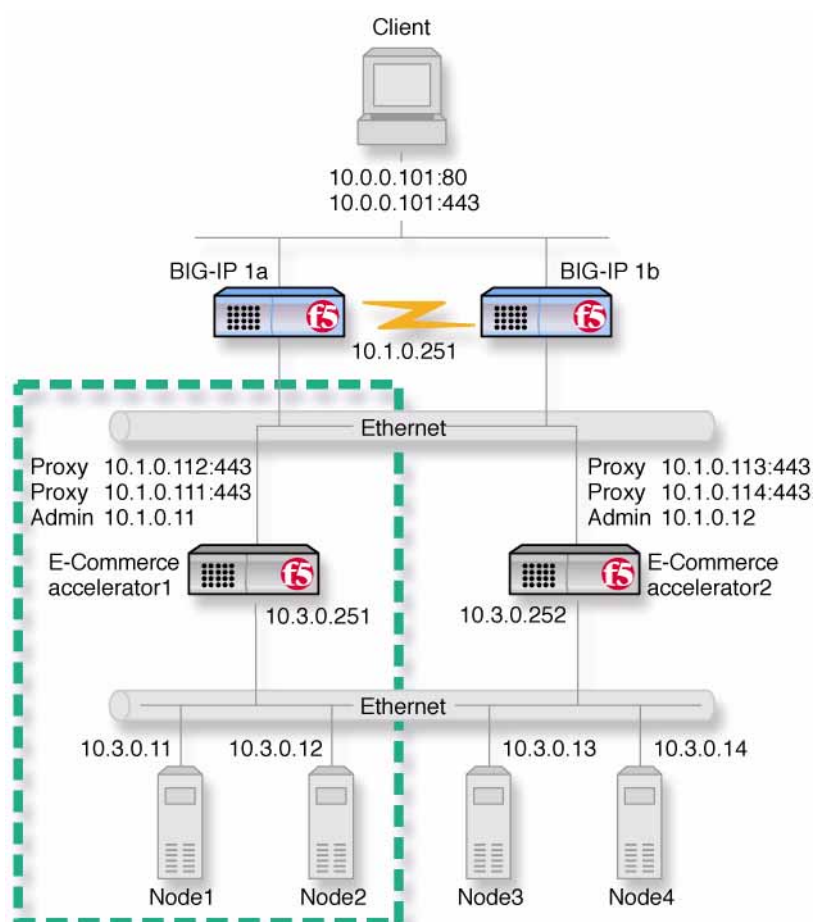


**Figure 3.6** An SSL accelerator cell configuration. The cell is outlined by the dashed line.

## Configuration tasks

To configure an SSL accelerator cell, you must configure the BIG-IP Controller redundant pair that load balances the SSL accelerators, each SSL accelerator, and each node that handles connections from the SSL accelerator.

First, complete the following tasks on the BIG-IP Controller that you want to use to load balance connections to the SSL accelerators:

- ❖ Configure interfaces on the BIG-IP Controller redundant system.
- ❖ Modify the **/etc/netstart** file on the BIG-IP Controller that you want to use to load balance the SSL accelerators.
- ❖ Create two load balancing pools. One pool load balances HTTP connections using the IP addresses of the web servers, the other pool load balances SSL connections to the SSL accelerators.
- ❖ Create virtual servers that reference the load balancing pools. Create one virtual server for the pool load balancing the SSL connections, and another virtual server that references the pool that load balances the HTTP connections to the SSL accelerators.
- ❖ Enable port 80 and port 443 on the controller.

Next, complete the following tasks for the SSL accelerator in the cell:

- ❖ Set up an SSL gateway for each node for which the SSL accelerator handles connections.
- ❖ Enable port 443.
- ❖ Set the idle connection timer for port 443.
- ❖ Turn on IP forwarding.

Finally, complete the following task on each node in the cell:

- ❖ Set the default route on each node in the cell to point to the internal interface (source processing) of the SSL accelerator serving that cell.

## Configuring the BIG-IP Controller which load balances the SSL accelerator cells

To configure the BIG-IP Controller which load balances the SSL accelerator cells, complete the following tasks on the BIG-IP Controller. This section describes how to complete each task.

- ❖ Configure interfaces on the BIG-IP Controller.
- ❖ Modify the **/etc/netstart** file on the BIG-IP Controller that you want to use to load balance the SSL accelerators.
- ❖ Create two load balancing pools One pool load balances HTTP connections using the IP addresses of the web servers, the other pool load balances SSL connections from the SSL accelerators.
- ❖ Create virtual servers that reference the load balancing pools.
- ❖ Enable port 80 and port 443 on the controller.

### Configuring interfaces on the BIG-IP Controller

You must configure the interfaces on the redundant BIG-IP Controller system (1a and 1b, in Figure 3.6) to process source and destination addresses. Note that in a basic controller configuration, one interface is configured as an internal interface (source processing), and the other interface is configured as an external interface (destination processing).

In order for the SSL accelerator cell load balancing to work, you must turn destination processing **on** for the internal interface, and source processing **on** for the external interface.

#### **To configure source and destination processing 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.

- To enable source processing for this interface, click the **Enable Source Processing** check box.
- To enable destination processing for this interface, click the **Enable Destination Processing** check box.

3. Click the **Apply** button.

### **To configure source and destination processing from the command line**

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

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

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

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

```
bigpipe interface exp0 dest enable
```

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

```
bigpipe interface exp1 source enable
```

### **Add routes for nodes to /etc/netstart**

In order for traffic to pass through this configuration correctly, you must configure routes for the nodes in the SSL accelerator cell configuration on the BIG-IP Controller. Add the routes for the nodes to the end of **/etc/netstart**. In the example shown in Figure 3.6, you must add routes for **Node1**, **Node2**, **Node3**, and **Node4**. The entries look like this in the **/etc/netstart** file:

```
route add -host 10.3.0.11 -gateway 10.1.0.11
```

```
route add -host 10.3.0.12 -gateway 10.1.0.11
```

```
route add -host 10.3.0.13 -gateway 10.1.0.12
```

```
route add -host 10.3.0.14 -gateway 10.1.0.12
```

## Create load balancing pools

This section describes how to create the load balancing pools required for the SSL accelerator configuration described in Figure 3.6. The two pools you need to create are:

- ❖ A load balancing pool for connections using the IP addresses of the web server. For this example, the HTTP pool is named **http\_virtual**. This pool contains the following members:  
**Node1 (10.3.0.11)**  
**Node2 (10.3.0.12)**  
**Node3 (10.3.0.13)**  
**Node4 (10.3.0.14)**
- ❖ A load balancing pool for SSL connections from the SSL accelerators. For this example, the SSL accelerator is named **ssl\_gateways**. This pool contains the following member:  
**accelerator1 (10.1.0.111)**  
**accelerator2 (10.1.0.112)**  
**accelerator3 (10.1.0.113)**  
**accelerator4 (10.1.0.114)**

---

### ◆ Note

*Note that the SSL accelerator pool should contain the SSL accelerator for each SSL accelerator cell.*

## 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 this example, you could create an HTTP pool named **http\_virtual**. This pool contains the following members:  
**Node1 (10.3.0.11)**

**Node2 (10.3.0.12)**

**Node3 (10.3.0.13)**

**Node4 (10.3.0.14)**

- For this example, you could create an SSL accelerator pool named **ssl\_gateways**. This pool contains the following member:  
**accelerator1 (10.1.0.111)**  
**accelerator2 (10.1.0.112)**  
**accelerator3 (10.1.0.113)**  
**accelerator4 (10.1.0.114)**
- 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 **http\_virtual** and the pool **ssl\_gateways**, you would type the following commands:

```
bigpipe pool http_virtual { lb_mode rr member 10.3.0.11:80 member  
  10.3.0.12:80 member 10.3.0.13:80 member 10.3.0.14:80 }
```

```
bigpipe pool ssl_gateways { lb_mode rr member 10.1.0.111:80 member  
  10.1.0.112:80 member 10.1.0.113:80 member 10.1.0.114:80 }
```

### Create the virtual servers

Create a virtual server that references the pool load balancing the SSL connections, and another virtual server that references the pool that load balances the HTTP connections through the SSL accelerators.

### To define a standard virtual server that references a pool 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. Fill in the attributes for the virtual server.

**Configuration notes**

- To create the configuration described in Figure 3.6, create a virtual server **10.0.0.101** on port **443** that references the pool of SSL accelerators.
- To create the configuration described in Figure 3.6, create a virtual server **10.0.0.101** on port **80** that references the pool of content servers.
- For additional information about this screen, click the **Help** button on the tool bar.

**To define a standard virtual server mapping from the command line**

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

**bigpipe vip <virt IP>:<port> use pool <pool\_name>**

To create the virtual servers for the configuration in Figure 3.6, you could type the following commands, where the pool of SSL accelerators is named **ssl\_gateways** and the pool for HTTP requests is named **http\_virtual**:

**bigpipe vip 10.0.0.101:443 use pool ssl\_gateways**

**bigpipe vip 10.0.0.101:80 use pool http\_virtual**

**Enable ports 80 and 443 on the BIG-IP Controller**

For security reasons, the BIG-IP Controller ports do not accept traffic until you enable them. In this configuration, the BIG-IP Controller accepts traffic on port 443 for SSL, and port 80 for HTTP. For this configuration to work, you must enable port 80 and port 443.

Use the following command to enable these ports:

**bigpipe port 80 443 enable**

## Configuring an SSL accelerator for use in a cell

The next part of the process in configuring an SSL accelerator cell is to configure the SSL accelerator. Complete the following tasks on each SSL accelerator in the cell:

- ❖ Set up an SSL gateway for each node for which the SSL accelerator handles connections.
- ❖ Enable port 443.
- ❖ Set the idle connection timer for port 443.
- ❖ Turn on IP forwarding.

### Set up an SSL gateway for each node in the SSL accelerator cell

The first task you must complete on the SSL accelerator is to set up an SSL gateway for each node for which the SSL accelerator handles connections. Using the example for creating an SSL Accelerator cell in Figure 3.6, you create two SSL gateways on **accelerator1**:

- ❖ An SSL gateway (10.1.0.111) with **Node1** (10.3.0.11) as a target
- ❖ An SSL gateway (10.1.0.112) with **Node2** (10.3.0.12) as a target

The following section includes procedures for adding an SSL gateway to the SSL Accelerator configuration.

#### Creating an SSL gateway using the Configuration utility

1. In the navigation pane, click **Proxies**.  
The Proxies screen opens.
2. On the toolbar, click **Add Proxy**.  
The Add Proxy screen opens.
3. In the **Proxy Address** box, type the IP address for the SSL gateway. For **accelerator1** SSL accelerator cell, the IP address for the gateway is **10.1.0.111**. For **accelerator2** SSL accelerator cell, the IP address for the gateway is **10.1.0.112**.
4. In the **Proxy Netmask** box, type the netmask you want to use for the SSL gateway. If you leave this setting blank, the BIG-IP Controller creates a default based on the network



class of the IP address on the external (destination processing) interface. Type a user-defined netmask only if necessary.

5. In the **Proxy Broadcast** box, type the broadcast address you want to use for this SSL gateway. The BIG-IP Controller automatically generates a broadcast address if you do not type one. Type a user-defined broadcast address only if necessary.
6. In the **Proxy Port** box, type the port number that the proxy server uses, or select a service from the list box. Note that if you select a service, the Configuration utility uses the default port number associated with that service.
7. For **Interface**, select the destination processing interface on which you want to create the SSL gateway. Select **default** to allow the Configuration utility to select the interface based on the network address of the SSL gateway.
8. In the **Destination Address** box, type the IP address or host name of the node to which the SSL gateway maps.
9. In the **Destination Port** box, type a port name or number, such as port **80** or **http**, or select the service name from the drop-down list.
10. In the **SSL Certificate** box, type the name of the SSL certificate you installed on the BIG-IP Controller. You can select the certificate you want to use from the drop down list.
11. In the **SSL Key** box, type the name of the SSL key for the certificate you installed on the BIG-IP Controller. You can select the key from the drop down list. It is important that you select the key used to generate the certificate you selected in the **SSL Certificate** box.
12. Click **Apply**.

### Creating an SSL gateway from the command line

Use the following command syntax to create an SSL gateway. Use this syntax if you want to configure a gateway

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

For example, to create the SSL gateways for the **accelerator1** SSL accelerator cell, you would use the following commands:

```
bigpipe proxy 10.1.0.111:443 exp0 { netmask 255.255.255.0  
broadcast 10.1.0.255 target server 10.3.0.11:80 ssl enable key  
my.server.net.key cert my.server.net.cert }  
bigpipe proxy 10.1.0.111:443 exp0 { netmask 255.255.255.0  
broadcast 10.1.0.255 target server 10.3.0.12:80 ssl enable key  
my.server.net.key cert my.server.net.cert }
```

### Enable port 443

For security reasons, the ports on the SSL accelerator do not accept traffic until you enable them. In this configuration, the SSL accelerator accepts traffic on port 443 for SSL. For this configuration to work, you must enable port 443. Use the following command to enable this port:

```
bigpipe port 443 enable
```

### Set the idle connection timer for port 443

In the SSL accelerator cell configuration, you should set the idle connection timer to clean up closed connections on port 443. You need to set an appropriate idle connection time-out value so that valid connections are not disconnected, and closed connections are cleaned up in a reasonable time.

#### To set the idle connection time-out using the Configuration utility

1. In the navigation pane, click **Virtual Servers**.
2. In the Virtual Servers list, click the virtual server you configured for SSL connections.  
The Virtual Server Properties screen opens.
3. In the **Port** box, click the port. For the example in this section, choose **443**.  
The Global Virtual Port Properties screen opens.

4. In the **Idle connection timeout TCP (seconds)** box, type a time-out value for TCP connections. The recommended time-out setting is 10 seconds.
5. In the **Idle connection timeout UDP (seconds)** box, type a time-out value for TCP connections. The recommended time-out setting is 10 seconds.
6. Click **Apply**.

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

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

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

For the example in Figure 3.6, the entries look like this:

```
treaper 443 10  
udp 443 10
```

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

### **Turn on IP forwarding**

In order for traffic from the nodes to be routed back to the client correctly, you must turn on IP forwarding for the SSL accelerator in the cell.

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.
4. Click the **Apply** 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 ***BIG-IP Controller Reference Guide**, System Control Variables*.

## **Setting the default route on each node in a cell**

The final task you must complete for this configuration is to set the default route on each node in the cell to point to the internal interface (source processing) of the SSL accelerator serving that cell.

In the configuration described in Figure 3.6, the default routes for the content servers should be set like this:

- ❖ You should set the default route on **Server1** and **Server2** to the internal address of **accelerator1**, which is **10.3.0.251**.
- ❖ You should set the default route on **Server3** and **Server4** to the internal address of **accelerator2**, which is **10.3.0.252**.

### **◆ Note**

*For information about how to set the default route on the content servers in your network, refer to the documentation provided with the content server.*

## Introducing the SSL accelerator half sandwich configuration

This chapter explains how to set up a scalable SSL accelerator configuration. This configuration is useful for any enterprise that handles a large amount of encrypted traffic.

With this configuration, you can increase the scale of the network by adding new SSL accelerators to the configuration. You can use this configuration to load balance encrypted traffic to the SSL accelerators while standard HTTP traffic is sent directly to a second BIG-IP Controller which load balances the connections to the content servers.

Figure 3.7 shows a configuration of an SSL accelerator half-sandwich. The following sections refer to Figure 3.7 as an example of how you can set up such a configuration.

---

### ◆ Note

*The IP addresses shown in the example configuration are fictitious. When implementing your configuration, choose IP addresses that are consistent with your network or networks.*

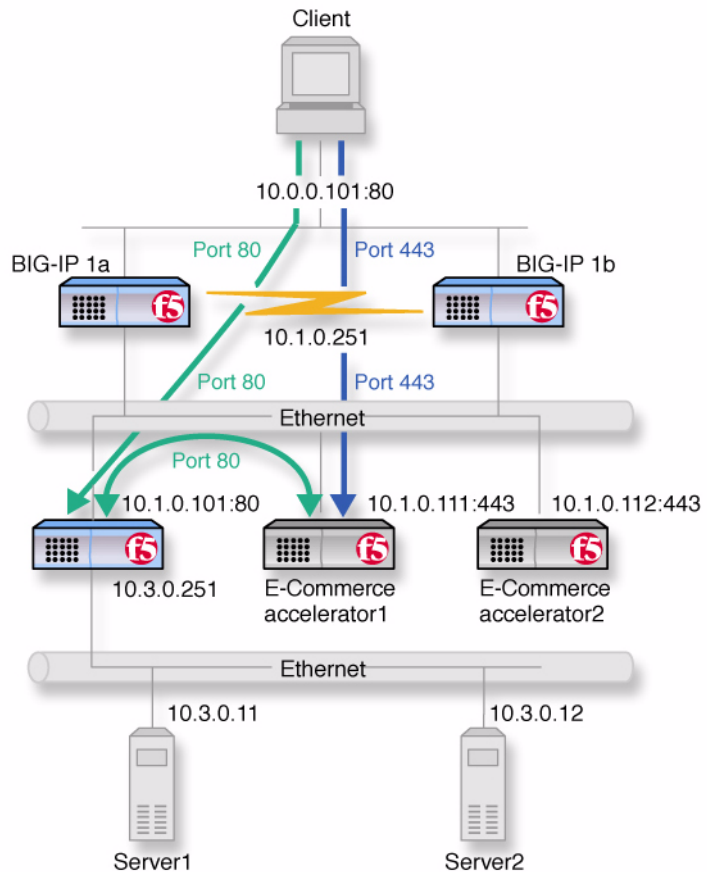


**Figure 3.7** An SSL accelerator half-sandwich.

## Configuration tasks

First, complete the following tasks on the BIG-IP Controllers **1a** and **1b** that you want to use to load balance traffic coming into your network:

- ❖ Create two load balancing pools. One pool load balances HTTP connections using the IP address of a virtual server on the second BIG-IP Controller (10.3.0.251), and another pool that load balances SSL connections to the SSL accelerators.
- ❖ Create two virtual servers. One virtual server references the pool that contains the IP address of the virtual server on the other controller. The second virtual server references the pool for load balancing the SSL accelerators.
- ❖ Enable port 80 and port 443 on the controller.

Next, complete the following tasks on each SSL accelerator in the half-sandwich:

- ❖ Configure interfaces on each SSL accelerator.
- ❖ Set up an SSL gateway that points to the virtual server that handles HTTP requests on the BIG-IP Controller (10.3.0.251).

Next, complete the following tasks on the BIG-IP Controller (10.3.0.251) that load balances HTTP requests from the SSL accelerators and HTTP requests from the BIG-IP Controllers **1a** and **1b**:

- ❖ Configure interfaces on the second BIG-IP Controller.
- ❖ Create a pool of web servers that handle HTTP connections.
- ❖ Create a pool of devices from which the controller receives HTTP connections.
- ❖ Create one virtual server that handles connections for the content servers.
- ❖ Creating a last hop pool of devices from which the controller receives requests
- ❖ Adding the last hop pool from which this controller receives HTTP connections to the virtual server
- ❖ Enable port 80.

Next, complete the following tasks on each content server:

- ❖ Set the default route on each node in the cell to point to the internal IP address of the second BIG-IP Controller.

## Configuring the BIG-IP Controllers handling inbound traffic

First, complete the following tasks on the BIG-IP Controllers **1a** and **1b** that you want to use to load balance traffic coming into your network:

- ❖ Create two load balancing pools. One pool load balances HTTP connections using the IP address of a virtual server on the second BIG-IP Controller (10.3.0.251), and another pool load balances SSL connections to the SSL accelerators.
- ❖ Create two virtual servers. One virtual server references the pool that contains the IP address of the virtual server on the other controller. The second virtual server references the pool for load balancing the SSL accelerators.
- ❖ Enable port 80 and port 443 on the controller.

### Create load balancing pools for HTTP and SSL requests

Create two load balancing pools. One pool load balances HTTP connections using the IP address of a virtual server on the second BIG-IP Controller (10.3.0.251), and another pool load balances SSL connections to the SSL accelerators.

This section describes how to create the load balancing pools required for the SSL accelerator configuration described in Figure 3.7. The two pools you need to create are:

- ❖ A load balancing pool that load balances HTTP connections using the IP address of a virtual server on the second BIG-IP Controller (10.3.0.251). For this example, the HTTP pool is named **http\_virtual**. This pool contains the member **10.1.0.101:80**.
- ❖ A load balancing pool for SSL connections to the SSL accelerators. For this example, the SSL accelerator is named **ssl\_gateways**. This pool contains the following members:  
**accelerator1 (10.1.0.111:443)**  
**accelerator2 (10.1.0.112:443)**

### 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 notes

- For this example, you could create an HTTP pool named **http\_virtual**. This pool contains the following member:  
**10.1.0.101:80**
- For this example, you could create an SSL accelerator pool named **ssl\_gateways**. This pool contains the following members:  
**accelerator1 (10.1.0.111:443)**  
**accelerator2 (10.1.0.112:443)**
- 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_method <lb_method> member
  <member_definition> ... member <member_definition>}
```

For example, if you want to create the pool **http\_virtual** and the pool **ssl\_gateways**, you would type the following command:

```
bigpipe pool http_virtual { lb_mode rr member 11.1.0.101:80 }
bigpipe pool ssl_gateways { lb_mode rr member 10.1.0.111:443 member
  10.1.0.112:443 }
```

### Creating the virtual servers that reference the HTTP and SSL pools

Create a virtual server that references the pool load balancing the SSL connections, and another virtual server that references the pool that load balances the HTTP connections to the SSL accelerators.

### To define a standard virtual server that references a pool 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. Fill in the attributes for the virtual server.

### Configuration notes

- To create the configuration described in Figure 3.7, create a virtual server **10.0.0.101** on port **443** that references the pool of SSL accelerators.
- To create the configuration described in Figure 3.7, create a virtual server **10.0.0.101** on port **80** that references the pool of content servers.
- For additional information about this screen, click the **Help** button on the tool bar.

### To define a standard virtual server mapping from the command line

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

**bigpipe vip <virt IP>:<port> use pool <pool\_name>**

To create the virtual servers for the configuration in Figure 3.7, you could type the following commands, where the pool of SSL accelerators is named **ssl\_gateways** and the pool for HTTP requests is named **http\_virtual**:

**bigpipe vip 10.0.0.101:443 use pool ssl\_gateways**

**bigpipe vip 10.0.0.101:80 use pool http\_virtual**

### Enable ports 80 and 443

For security reasons, the BIG-IP Controller ports do not accept traffic until you enable them. In this configuration, the BIG-IP Controller accepts traffic on port 443 for SSL, and port 80 for HTTP. For this configuration to work, you must enable port 80 and port 443. Use the following command to enable these ports:

**bigpipe port 80 443 enable**

## Configuring each SSL accelerator

Next, complete the following tasks on each SSL accelerator in the half-sandwich:

- ❖ Configure interfaces on each SSL accelerator.
- ❖ Set up an SSL gateway that points to the virtual server that handles HTTP requests on the BIG-IP Controller (10.3.0.251).
- ❖ Set the idle connection timer for port 443.

### Configuring interfaces on each SSL accelerator

You must configure the interfaces on the each SSL accelerator to process source and destination addresses. In a basic controller configuration, one interface is configured as an internal interface (source processing), and the other interface is configured as an external interface (destination processing).

In order for the SSL accelerator half sandwich to work, you must turn destination processing **on** for the internal interface, and source processing **on** for the external interface.

#### **To configure source and destination processing 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.
  - To enable source processing for this interface, click the **Enable Source Processing** check box.
  - To enable destination processing for this interface, click the **Enable Destination Processing** check box.
3. Click the **Apply** button.

**To configure source and destination processing from the command line**

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

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

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

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

```
bigpipe interface exp0 dest enable
```

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

```
bigpipe interface exp1 source enable
```

**Setting up an SSL gateway that points to the HTTP virtual server on the second BIG-IP Controller**

The next step is to set up an SSL gateway that points to the virtual server that handles HTTP requests on the BIG-IP Controller (10.3.0.251). The SSL gateway passes the HTTP request to the BIG-IP Controller which then load balances them to the content servers.

The first task you must complete on the SSL accelerator is to set up an SSL gateway for each node for which the SSL accelerator handles connections. Using the example for creating an SSL Accelerator cell in Figure 3.7, you create an SSL gateway on **accelerator1** and an SSL gateway on **accelerator2**:

- ❖ An SSL gateway on **accelerator1** that has the virtual server **10.1.0.101:80** as a target
- ❖ An SSL gateway on **accelerator2** that has the virtual server **10.1.0.101:80** as a target

The following section includes procedures for adding an SSL gateway to the SSL Accelerator configuration.

**Creating an SSL gateway using the Configuration utility**

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

2. On the toolbar, click **Add Proxy**.  
The Add Proxy screen opens.
3. In the **Proxy Address** box, type the IP address for the SSL gateway. For **accelerator1** SSL accelerator, the IP address for the gateway is **10.1.0.111:443**. When you create the second SSL gateway for **accelerator2**, the IP address for the gateway is **10.1.0.112:443**.
4. In the **Proxy Port** box, type the port number that the proxy server uses, or select a service from the list box. Note that if you select a service, the Configuration utility uses the default port number associated with that service.
5. For **Interface**, select the destination processing interface on which you want to create the SSL gateway. Select **default** to allow the Configuration utility to select the interface based on the network address of the SSL gateway.
6. In the **Destination Address** box, type the IP address or host name of the node to which the SSL gateway maps. In this example, the destination should be the virtual server **10.1.0.101** on the second BIG-IP Controller.
7. In the **Destination Port** box, type a port name or number, such as port **80** or **http**, or select the service name from the drop-down list.
8. In the **SSL Certificate** box, type the name of the SSL certificate you installed on the BIG-IP Controller. You can select the certificate you want to use from the drop down list.
9. In the **SSL Key** box, type the name of the SSL key for the certificate you installed on the BIG-IP Controller. You can select the key from the drop down list. It is important that you select the key used to generate the certificate you selected in the **SSL Certificate** box.
10. Click **Apply**.

### Creating an SSL gateway from the command line

Use the following command syntax to create an SSL gateway. Use this syntax if you want to configure a gateway:

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

For example, to create the SSL gateways for the **accelerator1** SSL accelerator cell, you would use the following commands:

```
bigpipe proxy 10.1.0.111:443 exp0 { netmask 255.255.255.0
  broadcast 10.1.0.255 target server 10.1.0.101:80 ssl enable key
  my.server.net.key cert my.server.net.cert }
bigpipe proxy 10.1.0.112:443 exp0 { netmask 255.255.255.0
  broadcast 10.1.0.255 target server 10.1.0.101:80 ssl enable key
  my.server.net.key cert my.server.net.cert }
```

## Configuring the BIG-IP Controller that load balances the content servers

Next, complete the following tasks on the BIG-IP Controller (10.3.0.251) that load balances HTTP requests from the SSL accelerators and HTTP requests from the BIG-IP Controllers **1a** and **1b**:

- ❖ Configure interfaces on the second BIG-IP Controller.
- ❖ Create a pool of web servers that handle HTTP connections.
- ❖ Create a virtual server that handles connections for the content servers.
- ❖ Creating a last hop pool of devices from which the controller receives requests
- ❖ Adding the last hop pool from which this controller receives HTTP connections to the virtual server
- ❖ Enable port 80.
- ❖ Set the default route on the controller to the internal IP alias of the BIG-IP Controllers **1a** and **1b**.

### Configure interfaces for the BIG-IP Controller

You must configure the interfaces on the redundant BIG-IP Controller system (**1a** and **1b**, in Figure 3.7) to process source and destination addresses. Note that in a basic controller configuration,

one interface is configured as an internal interface (source processing), and the other interface is configured as an external interface (destination processing).

In order for the SSL accelerator cell load balancing to work, you must turn destination processing **on** for the internal interface, and source processing **on** for the external interface.

### **To configure source and destination processing in 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.
  - To enable source processing for this interface, click the **Enable Source Processing** check box.
  - To enable destination processing for this interface, click the **Enable Destination Processing** check box.
3. Click the **Apply** button.

### **To configure source and destination processing from the command line**

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

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

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

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

```
bigpipe interface exp0 dest enable
```

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

```
bigpipe interface exp1 source enable
```

## Creating a pool for the content servers

This section describes how to create the load balancing pools required for the SSL accelerator configuration described in Figure 3.7.

The pool you need to create is a load balancing pool for connections using the IP addresses of the web server. For this example, the HTTP pool is named **http\_virtual**. This pool contains the following members:

**Server1 (10.3.0.11)**

**Server2 (10.3.0.12)**

### 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 notes

- For this example, you could create an HTTP pool named **http\_virtual**. This pool contains the following members:  
**server1 (10.3.0.11)**  
**server2 (10.3.0.12)**
- 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 **http\_virtual**, you would type the following command:



```
bigpipe pool http_virtual { lb_mode rr member 10.3.0.11:80 member
10.3.0.12:80 }
```

### Creating a virtual server that references the HTTP pool

Next, create a virtual server that references the pool load balancing HTTP connections.

#### To define a standard virtual server that references a pool 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. Fill in the attributes for the virtual server.

#### Configuration notes

- To create the configuration described in Figure 3.7, create a virtual server **10.0.0.101** on port **80** that references the pool of content servers.
- For additional information about this screen, click the **Help** button on the tool bar.

#### To define a standard virtual server mapping from the command line

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

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

To create the virtual server for the configuration in Figure 3.7, you could type the following command, where the pool for HTTP requests is named **http\_virtual**:

```
bigpipe vip 10.0.0.101:80 use pool http_virtual
```

## Creating a last hop pool of devices from which the controller receives requests

This section describes how to create the load balancing pools required for the SSL accelerator configuration described in Figure 3.7.

The pool you need to create is a load balancing pool you can use for a last hop pool for connections received from other devices by the controller. For this example, the HTTP pool is named **http\_sources**. This pool contains the following members:  
**BIG-IP 1a and 1b internal alias (10.1.0.251)**  
**accelerator1 (10.1.0.111)**  
**accelerator (10.1.0.112)**

### 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 notes

- For this example, you could create an HTTP pool named **http\_sources**. This pool contains the following members:  
**BIG-IP 1a and 1b internal alias (10.1.0.251:any)**  
**accelerator1 (10.1.0.111:any)**  
**accelerator (10.1.0.112:any)**
- Specify **any** for the port for each member.
- 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 **http\_sources**, you would type the following command:

```
bigpipe pool http_sources { lb_mode rr member 10.1.0.251:any member  
10.1.0.112:any member 10.1.0.112:any }
```

Adding the last hop pool from which this controller receives HTTP connections to the virtual server

The next step is to add the last hop pool of all the devices (**http\_sources**) from which the controller receives HTTP connections. This pool includes each SSL accelerator that passes on HTTP connections to the second BIG-IP Controller.

#### **To configure a last hop pool using the Configuration utility**

1. In the navigation pane, click **Virtual Servers**.  
The Virtual Servers screen opens.
2. In the Virtual Servers Current List, select the virtual server configured for HTTP connections. In this example the virtual server is **10.1.0.101:80**.  
The Virtual Server Properties screen opens.
3. In the Last Hop Pool section, select the pool for load balancing HTTP connections from all devices. In this example, the pool is **http\_sources**.
4. Click the **Apply** button.

#### **To configure a last hop pool from the command line**

To configure a last hop pool, you must first create a pool that contains the routers for the BIG-IP Controller. Use the following command to configure a last hop pool for the virtual server **10.1.0.101:80** that uses the pool **http\_sources**.

