BIG-IP® TMOS®: Routing Administration

Version 11.6

Table of Contents

Legal Notices

Publication Date

This document was published on December 21, 2015.

Publication Number

MAN-0412-06

Copyright

Trademarks

Patents

This product may be protected by one or more patents indicated at:
http://www.f5.com/about/guidelines-policies/patents

Export Regulation Notice

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

RF Interference 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 has been tested and found to comply with the limits for a Class A digital device pursuant to Part 15 of FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This unit generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful 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 ICES-003.

Standards Compliance

This product conforms to the IEC, European Union, ANSI/UL and Canadian CSA standards applicable to Information Technology products at the time of manufacture.

Acknowledgments

This product includes software developed by Bill Paul.

This product includes software developed by Jonathan Stone.

This product includes software developed by Manuel Bouyer.

This product includes software developed by Paul Richards.

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

This product includes software developed by the Politecnico di Torino, and its contributors.

This product includes software developed by the Swedish Institute of Computer Science and its contributors.

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 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 the University of Vermont and State Agricultural College and Garrett A. Wollman.

This product includes software developed by Balazs Scheidler (bazsi@balabit.hu), which is protected under the GNU Public License.

This product includes software developed by Niels Mueller (nisse@lysator.liu.se), which is protected under the GNU Public License.

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).

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

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

This product includes software developed by Jared Minch.

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/).

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com).

This product contains software based on oprofile, which is protected under the GNU Public License.

This product includes software with glib library utility functions, which is protected under the GNU Public License.

This product includes software with grub2 bootloader functions, which is protected under the GNU Public License.

This product includes software with the Intel Gigabit Linux driver, which is protected under the GNU Public License. Copyright ©1999 - 2012 Intel Corporation.

This product includes software with the Intel 10 Gigabit PCI Express Linux driver, which is protected under the GNU Public License. Copyright ©1999 - 2012 Intel Corporation.

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

This product contains software licensed from Dr. Brian Gladman under the GNU General Public License (GPL).

This product includes software developed by the Apache Software Foundation (http://www.apache.org/).

This product includes Hypersonic SQL.

This product contains software developed by the Regents of the University of California, Sun Microsystems, Inc., Scriptics Corporation, and others.

This product includes software developed by the Internet Software Consortium.

This product includes software developed by Nominum, Inc. (http://www.nominum.com).

This product contains software developed by Broadcom Corporation, which is protected under the GNU Public License.

This product contains software developed by MaxMind LLC, and is protected under the GNU Lesser General Public License, as published by the Free Software Foundation.

This product includes software developed by Andrew Tridgell, which is protected under the GNU Public License, copyright ©1992-2000.

This product includes software developed by Jeremy Allison, which is protected under the GNU Public License, copyright ©1998.

This product includes software developed by Guenther Deschner, which is protected under the GNU Public License, copyright ©2008.

This product includes Intel QuickAssist kernel module, library, and headers software licensed under the GNU General Public License (GPL).

This product includes gd-libgd library software developed by the following in accordance with the following copyrights:

- Portions copyright ©1994, 1995, 1996, 1997, 1998, 2000, 2001, 2002 by Cold Spring Harbor Laboratory. Funded under Grant P41-RR02188 by the National Institutes of Health.
- Portions copyright ©1996, 1997, 1998, 1999, 2000, 2001, 2002 by Boutell.Com, Inc.
- Portions relating to GD2 format copyright ©1999, 2000, 2001, 2002 Philip Warner.
- Portions relating to PNG copyright ©1999, 2000, 2001, 2002 Greg Roelofs.
- Portions relating to gdttf.c copyright ©1999, 2000, 2001, 2002 John Ellson (ellson@lucent.com).
- Portions relating to gdft.c copyright ©2001, 2002 John Ellson (ellson@lucent.com).
- Portions copyright ©2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007 2008 Pierre-Alain Joye (pierre@libgd.org).
- Portions relating to JPEG and to color quantization copyright ©2000, 2001, 2002, Doug Becker and copyright ©1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, Thomas G. Lane. This software is based in part on the work of the Independent JPEG Group.
- Portions relating to WBMP copyright 2000, 2001, 2002 Maurice Szmurlo and Johan Van den Brande. Permission has been granted to copy, distribute and modify gd in any context without fee, including a commercial application, provided that this notice is present in user-accessible supporting documentation.

This product includes software developed by Oracle America, Inc. Copyright ©2012.

1. Java Technology Restrictions. Licensee shall not create, modify, change the behavior of, or authorize licensees of licensee to create, modify, or change the behavior of, classes, interfaces, or subpackages that are in any way identified as "java", "javax", "sun" or similar convention as specified by Oracle in any naming convention designation. In the event that Licensee creates an additional API(s) which: (a) extends the functionality of a Java Environment; and (b) is exposed to third party software developers for the purpose of developing additional software which invokes such additional API, Licensee must promptly publish broadly an accurate specification for such API for free use by all developer.
2. Trademarks and Logos. This License does not authorize an end user licensee to use any Oracle America, Inc. name, trademark, service mark, logo or icon. The end user licensee acknowledges that Oracle owns the Java trademark and all Java-related trademarks, logos and icon including the Coffee Cup and Duke ("Java Marks") and agrees to: (a) comply with the Java Trademark Guidelines at http://www.oraclc.com/html/3party.html; (b) not do anything harmful to or inconsistent with Oracle's rights in the Java Marks; and (c) assist Oracle in protecting those rights, including assigning to Oracle any rights acquired by Licensee in any Java Mark.
3. Source Code. Software may contain source code that, unless expressly licensed for other purposes, is provided solely for reference purposes pursuant to the terms of your license. Source code may not be redistributed unless expressly provided for in the terms of your license.
4. Third Party Code. Additional copyright notices and license terms applicable to portion of the Software are set forth in the THIRDPARTYLICENSEREADME.txt file.
5. Commercial Features. Use of the Commercial Features for any commercial or production purpose requires a separate license from Oracle. "Commercial Features" means those features identified in Table I-I (Commercial Features In Java SE Product Editions) of tile Software documentation accessible at http://www.oracle.com/technetwork/java/javase/documentation/index.html.

This product includes utilities developed by Linus Torvalds for inspecting devices connected to a USB bus.

This product includes perl-PHP-Serialization software, developed by Jesse Brown, copyright ©2003, and distributed under the Perl Development Artistic License (http://dev.perl.org/licenses/artistic.html).

This product includes software developed by members of the CentOS Project under the GNU Public License, copyright ©2004-2011 by the CentOS Project.

OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

This product includes software developed by Douglas Gilbert. Copyright ©1992 - 2012 The FreeBSD Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE FREEBSD PROJECT "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE FREEBSD PROJECT OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The views and conclusions contained in the software and documentation are those of the authors and should not be interpreted as representing official policies, either expressed or implied, of the FreeBSD Project.

This product includes software developed as open source software. Copyright ©1994 - 2012 The FreeBSD Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com). Copyright ©1998 - 2011 The OpenSSL Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (http://www.openssl.org/)"

4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact openssl-core@openssl.org.
5. Products derived from this software may not be called "OpenSSL" nor may "OpenSSL" appear in their names without prior written permission of the OpenSSL Project.
6. Redistributions of any form whatsoever must retain the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/)"

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes software licensed from William Ferrell, Selene Scriven and many other contributors under the GNU General Public License, copyright ©1998 - 2006.

This product includes software developed by Thomas Williams and Colin Kelley. Copyright ©1986 - 1993, 1998, 2004, 2007

Permission to use, copy, and distribute this software and its documentation for any purpose with or without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. Permission to modify the software is granted, but not the right to distribute the complete modified source code. Modifications are to be distributed as patches to the released version. Permission to distribute binaries produced by compiling modified sources is granted, provided you

1. distribute the corresponding source modifications from the released version in the form of a patch file along with the binaries,
2. add special version identification to distinguish your version in addition to the base release version number,
3. provide your name and address as the primary contact for the support of your modified version, and
4. retain our contact information in regard to use of the base software.

Permission to distribute the released version of the source code along with corresponding source modifications in the form of a patch file is granted with same provisions 2 through 4 for binary distributions. This software is provided "as is" without express or implied warranty to the extent permitted by applicable law.

This product includes software developed by the Computer Systems Engineering Group at Lawrence Berkeley Laboratory. Copyright ©1990-1994 Regents of the University of California. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgment: This product includes software developed by the Computer Systems Engineering Group at Lawrence Berkeley Laboratory.

4. Neither the name of the University nor of the Laboratory may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes software developed by Sony Computer Science Laboratories Inc. Copyright © 1997-2003 Sony Computer Science Laboratories Inc. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY SONY CSL AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL SONY CSL OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product contains software developed by Google, Inc. Copyright ©2011 Google, Inc.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

This product includes software developed by Jeremy Ashkenas and DocumentCloud, and distributed under the MIT license. Copyright © 2010-2013 Jeremy Ashkenas, DocumentCloud.

This product includes gson software, distributed under the Apache License version 2.0. Copyright © 2008-2011 Google Inc.

This product includes owasp-jave-encoder software, copyright © 2014, Jeff Ichnowski, and distributed under the New BSD license.

This product may include Intel SDD software subject to the following license; check your hardware specification for details.

1. LICENSE. This Software is licensed for use only in conjunction with Intel solid state drive (SSD) products. Use of the Software in conjunction with non-Intel SSD products is not licensed hereunder. Subject to the terms of this Agreement, Intel grants to You a nonexclusive, nontransferable, worldwide, fully paid-up license under Intel's copyrights to:

   - copy the Software onto a single computer or multiple computers for Your personal, noncommercial use; and
   - make appropriate back-up copies of the Software, for use in accordance with Section 1a) above.

   The Software may contain the software or other property of third party suppliers, some of which may be identified in, and licensed in accordance with, any enclosed "license.txt" file or other text or file.

   Except as expressly stated in this Agreement, no license or right is granted to You directly or by implication, inducement, estoppel or otherwise. Intel will have the right to inspect or have an independent auditor inspect Your relevant records to verify Your compliance with the terms and conditions of this Agreement.

2. RESTRICTIONS. You will not:

   a. copy, modify, rent, sell, distribute or transfer any part of the Software, and You agree to prevent unauthorized copying of the Software; and,
   b. reverse engineer, decompile, or disassemble the Software; and,
   c. sublicense or permit simultaneous use of the Software by more than one user; and,
   d. otherwise assign, sublicense, lease, or in any other way transfer or disclose Software to any third party, except as set forth herein; and,
   e. subject the Software, in whole or in part, to any license obligations of Open Source Software including without limitation combining or distributing the Software with Open Source Software in a manner that subjects the Software or any portion of the Software provided by Intel hereunder to any license obligations of such Open Source Software. "Open Source Software" means any software that requires as a condition of use, modification and/or distribution of such software that such software or other software incorporated into, derived from or distributed with such software:

      a. be disclosed or distributed in source code form; or
      b. be licensed by the user to third parties for the purpose of making and/or distributing derivative works; or
      c. be redistributable at no charge.

   Open Source Software includes, without limitation, software licensed or distributed under any of the following licenses or distribution models, or licenses or distribution models substantially similar to any of the following:

   a. GNU’s General Public License (GPL) or Lesser/Library GPL (LGPL),
   b. the Artistic License (e.g., PERL),
   c. the Mozilla Public License,
   d. the Netscape Public License,
   e. the Sun Community Source License (SCSL),
   f. vi) the Sun Industry Source License (SISL),
   g. vii) the Apache Software license, and
   h. viii) the Common Public License (CPL).

3. OWNERSHIP OF SOFTWARE AND COPYRIGHTS. Title to all copies of the Software remains with Intel or its suppliers. The Software is copyrighted and protected by the laws of the United States and other countries, and international treaty provisions. You may not remove any copyright notices from the Software. Intel may make changes to the Software, or to materials referenced therein, at any time and without notice, but is not obligated to support or update the Software. Except as otherwise expressly provided, Intel grants no express or implied right or license under Intel patents, copyrights, trademarks, or other intellectual property rights.
4. Entire Agreement. This Agreement contains the complete and exclusive statement of the agreement between You and Intel and supersedes all proposals, oral or written, and all other communications relating to the subject matter of this Agreement. Only a written instrument duly executed by authorized representatives of Intel and You may modify this Agreement.
5. LIMITED MEDIA WARRANTY. If the Software has been delivered by Intel on physical media, Intel warrants the media to be free from material physical defects for a period of ninety (90) days after delivery by Intel. If such a defect is found, return the media to Intel for replacement or alternate delivery of the Software as Intel may select.
6. EXCLUSION OF OTHER WARRANTIES. EXCEPT AS PROVIDED ABOVE, THE SOFTWARE IS PROVIDED "AS IS" WITHOUT ANY EXPRESS OR IMPLIED WARRANTY OF ANY KIND, INCLUDING WARRANTIES OF MERCHANTABILITY, NONINFRINGEMENT, OR FITNESS FOR A PARTICULAR PURPOSE. Intel does not warrant or assume responsibility for any errors, the accuracy or completeness of any information, text, graphics, links or other materials contained within the Software.
7. LIMITATION OF LIABILITY. IN NO EVENT WILL INTEL OR ITS SUPPLIERS BE LIABLE FOR ANY DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, LOST PROFITS, BUSINESS INTERRUPTION OR LOST INFORMATION) ARISING OUT OF THE USE OF OR INABILITY TO USE THE SOFTWARE, EVEN IF INTEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOME JURISDICTIONS PROHIBIT EXCLUSION OR LIMITATION OF LIABILITY FOR IMPLIED WARRANTIES OR CONSEQUENTIAL OR INCIDENTAL DAMAGES, SO THE ABOVE LIMITATION MAY NOT APPLY TO YOU. YOU MAY ALSO HAVE OTHER LEGAL RIGHTS THAT VARY FROM JURISDICTION TO JURISDICTION.
8. TERMINATION OF THIS AGREEMENT. Intel may terminate this Agreement at any time if You violate its terms. Upon termination, You will immediately destroy the Software or return all copies of the Software to Intel.
9. APPLICABLE LAWS. Claims arising under this Agreement will be governed by the laws of Delaware, excluding its principles of conflict of laws and the United Nations Convention on Contracts for the Sale of Goods. You may not export the Software in violation of applicable export laws and regulations. Intel is not obligated under any other agreements unless they are in writing and signed by an authorized representative of Intel.
10. GOVERNMENT RESTRICTED RIGHTS. The Software is provided with "RESTRICTED RIGHTS." Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or their successors. Use of the Software by the Government constitutes acknowledgment of Intel's proprietary rights therein. Contractor or Manufacturer is Intel Corporation, 2200 Mission College Blvd., Santa Clara, CA 95054.

Chapter 1

Overview of TMOS Routing

Overview of routing administration in TMOS

As a BIG-IP ®system administrator, you typically manage routing on the system by configuring these BIG-IP system features.

Table 1: BIG-IP system features for route configuration

BIG-IP system feature	Benefit
Interfaces	For the physical interfaces on the BIG-IP system, you can configure properties such as flow control and sFlow polling intervals. You can also configure the Link Layer Discovery Protocol (LLDP), globally for all interfaces and on a per-interface basis.
Trunks	A *trunk* is a logical grouping of interfaces on the BIG-IP system. When you create a trunk, this logical group of interfaces functions as a single interface. The BIG-IP system uses a trunk to distribute traffic across multiple links, in a process known as link aggregation.
VLANs	You create VLANs for the external and internal BIG-IP networks, as well as for high-availability communications in a BIG-IP device clustering configuration. The BIG-IP system supports VLANs associated with both tagged and untagged interfaces.
Virtual and self IP addresses	You can create two kinds of IP addresses locally on the BIG-IP system. A *virtual IP address* is the address associated with a virtual server. A *self IP address* is an IP address on the BIG-IP system that you associate with a VLAN or VLAN group, to access hosts in that VLAN or VLAN group. Whenever you create virtual IP addresses and self IP addresses on the BIG-IP system, the system automatically adds routes to the system that pertain to those addresses, as directly-connected routes.
DHCP support	You can configure the BIG-IP system to function as a DHCP relay or renewal agent. You can also force the renewal of the DHCP lease for the BIG-IP system management port.

BIG-IP system feature	Benefit
Packet filtering	Using packet filters, you can specify whether a BIG-IP system interface should accept or reject certain packets based on criteria such as source or destination IP address. Packet filters enforce an access policy on incoming traffic.
IP address translation	You can configure network address translation (NATs) and source network address translation (SNATs) on the BIG-IP system. Creating a SNAT for a virtual server is a common way to ensure that pool members return responses to the client through the BIG-IP system.
Route domains	You create route domains to segment traffic associated with different applications and to allow devices to have duplicate IP addresses within the same network.
Static routes	For destination IP addresses that are not on the directly-connected network, you can explicitly add static routes. You can add both management (administrative) and TMM static routes to the BIG-IP system.
Dynamic routing	You can configure the advanced routing modules (a set of dynamic routing protocols and core daemons) to ensure that the BIG-IP system can learn about routes from other routers and advertise BIG-IP system routes. These advertised routes can include BIG-IP virtual addresses.
Spanning Tree Protocol (STP)	You can configure any of the Spanning Tree protocols to block redundant paths on a network, thus preventing bridging loops.
The ARP cache	You can manage static and dynamic entries in the ARP cache to resolve IP addresses into MAC addresses.
WCCPv2 support	*WCCPv2* is a content-routing protocol developed by Cisco® Systems. It provides a mechanism to redirect traffic flows in real time. The primary purpose of the interaction between WCCPv2-enabled routers and a BIG-IP® system is to establish and maintain the transparent redirection of selected types of traffic flowing through those routers.

About BIG-IP system routing tables

The BIG-IP system contains two sets of routing tables:

- The Linux routing tables, for routing administrative traffic through the management interface
- A special TMM routing table, for routing application and administrative traffic through the TMM interfaces

As a BIG-IP administrator, you configure the system so that the BIG-IP system can use these routing tables to route both management and application traffic successfully.

About BIG-IP management routes and TMM routes

The BIG-IP system maintains two kinds of routes:

Management routes

Management routes are routes that the BIG-IP system uses to forward traffic through the special management interface. The BIG-IP system stores management routes in the Linux (that is, kernel) routing table.

TMM routes

TMM routes are routes that the BIG-IP system uses to forward traffic through the Traffic Management Microkernel (TMM) interfaces instead of through the management interface. The BIG-IP system stores TMM routes in both the TMM and kernel routing tables.

Viewing routes on the BIG-IP system

You can use the `tmsh` utility to view different kinds of routes on the BIG-IP system.

1. Open a console window, or an SSH session using the management port, on the BIG-IP system.
2. Use your user credentials to log in to the system.
3. Perform one of these actions at the command prompt:
   - To view all routes on the system, type: `tmsh show /net route`
   - To view all configured static routes on the system, type: `tmsh list /net route`

You are now able to view BIG-IP system routes.

Chapter 2

Interfaces

Introduction to BIG-IP system interfaces

A key task of the BIG-IP® system configuration is the configuration of BIG-IP system interfaces. The interfaces on a BIG-IP system are the physical ports that you use to connect the BIG-IP system to other devices on the network. These other devices can be next-hop routers, Layer 2 devices, destination servers, and so on. Through its interfaces, the BIG-IP system can forward traffic to or from other network devices.

Note: *The term interface refers to the physical ports on the BIG-IP system.*

Every BIG-IP system includes multiple interfaces. The exact number of interfaces that you have on the BIG-IP system depends on the platform type.

A BIG-IP system has two types of interfaces:

A management interface
The *management interface* is a special interface dedicated to performing a specific set of system management functions.

TMM switch interfaces
TMM switch interfaces are those interfaces that the BIG-IP system uses to send or receive application traffic, that is, traffic slated for application delivery.

Each of the interfaces on the BIG-IP system has unique properties, such as the MAC address, media speed, duplex mode, and support for Link Layer Discovery Protocol (LLDP).

In addition to configuring interface properties, you can implement a feature known as *interface mirroring*, which you can use to duplicate traffic from one or more interfaces to another. You can also view statistics about the traffic on each interface.

Once you have configured the properties of each interface, you can configure several other features of the BIG-IP system that control the way that interfaces operate. For example, by creating a virtual local area network (VLAN) and assigning interfaces to it, the BIG-IP system can insert a VLAN ID, or tag, into frames passing through those interfaces. In this way, a single interface can forward traffic for multiple VLANs.

About link layer discovery protocol

The BIG-IP® system supports Link Layer Discovery Protocol (LLDP). LLDP is a Layer 2 industry-standard protocol (IEEE 802.1AB) that enables a network device such as the BIG-IP system to advertise its identity and capabilities to multi-vendor neighbor devices on a network. The protocol also enables a network device to receive information from neighbor devices.

LLDP transmits device information in the form of LLDP messages known as LLDP Data Units (LLDPDUs). In general, this protocol:

- Advertises connectivity and management information about the local BIG-IP device to neighbor devices on the same IEEE 802 LAN.
- Receives network management information from neighbor devices on the same IEEE 802 LAN.
- Operates with all IEEE 802 access protocols and network media.

Using the BIG-IP Configuration utility or `tmsh`, you can configure the BIG-IP system interfaces to transmit or receive LLDPDUs. More specifically, you can:

- Specify the exact content of LLDPDUs that a BIG-IP system interface transmits to a neighbor device. You specify this content by configuring the LLDP Attributes setting on each individual interface.
- Globally specify the frequencies of various message transmittal properties, and specify the number of neighbors from which each interface can receive messages. These properties apply to all interfaces on the BIG-IP system.

This figure shows a local LLDP-enabled BIG-IP system, configured to both transmit and receive LLDP messages from neighbor devices on a LAN.

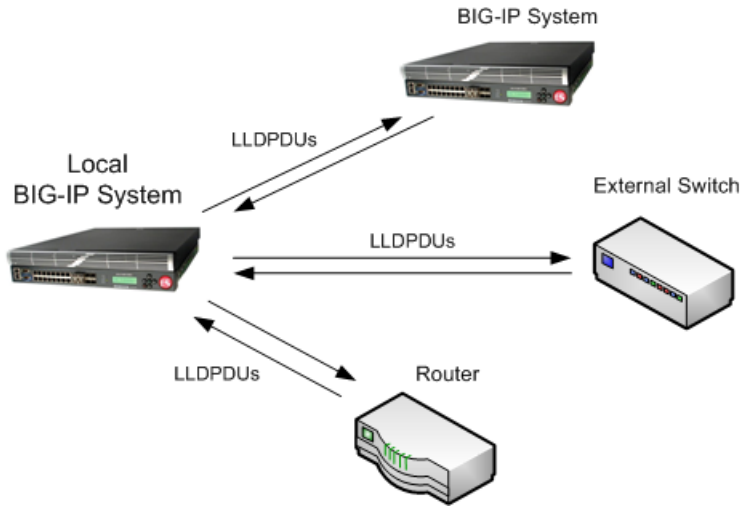

Figure 1: A local BIG-IP system that transmits and receives LLDPDUs

Interface properties

Each interface on the BIG-IP® system has a set of properties that you can configure, such as enabling or disabling the interface, setting the requested media type and duplex mode, and configuring flow control. Configuring the properties of each interface is one of the first tasks you do after running the Setup utility on the BIG-IP system. While you can change some of these properties, such as media speed and duplex mode, you cannot change other properties, such as the media access control (MAC) address.

Note: *You can configure STP-related properties on an interface by configuring one of the Spanning Tree protocols.*

Before configuring interface properties, it is helpful to understand interface naming conventions. Only users with either the Administrator or Resource Administrator user role can create and manage interfaces.

Interface naming conventions

By convention, the names of the interfaces on the BIG-IP® system use the format <s>.<p> where s is the slot number of the network interface card (NIC), and p is the port number on the NIC. Examples of interface names are `1.1`, `1.2`, and `2.1`. BIG-IP system interfaces already have names assigned to them; you do not explicitly assign them.

An exception to the interface naming convention is the management interface, which has the special name, MGMT.

About interface information and media properties

Using the BIG-IP Configuration utility, you can display a screen that lists all of the BIG-IP® system interfaces, as well as their current status (`UP` or `DOWN`). You can also view other information about each interface:

- MAC address of the interface
- Interface availability
- Media type
- Media speed
- Active mode (such as full)

This information is useful when you want to assess the way that a particular interface is forwarding traffic. For example, you can use this information to determine the specific VLANs for which an interface is currently forwarding traffic. You can also use this information to determine the speed at which an interface is currently operating.

Interface state

You can either enable or disable an interface on the BIG-IP® system. By default, each interface is set to Enabled, where it can accept ingress or egress traffic. When you set the interface to Disabled, the interface cannot accept ingress or egress traffic.

Fixed Requested Media

The Fixed Requested Media property shows that the interface auto-detects the duplex mode of the interface.

