



3-DNS® Reference Guide

version 4.1

MAN-0047-00

Service and Support Information

Product Version

This manual applies to version 4.1 of the 3-DNS® Controller.

Obtaining Technical Support

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

Contacting F5 Networks

Web	www.f5.com
Toll-free phone	(888) 88BIG-IP
Corporate phone	(206) 272-5555
Fax	(206) 272-5556
Email	sales@f5.com
Mailing Address	401 Elliott Avenue West Seattle, Washington 98119

Legal Notices

Copyright

Copyright 1998-2001, F5 Networks, Inc. All rights reserved.

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

Trademarks

F5 and the F5 logo, F5 Networks, BIG-IP, 3-DNS, GLOBAL-SITE, and SEE-IT are registered trademarks of F5 Networks, Inc. EDGE-FX, FireGuard, iControl, Internet Control Architecture, and IP Application Switch are trademarks of F5 Networks, Inc. In Japan, the F5 logo is trademark number 4386949, BIG-IP is trademark number 4435184, 3-DNS is trademark number 4435185, and SEE-IT is trademark number 4394516. All other product and company names are registered trademarks or trademarks of their respective holders.

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.

Export Warning

This is a Class A product. In a domestic environment, this product may cause radio interference in which case the user may be required to take adequate measures.

FCC Compliance

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

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

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

Canadian Regulatory Compliance

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

Standards Compliance

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

Acknowledgments

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

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

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

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

This product includes software developed by Adam Glass.

This product includes software developed by Christian E. Hopps.

This product includes software developed by Dean Huxley.

This product includes software developed by John Kohl.

This product includes software developed by Paul Kranenburg.

This product includes software developed by Terrence R. Lambert.

This product includes software developed by Philip A. Nelson.

This product includes software developed by Herb Peyerl.

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

This product includes software developed by Chris Provenzano.

This product includes software developed by Theo de Raadt.

This product includes software developed by David Muir Sharnoff.

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

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

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

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

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

This product includes software developed by Christos Zoulas.

This product includes software developed by Charles Hannum.

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

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

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

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

This product includes software developed by the Apache Group for use in the Apache HTTP server project (<http://www.apache.org/>).

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

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

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

This product includes software developed by Eric Young.

Rsync was written by Andrew Tridgell and Paul Mackerras, and is available under the Gnu Public License.

This product includes Malloc library software developed by Mark Moraes. (© 1988, 1989, 1993, University of Toronto).

This product includes open SSL software developed by Eric Young (eay@cryptsoft.com), (© 1995-1998).

This product includes open SSH software developed by Tatu Ylonen (ylo@cs.hut.fi), Espoo, Finland (© 1995).

This product includes open SSH software developed by Niels Provos (© 1999).

This product includes SSH software developed by Mindbright Technology AB, Stockholm, Sweden, www.mindbright.se, info@mindbright.se (© 1998-1999).

This product includes free SSL software developed by Object Oriented Concepts, Inc., St. John's, NF, Canada (© 2000).

This product includes software developed by Object Oriented Concepts, Inc., Billerica, MA, USA (© 2000).



Table of Contents

1		
Introduction		
	Getting started	1-1
	Using the Administrator Kit	1-1
	Stylistic conventions	1-2
	Finding help and technical support resources	1-3
2		
Access Control Lists		
	Access control lists	2-1
3		
The big3d Agent		
	Working with the big3d agent	3-1
	Setting up data collection with the big3d agent	3-1
	Collecting path data and server performance metrics	3-1
	Installing the big3d agent	3-2
	Understanding factories run by big3d agents	3-3
	Understanding the data collection and broadcasting sequence	3-4
	Tracking LDNS probe states	3-4
	Evaluating big3d agent configuration trade-offs	3-4
	Setting up communication between 3-DNS Controllers and other F5 servers	3-5
	Setting up iQuery communications for the big3d agent	3-6
	Allowing iQuery communications to pass through firewalls	3-7
4		
Extended Content Verification (ECV)		
	Working with the ECV Service Monitor	4-1
5		
Load Balancing		
	Working with load balancing modes	5-1
	Understanding load balancing	5-1
	Using static load balancing modes	5-2
	Using dynamic load balancing modes	5-6
	Configuring load balancing	5-10
	Understanding wide IPs	5-11
	Understanding pools	5-11
	Defining a wide IP	5-12
	An example of the wideip statement	5-14
	Using the LDNS round robin wide IP attribute	5-14
	Using the last resort pool designation	5-15
	Changing global variables that affect load balancing	5-16
	Setting global alternate and fallback methods	5-16
	Understanding TTL and timer values	5-17
	Troubleshooting manual configuration problems	5-20