```
bigpipe vip 10.1.0.101:80 lasthop pool http_sources
```

### Enable port 80

For security reasons, the BIG-IP Controller ports do not accept traffic until you enable them. In this configuration, the BIG-IP Controller accepts traffic on port 80 for HTTP. For this configuration to work, you must enable port 80. Use the following command to enable this port:

```
bigpipe port 80 enable
```

### Configuring the content servers

The final task you must complete for this configuration is to set the default route on each server to point to the internal interface (source processing) of the second BIG-IP Controller (10.3.0.251).

In the configuration described in Figure 3.7, the default routes for the content servers should be set like this:

You should set the default route on **Server1** and **Server2** to the internal address of the second BIG-IP Controller, which is **10.3.0.251**.

# 4

---

## Load Balancing VPNs

---

- VPN load balancing
- VPN and router load balancing



## VPN load balancing

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

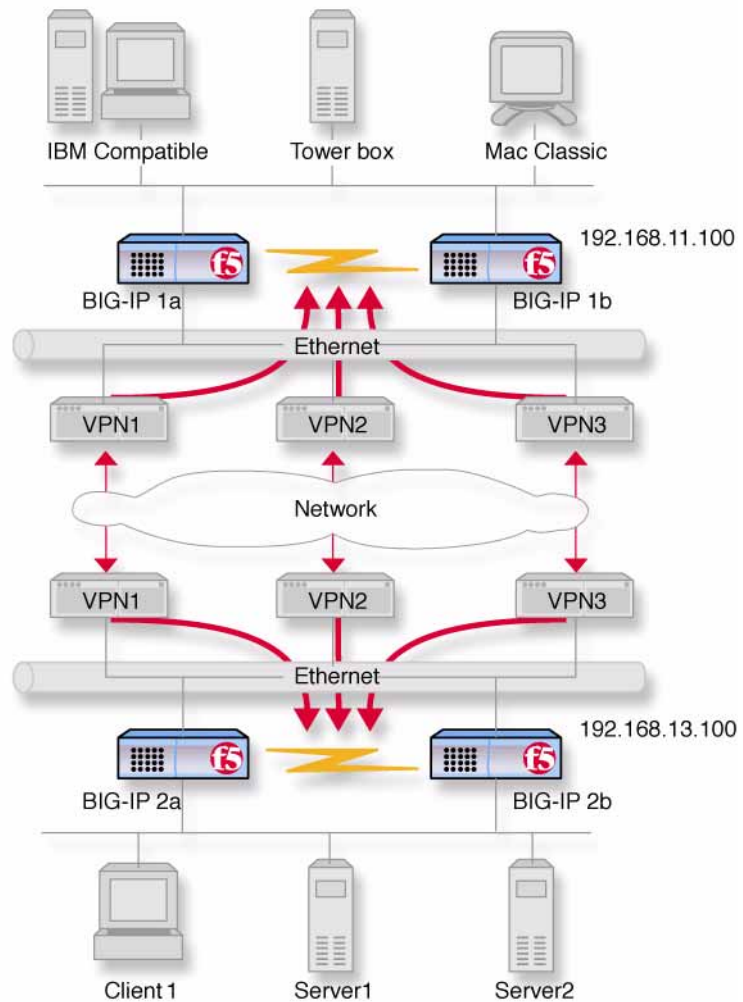


**Figure 4.1** An example of a VPN load balancing configuration



## Configuring interfaces for VPN load balancing

A VPN load balancing configuration requires special interface configuration. You must configure the interfaces on the redundant BIG-IP Controller system (1a and 1b, and 2a and 2b, in Figure 4.1) to process source and destination addresses. Note that in a basic controller configuration, one interface is configured as an internal interface (source processing), and the other interface is configured as an external interface (destination processing).

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

```
bigpipe interface exp1 dest enable
```

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

```
bigpipe interface exp0 source enable
```

## Configuring virtual servers for VPN load balancing

In the following examples only the configuration for the BIG-IP Controller on network 192.168.11 are shown (controllers 2a and 2b). The configuration for 192.168.13 is the same, only with different network numbers. Since VPNs are connection-oriented, you must set up a last hop pool for sending the return traffic back through the VPN that originated the traffic. After you create the pools, you can create the virtual servers that reference these pools.

### Defining the pools for VPN load balancing

First, define the pool **vpn\_insidest** for the internal addresses of the VPN routers. Use the following command to create the pool **vpn\_insidest**:

```
bigpipe pool vpn_insidest { lb_mode rr member <vpn1>:22 member  
  <vpn2>:22 member <vpn3>:22 }
```

Replace **<vpn1>**, **<vpn2>**, and **<vpn3>** with internal IP address of the respective routers. In this example the routers are service checked on port 22. Also note that this example uses the global round robin load balancing method.

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

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

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

### Defining the virtual servers for VPN load balancing

After you define the pools for the inside IP addresses of the routers, you can define the virtual servers for the redundant BIG-IP Controllers 2a and 2b.

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

### Inbound configuration

First, configure the controllers to handle inbound traffic from the remote network.

Create the virtual servers for controllers 2a and 2b with the following commands:

```
bigpipe vip 192.168.13.1:0 exp1 forward  
bigpipe vip 192.168.13.2:0 exp1 forward  
bigpipe vip 192.168.13.3:0 exp1 forward
```

Configure the virtual servers to use the last hop pool with the inside VPN router addresses:

```
bigpipe vip 192.168.13.1:0 lasthop pool vpn_insidess  
bigpipe vip 192.168.13.2:0 lasthop pool vpn_insidess  
bigpipe vip 192.168.13.3:0 lasthop pool vpn_insidess
```

## Outbound configuration

Next, configure controllers 2a and 2b to handle outbound traffic. Create a virtual server that sends traffic to the pool you created for the internal interfaces of the VPN routers (**vpn\_insid**s). Use the following commands to create virtual servers for connecting to the machines on the remote network:

```
bigpipe vip 192.168.11.1:0 exp1 use pool vpn_insid
bigpipe vip 192.168.11.1:0 translate addr disable
bigpipe vip 192.168.11.1:0 translate port disable
bigpipe vip 192.168.11.2:0 exp1 use pool vpn_insid
bigpipe vip 192.168.11.2:0 translate addr disable
bigpipe vip 192.168.11.2:0 translate port disable
bigpipe vip 192.168.11.3:0 exp1 use pool vpn_insid
bigpipe vip 192.168.11.3:0 translate addr disable
bigpipe vip 192.168.11.3:0 translate port disable
```

The addresses 192.168.11.1, 192.168.11.2, and 192.168.11.3 correspond to the IBM Compatible, Tower box, and Mac Classic on the remote network in Figure 4.1. Note that port translation has been turned off because the members in the **vpn\_insid**s pool were defined with port 22 for service checking. If port translation is not disabled, then all outbound connections would be translated to port 22.

## VPN and router load balancing

You can use the transparent device load balancing feature in the BIG-IP Controller to connect two private networks as well as load balance internet connections through multiple routers. Figure 4.2 is an example of this network configuration.

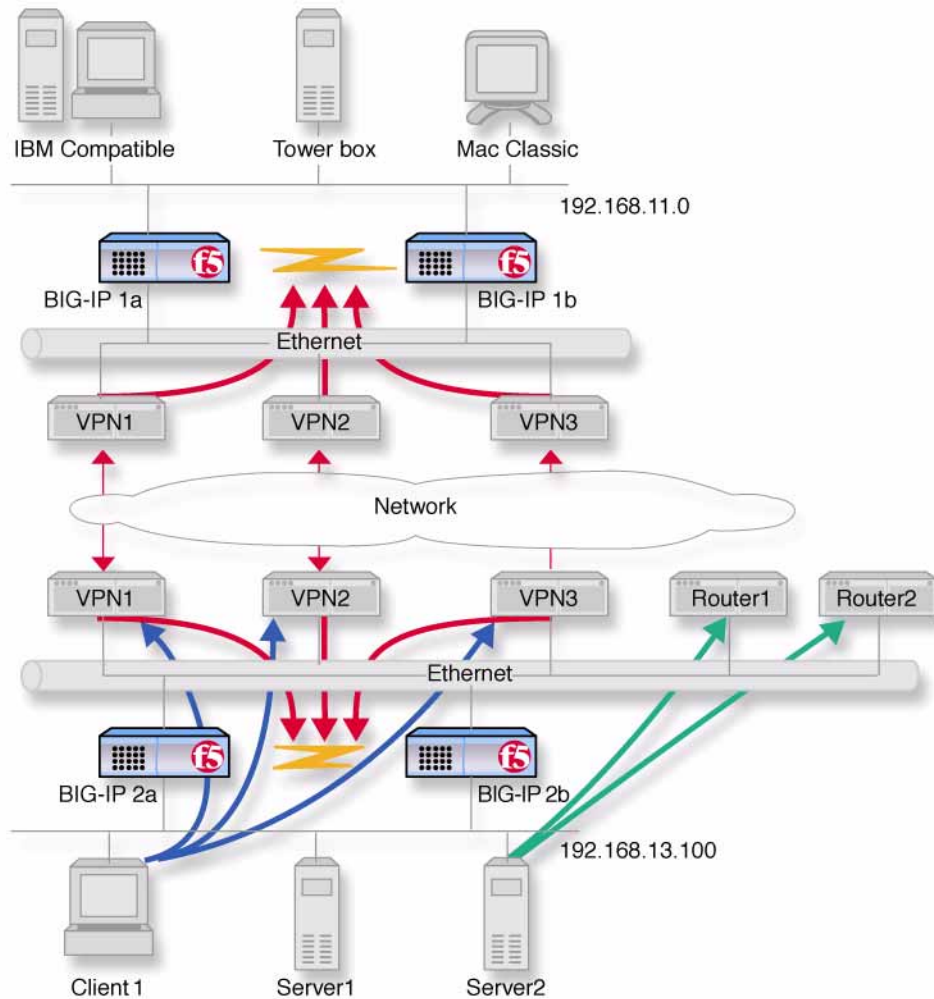


**Figure 4.2** An example of a VPN and multiple router load balancing configuration

## Configuring interfaces for VPN load balancing

A VPN load balancing configuration requires special interface configuration. The interfaces on the redundant BIG-IP Controller system (1a and 1b, and 2a and 2b, in Figure 4.2) must be set to process source and destination addresses. Note that in a basic controller configuration, one interface is configured as an internal interface (source processing), and the other interface is configured as an external interface (destination processing).

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

```
bigpipe interface exp1 dest enable
```

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

```
bigpipe interface exp0 source enable
```

## Configuring virtual servers for VPN and router load balancing

In the following examples, only the configuration for the BIG-IP Controller on network 192.168.11 are shown (controllers 2a and 2b). The configuration for 192.168.13 is the same, only with different network numbers. Since VPNs are connection-oriented, VPN and router load balancing requires you to create a pool for the inside interfaces on the VPNs and routers. After you create the pool, you can create the virtual servers that reference these pools.

### Defining the pools for VPN load balancing

First, define the pool **vpn\_insidess** for the internal addresses of the VPN routers. Use the following command to create the pool **vpn\_insidess**:

```
bigpipe pool vpn_insidess { lb_mode rr member <vpn1>:0 member  
  <vpn2>:0 member <vpn3>:0 }
```

Replace **<vpn1>**, **<vpn2>**, and **<vpn3>** with external IP address of the respective routers. Also note that this example uses the global round robin load balancing method.

### Defining pools for the additional routers

Next, define the pool **routers\_insid**es for the internal addresses of the routers. Use the following command to create the pool **routers\_insid**es:

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

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

### Defining a pool for the servers

Next, define the pool **server\_pool** for the nodes that handle the requests to virtual server 205.100.19.22:80:

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

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

### Defining a pool for all inbound traffic sources

Finally, define the pool **inbound\_sources** for all machines that can originate traffic for the virtual server 205.100.19.22:80:

```
bigpipe pool inbound_sources { lb_mode rr member <vpn1>:80 member <vpn2>:80 member <vpn3>:80 member <router1>:80 member <router2>:80 member <server1>:80 member <server2>:80 member <server3>:80 }
```

Replace **<vpn1>**, **<vpn2>**, and **<vpn3>** with internal IP address of the respective routers. Replace **<server1>**, **<server2>**, and **<server3>** with IP address of the respective server. Replace **<router1>** and **<router2>** with internal IP address of the respective routers. Also note that this example uses the global round robin load balancing method.

## Defining the virtual servers for the additional internet connection

After you define the pools for the inside IP addresses of the routers, you can define the virtual servers for the redundant BIG-IP Controllers 2a and 2b.

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

## Inbound configuration for the VPNs

First, configure the controllers to handle inbound traffic from the remote network.

Create the virtual server for controllers 2a and 2b with the following commands:

```
bigpipe vip 192.168.13.1:0 exp1 forward
bigpipe vip 192.168.13.2:0 exp1 forward
bigpipe vip 192.168.13.3:0 exp1 forward
```

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

```
bigpipe vip 192.168.13.1:0 lasthop pool vpn_insidess
bigpipe vip 192.168.13.2:0 lasthop pool vpn_insidess
bigpipe vip 192.168.13.3:0 lasthop pool vpn_insidess
```

Note that by using the last hop pool **vpn\_insidess**, only connections that originate from the remote network, through the VPNs, will be allowed to connect to the local 192.168.11 network.

## Outbound configuration for the VPNs

Next, configure controllers 2a and 2b to handle outbound traffic. Create a virtual server that sends traffic to the pool you created for the internal interfaces of the VPN routers (**vpn\_insidess**). Use the following commands to create virtual servers for connecting to the machines on the remote network:

```
bigpipe vip 192.168.11.1:0 exp1 use pool vpn_insidess
bigpipe vip 192.168.11.1:0 translate addr disable
```

```
bigpipe vip 192.168.11.2:0 exp1 use pool vpn_insidess
bigpipe vip 192.168.11.2:0 translate addr disable
bigpipe vip 192.168.11.3:0 exp1 use pool vpn_insidess
bigpipe vip 192.168.11.3:0 translate addr disable
```

The addresses 192.168.11.1, 192.168.11.2, and 192.168.11.3 correspond to the IBM Compatible, Tower box, and Mac Classic on the remote network in Figure 4.1, on page 4-2.

### Inbound configuration for internet traffic

First, configure the controllers to handle inbound traffic.

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

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

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

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

Note that by using the last hop pool **inbound\_sources**, this virtual server will accept connections that originate from either the remote network via the VPNs, or from the internet via the routers.

### Outbound configuration for internet traffic

Next, configure controllers 1a and 1b to handle outbound traffic.

Create a virtual server that sends traffic to the pool you created for the internal interfaces of the routers (**router\_insidess**). Use the following command to create the virtual server:

```
bigpipe vip 0.0.0.0:0 exp1 use pool router_insidess
```



# 5

---

## ISP Load Balancing

---

- ISP load balancing



## ISP load balancing

You may find that as your network grows, or network traffic increases, you need to add an additional connection to the internet. You can use this configuration to add an additional internet connection to your existing network. Figure 5.1 shows an additional internet connection:

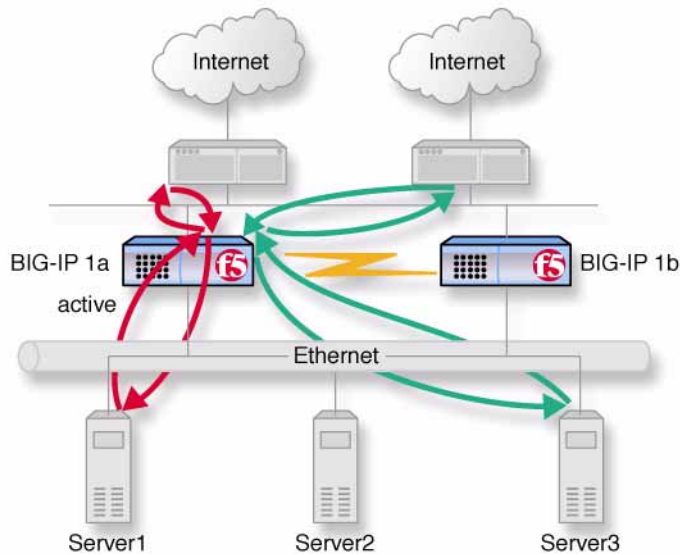


**Figure 5.1** An example of an additional internet connection

## Configuring interfaces for the additional internet connection

An additional internet connection requires special interface configuration. You must set interfaces on the redundant BIG-IP Controller system (1a and 1b in Figure 5.1) to process source and destination addresses. Note that in a basic controller configuration, one interface is configured as an internal interface (source processing), and the other interface is configured as an external interface (destination processing).

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

```
bigpipe interface exp1 dest enable
```

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

```
bigpipe interface exp0 source enable
```

## Configuring virtual servers for an additional internet connection

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

### Defining the pools for the additional internet connection

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

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

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

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

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

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

## Defining the virtual servers for the additional internet connection

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

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

### Inbound configuration

First, configure the controllers to handle inbound traffic.

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

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

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

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

### Outbound configuration

Next, configure controllers 1a and 1b to handle outbound traffic.

Create a virtual server that sends traffic to the pool you created for the internal interfaces of the routers (**router\_insides**). Use the following command to create the virtual server:

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



# 6

---

## One IP Network Topologies

---

- One IP network topology with one interface
- One IP network topology with two interfaces





## One IP network topology with one interface

Another configuration option you can use with the BIG-IP Controller is a single interface configuration when there is only one network in the topology. This configuration uses client SNAT which prevents servers from knowing the client IP address.

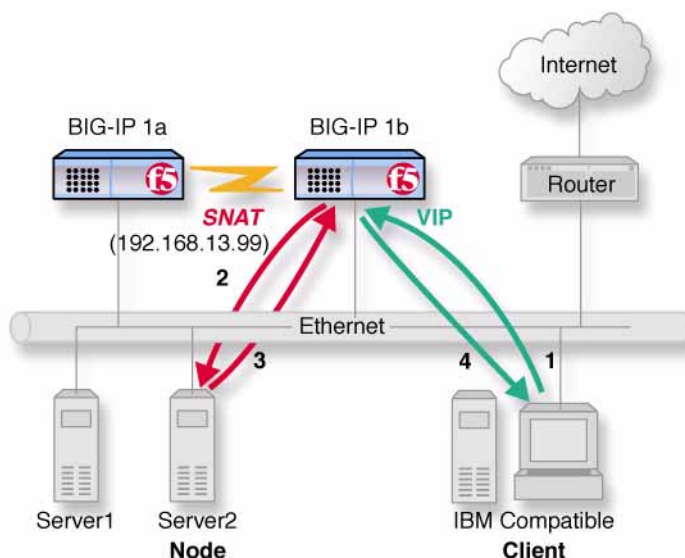


*Figure 6.1 An example of a single interface topology*

## Configuring the interface in the single interface topology

A single IP network topology with a single interface requires special interface configuration. You need to configure the single interface on the redundant BIG-IP Controller system (1a and 1b, in Figure 6.1) to process source and destination addresses. In a basic controller configuration, one interface is configured as an internal interface (source processing), and the other interface is configured as an external interface (destination processing).

Use the following commands to turn source and destination processing on for the interface; in this example, the interface name is **exp0**:

```
bigpipe interface exp0 source enable  
bigpipe interface exp0 dest enable
```

## Defining a pool for the servers

First, define the pool **server\_pool** for the nodes that handle the requests to virtual server 192.168.13.1:80:

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

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

## Virtual server configuration

Next, configure controllers 1a and 1b to load balance connections to the servers. Create a virtual server that sends traffic to the pool you created for the servers (**server\_pool**). Use the following commands to create a virtual server for connecting to the servers:

```
bipipe vip 192.168.13.1:80 use pool server_pool
```

## Client SNAT configuration

Finally, configure controllers 1a and 1b to handle connections originating from the client. A SNAT must be defined in order to change the source address on the packet to the SNAT external address, which is located on the BIG-IP Controller. If a SNAT were not defined, the server would return the packets directly to the client without giving the BIG-IP Controller the opportunity to translate the source address from the server address back to the virtual server. The client would not recognize the packet if the source

address of the returning packet is the IP address of the real server because the client sent its packets to the IP address of the *virtual* server.

**bipipe snat map client1 to 192.168.13.99**

Replace **client1** with the actual name of the client in your configuration.

## One IP network topology with two interfaces

The one IP network with two interfaces configuration is similar to the one IP network with one interface configuration, except that it uses two interfaces to optimize throughput.

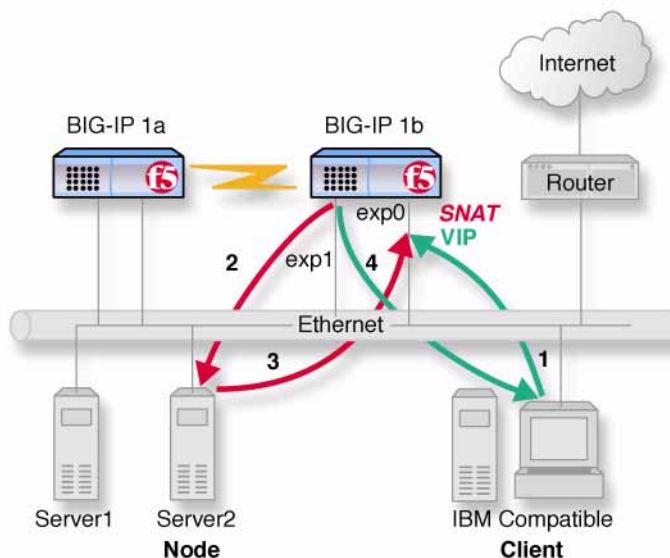


**Figure 6.2** An example of a single IP network with two interfaces topology

## Configuring the interfaces in the single IP network with two interfaces topology

A single IP network with two interfaces topology requires special interface configuration. You must configure both interfaces on the redundant BIG-IP Controller system (1a and 1b, in Figure 6.2) to process source and destination addresses. Note that in a basic controller configuration, one interface is configured as an internal interface (source processing), and the other interface is configured as an external interface (destination processing).

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

```
bigpipe interface exp1 dest enable
```

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

```
bigpipe interface exp0 source enable
```

## Routing issues

By setting up the IP addresses and interfaces properly, you can configure the BIG-IP Controller to receive all traffic through one interface and to send all traffic out the other interface. The key to optimizing the throughput in this configuration is routing.

In this example, the administrative IP addresses for the BIG-IP Controller are placed on **exp1**. This is setup when you first configure the BIG-IP Controller, or it can be changed anytime by editing the **/etc/netstart** file. Once the administrative address is configured, the BIG-IP Controller sets up a route to the IP network going through **exp1**. The **exp0** interface should not be configured with an IP address on the same IP network, because that creates a routing conflict and the BIG-IP Controller would not know which interface the IP network is accessible through. Once the route is set up properly, all traffic sent from the BIG-IP Controller to that IP network goes through **exp1**.

In order to receive traffic through **exp0**, the virtual server and SNAT external address in this example are explicitly declared to reside on **exp0**. This causes the BIG-IP Controller to respond to ARP requests for those addresses from the **exp0** interface. Virtual servers and SNAT addresses will not create a routing conflict for the IP network they are declared with. Only administrative or shared IP addresses create routes to the corresponding IP network through the interface that they are configured on. In other words, in this example, the BIG-IP Controller determines that the 192.168.13 network is on interface **exp1** and it sends all traffic to those addresses through that interface.

## Defining a pool for the servers

First, define the pool **server\_pool** for the nodes that handle the requests to virtual server 192.168.13.1:80:

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

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

## Virtual server configuration

Next, configure controllers 1a and 1b to load balance connections to the servers. Create a virtual server that sends traffic to the pool you created for the servers (**server\_pool**). Use the following commands to create a virtual server for connecting to the servers:

```
bigpipe vip 192.168.13.1:80 exp0 use pool server_pool
```

## Client SNAT configuration

Finally, configure controllers 1a and 1b to handle connections originating from the client. A SNAT must be defined in order to change the source address on the packet to the SNAT external address, which is located on the BIG-IP Controller. If a SNAT is

not defined, the server returns the packets directly to the client without giving the BIG-IP Controller the opportunity to translate the source address from the server address back to the virtual server. The client would not recognize the packet if the source address of the returning packet is the IP address of the real server because the client sent its packets to the IP address of the virtual server.