About flow control

You can configure the way that an interface handles pause frames for flow control. *Pause frames* are frames that an interface sends to a peer interface as a way to control frame transmission from that peer interface. Pausing a peer's frame transmissions prevents an interface's First-in, First-out (FIFO) queue from filling up and resulting in a loss of data. Possible values for this property are:

Pause None
Disables flow control.

Pause TX/RX
Specifies that the interface honors pause frames from its peer, and also generates pause frames when necessary. This is the default value.

Pause TX
Specifies that the interface ignores pause frames from its peer, and generates pause frames when necessary.

Pause RX
Specifies that the interface honors pause frames from its peer, but does not generate pause frames.

About the Ether Type property

The Ether Type property appears in the BIG-IP® Configuration utility only when the system includes ePVA hardware support. An *ether type* is a two-octet field in an Ethernet frame, used to indicate the protocol encapsulated in the payload. The BIG-IP system uses the value of this property when an interface or trunk is associated with a IEEE 802.1QinQ (double tagged) VLAN. By default, the system sets this value to **0x8100**.

About the LLDP property

The LLDP property is one of two properties related to LLDP that you can configure for a specific interface. The possible values for this setting are:

Disabled
When set to this value, the interface neither transmits (sends) LLDP messages to, nor receives LLDP messages from, neighboring devices.

Transmit Only
When set to this value, the interface transmits LLDP messages to neighbor devices but does not receive LLDP messages from neighbor devices.

Receive Only
When set to this value, the interface receives LLDP messages from neighbor devices but does not transmit LLDP messages to neighbor devices.

Transmit and Receive
When set to this value, the interface transmits LLDP messages to and receives LLDP messages from neighboring devices.

In addition to the LLDP-related settings that you can configure per interface, you can configure some global LLDP settings that apply to all interfaces on the system.

Moreover, you can view statistics pertaining to any neighbor devices that have transmitted LLDP messages to the local BIG-IP® system.

LLDP Attributes

The LLDP Attributes setting is one of two settings related to LLDP that you can configure for a specific interface. You use this interface setting to specify the content of an LLDP message being sent or received.

Each LLDP attribute that you specify with this setting is optional and is in the form of Type, Length, Value (TLV).

About interface mirroring

For reliability reasons, you can configure a feature known as interface mirroring. When you configure *interface mirroring*, you cause the BIG-IP® system to copy the traffic on one or more interfaces to another interface that you specify. By default, the interface mirroring feature is disabled.

Neighbor settings

When a BIG-IP® system interface receives LLDP messages from neighbor devices, the BIG-IP system displays chassis, port, and system information about the content of those messages. Specifically, the system displays values for the standard TLVs for each neighbor. These TLVs are:

Chassis ID
Identifies the chassis containing the IEEE 802 LAN station associated with the transmitting LLDP agent.

Port ID
Identifies the port component of the media service access point (MSAP) identifier associated with the transmitting LLDP agent.

Port description
An alpha-numeric string that describes the interface.

System name
An alpha-numeric string that indicates the administratively-assigned name of the neighbor device.

System description
An alpha-numeric string that is the textual description of the network entity. The system description should include the full name and version identification of the hardware type, software operating system, and networking software of the neighbor device.

System capabilities
The primary functions of the system and whether these primary functions are enabled.

Management address
An address associated with the local LLDP agent used to reach higher layer entities. This TLV might also include the system interface number that is associated with the management address, if known.

Configuring settings for an interface

You can use this procedure to configure the settings for an individual interface on the BIG-IP ®system.

1. On the Main tab, click **Network** > **Interfaces** > **Interface List**.
   The Interface List screen displays the list of interfaces on the system.
2. In the Name column, click an interface number.

This displays the properties of the interface.

3. For the **State** setting, verify that the interface is set to **Enabled**.
4. From the **LLDP** list, select a value.
5. For the **LLDP Attributes** setting, verify that the list of attributes in the **Send** field includes all Time Length Values (TLVs) that you want the BIG-IP system interface to send to neighbor devices.
6. Click the **Update** button.

After you perform this task, the interface is configured to send the specified LLDP information to neighbor devices.

Related configuration tasks

After you have configured the interfaces on the BIG-IP® system, one of the primary tasks you perform is to assign those interfaces to the virtual LANs (VLANs) that you create. A *VLAN* is a logical subset of hosts on a local area network (LAN) that reside in the same IP address space. When you assign multiple interfaces to a single VLAN, traffic destined for a host in that VLAN can travel through any one of these interfaces to reach its destination. Conversely, when you assign a single interface to multiple VLANs, the BIG-IP system can use that single interface for any traffic that is intended for hosts in those VLANs.

Another powerful feature that you can use for BIG-IP system interfaces is trunking, with link aggregation. A *trunk* is an object that logically groups physical interfaces together to increase bandwidth. Link aggregation, through the use of the industry-standard Link Aggregation Control Protocol (LACP), provides regular monitoring of link status, as well as failover if an interface becomes unavailable.

Finally, you can configure the BIG-IP system interfaces to work with one of the spanning tree protocols (STP, RSTP, and MSTP). *Spanning tree protocols* reduce traffic on your internal network by blocking duplicate routes to prevent bridging loops.

Chapter 3

Trunks

Introduction to trunks

A *trunk* is a logical grouping of interfaces on the BIG-IP® system. When you create a trunk, this logical group of interfaces functions as a single interface. The BIG-IP system uses a trunk to distribute traffic across multiple links, in a process known as *link aggregation*. With link aggregation, a trunk increases the bandwidth of a link by adding the bandwidth of multiple links together. For example, four fast Ethernet (100 Mbps) links, if aggregated, create a single 400 Mbps link.

With one trunk, you can aggregate a maximum of eight links. For optimal performance, you should aggregate links in powers of two. Thus, you ideally aggregate two, four, or eight links.

The purpose of a trunk is two-fold:

- To increase bandwidth without upgrading hardware
- To provide link failover if a member link becomes unavailable

You can use trunks to transmit traffic from a BIG-IP system to another vendor switch. Two systems that use trunks to exchange frames are known as *peer systems*.

Creating a trunk

You create a trunk on the BIG-IP® system so that the system can then aggregate the links to enhance bandwidth and ensure link availability.

1. On the Main tab, click **Network** > **Trunks**.
   The Trunk List screen opens.
2. Click **Create**.
3. Name the trunk.
4. For the **Interfaces** setting, in the **Available** field, select an interface, and using the Move button, move the interface to the **Members** field. Repeat this action for each interface that you want to include in the trunk.

   Trunk members must be untagged interfaces and cannot belong to another trunk. Therefore, only untagged interfaces that do not belong to another trunk appear in the **Available** list.
5. Select the **LACP** check box.
6. Click **Finished**.

After you create a trunk, the BIG-IP system aggregates the links to enhance bandwidth and prevent interruption in service.

How do trunks operate?

In a typical configuration where trunks are configured, the member links of the trunk are connected through Ethernet cables to corresponding links on a peer system.

This figure shows an example of a typical trunk configuration with two peers and three member links on each peer:

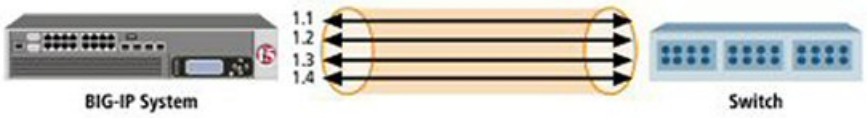

Figure 2: Example of a trunk configured for two switches

A primary goal of the trunks feature is to ensure that frames exchanged between peer systems are never sent out of order or duplicated on the receiving end. The BIG-IP® system is able to maintain frame order by using the source and destination addresses in each frame to calculate a hash value, and then transmitting all frames with that hash value on the same member link.

The BIG-IP system automatically assigns a unique MAC address to a trunk. However, by default, the MAC address that the system uses as the source and destination address for frames that the system transmits and receives (respectively), is the MAC address of the lowest-numbered interface of the trunk.

The BIG-IP system also uses the lowest-numbered interface of a trunk as a reference link. The BIG-IP system uses the reference link to take certain aggregation actions, such as implementing the automatic link selection policy. For frames coming into the reference link, the BIG-IP system load balances the frames across all member links that the BIG-IP system knows to be available. For frames going from any link in the trunk to a destination host, the BIG-IP system treats those frames as if they came from the reference link.

Finally, the BIG-IP system uses the MAC address of an individual member link as the source address for any LACP control frames.

Overview of LACP

A key aspect of trunks is Link Aggregation Control Protocol, or LACP. Defined by IEEE standard 802.3ad, *LACP* is a protocol that detects error conditions on member links and redistributes traffic to other member links, thus preventing any loss of traffic on the failed link. On a BIG-IP® system, LACP is an optional feature that you can configure.

You can also customize LACP behavior. For example, you can specify the way that LACP communicates its control messages from the BIG-IP system to a peer system. You can also specify the rate at which the peer system sends LACP packets to the BIG-IP system. If you want to affect the way that the BIG-IP system chooses links for link aggregation, you can specify a link control policy.

Interfaces for a trunk

Using the **Interfaces** setting, you specify the interfaces that you want the BIG-IP® system to use as member links for the trunk. Once you have created the trunk, the BIG-IP system uses these interfaces to perform link aggregation.

Tip: *To optimize bandwidth utilization, F5 Networks® recommends that, if possible, the number of links in the trunk be a power of 2 (for example, 2, 4, or 8). This is due to the frame balancing algorithms that the system uses to map data streams to links. Regardless of the hashing algorithm, a trunk that has 2, 4, or 8 links prevents the possibility of skewing, which can adversely affect data throughput.*

The BIG-IP system uses the lowest-numbered interface as the reference link. The system uses the reference link to negotiate links for aggregation.

The interfaces that you specify for the trunk must operate at the same media speed, and must be set at full-duplex mode. Otherwise, the BIG-IP system cannot aggregate the links. Because these media properties are dynamic rather than static (due to auto-negotiation), the `lacpd` service routinely monitors the current status of these properties and negotiates the links for aggregation accordingly. Thus, when the status of these properties qualifies a link to become a working member link, the system adds the link to the aggregation, and the link can begin accepting traffic.

Any interface that you assign to a trunk must be an untagged interface. Furthermore, you can assign an interface to one trunk only; that is, you cannot assign the same interface to multiple trunks. Because of these restrictions, the only interfaces that appear in the Interfaces list in the BIG-IP Configuration utility are untagged interfaces that are not assigned to another trunk. Therefore, before creating a trunk and assigning any interfaces to it, you should verify that each interface for the trunk is an untagged interface.

After creating the trunk, you assign the trunk to one or more VLANs, using the same VLAN screen that you normally use to assign an individual interface to a VLAN.

If you are using one of the spanning tree protocols (STP, RSTP, or MSTP), the BIG-IP system sends and receives spanning tree protocol packets on a trunk, rather than on individual member links. Likewise, use of a spanning tree protocol to enable or disable learning or forwarding on a trunk operates on all member links together, as a single unit.

About trunk configuration

For VIPRION® platforms, F5 Networks® strongly recommends that you create a trunk for each of the BIG-IP® system internal and external networks, and that each trunk contains interfaces from all slots in the cluster.

For example, a trunk for the external network should contain the external interfaces of all blades in the cluster. Configuring a trunk in this way prevents interruption in service if a blade in the cluster becomes unavailable and minimizes use of the high-speed backplane when processing traffic.

Also, you should connect the links in a trunk to a vendor switch on the relevant network.

Important: *When processing egress packets, including those of vCMP® guests, the BIG-IP system uses trunk member interfaces on local blades whenever possible. This behavior ensures efficient use of the backplane, thereby conserving backplane bandwidth for processing ingress packets.*

About the Ether Type property

The Ether Type property appears in the BIG-IP® Configuration utility only when the system includes ePVA hardware support. An *ether type* is a two-octet field in an Ethernet frame, used to indicate the protocol encapsulated in the payload. The BIG-IP system uses the value of this property when an interface or trunk is associated with a IEEE 802.1QinQ (double tagged) VLAN. By default, the system sets this value to **0x8100**.

About enabling LACP

As an option, you can enable LACP on a trunk. Containing a service called `lacpd`, LACP is an IEEE-defined protocol that exchanges control packets over member links. The purpose of LACP is to detect link error conditions such as faulty MAC devices and link loopbacks. If LCAP detects an error on a member link, the BIG-IP® system removes the member link from the link aggregation and redistributes the traffic for that link to the remaining links of the trunk. In this way, no traffic destined for the removed link is lost. LACP then continues to monitor the member links to ensure that aggregation of those links remains valid.

By default, the LACP feature is disabled, to ensure backward compatibility with previous versions of the BIG-IP system. If you create a trunk and do not enable the LACP feature, the BIG-IP system does not detect link error conditions, and therefore cannot remove the member link from link aggregation. The result is that the system cannot redistribute the traffic destined for that link to the remaining links in the trunk, thereby causing traffic on the failed member link to be lost.

Important: *To use LACP successfully, you must enable LACP on both peer systems.*

LACP mode

The **LACP Mode** setting appears on the Trunks screen only when you select the LACP setting. You use the **LACP Mode** setting to specify the method that LACP uses to send control packets to the peer system. The two possible modes are:

Active mode

You specify **Active** mode if you want the system to periodically send control packets, regardless of whether the peer system has issued a request. This is the default setting.

Passive mode

You specify **Passive** mode if you want the system to send control packets only when the peer system issues a request, that is, when the LACP mode of the peer system is set to Active.

If you set only one of the peer systems to Active mode, the BIG-IP® system uses Active mode for both systems. Also, whenever you change the LACP mode on a trunk, LACP renegotiates the links that it uses for aggregation on that trunk.

Tip: *We recommend that you set the LACP mode to Passive on one peer system only. If you set both systems to Passive mode, LACP does not send control packets.*

LACP timeout

The **LACP Timeout** setting appears on the Trunks screen only when you select the LACP setting. You use the **LACP Timeout** setting to indicate to the BIG-IP® system the interval in seconds at which the peer system should send control packets. The timeout value applies only when the LACP mode is set to Active on at least one of the switch systems. If both systems are set to Passive mode, LACP does not send control packets.

If LACP sends three consecutive control packets without receiving a response from the peer system, LACP removes that member link from link aggregation. The two possible timeout values are:

Short

When you set the timeout value to Short, the peer system sends LACP control packets once every second. If this value is set to Short and LACP receives no peer response after sending three consecutive packets, LACP removes the link from aggregation in three seconds.

Long

When you set the timeout value to Long, the peer system sends LACP control packets once every 30 seconds. A timeout value of Long is the default setting. If set to Long and LACP receives no peer response after sending three consecutive packets, LACP removes the link from aggregation in ninety seconds.

Whenever you change the LACP timeout value on a trunk, LACP renegotiates the links that it uses for aggregation on that trunk.

Link selection policy

In order for the BIG-IP® system to aggregate links, the media speed and duplex mode of each link must be the same on both peer systems. Because media properties can change dynamically, the BIG-IP system monitors these properties regularly, and if it finds that the media properties of a link are mismatched on the peer systems, the BIG-IP system must determine which links are eligible for aggregation.

The way the system determines eligible links depends on a link selection policy that you choose for the trunk. When you create a trunk, you can choose one of two possible policy settings: **Auto** and **Maximum Bandwidth**.

Note: *The link selection policy feature represents an F5 Networks® enhancement to the standard IEEE 802.3ad specification for LACP.*

Automatic link selection

When you set the link selection policy to Auto (the default setting), the BIG-IP® system uses the lowest-numbered interface of the trunk as a reference link. (A *reference link* is a link that the BIG-IP system uses to make a link aggregation decision.) The system then aggregates any links that have the same media properties and are connected to the same peer as the reference link.

For example, suppose that you created a trunk to include interfaces 1.2 and 1.3, each with a media speeds of 100 Mbps, and interface 1.4, with a different media speed of 1 Gbps. If you set the link selection policy to Auto, the BIG-IP system uses the lowest-numbered interface, 1.2, as a reference link. The reference link operates at a media speed of 100 Mbps, which means that the system aggregates all links with that media speed (interfaces 1.2 and 1.3). The media speed of interface 1.4 is different (1 Gbps), and therefore is not considered for link aggregation. Only interfaces 1.2 and 1.3 become working member links and start carrying traffic.

If the media speed of interface 1.4 changes to 100 Mbps, the system adds that interface to the aggregation. Conversely, if the media speed of interface 1.4 remains at 1 Gbps, and the speed of the reference link changes to 1 Gbps, then interfaces 1.2 and 1.4 become working members, and 1.3 is now excluded from the aggregation and no longer carries traffic.

Maximum bandwidth link selection

When you set the link selection policy to **Maximum Bandwidth**, the BIG-IP® system aggregates the subset of member links that provide the maximum amount of bandwidth to the trunk.

If interfaces 1.2 and 1.3 each operate at a media speed of 100 Mbps, and interface 1.4 operates at speed of 1 Gbps, then the system selects only interface 1.4 as a working member link, providing 1 Gbps of bandwidth to the trunk. If the speed of interface 1.4 drops to 10 Mbps, the system then aggregates links 1.2 and 1.3, to provide a total bandwidth to the trunk of 200 Mbps. The peer system detects any non-working member links and configures its aggregation accordingly.

Tip: *To ensure that link aggregation operates properly, make sure that both peer systems agree on the link membership of their trunks.*

Frame distribution hash

When frames are transmitted on a trunk, they are distributed across the working member links. The distribution function ensures that the frames belonging to a particular conversation are neither mis-ordered nor duplicated at the receiving end.

The BIG-IP® system distributes frames by calculating a hash value based on the source and destination addresses (or the destination address only) carried in the frame, and associating the hash value with a link. All frames with a particular hash value are transmitted on the same link, thereby maintaining frame order. Thus, the system uses the resulting hash to determine which interface to use for forwarding traffic.

The **Frame Distribution Hash** setting specifies the basis for the hash that the system uses as the frame distribution algorithm.

The default value is Source/Destination IP address.

Possible values for this setting are:

Source/Destination MAC address
This value specifies that the system bases the hash on the combined MAC addresses of the source and the destination.

Destination MAC address
This value specifies that the system bases the hash on the MAC address of the destination.

Source/Destination IP address
This value specifies that the system bases the hash on the combined IP addresses of the source and the destination.

Chapter 4

VLANs, VLAN Groups, and VXLAN

About VLANs

A *VLAN* is a logical subset of hosts on a local area network (LAN) that operate in the same IP address space. Grouping hosts together in a VLAN has distinct advantages. For example, with VLANs, you can:

- Reduce the size of broadcast domains, thereby enhancing overall network performance.
- Reduce system and network maintenance tasks substantially. Functionally-related hosts no longer need to physically reside together to achieve optimal network performance.
- Enhance security on your network by segmenting hosts that must transmit sensitive data.

You can create a VLAN and associate physical interfaces with that VLAN. In this way, any host that sends traffic to a BIG-IP® system interface is logically a member of the VLAN or VLANs to which that interface belongs.

Default VLAN configuration

By default, the BIG-IP® system includes VLANs named **internal** and **external**. When you initially ran the Setup utility, you assigned the following to each of these VLANs:

- A static and a floating self IP address
- A VLAN tag
- One or more BIG-IP system interfaces

A typical VLAN configuration is one in which the system has the two VLANs **external** and **internal**, and one or more BIG-IP system interfaces assigned to each VLAN. You then create a virtual server, and associate a default load balancing pool with that virtual server. This figure shows a typical configuration using the default VLANs external and internal.

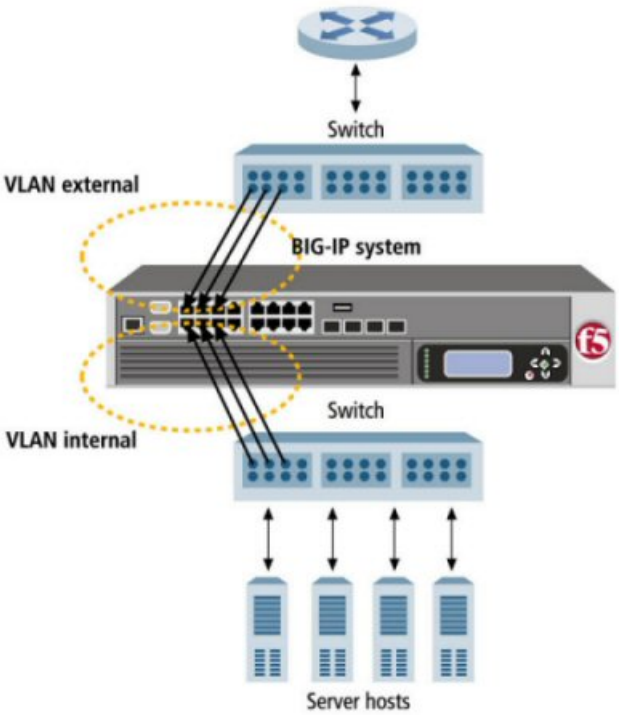

Figure 3: A typical configuration using the default VLANs

Note: *VLANs* `internal` *and* `external` *reside in partition* `Common`*.*

About VLANs and interfaces

VLANs are directly associated with the physical interfaces on the BIG-IP® system.

Interface assignments

For each VLAN that you create, you must assign one or more BIG-IP® system interfaces to that VLAN. When you assign an interface to a VLAN, you indirectly control the hosts from which the BIG-IP system interface sends or receives messages.

Tip: *You can assign not only individual interfaces to the VLAN, but also trunks.*

For example, if you assign interface 1.11 to `VLAN A`, and you then associate `VLAN A` with a virtual server, then the virtual server sends its outgoing traffic through interface 1.11, to a destination host in `VLAN A`. Similarly, when a destination host sends a message to the BIG-IP system, the host's VLAN membership determines the BIG-IP system interface that should receive the incoming traffic.

Each VLAN has a MAC address. The MAC address of a VLAN is the same MAC address of the lowest-numbered interface assigned to that VLAN.

About untagged interfaces

You can create a VLAN and assign interfaces to the VLAN as untagged interfaces. When you assign interfaces as *untagged interfaces*, you cannot associate other VLANs with those interfaces. This limits the

interface to accepting traffic only from that VLAN, instead of from multiple VLANs. If you want to give an interface the ability to accept and receive traffic for multiple VLANs, you add the same interface to each VLAN as a tagged interface.

About tagged interfaces

You can create a VLAN and assign interfaces to the VLAN as single- or double-tagged interfaces. When you assign interfaces as *tagged interfaces*, you can associate multiple VLANs with those interfaces.

A VLAN *tag* is a unique ID number that you assign to a VLAN, to identify the VLAN to which each packet belongs. If you do not explicitly assign a tag to a VLAN, the BIG-IP® system assigns a tag automatically. The value of a VLAN tag can be between 1 and 4094. Once you or the BIG-IP system assigns a tag to a VLAN, any message sent from a host in that VLAN includes this VLAN tag as a header in the message.

Important: *If a device connected to a BIG-IP system interface is another switch, the VLAN tag that you assign to the VLAN on the BIG-IP system interface must match the VLAN tag assigned to the VLAN on the interface of the other switch.*

About single tagging

This figure shows the difference between using three untagged interfaces (where each interface must belong to a separate VLAN) versus one single-tagged interface (which belongs to multiple VLANs).

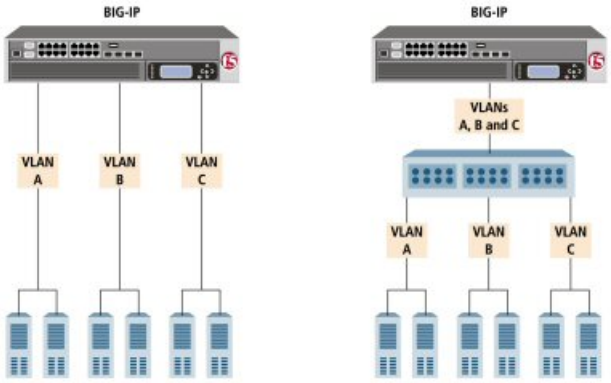

Figure 4: Solutions using untagged (left) and single-tagged interfaces (right)

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

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

Important: *If you are connecting another switch into a BIG-IP system interface, the VLAN tag that you assign to the VLAN on the BIG-IP system must match the VLAN tag on the interface of the other switch.*

About double tagging

For BIG-IP® systems with ePVA hardware support, the system includes support for the IEEE 802.1QinQ standard. Known informally as *Q-in-Q* or *double tagging*, this standard provides a way for you to insert

multiple VLAN tags into a single frame. This allows you to encapsulate single-tagged traffic from disparate customers with only one tag.

Double tagging expands the number of possible VLAN IDs in a network. With double tagging, the theoretical limitation in the number of VLAN IDs expands from 4096 to 4096*4096.

When you implement double tagging, you specify an *inner tag* that encapsulates all of the single-tagged traffic. You then designate all other tags as *outer tags*, or *customer tags* (C-tags), which serve to identify and segregate the traffic from those customers.