6

Network Map

Introducing the Network Map	6-1
Working with the Network Map	6-2
Using the information table on the Network Map	6-3

7

Production Rules

Controlling network traffic patterns with production rules	7-1
Setting up production rules in the Configuration utility	7-1
Viewing, adding, and deleting production rules	7-2
Choosing the rule type	7-2
Defining time-based triggers	7-2
Defining event-based triggers	7-4
Choosing the action taken	7-4
Working with the production rules scripting language	7-5
Inserting production rules in the wideip.conf file	7-5
Executing and managing production rules	7-6
Working with the if statement	7-7
Working with the when statement	7-8
Working with the every statement	7-9
Defining production rule actions	7-10
Production rule examples	7-11

8

Resource Records

Understanding resource records	8-1
Types of resource records	8-2
A (Address)	8-2
CNAME (Canonical Name)	8-3
MX (Mail Exchange)	8-3
NS (Name Server)	8-3
PTR (Pointer)	8-4
SOA (Start of Authority)	8-4
Additional resource record types	8-5

9

Scripts

Working with scripts	9-1
3dns_add script	9-1
3dns_admin_start script	9-1
3dns_dump script	9-1
3dns_sync_metrics script	9-1
3dns_web_config script	9-2
3dns_web_passwd script	9-2
3dnsmaint script	9-2
3dprint script	9-2
3ndc script	9-4
big3d_install script	9-4

big3d_restart script	9-4
big3d_version script	9-5
config_ssh script	9-5
edit_lock script	9-5
edit_wideip script	9-5
install_key script	9-6
syncd_checkpoint script	9-6
syncd_rollback script	9-7
syncd_start script	9-8
syncd_stop script	9-8

10

SNMP

Working with SNMP on the 3-DNS Controller	10-1
Configuring SNMP on the 3-DNS Controller	10-1
Downloading the MIBs	10-2
Understanding configuration file requirements	10-2
Configuring options for the checktrap.pl script	10-6
Configuring the 3-DNS SNMP agent using the Configuration utility	10-7
Configuring host SNMP settings on the 3-DNS Controller	10-7
Configuring the SNMP agent on host servers	10-10

11

Topology

Working with Topology load balancing	11-1
Setting up topology records	11-1
Using the Topology load balancing mode in a wide IP	11-3
Using the Topology load balancing mode in a pool	11-4
Working with the topology statement in the wideip.conf file	11-5

12

Utilities

Working with command line utilities	12-1
Accessing 3-DNS Controller utilities documentation	12-1

13

wideip.conf Configuration

Overview of the wideip.conf file	13-1
Using include files	13-1
Syntax for include files	13-2
Working with statements	13-4
Syntax rules	13-4
The globals statement	13-6
The server statement	13-19
The datacenter statement	13-32
The sync_group statement	13-33
The wide IP statement	13-34
The topology statement	13-42

Table of Contents

Access control lists	13-44
Working with comments	13-45
Syntax	13-46
Definition and usage	13-46
Understanding current values	13-47
Server definition current values	13-48
Virtual server definition current values	13-49
Local DNS server paths current values	13-50
Wide IP definition current values	13-51

Glossary

Index



I

Introduction

- Getting started
- Using the Administrator Kit
- Finding help and technical support resources

Getting started

The *3-DNS Reference Guide* includes information about the features of the 3-DNS Controller. It also contains information about system configuration files and variables, command line syntax, scripts and utilities, and other 3-DNS objects. Use the *3-DNS Reference Guide* for help in configuring a specific feature of the 3-DNS Controller. For load balancing and networking solutions, see the *3-DNS Administrator Guide*.

Using the Administrator Kit

The 3-DNS® Administrator Kit provides simple steps for quick, basic configuration, and also provides detailed information about more advanced features and tools, such as the **3dnsmaint** command line utility. The information is organized into three guides as described below.