```
bipipe snat map client1 to 192.168.13.99 exp0
```

Replace **client1** with the actual name of the client in your configuration.

# 7

---

## nPath routing

---

- nPath routing





## nPath routing

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

---

### ◆ Note

*This configuration does not support late binding features such as SSL persistence, cookie persistence, and content switching.*

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

The following tasks are required to configure the BIG-IP Controller to use nPath routing:

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

## Defining a virtual server with address translation disabled

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

### To define a standard virtual server mapping in 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 **Address** box, enter the virtual server's IP address or host name.
4. In the **Netmask** box, type an optional netmask. If you leave this setting blank, the BIG-IP Controller uses a default netmask based on the IP address you entered for the virtual server. Use the default netmask unless your configuration requires a different netmask.
5. In the **Broadcast** box, type the broadcast address for this virtual server. If you leave this box blank, the BIG-IP Controller generates a default broadcast address based on the IP address and netmask of this virtual server.
6. In the **Port** box, either type a port number, or select a service name from the drop-down list.
7. For **Interface**, select the external (destination processing) interface on which you want to create the virtual server. Select **default** to allow the Configuration utility to select the interface based on the network address of the virtual server.
8. In Resources, click the **Node List** button.
9. In the **Node Address** box, type the IP address or host name of the first node to which the virtual server maps. If you have already defined a node address, you can choose it from the list.
10. In the **Node Port** box, type the node port number, or select the service name from the drop-down list. If you have already defined a node port, you can choose it from the list.
11. Click the add button (>>) to add the node the Current Members list for the virtual server.

12. To add additional nodes to the virtual server mapping, type in a Node Address, Node Port, and click the add button (>>).
13. To remove nodes from the virtual server mapping, click the node listed in the Current Members list and click the remove button (<<).
14. After you have added or removed nodes from the Current Members list, click the **Add** button to save the virtual server.

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

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

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

### To define a virtual server mapping on the command line

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

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

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

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

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

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

## Setting the route through the BIG-IP Controller

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

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

## Setting the idle connection time-out

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

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

1. In the navigation pane, click **Virtual Servers**.
2. In the Virtual Servers list, click the wildcard virtual server you created for nPath routing.  
The Virtual Server Properties screen opens.
3. In the **Port** box, click the port.  
The Global Virtual Port Properties screen opens.

4. In the **Idle connection timeout TCP (seconds)** box, type a time-out value for TCP connections. The recommended time-out setting is 10 seconds.
5. In the **Idle connection timeout UDP (seconds)** box, type a time-out value for TCP connections. The recommended time-out setting is 10 seconds.
6. Click **Apply**.

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

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

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

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

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

## Configure the servers

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

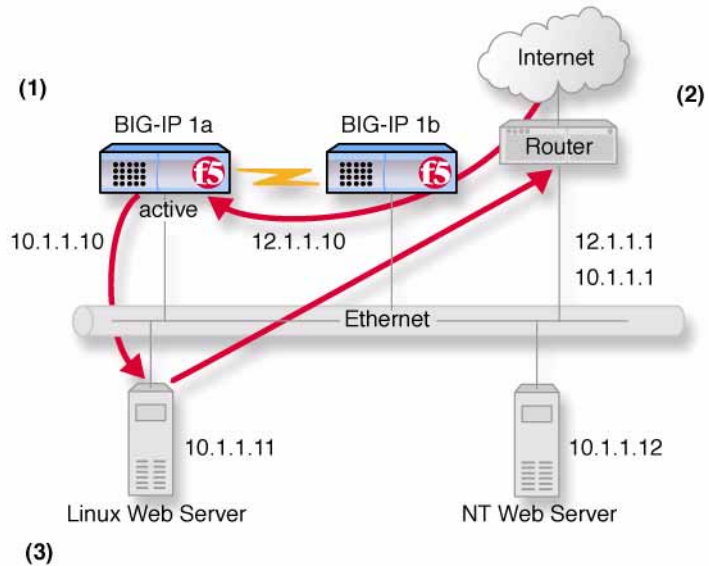


**Figure 7.1** An example *nPath* configuration with more than one virtual server

In this configuration, you must configure the BIG-IP Controllers, the router, and each web server. The following procedures describe each configuration task.

### Use the following commands to configure the BIG-IP Controllers in this example

Use the following commands to configure the BIG-IP Controllers in this example (Figure 7.1, number 1):

```
bigpipe vip 11.1.1.1:80 define 10.1.1.11:80
10.1.1.12:80

bigpipe vip 11.1.1.1:80 translate addr disable
bigpipe vip 11.1.1.2:80 define 10.1.1.11:80
bigpipe vip 11.1.1.2:80 translate addr disable
```

**Set up the following configuration on the router**

Set up the following configuration on the router (Figure 7.1, number 2)

This router has a route to the 11.1.1.0 network through BIGip 12.1.1.10.

**Set up the following configuration on the web servers**

Set up the following configuration on the web servers (Figure 7.1, number 3)

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





# 8

---

## Balancing Traffic Inbound to Enterprise Servers

---

- Introducing firewall load balancing for inbound traffic
- Configuration tasks
- Configuring routing to the internal network
- Creating pools for firewalls and servers
- Creating virtual servers for the firewall sandwich
- Configuring interfaces
- Configuring administrative routing



## Introducing firewall load balancing for inbound traffic

This chapter explains how to load balance a configuration in which users on the Internet request information from a pair of enterprise servers behind the enterprise's set of firewalls. Such a configuration is appropriate for any enterprise that wants to provide information by way of the Internet, but also wants to limit traffic to specific services.

In this configuration, both the set of firewalls and the pair of servers require load balancing. This necessitates two BIG-IP Controller redundant systems:

- ❖ A BIG-IP Controller redundant system on the outside (that is, the side nearest the Internet) of the firewalls, to balance traffic inbound across the firewalls.
- ❖ A BIG-IP Controller redundant system on the inside (that is, the side nearest the enterprise servers) of the firewalls, to balance traffic from the firewalls to the servers.

Collectively, this is known as a firewall sandwich configuration, because the BIG-IP Controller redundant systems sandwich the firewalls. Figure 8.1, following, illustrates this type of scenario, 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.

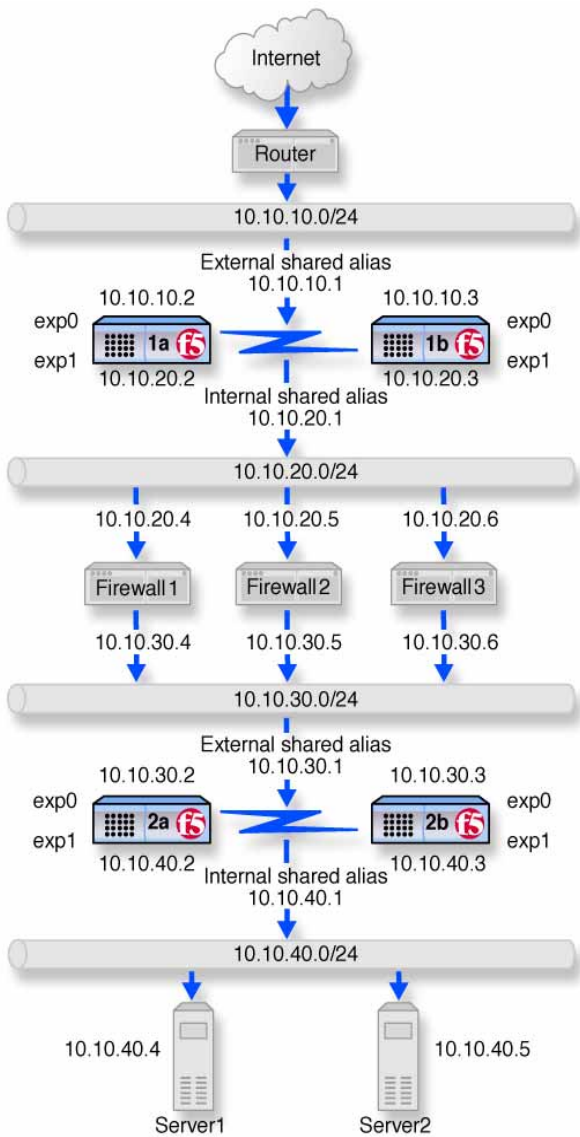


**Figure 8.1** *Balancing inbound traffic*

## Configuration tasks

To load balance traffic to enterprise servers across a set of firewalls using a firewall sandwich, you need to complete the following tasks in order:

- ❖ Configure routing to the internal network.
- ❖ Create load balancing pools for firewalls and servers.
- ❖ Create virtual servers for the firewall sandwich.
- ❖ Configure BIG-IP Controller interfaces for source and destination processing
- ❖ Configure administrative routing

Each of the following sections explains one of the tasks, and shows how you would perform the task in order to implement the example configuration shown in Figure 8.1.

---

### ◆ Note

*The procedures in this chapter detail how to configure a single BIG-IP Controller. In order to complete your configuration, synchronize the configured BIG-IP Controller with the other BIG-IP Controller in your BIG-IP Controller redundant system, as detailed in Synchronizing configurations between controllers, on page 17-41.*

## Configuring routing to the internal network

The external router should include a route for traffic bound for the network that includes the external interfaces of the internal BIG-IP Controller redundant system (that is, the redundant system that load balances the enterprise servers). This route should be by way of the external shared alias of the external BIG-IP Controller redundant system.

For example, in Figure 8.1, the network on which the internal BIG-IP Controller redundant system resides is 10.10.30.0, and the external shared alias of the external BIG-IP Controller redundant system is 10.10.10.1. Thus, a command to configure this routing might be:

```
route add -net 10.10.30.0 -gateway 10.10.10.1
```

The exact syntax of this command depends on the type of router.

## Creating pools for firewalls and servers

To use this configuration, you must create load balancing *pools*. A pool is a group of devices that you want a BIG-IP Controller redundant system to load balance. You need to create three pools:

- ❖ To load balance incoming requests across the external interfaces of your firewalls, you will create a pool that includes these interfaces. You create this pool on the outside BIG-IP Controller redundant system (1a and 1b in Figure 8.1).
- ❖ Requests that pass through the firewalls must be load balanced to the enterprise servers, so you create a pool that includes these servers. You create this pool on the inside BIG-IP Controller redundant system (2a and 2b in Figure 8.1).
- ❖ When a server responds to a request, the outgoing request must return by way of the internal interface of the firewall through which it originally came, so you create a pool that includes the internal interfaces of the firewalls. You create this pool on the inside BIG-IP Controller redundant system (2a and 2b in Figure 8.1).

### Creating a pool for outside firewall interfaces

First, create the pool for the outside addresses of the firewalls on the outside BIG-IP Controller redundant system.

To implement the configuration shown in Figure 8.1, you would create this pool on BIG-IP Controller redundant system 1a and 1b.

**To create the 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. Configure the following attributes on the Add Pool screen.  
For additional information about creating a pool, click the **Help** button.

**Configuration notes**

To create the configuration shown in Figure 8.1:

- Create a pool named **firewalls**.
- Add each firewall from the example, **10.10.20.4**, **10.10.20.5**, and **10.10.20.6**, to the pool. For each firewall you add to the pool, specify port **0**.

**To create the pool from the command line**

Use the **bigpipe pool** command to create the pool:

```
bigpipe pool <pool name> { lb_method <xx> member <Firewall1>:0  
    member <Firewall2>:0 member <Firewall3>:0 }
```

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

- ❖ **<pool name>** is a 1-31 character identifier for the pool.
- ❖ **<Firewall1>**, **<Firewall2>**, and **<Firewall3>** are the IP addresses of the outside interfaces of your respective servers.
- ❖ **lb\_method <xx>** designates the load balancing method. For more information, refer to *Understanding individual load balancing methods* in **BIG-IP Reference Guide**.

In Figure 8.1, for example, the pool for the outside addresses is **firewalls\_outside**, the outside addresses are **10.10.20.4**, **10.10.20.5**, and **10.10.20.6**, and the load balancing method is Round Robin.

Thus, the command would be:

```
bigpipe pool firewalls_outside { lb_method rr member 10.10.20.4:0  
    member 10.10.20.5:0 member 10.10.20.6:0 }
```

## Creating a pool for servers

Finally, create the pool for the nodes that handle requests to your enterprise servers on the inside BIG-IP Controller redundant system. Use the Configuration utility, or the **bigpipe pool** command, as you did to create the firewall pools. Choose a pool name appropriate for this pool.

### To create the 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. Configure the following attributes on the Add Pool screen. For additional information about creating a pool, click the **Help** button.

#### Configuration notes

To create the configuration shown in Figure 8.1:

- Create a pool named **servers**.
- Add the servers **10.10.40.4** and **10.10.40.5** to the pool. For each server, specify port **80**.

### To create the pool from the command line

Use the **bigpipe pool** command to create the pool:

```
bigpipe pool <pool name> { lb_method <xx> member <Server1>:0 member  
  <Server2>:0 }
```

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

- ❖ **<pool name>** is a 1-31 character identifier for the pool.
- ❖ **<Server1>** and **<Server2>** are the IP addresses of your enterprise servers.
- ❖ **lb\_method <xx>** designates the load balancing method. For more information, refer to *Understanding individual load balancing methods* in **Reference Guide**.



To implement the configuration shown in Figure 8.1, you would create this pool on BIG-IP Controller redundant systems 2a and 2b. In this example, the pool for the server addresses is **servers**, the server addresses are 10.10.40.4 and 10.10.40.5, and the load balancing method is Round Robin. Thus, the command to implement this configuration is:

```
bigpipe pool servers { lb_method rr member 10.10.40.4:80 member
10.10.40.5:80 }
```

## Creating a pool for inside firewall interfaces

Next, create a pool for the internal addresses of your firewalls on the inside BIG-IP Controller redundant system. Use the Configuration utility, or the **bigpipe pool** command, as you did to create the pool for the outside firewall addresses. Choose a pool name appropriate for this pool.

### To create the 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. Configure the following attributes on the Add Pool screen. For additional information about creating a pool, click the **Help** button.

#### Configuration notes

To create the configuration shown in Figure 8.1:

- Create a pool named **firewalls**.
- Add each firewall from the example, **10.10.30.4**, **10.10.30.5**, and **10.10.30.6**, to the pool. For each firewall you add to the pool, specify port **0**.

### To create the pool from the command line

Use the **bigpipe pool** command to create the pool:

```
bigpipe pool <pool name> { lb_method <xx> member <Firewall1>:0  
    member <Firewall2>:0 member <Firewall3>:0 }
```

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

- ❖ **<pool name>** is a 1-31 character identifier for the pool.
- ❖ **<Firewall1>**, **<Firewall2>**, and **<Firewall3>** are the IP addresses of the inside interfaces of your respective firewalls.

**lb\_method <xx>** designates the load balancing method. For more information, refer to *Understanding individual load balancing methods* in **Reference Guide**.

To implement the configuration shown in Figure 8.1, you would create this pool on BIG-IP Controller redundant systems 2a and 2b. In this example, the pool for the inside addresses is `firewalls_inside`, the inside addresses are 10.10.30.4, 10.10.30.5, and 10.10.30.6, and the load balancing method is Round Robin. Thus the command to implement this configuration would be:

```
bigpipe pool firewalls_inside { lb_method rr member 10.10.30.4:0  
    member 10.10.30.5:0 member 10.10.30.6:0 }
```

## Creating virtual servers for the firewall sandwich

After you define the pools for the internal and external interfaces of the firewalls, you can define the virtual servers for the BIG-IP Controller redundant systems. You will create a wildcard virtual server on the outside BIG-IP Controllers (1a and 1b in Figure 8.1) to load balance the firewalls, and a standard virtual server on the inside BIG-IP Controllers (2a and 2b in Figure 8.1) to load balance the enterprise servers.

## Creating a wildcard virtual server to load balance the outside firewall interfaces

The outside BIG-IP Controller redundant system load balances inbound connections across the outside interfaces of the firewalls. Therefore, you need to create a wildcard virtual server on this system (1a and 1b in Figure 8.1). The wildcard virtual server should reference the pool you created in *Creating a pool for outside firewall interfaces*, on page 8-4, that contains the outside interfaces of the firewalls.

### 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. Configure the appropriate attributes on the Add Virtual Server screen. For additional information about configuring a virtual server, click the **Help** button.

#### Configuration notes

To create the configuration shown in Figure 8.1:

- Add a virtual server with address **0.0.0.0** and port **0** (this designates a wildcard virtual server).
- In the Pool list, select **firewalls\_outside** (having created the **firewalls\_outside** pool in *Creating a pool for outside firewall interfaces*, on page 8-4).
- Select **none** for interface.

### 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 pool <pool 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 Controller on which you want to create this virtual server.
- ❖ **<pool name>** is the name of the pool you want this virtual server to use.

Repeat this command for each service you want to configure. For example the command to implement the configuration shown in Figure 8.1 would be:

```
bigpipe vip 0.0.0.0:0 none use pool firewalls_outside
```

### Enhancing security

To supplement the security offered by your firewalls, you may want to create a port-specific wildcard virtual server, or a standard virtual server, rather than a wildcard virtual server. In this configuration, a port-specific wildcard virtual server would reduce the number of accessible IP ports from 65,553 ports on all addresses to one port (and one service) on all addresses. A standard virtual server reduces the security exposure to one port on one address.

You can create a port-specific wildcard or standard virtual server to enhance the security of the configuration shown in Figure 8.1 using either the Configuration utility or the command line.

- ❖ To create a port-specific wildcard virtual server using the Configuration utility, follow the instructions in *To create a wildcard virtual server using the Configuration utility*, on page 8-9, using **80** for the **port** attribute.
- ❖ To create a port-specific wildcard virtual server from the command line, use the **bigpipe vip** command as explained in *To create a wildcard virtual server from the command line*, on page 8-9. The command is:

```
bigpipe vip 0.0.0.0:80 none use pool firewalls_outside
```

- ❖ To create a standard virtual server using the Configuration utility, follow the instructions in *To create a wildcard virtual server using the Configuration utility*, on page 8-9. Substitute **10.10.30.9** for the **Address** attribute and **80** for the **port** attribute.

- ❖ To create a standard virtual server from the command line, use the `bigpipe vip` command as explained in *To create a wildcard virtual server from the command line*, on page 8-9. The command is:

```
bigpipe vip 10.10.30.9:80 none use pool firewalls_outside
```

## Creating a standard virtual server to load balance the enterprise servers

After you configure the outside BIG-IP Controller redundant system to handle inbound traffic, configure the inside BIG-IP Controller redundant system (2a and 2b in Figure 8.1) to load balance the enterprise servers.

Use the Configuration utility, or the **bigpipe vip** command, as you did to create the wildcard virtual server for the inside BIG-IP Controller redundant system. Instead of using a wildcard IP address, use a standard IP address and pool appropriate for your network.

### To create a standard 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. Configure the appropriate attributes on the Add Virtual Server screen. For additional information about configuring a virtual server, click the **Help** button.

### Configuration notes

To create the configuration shown in Figure 8.1:

- Add a virtual server with address **10.10.30.9**, and port **80** (this designates a standard wildcard virtual server receiving HTTP traffic).

- In the Pool list, select **servers** (having created the **servers** pool in *Creating a pool for servers*, on page 8-6).

### To create a standard 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 pool <pool 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.

**<pool name>** is the name of the pool you want this virtual server to use.

To use the **bigpipe vip** command to implement the configuration shown in Figure 8.1, type:

```
bigpipe vip 10.10.30.9:80 use pool server_pool
```

## Designating the last hop pool

When a BIG-IP Controller redundant system is accepting connections for virtual servers from more than one firewall, it is typically necessary to return packets through the same firewall from which the connection originated. Returning the data through the originating firewall ensures that:

- ❖ Outbound traffic is balanced across the firewall set.
- ❖ In situations where the firewall is maintaining a connection for a client, packets can be returned to that client.

To configure the firewall sandwich along these lines, use the Configuration utility or the **bigpipe vip** command with the **lasthop** keyword to designate the pool containing the inside interfaces of the firewalls as the last hop pool.

### To configure a last hop pool using the Configuration utility

1. In the navigation pane, click **Virtual Servers**.  
The Virtual Servers screen opens.
2. In the virtual server list, click the virtual server for which you want to set up a last hop pool.  
The Virtual Server properties screen for the virtual server you clicked opens.
3. In the Virtual Server properties screen, select the pool you want to designate as the last hop. For additional information about configuring a last hop pool, click the **Help** button.

#### Configuration notes

To create the configuration shown in Figure 8.1, select **firewalls\_inside** as the last hop pool for the virtual server **10.10.30.9:80**.

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

Use the **bigpipe vip** command:

```
bigpipe vip <virtual server>:<service> lasthop pool <pool 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**.
- ❖ **<pool name>** is the name of the pool you want this virtual server to use.

For example, to implement the configuration shown in Figure 8.1, you would type:

```
bigpipe vip 10.10.30.9:http lasthop pool firewalls_inside
```

## Configuring interfaces

Typically, a BIG-IP Controller redundant system has two interfaces:

- ❖ An external interface, usually set for destination processing.
- ❖ An internal interface, usually set for source processing.

In order for the firewall sandwich configuration to work, you must set all interfaces on the BIG-IP Controller redundant system systems (1a and 1b, and 2a and 2b, in Figure 8.1) to process both source and destination addresses.

Thus, you must enable source processing for the external interfaces and enable destination processing for the internal interfaces.

### To configure source and destination processing 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 8.1, for both **exp0** and **exp1**, make sure the **Enable Source Processing** and the **Enable Destination Processing** check boxes are checked.

### To configure source processing from the command line

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

```
bigpipe interface <interface> source enable
```



where **<interface>** is the identifier for the internal interface of a BIG-IP Controller redundant system.

To implement the configuration shown in Figure 8.1, use the command:

```
bigpipe interface exp0 source enable
```

Repeat this process for each BIG-IP Controller redundant system.

### **To configure destination processing from the command line**

Use the **bigpipe interface** command with the **dest** keyword to enable destination processing for an interface:

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

where **<interface>** is the identifier for the external interface of a BIG-IP Controller redundant system.

For example, to implement the configuration shown in Figure 8.1, use the command:

```
bigpipe interface exp1 dest enable
```

Repeat this process for each BIG-IP Controller redundant system.

## Configuring administrative routing

In order to administer the outside BIG-IP Controller redundant system (BIG-IP Controllers 1a and 1b in Figure 8.1) from the inside BIG-IP Controller redundant system (BIG-IP Controllers 2a and 2b in Figure 8.1), and to administer the inside redundant system from the outside redundant system, you need to create routes between the systems, using the firewalls as gateways.

For example, to implement the configuration shown in Figure 8.1, you would use the following commands on BIG-IP Controllers 1a and 1b:

```
route add -host 10.10.30.1 -gateway 10.10.20.4
```

```
route add -host 10.10.30.2 -gateway 10.10.20.5
```

```
route add -host 10.10.30.3 -gateway 10.10.20.6
```

To complete the configuration, you would use the following commands on BIG-IP Controllers 2a and 2b:

```
route add -host 10.10.20.1 -gateway 10.10.30.4  
route add -host 10.10.20.2 -gateway 10.10.30.5  
route add -host 10.10.20.3 -gateway 10.10.30.6
```

# 9

---

## Balancing Traffic Outbound to the Internet

---

- Introducing firewall load balancing for outbound traffic
- Configuration tasks
- Verifying user network routing
- Creating a pool for the firewalls
- Creating a wildcard virtual server
- Configuring interfaces
- Configuring network address translation on firewalls
- Creating a route for return traffic



## Introducing firewall load balancing for outbound traffic

This chapter explains how to set up a configuration in which users behind a set of firewalls request information from Internet servers. Such a configuration is common for any enterprise that wants to provide users on its intranet access to the Internet, but wants to deny Internet users access to its intranet.

The job of the BIG-IP Controller redundant system is to load balance the firewalls so that they perform as efficiently and effectively as possible. You can also use the BIG-IP Controller redundant system to limit outbound traffic to specific ports, thereby enhancing security.

This type of configuration requires you to configure network address translation (NAT) on your firewalls. Before attempting to implement this configuration, verify that your firewalls can perform NAT.

Figure 9.1, following, illustrates this type of scenario, 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.

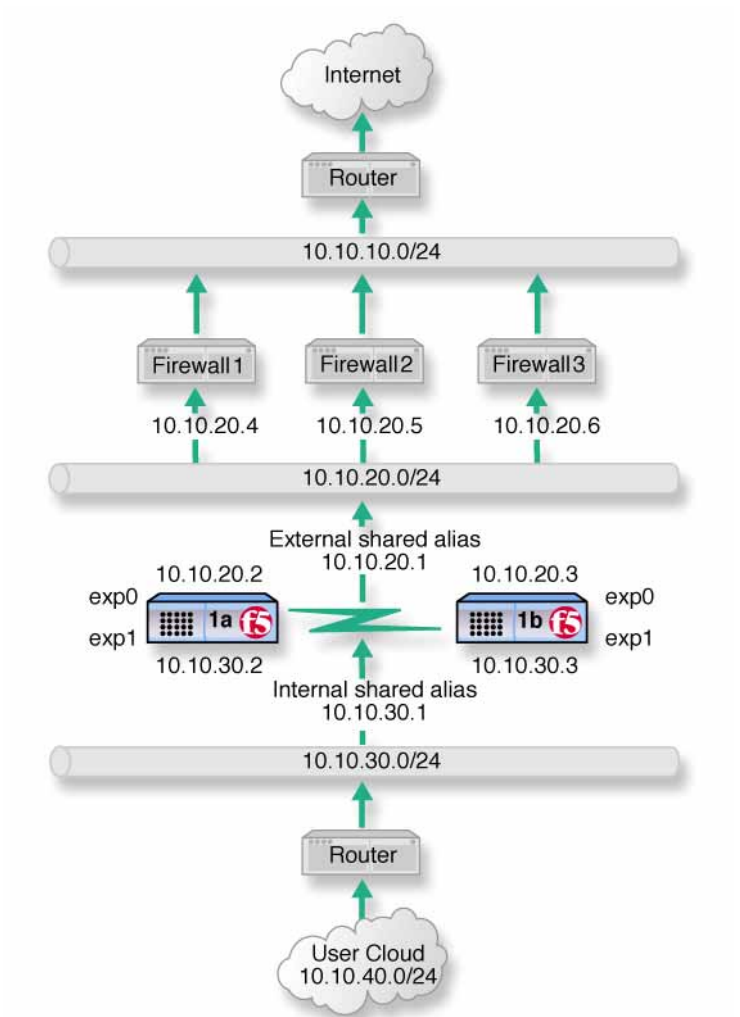


**Figure 9.1** *Balancing outbound traffic*

## Configuration tasks

To configure firewall load balancing for outbound traffic, you need to complete the following tasks in order:

- ❖ Verify user network routing.
- ❖ Create a load balancing pool for the firewalls.
- ❖ Create a wildcard virtual server that references the load balancing pool, so that outbound traffic is load balanced across the firewalls and forwarded to the Internet.
- ❖ Configure interfaces on the BIG-IP Controller redundant system.
- ❖ Configure address translation on the firewalls.
- ❖ Create a route on the firewalls for return traffic to the user network.

Each of the following sections explains one of these tasks, and shows how you perform the task in order to implement a configuration typical for an enterprise that wants to provide users limited access to the Internet, as shown in Figure 9.1.

---

### ◆ Note

*The procedures in this chapter detail how to configure a single BIG-IP Controller. In order to complete your configuration, synchronize the configured BIG-IP Controller with the other BIG-IP Controller in your BIG-IP Controller redundant system, as detailed in *Synchronizing configurations between controllers*, on page 17-41.*

## Verifying user network routing

Verify that the default route on the router between your client network and the BIG-IP Controller redundant system is the internal shared alias for the BIG-IP Controller redundant system. This alias is configured during setup, using the First-Time Boot utility. For more information about this utility, see *Running the First Time Boot*

utility in the **BIG-IP Installation Guide**. For more information about routing, see *Addressing routing issues* in the **BIG-IP Installation Guide**.

To implement the example configuration shown in Figure 9.1, make sure that the router between network 10.10.40.0/24 and the BIG-IP Controller redundant system is configured to point to 10.10.30.1.

---

◆ **Note**

*The router in Figure 9.1 is optional. If the client network is on the same network as the BIG-IP Controller redundant system, then you can use the BIG-IP Controller internal shared alias as a default gateway for the client network.*

## Creating a pool for the firewalls

To use this configuration, you must create a load balancing **pool**. A pool is a group of devices that you want the BIG-IP Controller redundant system to load balance. The devices you want to load balance are your firewalls, so you create a pool containing the inside interfaces of these firewalls. For more information about pools, refer to *Configuring a pool*, on page 17-5.

You can use either the Configuration utility or the **bigpipe pool** command to create the 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 the attributes required for the firewalls you want to add to the pool.  
For additional information about creating a pool, click the **Help** button.



**Configuration notes**

To create the configuration shown in Figure 9.1:

- Create a pool named **firewalls**.
- Add each firewall from the example, **10.10.20.4**, **10.10.20.5**, and **10.10.20.6**, to the pool. For each firewall you add to the pool, specify port **0**, which means this firewall accepts traffic on all ports.

**To create the pool from the command line**

Use the **bigpipe pool** command to create the pool:

```
bigpipe pool <pool name> { lb_method <xx> member <Firewall1>:0  
    member <Firewall2>:0 member <Firewall3>:0 }
```

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

- ❖ **<pool name>** is a 1-31 character identifier for the pool.
- ❖ **<Firewall1>**, **<Firewall2>**, and **<Firewall3>** are the inside IP addresses of your respective firewalls.
- ❖ **lb\_method <xx>** designates the load balancing method. For more information, refer to *Understanding individual load balancing methods* in **BIG-IP Reference Guide**.

In Figure 9.1, for example, the pool for the inside addresses is **firewalls**, the inside addresses are **10.10.20.4**, **10.10.20.5**, and **10.10.20.6**, and the load balancing method is Round Robin. Thus, the command to implement this configuration would be:

```
bigpipe pool firewalls { lb_method rr member 10.10.20.4:0 member  
    10.10.20.5:0 member 10.10.20.6:0 }
```

## Creating a wildcard virtual server

To configure the BIG-IP Controller redundant system for outbound connections, you must create a **wildcard virtual server**; that is, a virtual server that accepts all traffic from the user network, then load balances the traffic across the firewalls.

### 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 required for the virtual server you want to add.  
For additional information about creating a pool, click the **Help** button.

#### Configuration notes

To create the configuration shown in Figure 9.1:

- Add a virtual server with address **0.0.0.0** and port **0** (this designates a wildcard virtual server).
- In the Pool list, select **firewalls** (having created the **firewalls** pool in *Creating a pool for the firewalls*, on page 9-4).

### 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 pool <pool 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.
- ❖ **<pool name>** is the name of the pool you want this virtual server to use.

Repeat this command for each service you want to configure.

To implement the example configuration shown in Figure 9.1, the command is:

```
bigpipe vip 0.0.0.0:0 none use pool firewalls
```

## Enhancing security

To supplement the security offered by your firewalls, you may want to limit the types of traffic that can pass outbound to the Internet. You can use **port-specific wildcard virtual servers** to restrict traffic in this manner. While a standard wildcard virtual server forwards all traffic, a port-specific wildcard virtual server forwards traffic specific to only the specified port. For more information, see *Defining wildcard virtual servers*, on page 17-8.

You can use either the Configuration utility or the **bigpipe vip** command to create the virtual server.

### To create a port-specific wildcard server using the Configuration utility

Follow the instructions detailed in *To create a wildcard virtual server using the Configuration utility*, on page 9-6. In step 3, when you configure the **Port** attribute, choose the port to which you want outgoing traffic to be limited for that virtual server. Complete the rest of the directions as detailed on page 9-6, then repeat the process for any other ports you want to be accessible to outgoing traffic.

To implement the example configuration shown in Figure 9.1, to limit the traffic forwarded to HTTP and FTP, you would complete the steps in *To create a wildcard virtual server using the Configuration utility* three times (that is, once for each of three port-specific virtual servers). For the **Port** attribute, you would enter **80** for the HTTP service, and **20** and **21** for the FTP service.

### To create a port-specific wildcard server from the command line

To create a port-specific wildcard server, use the **bigpipe vip** command as you did in *To create a wildcard virtual server from the command line*, on page 9-6. Substitute the number of the port to which you want to limit access.

To limit the traffic forwarded to HTTP and FTP in the configuration shown in Figure 9.1, you would replace the command in *To create a wildcard virtual server from the command line* with the following commands:

```
bigpipe vip 0.0.0.0:80 none use pool firewall
```

```
bigpipe vip 0.0.0.0:20 none use pool firewall
bigpipe vip 0.0.0.0:21 none use pool firewall
```

## Configuring interfaces

Typically, a BIG-IP Controller has two interfaces:

- ❖ An external interface, typically set for destination processing. In Figure 9.1, the external interface is **exp0**.
- ❖ An internal interface, typically set for source processing. In Figure 9.1, the internal interface is **exp1**.

For this configuration, you set, or reset, interface processing so that the internal interface only processes destination addresses. This enables the wildcard virtual server you created in *Creating a wildcard virtual server*, on page 9-5, to process outbound user requests.

You must also set, or reset, interface processing on the external interface, so that this interface only processes source addresses. When the result of a user request returns from the Internet to the external interface, source processing looks up the connection established by the wildcard virtual server on the destination processing (internal) interface, then sends the result to the user.