A common use case is one in which an internet service provider creates a single VLAN within which multiple customers can retain their own VLANs without regard for overlapping VLAN IDs. Moreover, you can use double-tagged VLANs within route domains or vCMP® guests. In the latter case, vCMP host administrators can create double-tagged VLANs and assign the VLANs to guests, just as they do with single-tagged VLANs. For a vCMP guest running an older version of the BIG-IP software, double-tagged VLANs are not available for assignment to the guest.

Note: *On systems that support double tagging, if you configure a Fast L4 local traffic profile, you cannot enable Packet Velocity Asic (PVA) hardware acceleration.*

VLAN association with a self IP address

Every VLAN must have a static self IP address associated with it. The self IP address of a VLAN represents an address space, that is, the range of IP addresses pertaining to the hosts in that VLAN. When you ran the Setup utility earlier, you assigned one static self IP address to the VLAN external, and one static self IP address to the VLAN internal. When sending a request to a destination server, the BIG-IP system can use these self IP addresses to determine the specific VLAN that contains the destination server.

You associate a self IP address with either a VLAN or VLAN group:

Associating a VLAN with a self IP address

The self IP address with which you associate a VLAN should represent an address space that includes the IP addresses of the hosts that the VLAN contains. For example, if the address of one host is `11.0.0.1` and the address of the other host is `11.0.0.2`, you could associate the VLAN with a self IP address of `11.0.0.100`, with a netmask of `255.255.255.0`.

Associating a VLAN group with a self IP address

The self IP address with which you associate a VLAN group should represent an address space that includes the self IP addresses of the VLANs that you assigned to the group. For example, if the address of one VLAN is `10.0.0.1` and the address of the other VLAN is `10.0.0.2`, you could associate the VLAN group with a self IP address of `10.0.0.100`, with a netmask of `255.255.255.0`.

VLAN assignment to route domains

If you explicitly create route domains, you should consider the following facts:

- You can assign VLANs (and VLAN groups) to route domain objects that you create. Traffic pertaining to that route domain uses those assigned VLANs.
- During BIG-IP® system installation, the system automatically creates a default route domain, with an ID of 0. Route domain 0 has two VLANs assigned to it, VLAN `internal` and VLAN `external`.
- If you create one or more VLANs in an administrative partition other than `Common`, but do not create a route domain in that partition, then the VLANs you create in that partition are automatically assigned to route domain 0.

Maintaining the L2 forwarding table

Layer 2 forwarding is the means by which frames are exchanged directly between hosts, with no IP routing required. This is accomplished using a simple forwarding table for each VLAN. The L2 forwarding table is a list that shows, for each host in the VLAN, the MAC address of the host, along with the interface that the BIG-IP® system needs for sending frames to that host. The intent of the L2 forwarding table is to help the BIG-IP system determine the correct interface for sending frames, when the system determines that no routing is required.

The format of an entry in the L2 forwarding table is:

```
<MAC address> -> <if>
```

For example, an entry for a host in the VLAN might look like this:

```
00:a0:c9:9e:1e:2f -> 2.1
```

The BIG-IP system learns the interfaces that correspond to various MAC entries as frames pass through the system, and automatically adds entries to the table accordingly. These entries are known as *dynamic entries*. You can also add entries to the table manually, and these are known as *static entries*. Entering static entries is useful if you have network devices that do not advertise their MAC addresses. The system does not automatically update static entries.

The BIG-IP system does not always need to use the L2 forwarding table to find an interface for frame transmission. For instance, if a VLAN has only one interface assigned to it, then the BIG-IP system automatically uses that interface.

Occasionally, the L2 forwarding table does not include an entry for the destination MAC address and its corresponding BIG-IP system interface. In this case, the BIG-IP system floods the frame through all interfaces associated with the VLAN, until a reply creates an entry in the L2 forwarding table.

Additional VLAN configuration options

There are a number of settings that you can configure for a VLAN.

Source checking

When you enable source checking, the BIG-IP® system verifies that the return path for an initial packet is through the same VLAN from which the packet originated. Note that the system only enables source checking if the global setting Auto Last Hop is disabled.

Maximum transmission units

The value that you configure for the maximum transmission unit, or MTU, is the largest size that the BIG-IP® system allows for an IP datagram passing through a BIG-IP system interface. By default, the BIG-IP system uses the standard Ethernet frame size of 1518 bytes (1522 bytes if VLAN tagging is used), with a corresponding MTU value of 1500 bytes for a VLAN.

One reason for changing the value of the **MTU** setting is when your BIG-IP platform supports jumbo frames. A *jumbo frame* is an Ethernet frame with more than 1500 bytes, and fewer than 9000 bytes, of payload.

If your BIG-IP platform does not support jumbo frames and a VLAN receives a jumbo frame, the system discards the frame.

VLAN-based fail-safe

VLAN fail-safe is a feature you enable when you want to base redundant-system failover on VLAN-related events. To configure VLAN fail-safe, you specify a timeout value and the action that you want the system to take when the timeout period expires.

Auto last hop

When you create a VLAN, you can designate the VLAN as the last hop for TMM traffic.

CMP hash

The **CMP Hash** setting allows all connections from a client system to use the same set of TMMs. This improves system performance. In configuring the **CMP Hash** value, you can choose the traffic disaggregation criteria for the VLAN, either source IP address, destination IP address, or TCP/UDP source/destination ports. The default value uses TCP/UDP source/destination ports. Note that the **CMP Hash** setting appears only on the properties screen for an existing VLAN.

DAG round robin

You can use the **DAG Round Robin** setting on a VLAN to prevent stateless traffic from overloading a few TMM instances, a condition that can disable an entire BIG-IP system. When enabled, this setting causes the BIG-IP system to load balance the traffic among TMMs evenly, instead of using a static hash. Stateless traffic in this case includes non-IP Layer 2 traffic, ICMP, some UDP protocols, and so on. This setting is disabled by default.

This feature is particularly useful for firewall and Domain Name System (DNS) traffic. For example, this feature prevents certain types of DDoS attacks, such as an ICMP DDoS attack that can overload the system by sending the same packets repeatedly to a specific subset of TMMs.

Note: *The disaggregation of traffic occurs only to TMMs that are local to a given high-speed bridge (HSB).*

Specifying port numbers

Before you perform this task, confirm that you have enabled the **DAG Round Robin** setting on the relevant VLAN.

If you enable this feature on a VLAN, you must also configure a `bigdb` variable that specifies the relevant destination ports.

1. Access the Traffic Management Shell (`tmsh`).
2. At the `tmsh` prompt, type `modify sys db dag.roundrobin.udp.portlist value "`*`port_number`*`:`*`port_number`*`:`*`port_number`*`:`*`port_number`*`"`

   The values that you specify with this `bigdb` variable apply to all VLANs on which the **DAG Round Robin** setting is enabled.

   For example, you can type `modify sys db dag.roundrobin.udp.portlist value "53:26:19:45"`
   The example specifies that the system load balances packets destined for ports 53, 26, 19, and 45.

About sFlow polling intervals and sampling rates

You can change the sFlow settings for a specific VLAN when you want the traffic flowing through the VLAN to be sampled at a different rate than the global sFlow settings on the BIG-IP® system.

Creating a VLAN

You perform this task when you need to create a VLAN. A *VLAN* represents a logical collection of hosts that can share network resources, regardless of their physical location on the network. When you create a VLAN, you can assign the BIG-IP® interfaces as tagged or untagged interfaces. If your hardware platform supports ePVA, you have the additional option of configuring double tagging (also known as Q-in-Q tagging) for this VLAN.

1. On the Main tab, click **Network** > **VLANs**.
   The VLAN List screen opens.
2. Click **Create**.
   The New VLAN screen opens.
3. In the **Name** field, type a unique name for the VLAN.
4. In the **Tag** field, type a numeric tag, from 1-4094, for the VLAN, or leave the field blank if you want the BIG-IP system to automatically assign a VLAN tag.

   The VLAN tag identifies the traffic from hosts in the associated VLAN.
5. If you want to use Q-in-Q (double) tagging, use the **Customer Tag** setting to perform the following two steps. If you do not see the **Customer Tag** setting, your hardware platform does not support Q-in-Q tagging and you can skip this step.
   a) From the **Customer Tag** list, select **Specify**.
   b) Type a numeric tag, from 1-4094, for the VLAN.

   The customer tag specifies the inner tag of any frame passing through the VLAN.
6. For the **Interfaces** setting:
   a) From the **Interface** list, select an interface number.
   b) From the **Tagging** list, select **Tagged** or **Untagged**.

      Select **Tagged** when you want traffic for that interface to be tagged with a VLAN ID.
   c) If you specified a numeric value for the **Customer Tag** setting and from the **Tagging** list you selected **Tagged**, then from the **Tag Mode** list, select a value.
   d) Click **Add**.
   e) Repeat these steps for each interface that you want to assign to the VLAN.
7. If you want the system to verify that the return route to an initial packet is the same VLAN from which the packet originated, select the **Source Check** check box.
8. In the **MTU** field, retain the default number of bytes (**1500**).
9. From the **Configuration** list, select **Advanced**.
10. If you want to base redundant-system failover on VLAN-related events, select the **Fail-safe** check box.
11. From the **Auto Last Hop** list, select a value.
12. From the **CMP Hash** list, select a value.
13. To enable the **DAG Round Robin** setting, select the check box.
14. Configure the sFlow settings or retain the default values.
15. Click **Finished**.

The screen refreshes, and displays the new VLAN in the list.

About VLAN groups

A *VLAN group* is a logical container that includes two or more distinct VLANs. VLAN groups are intended for load balancing traffic in a Layer 2 network, when you want to minimize the reconfiguration of hosts on that network. This figure shows an example of a VLAN group.

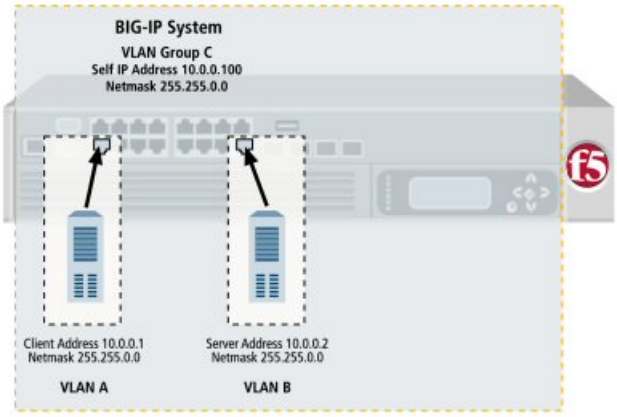

Figure 5: Example of a VLAN group

A VLAN group also ensures that the BIG-IP® system can process traffic successfully between a client and server when the two hosts reside in the same address space. Without a VLAN group, when the client and server both reside in the same address space, the client request goes through the virtual server, but instead of sending its response back through the virtual server, the server attempts to send its response directly to the client, bypassing the virtual server altogether. As a result, the client cannot receive the response, because the client expects the address of the response to be the virtual server IP address, not the server IP address.

Tip: *You can configure the behavior of the BIG-IP system so that it always creates a proxy for any ARP requests between VLANs.*

When you create a VLAN group, the two existing VLANs become child VLANs of the VLAN group.

VLAN groups reside in administrative partitions. To create a VLAN group, you must first set the current partition to the partition in which you want the VLAN group to reside.

Note: *Only users with the Administrator user role can create and manage VLAN groups.*

About VLAN group names

When creating a VLAN group, you must assign it a unique name. Once you have finished creating the VLAN group, the VLAN group name appears in the list of existing VLANs groups.

VLAN group ID

A *VLAN group ID* is a tag for the VLAN group. Every VLAN group needs a unique ID number. If you do not specify an ID for the VLAN group, the BIG-IP® system automatically assigns one. The value of a VLAN group ID can be between 1 and 4094.

About transparency mode

The BIG-IP® system is capable of processing traffic using a combination of Layer 2 and Layer 3 forwarding, that is, switching and IP routing. When you set the transparency mode, you specify the type of forwarding that the BIG-IP system performs when forwarding a message to a host in a VLAN. The default setting is translucent, which means that the BIG-IP system uses a mix of Layer 2 and Layer 3 processing. The allowed values are:

opaque
A proxy ARP with Layer 3 forwarding

translucent
Layer 2 forwarding with a locally-unique bit, toggled in ARP response across VLANs. This is the default setting. When you choose this value and you have a virtual server that references a Fast L4 profile, the BIG-IP system automatically changes the **PVA Acceleration** setting to **None**

transparent
Layer 2 forwarding with the original MAC address of the remote system preserved across VLANs. When you choose this value and you have a virtual server that references a Fast L4 profile, the BIG-IP system automatically changes the **PVA Acceleration** setting to **None**.

About traffic bridging

When you enable the traffic bridging option, you are instructing the VLAN group to forward all non-IP traffic. Note that IP traffic is bridged by default. The default value for this setting is disabled (unchecked).

About traffic bridging with standby units

When enabled, the **Bridge in Standby** setting ensures that the VLAN group can forward packets when the system is the standby device of a redundant system configuration. Note that this setting applies to non-IP and non-ARP frames only, such as Bridge Protocol Data Units (BPDUs).

This setting is designed for deployments in which the VLAN group is defined on a redundant system. You can use the **Bridge in Standby** setting in transparent or translucent modes, or in opaque mode when the global variable `Failover.Standby.LinkDownTime` is set to 0.

Warning: *This setting can cause adverse effects if the VLAN group exists on more than one device in a device group. The setting is intended for configurations where the VLAN group exists on one device only. The default setting is enabled (checked).*

About host exclusion from proxy ARP forwarding

A host in a VLAN cannot normally communicate to a host in another VLAN. This rule applies to ARP requests as well. However, if you put the VLANs into a single VLAN group, the BIG-IP® system can perform a proxied ARP request.

A *proxied ARP request* is an ARP request that the BIG-IP system can send, on behalf of a host in a VLAN, to hosts in another VLAN. A proxied ARP request requires that both VLANs belong to the same VLAN group.

In some cases, you might not want a host to forward proxied ARP requests to a specific host, or to other hosts in the configuration. To exclude specific hosts from receiving forwarded proxied ARP requests, you use the BIG-IP Configuration utility and specify the IP addresses that you want to exclude.

Warning: *Although hosts on an ARP exclusion list are specified using their IP addresses, this does not prevent the BIG-IP system from routing traffic to those hosts. A more secure way to prevent traffic from passing between hosts in separate VLANs is to create a packet filter for each VLAN.*

About migration keepalive frames

The Migration Keepalive setting for a VLAN group, when enabled, instructs the BIG-IP® system to send keepalive frames (that is, TCP keepalives and empty UDP packets, depending on the connection type) when a node is moved from one VLAN group member to another VLAN group member for all existing BIG-IP connections to that node.

Creating a VLAN group

VLAN groups consolidate Layer 2 traffic from two or more separate VLANs.

1. On the Main tab, click **Network** > **VLANs** > **VLAN Groups**.
   The VLAN Groups list screen opens.
2. Click **Create**.
   The New VLAN Group screen opens.
3. In the General Properties area, in the **VLAN Group** field, type a unique name for the VLAN group.
4. For the **VLANs** setting, use the Move button (**<<**) to move the VLANs that you want to include in the group from the **Available** list to the **Members** list.
5. From the **Transparency Mode** list, select a transparency mode, or retain the default setting, `Transparent`.

   The transparency mode determines the level of exposure of remote MAC addresses within the VLAN group traffic.

Mode	Purpose
Transparent	The MAC addresses of remote systems are exposed in Layer 2 traffic forwarding.
Translucent	Similar to `Transparent` mode, except the locally-unique bit is set in the MAC addresses of remote systems.
Opaque	The system uses proxy ARP with Layer 3 forwarding, so the MAC addresses of remote systems are not exposed.

6. Select the **Bridge All Traffic** check box if you want the VLAN group to forward all frames, including non-IP traffic.

   The default setting is disabled (not selected).

7. Retain the **Bridge in Standby** check box selection if you want the VLAN group to forward frames, even when the system is the standby unit of a redundant system.
8. For the remaining settings, retain the default values.
9. Click **Finished**.

About bridging VLAN and VXLAN networks

You can configure Virtual eXtended LAN (VXLAN) on a BIG-IP® system to enable a physical VLAN to communicate with virtual machines (VMs) in a virtual network.

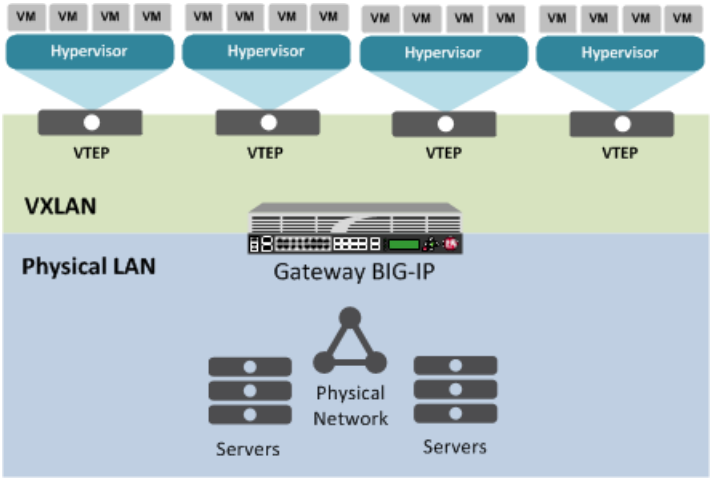

Figure 6: The VXLAN gateway

When you configure a BIG-IP system as an L2 VXLAN gateway, the BIG-IP system joins the configured multicast group, and can forward both unicast and multicast or broadcast frames on the virtual network. The BIG-IP system learns about MAC address and VTEP associations dynamically, thus avoiding unnecessary transmission of multicast traffic.

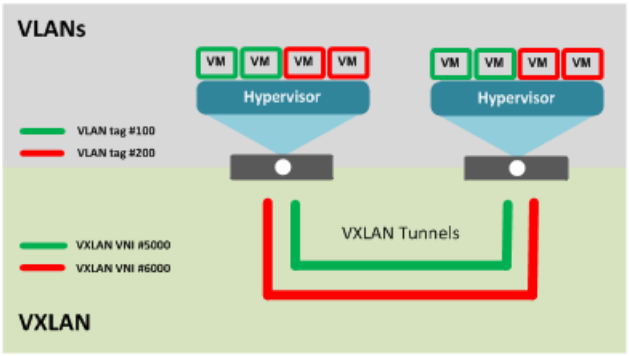

Figure 7: Multiple VXLAN tunnels

About VXLAN multicast configuration

In a VMware vSphere 5.1 environment, you can configure VXLAN without knowing all the remote tunnel endpoints. The BIG-IP® system uses multicast flooding to learn unknown and broadcast frames. VXLAN can extend the virtual network across a set of hypervisors, providing L2 connectivity among the hosted virtual machines (VMs). Each hypervisor represents a VXLAN tunnel endpoint (VTEP). In this environment, you can configure a BIG-IP system as an L2 VXLAN gateway device to terminate the VXLAN tunnel and forward traffic to and from a physical network.

Chapter 5

Self IP Addresses

Introduction to self IP addresses

A *self IP address* is an IP address on the BIG-IP® system that you associate with a VLAN, to access hosts in that VLAN. By virtue of its netmask, a self IP address represents an *address space*, that is, a range of IP addresses spanning the hosts in the VLAN, rather than a single host address. You can associate self IP addresses not only with VLANs, but also with VLAN groups.

Self IP addresses serve two purposes:

- First, when sending a message to a destination server, the BIG-IP system uses the self IP addresses of its VLANs to determine the specific VLAN in which a destination server resides. For example, if VLAN internal has a self IP address of `10.10.10.100`, with a netmask of `255.255.255.0`, and the destination server's IP address is `10.10.10.20` (with a netmask of `255.255.255.255`), the BIG-IP system recognizes that the server's IP address falls within the range of VLAN internal's self IP address, and therefore sends the message to that VLAN. More specifically, the BIG-IP system sends the message to the interface that you assigned to that VLAN. If more than one interface is assigned to the VLAN, the BIG-IP system takes additional steps to determine the correct interface, such as checking the Layer2 forwarding table.
- Second, a self IP address can serve as the default route for each destination server in the corresponding VLAN. In this case, the self IP address of a VLAN appears as the destination IP address in the packet header when the server sends a response to the BIG-IP system.

You normally assign self IP addresses to a VLAN when you initially run the Setup utility on a BIG-IP system. More specifically, you assign one static self IP address and one floating self IP address to each of the default VLANs (internal and external). Later, using the BIG-IP Configuration utility, you can create self IP addresses for other VLANs that you create.

Self IP addresses reside in administrative partitions/folders and are associated with traffic groups. The self IP addresses that you create when you run the Setup utility reside in partition Common (that is folder `/Common`).

Types of self IP addresses

There are two types of self IP addresses that you can create:

- A *static self IP address* is an IP address that the BIG-IP® system does not share with another BIG-IP system. Any self IP address that you assign to the default traffic group `traffic-group-local-only` is a static self IP address.
- A *floating self IP address* is an IP address that two BIG-IP systems share. Any self IP address that you assign to the default traffic group `traffic-group-1` is a floating self IP address.

Self IP addresses and MAC addresses

For each self IP address that you create for a VLAN, the BIG-IP® system automatically assigns a media access control (MAC) address.

As an alternative, you can globally configure the BIG-IP system to assign the same MAC address to all VLANs. This feature is useful if your network includes a type of switch that does not keep a separate Layer 2 forwarding table for each VLAN on that switch.

Self IP addresses for SNATs

When you configure the BIG-IP® system to manage local area traffic, you can implement a feature known as a secure network address translation (SNAT). A *SNAT* is an object that causes the BIG-IP system to translate the original source IP address of a packet to an IP address that you specify. A SNAT ensures that the target server sends its response back through the BIG-IP system rather than to the original client IP address directly.

When you create a SNAT, you can configure the BIG-IP system to automatically choose a translation address. This ability of the BIG-IP system to automatically choose a translation address is known as *SNAT automapping*, and in this case, the translation address that the system chooses is always an existing self IP address. Thus, for traffic going from the BIG-IP system to a destination server, configuring SNAT automapping ensures that the source IP address in the header of a packet is a self IP address.

When you create an automapped SNAT, the BIG-IP system actually creates a SNAT pool consisting of the system's internal self IP addresses, and then uses an algorithm to select and assign an address from that SNAT pool.

Self IP address properties

It is when you initially run the Setup utility on a BIG-IP® system that you normally create any static and floating self IP addresses and assign them to VLANs. However, if you want to create additional self IP addresses later, you can do so using the BIG-IP Configuration utility.

Note: *Only users with either the Administrator or Resource Administrator user role can create and manage self IP addresses.*

Note: *A self IP address can be in either IPv4 or IPv6 format.*

IP address

A self IP address, combined with a netmask, typically represents a range of host IP addresses in a VLAN. If you are assigning a self IP address to a VLAN group, the self IP address represents the range of self IP addresses assigned to the VLANs in that group.

Netmask

When you specify a netmask for a self IP address, the self IP address can represent a range of IP addresses, rather than a single host address. For example, a self IP address of `10.0.0.100` can represent several host IP addresses if you specify a netmask of `255.255.0.0`.

VLAN/Tunnel assignment

You assign a unique self IP address to a specific VLAN or a VLAN group:

Assigning a self IP address to a VLAN

The self IP address that you assign to a VLAN should represent an address space that includes the self IP addresses of the hosts that the VLAN contains. For example, if the address of one destination server in a VLAN is `10.0.0.1` and the address of another server in the VLAN is `10.0.0.2`, you could assign a self IP address of `10.0.0.100`, with a netmask of `255.255.0.0`, to the VLAN.

Assigning a self IP address to a VLAN group

The self IP address that you assign to a VLAN group should represent an address space that includes the self IP addresses of the VLANs that you assigned to the group. For example, if the self IP address of one VLAN in a VLAN group is `10.0.20.100` and the address of the other VLAN in a VLAN group is `10.0.30.100`, you could assign an address of `10.0.0.100`, with a netmask of `255.255.0.0`, to the VLAN group.

The VLAN/Tunnel list in the BIG-IP Configuration utility displays the names of all existing VLANs and VLAN groups.

Port lockdown

Each self IP address has a feature known as port lockdown. *Port lockdown* is a security feature that allows you to specify particular UDP and TCP protocols and services from which the self IP address can accept traffic.

You can determine the supported protocols and services by using the `tmsh` command `tmsh list net self-allow defaults`.

If you do not want to use the default setting (**Allow None**), you can configure port lockdown to allow either all UDP and TCP protocols and services (**Allow All**) or only those that you specify (**Allow Custom**).

Note: *High availability-related traffic from configured peer devices in a device group might not be subject to port lockdown settings.*

Traffic groups

If you want the self IP address to be a *floating IP address*, that is, an address shared between two or more BIG-IP devices in a device group, you can assign a floating traffic group to the self IP address. A floating traffic group causes the self IP address to become a floating self IP address.