- ◆ **Installation Guide**

The *3-DNS Installation Guide* walks you through the basic steps needed to get the hardware plugged in and the system connected to the network. Most users turn to this guide only the first time that they set up a 3-DNS Controller. The Installation Guide also covers general network administration issues, such as setting up common network administration tools including Sendmail. If you are running the 3-DNS software module on a BIG-IP unit, refer to the *BIG-IP Installation Guide*.

- ◆ **Administrator Guide**

The *3-DNS Administrator Guide* provides examples of common wide-area load balancing solutions supported by the 3-DNS Controller. For example, in the Administrator Guide, you can find everything from a basic DNS request load balancing solution to a more advanced content acceleration load balancing solution.

- ◆ **Reference Guide**

The *3-DNS Reference Guide* provides basic descriptions of individual 3-DNS Controller objects, such as wide IPs, pools, virtual servers, load balancing modes, the **big3d** agent, resource records, and production rules. It also provides syntax information for **3dnsmaint** commands, configuration utilities, the *wideip.conf* file, and system utilities.

- ◆ **Note**

*If you are configuring the 3-DNS module on the BIG-IP Controller, you use the **BIG-IP Installation Guide** to set up and configure the hardware.*

Stylistic conventions

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

◆ WARNING

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

Identifying new terms

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

Identifying references to objects, names, and commands

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

Identifying references

We use italic text to denote a reference to another document or another section in the current document. In references where we provide the name of a book as well as a specific chapter or section in the book, we show the book name in bold, italic text, and the chapter/section name in italic text to help quickly differentiate the two. For example, you can find information about ***3dnsmaint*** commands in the section *3dnsmaint script*, on page 9-2.

Identifying command syntax

We show actual, complete commands in bold Courier text. Note that we do not include the corresponding screen prompt, unless the command is shown in a figure that depicts an entire command line screen. For example, the following command shows the current status of the 3-DNS daemons:

```
ndc status
```

Table 1.1 explains additional special conventions used in command line syntax.

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

Table 1.1 Command line conventions used in this manual

Finding help and technical support resources

You can find additional technical documentation about the 3-DNS Controller in the following locations:

◆ Release notes

The release note for the current version of the 3-DNS Controller is available from the home page of the Configuration utility. The release note contains the latest information for the current version including a list of new features and enhancements, a list of fixes, and a list of known issues.

◆ Online help for 3-DNS Controller features

You can find help online in three different locations:

- The Configuration utility home page has PDF versions of the guides included in the Administrator Kit. Software upgrades for the 3-DNS Controller replace the guides with updated versions as appropriate.
- The Configuration utility has online help for each screen. Just click the **Help** button in the toolbar.
- Individual commands have online help, including command syntax and examples, in standard UNIX man page format. Simply type **man** followed by the command (for example **man ndc**), and the 3-DNS Controller displays the syntax and usage associated with the command.

- ◆ **Third-party documentation for software add-ons**

The Configuration utility contains online documentation for the third-party software included with the 3-DNS Controller, including NameSurfer and GateD.

- ◆ **Technical support through the World Wide Web**

The F5 Networks Technical Support web site, <http://tech.F5.com>, contains the Ask F5 knowledge base and provides the latest technical notes and updates for the Administrator Kit guides (in PDF and HTML formats). To access this site you must first email askf5@f5.com to obtain a customer ID and a password.



2

Access Control Lists

Access control lists

With access control lists (ACLs), you can block probing for members of the ACL when you use dynamic RTT probing on your 3-DNS Controller. Table 2.1 lists the ACL types and describes their functions.

ACL Type	Description
Prober	Prober ACLs limit round-trip time probes.
Hops	Hops ACLs limit traceroute probes.
Discovery	Discovery ACLs limit port discovery probes.

Table 2.1 Access control list types and descriptions

To define ACLs using the Configuration utility

1. In the navigation pane, click **System**.
The System - General screen opens.
2. On the toolbar, click **ACL**.
The ACL Configuration screen opens.
3. Add the settings for the ACLs you want to create, and click **Update**.
For more information on this screen, click **Help** on the toolbar.