### To configure source and destination processing using the Configuration utility

1. In the navigation pane, click **NICs**.
2. The Network Interface Cards screen opens. You can view the current settings for each interface in the Network Interface Card table.

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 9.1:

- For **exp0**, make sure the **Enable Source Processing** check box is checked.
- For **exp1**, make sure the **Enable Destination Processing** check box is checked.

**To configure source processing 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> source [ <enable> | <disable> ]
```

where **<interface>** is the identifier for the internal interface of a BIG-IP Controller.

To implement the example configuration shown in Figure 9.1, you use the commands:

```
bigpipe interface exp0 source enable  
bigpipe interface exp1 source disable
```

**To configure destination processing from the command line**

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

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

where **<interface>** is the identifier for the external interface of a BIG-IP Controller.

For the example configuration shown in Figure 9.1, you would use the command:

```
bigpipe interface exp1 dest enable  
bigpipe interface exp0 dest disable
```

## Configuring network address translation on firewalls

This example configuration includes a single BIG-IP Controller redundant system on the inside of your firewalls. In this respect it is different from other configurations in this guide, which include BIG-IP Controller redundant pairs on both sides of your firewalls. Although the use of one BIG-IP Controller redundant pair simplifies some aspects of the configuration, it also requires additional work from your firewalls.

Specifically, you must configure your firewalls so that they perform network address translation (NAT). This ensures that, in situations where a firewall opens and maintains a connection for a user, packets sent back to that user return to the user through that firewall. Refer to your firewall documentation for instructions on NAT configuration.

## Creating a route for return traffic

For outbound traffic, the virtual server you created in *Creating a wildcard virtual server*, on page 9-5, determines the **next hop**; that is, the next device to which a request should be sent in order to reach its ultimate destination. Therefore, no special routing is required. However, in order to provide a route back to your internal network, you must add a route on your firewalls to this network, using the external shared alias of the BIG-IP Controller redundant system as a gateway.

In Figure 9.1, the external shared alias of the BIG-IP Controller system is 10.10.20.1 and the internal network is 10.10.40.0/24. Thus, a command to configure this routing might be:

```
route add -net 10.10.40.0 -gateway 10.10.20.1
```

The exact syntax of this command depends on the type of router.

# 10

---

## Balancing Two-Way Traffic with Destination Processing

---

- Introducing two-way firewall load balancing with destination processing
- Configuration tasks
- Configuring routing to the internal network
- Creating pools for firewalls and servers
- Creating virtual servers for inbound traffic
- Creating virtual servers for outbound traffic
- Configuring interfaces
- Configuring administrative routing





## Introducing two-way firewall load balancing with destination processing

With a few modifications, you can use the firewall sandwich configuration from Chapter 5, *Introducing firewall load balancing for two-way traffic*, to load balance servers on your intranet as well as two-way traffic across your firewalls. As with the Chapter 5 configuration, traffic is generated in two ways:

- ❖ Users on the Internet request information from a pair of enterprise servers behind the enterprise's set of firewalls, generating inbound traffic.
- ❖ Users behind a set of firewalls request information from Internet servers, generating outbound traffic.

Traffic is again balanced across the firewall set in both the inbound and the outbound directions, necessitating two BIG-IP Controller redundant systems:

- ❖ A BIG-IP Controller redundant system on the outside (that is, the side nearest the Internet) of the firewalls, to balance traffic inbound across the outside interfaces of the firewalls.
- ❖ A BIG-IP Controller redundant system on the inside (that is, the side nearest the enterprise servers) of the firewalls, to balance traffic outbound across the inside interfaces of the firewalls.

However, in this configuration, this redundant system also load balances servers on the intranet.

Figure 10.1, following, illustrates this type of scenario, 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.

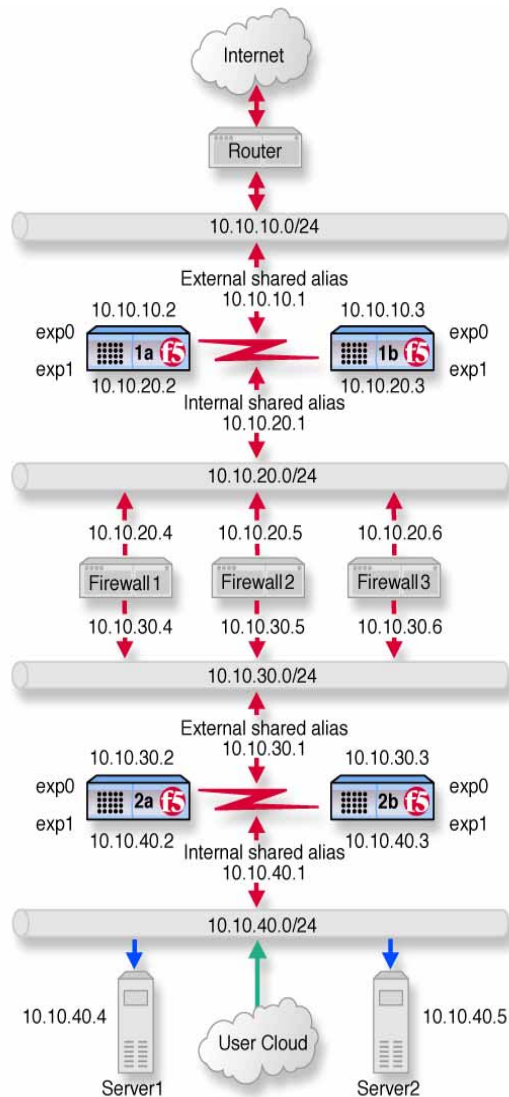


**Figure 10.1** Two-way traffic with destination processing

## Configuration tasks

To load balance enterprise servers as well as two-way traffic across a set of firewalls using a firewall sandwich configuration, you need to complete all the following tasks in order:

- ❖ Configure routing to the internal network.
- ❖ Create pools for firewalls and servers.
- ❖ Create virtual servers for inbound traffic.
- ❖ Create virtual servers for outbound traffic.
- ❖ Configure BIG-IP Controller interfaces for source and destination processing.
- ❖ Configure administrative routing.

This chapter explains how to set up this configuration, using the sample IP addresses and device names in Figure 10.1 as an example.

---

### ◆ Note

*The procedures in this chapter detail how to configure a single BIG-IP Controller. In order to complete your configuration, synchronize the configured BIG-IP Controller with the other BIG-IP Controller in your BIG-IP Controller redundant system, as detailed in Synchronizing configurations between controllers, on page 17-41.*

## Configuring routing to the internal network

The external router should route traffic bound for the network that includes your intranet by way of the external shared alias of the external BIG-IP Controller redundant system.

In Figure 10.1, the internal BIG-IP Controllers are 2a and 2b, the network is 10.10.30.0, and the external shared alias of the external BIG-IP Controller redundant system is 10.10.10.1. Thus, a command to configure this routing might be:

**Route add -net 10.10.30.0 -gateway 10.10.10.1**

The exact syntax of this command depends on the type of router.

## Creating pools for firewalls and servers

To use this configuration, you must create load balancing **pools**. A pool is a group of devices that you want a BIG-IP Controller redundant system to load balance. You will create three pools:

- ❖ To load balance incoming requests across the external interfaces of your firewalls, you create a pool that includes these external interfaces.
- ❖ Requests that pass through the firewalls must be load balanced to the enterprise servers, so you create a pool that includes these enterprise servers.
- ❖ When the servers respond to requests, the outgoing requests must be balanced across the internal interfaces of your firewalls, so you create a pool that includes these internal interfaces.

### Creating a pool for outside firewall interfaces

When using this configuration, you first create the pool for the outside addresses of the firewalls on the outside BIG-IP Controller redundant system.

To implement the configuration shown in Figure 10.1, you create this pool on BIG-IP Controllers 1a and 1b.

#### To create the 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. Configure the following attributes on the Add Pool screen. For additional information about creating a pool, click the **Help** button.

**Configuration notes**

To create the configuration shown in Figure 10.1:

- Create a pool named **firewalls\_outside**.
- Add each firewall from the example, **10.10.20.4**, **10.10.20.5**, and **10.10.20.6**, to the pool. For each firewall you add to the pool, specify port **0**.

**To create the pool from the command line**

Use the **bigpipe pool** command to create the pool:

```
bigpipe pool <pool name> { lb_method <xx> member <Firewall1>:0  
    member <Firewall2>:0 member <Firewall3>:0 }
```

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

- ❖ **<pool name>** is a 1-31 character identifier for the pool.
- ❖ **<Firewall1>**, **<Firewall2>**, and **<Firewall3>** are the IP addresses of the outside interfaces of your respective firewalls.
- ❖ **lb\_method <xx>** designates the load balancing method. For more information, refer to *Understanding individual load balancing methods* in the **BIG-IP Reference Guide**.

In Figure 10.1, for example, the pool for the outside addresses is **firewalls\_outside**, the outside addresses are **10.10.20.4**, **10.10.20.5**, and **10.10.20.6**, and the load balancing method is Round Robin. Thus, the command would be:

```
bigpipe pool firewalls_outside { lb_method rr member 10.10.20.4:0  
    member 10.10.20.5:0 member 10.10.20.6:0 }
```

## Creating a pool for inside firewall interfaces

Next, create a pool for the internal addresses of your firewalls on the inside BIG-IP Controllers redundant system (2a and 2b in Figure 10.1). Use the Configuration utility, or the **bigpipe pool** command, as you did to create the pool for the outside firewall addresses. Choose a pool name appropriate for this pool.

**To create the 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. Configure the following attributes on the Add Pool screen.  
For additional information about creating a pool, click the **Help** button.

**Configuration notes**

To create the configuration shown in Figure 10.1:

- Create a pool named **firewalls\_inside**.
- Add each firewall from the example, **10.10.30.4**, **10.10.30.5**, and **10.10.30.6**, to the pool. For each firewall you add to the pool, specify port **0**.

**To create the pool from the command line**

Use the **bigpipe pool** command to create the pool:

```
bigpipe pool <pool name> { lb_method <xx> member <Firewall1>:0  
  member <Firewall2>:0 member <Firewall3>:0 }
```

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

- ❖ **<pool name>** is a 1-31 character identifier for the pool.
- ❖ **<Firewall1>**, **<Firewall2>**, and **<Firewall3>** are the IP addresses of the inside interfaces of your respective firewalls.
- ❖ **lb\_method <xx>** designates the load balancing method. For more information, refer to *Understanding individual load balancing methods* in the **BIG-IP Reference Guide**.

To implement the configuration shown in Figure 10.1, you create this pool on BIG-IP Controller redundant systems 2a and 2b. In this example, the pool for the inside addresses is **firewalls\_inside**, the inside addresses are **10.10.30.4**, **10.10.30.5**, and **10.10.30.6**, and the load balancing method is Round Robin. Thus the command to implement this configuration would be:

```
bigpipe pool firewalls_inside { lb_method rr member 10.10.30.4:0  
    member 10.10.30.5:0 member 10.10.30.6:0 }
```

## Creating a pool for servers

Finally, create the pool for the nodes that handle requests to your enterprise servers on the inside BIG-IP Controller redundant system. Use the Configuration utility, or the **bigpipe pool** command, as you did to create the firewall pools. Choose a pool name appropriate for this pool.

### To create the 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. Configure the following attributes on the Add Pool screen.  
For additional information about creating a pool, click the **Help** button.

#### Configuration notes

To create the configuration shown in Figure 10.1:

- Create a pool named **servers**.
- Add the servers **10.10.40.4** and **10.10.40.5** to the pool.  
For each server, specify port **0**.

### To create the pool from the command line

Use the **bigpipe pool** command to create the pool:

```
bigpipe pool <pool name> { lb_method <xx> member <Server1>:0 member  
    <Server2>:0 }
```

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

- ❖ **<pool name>** is a 1-31 character identifier for the pool.
- ❖ **<Server1>** and **<Server2>** are the IP addresses of your enterprise servers.

- ❖ **lb\_method <xx>** designates the load balancing method. For more information, refer to *Understanding individual load balancing methods* in **BIG-IP Reference Guide**.

To implement the configuration shown in Figure 10.1, you create this pool on BIG-IP Controllers 2a and 2b. In this example, the pool for the server addresses is **servers**, the server addresses are **10.10.40.4** and **10.10.40.5** and the load balancing method is Round Robin. Thus, the command to implement this configuration would be:

```
bigpipe pool servers { lb_method rr member 10.10.40.4:80 member  
10.10.40.5:80 }
```

## Creating virtual servers for inbound traffic

After you define the pools for the outer interfaces of the firewalls, you can define virtual servers on the BIG-IP Controller redundant systems to load balance inbound connections. To do this you:

- ❖ Create a **network virtual server** on the outside BIG-IP Controllers (1a and 1b in Figure 10.1) to load balance the firewalls. A network virtual server is a virtual server that handles a whole network range, instead of just one IP address.
- ❖ Create a standard virtual server on the inside BIG-IP Controllers (2a and 2b in Figure 10.1) to load balance the enterprise servers.

### Creating a network virtual server to load balance the firewalls

Because the outside BIG-IP Controller redundant system load balances inbound connections across the outside interfaces of the firewalls, you need to create a virtual server on that system (1a and 1b in Figure 10.1). This virtual server will reference the pool you created in *Creating a pool for outside firewall interfaces*, on page 10-4 that contains these outside firewall interfaces.

In order to accommodate potential multiple virtual servers for your enterprise servers, create a **network virtual server**. A network virtual server is a virtual server that handles a whole network range,



instead of just one IP address. For example, in Figure 10.1, the virtual server 10.10.30.0 load balances traffic across the firewall set to all virtual servers on the 10.10.30.0/24 network.

### To create a network 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. Configure the appropriate attributes on the Add Virtual Server screen.  
For additional information about configuring a virtual server, click the **Help** button.

#### Configuration notes

To create the configuration shown in Figure 10.1:

- Add a virtual server with address **10.10.30.0** and port **0** (this designates a wildcard virtual server).
- In the Pool list, select **firewalls\_outside** (having created the **firewalls\_outside** pool in *Creating a pool for outside firewall interfaces*, on page 10-4).

### To create a network 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 pool <pool 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.

❖ **<pool name>** is the name of the pool you want this virtual server to use.

Repeat this command for each service you want to configure. To implement the configuration shown in Figure 10.1, you use the command

```
bigpipe vip 10.10.30.0 none use pool firewall_outsides
```

### Enhancing security

To supplement the security offered by your firewalls, you may want to create a standard virtual server rather than a network virtual server. For example, in the configuration shown in Figure 10.1, you really only need a virtual server for 10.10.30.9. In this configuration, using a standard virtual server would reduce the number of accessible addresses from 254 to one.

- ❖ To create a standard virtual server to enhance the security of the configuration shown in Figure 10.1 using the Configuration utility, follow the instructions in *To create a network virtual server using the Configuration utility*, on page 10-9. Substitute **10.10.30.9** for the **Address** attribute, and **80** for **Port**.
- ❖ To create a wildcard port virtual server to enhance the security of the configuration shown in Figure 10.1 from the command line, use the **bigpipe vip** command as explained in *To create a network virtual server from the command line*, on page 10-9. The command would be:

```
bigpipe vip 10.10.30.9:80 none use pool firewall_outsides
```

## Creating a standard virtual server to load balance intranet servers

After you configure the outside BIG-IP Controller redundant system to handle inbound traffic, configure the inside BIG-IP Controller redundant system (2a and 2b in Figure 10.1) to load balance the enterprise servers.

Use the Configuration utility, or the **bigpipe vip** command, as you did to create the wildcard virtual server for the inside BIG-IP Controller redundant system. Instead of using a wildcard IP address, use a standard IP address and pool appropriate for your network.

### To create a standard 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. Configure the appropriate attributes on the Add Virtual Server screen.  
For additional information about configuring a virtual server, click the **Help** button.

#### Configuration notes

To create the configuration shown in Figure 10.1:

- Add a virtual server with address **10.10.30.9**, port **80**, and interface **exp0**.
  - In the Pool list, select **firewalls\_outside** (having created the **firewalls\_outside** pool in *Creating a pool for servers*, on page 10-7).
4. In the Pool list, select the pool you want to apply to the virtual server. For example, to implement the configuration shown in Figure 10.1, you would choose **servers** (having created the **servers** pool in *Creating a pool for servers*, on page 10-7).

### To create a standard 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 pool <pool 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.
- ❖ **<pool name>** is the name of the pool you want this virtual server to use.

For example, to use the **bigpipe vip** command to implement the configuration shown in Figure 10.1, the command would be:

```
bigpipe vip 10.10.30.9:80 use pool server_pool
```

## Designating a last hop pool for inbound traffic

When a BIG-IP Controller redundant system is accepting inbound connections for virtual servers from more than one firewall, it is typically necessary to return outbound packets through the same firewall they came through inbound. Returning the data through the originating firewall provides two potential benefits:

- ❖ It balances the outbound load across the firewall set.
- ❖ It guarantees, in situations where the firewall is maintaining a connection for a client, that packets can be returned to that client.

To configure your firewall sandwich along these lines, use the Configuration utility or the **bigpipe vip** command with the **lasthop** keyword to designate the pool containing the inside interfaces of the firewalls as the last hop pool.

### To configure a last hop pool using the Configuration utility

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

3. In the Virtual Server properties screen, select the pool you want to designate as the last hop.

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

#### Configuration notes

To create the configuration shown in Figure 10.1, select **firewalls\_inside** as the last hop pool for the virtual server **10.10.30.9:80** on interface **exp0**.

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

Use the **bigpipe vip** command:

```
bigpipe vip <virtual server>:<service> <interface> lasthop pool  
          <pool 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 network interface card on which you want this virtual server to process traffic.
- ❖ **<pool name>** is the name of the pool you want this virtual server to use.

For example, to implement the configuration shown in Figure 10.1, you would type:

```
bigpipe vip 10.10.30.9:80 exp0 lasthop pool firewall_insides
```

## Creating virtual servers for outbound traffic

After you define the pools for the internal interfaces of the firewalls, you can define virtual servers on the BIG-IP Controller redundant systems to load balance outbound connections. To do this you:

- ❖ Create a wildcard virtual server on the inside BIG-IP Controllers (2a and 2b in Figure 10.1) to balance traffic outbound to the firewalls.
- ❖ Create a **forwarding wildcard virtual server** on the outside BIG-IP Controllers (1a and 1b in Figure 10.1) to forward traffic to the Internet. A forwarding virtual server is a virtual server that merely forwards traffic, rather than balancing it across nodes.

## Creating a wildcard virtual server for balancing traffic to the firewalls

To configure the inside BIG-IP Controller redundant system (2a and 2b in Figure 10.1) for outbound connections, create a wildcard virtual server that accepts all traffic from the internal network, then load balances the traffic through the firewalls.

### 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. Configure the following attributes on the Add Virtual Server screen.  
For additional information about configuring a virtual server, click the **Help** button.

#### Configuration notes

To create the configuration shown in Figure 10.1, select **firewalls\_inside** as the last hop pool for the virtual server **0.0.0.0:0**.

### 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 pool <pool name>
```

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

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

To use the **bigpipe vip** command to implement the configuration shown in Figure 10.1, you type:

```
bigpipe vip 0.0.0.0:0 none use pool firewall_insidess
```

## Creating a forwarding wildcard virtual server to forward traffic to the Internet

After the appropriate firewall has processed outbound traffic, you want the outside BIG-IP Controller redundant system (1a and 1b in Figure 10.1) to forward the traffic to the Internet. To accomplish this, create a wildcard virtual server as you did in *Creating a wildcard virtual server for balancing traffic to the firewalls*, on page 10-14, using either the Configuration utility or the command line.

- ❖ If you use the Configuration utility, use the address and port **0.0.0.0:0**, and select **Forwarding** in the **Resources** section.
- ❖ From the command line, to implement the configuration shown in Figure 10.1, you type:

```
bigpipe vip 0.0.0.0:0 none forward
```

## Enhancing security

In some situations, you may want to limit the types of traffic that can pass outbound to the Internet. You can use **port-specific wildcard virtual servers** to restrict traffic in this manner. While a standard wildcard virtual server forwards all traffic, a port-specific wildcard virtual server forwards traffic specific to only the specified port. For more information, see *Defining wildcard virtual servers*, on page 17-8.

### **To create a port-specific wildcard server using the Configuration utility**

Follow the instructions detailed in *To create a wildcard virtual server using the Configuration utility*, on page 10-14.

- ❖ In step 3, when you configure the **Port** attribute, choose the port to which you want outgoing traffic to be limited for that virtual server.
- ❖ Complete the rest of the steps as detailed on page 10-14, then repeat the process for any other ports you want to be accessible to outgoing traffic.

For example, to implement the configuration shown in Figure 10.1, to limit the traffic forwarded to HTTP and FTP, you would follow the instructions in *To create a wildcard virtual server using the Configuration utility* three times. That is, once for each of three port-specific virtual servers, entering **80**, **20**, and **21** for the **Port** attribute.

### **To create a port-specific wildcard server from the command line**

To create a port-specific wildcard server, use the **bigpipe vip** command as you did in *To create a network virtual server from the command line*, on page 10-9. For the sample port number, substitute the number of the port to which you want to limit access.

For example, in the configuration shown in Figure 10.1, to limit the traffic forwarded to HTTP and FTP, you replace the command in the preceding section with the following commands:

```
bigpipe vip 0.0.0.0:80 none use pool firewall  
bigpipe vip 0.0.0.0:20 none use pool firewall  
bigpipe vip 0.0.0.0:21 none use pool firewall
```



## Designating a last hop pool for outbound traffic

Just as you used a last hop pool to balance and maintain inbound connections in *Designating a last hop pool for inbound traffic*, on page 10-12, you now create a last hop pool for outbound traffic for the same purposes. Create the pool on the outside BIG-IP Controller redundant system (1a and 1b in Figure 10.1).

Create the last hop pool for outbound traffic as you did for inbound traffic, using either the Configuration utility or the command line.

- ❖ If you use the Configuration utility, use the address and port **0.0.0.0:0** and select **firewall\_outsides** in the **Last Hop Pool** section.
- ❖ From the command line, to implement the configuration shown in Figure 10.1, you would type:

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

## Configuring interfaces

Typically, a BIG-IP Controller redundant system has two interfaces:

- ❖ An external interface, usually set for destination processing
- ❖ An internal interface, usually set for source processing

In order for the firewall sandwich configuration to work, you must set all interfaces on the BIG-IP Controller redundant system (1a and 1b, and 2a and 2b, in Figure 10.1) to process both source and destination addresses.

Thus, you must enable source processing for the external interfaces and disable destination processing for the internal interfaces.

### To configure source and destination processing 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 10.1, for both **exp0** and **exp1**, make sure the **Enable Source Processing** and the **Enable Destination Processing** check boxes are checked.

### To configure source processing from the command line

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

**bigpipe interface <interface> source enable**

where **<interface>** is the identifier for the internal interface of a BIG-IP Controller redundant system.

To implement the configuration shown in Figure 10.1, you would use the command:

**bigpipe interface exp0 source enable**

Repeat this process for each BIG-IP Controller redundant system.

### To configure destination processing from the command line

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

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

where **<interface>** is the identifier for the external interface of a BIG-IP Controller redundant system.

To implement the configuration shown in Figure 10.1, you would use the command:

```
bigpipe interface exp1 dest enable
```

Repeat this process for each BIG-IP Controller.

## Configuring administrative routing

In order to administer the outside BIG-IP Controller redundant system (BIG-IP Controllers 1a and 1b in Figure 10.1) from the inside BIG-IP Controller redundant system (BIG-IP Controllers 2a and 2b in Figure 10.1), and to administer the inside redundant system from the outside redundant system, you need to create routes between the systems, using the firewalls as gateways.

To implement the configuration shown in Figure 10.1, you would use the following commands on BIG-IP Controllers 1a and 1b:

```
route add -host 10.10.30.1 -gateway 10.10.20.4
```

```
route add -host 10.10.30.2 -gateway 10.10.20.5
```

```
route add -host 10.10.30.3 -gateway 10.10.20.6
```

To complete the configuration, you would use the following commands on BIG-IP Controllers 2a and 2b:

```
route add -host 10.10.20.1 -gateway 10.10.30.4
```

```
route add -host 10.10.20.2 -gateway 10.10.30.5
```

```
route add -host 10.10.20.3 -gateway 10.10.30.6
```



# II

---

## Balancing Two-Way Traffic

---

- Introducing firewall load balancing for two-way traffic
- Configuration tasks
- Configuring routing to the internal network
- Creating pools for the firewalls
- Creating virtual servers for inbound traffic
- Creating virtual servers for outbound traffic
- Configuring interfaces
- Configuring administrative routing



## Introducing firewall load balancing for two-way traffic

This chapter explains how to load balance a configuration in which:

- ❖ Users on the Internet request information from an intranet behind the enterprise's set of firewalls, generating inbound traffic.
- ❖ Users behind a set of firewalls request information from Internet servers, generating outbound traffic.

Traffic must therefore be balanced across the firewall set in both the inbound and the outbound directions. This necessitates two BIG-IP Controller redundant systems:

- ❖ A BIG-IP Controller redundant system on the outside (that is, the side nearest the Internet) of the firewalls, to balance traffic inbound across the outside interfaces of the firewalls.
- ❖ A BIG-IP Controller redundant system on the inside (that is, the side nearest the enterprise servers) of the firewalls, to balance traffic outbound across the inside interfaces of the firewalls.

Like the configuration in Chapter 8, *Balancing Traffic Inbound to Enterprise Servers*, this is known as a firewall sandwich configuration, because the BIG-IP Controller redundant systems sandwich the firewalls.

Figure 11.1, following, illustrates this type of scenario, 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.

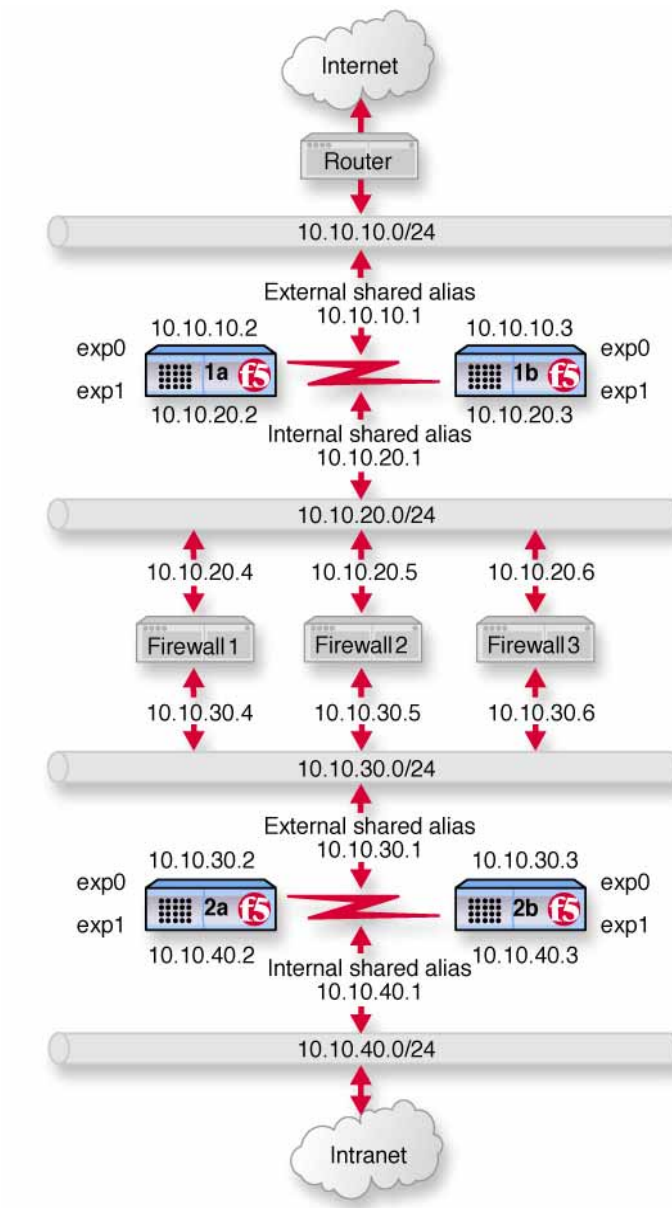


Figure 11.1 Balancing two-way traffic



## Configuration tasks

To load balance two-way traffic across a set of firewalls using a firewall sandwich configuration, you need to complete the following tasks in order.

- ❖ Configure routing to the internal network.
- ❖ Create pools for the firewalls.
- ❖ Create virtual servers for inbound traffic.
- ❖ Create virtual servers for outbound traffic.
- ❖ Configure BIG-IP Controller interfaces for source and destination processing.
- ❖ Configure administrative routing.

Each of the following sections explains one of the tasks, and shows how you would perform the task in order to implement the example configuration shown in Figure 11.1.

---

### ◆ Note

*The procedures in this chapter detail how to configure a single BIG-IP Controller. In order to complete your configuration, synchronize the configured BIG-IP Controller with the other BIG-IP Controller in your BIG-IP Controller redundant system, as detailed in Synchronizing configurations between controllers, on page 17-41.*

## Configuring routing to the internal network

The external router should route traffic bound for the network that includes your intranet by way of the external shared alias of the external BIG-IP Controller redundant system.

In Figure 11.1, the intranet resides on the network 10.10.40.0, and the external shared alias is 10.10.10.1. Thus, a command to configure this routing might be:

```
route add -net 10.10.40.0 -gateway 10.10.10.1
```

The exact syntax of this command depends on the type of router.

## Creating pools for the firewalls

To use this configuration, you must create load balancing **pools**. A pool is a group of devices that you want a BIG-IP Controller redundant system to load balance. You will create two pools:

- ❖ To load balance incoming requests across the outside interfaces of your firewalls, you create a pool that includes these outside interfaces.
- ❖ To load balance outgoing requests across the inside interfaces of your firewalls, you create a pool that includes these inside interfaces.

### Creating a pool for outside firewall interfaces

When using this configuration, you first create the pool for the outside interfaces of the firewalls on the outside BIG-IP Controller redundant system.

To implement the configuration shown in Figure 11.1, you create this pool on BIG-IP Controllers 1a and 1b.

#### To create the 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. Configure the following attributes on the Add Pool screen. For additional information about creating a pool, click the **Help** button.

#### Configuration notes

To create the configuration shown in Figure 11.1:

- Create a pool named **firewalls\_outside**.

- Add each firewall from the example, **10.10.20.4**, **10.10.20.5**, and **10.10.20.6**, to the pool. For each firewall you add to the pool, specify port **0**.

### To create the pool from the command line

Use the **bigpipe pool** command to create the pool:

```
bigpipe pool <pool name> { lb_mode <xx> member <Firewall1>:0 member
  <Firewall2>:0 member <Firewall3>:0 }
```

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

- ❖ **<pool name>** is a 1-31 character identifier for the pool.
- ❖ **<Firewall1>**, **<Firewall2>**, and **<Firewall3>** are the IP addresses of the outside interfaces of your respective firewalls.
- ❖ **lb\_mode <xx>** designates the load balancing method. For more information, refer to *Understanding individual load balancing methods* in **BIG-IP Reference Guide**.

In Figure 11.1, for example, the pool for the outside addresses is **firewalls\_outside**, the outside addresses are **10.10.20.4**, **10.10.20.5**, and **10.10.20.6**, and the load balancing method is Round Robin. Thus, the command would be:

```
bigpipe pool firewalls_outside { lb_mode rr member 10.10.20.4:0
  member 10.10.20.5:0 member 10.10.20.6:0 }
```

## Creating a pool for inside firewall interfaces

Next, create a pool for the inside interfaces of your firewalls on the inside BIG-IP Controller redundant system (2a and 2b in Figure 11.1). Use the Configuration utility, or the **bigpipe pool** command, as you did to create the pool for the outside firewall addresses. Choose a pool name appropriate for this pool.

### To create the 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. Configure the following attributes on the Add Pool screen.  
For additional information about creating a pool, click the **Help** button.

### Configuration notes

To create the configuration shown in Figure 11.1:

- Create a pool named **firewalls\_inside**.
- Add each firewall from the example, **10.10.30.4**, **10.10.30.5**, and **10.10.30.6**, to the pool. For each firewall you add to the pool, specify port **0**.

### To create the pool from the command line

Use the **bigpipe pool** command to create the pool:

```
bigpipe pool <pool name> { lb_mode <xx> member <Firewall1>:0 member  
  <Firewall2>:0 member <Firewall3>:0 }
```

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

- ❖ **<pool name>** is a 1-31 character identifier for the pool.
- ❖ **<Firewall1>**, **<Firewall2>**, and **<Firewall3>** are the IP addresses of the inside interfaces of your respective firewalls.
- ❖ **lb\_mode <xx>** designates the load balancing method. For more information, refer to *Understanding individual load balancing methods* in **BIG-IP Reference Guide**.

To implement the configuration shown in Figure 11.1, you would create this pool on BIG-IP Controllers 2a and 2b. In this example, the pool for the inside addresses is **firewalls\_inside**, the inside addresses are **10.10.30.4**, **10.10.30.5**, and **10.10.30.6**, and the load balancing method is Round Robin. Thus the command to implement this configuration would be:

```
bigpipe pool firewalls_inside { lb_mode rr member 10.10.30.4:0  
  member 10.10.30.5:0 member 10.10.30.6:0 }
```

## Creating virtual servers for inbound traffic

After you define pools for the outer interfaces of the firewalls, you can define virtual servers on the BIG-IP Controller redundant systems to load balance inbound connections. To do this you:

- ❖ Create a **network virtual server** on the outside BIG-IP Controllers (1a and 1b in Figure 11.1) to load balance the firewalls. A network virtual server is a virtual server that handles a whole network range, instead of just one IP address.
- ❖ Create a **forwarding virtual server** on the inside BIG-IP Controllers (2a and 2b in Figure 11.1) to load balance the enterprise servers. A forwarding virtual server is a virtual server that merely forwards traffic, rather than balancing it across nodes.

### Creating a network virtual server

Because the outside BIG-IP Controller redundant system load balances inbound connections across the outside interfaces of the firewalls, you need to create a virtual server on that system (1a and 1b in Figure 11.1). This virtual server will reference the pool you created in *Creating a pool for outside firewall interfaces*, on page 11-4, that contains these outside firewall interfaces.

In order to accommodate potential multiple virtual servers on your intranet, create a **network virtual server**. A network virtual server is a virtual server that handles a whole network range, instead of just one IP address. In Figure 11.1, the virtual server 10.10.40.0 load balances traffic across the firewall set to all virtual servers on the 10.10.40.0/24 network.

#### To create a network 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. Configure the appropriate attributes on the Add Virtual Server screen.

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

### Configuration notes

To create the configuration shown in Figure 11.1:

- Add a virtual server with address **10.10.40.0**, port **0** (this designates a network virtual server), and the Class C netmask **255.255.255.0**.
- In the Pool list, select **firewalls\_outside** (having created the **firewalls\_outside** pool in *Creating a pool for outside firewall interfaces*, on page 11-4).

### To create a network 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 pool <pool 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 Controller on which you want to create this virtual server.
- ❖ **<pool name>** is the name of the pool you want this virtual server to use.

Repeat this command for each service you want to configure. To implement the configuration shown in Figure 11.1, you use the command

```
bigpipe vip 10.10.40.0:0 none netmask 255.255.255.0 use pool firewalls_outside
```

## Enhancing security

In some situations, you may want to limit the types of traffic that can pass inbound to your intranet. You can use **port-specific network virtual servers** to restrict traffic in this manner. While a standard network virtual server processes all traffic for a network, a port-specific network virtual server processes traffic specific to only the specified port.

### To create a port-specific wildcard server using the Configuration utility

Follow the instructions detailed in *To create a network virtual server using the Configuration utility*, on page 11-7. When you configure the **Port** attribute, choose the port to which you want to limit the outgoing traffic for that virtual server. Repeat the process for any other ports you want accessible to incoming traffic.

For example, to implement the configuration shown in 11.1 and limit the traffic forwarded to HTTP and FTP, you complete the instructions in *To create a network virtual server using the Configuration utility* three times. That is, once for each of three port-specific virtual servers, entering **80**, **20**, and **21** for the **Port** attribute.

### To create a port-specific wildcard server from the command line

To create a port-specific wildcard server, use the `bigpipe vip` command as you did in *To create a network virtual server from the command line*, on page 11-8. Substitute the number of the port to which you want to limit access.

For the configuration shown in Figure 11.1, to limit the traffic forwarded to HTTP and FTP, you would replace the command in *To create a network virtual server from the command line* with the following commands:

```
bigpipe vip 10.10.40.0:80 none use pool firewalls_outside
bigpipe vip 10.10.40.0:20 none use pool firewalls_outside
bigpipe vip 10.10.40.0:21 none use pool firewalls_outside
```

## Creating a forwarding virtual server

Because the inside BIG-IP Controller redundant system forwards inbound connections to the intranet, you need to create a forwarding virtual server on that system (2a and 2b in Figure 11.1). This virtual server will forward traffic to network 10.10.40.0/24.

### To create a forwarding 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. Configure the appropriate attributes on the Add Virtual Server screen.  
For additional information about configuring a network virtual server, click the **Help** button.

#### Configuration notes

To create the configuration shown in Figure 11.1:

- Add a virtual server with address **10.10.40.0**, port **0**, interface **exp0** and the Class C netmask **255.255.255.0**.
- In the Resources, click the **Forwarding** button.

### To create a network 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> forward
```

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 Controller on which you want to create this virtual server.



❖ **<pool name>** is the name of the pool you want this virtual server to use.

Repeat this command for each service you want to configure. For example, to implement the configuration shown in Figure 11.1, you use the command

```
bigpipe vip 10.10.40.0:0 exp0 forward
```

## Designating a last hop pool for inbound traffic

When a BIG-IP Controller redundant system is accepting inbound connections for virtual servers from more than one firewall, it is typically necessary to return outbound packets through the same firewall they came through inbound. Returning the data through the originating firewall has two potential benefits:

- ❖ It balances the resulting outbound traffic across the firewall set.
- ❖ It guarantees, in situations where the firewall is maintaining a connection for a client, that packets can be returned to the originating client.

To configure the firewall sandwich along these lines, use the Configuration utility or the **bigpipe vip** command with the **lasthop** keyword to designate the pool containing the inside interfaces of the firewalls as the last hop pool. Designate this pool on the inside BIG-IP Controllers (2a and 2b in Figure 11.1).

### To configure a last hop pool using the Configuration utility

1. In the navigation pane, click **Virtual Servers**.  
The Virtual Servers screen opens.
2. In the virtual server list, click the virtual server for which you want to set up a last hop pool.  
For example, to implement the configuration shown in Figure 11.1, you would select **10.10.40.0:0**.  
  
The properties screen for the virtual server you clicked opens.
3. Click the **Apply** button.

4. In the Virtual Server properties screen, select the pool you want to designate as the last hop.

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

#### **Configuration notes**

To create the configuration shown in Figure 11.1, select **firewalls\_inside** as the last hop pool for the virtual server **10.10.40.0:0**.

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

Use the **bigpipe vip** command:

```
bigpipe vip <virtual server>:<service> lasthop pool <pool 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**.
- ❖ **<pool name>** is the name of the pool you want this virtual server to use.

For example, to implement the configuration shown in Figure 11.1, you would type:

```
bigpipe vip 10.10.40.0:0 none lasthop pool firewalls_inside
```

## **Creating virtual servers for outbound traffic**

After you define a pool for the internal interfaces of your firewalls, you can define virtual servers on the inside BIG-IP Controller redundant system to load balance outbound connections. To do this you:

- ❖ Create a wildcard virtual server on the inside BIG-IP Controllers (2a and 2b in Figure 11.1) to balance traffic outbound to the firewalls.

- ❖ Create a forwarding wildcard virtual server on the outside BIG-IP Controllers (1a and 1b in Figure 11.1) to forward traffic to the Internet.

## Creating a wildcard virtual server for balancing outbound traffic to the firewalls

To configure the inside BIG-IP Controller redundant system (2a and 2b in Figure 11.1) for outbound connections, create a wildcard virtual server that accepts all traffic from the internal network, then load balances the traffic through the firewalls.

### 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. Configure the following attributes on the Add Virtual Server screen. For additional information about configuring a virtual server, click the **Help** button.

#### Configuration notes

To create the configuration shown in Figure 11.1, select **firewalls\_inside** as the last hop pool for the virtual server **0.0.0.0:0**.

### 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 pool <pool name>
```

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

- ❖ **<interface>** is the interface on the BIG-IP Controller on which you want to create this virtual server.

❖ **<pool name>** is the name of the pool you want this virtual server to use.

For example, to use the **bigpipe vip** command to implement the configuration shown in Figure 11.1, you would type:

```
bigpipe vip 0.0.0.0:0 none use pool firewalls_inside
```

## Creating a forwarding wildcard virtual server to forward traffic to the Internet

After the appropriate firewall has processed outbound traffic, you want the outside BIG-IP Controller redundant system (1a and 1b in Figure 11.1) to forward traffic to the Internet. To accomplish this, create a wildcard virtual server as you did in *Creating a wildcard virtual server for balancing outbound traffic to the firewalls*, on page 11-13, using either the Configuration utility or the command line.

- ❖ If you use the Configuration utility, use the address and port **0.0.0.0:0** and select **Forwarding** in the **Resources** section.
- ❖ From the command line, to implement the configuration shown in Figure 11.1, you would type:

```
bigpipe vip 0.0.0.0:0 none forward
```

## Enhancing security

In some situations, you may want to limit the types of traffic that can pass outbound to the Internet. You can use **port-specific forwarding wildcard virtual servers** to restrict traffic in this manner. While a standard forwarding wildcard virtual server forwards all traffic, a port-specific forwarding wildcard virtual server forwards traffic specific to only the specified port.

### To create a port-specific wildcard server using the Configuration utility

Follow the instructions detailed in *To create a wildcard virtual server using the Configuration utility*, on page 11-13.

- ❖ When you configure the **Port** attribute, choose the port to which you want outgoing traffic to be limited for that virtual server.
- ❖ Complete the rest of the steps as detailed on page 11-13, then repeat the process for any other ports you want to be accessible to outgoing traffic.

For example, to implement the configuration shown in 11.1, to limit the traffic forwarded to HTTP and FTP, you would follow the instructions in *To create a wildcard virtual server using the Configuration utility* three times: That is, once for each of three port-specific virtual servers, entering **80**, **20**, and **21** for the **Port** attribute.

### **To create a port-specific wildcard server from the command line**

To create a port-specific wildcard server, use the **bigpipe vip** command as you did in *To create a network virtual server from the command line*, on page 11-8. For the sample port numbers, substitute the number of the port to which you want to limit access.

For example, in the configuration shown in Figure 11.1, to limit the traffic forwarded to HTTP and FTP, you replace the command in *To create a network virtual server from the command line* with the following commands:

```
bigpipe vip 0.0.0.0:80 none use pool firewall  
bigpipe vip 0.0.0.0:20 none use pool firewall  
bigpipe vip 0.0.0.0:21 none use pool firewall
```

## Designating a last hop pool for outbound traffic

Just as you used a last hop pool to balance and maintain inbound connections in *Designating a last hop pool for inbound traffic*, on page 11-11, you now create a last hop pool for outbound traffic for the same purposes. Create the pool on the outside BIG-IP Controller redundant system (1a and 1b in Figure 11.1).

Create the last hop pool for outbound traffic as you did for inbound traffic, using either the Configuration utility or the command line.

- ❖ If you use the Configuration utility, select the address and port **0.0.0.0:0**, then select **firewalls\_outside** in the **Last Hop Pool** section.
- ❖ From the command line, to implement the configuration shown in Figure 11.1, you type:

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

## Configuring interfaces

Typically, a BIG-IP Controller redundant system has two interfaces:

- ❖ An external interface, usually set for destination processing
- ❖ An internal interface, usually set for source processing

In order for the firewall sandwich configuration to work, you must set all interfaces on the BIG-IP Controller redundant system (1a and 1b, and 2a and 2b, in Figure 11.1) to process both source and destination addresses.

Therefore, you must enable both source processing for the external interfaces, and destination processing for the internal interfaces.

### **To configure source and destination processing 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 11.1, for both **exp0** and **exp1**, make sure the **Enable Source Processing** and the **Enable Destination Processing** check boxes are checked.

**To configure source processing from the command line**

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

**bigpipe interface <interface> source enable**

where **<interface>** is the identifier for the internal interface of a BIG-IP Controller redundant system.

To implement the configuration shown in Figure 11.1, you use the command:

**bigpipe interface exp0 source enable**

Repeat this process for each BIG-IP Controller redundant system.

**To configure destination processing from the command line**

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

**bigpipe interface <interface> dest enable**

where **<interface>** is the identifier for the external interface of a BIG-IP Controller redundant system.

To implement the configuration shown in Figure 11.1, you would use the command:

**bigpipe interface exp1 dest enable**

Repeat this process for each BIG-IP Controller.

## Configuring administrative routing

In order to administer the outside BIG-IP Controller redundant system (BIG-IP Controllers 1a and 1b in Figure 11.1) from the inside BIG-IP Controller redundant system (BIG-IP Controllers 2a and 2b in Figure 11.1), and to administer the inside redundant system from the outside redundant system, you need to create routes between the systems, using the firewalls as gateways.

To implement the configuration shown in Figure 11.1, you would use the following commands on BIG-IP Controllers 1a and 1b:

```
route add -host 10.10.30.1 -gateway 10.10.20.4  
route add -host 10.10.30.2 -gateway 10.10.20.5  
route add -host 10.10.30.3 -gateway 10.10.20.6
```

To complete the configuration, you would use the following commands on BIG-IP Controllers 2a and 2b:

```
route add -host 10.10.20.1 -gateway 10.10.30.4  
route add -host 10.10.20.2 -gateway 10.10.30.5  
route add -host 10.10.20.3 -gateway 10.10.30.6
```



# 12

---

## 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 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 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 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 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 12.1, 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.

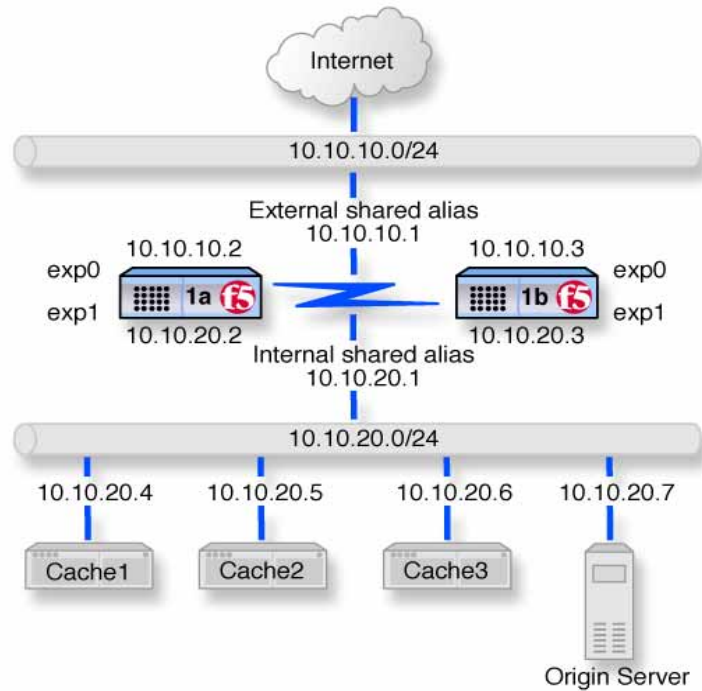


**Figure 12.1** Local server acceleration

## 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 12.1. 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 Controller redundant system to direct traffic. For more information about pools, refer to .

You will create these pools:

### ❖ **Cache server pool**

The BIG-IP 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 12-8.
- 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 Controller redundant system directs the request to this pool.

**◆ Note**

---

*While the configuration shown in Figure 12.1 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 12.1:

- 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 12.1, 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 12.1:

- 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 12.1, 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 12.1:

- 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 12.1, 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 Controller directs traffic. A **cache control rule** determines where and how the BIG-IP 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 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 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 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 Controller 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 12.1, 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 12.1, 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 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 12.1:

- 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 12.1, 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 Controller, so that the cache sends miss requests to the internal shared interface of the BIG-IP Controller. The BIG-IP 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 Controller rather than to the original cache server that generated the miss request, the BIG-IP 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 Controller.
- ❖ Enable destination processing on the internal interface of the BIG-IP 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 17-26.

### 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 12.1, 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 12.1, you use the command:

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

## Configuring interfaces

Typically, a BIG-IP 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 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 12.1, 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 12.1 from the command line, you use the command:

```
bigpipe interface exp1 dest enable
```

# 13

---

## 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 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 12, *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 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 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 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 13.1, 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.

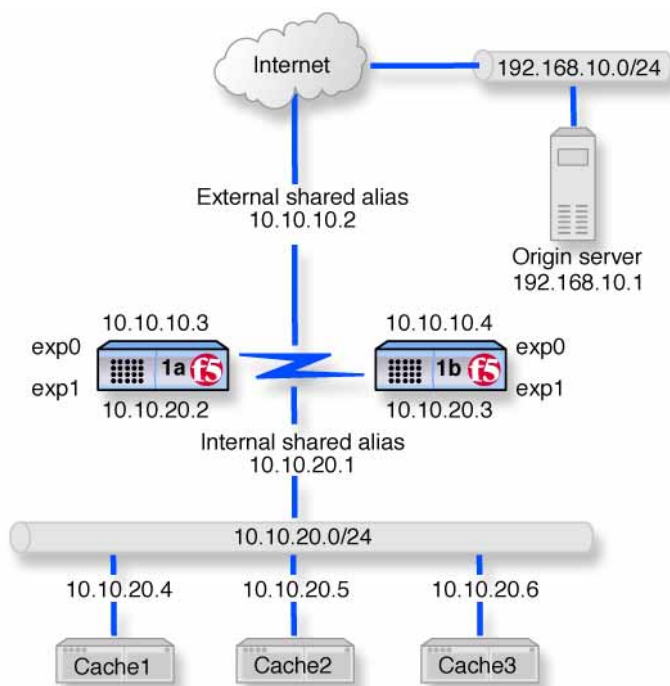


**Figure 13.1** Remote server acceleration

## 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 13.1. 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 Controller redundant system to direct traffic. For more information about pools, refer to *Configuring a pool*, on page 17-5.

You will create these pools:

- ❖ **Cache server pool**

The BIG-IP 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 13-9.
- 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 Controller redundant system directs the request to this pool.

---

◆ **Note**

*While the configuration shown in Figure 13.1 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 13.1:

- 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 13.1, 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 13.1:

- 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 13.1, 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 13.1:

- 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 13.1, 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 Controller directs traffic. A **cache control rule** determines where and how the BIG-IP 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 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 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 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 Controller 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 13.1, 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 13.1, 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 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 13.1:

- 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 13.1, 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 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 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 Controller, so that the cache sends miss requests to the internal shared interface of the BIG-IP Controller. The BIG-IP 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 Controller rather than to the original cache server that generated the miss request, the BIG-IP 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 Controller.
- ❖ Enable source translation, but disable source processing on the external interface of the BIG-IP Controller.
- ❖ Enable destination and source processing on the internal interface of the BIG-IP 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 17-26.

### 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 13.1, 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 13.1, you use the command:

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

## Configuring interfaces

Typically, a BIG-IP 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 Controller redundant system and the cache servers). Thus, when a BIG-IP 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 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 13.1, 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 13.1 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 Controller redundant system, you must add destination processing to the internal interface. Adding destination processing to the internal interface enables the BIG-IP 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 13.1, for **exp1**, make sure the **Enable Destination Processing** checkbox 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 13.1 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 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 13.1, 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 13.1, you use the command:

**bigpipe 192.168.10.1 remote**





# 14

---

## 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 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 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 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 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 14.1, 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.

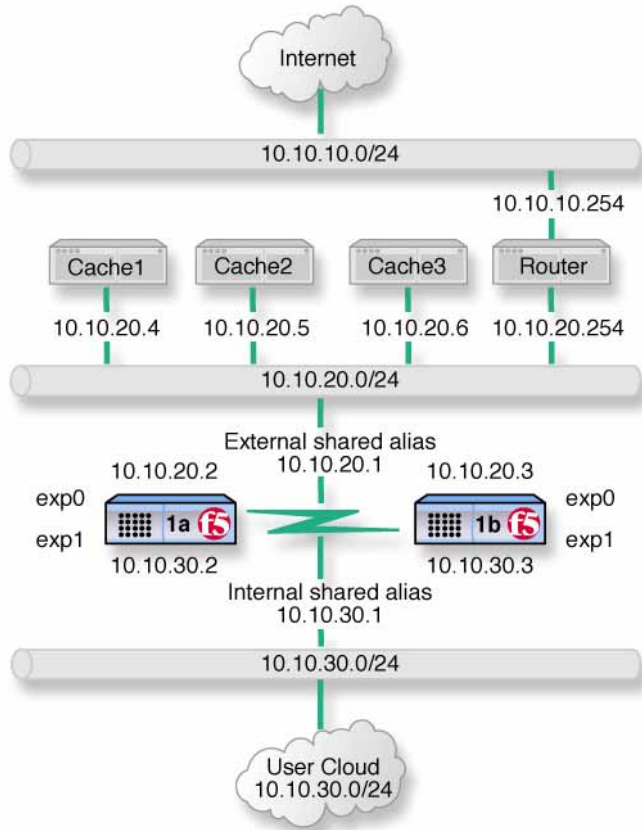


**Figure 14.1** Caching Internet content

## 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 14.1. 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 Controller redundant system to direct traffic. For more information about pools, refer to *Configuring a pool*, on page 17-5.

You create three pools:

- ❖ **Cache server pool**

The BIG-IP 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 14-8.
- 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 Controller redundant system directs the request to this pool.

**◆ Note**

---

*While the configuration shown in Figure 14.1 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 14.1:

- 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 14.18, 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 14.1:

- 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 14.1, 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 14.1:

- 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 14.1, 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 Controller directs traffic. A **cache control rule** determines where and how the BIG-IP 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 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 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 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 Controller 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 14.1, 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 14.1, 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 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 14.1, you would use the command:

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





# 15

---

## SNAT and Virtual Servers Combined

---

- SNAT and virtual servers combined



## SNAT and virtual servers combined

In some cases you may want to configure outbound transparent device load balancing and SNAT source translations. In this configuration, the BIG-IP Controller changes the source address of the clients to the external SNAT address. In this way, the actual IP addresses are not exposed to the Internet. At the same time the BIG-IP Controller can load balance the same connection across multiple nodes. Therefore, both SNAT translation and virtual server load balancing can operate on the same connection in this configuration.

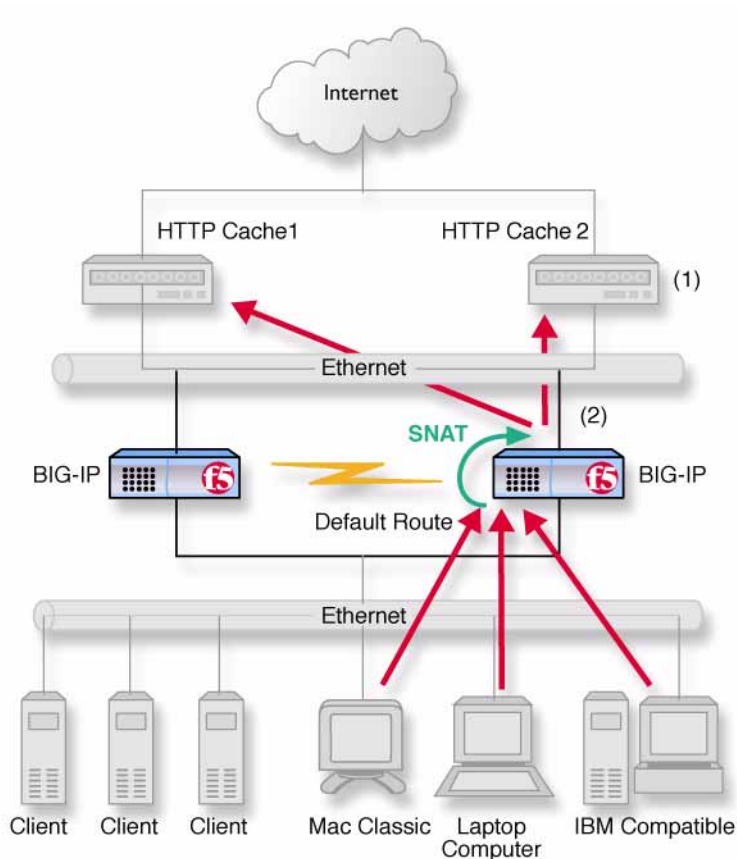


**Figure 15.1** An example of a virtual server/SNAT combination

### Configuring interfaces for the SNAT and virtual server combination

The SNAT and virtual server combination does not require additional interface configuration. However, in this configuration, the destination processing interface must be on the internal network and the source processing interface must be on the external network.

### Defining a pool for the HTTP cache servers

Finally, define the pool **cache\_pool** for the nodes that handle the requests to virtual server 0.0.0.0:0:

```
bigpipe pool cache_pool { lb_mode rr member <HTTPcache1>:80 member  
  <HTTPcache2>:80 }
```

Replace **<HTTPcache1>** and **<HTTPcache2>** with IP address of the respective HTTP cache server. Also note that this example uses the global round robin load balancing method.

### Outbound configuration

Next, configure controllers 1a and 1b to handle outbound traffic. Create a virtual server that sends traffic to the pool you created for the internal interfaces of the HTTP cache servers (**cache\_pool**). Use the following commands to create a virtual server for connecting to cache servers:

```
bipipe vip 0.0.0.0:0 exp1 use pool cache_pool  
bipipe vip 0.0.0.0:0 translate port disable
```

Note that port translation has been turned off because the members in the **vpn\_insid** pool were defined with port 80 for service checking. If port translation is not disabled, then all outbound connections would be translated to port 80.

After you create the virtual server, type the following SNAT command:

```
bipipe snat map client1 client2 client3 to 205.100.19.23
```

Replace **client1**, **client2**, and **client3** with the actual names of the clients in your configuration.



# 16

---

## Setting up 802.1q VLAN Trunk Mode

---

- Setting up 802.1q VLAN trunk mode





## Setting up 802.1q VLAN trunk mode

The BIG-IP Controller supports VLANs based on the IEEE 802.1q Trunk mode on BIG-IP Controller internal interfaces. VLAN tags are not supported on the external interfaces. You can define a single VLAN tag for each IP address defined for each BIG-IP Controller internal interface. This includes node network addresses, administrative addresses, shared administrative aliases, and additional aliases.

### ◆ Note

*In order for 802.1q VLAN trunk mode to operate on a BIG-IP Controller interface, all IP addresses on that interface must have a VLAN tag.*

In order to use VLAN tags, you must edit **/etc/netstart**. Additionally, if you plan to use VLAN tags on a redundant BIG-IP system, you must add VLAN tags to the shared IP aliases in BIG/db using the **bigpipe ipalias** command.