A floating self IP address ensures that application traffic reaches its destination. More specifically, a floating self IP address enables a source node to successfully send a request, and a destination node to successfully send a response, when the relevant BIG-IP device is unavailable.

If you want the self IP address to be a static (non-floating) IP address (used mostly for standalone devices), you can assign a non-floating traffic group to the self IP address. A non-floating traffic group causes the self IP address to become a non-floating self IP address. An example of a non-floating self IP address is the address that you assign to the default VLAN named HA, which is used strictly to process failover communications between BIG-IP devices, instead of processing application traffic.

Creating a self IP address

Before you create a self IP address, ensure that you have created a VLAN that you can associate with the self IP address.

A self IP address enables the BIG-IP® system and other devices on the network to route application traffic through the associated VLAN or VLAN group. When you do not intend to provision the vCMP® feature, you typically create self IP addresses when you initially configure the BIG-IP system on the VIPRION® platform.

If you plan to provision vCMP, however, you do not need to create self IP addresses during initial BIG-IP configuration. Instead, the host administrator creates VLANs for use by guests, and the guest administrators create self IP addresses to associate with those VLANs.

1. On the Main tab, click **Network** > **Self IPs**.
2. Click **Create**.
   The New Self IP screen opens.
3. In the **Name** field, type a unique name for the self IP address.
4. In the **IP Address** field, type an IPv4 or IPv6 address.

   This IP address should represent the address space of the VLAN that you specify with the **VLAN/Tunnel** setting.
5. In the **Netmask** field, type the full network mask for the specified IP address.
6. From the **VLAN/Tunnel** list, select the VLAN to associate with this self IP address.
   - On the internal network, select the internal or high availability VLAN that is associated with an internal interface or trunk.
   - On the external network, select the external VLAN that is associated with an external interface or trunk.
7. From the **Port Lockdown** list, select **Allow Default**.
8. From the **Traffic Group** list, retain the default value or select a traffic group.
9. Click **Finished**.
   The screen refreshes, and displays the new self IP address.

After you perform this task, the BIG-IP system can send and receive traffic through the specified VLAN or VLAN group. If the self IP address is member of a floating traffic group and you configure the system for redundancy, the self IP address can fail over to another device group member if necessary.

After creating the self IP address, ensure that you repeat this task to create as many self IP addresses as needed.

Creating a self IP for a VLAN group

Before you create a self IP address, ensure that you have created at least one VLAN group.

You perform this task to create a self IP address for a VLAN group. The self IP address for the VLAN group provides a route for packets destined for the network. With the BIG-IP® system, the path to an IP network is a VLAN. However, with the VLAN group feature used in this procedure, the path to the IP network `10.0.0.0` is actually through more than one VLAN. As IP routers are designed to have only one physical route to a network, a routing conflict can occur. With a self IP address on the BIG-IP system, you can resolve the routing conflict by associating a self IP address with the VLAN group.

1. On the Main tab, click **Network** > **Self IPs**.
2. Click **Create**.
   The New Self IP screen opens.
3. In the **IP Address** field, type an IPv4 or IPv6 address.

   This IP address should represent the address space of the VLAN group that you specify with the **VLAN/Tunnel** setting.
4. In the **Netmask** field, type the full network mask for the specified IP address.

   For example, you can type `ffff:ffff:ffff:ffff:0000:0000:0000:0000` or `ffff:ffff:ffff:ffff::`.
5. From the **VLAN/Tunnel** list, select the VLAN group with which to associate this self IP address.
6. From the **Port Lockdown** list, select **Allow Default**.
7. From the **Traffic Group** list, retain the default value or select a traffic group.
8. Click **Finished**.
   The screen refreshes, and displays the new self IP address.

The BIG-IP system can send and receive traffic through the specified VLAN or VLAN group.

Chapter 6

Packet Filters

Introduction to packet filtering

Packet filters enhance network security by specifying whether a BIG-IP® system interface should accept or reject certain packets based on criteria that you specify. Packet filters enforce an access policy on incoming traffic. They apply to incoming traffic only.

You implement packet filtering by creating packet filter rules, using the BIG-IP Configuration utility. The primary purpose of a packet filter rule is to define the criteria that you want the BIG-IP system to use when filtering packets. Examples of criteria that you can specify in a packet filter rule are:

- The source IP address of a packet
- The destination IP address of a packet
- The destination port of a packet

You specify the criteria for applying packet filter rules within an expression. When creating a packet filter rule, you can instruct the BIG-IP system to build an expression for you, in which case you need only choose the criteria from predefined lists, or you can write your own expression text, using the syntax of the `tcpdump` utility. For more information on the `tcpdump` utility, see the online man page for the `tcpdump` command.

Note: *Packet filter rules are unrelated to iRules®*

You can also configure global packet filtering that applies to all packet filter rules that you create.

Global settings

Global settings for packet filtering are divided into two categories: Properties and Exemptions. The BIG-IP® system applies global settings to all packets coming into the BIG-IP system.

Important: *Note that one of the global settings, Packet Filtering, enables packet filtering. When you disable this setting, no packet filter settings or packet filter rules operate, and the BIG-IP system allows all traffic by default.*

Global properties

You can configure three specific global properties for packet filtering.

Packet filter enabling

Before you can implement packet filtering on the BIG-IP® system, you must enable the packet filter feature. You do this by changing the **Packet Filtering** setting to **Enabled**. The default setting for packet filtering is **Disabled**.

Control of unhandled packets

Sometimes a packet does not match any of the criteria that you have specified in the packet filter rules that you have created. For this reason, you must configure the *Unhandled Packet Action* property, which specifies the action that the BIG-IP system should take when the packet does not match packet filter rule criteria.

Possible values for this setting are **Accept**, **Discard**, and **Reject**. The default value is **Accept**.

Warning: *Changing the default value of the Unhandled Packet Action property can produce unwanted consequences. Before changing this value to* ***Discard*** *or* ***Reject****, make sure that any traffic that you want the BIG-IP system to accept meets the criteria specified in your packet filter rules.*

Other options

Using the Options property, you can configure two other options:

Filter established connections

When you enable (check) this option, the BIG-IP system filters all ingress packets, even if the packets are part of an existing connection. The default setting is disabled (unchecked). Note that checking this option does not typically enhance security, and can impact system performance.

Send ICMP error on packet reject

When you enable (check) this option, the system sends an ICMP type 3 (destination unreachable), code 13 (administratively prohibited) packet when an ingress packet is rejected. When you disable (clear) this option, the BIG-IP system sends an ICMP reject packet that is protocol-dependent. The default setting for this option is disabled (cleared).

Global exemptions

There are a number of exemptions you can set for packet filtering. When filtering packets, the BIG-IP® system always applies these exemptions, effectively overriding certain criteria you might have previously set within an individual packet filter rule.

VLANs

Using the **VLANs** setting, you can configure the BIG-IP system so that traffic from one or more specified VLANs is exempt from packet filtering. In this case, the system does not attempt to match packets from the specified VLAN or VLANs to any packet filter rule. Instead, the BIG-IP system always accepts traffic from the specified VLAN or VLANs.

For example, if you specify VLAN internal, then no incoming packets from VLAN internal are subject to packet filtering, even if a packet matches the criteria of a packet filter rule.

Possible values are:

Always Accept
When you select this value, a VLAN List setting appears. You can then specify one or more VLANs from which traffic should be exempt from packet filtering.

None
When you select this value, traffic from all VLANs is subject to packet filtering, according to existing packet filter rule criteria. This is the default value.

Protocols

With the **Protocols** setting, you can specify whether ARP and certain ICMP messages are exempt from packet filtering. The individual settings are:

Always accept ARP
When you enable (check) this setting, the system automatically accepts all ARP packets and therefore does not subject them to packet filtering. The default setting is enabled (checked).

Always accept important ICMP
When you enable (check) this setting, the system automatically accepts the following ICMP packet types for IPv4, and therefore does not subject them to packet filtering:

- UNREACH
- SOURCEQUENCH
- REDIRECT
- TIMEXCEED

The default setting is enabled.

MAC addresses

You can use the **MAC Addresses** setting to exempt traffic from certain MAC addresses from packet filtering. Possible values are:

Always Accept
When you select this value, a MAC Address List setting appears. You can then specify one or more MAC addresses from which traffic should be exempt from packet filtering.

None
When you select this value, traffic from all MAC addresses is subject to packet filtering, according to existing packet filter rule criteria. This is the default value.

IP addresses

You can use the **IP Addresses** setting to exempt traffic from certain IP addresses from packet filtering. Possible values are:

Always Accept
When you select this value, an IP Address List setting appears. You can then specify one or more IP addresses from which traffic should be exempt from packet filtering.

None
When you select this value, traffic from all IP addresses is subject to packet filtering, according to existing packet filter rule criteria. This is the default value.

VLANs

Using the **VLANs** setting, you can configure the BIG-IP® system so that traffic from one or more specified VLANs is exempt from packet filtering. In this case, the system does not attempt to match packets from the specified VLAN or VLANs to any packet filter rule. Instead, the BIG-IP system always accepts traffic from the specified VLAN or VLANs.

For example, if you specify VLAN internal, then no incoming packets from VLAN internal are subject to packet filtering, even if a packet matches the criteria of a packet filter rule.

Possible values are:

Always Accept
When you select this value, a **VLAN List** setting appears. You can then specify one or more VLANs from which traffic should be exempt from packet filtering.

None
When you select this value, traffic from all VLANs is subject to packet filtering, according to existing packet filter rule criteria. This is the default value.

Order of packet filter rules

You use the **Order** setting to specify the order in which you want the BIG-IP® system to apply existing packet filter rules. This setting is required. Possible values for this setting are:

First
Select this value if you want this packet filter rule to be the first rule that the BIG-IP system applies.

Last
Select this value if you want this packet filter rule to be the last rule that the BIG-IP system applies.

After
Select this value, and then select a packet filter rule from the list, if you want the system to apply this packet filter after the packet filter that you select from the list. Note that this setting is most useful when you have more than three packet filter rules configured.

About the action setting in packet filter rules

When a packet matches the criteria that you have specified in a packet filter rule, the BIG-IP® system can take a specific action. You define this action using the **Action** setting. You can choose one of these actions:

Accept
Select **Accept** if you want the system to accept the packet, and stop processing additional packet filter rules, if any exist. This is the default setting.

Discard
Select **Discard** if you want the system to drop the packet, and stop processing additional packet filter rules, if any exist.

Reject
Select **Reject** if you want the system to drop the packet, and also send a rejection packet to the sender, indicating that the packet was refused. Note that the behavior of the system when you select the **Reject** action depends on how you configured the general packet filter Options property, Send ICMP Error on Packet Reject.

Continue
Select **Continue** if you simply want the system to acknowledge the packet for logging or statistical purposes. Setting the **Action** value to **Continue** does not affect the way that the BIG-IP system handles the packet; the system continues to evaluate traffic matching a rule, starting with the next packet filter rule listed.

Rate class assignment

Using the **Rate Class** setting, you can assign a rate class to traffic that matches the criteria defined in a packet filter rule. Note that this setting applies only when you have the rate shaping feature enabled.

The default value for this setting is None. If you previously created rate classes using the rate shaping feature, you can choose one of those rate classes from the **Rate Class** list.

One or more VLANs

You use the **Apply to VLAN** setting to display a list of VLANs and then select a VLAN or VLAN group name. Selecting a VLAN from the list means that the packet filter rule filters ingress traffic from that VLAN only. For example, if you select the value ***All VLANS**, the BIG-IP® system applies the packet filter rule to all traffic coming into the BIG-IP system.

Similarly, if you select the **VLAN internal**, the BIG-IP system applies the packet filter rule to traffic from VLAN internal only. The default value is ***All VLANS**.

If you select the name of a VLAN group instead of an individual VLAN, the packet filter rule applies to all VLANs in that VLAN group.

Logging

If you want to generate a log message each time a packet matches a rule, you can enable logging for the packet filter rule. With this configuration, you can then display the Logging screen in the BIG-IP Configuration utility and view events related to packet filtering.

About filter expression creation

To match incoming packets, the BIG-IP® system must use a filter expression. A *filter expression* specifies the criteria that you want the BIG-IP system to use when filtering packets. For example, the BIG-IP system can filter packets based on the source or destination IP address in the header of a packet.

Using the BIG-IP Configuration utility, you can create a filter expression in either of two ways:

- You can write your own expression, using a Filter Expression box.
- You can specify a set of criteria (such as source or destination IP addresses) that you want the BIG-IP system to use when filtering packets. When you use this method, the BIG-IP system builds a filter expression for you.

You can have as many rules as you want, limited only by the available memory. Of course, the more statements you have, the more challenging it is to understand and maintain your packet filters.

Enabling packet filtering

Before creating a packet filtering rule, you must enable packet filtering. When you enable packet filtering, you can specify the MAC addresses, IP addresses, and VLANs that you want to be exempted from packet filter evaluation.

1. On the Main tab, click **Network** > **Packet Filters**.
   The Packet Filters screen opens.
2. From the **Packet Filtering** list, select **Enabled**.
3. From the **Unhandled Packet Action** list, select **Accept**.
4. For the **Options** setting, retain the default value or select the check boxes as needed.
5. For the **Protocols** setting, retain the default value or clear the check boxes as needed.
6. From the **MAC Addresses** list, specify a value:

Value	Description
None	When you select this value, all MAC addresses are exempt from packet filter evaluation.
Always Accept	When you select this value, you can specify the MAC addresses that are exempt from packet filter evaluation, and the BIG-IP Configuration utility displays additional settings.

7. If you directed the **MAC Addresses** setting to always accept specific MAC addresses, provide the details:
   a) In the **Add** field, type a MAC address and click **Add**.
   The MAC address appears in the **MAC Address List** field.
   b) Repeat this step for each MAC address that you want the system to exempt from packet filter evaluation.

8. From the **IP Addresses** list, specify a value:

Value	Description
None	When you select this value, all IP addresses are exempt from packet filter evaluation.
Always Accept	When you select this value, you can specify the IP addresses that are exempt from packet filter evaluation. The BIG-IP Configuration utility displays additional settings.

9. If you directed the **IP Addresses** setting to always accept specific IP addresses, provide the details:
   a) In the **Add** field, type an IP address and click **Add**.
   The IP address appears in the **IP Address List** field.
   b) Repeat this step for each IP address that you want the system to exempt from packet filter evaluation.

10. From the **VLANs** list, specify a value:

Value	Description
None	When you select this value, all VLANs are exempt from packet filter evaluation.
Always Accept	When you select this value, you can specify the VLANs that are exempt from packet filter evaluation. The BIG-IP Configuration utility displays additional settings.

11. If you configured the **VLANs** setting to always accept specific VLANs, then use the **Move** button to move one or more VLAN names from the **Available** list to the **Selected** list.
12. Click **Update**.

After you enable packet filtering, the BIG-IP® system filters packets according to the criteria in the packet filter rule and the values you configured when enabling the packet filter.

Creating a packet filter rule

When implementing packet filtering, you need to create a packet filter rule.

1. On the Main tab, click **Network** > **Packet Filters**.
   The Packet Filters screen opens.
2. Click **Rules**.
3. Click **Create**.
4. Name the rule.
5. From the **Order** list, select **First**.
6. From the **Action** list, select **Reject**.
7. From the **VLAN / Tunnel** list, select **internal**.

8. From the **Logging** list, select **Enabled**.
9. From the **Filter Expression Method** list, select **Enter Expression Text**.
10. In the **Filter Expression** field, type an expression.
    For example: `not dst port 80 and not dst port 443 and not dst port 53 and not dst port 22 and not dst port 20 and not dst port 21 and not dst host <internal self IP address>`

    Note: *Replace `<internal self IP address>` with the actual self IP address of VLAN internal.*

11. Click **Finished**.

The packet filter rule is available.

Chapter 7

NATS and SNATs

Introduction to NATs and SNATs

You can configure the BIG-IP® system to translation IP addresses in packets that pass through the system. You can configure objects for both network address translation (NATs) and source network address translation (SNATs).

Comparison of NATs and SNATs

A SNAT is similar to a NAT, except for the differences listed in this table.

NATs	SNATs
You can map only one original address to a translation address.	You can map multiple original addresses to a single translation address. You can even map all node addresses on your network to a single public IP address, in a single SNAT object.
All ports on the internal node are open.	By default, SNATs support UDP and TCP only. This makes a SNAT more secure than a NAT.
Local Traffic Manager™ does not track NAT connections.	Local Traffic Manager tracks SNAT connections, which, in turn, allows SNATs and virtual servers to use the same public IP addresses.
You must explicitly enable a NAT on the internal VLAN where the internal node's traffic arrives on the BIG-IP® system.	By default, a SNAT that you create is enabled on all VLANs.

About NATs

In some cases, you might want to allow a client on an external network to send a request directly to a specific internal node (thus bypassing the normal load balancing server selection). To send a request directly to an internal server, a client normally needs to know the internal node's IP address, which is typically a private class IP address. Because private class IP addresses are non-routable, you can instead create a network translation address (NAT). A *NAT* is a feature of BIG-IP® Local Traffic Manager™ that provides a routable IP address that an external node can use to send traffic to, or receive traffic from, an internal node.

More specifically, a NAT is an address translation object that instructs Local Traffic Manager (LTM®) to translate one IP address in a packet header to another IP address. A NAT consists of a one-to-one mapping of a public IP address to an internal private class IP address.

You can use a NAT in two different ways:

To translate a private class destination address to a public address
When an external node sends traffic to the public IP address defined in a NAT, Local Traffic Manager automatically translates that destination address to the associated private class IP address, which represents a specific node on the internal network. This translation is hidden from the external node that sent the traffic.

To translate a private class source address to a public address
You can also use a NAT to translate an internal node's private class source IP address to a public IP address. This translation is hidden from the external node that receives the traffic.

To summarize, a NAT provides a routable address for sending packets to or from a node that has a private class IP address.

When you create a NAT, you can map only one private class IP address to a specific public IP address. That is, a NAT always represents a one-to-one mapping between a private class IP address and a public IP address. If you want to map more than one private class IP address (that is, multiple internal nodes) to a single public IP address, you can create a SNAT instead.

Note: *NATs do not support port translation, and are not appropriate for protocols that embed IP addresses in the packet, such as FTP, NT Domain, or CORBA IIOP.*

Tip: *When you use a NAT to provide access to an internal node, all ports on that internal node are open. To mitigate this security risk, consider using a SNAT instead.*

Local Traffic Manager can apply a NAT to either an inbound or an outbound connection.

NATs for inbound connections

With respect to NATs, an *inbound* connection is a connection that is initiated by a node on an external network, and comes into the BIG-IP® system to a node on the internal network.

Without a NAT

Normally, traffic coming into the BIG-IP system is load balanced to a server in a pool, based on the load balancing method configured for that pool, in the following way:

- A client on an external network typically sends traffic to a virtual server on the BIG-IP system. The destination IP address in this case is the virtual server address.

- Upon receiving a packet, the virtual server typically translates that destination IP address to the IP address of a pool member, for the purpose of load balancing that packet.
- The pool member then sends its response back through the BIG-IP system, using a route specified in the server node's routing table (ideally, a floating IP address assigned to an internal VLAN). On receiving the response, Local Traffic Manager™ then performs the reverse translation; that is, the system translates the pool member's actual source address to the virtual server address. This results in the source address in the response to the client being the virtual server address, which is the source address that the client expects to see.

This typical load balancing scenario ensures that for load balanced traffic, the client system never sees the internal private class IP address of an internal node.

With a NAT

If the client system wants to bypass the load balancing mechanism to send packets directly to a specific node on the internal network, the client needs a routable IP address to use to send packets to that server node.

A NAT solves this problem by providing a routable address that a client can use to make a request to an internal server directly. In this way, a NAT performs the same type of address translation that a virtual server performs when load balancing connections to pool members. In the case of a NAT, however, no load balancing occurs, because the client is sending a request to a specific node. The NAT translates the public destination IP address in the request to the private class IP address of the internal node.

When the server node sends the response, Local Traffic Manager performs the reverse translation, in the same way that a virtual server behaves.

Note: *Local Traffic Manager does not track NAT connections. Therefore, the public IP address that you define in a NAT cannot be the same address as a virtual address or SNAT address.*

For example, suppose a node on the internal network (such as a load balancing server) has a private class IP address of `172.16.20.3`. You can create a NAT designed to translate a public destination address of your choice (such as `207.10.1.103`) to the private class address `172.16.20.3`. Consequently, whenever a node on the external network initiates a connection to the address `207.10.1.103`, Local Traffic Manager translates that public destination address to the private class address `172.16.20.3`.

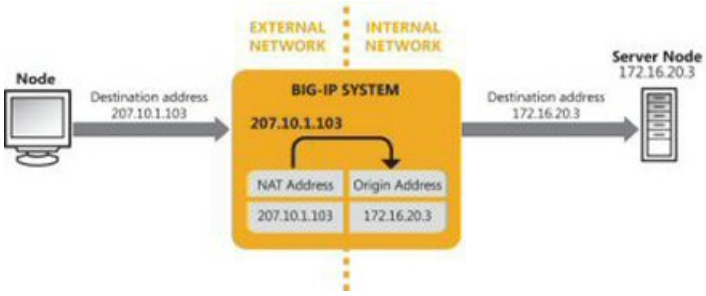

Figure 8: Sample NAT for an inbound connection

In this example, the NAT provides a routable address for an external node to initiate a connection to an internal node.

When you create a NAT, you must define two settings: NAT Address and Origin Address. In our example:

- The NAT address is `207.10.1.103`, and the origin address is `172.16.20.3`.
- The connection is an inbound connection, meaning that the connection is being initiated from the external network, through the BIG-IP system, to the internal network.
- Local Traffic Manager translates the NAT address to the origin address.
- The NAT address and the origin address are destination addresses.

NATs for outbound connections

The previous section summarized how a BIG-IP® system normally load balances incoming traffic, and translates the source IP address in a response back to the virtual address.

Sometimes, however, an internal node needs to initiate a connection, rather than simply respond to a request. When a node on an internal network initiates a connection, the connection is considered to be an *outbound* connection. In this case, because the outgoing packets do not represent a response to a load-balanced request, the packets do not pass through a virtual server, and therefore the system does not perform the usual source IP address translation.

Without a NAT, the source IP address is a non-routable address. With a NAT, however, Local Traffic Manager™ translates the internal node's private class IP address to a public IP address, to which the external node can then route its response.

For example, suppose an internal node (such as a mail server) has a private class IP address of `172.16.20.1`. You can create a NAT designed to translate the private class address `172.16.20.1` to a public source address of your choice (such as `207.10.1.101`). Consequently, whenever the internal node `172.16.20.1` initiates a connection destined for a node on the external network, the system translates that source address of `172.16.20.1` to its public address (`207.10.1.101`).

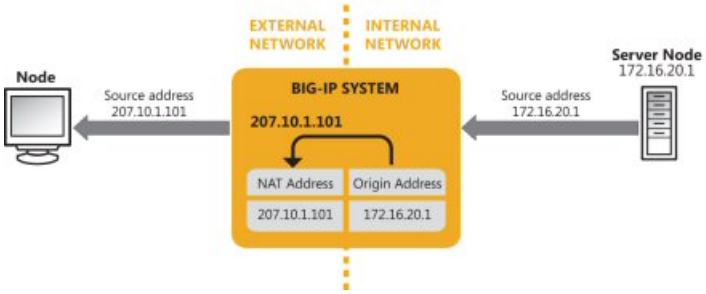

Figure 9: Sample NAT for an outbound connection

In this example, the NAT provides a way for an internal node to initiate a connection to a node on an external network, without showing a private class IP address as the source address.

A NAT has two settings; **NAT Address** and **Origin Address**. In this example:

- The NAT address is `207.10.1.101`, and the origin address is `172.16.20.1`.
- The connection is an outbound connection, meaning that the connection is being initiated from the internal network, through Local Traffic Manager, to the external network.
- Local Traffic Manager translates the origin address to the NAT address.
- The origin address and the NAT address are source addresses.

A NAT always represents a one-to-one mapping between a public address and a private class address. However, if you would like to map multiple internal nodes to a single public address, you can use a secure network translation address (SNAT) instead of a NAT. You can use SNATs for outbound connections only.

Creating a NAT

You enable network address translation (NAT) so that the BIG-IP® system can translate one IP address to another.

1. On the Main tab, click **Local Traffic** > **Address Translation** > **NAT List**.
   This displays a list of NATs on the system.

2. Click the **Create** button.
3. In the **Name** field, type a name for the NAT.
4. In the **NAT Address** field, type the IP address that you want to use as the translation address, that is, the address to which you want to translate the origin address.
5. In the **Origin Address**field, type the IP address of the node that you want traffic to reach when the BIG-IP system applies the NAT address.
6. Configure the remaining settings or retain the default values.

After you perform this task, the NAT appears in the list of NATs on the system.