To define ACLs from the command line

1. If one does not already exist, create a file called **region.ACL** in the **/var/3dns/include** directory. You must add the **include** file at the beginning of the **wideip.conf** file.
2. Add the file to **/etc/wideip.conf** by typing, at the command line:

```
include "region.ACL"
```

◆ Tip

*When you create ACLs by editing the **wideip.conf** file from the command line, we strongly recommend that you put the ACLs in a separate **include** file.*

The ACLs you can create are **probe_acl**, **hops_acl**, and **discovery_acl**. Figure 2.1 is an example the syntax for a **region.ACL** file with definitions for the three ACL types.

```
actions {
    NO_RELAY
    delete rdb ACL region "probe_acl"
    delete rdb ACL region "hops_acl"
    delete rdb ACL region "discovery_acl"
}
region_db ACL {
    region {
        name "probe_acl"
        region "probe_acl"
        192.168.4.0/24
    }
    region {
        name "hops_acl"
        192.168.2.0/16
    }
    region {
        name "discovery_acl"
        192.168.11.11/32
        192.168.4.0/24
    }
}
```

Figure 2.1 Sample region.ACL file



3

The big3d Agent

- Working with the big3d agent
- Installing the big3d agent
- Understanding factories run by big3d agents
- Understanding the data collection and broadcasting sequence
- Setting up communication between 3-DNS Controllers and other F5 servers

Working with the big3d agent

The **big3d** agent collects performance information on behalf of the 3-DNS Controller. The **big3d** agent runs on 3-DNS Controllers, BIG-IP Controllers, EDGE-FX Caches, and GLOBAL-SITE Controllers; the default setting is to run a **big3d** agent on all of these controllers in the network, but you can turn off the **big3d** agent on any controller at any time.

Setting up data collection with the big3d agent

Setting up the **big3d** agents involves the following tasks:

- ◆ **Installing big3d agents on BIG-IP Controllers and EDGE-FX Caches**
Each new version of the 3-DNS Controller software includes the latest version of the **big3d** agent. You need to distribute that copy of the **big3d** agent to the BIG-IP Controllers and EDGE-FX Caches in the network. See the release notes provided with the 3-DNS Controller software for information about which BIG-IP Controller and EDGE-FX Cache versions the current **big3d** agent supports. For details on installing the **big3d** agent, see *Installing the big3d agent*, on page 3-2.
- ◆ **Specifying which factories a specific big3d agent manages**
When you define BIG-IP Controllers, EDGE-FX Caches, and 3-DNS Controller servers, you can change the default **big3d** agent settings on a specific controller. You can change the number of factories the **big3d** agent runs and turn specific factories on and off.
- ◆ **Setting up communications between big3d agents and controllers**
Before the **big3d** agents can communicate with the 3-DNS Controllers in the network, you need to configure the appropriate ports and tools to allow communication between the devices running the **big3d** agent and 3-DNS Controllers in the network. These planning issues are discussed in *Setting up communication between 3-DNS Controllers and other F5 servers*, on page 3-5.

Collecting path data and server performance metrics

A **big3d** agent collects the following types of performance information used for load balancing. This information is broadcast to all 3-DNS Controllers in your network.

- ◆ **Virtual server availability**
The **big3d** agent queries virtual servers to verify whether they are up and available to receive connections. For name resolution, the 3-DNS Controller uses only those virtual servers that are **up**.

◆ **Network path round trip time**

The **big3d** agent calculates the round trip time for the network path between the data center and the client's LDNS server that is making the resolution request. Round trip time is used in determining the best virtual server when using the Round Trip Times or the Quality of Service modes.

◆ **Network path packet loss**

The **big3d** agent calculates the packet completion percentage for the network path between the data center and the client's LDNS server that is making the resolution request. Packet completion is used in determining the best virtual server when using the Completion Rate or the Quality of Service modes.

◆ **Hops along the network path**

The **big3d** agent calculates the number of intermediate systems transitions (hops) between the data center and the client's LDNS server. Hops are used in determining the best virtual server when using the Hops or the Quality of Service load balancing modes.

◆ **Server performance**

The **big3d** agent calculates the packet rate of the BIG-IP Controller or SNMP-enabled hosts. Packet rate is used in determining the best virtual server when using the Packet Rate or the Quality of Service load balancing modes.

◆ **Virtual server performance**

The **big3d** agent calculates the number of connections to virtual servers defined on BIG-IP Controllers or SNMP-enabled hosts. The number of connections is used to determine the best virtual server when using the Least Connections load balancing mode.

Installing the big3d agent

You can easily install the **big3d** agent on the BIG-IP Controllers, EDGE-FX Caches, and GLOBAL-SITE Controllers in your network by using the 3-DNS Maintenance menu.