## Adding VLAN tag definitions to /etc/netstart

You must specify the VLAN tag ID for the network at the time you define the network address for a particular internal interface. You can do this by extending the **additional\_xxx** definition for the internal interface (where **xxx** is the interface name, such as **exp0**, **exp1**, or **hmc0**). For example, if you have an internal interface IP defined as:

```
ipaddr_exp1="10.1.1.1"
netmask_exp1="255.0.0.0"
linkarg_exp1="media 100BaseTX, FDX"
additional_exp1="broadcast 10.255.255.255"
```

To define a VLAN tag ID 12 for this network (10.0.0.0), extend the **additional\_exp1** definition in the following manner:

```
additional_exp1="broadcast 10.255.255.255 vlan 12"
```

Do this for each internal interface for which you want to define a VLAN tag ID.

## Adding VLAN tag definitions to BIG/db

For a redundant configuration, the BIG/db database contains the shared IP addresses for the internal and external interfaces for the BIG-IP Controller. If you plan to use VLAN tags on a redundant BIG-IP system, you must add the shared IP addresses to this database. Use the following syntax to add VLAN tag definitions to BIG/db.

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

For example, using the previous example, this line is extended with the same VLAN tag defined for its primary address, in this case 12:

```
bigpipe ipalias exp1 10.1.1.10 netmask 255.0.0.0 broadcast  
10.255.255.255 tag 12
```

## Configuring multiple VLANs on one interface

In order to set up multiple VLANs on the same interface, you need to add a new IP address for the interface. The BIG-IP Controller only supports one VLAN ID per network.

For example, to support an additional network, 12.0.0.0, with a VLAN tag ID of 15 on the same interface, add the following line to your `/etc/netstart` file after the `ifconfig` command:

```
/sbin/ifconfig exp1 add 12.1.1.1 netmask 255.0.0.0 media  
100BaseTX,FDX broadcast 12.255.255.255 vlan 15
```

Note that you must add a shared address to the BIG/db file with the **bigpipe ipalias** command in a redundant BIG-IP system:

```
bigpipe ipalias exp1 12.1.1.1 netmask 255.0.0.0 broadcast  
12.255.255.255 tag 15
```

## To enable or disable VLAN tags on the command line

Once you have added VLAN tags, you can use the **bigpipe interface** command to enable, disable, or show the current settings for the interface. To globally enable or disable the VLAN tags for an internal interface, use the following syntax:

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

For example, use the following command to enable VLAN tags on the interface **exp1**:

```
bigpipe interface exp1 vlans enable
```

## Using ifconfig to add another VLAN

You must use the **ifconfig** command to define multiple, different VLAN tagged networks on the same interface. For example, use the following syntax to add a new VLAN tagged network on the same interface:

```
ifconfig exp1 add <address> netmask <mask> broadcast <address> vlan  
<tag>
```

Note that the BIG-IP Controller allows one VLAN tag per network. In a redundant configuration, you need to add a new shared address on the new network with the identical VLAN tag ID in the BIG/db database with the **bigpipe ipalias** command.

You can also use **ifconfig** to display VLAN information for the interface **exp1** with the following command:

```
ifconfig exp1
```

## Using netstat to view VLAN tags

You can also use the **netstat** utility to display VLAN tag information with the route table for the BIG-IP Controller. Use the following syntax to display VLAN tag information with **netstat**:

```
netstat -nrT
```

## Disabling and enabling VLAN tags using the Configuration utility

You can use the Configuration utility to enable or disable VLAN tags once they are configured on the BIG-IP Controller.

1. In the navigation pane, select **NICs**.  
The Network Interface Cards screen opens.
2. In the Network Interface Cards list, select the internal NIC for which you want to enable VLAN tags.  
The Network Interface Card Properties screen opens.
3. In the **Network Interface Card Properties** screen, navigate to the **Enable VLANs** check box.  
Click the **Enable VLANs** check box to enable the VLAN tags for the interface. Clear the check box to disable VLAN tags on the interface.
4. Click the **Apply** button.

---

### ◆ Note

*You can only enable or disable VLAN tags in the Configuration utility. VLAN tags must be configured by adding VLAN tag values to the **/etc/netstart** file (and the **BIG/db** with the **bigpipe ipalias** command for redundant configurations). The Configuration utility can only enable or disable VLAN tags that have been configured in those files.*

# 17

---

## Essential Configuration Tasks

---

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



## Determining 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 advanced 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 17.1 describes the different types of connection configurations available on the BIG-IP Controller.

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

**Table 17.1** 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 17-21)
- ❖ Extended Content Verification service checking (see ***Configuring Extended Content Verification service checking*** on page 17-28)
- ❖ Persistence for connections that should return to the node to which they last connected (see ***Configuring persistence for e-commerce and other dynamic content sites*** on page 17-35)

## 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 note**

- 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 17-18, 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.

### 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.

#### Configuration note

- 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.

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

### 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 note

- 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

---

*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.

### 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
- For additional information about the options on this screen, click the **Help** button.

**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.

---

**◆ 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.

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

**To set TCP idle connection timers 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 17-28).

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

#### Configuration notes

- For the **Frequency** setting, we recommend 5 seconds for most configurations.
- For the **Timeout** setting, we recommend 16 seconds for most configurations.
- For additional information about the options on this screen, click the **Help** button.

**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 17-17.

---

**◆ Note**

*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 three main tasks of a basic configuration. This means you already have:

- ❖ 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**.

#### Configuration note

- 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, 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 three main tasks of a basic configuration. This means you already have:

- ❖ 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 **BIG-IP Controller 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 17-27) that can help protect your nodes.

### ◆ **WARNING**

*NATs and SNATs do not support NT Domain or CORBA IIOP. Instead of using NATs or SNATs, you need to configure IP forwarding (see **Setting up IP forwarding** on page 17-27).*

## 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.

#### Configuration note

- 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.

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

**To configure SNAT global properties from the command line**

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

```
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.

#### Configuration note

- 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 IIOP, 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.

#### Configuration note

- 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 other routers in the network. 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 17-13.*

## 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 send and receive strings 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 string 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 check 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:

**we!come home!**

8. Click the **Apply** button.

#### **Configuration note**

- 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 persistence for e-commerce and other dynamic content sites

Persistence is another feature you can configure after you have completed the three main tasks of a basic configuration. This means you already have:

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

If you are setting up an e-commerce or other type of dynamic content site, you may need to configure persistence on the BIG-IP Controller. Whether you need to configure persistence or not simply depends on how you store client-specific information, such as items in a shopping cart, or airline ticket reservations. For example, you may store the airline ticket reservation information in a back-end database that all nodes can access; or on the specific node to which the client originally connected; or in a cookie on the client's machine.

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

The BIG-IP Controller tracks information about individual persistent connections, and keeps the information only for a given period of time. The way in which persistent connections are identified depends on the type of persistence. The BIG-IP Controller supports two basic types of persistence:

❖ **SSL persistence**

SSL persistence is a type of persistence that tracks SSL connections using the SSL session ID, and it is a property of each individual pool. Using SSL persistence can be particularly important if your clients typically have translated IP addresses or dynamic IP addresses, such as those that Internet service providers usually assign. Even when the client's IP address changes, the BIG-IP Controller still recognizes the connection as being persistent, based on the session ID.

❖ **Simple persistence**

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

You may want to use SSL persistence and simple persistence together. In situations where an SSL session ID times out, or where a returning client does not provide a session ID, you may want the BIG-IP Controller to direct the client to the original node based on

the client's IP address. As long as the client's simple persistence record has not timed out, the BIG-IP Controller can successfully return the client to the appropriate node.

---

◆ **Note**

*The BIG-IP Controller also supports several advanced persistence modes. For more information about these advanced modes, see [Setting up persistence for a pool](#), on page 1-65.*

## Setting up SSL persistence

SSL persistence is a property of a pool. You can set up SSL persistence from the command line or using the Configuration utility. To set up SSL persistence, you need to do two things:

- ❖ Turn SSL persistence on.
- ❖ Set the SSL session ID timeout, which determines how long the BIG-IP Controller stores a given SSL session ID before removing it from the system.

### **To configure SSL persistence using the Configuration utility**

1. In the navigation pane, click **Pools**.  
The Pools screen opens.
2. Click the appropriate pool in the list.  
The Pool Properties screen opens.
3. In the toolbar, click the **Persistence** button.  
The Pool Persistence screen opens.
4. Click the **SSL Persistence** button.
5. In the **Timeout** box, type the number of seconds that the BIG-IP Controller should store SSL sessions IDs before removing them from the system.

6. Click the **Apply** button.

#### **Configuration note**

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

#### **To configure SSL persistence from the command line**

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

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

For example, if you want to set SSL persistence on the pool **my\_pool**, type the following command:

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

## Setting up simple persistence

You can set simple persistence properties for a pool. Pool persistence settings can override those of the port. When you set simple persistence on a port, all virtual servers that use the given port inherit the port's persistence settings.

### Setting simple persistence for a pool

Persistence settings for pools apply to both TCP and UDP persistence. When the persistence timer is set to a value greater than 0, persistence is **on**. When the persistence timer is set to 0, persistence is **off**.

#### **To configure simple persistence for pools using the Configuration utility**

1. In the navigation pane, click **Pools**.  
The Pools screen opens.
2. Select the pool for which you want to configure simple persistence.  
The Pool Properties screen opens.

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

**Configuration note**

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

**To configure simple persistence for pools from the command line**

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

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

For example, if you want to set SSL persistence on the pool **my\_pool**, type the following command:

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

## Configuring and synchronizing redundant systems

Another optional feature you can configure after you have performed the three 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.

**Configuration note**

- 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.

#### Configuration note

- 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**

# 18

---

## Monitoring and Administration

---

- Monitoring and administration utilities provided on the BIG-IP Controller
- Using the bigpipe command utility as a monitoring tool
- Working with the BIG/stat utility
- Working with the BIG/top utility
- Working with the Syslog utility
- Removing and returning items to service
- Viewing system statistics and log files
- Printing the connection table
- Changing passwords for the BIG-IP Controller
- Working with the BIG/db database



## Monitoring and administration utilities provided on the BIG-IP Controller

The BIG-IP platform provides several utilities for monitoring and administration of the BIG-IP Controller. You can monitor system statistics, as well as statistics specific to virtual servers and nodes, such as the number of current connections, and the number of packets processed since the last reboot.

The BIG-IP platform provides the following monitoring and configuration and administration utilities:

- ❖ **bigpipe**

If you type certain **bigpipe** commands, such as **bigpipe vip** or **bigpipe node**, and use the **show** keyword in the command, the command displays statistical information about the elements that you configure using that command. You can also use **bigpipe** commands to selectively reset any statistic collected by the BIG-IP Controller.

- ❖ **The Configuration utility**

You can use the Configuration utility to configure any feature on the BIG-IP Controller. You can reset any statistic, or all statistics, for virtual servers, nodes, NATs, and SNATs in the Configuration utility.

- ❖ **BIG/stat**

This utility is provided specifically for statistical monitoring of virtual servers, nodes, NATs, SNATs, and services. One benefit of using BIG/stat is that it allows you to customize the display of statistical information.

- ❖ **BIG/top**

BIG/top provides statistical monitoring. You can set a refresh interval, and you can specify a sort order.

- ❖ **Syslog**

Syslog is the standard UNIX system logging utility, which monitors critical system events, as well as configuration changes made on the BIG-IP Controller.

- ❖ **BIG/db**

BIG/db is a database that contains various configuration information for the BIG-IP Controller.

## Using the **bigpipe** command utility as a monitoring tool

Using the **bigpipe** utility, you can view information about the BIG-IP Controller itself, as well as elements such as virtual servers, virtual addresses, virtual ports, nodes, and node addresses.

Typically, the **bigpipe** utility provides the following statistics:

- ❖ Current number of connections
- ❖ Maximum number of concurrent connections
- ❖ Total number of connections since the last system reboot
- ❖ Total number of bits (inbound, outbound, total)
- ❖ Total number of packets (inbound, outbound, total)

### Monitoring the BIG-IP Controller

The **bigpipe summary** command displays performance statistics for the BIG-IP Controller itself. This display summary includes current usage statistics, such as the amount of time a BIG-IP Controller has been running since the last reboot. Type the following command:

**bigpipe summary**

The performance statistics display in the format shown in Figure 18.1 (the output has been truncated for this example).

```

BIG-IP total uptime          = 1 (day) 4 (hr) 40 (min) 8 (sec)
BIG-IP total uptime (secs)   = 103208
BIG-IP total # connections   = 0
BIG-IP total # pkts          = 0
BIG-IP total # bits          = 0
BIG-IP total # pkts(inbound) = 0
BIG-IP total # bits(inbound) = 0
BIG-IP total # pkts(outbound) = 0
BIG-IP total # bits(outbound) = 0
BIG-IP error no nodes available      = 0
BIG-IP tcp port deny                  = 0
BIG-IP udp port deny                  = 0
BIG-IP vip tcp port deny              = 0
BIG-IP vip udp port deny              = 0
BIG-IP max connections deny          = 0
BIG-IP vip duplicate syn ssl          = 0
BIG-IP vip duplicate syn wrong dest  = 0
BIG-IP vip duplicate syn node down   = 0
BIG-IP vip maint mode deny           = 0
BIG-IP virtual addr max connections deny = 0
BIG-IP virtual path max connections deny = 0
BIG-IP vip non syn                   = 0
BIG-IP error not in out table        = 0
BIG-IP error not in in table         = 0
BIG-IP error vip fragment no port    = 0
BIG-IP error vip fragment no conn    = 0
BIG-IP error standby shared drop     = 0
BIG-IP dropped inbound               = 0
BIG-IP dropped outbound              = 0
BIG-IP reaped                       = 0
BIG-IP ssl reaped                    = 0
BIG-IP persist reaped                = 0
BIG-IP udp reaped                    = 0
BIG-IP malloc errors                 = 0
BIG-IP bad type                      = 0
BIG-IP mem pool total 96636758 mem pool used 95552 mem percent
used 0.10

```

**Figure 18.1** The *bigpipe* summary display screen

Table 18.1 contains descriptions of each individual statistic included in the summary display screen.

| Statistic                    | Description                                                                                                                                                        |
|------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| total uptime                 | Total time elapsed since the BIG-IP Controller was last booted.                                                                                                    |
| total uptime (secs)          | Total uptime displayed in seconds.                                                                                                                                 |
| total # connections          | Total number of connections handled.                                                                                                                               |
| total # pkts                 | Total number of packets handled.                                                                                                                                   |
| total # bits                 | Total number of bits handled.                                                                                                                                      |
| total # pkts (inbound)       | Total number of incoming packets handled.                                                                                                                          |
| total # bits (inbound)       | Total number of incoming bits handled.                                                                                                                             |
| total # pkts (outbound)      | Total number of outgoing packets handled.                                                                                                                          |
| total # bits (outbound)      | Total number of outgoing bits handled.                                                                                                                             |
| error no nodes available     | The number of times the BIG-IP Controller tries to make a connection to a node, but no nodes are available.                                                        |
| tcp port deny                | The number of times a client attempted to connect to an unauthorized TCP port on the BIG-IP Controller (unauthorized port and source IP are logged in the syslog). |
| udp port deny                | The number of times a client attempted to connect to an unauthorized UDP port on the BIG-IP Controller (unauthorized port and source IP are logged in the syslog). |
| vip tcp port deny            | The number of times a client attempted to connect to an unauthorized TCP port on a virtual address (unauthorized port and source IP are logged in the syslog).     |
| vip udp port deny            | The number of times a client attempted to connect to an unauthorized UDP port on a virtual address (unauthorized port and source IP are logged in the syslog).     |
| max connections deny         | The total number of connections denied because the maximum number of connections allowed was exceeded.                                                             |
| vip duplicate syn ssl        | The number of duplicate connection attempts to existing SSL connections from the same client.                                                                      |
| vip duplicate syn wrong dest | The number of duplicate connection attempts from the same client (address and port combination) to a different virtual server.                                     |
| vip duplicate syn node down  | The number of duplicate connection attempts to a server that is down when a connection to the server was made previously.                                          |

**Table 18.1** *bigpipe monitoring statistics*



| Statistic                         | Description                                                                                                                                     |
|-----------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| vip maint mode deny               | The number of times a connection to a virtual server was denied while the BIG-IP Controller is in maintenance mode.                             |
| virtual addr max connections deny | The number of virtual address connections dropped because the maximum number of connections was exceeded.                                       |
| virtual path max connections deny | The number of virtual path connections dropped because the maximum number of connections was exceeded.                                          |
| vip non syn                       | The number of packets received which are not connection requests, and are destined to a virtual address, but not a valid virtual server (port). |
| error vip fragment no port        | The number of IP fragments for which there is no port.                                                                                          |
| error vip fragment no conn        | The number of IP fragments for which there is no connection.                                                                                    |
| error standby shared drop         | The number of packets destined to the shared IP address in a redundant system that are received and ignored by the standby system.              |
| dropped inbound                   | The total number of inbound packets dropped by the BIG-IP Controller.                                                                           |
| dropped outbound                  | The total number of outbound packets dropped by the BIG-IP Controller.                                                                          |
| reaped                            | The total number of connections that timed-out, and are deleted by the BIG-IP Controller.                                                       |
| ssl reaped                        | The total number of SSL session ID records that timed-out, and are closed by the BIG-IP Controller.                                             |
| persist reaped                    | The total number of persistence records that timed-out, and are closed by the BIG-IP Controller.                                                |
| udp reaped                        | The total number of UDP connections that timed-out, and are closed by the BIG-IP Controller.                                                    |
| malloc errors                     | The number of times a connection could not be created because the system is low on memory.                                                      |
| mem pool total                    | The total amount of memory available in all combined memory pools.                                                                              |
| mem pool used                     | The total amount of memory, in all combined memory pools, in use by the BIG-IP Controller.                                                      |
| mem percent used                  | The total percentage of memory in use by all combined memory pools.                                                                             |

**Table 18.1** *bigpipe* monitoring statistics

## Resetting statistics on the BIG-IP Controller

The **bigpipe** commands allow you to selectively reset any statistic on the BIG-IP Controller. The statistics you can reset selectively include:

- ❖ Virtual address
- ❖ Virtual server
- ❖ Node address
- ❖ Node server
- ❖ Virtual port
- ❖ Network address translations (NATs)
- ❖ Global statistics

When you reset one of these items, the packets in, packets out, bytes in, and bytes out counters of the target item are reset to zero. The max connection count counter is also reset. The current connections counter is not reset and the total connections counter is set equal to the number of current connections.

---

### ◆ Note

*The statistics are reset for the specified items only. Statistics for dependent items, such as node servers for a given virtual address, are not modified by these commands. The only exception is the global statistics reset option which resets traffic statistics for all items. After an item-level reset, statistics for all other dependent items do not add up.*

You can create an audit trail for reset events by setting an optional system control variable. You can set this variable to generate a syslog log entry. To set this variable, type the following command:

```
sysctl -w bigip.verbose_log_level=4
```

### **Resetting statistics for virtual servers and virtual addresses**

Use the following syntax to reset statistics for the virtual address specified by the IP address **<virtual ip>**.

```
bigpipe vip <virtual ip> stats reset
```

For example, if you want to reset statistics for the virtual address 172.20.1.100, type the following command:

```
bigpipe vip 172.20.1.100 stats reset
```

If you want to reset statistics for a list of virtual addresses, type the command with a list of addresses separated by spaces:

```
bigpipe vip 172.20.1.100 172.20.1.101 172.20.1.102 stats reset
```

If you want to reset statistics for all virtual servers, use the following command:

```
bigpipe vip stats reset
```

Use the following syntax to reset statistics for the virtual server IP:port combination <virtual IP>:<port>.

```
bigpipe vip <virtual IP>:<port> stats reset
```

For example, if you want to reset statistics for the virtual address/port combination 172.20.1.100:80, type the following command:

```
bigpipe vip 172.20.1.100:80 stats reset
```

If you want to reset statistics for a list of virtual address/port combinations, type the command with the list of addresses separated by spaces:

```
bigpipe vip 172.20.1.100:80 172.20.1.100:23 172.20.1.101:80 stats  
reset
```

### **Resetting statistics for node servers and node addresses**

Use the following syntax to reset statistics for all node addresses and node servers.

```
bigpipe node stats reset
```

You can reset statistics for the node address specified by the IP address <node IP>.

```
bigpipe node <node IP> stats reset
```

For example, to reset the statistics for the node address 10.1.1.1, use the following syntax:

```
bigpipe node 10.1.1.1 stats reset
```

If you want to reset statistics for a list of node addresses, type the command with the list of addresses separated by spaces:

```
bigpipe node 10.1.1.1 10.1.1.2 10.1.1.3 stats reset
```

Use the following syntax to reset statistics for the node server specified by the IP:port combination **<node IP>:<port>**.

```
bigpipe node <node IP>:<port> stats reset
```

For example, to reset the statistics for the node server 10.1.1.1:80, use the following syntax:

```
bigpipe node 10.1.1.1:80 stats reset
```

If you want to reset statistics for a list of node server addresses, type the command with the list of addresses separated by spaces:

```
bigpipe node 10.1.1.1:80 10.1.1.2:23 stats reset
```

### **Resetting statistics for virtual ports**

Use the following syntax to reset statistics for all virtual ports:

```
bigpipe port stats reset
```

Use the following syntax to reset statistics for the virtual port **<port>**. You can specify a list of virtual ports separated by spaces.

```
bigpipe port <port> stats reset
```

For example, to reset the statistics for the virtual port 80, use the following command:

```
bigpipe port 80 stats reset
```

To reset the statistics for a list of virtual ports, use the following syntax:

```
bigpipe port 23 80 443 stats reset
```

### **Resetting statistics for network address translations (NATs)**

Use the following syntax to reset statistics for all NATs:

```
bigpipe nat stats reset
```

Use the following syntax to reset statistics for the NAT specified by the IP address **<nat IP>**.

```
bigpipe nat <nat IP> stats reset
```

For example, to reset the statistics for the NAT 172.20.3.101, use the following command:

```
bigpipe nat 172.20.3.101 stats reset
```

To reset the statistics for a list of NATs, use the following command where NATs are separated by spaces:

```
bigpipe nat 172.20.3.101 172.20.3.102 stats reset
```

### **Resetting statistics for secure network address translations (SNATs)**

Use the following syntax to reset statistics for all SNATs:

```
bigpipe snat stats reset
```

Use the following syntax to reset statistics for the SNAT specified by the IP address <snat IP>.

```
bigpipe snat <snat IP> stats reset
```

For example, to reset the statistics for the SNAT 172.20.3.101, use the following command:

```
bigpipe snat 172.20.3.101 stats reset
```

To reset the statistics for a list of SNATs, use the following command where SNATs are separated by spaces:

```
bigpipe snat 172.20.3.101 172.20.3.102 stats reset
```

### **Resetting global statistics**

Use the following command to reset all statistics for all items.

```
bigpipe global stats reset
```

### **To reset any statistic in the Configuration utility**

A **Reset** button is located in each of the following tables in the Configuration utility:

- ❖ Virtual address
- ❖ Virtual server
- ❖ Node address
- ❖ Node server
- ❖ Virtual port

- ❖ Network address translations (NATs)
- ❖ Global statistics

To reset a statistic for a particular item, click the **Reset** button next to the item in one of these tables.

### Viewing the status of the interface cards

The **bigpipe interface** command displays the current status and the settings for external and internal interface cards. You can also use the **bigpipe interface** command to view information for a specific interface card, using the following command syntax:

```
interface <ifname>
```

## Monitoring virtual servers, virtual addresses, and services

You can use different variations of the **bigpipe vip** command, as well as the **bigpipe port** command, to monitor information about virtual servers, virtual addresses, and services managed by the BIG-IP Controller.

### Displaying information about virtual servers and virtual addresses

The **bigpipe vip** command displays the status of virtual servers (**up**, **down**, **unchecked**, or **disabled**), the current number of connections to each virtual server, and the status of the member nodes that are included in each virtual server mapping. The status for individual member nodes includes whether the node is **up**, **down**, **unchecked**, or **disabled**, and also includes the cumulative count of packets and

bits received and sent by the node on behalf of the virtual server.  
The BIG-IP Controller displays the statistics as shown in Figure 18.2.

```

VIP +-----> 11.11.11.50          UNIT 1
|          (cur, max, limit, tot) = (0, 8, 0, 370)
|          (pckts,bits) in = (10704, 8744872), out = (21480, 230874016)
+---+---> PORT http              UP
|          (cur, max, limit, tot) = (0, 8, 0, 370)
|          (pckts,bits) in = (10704, 8744872), out = (21480, 230874016)
POOL appgen_11.11.11.50.80
MEMBER 11.12.11.100:http        UP
|          (cur, max, limit, tot) = (0, 8, 0, 370)
|          (pckts,bits) in = (10704, 8744872), out = (21480, 230874016)

VIP +-----> 11.11.11.101         UNIT 1
|          (cur, max, limit, tot) = (0, 2, 0, 4)
|          (pckts,bits) in = (4532, 2090768), out = (6824, 82113984)
+---+---> PORT http              UP
|          (cur, max, limit, tot) = (0, 2, 0, 4)
|          (pckts,bits) in = (4532, 2090768), out = (6824, 82113984)
POOL my_website_pool
MEMBER 11.12.11.100:http        UP
|          (cur, max, limit, tot) = (0, 2, 0, 4)
|          (pckts,bits) in = (4532, 2090768), out = (6824, 82113984)

```

**Figure 18.2** Virtual server statistics

If you want to view statistical information about one or more specific virtual servers, simply include the virtual servers in the **bigpipe vip show** command as shown below:

**bigpipe vip <virt addr>:<port>... <virt addr>:<port> show**

If you want to view statistical information about traffic going to one or more virtual addresses, specify only the virtual address information in the command:

**bigpipe vip <virt addr>... <virt addr> show**

### Displaying information about services

The **bigpipe port show** command allows you to display information about specific virtual ports managed by the BIG-IP Controller. You can use the command to display information about all virtual services, or you can specify one or more particular virtual services.

To view information about all virtual services, use the following syntax:

**bigpipe port show**

To view statistical information about one or more specific virtual services, simply include the service names or port numbers as shown below:

**bigpipe port <port>... <port> show**

## Monitoring nodes and node addresses

The **bigpipe node** command displays the status of all nodes configured on the BIG-IP Controller. The information includes whether or not the specified node is **up**, **down**, **disabled**, or **unchecked**, and the number of cumulative packets and bits sent and received by each node on behalf of all virtual servers. The BIG-IP Controller displays the statistical information as shown in Figure 18.3.

```

NODE 11.12.11.100      UP
|      (cur, max, limit, tot) = (0, 8, 0, 374)
|      (pkts,bits) in = (15236, 10835640), out = (28304, 312988000)
+-      PORT http      UP
|      (cur, max, limit, tot) = (0, 8, 0, 374)
|      (pkts,bits) in = (15236, 10835640), out = (28304, 312988000)

```

*Figure 18.3 Node statistics screen*

If you want to view statistical information about one or more specific nodes, simply include the nodes in the **bigpipe node show** command as shown below:

**bigpipe node <node addr>:<port>... <node addr>:<port> show**

If you want to view statistical information about traffic going to one or more node addresses, specify only the node address information in the command:

**bigpipe node <node addr>... <node addr> show**



## Monitoring NATs

The **bigpipe nat show** command displays the status of the NATs configured on the BIG-IP Controller. The information includes the number of cumulative packets and bits sent and received by each NAT. Use the following command to display the status of all NATs included in the configuration:

**bigpipe nat show**

Use the following syntax to display the status of one or more selected NATs:

**bigpipe nat <node addr> [...<node addr>] show**

An example of the output for this command is in Figure 18.4.

```
NAT { 10.10.10.3 to 9.9.9.9 }
  (pckts, bits) in = (0, 0), out = (0, 0)
NAT { 10.10.10.4 to 12.12.12.12
netmask 255.255.255.0 broadcast 12.12.12.255 }
  (pckts, bits) in = (0, 0), out = (0, 0)
```

*Figure 18.4 NAT statistics*

## Monitoring SNATs

The **bigpipe snat show** command displays the status of the SNATs configured on the BIG-IP Controller. The information includes connections and global SNAT settings. Use the following **bigpipe** command to show SNAT mappings:

**bigpipe snat [<SNAT addr>] [...<SNAT addr>] show**

**bigpipe snat show**

Use the following command to show the current SNAT connections:

**bigpipe snat [<SNAT addr>] [...<SNAT addr>] dump [ verbose ]**

**bigpipe snat dump [ verbose ]**

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

The following command prints the global SNAT settings:

**bigpipe snat globals show**

## Working with the BIG/stat utility

BIG/stat™ is a utility that allows you to quickly view the status of the following elements:

- ❖ Virtual servers
- ❖ Services  
(cur, max, limit, tot) (pkts,bits) in out
- ❖ Nodes  
(cur, max, limit, tot) (pkts,bits) in out
- ❖ Ports
- ❖ Network address translations (NATs)

You can customize the BIG/stat utility statistics display. For example, you can customize your output to display statistics for a single element, or for selected elements. You can set the display to automatically update at time intervals you specify.

The **bigstat** command accepts one or more options, which allow you to customize the statistical display. When you use the **bigstat** command without specifying any options, the BIG/stat utility displays the list of virtual servers, services, nodes, NATs, and SNATs only one time. The basic command syntax is:

**bigstat [ options... ]**

The following table, Table 18.2, describes the options that you can use in the **bigstat** command.

| Option                  | Description                                              |
|-------------------------|----------------------------------------------------------|
| <b>-bigip</b>           | Displays totals for the BIG-IP Controller overall.       |
| <b>-c &lt;count&gt;</b> | Sets the interval at which new information is displayed. |
| <b>-h and -help</b>     | Displays the help options.                               |
| <b>-n</b>               | Displays data in numeric format.                         |
| <b>-nat</b>             | Displays network address table (NAT) entries only.       |

**Table 18.2** The **bigstat** command options

| Option             | Description                                     |
|--------------------|-------------------------------------------------|
| <b>-no_viptot</b>  | Removes virtual server totals from the display. |
| <b>-no_nodetot</b> | Removes node totals from the display.           |
| <b>-node</b>       | Displays nodes only.                            |
| <b>-port</b>       | Displays ports only.                            |
| <b>-v</b>          | Displays version information.                   |
| <b>-vip</b>        | Displays virtual servers only.                  |

**Table 18.2** The **bigstat** command options

Figure 18.5 contains an example of the output from the **bigstat** command. Table contains descriptions of each of the items in this example.