About SNATs

When you need to ensure that server responses always return through the BIG-IP® system, or when you want to hide the source addresses of server-initiated requests from external devices, you can implement a SNAT.

A *secure network address translation (SNAT)* is a BIG-IP Local Traffic Manager™ feature that translates the source IP address within a connection to a BIG-IP system IP address that you define. The destination node then uses that new source address as its destination address when responding to the request.

For inbound connections, that is, connections initiated by a client node, SNATs ensure that server nodes always send responses back through the BIG-IP system, when the server's default route would not normally do so. Because a SNAT causes the server to send the response back through the BIG-IP system, the client sees that the response came from the address to which the client sent the request, and consequently accepts the response.

For outbound connections, that is, connections initiated by a server node, SNATs ensure that the internal IP address of the server node remains hidden to an external host when the server initiates a connection to that host.

Important: *F5 recommends that before implementing a SNAT, you understand NATs.*

SNATs for client-initiated (inbound) connections

In the most common client-server network configuration, the Local Traffic Manager™ standard address translation mechanism ensures that server responses return to the client through the BIG-IP® system, thereby reversing the original destination IP address translation. This typical network configuration is as follows:

- The server nodes are on the same subnet as the BIG-IP system.
- The client nodes are on a different subnet from the server nodes.
- The BIG-IP system is the default gateway for the server subnet.

However, there are atypical network configurations in which the standard BIG-IP system address translation sequence by itself does not ensure that server responses use the required return path. Examples of these atypical configurations are:

When clients and servers are on the same network

If you want to load balance requests to server nodes that are on the same network as the client nodes, you can create a SNAT so that server responses are sent back through the virtual server, rather than directly from the server node to the client node. Otherwise, problems can occur such as the client rejecting the response because the source of the response does not match the destination of the request. Known as *virtual server bounceback*, this SNAT configuration causes the source of the response to match the

destination of the request, thus ensuring that the client node accepts the response. You can use this kind of configuration when you want to load balance requests from web servers to application servers on the same network.

When the default gateway of the server node is not the BIG-IP system

For various reasons, the server node's default route cannot always be defined to be a route back through the BIG-IP system. Again, this can cause problems such as the client rejecting the response because the source of the response does not match the destination of the request. The solution is to create a SNAT. When Local Traffic Manager then translates the client node's source IP address in the request to the SNAT address, this causes the server node to use that SNAT address as its destination address when sending the response. This, in turn, forces the response to return to the client node through the BIG-IP system rather than through the server node's default gateway.

When using the OneConnect feature

Local Traffic Manager OneConnect™ feature allows client requests to re-use idle server-side connections. Without a SNAT, the source IP address in the server-side connection remains the address of the client node that initially established the connection, regardless of which other client nodes re-use the connection. Although this is not an issue for traffic routing, you might find it confusing when examining various types of system output. A SNAT solves this problem.

Note: *Using a SNAT for inbound connections can impact the availability of ephemeral ports. This can lead to the SNAT being unable to process additional connections until some source ports become available.*

This image shows a typical problem for client-initiated connections when Local Traffic Manager is not defined as the server's default gateway, and you have not configured a SNAT for inbound traffic.

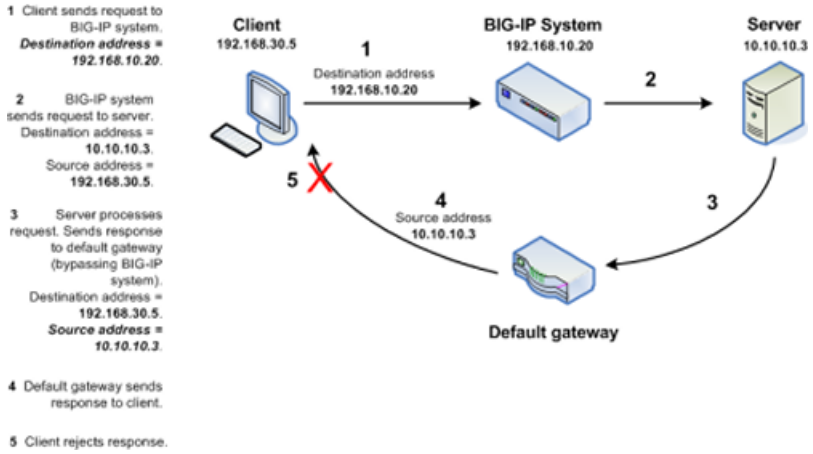

Figure 10: Client rejects response due to non-matching destination and source IP addresses

To prevent these problems, you can configure an inbound SNAT. An *inbound SNAT* translates the original client source IP address in a request to a BIG-IP system virtual server or BIG-IP system self IP address, forcing subsequent server response to return directly to Local Traffic Manager. When an inbound SNAT is configured on the system, Local Traffic Manager translates not only the destination IP address in the request (using the standard address translation mechanism), but also the source IP address in the request (using a SNAT).

The figure below shows that by configuring a SNAT, you ensure that the response returns through the BIG-IP system instead of through the default gateway, thus ensuring that the client can accept the server response.

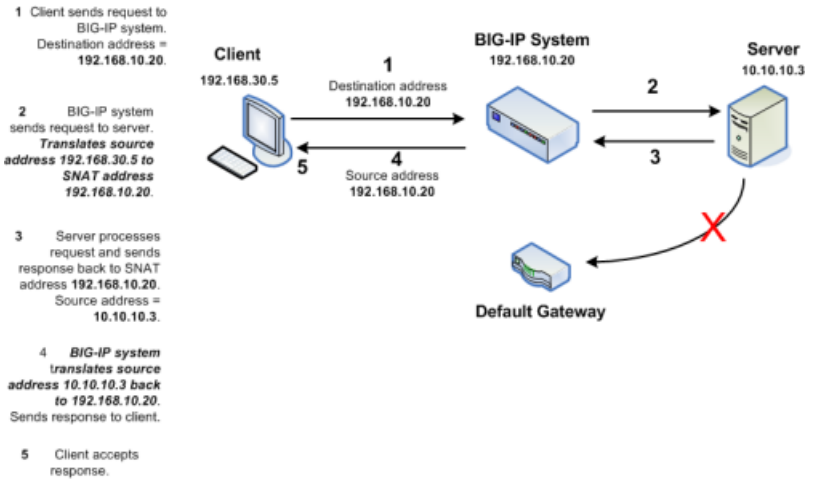

Figure 11: Client accepts response due to matching destination and source IP addresses

SNATs for server-initiated (outbound) connections

When an internal server initiates a connection to an external host, a SNAT can translate the private, source IP addresses of one or more servers within the outgoing connection to a single, publicly-routable address. The external destination host can then use this public address as a destination address when sending the response. In this way, the private class IP addresses of the internal nodes remain hidden from the external host.

More specifically, a SNAT for an outgoing connection works in the following way:

1. Local Traffic Manager™ receives a packet from an original IP address (that is, an internal server with a private IP address) and checks to see if that source address is defined in a SNAT.
2. If the original IP address is defined in a SNAT, Local Traffic Manager changes that source IP address to the translation address defined in the SNAT.
3. Local Traffic Manager then sends the packet, with the SNAT translation address as the source address, to the destination host.

In this example of an outgoing SNAT, Local Traffic Manager causes three internal nodes, with the IP addresses `172.16.20.4`, `172.16.20.5`, and `172.16.20.6`, to advertise the public IP address `207.10.1.102` as the source IP address in the three outgoing connections.

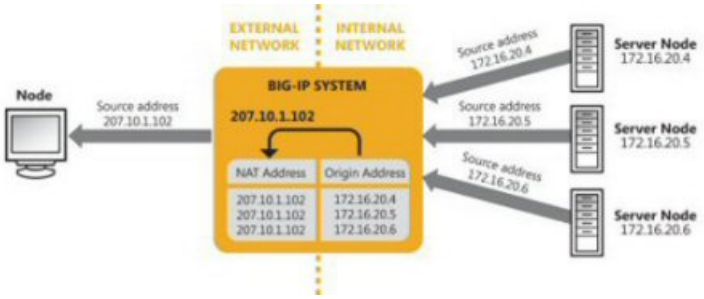

Figure 12: Sample SNAT for multiple outgoing connections

SNAT types

The types of SNATs you can create are:

Standard SNAT

A standard SNAT is an object you create, using the BIG-IP Configuration utility, that specifies the mapping of one or more original IP addresses to a translation address. For this type of SNAT, the criteria that the BIG-IP™ system uses to decide when to apply the translation address is based strictly on the original IP address. That is, if a packet arrives from the original IP address that you specified in the SNAT, then the BIG-IP system translates that address to the specified translation address. There are three types of standard SNATs that you can create:

- A SNAT in which you specify a specific translation address
- A SNAT that uses the automap feature
- A SNAT in which you specify a SNAT pool as your translation address

SNAT pool assigned as a virtual server resource

This type of SNAT consists of just a SNAT pool that you directly assign as a resource to a virtual server. When you implement this type of SNAT, you create a SNAT pool only; you do not need to create a SNAT object or an iRule.

Intelligent SNAT

Like a standard SNAT, an intelligent SNAT is the mapping of one or more original IP addresses to a translation address. However, you implement this type of SNAT mapping within an iRule instead of by creating a SNAT object. For this type of SNAT, the criteria that Local Traffic Manager uses to decide when to apply a translation address is based on any piece of data you specify within the iRule, such as an HTTP cookie or a server port.

About translation addresses

You can specify the translation addresses that you want to map to your original IP addresses. A translation address can be in these three forms:

An IP Address

When creating a SNAT, you can specify a particular IP address that you want the SNAT to use as a translation address.

A SNAT pool

Specifying this value allows you to specify an existing SNAT pool to which you want to map your original IP address.

SNAT automap

Similar to a SNAT pool, the SNAT automap feature allows you to map one or more original IP addresses to a pool of translation addresses. With the SNAT automap feature, you do not need to create the pool. Instead, Local Traffic Manager™ effectively creates a pool for you, using self IP addresses as the translation addresses for the pool.

Original IP addresses

You can specify the original IP addresses that you want to map to translation addresses. You can specify one IP address or multiple IP addresses.

VLAN traffic

You can specify one or more VLANs to which you want the SNAT to apply.

Creating a SNAT

A *secure network address translation (SNAT)* is a BIG-IP® feature that translates the source IP address within a connection to a BIG-IP system IP address that you define. The destination node then uses that new source address as its destination address when responding to the request.

1. On the Main tab, click **Local Traffic** > **Address Translation**.
   The **SNAT List** screen displays a list of existing SNATs.
2. Click **Create**.
3. Name the new SNAT.
4. From the **Translation** list, select a translation type to use.
5. From the **VLAN / Tunnel Traffic** list, select **Enabled on**.
6. For the **VLAN / Tunnel List** setting, in the **Available** field, select **external** and **internal**, and using the Move button, transfer the VLANs to the **Selected** field.
7. From the **Auto Last Hop** list, select a value or retain the default value, **Default**.

   Selecting **Default** causes the BIG-IP system to behave according to the value of the system-wide **Auto Last Hop** setting, rather than enabling **Auto Last Hop** for this SNAT only.
8. Click the **Finished** button.

After you perform this task, the SNAT appears in the list of SNATs on the system

You must assign the SNAT to a virtual server.

Creating a SNAT pool

You create a SNAT pool on the BIG-IP® system to identify IP addresses that you want the BIG-IP system to use as a SNAT translation address.

Note: *In a device service clustering configuration, you perform this task on only one device in the device group. You will later synchronize this configuration to the other devices in the device group.*

---

1. On the Main tab, click **Local Traffic** > **Address Translation** > **SNAT Pool List**.
   The SNAT Pool List screen displays a list of existing SNATs.
2. Click **Create**.
3. In the **Name** field, type a name for the SNAT pool.
   An example of a name is `snat-pool-1`.
4. For the **Member List** setting:
   a) In the **IP Address** field, type an IP address.
      The BIG-IP system uses this address as a SNAT translation address.

      Important: *This address must NOT be on a directly-connected network.*

      ---

   b) Click **Add**.

c) Repeat these steps for each IP address that you want to include in the SNAT pool.

5. Click the **Finished** button.

After you perform this task, a SNAT pool resides on the BIG-IP system.

To enable the SNAT pool, you must assign the SNAT pool to a SNAT object, or specify the SNAT pool within an iRule.

Chapter 8

Route Domains

What is a route domain?

A *route domain* is a configuration object that isolates network traffic for a particular application on the network.

Because route domains segment network traffic, you can assign the same IP address or subnet to multiple nodes on a network, provided that each instance of the IP address resides in a separate routing domain.

Note: *Route domains are compatible with both IPv4 and IPv6 address formats.*

Important: *For systems that include both BIG-IP® Local Traffic Manager™ (LTM) and BIG-IP Global Traffic Manager™ (GTM), you can configure route domains on internal interfaces only.*

Benefits of route domains

Using the route domains feature of the BIG-IP® system, you can provide hosting service for multiple customers by isolating each type of application traffic within a defined address space on the network.

With route domains, you can also use duplicate IP addresses on the network, provided that each of the duplicate addresses resides in a separate route domain and is isolated on the network through a separate VLAN. For example, if you are processing traffic for two different customers, you can create two separate route domains. The same node address (such as `10.0.10.1`) can reside in each route domain, in the same

pool or in different pools, and you can assign a different monitor to each of the two corresponding pool members.

Sample partitions with route domain objects

This illustration shows two route domain objects on a BIG-IP system, where each route domain corresponds to a separate customer, and thus resides in its own partition. Within each partition, the customer created the network objects and local traffic objects required for that customer's application (`AppA` or `AppB`).

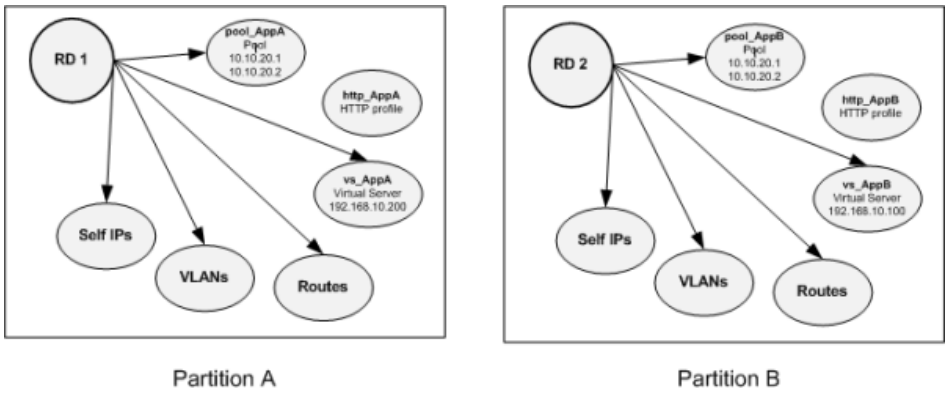

Figure 13: Sample partitions with route domains

Sample route domain deployment

A good example of the use of route domains is a configuration for an ISP that services multiple customers, where each customer deploys a different application. In this case, the BIG-IP system isolates traffic for two different applications into two separate route domains. The routes for each application's traffic cannot cross route domain boundaries because cross-routing restrictions are enabled on the BIG-IP system by default.

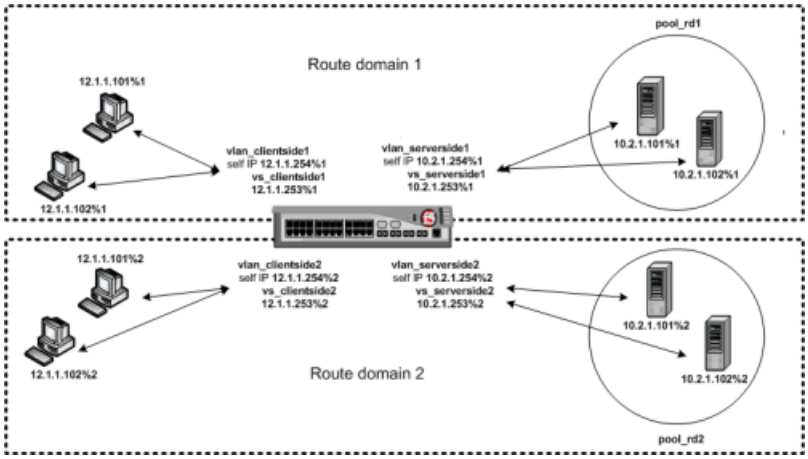

Figure 14: A sample route domain deployment

About route domain IDs

A *route domain ID* is a unique numerical identifier for a route domain. You can assign objects with IP addresses (such as self IP addresses, virtual addresses, pool members, and gateway addresses) to a route domain by appending the %*ID* to the IP address.

The format required for specifying a route domain ID in an object's IP address is `A.B.C.D%ID`, where *ID* is the ID of the relevant route domain. For example, both the local traffic node object `10.10.10.30%2` and the pool member `10.10.10.30%2:80` pertain to route domain `2`.

The BIG-IP system includes a default route domain with an ID of `0`. If you do not explicitly create any route domains, all routes on the system pertain to route domain `0`.

Important: *A route domain ID must be unique on the BIG-IP system; that is, no two route domains on the system can have the same ID.*

Traffic forwarding across route domains

You can create a parent-child relationship between two route domains, and configure strict isolation, to control the extent to which the BIG-IP® system can forward traffic from one route domain to another.

About parent IDs

When you create a route domain, you can specify the ID of another route domain as the parent route domain. The *parent ID* identifies another route domain that the system can search to find a route if the system cannot find the route within the child route domain.

For example, using the BIG-IP® Configuration utility, suppose you create route domain `1` and assign it a parent ID of `0`. For traffic pertaining to route domain `1`, the system looks within route domain `1` for a route for the specified destination. If no route is found, the system searches the routes in route domain `0`.

By default, if the system finds no route in the parent route domain, the system searches the parent route domain's parent, and so on, until the system finds either a match or a route domain with no parent. In the latter case, the system refrains from searching any other route domains to find a match, thus preventing the system from using a route from another route domain.

You can disable this behavior on a route domain.

About strict isolation

You can control the forwarding of traffic across route domain boundaries by configuring the *strict isolation* feature of a route domain:

- If strict isolation is enabled, the BIG-IP® system allows traffic forwarding from that route domain to the specified parent route domain only. This is the default behavior. Note that for successful isolation, you must enable the strict isolation feature on both the child and the parent route domains.
- If strict isolation is disabled, the BIG-IP system allows traffic forwarding from that route domain to any route domain on the system, without the need to define a parent-child relationship between route domains.

Note that in this case, for successful forwarding, you must disable the strict isolation feature on both the forwarding route domain and the target route domain (that is, the route domain to which the traffic is being forwarded).

About default route domains for administrative partitions

The route domains feature includes the concept of default route domains, to minimize the need for you to specify the %ID notation. When you designate a route domain as the *default route domain* in a partition, any BIG-IP system objects in that partition that do not include the %ID notation in their IP addresses are automatically associated with the default route domain.

The default route domain for partition Common

The BIG-IP system, by default, includes one route domain, named route domain `0`. Route domain `0` is known as the *default route domain* on the BIG-IP system, and this route domain resides in administrative partition `Common`. If you do not create any other route domains on the system, all traffic automatically pertains to route domain `0`.

If you want to segment traffic into multiple route domains, you can create additional route domains in partition `Common` and then segment application traffic among those route domains. Any BIG-IP addresses that do not include the route domain ID notation are automatically associated with the default route domain.

Note: *Any VLANs that reside in partition* `Common` *are automatically assigned to the default route domain.*

The default route domain for other partitions

For administrative partitions other than `Common`, you can create a route domain and designate it as a *partition default route domain*. A partition can contain one partition default route domain only.

The benefit of having a partition default route domain is that when you create objects such as a virtual server and pool members within that partition, you do not need to specify the ID of that default route domain within the addresses for those objects. For example, if you create a partition default route domain with an ID of `2` in partition `A`, the system automatically assigns any partition `A` object IP addresses without a route domain ID to route domain `2`.

If no partition default route domain exists within the partition, the system associates those addresses with route domain `0` in partition `Common`.

About VLAN and tunnel assignments for a route domain

You can assign one or more VLANs, VLAN groups, or tunnels to a route domain. The VLANs, VLAN groups, or tunnels that you assign to a route domain are those pertaining to the particular traffic that you want to isolate in that route domain. Each VLAN, VLAN group, or tunnel can be a member of one route domain only.

When you assign a VLAN group to a route domain, the BIG-IP system automatically assigns the VLAN group members to the route domain.

Please note the following facts:

- If you delete a VLAN group from the system, the VLAN group members remain assigned to the route domain.

- If a VLAN is assigned to a non-default route domain and you delete that route domain, the BIG-IP system automatically assigns the VLAN to the default route domain for that partition.
- When you create VLANs, VLAN groups, and tunnels, the BIG-IP system automatically assigns them to the default route domain of the current partition. You can change this assignment when you create other route domains in the partition.

Important: *You cannot assign a VLAN that resides in partition* `Common` *to a route domain in another partition.*

About advanced routing modules for a route domain

For each route domain that you configure, you can enable one or more dynamic routing protocols, as well as the network protocol Bidirectional Forwarding Detection (BFD). Use of dynamic routing and BFD for route domain `0` or any other route domain is optional.

About throughput limits on route domain traffic

When you configure more than one route domain on the BIG-IP system, the traffic from one particular route domain can potentially consume an inordinate amount of BIG-IP system resource. To prevent this, you can define the amount of BIG-IP system resource that traffic for each route domain can consume.

You do this by assigning a different throughput limit to each route domain. This throughput limit is defined in a *bandwidth controller policy*. For example, for route domain `1`, you can assign a static bandwidth controller policy that specifies a throughput limit of 10 Gbps, while for route domain `2`, you can assign a static bandwidth controller policy that specifies a throughput limit of 20 Gbps. When you assign a different bandwidth controller policy to each route domain, traffic for one route domain does not cross the boundary into another route domain on the system.

Important: *The BIG-IP system applies a bandwidth controller policy to a route domain's egress traffic only, that is, the traffic that a server within a particular route domain sends back through the BIG-IP system on its way to the client on the public network. A bandwidth controller policy is not applied to traffic coming from the public network to the route domain on the internal network.*

Creating a route domain on the BIG-IP system

Before you create a route domain:

- Ensure that an external and an internal VLAN exist on the BIG-IP® system.
- If you intend to assign a static bandwidth controller policy to the route domain, you must first create the policy. You can do this using the BIG-IP Configuration utility.
- Verify that you have set the current partition on the system to the partition in which you want the route domain to reside.

You can create a route domain on BIG-IP system to segment (isolate) traffic on your network. Route domains are useful for multi-tenant configurations.

1. On the Main tab, click **Network** > **Route Domains**.
   The Route Domain List screen opens.
2. Click **Create**.
   The New Route Domain screen opens.
3. In the **Name** field, type a name for the route domain.

   This name must be unique within the administrative partition in which the route domain resides.
4. In the **ID** field, type an ID number for the route domain.

   This ID must be unique on the BIG-IP system; that is, no other route domain on the system can have this ID.
5. In the **Description** field, type a description of the route domain.
   For example: `This route domain applies to traffic for application MyApp.`
6. For the **Strict Isolation** setting, select the **Enabled** check box to restrict traffic in this route domain from crossing into another route domain.
7. For the **Parent Name** setting, retain the default value.
8. For the **VLANs** setting, from the **Available** list, select a VLAN name and move it to the **Members** list.

   Select the VLAN that processes the application traffic relevant to this route domain.

   Configuring this setting ensures that the BIG-IP system immediately associates any self IP addresses pertaining to the selected VLANs with this route domain.
9. For the **Dynamic Routing Protocols** setting, from the **Available** list, select one or more protocol names and move them to the **Enabled** list.

   You can enable any number of listed protocols for this route domain.
10. From the **Bandwidth Controller** list, select a static bandwidth control policy to enforce a throughput limit on traffic for this route domain.
11. From the **Partition Default Route Domain** list, select either **Another route domain (0) is the Partition Default Route Domain** or **Make this route domain the Partition Default Route Domain**.

    This setting does not appear if the current administrative partition is partition `Common`.

    When you configure this setting, either route domain `0` or this route domain becomes the default route domain for the current administrative partition.
12. Click **Finished**.
    The system displays a list of route domains on the BIG-IP system.

You now have another route domain on the BIG-IP system.

Chapter 9

Static Routes

Static route management on the BIG-IP system

Part of managing routing on a BIG-IP® system is to add static routes for destinations that are not located on the directly-connected network. If you are using the route domains feature, you can specify a route domain ID as part of each IP address that you include in a static route entry.

Adding a static route

Before adding a route, if the IP addresses in the route pertain to any route domains, verify that the relevant route domains are present on the system.

Perform this task when you want to explicitly add a route for a destination that is not on the directly-connected network. Depending on the settings you choose, the BIG-IP system can forward packets to a specified network device (such as a next-hop router or a destination server), or the system can drop packets altogether.