To install the big3d agent from the command line

1. Log on to the 3-DNS Controller using either a remote shell, a serial terminal, or a keyboard and monitor attached directly to the controller.
2. At the command prompt, type **3dnsmaint**.
The 3-DNS Maintenance menu opens.
3. Choose the **Install and Start big3d** command from the menu and press Enter.

Understanding factories run by big3d agents

To gather performance information, the **big3d** agent uses different types of factories. A *factory* is a process that collects different types of data. The **big3d** agent currently supports five factory types:

- ◆ **Prober factory**

A prober factory collects several types of information using ICMP, TCP, UDP, DNS_DOT, or DNS_REV protocols. This factory queries host virtual servers and local DNS servers. Host virtual servers are checked to determine their **up** or **down** state. For local DNS servers, the prober factory uses the response time to calculate the round trip time and packet loss between the LDNS and the data center.

- ◆ **Hops factory**

A hops factory uses the traceroute method to calculate the number of intermediate systems transitions along the network path between a specific data center and a client LDNS.

- ◆ **SNMP factory**

An SNMP factory uses conversations with SNMP agents that run on host servers to collect performance metrics for the host.

- ◆ **Discovery factory**

A discovery factory acts as a backup to a prober factory. The **big3d** agent runs a discovery factory only when a prober factory fails to get a response from a specific LDNS. The **big3d** agent uses the discovery factory to look for an alternate port on an LDNS that can respond to the queries issued by a prober factory. If the discovery factory finds an open port, it returns the port number to the 3-DNS Controller, which stores the number to use for future path probe attempts.

- ◆ **Permanent factories**

Two permanent factories collect performance information. One factory collects information from the BIG-IP Controller when it exists, the other collects the number of packets being processed per second. These factories are not configurable.

The standard configuration specifies that each BIG-IP Controller, EDGE-FX Cache, and 3-DNS Controller in the network run a **big3d** agent using five prober factories, one SNMP factory, one discovery factory, no hops factories, and the two permanent factories. In the BIG-IP Controller or 3-DNS Controller server definition, you can change the number of factories that the **big3d** agent runs. For example, the default number of hops factories is set to **0**; if you want to run a hops factory, you change the setting to **1** or more.

Understanding the data collection and broadcasting sequence

The **big3d** agents collect and broadcast information on demand. The principal 3-DNS Controller in the sync group issues a data collection request to all **big3d** agents running in the network. In turn, the **big3d** agents collect the requested data using the factories, and then broadcast that data to all 3-DNS Controllers running in the network, including the principal controller that issued the request.

Tracking LDNS probe states

The 3-DNS Controller tracks the state of path data collection for each LDNS that has ever requested a name resolution from the controller. Table 3.1 shows the six states that can be assigned to an LDNS. Note that you can view the state of LDNS servers in the Local DNS Statistics screen in the Configuration utility.

State	Description
Needs Probe	The big3d agent has never collected data for the LDNS, or the data has expired.
Idle	The big3d agent successfully collected data for the LDNS, and is waiting for the next collection request.
In Probe	The big3d agent is currently collecting data for the LDNS.
Needs Discovery	The big3d agent failed to collect data for the LDNS using its standard protocols and ports, and now needs to run the LDNS through a discovery factory.
In Discovery	The big3d agent is currently running the LDNS through a discovery factory to look for an alternate available port.
Suspended	The big3d agent failed to discover a port open for data collection, and the LDNS is no longer eligible for probing.

Table 3.1 Probe and discovery states for individual client LDNS servers

Evaluating big3d agent configuration trade-offs

You must run a **big3d** agent on each BIG-IP Controller, 3-DNS Controller, and EDGE-FX Cache. If you are using advanced load balancing modes, you must have a **big3d** agent running on at least one controller in each data center to gather the necessary path metrics.

The load on the **big3d** agents depends on two factors: the timer settings that you assign to the different types of data the **big3d** agents collect, and the number of factories that each **big3d** agent runs. The shorter the timers, the more frequently the agent needs to refresh the data. While short timers guarantee that you always have valid data readily available for load balancing, they also increase the frequency of data collection. The more factories a **big3d** agent runs, the more metrics it can refresh at one time, and the more quickly it can refresh data for the 3-DNS Controller.