```

bigip springbank
  (cur, max, tot) = (0, 8, 374)
  (pckts,bits) in = (15310, 10860064), out = (28363, 313009048)
vip  11.11.11.50
  (cur, max, limit, tot) = (0, 8, 370, 370)
  (pckts,bits) in = (10704, 8744872), out = (21480, 230874016)
vip  11.11.11.50:http
      UP
  (cur, max, limit, tot) = (0, 8, 370, 370)
  (pckts,bits) in = (10704, 8744872), out = (21480, 230874016)
vip  11.11.11.101
  (cur, max, limit, tot) = (0, 2, 4, 4)
  (pckts,bits) in = (4532, 2090768), out = (6824, 82113984)
vip  11.11.11.101:http
      UP
  (cur, max, limit, tot) = (0, 2, 4, 4)
  (pckts,bits) in = (4532, 2090768), out = (6824, 82113984)
node 11.12.11.100
      UP
  (cur, max, limit, tot) = (0, 8, 374, 374)
  (pckts,bits) in = (15236, 10835640), out = (28304, 312988000)
node 11.12.11.100:http
      UP
  (cur, max, limit, tot) = (0, 8, 374, 374)
  (pckts,bits) in = (15236, 10835640), out = (28304, 312988000)
port WILDCARD PORT
  (cur, max, limit, tot, reaped) = (0, 0, 0, 0, 0)
  (pckts,bits) in = (0, 0), out = (0, 0)
port 80:http
  (cur, max, limit, tot, reaped) = (0, 8, 374, 374, 6)
  (pckts,bits) in = (15236, 10835640), out = (28304, 312988000)

```

**Figure 18.5** Sample output of the **bigstat** command

The following table contains a descriptions of each of the metrics collected for the BIG-IP Controller.

| BIG/stat Item     | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| BIG-IP Controller | <b>cur</b> Shows the number of current connections handled by the BIG-IP Controller<br><b>max</b> Shows the maximum number of connections handled by the BIG-IP Controller<br><b>tot</b> Shows the total number of connections handled by the BIG-IP Controller<br><b>pckts,bits in</b> Shows the total number of packets and bits coming into the BIG-IP Controller<br><b>pckts,bits out</b> Shows the total number of packets and bits going out of the BIG-IP Controller                                                           |
| virtual server    | <b>cur</b> Shows the number of current connections handled by the virtual server<br><b>max</b> Shows the maximum number of connections handled by the virtual server<br><b>limit</b> Shows the connection limit reached by the virtual server<br><b>tot</b> Shows the total number of connections handled by the virtual server<br><b>pckts,bits in</b> Shows the total number of packets and bits coming into the virtual server<br><b>pckts,bits out</b> Shows the total number of packets and bits going out of the virtual server |

**Table 18.3** Data displayed by the *bigstat* utility

| BIG/stat Item | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| service       | <b>cur</b> Shows the number of current connections handled by the service<br><b>max</b> Shows the maximum number of connections handled by the service<br><b>limit</b> Shows the connection limit reached by the service<br><b>tot</b> Shows the total number of connections handled by the BIG-IP service<br><b>pckts,bits in</b> Shows the total number of packets and bits coming into the service<br><b>pckts,bits out</b> Shows the total number of packets and bits going out of the service                                                   |
| nodes         | <b>cur</b> Shows the number of current connections handled by the node<br><b>max</b> Shows the maximum number of connections handled by the node<br><b>limit</b> Shows the connection limit reached by the node<br><b>tot</b> Shows the total number of connections handled by the BIG-IP node<br><b>pckts,bits in</b> Shows the total number of packets and bits coming into the node<br><b>pckts,bits out</b> Shows the total number of packets and bits going out of the node                                                                     |
| ports         | <b>cur</b> Shows the number of current connections handled by the port<br><b>max</b> Shows the maximum number of connections handled by the port<br><b>limit</b> Shows the connection limit reached by the port<br><b>tot</b> Shows the total number of connections handled by the BIG-IP port<br><b>reaped</b> Shows the number of connections reaped on the port<br><b>pckts,bits in</b> Shows the total number of packets and bits coming into the port<br><b>pckts,bits out</b> Shows the total number of packets and bits going out of the port |

**Table 18.3** Data displayed by the *bigstat* utility

## Working with the BIG/top utility

BIG/top™ is a real-time statistics display utility. The display shows the date and time of the latest reboot and lists activity in bits, bytes, or packets. Similar to BIG/stat, the BIG/top utility accepts

options which allow you to customize the display of information. For example, you can set the interval at which the data is refreshed, and you can specify a sort order. The BIG/top displays the statistics as shown in the following figure, Figure 18.6.

|                  |        |                     |                     |             |
|------------------|--------|---------------------|---------------------|-------------|
|                  |        | bits since          | bits in prior       | current     |
|                  |        | Nov 28 18:47:50     | 3 seconds           | time        |
| BIG-IP           | ACTIVE | ---In---Out---Conn- | ---In---Out---Conn- | 00:31:59    |
| 227.19.162.82    |        | 1.1G 29.6G 145      | 1.6K 0 0            |             |
| VIP ip:port      |        | ---In---Out---Conn- | ---In---Out---Conn- | -Nodes Up-- |
| 217.87.185.5:80  |        | 1.0G 27.4G 139.6K   | 1.6K 0 0            | 2           |
| 217.87.185.5:20  |        | 47.5M 2.1G 3.1K     | 0 0 0               | 2           |
| 217.87.185.5:20  |        | 10.2M 11.5M 2.6K    | 0 0 0               | 2           |
| NODE ip:port     |        | ---In---Out---Conn- | ---In---Out---Conn- | --State---- |
| 129.186.40.17:80 |        | 960.6M 27.4G 69.8K  | 672 0 0             | UP          |
| 129.186.40.17:20 |        | 47.4M 2.1G 3.1K     | 0 0 0               | UP          |
| 129.186.40.18:80 |        | 105.3M 189.0K 69.8K | 1.0K 0 0            | UP          |
| 129.186.40.17.21 |        | 9.4M 11.1M 1.3K     | 0 0 0               | UP          |
| 129.186.40.18:21 |        | 700.8K 414.7K 1.3K  | 0 0 0               | UP          |
| 129.186.40.18:20 |        | 352 320 1           | 0 0 0               | UP          |

Figure 18.6 The BIG/top screen display

Using BIG/top command options

The **bigtop** command uses the syntax below, and it supports the options outlined in Table 18.4:

**bigtop** [options...]

| Option         | Description                                                                 |
|----------------|-----------------------------------------------------------------------------|
| -bytes         | Displays counts in bytes (the default is bits).                             |
| -conn          | Sorts by connection count (the default is to sort by byte count).           |
| -delay <value> | Sets the interval at which data is refreshed (the default is four seconds). |
| -delta         | Sorts by count since last sample (the default is to sort by total count).   |
| -help          | Displays BIG/top help.                                                      |
| -nodes <value> | Sets the number of nodes to print (the default is to print all nodes).      |
| -nosort        | Disables sorting.                                                           |
| -once          | Prints the information once and exits.                                      |

Table 18.4 BIG/top command options

| Option                     | Description                                                                                |
|----------------------------|--------------------------------------------------------------------------------------------|
| <b>-pkts</b>               | Displays the counts in packets (the default is bits).                                      |
| <b>-scroll</b>             | Disables full-screen mode.                                                                 |
| <b>-vips &lt;value&gt;</b> | Sets the number of virtual servers to print (the default is to print all virtual servers). |

**Table 18.4** *BIG/top command options*

### Using runtime commands in BIG/top

Unless you specified the **-once** option, the BIG/top utility continually updates the display at the rate indicated by the **-delay** option, and you can also use the following runtime options at any time:

- ❖ The **u** option cycles through the display modes; bits, bytes, and packets.
- ❖ The **q** option quits the BIG/top utility.

## Working with the Syslog utility

The BIG-IP Controller supports logging via the **Syslog** utility. The logs are generated automatically, and saved in user-specified files. These logs contain all changes made to the BIG-IP Controller configuration, such as those made with the **bigpipe vip** command, or other **bigpipe** commands, as well as all critical events that occur in the system.

### ◆ Note

*You can configure the Syslog utility to send email or activate pager notification based on the priority of the logged event.*

The Syslog log files track system events based on information defined in the **/etc/syslog.conf** file. You can view the log files in a standard text editor, or with the **less** file page utility.

Sample log messages

Table 18.5 shows sample log messages give you an idea of how the Syslog utility tracks events that are specific to the BIG-IP Controller.

| Sample message                                                                | Description                                                                                                                                             |
|-------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>bigd: allowing connections on port 20</b>                                  | A user specifically allowed connections on virtual port 20.                                                                                             |
| <b>bigd: node 192.168.1.1 detected up</b>                                     | The 192.168.1.1 node address was successfully pinged by the BIG-IP Controller.                                                                          |
| <b>bigd: added service port 20 to node 192.168.1.1</b>                        | A user defined a new node, 192.168.1.1:20.                                                                                                              |
| <b>kernel: security: port denial 207.17.112.254:4379 -&gt; 192.168.1.1:23</b> | A client was denied access to a specific port. The client is identified as coming from 207.17.112.254:4379, and the destination node is 192.168.1.1:23. |

*Table 18.5 Sample Syslog messages*

## Removing and returning items to service

Once you have completed the initial configuration on the BIG-IP Controller, you may want to temporarily remove specific items from service for maintenance purposes. For example, if a specific network server needs to be upgraded, you may want to disable the nodes associated with that server, and then enable them once you finish installing the new hardware and bring the server back online.

If you specifically disable the nodes associated with the server, the BIG-IP Controller allows the node to go down only after all the current connections are complete. During this time, the BIG-IP Controller does not attempt to send new connections to the node. Although the BIG-IP Controller’s monitoring features would eventually determine that the nodes associated with the server are down, specifically removing the nodes from service can prevent interruptions on long duration client connections.



You can remove the entire BIG-IP Controller from service, or you can remove the following individual items from service:

- ❖ Virtual servers
- ❖ Virtual addresses
- ❖ Virtual ports
- ❖ Nodes
- ❖ Node addresses

## Removing the BIG-IP Controller from service

The BIG-IP platform offers a Maintenance mode, which allows you to remove the BIG-IP Controller from network service. This is useful if you want to perform hardware maintenance, or make extensive configuration changes. When you activate Maintenance mode, the BIG-IP Controller no longer accepts connections to the virtual servers it manages. However, the existing connections are allowed to finish processing so that current clients are not interrupted.

The **bigpipe maint** command toggles the BIG-IP Controller into or out of Maintenance mode. Use the following command to put the BIG-IP Controller in maintenance mode:

**bigpipe maint**

If the BIG-IP Controller runs in Maintenance mode for less than 20 minutes and you return the machine to the normal service, the BIG-IP Controller quickly begins accepting connections. However, if the BIG-IP Controller runs in Maintenance mode for more than 20 minutes, returning the Controller to service involves updating all network ARP caches. This process can take a few seconds, but you can speed the process up by reloading the **/etc/bigip.conf** file using the following command:

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

### To activate maintenance mode in the Configuration utility

1. In the navigation pane, click the BIG-IP logo.  
The BIG-IP System Properties screen opens.
2. Click the **Maintenance Mode** box.
3. Click on the **Apply** button.

## Removing individual virtual servers, virtual addresses, and ports from service

The BIG-IP Controller also supports taking only selected virtual servers, addresses, or ports out of service, rather than removing the BIG-IP Controller itself from service. Each **bigpipe** command that defines virtual servers and their components supports **enable** and **disable** keywords, which allow you to remove or return the elements from service.

When you remove a virtual address or a virtual port from service, it affects all virtual servers associated with the virtual address or virtual port. Similarly, if you remove a node address from service, it affects all nodes associated with the node address.

### Enabling and disabling virtual servers and virtual addresses

The **bigpipe vip** command allows you to enable or disable individual virtual servers, as well as virtual addresses. To enable or disable a virtual server, type the appropriate command:

```
bigpipe vip <virtual addr>:<virtual port> enable  
bigpipe vip <virtual addr>:<virtual port> disable
```

To enable or disable a virtual address, type the appropriate command:

```
bigpipe vip <virtual addr> enable  
bigpipe vip <virtual addr> disable
```

## Enabling and disabling virtual ports

The **bigpipe port** command allows you to allow or deny traffic on a virtual port:

```
bigpipe port <virtual port> enable
```

```
bigpipe port <virtual port> disable
```

## Removing individual nodes and node addresses from service

### Enabling and disabling nodes and node addresses

The **bigpipe node** command allows you to enable or disable individual nodes, as well as node addresses.

To enable or disable a **node**, type the appropriate command:

```
bigpipe node <node addr>:<node port> enable
```

```
bigpipe node <node addr>:<node port> disable
```

To enable or disable a **node address**, type the appropriate command:

```
bigpipe node <node addr> enable
```

```
bigpipe node <node addr> disable
```

## Viewing the currently defined virtual servers and nodes

When used with the **show** parameter, **bigpipe** commands typically display currently configured elements. For example, the **bigpipe vip show** command displays all currently defined virtual servers, and the **bigpipe node** command displays all nodes currently included in virtual server mappings. For additional information about using **bigpipe** commands on the BIG-IP Controller, see the *BIG-IP Reference Guide*, *bigpipe Command Reference*.

## Viewing system statistics and log files

The Configuration utility allows you to view a variety of system statistics and system log files. Note that from each statistics screen, you can access property settings for individual virtual servers, nodes, IP addresses, and ports by selecting the individual item in the statistics table.

### Viewing system statistics

The Configuration utility allows you to view the following statistical information:

- ❖ BIG-IP system statistics, including the elapsed time since the last system reboot, the number of packets and connections handled by the system, and the number of dropped connections.
- ❖ Virtual servers, including virtual servers, virtual address only, or virtual ports only.
- ❖ Nodes, including nodes, node addresses only, or node ports only.
- ❖ NAT statistics, such as the number of packets handled by each NAT.
- ❖ SNAT statistics, such as SNAT mappings.
- ❖ IP filter statistics, including the number of packets accepted and rejected by individual IP filters.
- ❖ Rate filter statistics, including the number of bits passed through, delayed, and dropped by individual rate filters.
- ❖ Information about illegal connection attempts, such as the source IP addresses from which the illegal connection is initiated.

Statistics are displayed in real-time. You can specify the update frequency by setting an interval (in seconds), and then clicking **Update**.

## Viewing log files

The Configuration utility allows you to display three different log files:

- ❖ The BIG-IP system log, which displays standard UNIX system events
- ❖ The BIG-IP log, which displays information specific to BIG-IP events, such as defining a virtual server
- ❖ The Pinger log, which displays status information determined by each node ping issued by the BIG-IP Controller

## Printing the connection table

The **bigpipe** command line utility also offers a useful diagnostic tool that prints the list of current connections. Normally, the **bigpipe conn** command prints the client, virtual server, and node addresses.

## Changing passwords for the BIG-IP Controller

During the First-Time Boot utility, you define a password that allows remote access to the BIG-IP Controller, and you also define a password for the BIG-IP web server. You can change these passwords at any time.

### To change the BIG-IP Controller password

1. At the BIG-IP Controller command line prompt, log on as the root user and use the **passwd** command.
2. At the password prompt, enter the password you want to use for the BIG-IP Controller and press **Return**.
3. To confirm the password, retype it and press **Return**.

## Changing passwords and adding new user IDs for the BIG-IP web server

You can create new users for the BIG-IP web server in the Configuration utility.

### Creating new users for the BIG-IP web server

The user accounts you create in the Configuration utility can have full, partial, or read-only access to the BIG-IP Controller.

#### To create user accounts in the Configuration utility

1. In the navigation pane, click **User Admin**.  
The User Administration screen opens.
2. In the Add User section, type the following information.
  - **User ID**  
Type the user ID you want to assign the user.
  - **Password**  
Type the password you want to assign the user.
  - **Retype Password**  
Retype the password you want to assign the user.
3. In the Access Level list, select the access level for the user.  
The access levels available are:
  - **Read Only**  
This access level allows the user only to view information in the Configuration utility. Users with this access level do not have access to **Add** buttons, certain toolbar items, **Apply** buttons, or **Remove** buttons.
  - **Partial Read/Write**  
In addition to allowing the user to view information, a Partial Read/Write user can also change the status of node addresses to either **enabled** or **disabled**.
  - **Full Read/Write**  
This access level provides the user with full access to all administrative tasks.

4. After you select the access level for the user, click the **Add** button.

The Current User list on the User Administration screen contains all users configured to access the Configuration utility. You can delete any user added through the Configuration utility by clicking the **Remove** button next to the user in the list. The BIG-IP web server administrator account you created with the First-Time Boot utility shows up in this list. However, you cannot edit or delete this account from the Configuration utility. To edit this account, you must run the **reconfig-httpd** command line utility. For more information about this utility, see the ***BIG-IP Controller Reference Guide**, BIG-IP Controller Configuration Utilities*.

## Working with the BIG/db database

The BIG/db™ database holds certain configuration information for the BIG-IP Controller. Most BIG-IP Controller utilities currently use the configuration stored in BIG/db. The **bigdba** utility is provided for loading configuration information into BIG/db. An additional **default.txt** file is included with the BIG-IP Controller which contains default information you can load into the BIG/db database.

### Using bigdba

Use the **bigdba** utility to modify the keys in the BIG/db database. The **bigdba** utility allows you to edit the database and insert and modify keys and values. All values are entered into BIG/db as strings.

## Accessing and modifying the default database

The default BIG/db database is created when you run the First-Time Boot utility. To use **bigdba** from the command line run **bigdba** with the name of the database.

**bigdba**

Database **"/var/f5/bigdb/user.db"** opened.

## Using bigdba commands

Table 18.6 describes the commands you can use in the **bigdba** utility.

| Command                                                  | Description                                                                                                                                                                                               |
|----------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>subkey &lt;string&gt;</b><br><b>sk &lt;string&gt;</b> | Add subkey to current key level                                                                                                                                                                           |
| <b>p &lt;name&gt;</b><br><b>p &lt;regex&gt;</b>          | Print the value stored at <b>&lt;name&gt;</b> . If you use a regular expression <b>&lt;regex&gt;</b> , all records under the current subkey which match the regular expression print.                     |
| <b>up</b>                                                | Back up one subkey                                                                                                                                                                                        |
| <b>up &lt;string&gt;</b>                                 | Back up through subkey <b>&lt;string&gt;</b>                                                                                                                                                              |
| <b>d &lt;string&gt;</b><br><b>d &lt;regex&gt;</b>        | Delete value stored under current key <b>&lt;string&gt;</b> . If you use a regular expression <b>&lt;regex&gt;</b> , all records under the current subkey which match the regular expression are deleted. |
| <b>&lt;name&gt; = &lt;value&gt;</b>                      | Store <b>&lt;value&gt;</b> under name <b>&lt;name&gt;</b> within the current key                                                                                                                          |
| <b>set confirm on</b>                                    | Confirm delete operations                                                                                                                                                                                 |
| <b>set confirm off</b>                                   | Do not confirm deletions                                                                                                                                                                                  |
| <b>set comments on</b>                                   | Show comments. By default, comments are <b>off</b>                                                                                                                                                        |
| <b>set comments off</b>                                  | Do not show comments. By default, comments are <b>off</b>                                                                                                                                                 |
| <b>dump &lt;file&gt;</b>                                 | Dumps the database to the file name specified                                                                                                                                                             |
| <b>load &lt;file&gt;</b>                                 | Loads the database with the file specified                                                                                                                                                                |
| <b>quit</b><br><b>q</b><br><b>EOF</b>                    | Quits the <b>bigdba</b> utility                                                                                                                                                                           |
| <b>help</b><br><b>?</b>                                  | Display the help text for the <b>bigdba</b> utility                                                                                                                                                       |

**Table 18.6** The **bigdba** commands



## Working with the default.txt file

The **default.txt** file documents the keys that are valid in the BIG/store database. This file is located at **/var/f5/bigdb/default.txt**. This text file, which can be loaded with the **bigdba** program, contains all the possible database keys, comments that document these keys, and the default values used by programs that run on the BIG-IP Controller.

---

### ◆ Note

*The values in the default.txt file are default values, several of the keys listed are not present in the BIG/db database.*

The **default.txt** file is intended to serve as documentation only. Some of the records, such as those that represent IP addresses and port numbers, need to be set to values other than the default values for the system to work. Additionally, some of the key names listed are wildcard keys. These keys are not valid key names.

If you want to load **default.txt** into the BIG/db database, it is recommended that you dump the existing database to another text file. Make a copy of **default.txt**, and then edit the copy so that the records which are present in your dump file match the values contained in the default.txt file. After the values match, you can load the edited copy of **default.txt**.

For a complete list of the keys available in the BIG/db, see the **Reference Guide**, *BIG/db Configuration Keys*.



# 19

---

## Configuring SNMP

---

- Working with SNMP on the BIG-IP® Controller
- Configuring SNMP on the BIG-IP Controller
- Understanding configuration file requirements



## Working with SNMP on the BIG-IP Controller

This chapter covers the management and configuration tasks for the simple network management protocol (SNMP) agent and management information bases (MIBs) available with the BIG-IP Controller.

---

### ◆ **Note**

*The SNMP agent must be configured on the BIG-IP Controller in order to use the F5 Networks SEE-IT Network Manager.*

The BIG-IP SNMP agent and MIBs allow you to manage the BIG-IP Controller by configuring traps for the SNMP agent or polling the controller with your standard network management station (NMS).

You can configure the BIG-IP SNMP agent to send traps to your management system with the Configuration utility. You can also set up custom traps by editing several configuration files.

Security options are available that let you securely manage information collected by the BIG-IP SNMP agent, including Community names, TCP wrappers, and View access control mechanism (VACM).

## Configuring SNMP on the BIG-IP Controller

There are seven basic tasks you must complete in order to use SNMP on the BIG-IP Controller.

### ❖ **Download the MIBs**

Download the BIG-IP MIBs and load them into your network management station.

### ❖ **Set up administrative access**

Configure `/etc/hosts.allow` to allow administrative access to the SNMP agent.

❖ **Configure `snmpd.conf`**

This file configures the SNMP agent. You can configure this file with the Configuration utility, or by editing it directly with a text editor.

❖ **Configure `rc.local`**

Configure `/etc/rc.local` to automatically start the SNMP agent.

❖ **Configure `snmptrap.conf`**

The configuration in `/etc/snmptrap.conf` determines which messages generate traps and what those traps are.

❖ **Configure `syslog.conf`**

Configure `/etc/syslog.conf` to pipe specified message types through `checktrap.pl`.

❖ **Configure `checktrap.pl`**

The BIG-IP Controller generates SNMP traps by piping `syslog` messages through the `/sbin/checktrap.pl`.

## Downloading the MIBs

The BIG-IP platform includes a private BIG-IP SNMP MIB. This MIB is specifically designed for use with the BIG-IP Controller. You can configure the SNMP settings in the Configuration utility, or on the command line.

SNMP management software requires that you use the MIB files associated with the device. You may obtain two MIB files from the BIG-IP directory `/usr/contrib/f5/mibs`, or you can download the files from the **Additional Software Downloads** section of the Configuration utility home page.

❖ **LOAD-BAL-SYSTEM-MIB.txt** This is a vendor MIB that contains specific information for properties associated with specific F5 functionality (load balancing, NATs, and SNATs)

❖ **UCD-SNMP-MIB.txt** This is a MIB-II (RFC 1213) that provides standard management information.

For information about the objects defined in the **LOAD-BAL-SYSTEM-MIB.txt**, refer to the descriptions in the object identifier (OID) section of the MIB file. For information about the objects defined in **UCD-SNMP-MIB.txt**, refer to RFC 1213.

## Understanding configuration file requirements

You need to make changes to several configuration files on the BIG-IP Controller before you use the SNMP agent. Once you change these configuration files, you need to restart the SNMP agent.

### `/etc/hosts.deny`

This file must be present to deny by default all UDP connections to the SNMP agent. The contents of this file are as follows:

```
ALL : ALL
```

### `/etc/hosts.allow`

The **`/etc/hosts.allow`** file is used to specify which hosts are allowed to access the SNMP agent. There are two ways to configure access to the SNMP agent with the **`/etc/hosts.allow`** file. You can type in an IP address, or list of IP addresses, that are allowed to access the SNMP agent, or you can type in an IP address and mask to allow a range of addresses in a subnetwork to access the SNMP agent.

For a specific list of addresses, type in the list of addresses you want to allow to access the SNMP agent. Addresses in the list must be separated by blank space or by commas. The basic syntax is as follows:

```
daemon: <IP address> <IP address> <IP address>
```

For example, you can type the following line which sets the SNMP agent to accept connections from the IP addresses specified:

```
bigsnmpd: 128.95.46.5 128.95.46.6 128.95.46.7
```

For a range of addresses, the basic syntax is as follows, where **`daemon`** is the name of the daemon, and **`IP/MASK`** specifies the network that is allowed access. The **`IP`** must be a network address:

```
daemon: IP/MASK
```

For example, you might use the following line which sets the **bigsnmpd** daemon to allow connections from the **128.95.46.0/255.255.255.0** address:

**bigsnmpd: 128.95.46.0/255.255.255.0**

The example above allows the 254 possible hosts from the network address **128.95.46.0** to access the SNMP daemon. Additionally, you may use the keyword **ALL** to allow access for all hosts or all daemons.

### **To allow access to the SNMP agent in the Configuration utility**

1. In the navigation pane, click **SNMP**.  
The SNMP Configuration screen opens.
2. In the BIG-IP SNMP Configuration screen, check **Enabled** to allow access to the BIG-IP SNMP agent.
3. In the Client Access Allow list section, type the following information:
  - **IP Address or Network Address**  
Type in an IP address or network address from which the SNMP agent can accept requests. Click the add (>>) button to add the address to the Current List. For a network address, type in a netmask.
  - **Netmask**  
If you type a network address in the IP Address or Network Address box, type the netmask for the network address in this box. Click the add (>>) button to add the network address to the Current List.
4. Click the **Apply** button.

## **/etc/snmpd.conf**

The **/etc/snmpd.conf** file controls most of the SNMP agent. This file is used to set up and configure certain traps, passwords, and general SNMP variable names. A few of the necessary variables are listed below:



**❖ System Contact Name**

The System Contact is a MIB-II simple string variable defined by almost all SNMP boxes. It usually contains a user name, as well as an email address. This is set by the **syscontact** key.

**❖ Machine Location (string)**

The Machine Location is a MIB-II variable that almost all boxes support. It is a simple string that defines the location of the box. This is set by the **syslocation** key.

**❖ Community String**

The community string clear text password is used for basic SNMP security. This also maps to VACM groups, but for initial read/only access, it is limited to only one group.

**❖ Trap Configuration**

Trap configuration is controlled by these entries in the **/etc/snmpd.conf** file:

- **trapsink <host>**  
This sets the host to receive trap information. The **<host>** is an IP address.
- **trapport <port>**  
This sets the port on which traps are sent. There must be one **trapport** line for each **trapsink** host.
- **trapcommunity <community string>**  
This sets the community string (password) to use for sending traps. If set, it also sends a trap upon startup: **coldStart(0)**.
- **authtrapenable <integer>**  
Setting this variable to **1** enables traps to be sent for authentication warnings. Setting it to **2** disables it.
- **data\_cache\_duration <seconds>**  
This is the time in seconds data is cached. The default value for this setting is one second.

---

**◆ Note**

*To change the trap port, the **trapport** line must precede the **trapsink** line. If you use more than one **trapport** line, then there must be one **trapport** line before each **trapsink** line. The same follows for*

***trapcommunity.** If you use more than one **trapcommunity** line, then there must be one **trapcommunity** line before each **trapsink** line.*

### **To set SNMP properties in the Configuration utility**

1. Click **SNMP** in the navigation pane.  
The SNMP Configuration screen opens.
2. To enable the SNMP agent, click the **Enable** box.
3. In the Client Access Allow List, type an IP address or network address from which the SNMP agent can accept requests. Click the add (>>) button to add the address to the Current List. For a network address, type in a netmask. To remove an IP address or network address from the list, click the address, and click the remove (<<) button.
4. In the System Information section, type the following information:
  - In the **System Contact** box, enter the contact name and email address for the person who should be contacted if this BIG-IP Controller generates a trap.
  - In the **Machine Location** box, enter a machine location, such as *First Floor*, or *Building 1*, that describes the physical location of the BIG-IP Controller.
  - In the **Community String** box, enter a community name. The community name is a clear text password used for basic SNMP security and for grouping machines that you manage.
5. In the Trap Configuration section, type the following information:
  - Check **Auth Trap Enabled** to allow traps to be sent for authentication warnings.
  - In the **Community** box, enter the community name to which this BIG-IP controller belongs. Traps sent from this box are sent to the management system managing this community.

- In the **Port** box, enter the community name to which this BIG-IP controller belongs. Traps sent from this box are sent to the management system managing this community.
  - In the **Trap** box, enter the host that should be notified when a trap is sent by the BIG-IP SNMP agent. After you type the **Community**, **Port**, and **Trap** for the trap sink, click the add (>) button to add it to the Current List.  
To remove a trap sink from the list, click the trap sink you want to remove, and click the remove (<) button.
6. Click the **Apply** button.

## /etc/rc.local

The following entry in the **/etc/rc.local** automatically starts up the SNMP agent when the system boots up (Figure 19.1).