1. On the Main tab, click **Network** > **Routes**.
2. Click **Add**.
   The New Route screen opens.
3. In the **Name** field, type a unique user name.
   This name can be any combination of alphanumeric characters, including an IP address.
4. In the **Description** field, type a description for this route entry.
   This setting is optional.
5. In the **Destination** field, type either the destination IP address for the route, or IP address `0.0.0.0` for the default route.
   This address can represent either a host or a network. Also, if you are using the route domains and the relevant route domain is the partition default route domain, you do not need to append a route domain ID to this address.
6. In the **Netmask** field, type the network mask for the destination IP address.
7. From the **Resource** list, specify the method through which the system forwards packets:

Option	Description
Use Gateway	Select this option when you want the next hop in the route to be a network IP address. This choice works well when the destination is a pool member on the same internal network as this gateway address.
Use Pool	Select this option when you want the next hop in the route to be a pool of routers instead of a single next-hop router. If you select this option, verify that you have created a pool on the BIG-IP system, with the routers as pool members.
Use VLAN/Tunnel	Select this option when you want the next hop in the route to be a VLAN or tunnel. This option works well when the destination address you specify in the routing entry is a network address. Selecting a VLAN/tunnel name as the resource implies that the specified network is directly connected to the BIG-IP system. In this case, the BIG-IP system can find the destination host simply by sending an ARP request to the hosts in the specified VLAN, thereby obtaining the destination host's MAC address.
Reject	Select this option when you want the BIG-IP system to reject packets sent to the specified destination.

8. In the **MTU** field, specify in bytes a maximum transmission unit (MTU) for this route.
9. Click **Finished**.

After you perform this task, a static route is defined on the BIG-IP system with IP addresses that can pertain to one or more route domains.

You should define a default route for each route domain on the system. Otherwise, certain types of administrative traffic that would normally use a TMM interface might instead use the management interface.

Chapter 10

Dynamic Routing

Dynamic routing on the BIG-IP system

By enabling and configuring any of the BIG-IP® advanced routing modules, you can configure dynamic routing on the BIG-IP system. You enable one or more advanced routing modules, as well as the Bidirectional Forwarding Detection (BFD) protocol, on a per-route-domain basis. Advanced routing module configuration on the BIG-IP system provides these functions:

- Dynamically adds routes to the Traffic Management Microkernel (TMM) and host route tables.
- Advertises and redistributes routes for BIG-IP virtual addresses to other routers.
- When BFD is enabled, detects failing links more quickly than would normally be possible using the dynamic routing protocols' own detection mechanisms.

Note: *On the BIG-IP system, directly-connected and static routes take precedence over dynamically-learned routes.*

Supported protocols for dynamic routing

The BIG-IP® advanced routing modules support these protocols.

Table 2: Dynamic routing protocols

Protocol Name	Description	Daemon	IP version supported
BFD	*Bidirectional Forwarding Detection* is a protocol that detects faults between two forwarding engines connected by a link. On the BIG-IP system, you can enable the BFD protocol for the OSPFv2, BGP4, and IS-IS dynamic routing protocols specifically.	`oamd`	IPv4 and IPv6
BGP4	*Border Gateway Protocol (BGP)* with multi-protocol extension is a dynamic routing protocol for external networks that supports the IPv4 and IPv6 addressing formats.	`bgpd`	IPv4 and IPv6
IS-IS	*Intermediate System-to-Intermediate System (IS-IS)* is a dynamic routing protocol for internal networks, based on a link-state algorithm.	`isisd`	IPv4 and IPv6
OSPFv2	The *Open Shortest Path First (OSPF)* protocol is a dynamic routing protocol for internal networks, based on a link-state algorithm.	`ospfd`	IPv4
OSPFv3	The *OSPFv3* protocol is an enhanced version of OSPFv2.	`ospf6d`	IPv6
RIPv1/RIPv2	*Routing Information Protocol (RIP)* is a dynamic routing protocol for internal networks, based on a distance-vector algorithm (number of hops).	`ripd`	IPv4
RIPng	The *RIPng* protocol is an enhanced version of RIPv2.	`ripngd`	IPv6

About the Bidirectional Forwarding Detection protocol

Bidirectional Forwarding Detection (BFD) is an industry-standard network protocol on the BIG-IP® system that provides a common service to the dynamic routing protocols BGPv4, OSPFv2, and IS-IS. Enabled on a per-route domain basis, BFD identifies changes to the connectivity between two forwarding engines, or endpoints, by transmitting periodic BFD control packets on each path between the two endpoints. When either endpoint fails to receive these control packets for a specific duration of time, the connectivity between the endpoints is considered lost, and BFD notifies the associated dynamic routing protocols. In general, BFD detects connectivity changes more rapidly than the endpoints' standard Hello mechanisms, leading to quicker network convergence, which is highly desirable to data center applications.

BFD operates by establishing a session between two endpoints, sending BFD control packets over the link. If more than one link exists between two endpoints, BFD can establish multiple sessions to monitor each link.

A BFD session can operate in one of two modes, either asynchronous mode or demand mode:

- You configure BFD to operate in *asynchronous mode* when you want both endpoints to verify connectivity by periodically sending Hello packets to each other. This is the most commonly-used mode.
- You configure BFD to operate in *demand mode* when you want the endpoints to use another way to verify connectivity to each other instead of sending Hello packets. For example, the endpoints might verify connectivity at the underlying physical layer. Note, however, that in demand mode, either host can send Hello packets if needed.

Note: *BFD failure detection between two BIG-IP systems does not trigger failover.*

Configuration overview

The first step in configuring the Bidirectional Forwarding Detection (BFD) protocol on the BIG-IP® system is to use the IMI Shell within `tmsh` to configure the protocol for the relevant advanced routing modules (BGP4, OSPFv2, and IS-IS):

- Because BFD does not include a discovery mechanism, you must explicitly configure BFD sessions between endpoints.
- The BFD protocol requires you to commit a nominal amount of additional system resources, in the form of timers, interface bandwidth, and system memory.

After configuring BFD protocol behavior, you enable the protocol on one or more specific route domains.

Important: *You can find detailed documentation on BFD commands in the AskF5™ knowledge base at* `http://support.f5.com`.

Enabling the BFD protocol for a route domain

Before you perform this task, verify that you have configured the **Port Lockdown** setting on all self IP addresses with which routers must communicate. Specifically, you must configure self IP addresses to allow TCP connections on the relevant service port.

You must enable the Bidirectional Forwarding Detection (BFD) network protocol on a per-route domain basis. Use this task to enable BFD on an existing route domain.

1. On the Main tab, click **Network** > **Route Domains**.
   The Route Domain List screen opens.
2. In the Name column, click the name of the relevant route domain.
3. For the **Dynamic Routing Protocols** setting, from the **Available** list, select **BFD** and move it to the **Enabled** list.

   When you enable BFD, the BIG-IP system starts one BFD session for the route domain, and this session supports the BGP4, IS-IS, and OSPFv2 protocols.
4. Click **Update**.
   The system displays the list of route domains on the BIG-IP system.

After you perform this task, the BIG-IP® system starts the daemon `oamd`. Once enabled, the BFD protocol automatically restarts whenever the BIG-IP system is restarted.

Common commands for BFD base configuration

There are two common BFD commands that you can use to perform BFD base configuration. To use these commands, you use the IMI Shell within `tmsh`.

Sample command line sequence	Result
`bigip (config-if)# bfd interval 100 minrx 200 multiplier 4`	Sets desired Min Tx, required Min Rx, and detect Multiplier.
`bigip (config)# bfd slow-timer 2000`	Sets BFD slow timer to two seconds.

Common commands for BFD routing configuration

There are a number of common BFD commands that you can use to perform BFD routing configuration. To use these commands, you use the IMI Shell within `tmsh`.

Protocol	Sample command line sequence	Result
BGP4	`bigip (config-if)# neighbor 1.1.1.1 fallover bfd multihop`	Enables multi-hop bidirectional forwarding detection to BGP neighbor `1.1.1.1`.
OSPFv2	`bigip (config)# bfd all-interfaces`	Enables single-hop bidirectional forwarding detection for all OSPF neighbors.
OSPFv2	`bigip (config)# area 1 virtual-link 3.3.3.3 fallover bfd`	Enables multi-hop bidirectional forwarding detection to OSPF router `3.3.3.3`.
IS-IS	`bigip (config-if)# bfd all-interfaces`	Enables bidirectional forwarding detection for all IS-IS neighbors.

About ECMP routing

Some of the advanced routing modules on the BIG-IP® system include support for Equal Cost Multipath (ECMP) routing. *ECMP* is a forwarding mechanism for routing a traffic flow along multiple paths of equal cost, with the goal of achieving equally-distributed link load sharing. By load balancing traffic over multiple paths, ECMP offers potential increases in bandwidth, as well as some level of fault tolerance when a path on the network becomes unavailable.

Advanced routing modules that support ECMP

The BIG-IP® system deploys Equal Cost Multipath (ECMP) routing with these advanced routing modules:

- BGP4
- IS-IS
- OSPFv2
- OSPFv3
- RIPv1
- RIPv2

The ECMP protocol is enabled by default for all of these advanced routing modules except BGP4. For BGP4, you must explicitly enable the ECMP forwarding mechanism.

Enabling the ECMP protocol for BGP4

You can enable the Equal Cost Multipath (ECMP) forwarding mechanism for the BGP4 advanced routing module, using the Traffic Management Shell (`tmsh`) command line interface. When you enable ECMP for BGP4, the BIG-IP® system provides multiple paths for a traffic flow to choose from, in order to reach the destination.

Important: *For all other advanced routing modules, the ECMP protocol is enabled by default.*

1. Open a console window, or an SSH session using the management port, on a BIG-IP system.
2. Use your user credentials to log in to the system.
3. At the command prompt, type `tmsh`.
   This opens the `tmsh` shell.
4. Type this command: `run /util imish -r ID`.
   The `ID` variable represents the route domain ID.
   This command invokes the IMI shell.
5. Type `enable`.
6. Type `configure terminal`.
7. Type `router bgp as-number`.
8. Type this command: `bgp max-paths (ebgp|ibgp|) 2-64`

After you perform this task, the ECMP forwarding mechanism is enabled for the BGP4 advanced routing module.

Viewing routes that use ECMP

You can perform this task to view the dynamic routes on the system that are using the Equal Cost Multipath (ECMP) forwarding mechanism.

1. Open a console window, or an SSH session using the management port, on a BIG-IP® system.
2. Use your user credentials to log in to the system.
3. At the command prompt, type `tmsh show net route`.

The system displays all dynamic routes and indicates the routes that are using ECMP.

Location of startup configuration for advanced routing modules

When you enable advanced routing modules for a route domain, the BIG-IP system creates a dynamic routing startup configuration. Each route domain has its own dynamic routing configuration, located in the folder `/config/zebos/rdn`, where `n` is the numeric route domain ID.

Warning: *F5 Networks strongly discourages manual modifications to the startup configuration (such as by using a text editor). Doing so might lead to unexpected results.*

Accessing the IMI Shell

Perform this task when you want to use IMI Shell (`imish`) to configure any of the dynamic routing protocols. Note that if you are using the route domains feature, you must specify the route domain pertaining to the dynamic routing protocol that you want to configure.

1. Open a console window, or an SSH session using the management port, on a BIG-IP device.

2. Use your user credentials to log in to the system.
3. At the command prompt, type `tmsh`.
   This opens the `tmsh` shell.
4. Type this command: `run /util imish -r ID`.

   If the route domain for the protocol you want to configure is the default route domain for the current partition, you do not need to use the `-r` option to specify the route domain ID.

   This command invokes the IMI shell.

You can now use any of the IMI shell commands.

Relationship of advanced routing modules and BFD to route domains

For each route domain on the BIG-IP system (including route domain `0`), you can enable one or more dynamic routing protocols, as well as the network protocol Bidirectional Forwarding Detection (BFD). For example, you can enable BGP4 and OSPFv3 on a specific route domain. Use of dynamic routing protocols for a route domain is optional.

When you enable dynamic routing on a specific route domain, the BIG-IP system creates a dynamic routing instance. This dynamic routing instance is made up of the core dynamic routing daemons (`imi` and `nsm`), as well each relevant dynamic routing protocol daemon. If you enable BFD, the BFD instance is made up of the `oamd` protocol daemon. Thus, each dynamic routing instance for a route domain has a separate configuration. You manage a dynamic routing configuration using the IMI shell (`imish`).

Enabling a protocol for a route domain

Before you perform this task, verify that you have configured the **Port Lockdown** setting on all self IP addresses with which routers must communicate. Specifically, you must configure self IP addresses to allow TCP connections on the relevant service port. For example, for BGP4, you must configure self IP addresses to allow TCP connections for port `179`, the well-known port for BGP4.

The first step in configuring dynamic routing protocols on the BIG-IP system is to enable one or more routing protocols, as well as the optional the Bidirectional Forwarding Detection (BFD) network protocol. A protocol is enabled when at least one instance of the protocol is enabled on a route domain.

Important: *The BIG-IP system does not synchronize enabled protocols at runtime during configuration synchronization in a redundant system configuration. This can adversely affect the OSPFv2 and OSPFv3 protocols. To prevent these effects, always enable the protocol on an active device. Then synchronize the configuration to a standby device.*

1. On the Main tab, click **Network** > **Route Domains**.
   The Route Domain List screen opens.
2. In the Name column, click the name of the relevant route domain.
3. For the **Dynamic Routing Protocols** setting, from the **Available** list, select a protocol name and move it to the **Enabled** list.

   You can enable any number of listed protocols for this route domain.

   Important: *When you enable BFD, the BIG-IP system starts one BFD session for the route domain, and this session supports the BGP4, IS-IS, and OSPFv2 protocols only.*

4. Click **Update**.
   The system displays the list of route domains on the BIG-IP system.

After performing this task, the BIG-IP system starts an instance of the specified protocol daemon for the specified route domain, and starts the core daemons `nsm` and `imi`. If BFD is enabled, the system also starts the daemon `oamd`. Once enabled, a protocol automatically restarts whenever the BIG-IP system is restarted.

Disabling a protocol for a route domain

Perform this task to disable an instance of a routing or network protocol that is currently associated with a route domain other than route domain`0`.

Important: *The BIG-IP system does not synchronize disabled protocols at runtime during configuration synchronization in a device service clustering (redundant) configuration. This can adversely affect the OSPFv2 and OSPFv3 protocols. To prevent these effects, always disable the protocol on a standby device. Then synchronize the configuration to an active device.*

---

1. On the Main tab, click **Network** > **Route Domains**.
   The Route Domain List screen opens.
2. In the Name column, click the name of the relevant route domain.
3. For the **Dynamic Routing Protocols** setting, from the **Enabled** list, select a protocol name and move it to the **Available** list.

   You can disable any number of listed protocols for this route domain.
4. Click **Update**.
   The system displays the list of route domains on the BIG-IP system.

After disabling a dynamic routing protocol for a route domain, the BIG-IP system stops the daemon of the specified protocol, resulting in these effects:

- If the specified protocol was the only protocol enabled on the system, the system stops the common daemons `nsm` and `imi`, and possibly the `oamd` daemon. You will no longer see these daemons running on the system.
- The relevant configuration is removed from the runtime configuration, but the configuration is stored on the system until you explicitly save the running configuration.
- If restarted later, the BIG-IP system does not automatically re-enable the protocol. In this case, you must explicitly re-enable the protocol after the system restarts.

Displaying the status of enabled protocols

Perform this task to display the status of instances of any dynamic routing protocols (including the Bidirectional Forwarding Detection (BFD) protocol) that are enabled for a specific route domain.

1. Open a console window, or an SSH session using the management port, on a BIG-IP device.
2. Use your user credentials to log in to the system.
3. At the command prompt, type `tmsh`.
   This opens the `tmsh` shell.
4. Type `run util zebos check`
   This displays the status and process IDs of any enabled dynamic routing protocols or BFD protocol for the specified route domain.

After performing this task, you can see the status and process IDs of any enabled protocols. The following shows sample output:

```
bgpd    is running [22320]
```

About Route Health Injection

Route Health Injection (RHI) is the system process of advertising the availability of virtual addresses to other routers on the network. You can configure two aspects of RHI: route advertisement and route redistribution.

About route advertisement of virtual addresses

Route advertisement is the function that the BIG-IP® system performs when advertising a route for a virtual address to the Traffic Management Microkernel (TMM) routing table. You must configure route advertisement to ensure that the dynamic routing protocols propagate this route to other routers on the network.

When configuring route advertisement for a virtual address, you can specify the particular condition under which you want the BIG-IP system to advertise the address. The available conditions that you can choose from, and their descriptions, are:

When any relevant virtual server is available
If the system has multiple virtual servers for that virtual address and at least one of them is available, the system advertises the route for the virtual address.

When all relevant virtual servers are available
The system only advertises the route for the virtual address when all of the relevant virtual servers are available.

Always
The system can advertise the route even when all relevant virtual servers are unavailable. For example, the system can advertise the route when the virtual server is disabled but the virtual address is enabled and the assigned pool is available.

After you specify the desired behavior of the system with respect to route advertisement, the `tmrouted` daemon attempts to comply. The daemon only succeeds in advertising the route for the virtual address when the relevant virtual servers, pool, and pool members collectively report their status in specific combinations.

Note: *When you configure RHI in a device group configuration, only devices with active traffic groups attempt to advertise routes to virtual addresses.*

Determination of UP state for a virtual address

The `tmrouted` daemon within the BIG-IP® system considers a virtual IP address to be in an UP state when any one of the following conditions are true:

- The BIG-IP Configuration utility shows blue, green, or yellow status for the virtual address.
- The virtual address is a member of an active traffic group.
- The virtual address is enabled and is currently being advertised.

Conditions for route advertisement of virtual addresses

This table shows the ways that Local Traffic Manager™ (LTM®) object status affects whether the BIG-IP® system advertises a route to a virtual address. In the table, the colors represent object status shown on the Local Traffic screens within the BIG-IP Configuration utility. The table also summarizes the collective LTM object status that determines route advertisement.

Table 3: Route advertisement for virtual addresses based on LTM object status

	LTM object status				
Route advertised?	**Pool member**	**Pool**	**Virtual server**	**Virtual address**	**Status summary**
Yes					Pool members are monitored and `UP`. The virtual address is `UP`.
Yes					Pool or pool members are unmonitored. The virtual address is enabled.
Yes					Pool members are disabled. Other objects are enabled.
Yes					Virtual server is disabled. Virtual address is enabled.
Yes	N/A				The pool has no members. The virtual address is enabled.
Yes	N/A	N/A			Virtual server has no pool assigned.
No					Pool members are monitored and `DOWN`.
No					Virtual server and virtual address are disabled.
No					Virtual address is disabled. Other objects are enabled.

LTM object status indicators

The BIG-IP® Configuration utility displays various colored icons to report the status of virtual servers, virtual addresses, pools, and pool members.

Green circle 

The object is available in some capacity. The BIG-IP system services traffic destined for this object.

Blue square 

The availability of the object is unknown. Sample causes of this status are when the object is not configured for service checking, the IP address of the object is misconfigured, or the object is disconnected fromthe network.

Yellow triangle 

The object is not currently available but might become available later with no user intervention. For example, an object that has reached its configured connection limit might show yellow status but later switch to green when the number of connections falls below the configured limit.

Red diamond
The object is not available. The BIG-IP system cannot service traffic destined for this object. A sample cause of this status is when a node fails service checking because it has become unavailable. This status requires user intervention to restore the object status to green.

Black circle
A user has actively disabled an available object.

Black diamond
A user has actively disabled an unavailable object.

Gray icons
A parent object disabled the object, or the object is enabled but unavailable because of another disabled object.

Configuring route advertisement on virtual addresses

Before performing this task, verify that you have created the relevant virtual server on the BIG-IP system. Also, the virtual address that you want to advertise must have a status of Up, Unavailable, or Unknown.

Perform this task to specify the criterion that the BIG-IP system uses to advertise routes for virtual addresses. You must perform this task if you want the dynamic routing protocols to propagate this route to other routers on the network.

1. On the Main tab, click **Local Traffic** > **Virtual Servers**.
   The Virtual Server List screen displays a list of existing virtual servers.
2. On the menu bar, click **Virtual Address List**.
3. Click the name of the virtual server you want to configure.
4. For the **Advertise Route** setting, select an option:
   - **When any virtual server is available**
   - **When all virtual server(s) are available**
   - **Always**
5. Click **Update**.

After you perform this task, properly-configured dynamic routing protocols can redistribute the advertised route to other routers on the network.

Displaying advertised routes for virtual addresses

Before you perform this task, depending on the dynamic routing protocol, you might need to configure the protocol's router definition to redistribute the kernel.

Perform this task when you want to display routes for virtual addresses that the BIG-IP system has advertised to other routers on the network.

1. Open a console window, or an SSH session using the management port, on a BIG-IP device.
2. Use your user credentials to log in to the system.
3. At the command prompt, type `tmsh`.
   This opens the `tmsh` shell.
4. Type `run /util imish -r` *`ID`*
   The variable *`ID`* is the ID of the relevant route domain. This ID must be an integer.

This opens the IMI shell.

5. At the prompt, type `show ip route kernel`.

After performing this task, you should see the advertised routes for virtual addresses. For example, advertised routes for virtual addresses **10.1.51.80/32** and **10.2.51.81/32** appear as follows:

```
K       10.1.51.80/32 is directly connected, tmm0
K       10.1.51.81/32 is directly connected, tmm0
```

The `/32` netmask indicates that the IP addresses pertain to individual hosts, and the `tmm0` indicator shows that protocols on other routers have learned these routes from the Traffic Management Microkernel (TMM).

Delaying the withdrawal of RHI routes

Perform this task to delay the withdrawal of RHI routes when operation status changes. Delaying route withdrawal prevents short route flaps that might occur due to both the short period during failover when both devices are in a standby state, and the periodic housekeeping processes in routing protocol daemons (specifically `bgpd`).

1. Open a console window, or an SSH session using the management port, on a BIG-IP device.
2. Use your user credentials to log in to the system.
3. At the command prompt, type `tmsh`.
   This opens the `tmsh` shell.
4. Set the `bigdb` variable to the needed delay by typing this command: `modify /sys db tmrouted.rhifailoverdelay value` *`delay_in_seconds`*

The BIG-IP system now delays the withdrawal of RHI routes by the number of seconds that you specified.

Redistribution of routes for BIG-IP virtual addresses

You can explicitly configure each dynamic routing protocol to redistribute routes for advertised virtual addresses, to ensure that other routers on the network learn these routes. For purposes of redistribution, the dynamic routing protocols consider any route generated through Route Health Injection (RHI) to be a host route.

Note: *For all dynamic routing protocols, you must configure route redistribution for IPv4 addresses separately from that of IPv6 addresses.*

This example shows an entry in the OSPF configuration. When you add this statement to the OSPF configuration, the BIG-IP system redistributes the route for the virtual address.

```
router ospf
   redistribute kernel
```

You can optionally specify a `route-map` reference that specifies the route map to use for filtering routes prior to redistribution. For example:

```
redistribute kernel route-map external-out
```

Route maps provide an extremely flexible mechanism for fine-tuning redistribution of routes using the dynamic routing protocols.

Advertisement of next-hop addresses

The BIG-IP system advertises all self IP addresses, including floating self IP addresses, to the dynamic routing protocols. The protocols store floating addresses so that the protocols can prefer a floating address as the advertised next hop. This applies only to protocols that allow explicit next-hop advertisement.

IPv6 next-hop address selection (BGP4 only)

When you are using BGP4 and IPv6 addressing, you can advertise one or two next-hop addresses for each route. The BIG-IP system selects the addresses to advertise based on several factors.

Parameter combinations for next-hop address selection

For BGP-4 only, you can choose from several combinations of configuration parameters to control the selection of next-hop IPv6 addresses.

Table 4: P = Peering, X = Configured

Link-local autoconf. (LL-A)	Link-local (LL)	Link-local floating (LL-F)	Global (G)	Global floating (G-F)	EBGP multihop	Advertised nexthop addresses
P						LL-A
X	P					LL
X	P	X				LL-F
X	P		X			G, LL
X			P			G, LL-A
X	X	X	P		X	G
X	X	X	P			LL-F
X			P	X		G-F
X	P		X	X		GF, LL
X	X		P	X		GF
X	P	X	X	X		LL-F
X	X	X	P	X		GF-F

Visibility of static routes

The dynamic routing protocols view Traffic Management Microkernel (TMM) static routes as kernel routes. (*TMM static routes* are routes that you configure using `tmsh` or the BIG-IP Configuration utility.) Because TMM static routes are viewed as kernel routes, a TMM static route has a higher precedence than a dynamic route (with an identical destination).

Management routes and addresses are not visible to the dynamic routing protocols and cannot be advertised. Routes to the networks reachable through the management interface can be learned by dynamic routing protocols if they are reachable through a VLAN, VLAN group, or tunnel.

About dynamic routing for redundant system configurations

If the BIG-IP system that you are configuring for dynamic routing is part of a redundant system configuration, you should consider these factors:

- You must configure the dynamic routing protocols on each member of the device group.
- For protocols that include the router ID attribute, you should verify that each member of the device group has a unique router ID.
- When you configure Route Health Injection (RHI), only active device group members advertise routes to virtual addresses.

Special considerations for BGP4, RIP, and IS-IS

For the BGP, RIP, RIPng, and IS-IS protocols, you no longer need to specifically configure these protocols to function in active-standby configurations. Each member of the device group automatically advertises the first floating self IP address of the same IP subnet as the next hop for all advertised routes. This applies to both IPv4 and IPv6 addresses.

Advertising a next-hop address that is always serviced by an active device guarantees that all traffic that follows routes advertised by any device in the redundant pair is forwarded based on the active LTM® configuration.

Special considerations for OSPF

For OSPF protocols, the BIG-IP system ensures that standby device group members are the least preferred next-hop routers. The system does this by automatically changing the runtime state as follows:

Table 5: Dynamic routing protocols

Protocol Name	Runtime state change
OSPFv2	The OSPF interface cost is increased on all interfaces to the maximum value (65535) when the status of the device is Standby. Also, all external type 2 Link State Advertisements (LSAs) are aged out.
OSPFv3	The OSPF interface cost is increased on all interfaces to the maximum value.

Displaying OSPF interface status

When you display OSPF interface status, you can see the effect of runtime state changes.

1. Open a console window, or an SSH session using the management port, on a BIG-IP device.
2. Use your user credentials to log in to the system.

3. At the command prompt, type `tmsh`.
4. Type this command: `run /util imish -r ID`.
5. Type `sh ip ospf interface`.
   The variable *id* is the ID of the relevant route domain.

Listing the OSPF link state database

When you list the contents of the OSPF link state database, you can see the effect of runtime state changes.

1. Open a console window, or an SSH session using the management port, on a BIG-IP device.
2. Use your user credentials to log in to the system.
3. At the command prompt, type `tmsh`.
4. Type this command: `run /util imish -r ID`
5. Type `sh ip ospf database external self-originate`.
   The variable *id* is the ID of the relevant route domain.

Dynamic routing on a VIPRION system

If you have a VIPRION® system, it is helpful to understand how the cluster environment affects the dynamic routing functionality.

VIPRION appearance as a single router

On a VIPRION® system, the dynamic routing system behaves as if the cluster were a single router. This means that a cluster always appears as a single router to any peer routers, regardless of the dynamic routing protocol being used.

From a management perspective, the VIPRION system is designed to appear as if you are configuring and managing the routing configuration on a single appliance. When you use the cluster IP address to configure the dynamic routing protocols, you transparently configure the primary blade in the cluster. The cluster synchronization process ensures that those configuration changes are automatically propagated to the other blades in the cluster.

Redundancy for the dynamic routing control plane

The dynamic routing system takes advantage of the redundancy provided by the cluster environment of a VIPRION® chassis, for the purpose of providing redundancy for the dynamic routing control plane. Two key aspects of dynamic routing control plane redundancy are the VIPRION cluster's appearance to the routing modules as a single router, and the operational modes of the enabled dynamic routing protocols.

Operational modes for primary and secondary blades

Enabled dynamic routing protocols run on every blade in a cluster in one of these operational modes: MASTER, STANDBY, or SLAVE.

This table shows the operational modes for primary and secondary blades, on both the active cluster and the standby cluster.

Table 6: Operational modes for dynamic routing protocols per blade type

Blade Type	Active Cluster	Standby Cluster	Notes
Primary	MASTER mode	STANDBY mode	The dynamic routing protocols: • Actively participate in dynamic routing protocol communication with peer routers. • Maintain TMM and host route tables on all blades in the cluster.
Secondary	SLAVE mode	SLAVE mode	The dynamic routing protocols: • Do not transmit any dynamic routing protocol traffic. • Track communication between a module and the peer routers, or wait for transition to MASTER or STANDBY mode.

In MASTER and STANDBY modes, all routes learned by way of dynamic routing protocols on the primary blade are (in real-time) propagated to all secondary blades. The difference between MASTER and STANDBY mode is in the parameters of advertised routes, with the goal to always make the active unit the preferred next hop for all advertised routes.

The transition from SLAVE to MASTER or STANDBY mode takes advantage of standard dynamic routing protocol graceful restart functionality.

Viewing the current operational mode

Perform this task to display the current operational mode (MASTER, STANDBY, or SLAVE) of a blade.

1. Open a console window, or an SSH session using the management port, on a BIG-IP device.
2. Use your user credentials to log in to the system.
3. At the command prompt, type `tmsh`.
4. Type `run /util imish -r` *ID*.

   If the route domain for the protocol you want to configure is the default route domain for the current partition, you do not need to use the `-r` option to specify the route domain ID.

   This command invokes the IMI shell.
5. Type `show state`.

The BIG-IP system displays a message such as `Current operational state: MASTER`.

About graceful restart on the VIPRION system

With the *graceful restart* function, the dynamic routing protocol control plane moves from one blade to another without disruption to traffic. Graceful restart is enabled for most supported protocols and address families by default.

To operate successfully, the graceful restart function must be supported and enabled on all peer routers with which the VIPRION® system exchanges routing information. If one or more peer routers does not support graceful restart for one or more enabled dynamic routing protocols, a change in the primary blade causes full dynamic routing reconvergence, and probably traffic disruption. The traffic disruption is caused primarily by peer routers discarding routes advertised by the VIPRION system.

The BIG-IP system always preserves complete forwarding information (TMM and host route tables) on VIPRION systems during primary blade changes, regardless of support for graceful restart on peer routers.

Runtime monitoring of individual blades

The BIG-IP system automatically copies the startup configuration to all secondary blades and loads the new configuration when the running configuration is saved on the primary blade.

You can display information about the runtime state of both the primary and secondary blades. However, some information displayed on secondary blades might differ from the information on the primary blade. For troubleshooting, you should use the information displayed on the primary blade only, because only the primary blade both actively participates in dynamic routing communication and controls route tables on all blades.

Troubleshooting information for dynamic routing

Dynamic route propagation depends on a BIG-IP® system daemon named `tmrouted`. The BIG-IP system starts the `tmrouted` daemon when you enable the first dynamic routing protocol. and restarts the daemon whenever the BIG-IP system restarts.

In the rare case when you need to manage the `tmrouted` daemon due to a system issue, you can perform a number of different tasks to troubleshoot and solve the problem.

Checking the status of the tmrouted daemon

Use this procedure to verify that the `tmrouted` daemon is running. This daemon must be running for the enabled dynamic routing protocols to propagate routes.

1. Open a console window, or an SSH session using the management port, on a BIG-IP device.
2. Type your user credentials to log in to the system.
3. If the system has granted you access to the BASH shell prompt, type `tmsh`. Otherwise, skip this step.
4. Type `show /sys service tmrouted`.

The BIG-IP system displays information about `tmrouted` such as: `tmrouted run (pid 5113) 1 days`

Stopping the tmrouted daemon

Before you can stop an instance of the `tmrouted` daemon, the associated protocol instance must be enabled on the BIG-IP system. Also, the BIG-IP system `mcpd` and `tmm` daemons must be running on the system.

You perform this task to stop an instance of the `tmrouted` daemon.

Important: *Manage the `tmrouted` daemon using the `tmsh` utility only. Attempting to manage `tmrouted` using a Linux command or with invalid parameters might cause the daemon to fail.*

1. Open a console window, or an SSH session using the management port, on the BIG-IP device.
2. Type your user credentials to log in to the system.
3. If the system has granted you access to the BASH shell prompt, type `tmsh`. Otherwise, skip this step.
4. At the `tmsh` shell prompt, type `stop /sys service tmrouted`.

This command stops any instances of `tmrouted` that are running on the system, causing the associated protocol instance to stop propagating routes.

Restarting the tmrouted daemon

Before restarting an instance of the `tmrouted` daemon, verify that the associated protocol instance is enabled on the BIG-IP system. Also, verify that the BIG-IP system `mcpd` and `tmm` daemons are running on the system.

You perform this task to restart an instance of the `tmrouted` daemon. Whenever the BIG-IP system reboots for any reason, the BIG-IP system automatically starts an instance of `tmrouted` for each instance of an enabled dynamic routing protocol.

Important: *Manage the `tmrouted` daemon using the `tmsh` utility only. Attempting to manage `tmrouted` using a Linux command or with invalid parameters might cause the daemon to fail.*

1. Open a console window, or an SSH session using the management port, on the BIG-IP system.
2. Type your user credentials to log in to the system.
3. If the system has granted you access to the BASH shell prompt, type `tmsh`. Otherwise, skip this step.
4. At the `tmsh` shell prompt, type `restart /sys service tmrouted`.

This command restarts any instances of `tmrouted` that are currently stopped. The daemon also communicates with the `nsm` daemon to propagate dynamically-learned routes to other BIG-IP system processes that need to direct application traffic.

Configuring tmrouted recovery actions

Use this task to configure recovery actions when the `tmrouted` daemon restarts.

1. Open a console window, or an SSH session using the management port, on a BIG-IP device.
2. Use your user credentials to log in to the system.
3. At the command prompt, type `tmsh`.
4. Type `modify /sys daemon-ha tmrouted running [enabled|disabled]`

- If you want to enable the `running-timeout` and `non-running-action` options, type `enabled`.
- If you want to disable the `running-timeout` and `non-running-action` options, type `disabled`.

Typing this command with the `enabled` option causes the active BIG-IP device to fail over to another device in the device group whenever the `tmrouted` daemon restarts.

5. Type `modify /sys daemon-ha tmrouted heartbeat [enabled|disabled]`

   - If you want to enable monitoring for the `tmrouted` heartbeat, type `enabled`.
   - If you want to disable monitoring for the `tmrouted` heartbeat, type `disabled`.

   When you type this command with the `enabled` option and the `tmrouted` heartbeat is subsequently lost, the system behaves according to the action specified by the `heartbeat-action` option.

Location and content of log files

For each dynamic routing protocol, the BIG-IP system logs messages to a file that pertains to the route domain in which the protocol is running. An example of the path name to a dynamic routing log file is `/var/log/zebos/rd1/zebos.log` file, where `rd1` is the route domain of the protocol instance.

The system logs additional messages to the files `/var/log/daemon.log` and `/var/log/ltm`. The system logs protocol daemon information for protocol-specific issues, and logs `nsm` and `imi` daemon information for core daemon-related issues.

If a core dynamic routing daemon exits, the system logs an error message similar to the following to the `/var/log/daemon.log` file:

```
Mar  5 22:43:01 mybigip LOGIN: Re-starting tmrouted
```

In addition, the BIG-IP system logs error messages similar to the following to the `/var/log/ltm` file:

```
mcpd[5157]: 01070410:5: Removed subscription with subscriber id bgpd
mcpd[5157]: 01070533:3: evWrite finished with no byte sent to connection 0xa56f9d0 (user
Unknown) - connection deleted
```

Creating a debug log file

Perform this task to create a log file for debugging. With a debug log file, you can more effectively troubleshoot any issues with a dynamic routing protocol.

1. Open a console window, or an SSH session using the management port, on a BIG-IP device.
2. Use your user credentials to log in to the system.
3. At the command prompt, type `tmsh`.
   This opens the `tmsh` shell.
4. At the `tmsh` prompt, type this command: `run /util imish -r ID`.

   If the route domain for the protocol you want to configure is the default route domain for the current partition, you do not need to specify the route domain ID.

   This command invokes the IMI shell.
5. Type the command `log file /var/log/zebos/rdn/zebos.log`.

   The variable *n* represents the relevant route domain ID. This ID must be an integer.

   The system creates a debug log file.
6. Type `write`.

This action saves the log file.

Chapter 11

Address Resolution Protocol

Address Resolution Protocol on the BIG-IP system

The BIG-IP® system is a multi-layer network device, and as such, needs to perform routing functions. To do this, the BIG-IP system must be able to find destination MAC addresses on the network, based on known IP addresses. The way that the BIG-IP system does this is by supporting *Address Resolution Protocol (ARP)*, an industry-standard Layer 3 protocol.

What are the states of ARP entries?

When you use the BIG-IP Configuration utility to view the entries in the ARP cache, you can view the state of each entry:

RESOLVED
Indicates that the system has successfully received an ARP response (a MAC address) for the requested IP address within two seconds of initiating the request. An entry in a RESOLVED state remains in the ARP cache until the timeout period has expired.

INCOMPLETE
Indicates that the system has made one or more ARP requests within the maximum number of requests allowed, but has not yet received a response.

DOWN
Indicates that the system has made the maximum number of requests allowed, and still receives no response. In this case, the system discards the packet, and sends an ICMP host unreachable message to the sender. An entry with a DOWN state remains in the ARP cache until the first of these events occurs:

- Twenty seconds elapse.

- The BIG-IP system receives either a resolution response or a gratuitous ARP from the destination host. (A *gratuitous ARP* is an ARP message that a host sends without having been prompted by an ARP request.)
- You explicitly delete the entry from the ARP cache.

About BIG-IP responses to ARP requests from firewall devices

The system does not respond to ARP requests sent from any firewall that uses a multicast IP address as its source address.

About gratuitous ARP messages

When dynamically updating the ARP cache, the BIG-IP system includes not only entries resulting from responses to ARP requests, but also entries resulting from gratuitous ARP messages.

For security reasons, the system does not fully trust gratuitous ARP entries. Consequently, if there is no existing entry in the cache for the IP address/MAC pair, and the BIG-IP system cannot verify the validity of the gratuitous ARP entry within a short period of time, the BIG-IP system deletes the entry.

Management of static ARP entries

You can manage static entries in the ARP cache in various ways.

Task summary

Adding a static ARP entry
Viewing static ARP entries
Deleting static ARP entries

Adding a static ARP entry

Perform this task to add entries to the ARP cache on the BIG-IP system. Adding a static entry for a destination server to the ARP cache saves the BIG-IP system from having to send an ARP broadcast request for that destination server. This can be useful when you want the system to forward packets to a special MAC address, such as a shared MAC address, or you want to ensure that the MAC address never changes for a given IP address.

1. On the Main tab, click **Network** > **ARP** > **Static List**.
2. Click **Create**.
3. In the **Name** field, type a name for the ARP entry.
4. In the **IP Address** field, type the IP address with which you want to associate a MAC address.
5. In the **MAC Address** field, type the MAC address that you want to associate with the specified IP address.
6. Click **Finished**.

When the BIG-IP system must forward packets to the specified IP address, the system checks the ARP cache to find the MAC address. The system then checks the VLAN's Layer 2 forwarding table to determine the appropriate outgoing interface.

Viewing static ARP entries

Perform this task to view static entries in the ARP cache.

1. On the Main tab, click **Network** > **ARP** > **Static List**.
2. View the list of static ARP entries.

You can now see all static entries in the ARP cache.

Deleting static ARP entries

Perform this task to delete a static entry from the ARP cache.

1. On the Main tab, click **Network** > **ARP** > **Static List**.
2. Locate the entry you want to delete, and to the left of the entry, select the check box.
3. Click **Delete**.
   A confirmation message appears.
4. Click **Delete**.

The deleted entry is no longer in the BIG-IP system ARP cache.

Management of dynamic ARP entries

You can manage dynamic entries in the ARP cache in various ways.

Task summary

Viewing dynamic ARP entries
Deleting dynamic ARP entries
Configuring global options for dynamic ARP entries

Viewing dynamic ARP entries

Perform this task to view dynamic entries in the ARP cache.

1. On the Main tab, click **Network** > **ARP** > **Dynamic List**.
2. View the list of dynamic ARP entries.

You can now see the list of dynamic ARP entries.

Deleting dynamic ARP entries

Perform this task to delete a dynamic entry from the ARP cache.

1. On the Main tab, click **Network** > **ARP** > **Dynamic List**.
2. Locate the entry you want to delete and, to the left of the entry, select the check box.
3. Click **Delete**.
   A confirmation message appears.
4. Click **Delete**.

The deleted entry is no longer in the BIG-IP system ARP cache.

Configuring global options for dynamic ARP entries

Perform this task to apply global options to all dynamic ARP entries.

1. On the Main tab, click **Network** > **ARP** > **Options**.
2. In the **Dynamic Timeout** field, specify a value, in seconds.

   The seconds begin to count down toward `0` for any dynamically-added entry. When the value reaches `0`, the BIG-IP system automatically deletes the entry from the cache. If the entry is actively being used as the time approaches `0`, ARP attempts to refresh the entry by sending an ARP request.
3. In the **Maximum Dynamic Entries** field, specify a maximum number of entries.

   Configure a value large enough to maintain entries for all directly-connected hosts with which the BIG-IP system must communicate. If you have more than 2000 hosts that are directly connected to the BIG-IP system, you should specify a value that exceeds the default value of `2048`.

   If the number of dynamic entries in the cache reaches the limit that you specified, you can still add static entries to the cache. This is possible because the system can remove an older dynamic entry prematurely to make space for a new static entry that you add.
4. In the **Request Retries** field, specify the number of times that the system can resend an ARP request before marking the host as unreachable.
5. For the **Reciprocal Update** setting, select or clear the check box to enable or disable the setting.

Option	Description
Enabled	Creates an entry in the ARP cache whenever the system receives who-has packets from another host on the network. When you enable this option, you slightly enhance system performance by eliminating the need for the BIG-IP system to perform an additional ARP exchange later.
Disabled	Prevents a malicious action known as ARP poisoning. *ARP poisoning* occurs when a host is intentionally altered to send an ARP response containing a false MAC address.

6. Click **Update**.

The BIG-IP system now applies these values to all dynamic ARP entries.

Global options for dynamic ARP cache entries

You can configure a set of global options for controlling dynamic ARP cache entries.

Table 7: Global options for dynamic ARP entries

Option	Description
Dynamic Timeout	Specifies the maximum number of seconds that a dynamic entry can remain in the ARP cache before the BIG-IP system automatically removes it.
Maximum Dynamic Entries	Limits the number of dynamic entries that the BIG-IP system can hold in the ARP cache at any given time. This setting has no effect on the number of static entries that the ARP cache can hold.
Request Retries	Specifies the number of times that the BIG-IP system resends an ARP request before finally marking the host as unreachable.
Reciprocal Update	Enables the BIG-IP system to store additional information, which is information that the system learns as a result of other hosts on the network sending ARP broadcast requests to the BIG-IP system.

Chapter 12

Spanning Tree Protocol

Introduction to spanning tree protocols

On networks that contain redundant paths between Layer 2 devices, a common problem is bridging loops. Bridging loops occur because Layer 2 devices do not create boundaries for broadcasts or packet floods. Consequently, Layer 2 devices can use redundant paths to forward the same frames to each other continuously, eventually causing the network to fail.

To solve this problem, the BIG-IP® system supports a set of industry-standard, Layer 2 protocols known as spanning tree protocols. *Spanning tree protocols* block redundant paths on a network, thus preventing bridging loops. If a blocked, redundant path is needed later because another path has failed, the spanning tree protocols clear the path again for traffic. The spanning tree protocols that the BIG-IP system supports are Spanning Tree Protocol (STP), Rapid Spanning Tree Protocol (RSTP), and Multiple Spanning Tree Protocol (MSTP).

Central to the way that spanning tree protocols operate is the use of bridge protocol data units (BPDUs). When you enable spanning tree protocols on Layer 2 devices on a network, the devices send BPDUs to each other, for the purpose of learning the redundant paths and updating their L2 forwarding tables accordingly, electing a root bridge, building a spanning tree, and notifying each other about changes in interface status.

Note: *The term bridge refers to a Layer 2 device such as a switch, bridge, or hub.*

About STP protocol

STP is the original spanning tree protocol, designed to block redundant paths as a way to prevent bridging loops. The STP algorithm creates one, and only one, spanning tree for the entire network. A *spanning tree* is a logical tree-like depiction of the bridges on a network and the paths that connect them.

Because STP is unable to recognize VLANs and usually exhibits poor performance overall, STP is not the preferred spanning tree protocol to use in VLAN-rich environments. However, all participating interfaces in the spanning tree must use the same spanning tree protocol at any given time. Thus, when you have legacy bridges in your environment that are running STP, interfaces on the BIG-IP® system must have the ability to automatically degrade to STP.

Because STP has no knowledge of VLANs, you can have only one spanning tree instance on the BIG-IP system when using STP.

About the RSTP protocol

RSTP is an enhancement to STP, and was designed specifically to improve spanning tree performance. Like STP, RSTP can create only one spanning tree (instance 0), and therefore cannot take VLANs into account when managing redundant paths. However, RSTP's performance improvements generally make it preferable to STP in non-VLAN environments.

In the case where legacy RSTP bridges are on the network, BIG-IP® system interfaces running MSTP can degrade to RSTP, just as they can degrade to STP.

Like STP, RSTP allows only one spanning tree instance on the BIG-IP system.

About the MSTP protocol

MSTP is an enhancement to RSTP and is the preferred spanning tree protocol for the BIG-IP® system. MSTP is specifically designed to understand VLANs and VLAN tagging (specified in IEEE 802.1q). Unlike STP and RSTP, which allow only one spanning tree instance per system, MSTP allows multiple spanning tree instances. Each instance corresponds to a spanning tree, and can control one or more VLANs that you specify when you create the instance. Thus, for any BIG-IP system interface that you assigned to multiple VLANs, MSTP can block a path on one VLAN, while still keeping a path in another VLAN open for traffic. Neither STP nor RSTP has this capability.

A unique feature of MSTP is the concept of spanning tree regions. A *spanning tree region* is a logical set of bridges on the network that share the same values for certain MSTP configuration settings. These configuration settings are: The MSTP configuration name, the MSTP configuration number, the instance numbers, and the VLAN members of each instance. When the values of these settings are identical on two or more bridges, the spanning tree algorithm considers these bridges to constitute an MSTP region. An MSTP region indicates to the spanning tree algorithm that it can use MSTP for all bridges in that region, and thus take VLANs into account when blocking and unblocking redundant paths.

You do not explicitly create a region. The spanning tree algorithm automatically groups bridges into regions, based on the values you assign to the MSTP configuration name, revision number, instance numbers, and instance members.

MSTP can only operate on bridges that are within a region. However, if the BIG-IP system connects to a bridge in a different MSTP region or outside of an MSTP region, the system still participates in spanning tree. In this case, the system is part of the spanning tree instance 0, also known as the Common and Internal Spanning Tree (CIST).

Note: *BIG-IP systems released prior to version 9.0 do not support MSTP.*

About spanning tree with legacy bridges

A key concept about spanning tree protocols on the BIG-IP® system is the concept of protocol degradation. *Protocol degradation* occurs when the spanning tree mode on the BIG-IP system is set to MSTP or RSTP, but the system detects legacy bridges (that is, bridges running an older protocol type) on the network. In this case, the BIG-IP system automatically degrades the spanning tree protocol that is running on each applicable interface to match the protocol running on the legacy device.

For example, suppose you set the BIG-IP system to run in MSTP mode. Later, if a bridge running STP is added to the network, the BIG-IP system will detect the legacy device and automatically degrade the protocol running on the BIG-IP system interfaces from MSTP to STP. The mode is still set to MSTP, but the interfaces actually run STP.

If the legacy device is later removed from the network, you can choose, for each BIG-IP system interface, to manually reset the spanning tree protocol back to MSTP.

The basic principle of protocol degradation is that each BIG-IP system interface in a spanning tree runs the oldest protocol that the system detects on the Layer 2 devices of the network. Thus, if a legacy bridge running STP is added to the network, BIG-IP system interfaces running MSTP or RSTP degrade to STP. Similarly, if a legacy bridge is running RSTP (and no bridges are running STP), interfaces running MSTP degrade to RSTP.

Note that when a bridge running MSTP must degrade to RSTP, the spanning tree algorithm automatically puts the degraded bridge into a separate MSTP region.

Configuration overview

Regardless of which spanning tree protocol you choose to use, the BIG-IP® Configuration utility offers a complete set of default configuration settings. Except for choosing a preferred spanning tree protocol to use, there are very few configuration settings that you need to modify to use the spanning tree feature effectively.

Note: *An alternate way to configure spanning tree protocols is to use tmsh.*

When you configure spanning tree on a BIG-IP system, you must first decide which protocol, or mode, you want to enable. Because MSTP recognizes VLANs, using MSTP is preferable for the BIG-IP system. However, all bridges in a network environment that want to use spanning tree must run the same spanning tree protocol. If a legacy bridge running RSTP or STP is added to the network, the BIG-IP system must switch to that same protocol.