```
# BIG-IP SNMP Agent
if [ -f /etc/snmpd.conf ]; then
    /sbin/bigsnmpd -c /etc/snmpd.conf
fi
```

**Figure 19.1** Starting the SNMP agent in the **/etc/rc.local** file.

If the **/etc/snmpd.conf** is present on your system, the SNMP agent is automatically started.

## /etc/snmptrap.conf

This configuration file includes OID, trap, and regular expression mappings. The configuration file specifies whether to send a specific trap based on a regular expression. An excerpt of the configuration file is shown in Figure 19.2.

```
# Default traps.
.1.3.6.1.4.1.3375.1.1.110.2.6 (ROOT LOGIN) ROOT LOGIN
.1.3.6.1.4.1.3375.1.1.110.2.5 (denial) REQUEST DENIAL
.1.3.6.1.4.1.3375.1.1.110.2.4 (BIG/ip Reset) SYSTEM RESET
.1.3.6.1.4.1.3375.1.1.110.2.3 (Service detected UP) SERVICE UP
.1.3.6.1.4.1.3375.1.1.110.2.2 (Service detected DOWN) SERVICE DOWN
#.1.3.6.1.4.1.3375.1.1.110.2.1 (error) Unknown Error
#.1.3.6.1.4.1.3375.1.1.110.2.1 (failure) Unknown Failure
```

**Figure 19.2** Excerpt from the `/etc/snmptrap.conf` file

Some of the OIDs have been permanently mapped to BIG-IP specific events. The OIDs that are permanently mapped for the BIG-IP Controller include:

- ❖ Root login
- ❖ Request denial
- ❖ System reset
- ❖ Service up
- ❖ Service down

You may, however, insert your own regular expressions and map them to the 110.1 OID. The `/etc/snmptrap.conf` file contains two examples for mapping your own OIDs:

- ❖ Unknown error
- ❖ Unknown failure

By default, the lines for these files are commented out. Use these OIDs for miscellaneous events. When lines match your expression, they are sent to your management software with the 110.2.1 OID.

## Syslog

In order to generate traps, you must configure **syslog** to send syslog lines to **checktrap.pl**. If the syslog lines make a match to the specified configuration in the **snmptrap.conf** file, a valid SNMP trap is generated. The following lines in the **/etc/syslog.conf** file require the **syslog** look at information logged, scan the **snmptrap.conf** file, and determine if a trap should be generated:

```
local0.* | exec /sbin/checktrap.pl.  
local1.* | exec /sbin/checktrap.pl.  
auth.* | exec /sbin/checktrap.pl.
```

---

### ◆ Note

*If you uncomment these lines, make sure you restart **syslogd**. For more information about working with the **Syslog** utility, see **Working with the Syslog utility** on page 18-19.*



---

---

# 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 3-DNS 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 is 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 *cool*, *hot*, and *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**

In products that support BIG-IP Cache, the **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 3-DNS Controller uses for load balancing.

**hot**

In products that support BIG-IP Cache, **hot** is 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 3-DNS 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 3-DNS Controllers. The iQuery protocol is officially registered for port 4353.

**last hop**

A last hop is the previous 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**

Node status indicates 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 server**

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**

In products that support caching, **stripes** are cacheable content subsets distributed among your cache servers.

**transparent cache server**

A cache server that can intercept requests destined for a web server, but are incapable of receiving 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.



---

# Index

---



/etc/bigd.conf file  
     configuring 17-33–17-35  
     data 17-28, 17-41  
 /etc/bigip.conf file 17-41  
     defining pools in 17-5  
     setting time-out in 3-33, 7-5  
 /etc/hosts.allow file 17-41, 19-3  
 /etc/hosts.deny file 17-41  
 /etc/ipfw.conf file 17-41  
 /etc/ipfw.filt file 17-41  
 /etc/ipfwrate.conf file 17-41  
 /etc/ipfwrate.filt file 17-41  
 /etc/netstart script  
     changing IP addresses 6-4  
     supporting networks 16-2  
     using VLAN tags 16-1, 16-4  
 /etc/rateclass.conf file 17-41  
 /etc/snmpd.conf file 17-42, 19-4  
 /etc/snmptrap.conf file 19-8  
 /etc/syslog.conf file 18-19  
  
 802.1q VLAN Trunk mode. See VLAN Trunk mode

## A

access levels 18-26  
 active units 17-41  
 address translation  
     See also SNATs  
 administrative IP addresses 6-4, 6-5  
 ARP 17-43  
 ARP caches  
     updating 18-21

ARP protocol 7-5  
 attributes  
     optional 12-9, 13-10, 14-9  
     selecting 12-2, 13-2, 14-2  
 audit trails  
     for reset events 18-6

## B

BIG/db configuration keys 18-29  
 BIG/db database  
     and VLAN tags 16-2, 16-4  
     creating and editing 18-28  
     defined 18-1, 18-27  
 BIG/db database keys 18-29  
 BIG/db keys  
     modifying 18-27  
 BIG-IP Controller types I-9  
 BIG-IP system log 18-25  
 BIG/pipe utility I-4, 18-1  
 BIG/stat command line options 18-14  
 BIG/stat utility  
     customizing 18-14  
     defined 18-1, 18-14  
 BIG/top command 18-1  
 BIG/top command line options 18-18  
 BIG/top runtime commands 18-19  
 BIG/top utility  
     described I-4, 18-17  
 bigdba commands 18-28  
 bigdba utility  
     defined 18-27  
 bigdba utility commands 18-28  
 BIG-IP Controller passwords  
     changing 18-25  
 BIG-IP Controller removal 18-21  
 BIG-IP system log 18-25

- BIG-IP web server account
  - deleting 18-27
- BIG-IP web server passwords
  - changing 18-25
- bigpipe conn command 18-25
- bigpipe global command 18-9
- bigpipe interface command
  - for destination processing 12-16, 13-18
  - for interface settings 16-3, 18-10
  - for source and destination processing 3-26, 3-42, 3-45
  - for source processing 13-17
  - using 18-10
- bigpipe ipalias command
  - and VLAN tags 16-1, 16-4
- bigpipe maint command 18-21
- bigpipe nat command 18-8
- bigpipe node command 18-7
  - for remote nodes 13-19
- bigpipe pool command
  - for cache servers 12-6, 13-6, 14-6
  - for hot content 12-8, 13-8, 14-8
  - for origin servers 12-7, 13-7, 14-7
- bigpipe port command 18-8
- bigpipe rule command
  - for cache statement rules 13-12, 14-12
- bigpipe snat command 15-3, 18-9
  - for SNAT mapping 12-15, 13-15
- bigpipe summary command 18-2
- bigpipe vip command
  - and mapping 7-3
  - and statistics 18-6
  - creating servers 5-3
- bigstat command 18-14
- bigtop command 18-18
- bit activity
  - displaying 18-17
- bit statistics 18-2
- bit status 18-13

- broadcast addresses 7-2
- byte activity
  - displaying 18-17
- byte counters
  - resetting 18-6

## C

- cache
  - configuration 12-13
- cache configuration 13-14
- cache control rules
  - and intelligent cache population 12-13, 13-14
- cache memory
  - efficient use of 12-1, 13-1, 14-1
- cache population
  - speeding 12-2, 13-2
- cache server availability 12-4
- cache server content 12-2, 13-2, 14-2
- cache server efficiency
  - enhancing 12-1
- cache server groups
  - See hot pools
- cache server memory
  - maximizing 12-2, 13-2, 14-2
- cache server pools
  - defined 12-4
- cache server response 12-14
- cache server types 1-8, 12-1, 12-8, 13-2, 13-9, 14-8
- cache servers 2-4
  - and hot content 12-2, 13-2, 14-2
  - balancing load 12-5
  - creating pools for 13-5, 13-6
- cache statement rules
  - creating 12-11

- cache statements
  - contents of 12-8, 13-8, 14-8
  - examples 12-11, 13-12
  - nesting 12-8, 13-8, 14-8
- cache\_pool attribute
  - defined 12-9, 13-9, 14-9
- cache\_server pools
  - defined 14-4
- cacheable content determination
  - accessing 13-5
  - defined 1-8, 12-1, 13-1, 14-1
- cacheable content expressions 12-8, 13-9, 14-8
  - in cache rule statements 12-4, 13-5, 14-4
- cache-to-content association
  - See content affinity
- checktrap script
  - configuring options for 19-9
- client connections 6-5
- config\_failover command 17-40
- configuration 1-2
- configuration examples 2-1–2-4
  - Internet 1-1
  - intranet 2-1
- configuration tasks 17-1–17-4
- Configuration utility
  - configuring a pool 3-27, 3-38, 3-46, 3-48, 17-5
  - described 1-3, 18-1
  - resetting statistics 18-9
  - setting SNMP properties 19-6
- Configuration utility requirements 1-3
- connection count counter
  - resetting 18-6
- connection requests 17-8
- connection statistics 18-2
- connection status
  - displaying 18-10
- connection time-out values
  - See idle connection time-out values
- connections 17-7
  - adding more 5-1
  - and Maintenance mode 18-21
  - for virtual server/SNAT combinations 15-1
  - from clients 6-5
  - load balancing across nodes 15-1
  - load balancing to servers 6-2, 6-5
  - making FIN/ACK sequences 7-4
  - monitoring concurrent 18-1
  - outbound 5-2
  - See also internet connections
  - See also nPath routing
  - See also stale connections
  - viewing 18-24
- content
  - expired 13-14
- content affinity
  - accessing 13-5
  - defined 12-1, 13-1, 14-1
- content request frequency 12-2, 13-2, 14-2
- content requests
  - and hit\_period attribute 12-10, 13-10, 14-10
  - directing 13-14
  - from cache servers 14-4
  - receiving 14-1
  - routing 12-13
  - specifying minimum and maximum 12-10, 13-10, 14-10
  - via origin servers 13-14
- content retrieval 1-8, 12-1, 12-8, 13-2, 13-9, 14-8
- content subsets
  - See hot content subsets
- content types
  - for caching 1-8, 12-1, 13-1, 14-1
- content\_hash\_size 12-10, 13-11, 14-10
- content\_hash\_size attribute
  - defined 12-10, 13-11, 14-10
- controller synchronization 17-40, 17-41

cool content 12-2, 13-2, 14-2

## D

default routers 17-28, 17-43

default.txt file

- defined 18-27, 18-29
- dumping to text file 18-29
- loading 18-29
- location of 18-29

Destination addresses 2-4

destination IP addresses  
processing 5-1

destination packets 7-1

destination processing

- adding to interface 12-15
- and internet connections 5-1
- and outbound connections 5-2
- defined 12-15
- enabling with single interface 6-1, 6-2
- for virtual servers 7-2
- for VPN load balancing 4-3, 4-7
- via two interfaces 6-4

destination processing interfaces  
for SNAT/virtual server combinations  
15-2

destination translation 13-14

disable keyword 18-22

dropped connections  
viewing 18-24

## E

ECV 17-28

- normal 17-29
- regular expressions 17-30
- reverse 17-30
- SSL 17-29

efficiency  
enhancing 12-1

elapsed time  
viewing 18-24

email

sending 18-19

enable keyword 18-22

encrypted connections I-10

Extended Content Verification (ECV). See  
ECV

external interfaces 17-22  
and outbound connections 5-2  
for VPN load balancing 4-3, 4-7  
with single IP network 6-4

external network. See external interfaces

## F

fail-safe settings 17-40–17-44

FIN/ACK sequences 7-4

firewall load balancing

- balancing outbound traffic 9-1
- balancing traffic to enterprise servers  
using a firewall sandwich 8-1
- balancing two-way traffic using a firewall  
sandwich 11-1

First-Time Boot utility  
defined I-3

forward proxy caching tasks 14-3

F-Secure SSH client option  
and encrypted communications I-10  
as a remote shell I-4

FTP

on ports 17-13

Full Read/Write access level 18-26

## G

gateways

and nPath routing 7-4



global statistics  
resetting 18-6

## H

hardware maintenance  
performing 18-21

hot cache server pools  
defined 12-4

hot content  
and attributes 12-9, 13-10, 14-9  
and cache servers 12-2, 13-2, 14-2  
creating pools for 13-7, 13-8

hot content load balancing  
defined 12-1, 13-1, 14-1

hot content requests  
distributing 12-2, 13-2, 14-2  
redirecting 12-1, 13-1, 14-1

hot content subset 12-1, 13-1, 14-1

hot content subsets  
requesting 12-11, 13-11, 14-11  
specifying 12-10, 13-11, 14-10

hot pools  
defined 12-1, 13-1, 14-1

hot\_pool attribute  
defined 12-10, 13-10, 14-9

hot\_pool value  
specifying 12-11, 13-11, 14-10

HTTP cache servers  
and outbound traffic 15-3  
defining pools for 15-3

HTTP header variables 12-8, 13-9, 14-8

HTTP request headers  
and content caching 1-8, 12-1, 13-1, 14-1

## I

idle connection time-out values 3-33, 7-1, 7-4, 7-5  
See also TCP connection time-out values

See also UDP connection time-out values

ifconfig command  
displaying VLAN information 16-3

illegal connection attempt statistics  
viewing 18-24

In 1-2

inbound connections  
load balancing for 8-8, 10-8, 10-13  
load balancing for internet traffic 4-10  
load balancing for VPNs 4-3, 4-4

inbound routing 7-4

inbound traffic  
and VPNs 4-4  
configuring 5-3  
configuring virtual servers for 8-11, 10-10

intelligent cache population  
defined 1-8, 12-1, 12-8, 13-2, 13-9, 14-8

interface card status 18-10

interface cards 17-43

interface configuration  
and routing conflicts 6-4  
and SNAT addresses 6-5  
for internet connections 5-1  
for SNAT/virtual server combinations 15-2  
for source and destination processing 6-1  
for VPN load balancing 4-3, 4-7  
with single IP network 6-1, 6-2

interface types 12-15, 13-15

internal interfaces 17-22  
and outbound connections 5-2  
and VLAN tags 16-1, 16-4  
for VPN load balancing 4-3, 4-7  
with single IP network 6-4

internal IP addresses  
defining pools for 4-3  
replacing 4-4, 4-8

internal network. See internal interfaces

internal shared interfaces 12-13

- internet connections
  - adding more 5-1, 5-2
  - balancing load through routers 4-5
  - example 5-1
- Internet content
  - storing 14-1
- Internet content caching
  - illustrated 14-3
- IP address translation
  - enabling and disabling 7-1, 7-3
  - preventing 7-1
- IP addresses
  - and loopback interface 7-5
  - and nPath routing 7-4
  - defining I-3
  - destination 17-8
  - filtering 2-4
  - for virtual servers 7-2
  - See administrative IP addresses
  - setting up 6-4
- IP aliases
  - and nPath routing 7-4
- IP forwarding
  - setting up 17-21, 17-27–17-28
  - system control variables 3-34, 17-28
- IP network topology
  - with single interface 6-1, 16-1
  - with two interfaces 6-3
- IP networks
  - and routing conflicts 6-5
  - configuring dual interfaces 6-4
- IP packet filter statistics
  - viewing 18-24
- IP packet filters
  - and illegal connection attempts 18-24
- IP packets
  - and address translation 7-1
  - recognition by clients 6-2
  - routing incorrectly 7-5
- ISP load balancing 5-3

## K

- keys
  - in BIG/db database 18-27, 18-29

## L

- last hop pools 4-10
  - and inbound configurations 5-3, 8-11, 10-10
  - defining for VPNs 4-3
- less file page utility 18-19
- load balancing
  - and ECV service checks 17-28
  - and transparent devices 4-5
  - configuring I-3
  - for inbound connections
  - for internet connections 5-1
  - for outbound connections 4-9, 5-2
  - monitoring I-3
  - transparent nodes 17-8
- load balancing mode
  - global 17-19
- load balancing pool types
  - described 12-4, 14-4
  - listed 13-4
- load balancing pools
  - creating for internet connections 5-2
  - defining 6-2, 17-5
  - defining for HTTP cache servers 15-3
  - defining for routers 4-8
  - defining for traffic origination 4-8
  - defining for virtual server requests 4-8, 6-5
  - defining for VPNs 4-3, 4-7
  - for cache servers 12-5, 13-5, 14-5
  - for hot content 13-7, 14-7
  - for origin servers 12-6, 13-6, 14-6
- load balancing requests 12-2, 13-2, 14-2
- local server acceleration
  - illustrated 12-3
  - setting up 12-1

log files  
     viewing 18-25

log messages  
     samples of 18-20

logging  
     via Syslog utility 18-19

loopback interface  
     and IP addresses 7-5  
     described 7-5

## M

Maintenance mode  
     activating 18-21

member lists  
     adding 7-2

member node status  
     displaying 18-10

memory efficiency  
     affecting 12-1, 13-1, 14-1

MIB I-3

mirroring. See redundant systems

miss requests  
     initiating 12-13, 13-14

monitoring methods  
     and command-line utilities I-4

MS Loopback interface 7-5

## N

NAT statistics  
     resetting 18-6, 18-8  
     viewing 18-24

NAT status  
     displaying 18-13  
     viewing 18-14

NATs  
     administrative connections 1-4  
     configuring 17-21–17-24  
     defining 17-23

netmask 7-2

netstat utility  
     and VLAN tags 16-3

network adaptor list 7-5

network address translations. See NATs

network configurations  
     802.1q VLAN trunk mode 16-1  
     IP network topology 6-1, 6-3, 16-1  
     ISP load balancing 5-3  
     SNAT and virtual servers 6-1, 16-1  
     VPN and router load balancing 4-5, 4-10

Network Interface Cards. See NICs (Network Interface Cards)

network traffic 17-43  
     and additional connections 5-1  
     and pools 6-2, 6-5  
     managing 7-1  
     optimizing 6-4  
     receiving 6-4, 6-5  
     See also inbound traffic  
     See also outbound traffic

network traffic statistics  
     viewing 18-11, 18-12

NICs (Network Interface Cards)  
     viewing settings for 3-25, 3-41, 3-45,  
     8-14, 10-18, 11-16, 12-15, 13-16,  
     13-18

node address statistics  
     resetting 18-6, 18-7  
     viewing 18-2, 18-24

node address status  
     displaying 18-12

node addresses 17-22, 17-23  
     enabling and disabling 18-23  
     removing from service 18-21, 18-23

node configuration 17-22–17-28

node ping timer 17-14

node ports  
     defining 7-2

- node server statistics
  - resetting 18-6, 18-7
- node statistics
  - monitoring 18-1
  - viewing 18-2, 18-24
- node status
  - displaying 18-12
  - viewing 18-14
- nodes
  - enabling and disabling 18-23
  - removing from service 18-20, 18-23
  - viewing 18-23
- non-cacheable content
  - defined 14-4
- non-cacheable content requests 12-4, 13-5
- non-transparent cache servers
  - described 1-8, 12-1, 12-8, 13-2, 13-9, 14-8
- nPath mode 7-5
- nPath routing 7-1
  - and IP address translation 7-1
  - defining virtual servers 7-1, 7-4
  - example 7-6
  - setting idle connection timeout values 7-4
  - setting up 7-4
  - using 7-1

## O

- origin server nodes
  - marking as remote 13-14, 13-18
- origin server pools
  - defined 12-4
- origin server response 12-14
- origin servers
  - as router 14-6
  - balancing load 12-4, 12-6
  - creating pools for 13-6, 13-7, 14-6
  - defined 12-4

- origin web server load
  - minimizing 12-2, 13-2
- outbound connections
  - load balancing 5-2, 8-8, 10-8, 10-13
  - load balancing for internet traffic 4-10
  - load balancing for VPNs 4-3, 4-4, 4-9
  - translating 15-3
- outbound routing
- outbound throughput
  - increasing
- outbound traffic 5-3
  - and VPNs 4-5
  - configuring controllers for 15-3
- outbound transparent devices
  - configuring 15-1

## P

- packet activity
  - displaying 18-17
- packet counters
  - resetting 18-6
- packet rejection 6-5
- packet statistics 18-2
- packet status 18-13
- packets
  - monitoring 18-1
  - viewing 18-24
- pager notifications
  - activating 18-19
- Partial Read/Write access level 18-26
- passwords
  - and BIG-IP web server 18-26
  - changing 18-25
- performance statistics
  - displaying 18-2
  - summary table 18-4
- persistence 1-3
  - configuring 17-35

- for connections 17-7
  - simple 17-38
  - SSL 17-37
- Pinger log 18-25
- pool types
  - See load balancing pool types
- pools
  - persistence 17-37
- pools. See load balancing pools
- port statistics
  - resetting 18-8
- port translation 4-5, 15-3
- ports
  - access to 17-12
  - service checking 17-18–17-19
  - wildcard 17-9–17-12
- private networks
  - connecting 4-5
- procedures
  - configuring NATs 17-24
  - configuring persistence for virtual servers 17-38–17-39
  - configuring SNAT address mappings 17-26
  - configuring SNAT global properties 17-25
  - configuring SSL persistence 17-37–17-38
  - defining virtual server mappings 17-10
  - setting idle connections 17-15
  - setting IP forwarding 3-33, 17-27
  - setting node ping timer 17-14
  - setting service check timer 17-17
  - setting up ECV service checking 17-32
  - turning off port translation 17-10
- processing power
  - maximizing 12-2, 13-2, 14-2
- R**
- rate filter statistics
  - viewing 18-24
- Ratio mode 17-19, 17-20
- Read Only access level 18-26
- real-time statistics
  - displaying 18-17
- receive expressions 17-31
- reconfig-httpd utility
  - running 18-27
- redundant controllers 8-8, 10-8, 10-13
- redundant systems 17-40–17-44
  - active unit 17-41
  - adding shared addresses 16-3
  - configuring hop pools 3-49
  - configuring interfaces on 5-1
  - fail-safe interfaces 17-44
  - mirroring 2-4
  - standby unit 17-41
  - using VLAN tags 16-2
- refresh interval
  - resetting 18-18
- remote server acceleration
  - illustrated 13-3
- remote server acceleration tasks
  - listed 13-4
- remote vs.local acceleration
  - compared 13-1
- requests
  - directing 12-15
- response
  - ensuring 13-14
- response time
  - improving 12-1
- root password 17-41
  - defining 1-3
- round robin 17-19
- routers
  - See also nPath routing
- routes
  - defining for nPath routing 7-4

- routing
  - for throughput optimization 6-4
  - via origin servers 14-6
- routing conflicts 6-4, 6-5
- routing table 17-28
- rule command
  - for cache statement rules 12-12

## S

- secure connections I-10
- secure network address translation (SNAT). See SNATs
- Secure Network Address Translations. See SNATs
- secure network address translations. See SNATs
- security
  - and illegal connection attempts 18-24
  - changing passwords 18-26
- send strings 17-31
- service checks
  - configuring 17-28–17-35
  - EAV 17-17
  - ECV 17-17, 17-28, 17-29
  - simple 17-17
- services 17-12
  - monitoring 18-10
- services status
  - viewing 18-14
- shared internal interfaces 13-14
- shared IP addresses
  - and route creation 6-5
  - in BIG/db database 16-2
- shared IP aliases
  - adding VLAN tags to 16-1
- simple keyword 17-18
- SNAT address mappings
  - configuring 13-15

- SNAT addresses 6-5
  - and interfaces 6-5
- SNAT connections
  - initiating 13-18
  - showing 18-13
- SNAT settings
  - printing 18-13
- SNAT source translations
  - configuring 6-1, 15-1, 16-1
- SNAT statistics
  - resetting 18-9
  - viewing 18-24
- SNAT status
  - displaying 18-13
- SNATs
  - address mappings 17-26
  - connection limits 17-24
  - defining 17-24–17-26
  - global properties 17-24
  - TCP idle connection timeout 17-25
  - UDP idle connection timeout 17-25
- SNATs (Secure Network Address Translations)
  - defined 12-14
  - defining 6-2
- SNMP
  - /etc/hosts.allow file 19-3
  - /etc/hosts.deny file 19-3
  - client access 19-5
  - configuring 19-1
  - downloading MIBs 19-2
  - in the Configuration utility 19-4
- SNMP MIB I-3
- SNMP OIDs 19-8
- SNMP protocol
  - configuring 19-1
- SNMP syslog 19-9
- SNMP trap configuration 19-5
- software configuration 17-1–17-4
- source address translation
  - and SNATs 6-2, 6-5

- source IP addresses 17-22
    - changing to SNAT addresses 6-5, 15-1
    - processing 5-1
  - source processing
    - defined 12-15, 13-16
    - enabling 5-1, 5-2
    - enabling with single interface 6-1, 6-2
    - for VPN load balancing 4-3, 4-7
    - via two interfaces 6-4
  - source processing interfaces 15-2
  - source translation 13-14
    - configuring for SNATs 15-1
  - SSH client option
    - See F-Secure SSH client option
  - SSL Accelerator
    - additional configuration options 3-18
    - configuring 3-3
    - configuring with certificates and keys 3-10
    - creating an HTTP virtual server 3-12
    - creating an SSL Gateway 3-13
    - deleting 3-16
    - disabling 3-16
    - enabling 3-16
    - hardware acceleration 3-2
    - last hop pool with additional network devices 3-19
    - obtaining certificates and keys 3-3
    - view configuration information 3-17
  - SSL accelerator cell
    - configuration tasks 3-24
    - configuring an SSL accelerator for use in a cell 3-30
    - configuring the BIG-IP Controller for the SSL accelerator cell 3-25
    - setting the default route on each node in a cell 3-34
  - SSL accelerator half sandwich 3-35
    - configuration tasks 3-36
    - configuring each SSL accelerator 3-41
    - configuring the BIG-IP Controller that load balances the content servers 3-44
    - configuring the BIG-IP Controllers handling inbound traffic 3-38
    - configuring the content servers 3-50
  - SSL connections I-10
  - stale connections
    - removing
  - standby units 17-41
  - statistical displays
    - customizing 18-14
  - statistics
    - monitoring 18-1
    - resetting 18-6
    - resetting global 18-9
    - viewing 18-24
  - synchronization. See controller synchronization
  - syslog file entries
    - generating 18-6
  - Syslog utility 18-1, 18-19, 19-9
  - system control variables
    - setting 18-6
  - system log files
    - viewing 18-25
  - system setup I-3
  - system statistics
    - monitoring 18-1
- ## T
- TCP
    - setting idle connection timers 17-16
    - traffic 17-15
  - TCP connections 3-33, 7-5
  - throughput
    - optimizing 6-2, 6-4
  - timer settings
    - configuring 17-13–17-19

- idle connections 17-15–17-16
    - node ping 17-14–17-15
    - service check 17-17–17-18
  - transparent cache servers
    - described 12-1, 13-2
  - transparent devices 4-5
  - transparent network devices 17-6, 17-8
  - transparent nodes 17-8, 17-9
  - transparent virtual servers
    - setting up 7-3
- ## U
- UDP
    - setting idle connection timers 17-16
    - traffic 17-15
  - user accounts
    - creating 18-26
    - deleting 18-27
  - user IDs
    - adding 18-26
  - users
    - creating new 18-26
  - utilities I-4
- ## V
- verbose keyword 18-13
  - virtual address statistics 18-2
    - resetting 18-6
    - viewing 18-24
  - virtual addresses
    - enabling and disabling 18-22
    - monitoring 18-10
    - removing from service 18-21, 18-22
  - virtual port statistics 18-2
    - resetting 18-6, 18-8
    - viewing 18-11, 18-24
  - virtual ports
    - allowing 18-23
    - enabling and disabling 18-23
    - removing from service 18-21, 18-22
  - virtual server configuration
    - for VPN load balancing 4-3
  - virtual server mappings
    - adding or removing nodes 7-3
    - creating 7-3
    - defining standard 3-28, 3-39, 3-47, 17-7–17-8
    - defining wildcard 17-10–17-12
    - included nodes 18-23
  - virtual server statistics
    - monitoring 18-1
    - resetting 18-6
    - viewing 18-2, 18-11, 18-24
  - virtual server status
    - displaying 18-10
    - viewing 18-14
  - virtual servers
    - adding member lists 7-2
    - additional features 17-7
    - and destination processing 7-2
    - and firewall sandwiches 8-8, 10-8, 10-13
    - and last hop pools 5-3
    - and inbound configurations 4-10
    - and pools 5-3
    - and SNATs 6-1, 16-1
    - components 1-2
    - configuring 17-6–17-12
    - configuring hop pools 3-49
    - connecting to cache servers 15-3
    - defining for VPNs 4-3, 4-4
    - defining standard 17-6, 17-8
    - defining wildcard 17-6, 17-8–17-12
    - enabling and disabling 18-22
    - for traffic distribution 12-12, 13-12
    - forwarding 17-22
    - handling requests to 6-5
    - mapping to IP addresses 7-2
    - monitoring 18-10
    - persistence 17-38



- removing from service 18-21, 18-22
- responding to IP addresses 7-5
- routing network traffic 6-5
- See also inbound connections
- See also outbound connections
- sending traffic to pools 6-5
- setting up transparent 7-3
- using last hop pool 8-11, 10-10
- viewing 18-23

#### VLAN IDs

- supporting 16-2

#### VLAN tag definitions

- extending 16-1

#### VLAN tag values

- adding 16-4

#### VLAN tags

- displaying 16-3
- enabling and disabling 16-3, 16-4
- showing settings for 16-3
- using 16-1

#### VLAN Trunk mode

- setting up 16-1

#### VLAN trunk mode

- adding tag definitions 16-2
- configuring multiple VLANs 16-2
- setting up 16-4
- using ifconfig 16-3

#### VPN and router load balancing

- configuring 4-5, 4-10

#### VPN load balancing

- configuring 4-3, 4-7

## W

- web server access 1-3

- web servers 17-29

- wildcard keys 18-29

- wildcard virtual servers

- creating 3-32, 7-4
- for traffic forwarding 14-12