Fortunately, you do not need to continually reconfigure the BIG-IP system spanning tree mode whenever a legacy bridge is added to the network. Instead, a BIG-IP system interface can detect the addition of a legacy bridge and automatically fall back to either RSTP or STP mode. If the legacy bridge is later removed from the network, you can use the BIG-IP Configuration utility to manually reset the interface back to running MSTP.

Once you have enabled a spanning tree mode, you can configure a set of global options. These options are the same options that are defined in the IEEE standards for the spanning tree protocols. While you can use the default settings in most cases, a few settings require user input.

Spanning tree mode

The Mode option specifies the particular spanning tree protocol that you want to use on the BIG-IP® system. The default value is Pass Through. The possible values are:

Disabled
Specifies that when the BIG-IP system receives spanning tree frames (BPDUs), it discards the frames.

Pass Through
Specifies that when the BIG-IP system receives spanning tree frames (BPDUs), it forwards them to all other interfaces. This is the default setting. When you use Pass Through mode, the BIG-IP system is transparent to spanning tree BPDUs. When set to Pass Through mode, the BIG-IP system is not part of any spanning tree. Note that Pass Through mode is not part of the IEEE spanning tree protocol specifications.

STP
Specifies that the BIG-IP system handles spanning tree frames (BPDUs) in accordance with the STP protocol. This mode allows for legacy systems on the network.

RSTP
Specifies that the BIG-IP system handles spanning tree frames (BPDUs) in accordance with the RSTP protocol.

MSTP
Specifies that the BIG-IP system handles spanning tree frames (BPDUs) in accordance with the MSTP protocol.

When you set the mode to MSTP or RSTP, and a legacy bridge running STP is subsequently added to the spanning tree, the applicable BIG-IP system interface automatically changes to running STP. However, you can manually reset an interface to resume operation in RSTP or MSTP mode if the legacy bridge is later removed from the spanning tree.

Global timers

All three spanning tree protocols, have the same three global timer values that you can specify: Hello Time, Maximum Age, and Forward Delay.

About the hello time option

When you change the value of the Hello Time option, you change the time interval, in seconds, that the BIG-IP® system transmits spanning tree information (through BPDUs) to adjacent bridges in the network. The default value for this option is 2.

Warning: *Although valid values are in the range of 1 to 10 seconds, F5 Networks® highly recommends that you use the default value (2 seconds). This value is optimal for almost all configurations.*

Note that when running RSTP, you must maintain the following relationship between the Maximum Age and Hello Time options:

```
Maximum Age >= 2 * (Hello Time + 1)
```

About the maximum age option

When you change the value of the Maximum Age option, you change the amount of time, in seconds, that spanning tree information received from other bridges is considered valid. The default value is 20, and the valid range is 6 to 40.

Note that when running RSTP, you must maintain the following relationships between the Maximum Age and the Hello Time and Forward Delay options:

```
Maximum Age >= 2 * (Hello Time + 1)
```

```
Maximum Age <= 2 * (Forward Delay - 1)
```

About the forward delay option

Primarily used for STP, the Forward Delay option specifies the amount of time, in seconds, that the system blocks an interface from forwarding network traffic when the spanning tree algorithm reconfigures a spanning tree. The default value is 15, and the valid range is 4 to 30.

This option has no effect on the BIG-IP® system when running in RSTP or MSTP mode, provided that all bridges in the spanning tree use the RSTP or MSTP protocol. However, if the addition of legacy STP bridges causes neighboring bridges to fall back to running the STP protocol, then the spanning tree algorithm uses the Forward Delay option when reconfiguring the spanning tree.

Note that when running RSTP, you must maintain the following relationship between the Forward Delay and Maximum Age options:

```
Maximum Age <= 2 * (Forward Delay - 1)
```

About the transmit hold count option

When you change the value of the Transmit Hold Count option, you change the maximum number of spanning tree frames (BPDUs) that the system can transmit on a port within the Hello Time interval. This setting ensures that the spanning tree frames do not overload the network, even in unstable network conditions. The default value is 6, and the valid range is 1 to 10.

MSTP-specific global properties

If you are running MSTP, you can configure three additional global properties:

MSTP configuration name
Applicable to MSTP only, the MSTP Configuration Name setting represents a global name that you assign to all bridges in a spanning tree region. A spanning tree region is a group of bridges with identical MSTP configuration names and MSTP configuration revision levels, as well as identical assignment of

VLANs to spanning tree instances. All bridges in the same region must have this same configuration name. The name must contain from 1 to 32 characters. This option only appears on the screen when you set the Mode property to MSTP.

MSTP configuration revision

Applicable to MSTP only, the MSTP Configuration Revision setting represents a global revision number that you assign to all bridges in a spanning tree region. All bridges in the same region must have this same configuration revision number. The default value is 0. You can type any value between 0 and 65535. This option only appears on the screen when you set the Mode property to MSTP.

Maximum hop number

Applicable to MSTP only, this global property specifies the maximum number of hops that a spanning tree frame (BPDU) can traverse before it is discarded. The default value is 20. You can specify a value between 1 and 255. This option only appears on the screen when you set the Mode property to MSTP.

Management of spanning tree instances

By default, the spanning tree protocol STP is enabled on all of the interfaces of the BIG-IP® system. The default spanning tree configuration includes a single spanning tree instance, named 0. A spanning tree instance is a discrete spanning tree for a network. While STP and RSTP allow only one spanning tree instance (instance 0), MSTP allows you to create multiple spanning tree instances, to manage redundant paths for specific VLANs on the network.

When running MSTP, instances that you create have instance members. An instance member is a VLAN that you assign to an instance when you create that instance. You can assign as many or as few members to an instance as you deem necessary. By default, all VLANs on the BIG-IP system are members of instance 0.

If you create an instance and attempt to add a VLAN that is already a member of another instance, the BIG-IP system deletes the VLAN from the existing instance and adds the VLAN to the new instance.

Each instance name must be a numeric value that you assign when you create the instance.

Note: *Only users with either the Administrator or Resource Administrator role can manage spanning tree instances.*

Spanning tree instances list

You can view a list of existing spanning tree instances using the BIG-IP Configuration utility. For STP and RSTP, the only instance listed is instance 0. For MSTP, the list shows instance 0, plus any other instances that you have explicitly created.

When you view a list of instances, you can see the following information for each instance:

- The name of the instance
- The bridge priority number
- The MAC address of the root bridge
- The MAC address of the regional root bridge
- The number of instance members

About spanning tree instance (MSTP-only) creation

The STP and RSTP protocols allow only one spanning tree instance, instance 0, which the BIG-IP® system creates automatically when you enable spanning tree. When running STP or RSTP, you can modify the properties of instance 0, but you cannot create additional instances.

When you are running MSTP, however, the MSTP algorithm can explicitly create instances. The reason that you can create instances is that MSTP recognizes VLANs. By creating an instance and assigning one or more VLANs to it, you can control bridge loops and redundant paths within those VLANs.

For example, suppose you have two interfaces. One interface is assigned to VLAN A, while the other interface is assigned to VLANs A and B. If you are using the STP or RSTP protocol, both of which disregard VLANs, the protocol might block traffic for both VLANs, as shown in this figure.

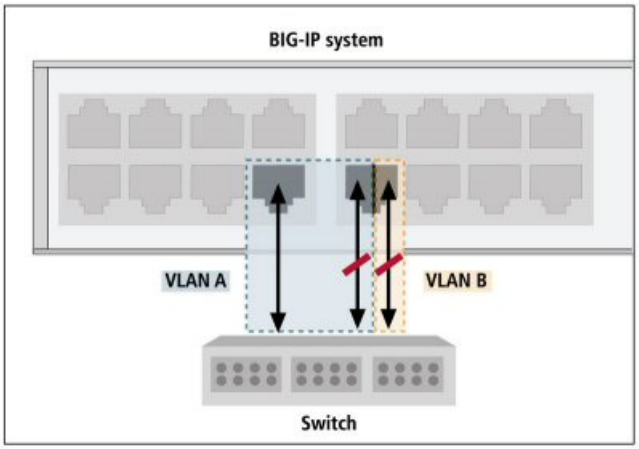

Figure 15: Using STP or RSTP to block redundant paths

By contrast, the MSTP protocol can make blocking decisions on a per-VLAN basis. In our example, on the interface that carries traffic for two VLANs, you can block traffic for VLAN A, but leave a path open for VLAN B traffic. For example:

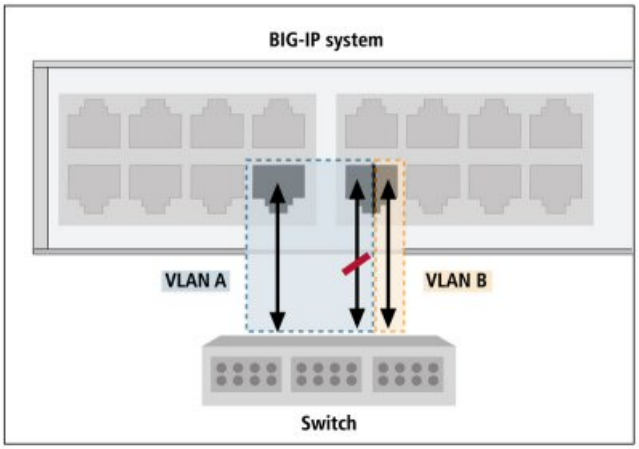

Figure 16: A local BIG-IP system that transmits and receives LLDPDUs

Because all BPDUs exchanged within a region always reference instance 0, instance 0 is active on all interfaces. This, in turn, can cause blocking problems. To avoid this, make sure that each VLAN on a BIG-IP system is a member of an instance that you explicitly create, rather than a member of instance 0 only. For example, suppose you create the following:

- Instance 1 with VLAN A as a member, where VLAN A is associated with interface 1.2
- Instance 2 with VLAN B as a member, where VLAN B is associated with interface 1.4

In this case, neither interface will be blocked, because the BPDUs sent from each interface reference a unique instance (either instance 1 or instance 2).

Tip: *Because all BPDUs exchanged within a region always reference instance 0, thereby causing instance 0 to be active on all interfaces, unwanted blocking problems can occur. To avoid this, make sure that each VLAN on a BIG-IP system is a member of an instance that you explicitly create, rather than a member of instance 0 only.*

About instance ID assignment

When you configure the Instance ID setting, you specify a numeric value for the instance, in the range of 1 to 255. The reason that instance names must be numeric is to handle the requirement that all cooperating bridges agree on the assignment of VLANs to instance IDs. Using numeric values instead of names makes this requirement easier to manage.

Bridge priority

The bridge in the spanning tree with the lowest relative priority becomes the root bridge. A *root bridge* represents the root of a spanning tree, and is responsible for managing loop resolution on the network. F5 Networks® recommends that you configure this setting so that the BIG-IP® system never becomes the root bridge. For this reason, the default value for the **Bridge Priority** setting is 61440, the highest value that you can select. Note that a bridge priority must be in increments of 4096.

VLAN assignment

If you are running MSTP, you can add members to a spanning tree instance. An *instance member* is a VLAN. You add members to an instance by associating one or more VLANs with the instance. The interfaces or trunks associated with each VLAN automatically become part of the spanning tree corresponding to that instance.

For two or more bridges to operate in the same spanning tree, all of those bridges must be in the same region, and therefore must have the same instance numbers, instance members, and VLAN tags.

For example, if a bridge has instance 1, with two VLAN members whose tags are 1000 and 2000, then any other bridges that you want to operate in that spanning tree must also have instance 1 with two VLAN members whose tags are 1000 and 2000.

A particular VLAN cannot be associated with more than one spanning tree instance. For example, if you have two instances named 0 and 1, you can only associate VLAN external with one of those instances, not both. Therefore, before creating an instance, verify that each VLAN you intend to associate with the instance is not a member of another instance.

Tip: *If no VLANs appear in the **Available** list when creating an instance, it is likely that all VLANs on the BIG-IP® system are members of other instances. You can verify this by viewing the members of other instances.*

About viewing and modifying a spanning tree instance

Using the BIG-IP Configuration utility, you can view and modify properties of any instance, including instance 0. If you are running MSTP, you can modify the Bridge Priority and VLANs properties. If you are running RSTP or STP, you can modify only the Bridge Priority property. In no case can you modify the instance ID.

About deleting a spanning tree instance or its members (MSTP-only)

If you are running MSTP, you might have explicitly created some spanning tree instances. If so, you can delete any spanning tree instance except instance 0.

You can also remove VLAN members from an instance. When you remove a VLAN from an instance, the VLAN automatically becomes a member of instance 0. (By default, instance 0 includes any VLAN that is not a member of another instance.)

If you remove all members from an instance, the BIG-IP® system automatically deletes the instance.

Note: *If you are running RSTP or STP, you cannot delete instance 0 or remove members from it.*

Interfaces for spanning tree

Some of the configuration tasks you perform when managing a spanning tree protocol pertain to BIG-IP® system interfaces. The interface-related tasks you perform are:

- Configuring settings on each interface that is to be part of the spanning tree
- Managing interfaces per spanning tree instance

About enabling and disabling spanning tree

When you select the check box for the **STP** setting, you are specifying that the interface can become part of a spanning tree. Once the interface becomes part of the spanning tree, the spanning tree protocol (STP) takes control of all learning and frame forwarding on that interface.

If you disable this setting, the spanning tree protocol treats the interface as non-existent, and does not send BPDUs to that interface. Also, the interface, and not the spanning tree protocol, controls all learning and frame forwarding for that interface.

Note that you can also enable spanning tree for a trunk. If spanning tree is enabled on the reference link of a trunk (that is, the lowest-numbered interface of the trunk), then spanning tree is automatically enabled on that trunk.

If you want to disable STP on a trunk, you must disable the protocol on the trunk itself. Disabling STP on an interface does not disable STP on a trunk.

STP link type

When you specify an STP link type, you ensure that STP uses the correct optimizations for the interface. Possible values are:

auto
When you set the STP link type to auto, the BIG-IP® system determines the spanning tree link type, which is based on the Active Duplex interface property.

p2p
When you set the STP link type to p2p, the BIG-IP system uses the optimizations for point-to-point spanning tree links. Point-to-point links connect two spanning tree bridges only. For example, a point-to-point link might connect a 10 Gigabit link to another bridge. For point-to-point links, the Active Duplex property interface should be set to full. Note that p2p is the only valid STP link type for a trunk.

shared
When you set the STP link type to shared, the BIG-IP system uses the optimizations for shared spanning tree links. Shared links connect two or more spanning tree bridges. For example, a shared link might be a 10 Megabit hub. Note that for shared links, the Active Duplex interface property should be set to half.

STP edge port

When you enable the **STP Edge Port** setting, you are explicitly designating the interface as an edge port. An *edge port* is an interface that connects to an end station rather than to another spanning tree bridge. The default setting is disabled (not checked).

If you would rather have the system automatically designate the interface as an edge port, you can enable the STP Edge Port Detection setting instead.

If you enable (check) the **STP Edge Port** setting and the interface subsequently receives STP, RSTP, or MSTP frames (BPDUs), the system disables the setting automatically, because only non-edge interfaces receive BPDUs.

Detection of an STP edge port

When you enable the **STP Edge Port Detection** setting, the system determines whether the interface is an edge port, and if so, automatically designates the interface as an edge port. The system determines edge port status by monitoring the interface and verifying that it does not receive any incoming STP, RSTP, or MSTP frames (BPDUs).

If the system determines that the interface is not an edge port, but you enabled the **STP Edge Port** setting to explicitly designate the interface as an edge port, the system removes the edge port designation from the interface. No interface that receives BPDUs from a bridge can have edge port status, despite the values of the STP Edge Port and STP Edge Port Detection settings.

About spanning tree protocol reset

The spanning tree algorithm automatically detects the presence of legacy STP bridges on the network, and falls back to STP mode when communicating with those bridges. Because legacy STP bridges do not send spanning tree BPDUs periodically in all circumstances, the BIG-IP® system cannot detect when a legacy STP bridge is removed from the network. Therefore, it is necessary to manually notify the BIG-IP system

that the algorithm can switch to the RSTP or MSTP protocol again, whenever a legacy bridge has been removed.

About managing interfaces for a specific instance

When you manage an interface for a specific spanning tree instance, you can:

- View a list of interfaces for an instance
- View instance-specific properties of an interface
- Configure instance-specific settings for an interface

About viewing a list of interface IDs for an instance

Using the BIG-IP Configuration utility, you can view a list of the interface IDs associated with a specific spanning tree instance.

If you are using MSTP, the interface IDs that appear in the list are the interfaces assigned to the VLANs that you specified when you created the instance. If you are using STP or RSTP, the interface IDs listed are those that the BIG-IP® system automatically assigned to instance 0.

The list of interface IDs also displays the following information for each interface:

- The STP instance ID
- The priority
- The external path cost
- The port role

About port roles

The Port Role property of a per-instance interface specifies the interface's role in the spanning tree instance. You cannot specify a value for this property; the BIG-IP® system automatically assigns a role to the interface.

The BIG-IP system can assign one of the following roles to an instance interface:

Disabled
The interface has no active role in the spanning tree instance.

Root
The interface provides a path to a root bridge.

Alternate
The interface provides an alternate path to a root bridge, if the root interface is unavailable.

Designated
The interface provides a path away from the root bridge.

Backup
The interface provides an alternate path away from the root bridge, if an interface with a port role of Designated is unavailable. The Backup role assignment is rare.

Port states

The Port State property of an interface specifies the way that the interface processes normal data packets. You cannot specify a value for this property; the BIG-IP® system automatically assigns a state to the interface.

An interface can be in one of the following states at any given time:

Blocking
The interface disregards any incoming frames, and does not send any outgoing frames.

Forwarding
The interface passes frames as needed.

Learning
The interface is determining information about MAC addresses, and is not yet forwarding frames.

Settings to configure for an interface for a specific instance

Selecting an interface priority

Each interface has an associated priority within a spanning tree instance. The relative values of the interface priorities affect which interfaces the system chooses to carry network traffic. Using the **Interface Priority** setting, you can select the interface's priority in relation to the other interfaces that are members of the spanning tree instance.

Typically, the system is more likely to select interfaces with lower numeric values to carry network traffic. A priority value that you assign to an interface can be in the range of 0 to 240, in increments of 16. Thus, the value you assign to an interface can be 0, 16, 32, 64, and so on, up to 240.

The default priority for an interface is 128, the middle of the valid range.

Specifying path cost

Each interface has an associated path cost within a spanning tree instance. The *path cost* represents the relative cost of sending network traffic through that interface. When calculating the spanning tree, the spanning tree algorithm attempts to minimize the total path cost between each point of the tree and the root bridge. By manipulating the path costs of different interfaces, you can steer traffic toward paths that are either faster, more reliable, more economical, or have all of these qualities.

The value of a path cost can be in the range of 1 to 200,000,000, unless you have legacy STP bridges. In that case, because some legacy implementations support a range of only 1 to 65535, you should use this more restricted range when setting path costs on interfaces.

The default path cost for an interface is based on the maximum speed of the interface rather than the actual speed.

For example, an interface that has a maximum speed of 1000 Mb/s (1 Gb/s), but is currently running at a speed of 10 Mb/s, has a default path cost of 20,000.

Link aggregation does not affect the default path cost. For example, if a trunk has four 1 Gb/s interfaces, the default path cost is 20,000.

For MSTP, you can set two kinds of path costs, external and internal. For STP and RSTP, you can set an external path cost only.

External Path Cost

The **External Path Cost** setting is used to calculate the cost of sending spanning tree traffic through the interface to reach an adjacent spanning tree region. The spanning tree algorithm tries to minimize the total path cost between each point of the tree and the root bridge. The external path cost applies only to those interfaces (and trunks) that are members of instance 0.

Internal Path Cost

The **Internal Path Cost** setting allows you to specify the relative cost of sending spanning tree traffic through the interface to adjacent bridges within a spanning tree region. Note that the internal path cost applies only to bridges that support the MSTP mode. The internal path cost applies to those interfaces (and trunks) that are members of any instance, including instance 0.

To summarize, STP and RSTP use external path costs only, and the costs apply to instance 0 interfaces only. MSTP uses both external and internal path costs, and the internal costs apply to interfaces in all spanning tree instances, including instance 0.

Chapter 13

WCCP

About WCCPv2 redirection on the BIG-IP system

The BIG-IP® system includes support for Web Cache Communication Protocol version 2 (WCCPv2). *WCCPv2* is a content-routing protocol developed by Cisco® Systems. It provides a mechanism to redirect traffic flows in real time. The primary purpose of the interaction between WCCPv2-enabled routers and a BIG-IP® system is to establish and maintain the transparent redirection of selected types of traffic flowing through those routers.

To use WCCPv2, you must enable WCCPv2 on one or more routers connected to the BIG-IP® system, and configure a service group on the BIG-IP system that includes the router information. The BIG-IP system then receives all the network traffic from each router in the associated service group, and determines both the traffic to optimize and the traffic to which to apply a service.

In configuring WCCPv2 on a network, you define a *service group* on the BIG-IP system, which is a collection of WCCPv2 services configured on the BIG-IP system. A WCCPv2 *service* in this context is a set of redirection criteria and processing instructions that the BIG-IP system applies to any traffic that a router in the service group redirects to the BIG-IP system. Each service matches a service identifier on the router.

The following illustration shows a one-arm configuration on one side of the WAN and an inline (bridge) configuration on the other side.

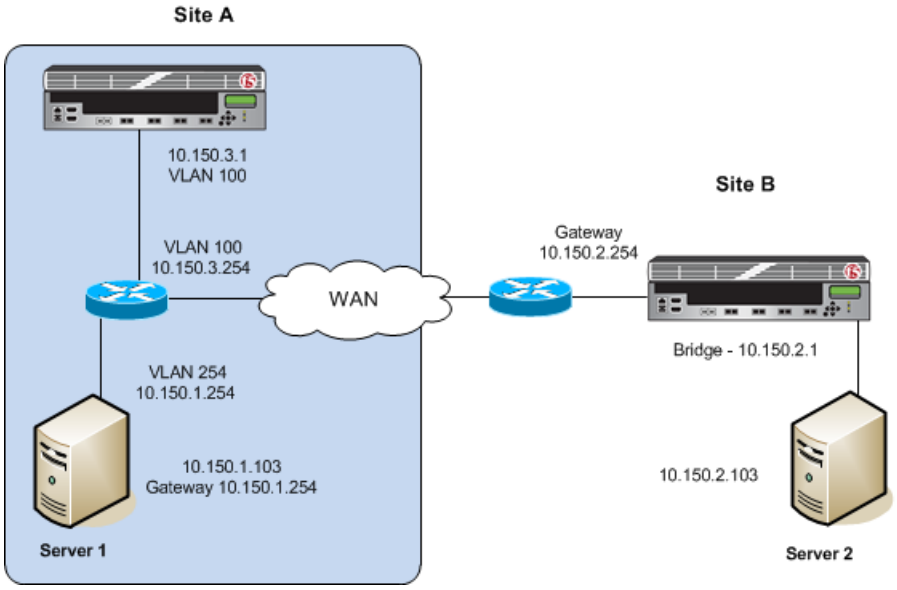

Figure 17: Example of a one-arm configuration

A common deployment of the WCCPv2 protocol

In certain cases, it is not advantageous or even possible to deploy the BIG-IP® system inline. For example, in the case of a collapsed backbone where the WAN router and the LAN switch are in one physical device, you might not be able to deploy the BIG-IP system inline.

If you choose not to deploy the BIG-IP system inline, you can use a one-arm deployment. In a *one-arm deployment*, the BIG-IP system has a single (hence, one-arm) connection to the WAN router or LAN switch. The WAN router (or switch) redirects all relevant traffic to the BIG-IP system. In this configuration, the WAN router typically uses Web Cache Communication Protocol version 2 (WCCPv2) to redirect traffic to the BIG-IP system.

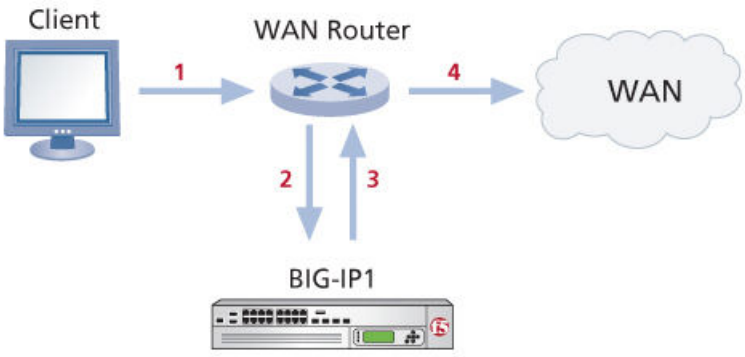

Figure 18: Network topology for a one-arm connection

The traffic flow sequence in this illustration is as follows:

1. The client initiates a session.
2. A WAN router redirects traffic to the BIG-IP system.
3. The BIG-IP1 processes traffic and sends it back to the WAN router.

4. The WAN router forwards traffic across the WAN.

Index

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

V

W