Another factor that can affect data collection is the number of client LDNS servers that make name resolution requests. The more LDNS servers that make resolution requests, the more paths that the **big3d** agent has to collect. While round trip time for a given path may vary constantly due to current network load, the number of hops along a network path between a data center and a specific LDNS does not often change. Consequently, you may want to set short timer settings for round trip time data so that it refreshes more often, but set high timer settings for hops data because it does not need to be refreshed often.

Setting up communication between 3-DNS Controllers and other F5 servers

In order to copy **big3d** agents from the 3-DNS Controllers to BIG-IP Controllers and EDGE-FX Caches, the 3-DNS Controllers need to communicate with BIG-IP Controllers. If you use exclusively crypto 3-DNS Controllers and crypto BIG-IP Controllers, or exclusively non-crypto 3-DNS Controllers and non-crypto BIG-IP Controllers, the communication tools set up by the First-Time Boot utility are all you need. Crypto controllers all use **ssh** and **scp**, and non-crypto controllers all use **rsh** and **rcp**.

However, if you work in a mixed environment where some controllers are crypto, and other controllers are non-crypto, you need to enable the **rsh** and **rcp** tools on the crypto controllers. These tools come pre-installed on all crypto 3-DNS Controllers and BIG-IP Controllers, but you must explicitly enable them.

To enable the rlogin tools on a BIG-IP Controller

From the command line, run the **rsetup** script.

◆ Note

*You can disable **rsh** and **rcp** access at any time either by running the **config_rshd** script, or by changing the **bigip.open_rsh_ports** system control variable to **0** in **/etc/rc.sysctl**.*

Table 3.2 shows the ports and protocols that 3-DNS Controllers use to communicate with BIG-IP Controllers and EDGE-FX Caches.

From	To	Protocol	From Port	To Port	Purpose
Crypto 3-DNS Controller	Crypto BIG-IP Controller, Crypto EDGE-FX Cache	TCP	<1023	22	SSH/SCP
Non-crypto 3-DNS Controller	Non-crypto BIG-IP Controller, Non-crypto EDGE-FX Cache	TCP	>1024	514	RSH/RCP
Crypto 3-DNS Controller	Non-crypto BIG-IP Controller, Non-crypto EDGE-FX Cache	TCP	>1024	514	RSH/RCP
Non-crypto BIG-IP Controller, Non-crypto EDGE-FX Cache	Crypto 3-DNS Controller	N/A	N/A	N/A	N/A

Table 3.2 Communications between 3-DNS Controllers, BIG-IP Controllers, and EDGE-FX Caches

Note that if you run **big3d** agents in a mixed crypto/non-crypto environment, the crypto controllers automatically turn off Blowfish encryption when communicating with non-crypto **big3d** agents. When communicating with crypto **big3d** agents, however, crypto 3-DNS Controllers always use Blowfish encryption by default, although you can manually disable it if you prefer.

Setting up iQuery communications for the big3d agent

The iQuery protocol uses one of two ports to communicate between the **big3d** agents and the 3-DNS Controllers. The ports used by iQuery traffic change, depending on whether the traffic is inbound from the **big3d** agent or outbound from the 3-DNS Controller.

Table 3.3 shows the protocols and corresponding port numbers used for iQuery communications between 3-DNS Controllers and **big3d** agents that run on 3-DNS Controllers, BIG-IP Controllers, or EDGE-FX Caches.

From	To	Protocol	From Port	To Port	Purpose
3-DNS Controller	big3d agent	UDP	245, 4354	245	Old standard iQuery port for outbound traffic
3-DNS Controller	big3d agent	UDP TCP	4353 or 4354	4354 or 4353	Alternate iQuery port for outbound traffic (open this port only when the use_alternate_iq global variable is set to yes)
big3d agent	3-DNS Controller	UDP TCP	245 or 4353	4354	Ephemeral port used for inbound iQuery traffic (when multiplex_iq is set to no)
big3d agent	3-DNS Controller	UDP	245	245	Single port used for multiplexed inbound iQuery traffic (open this port only when the multiplex_iq global variable is set to yes)
big3d agent	3-DNS Controller	UDP TCP	4353 4354 4353	4353 4353 4354	Single port used for multiplexed inbound iQuery traffic (open this port only when both the use_alternate_iq and the multiplex_iq global variables are set to yes)

Table 3.3 Communication protocols and ports between 3-DNS Controllers and **big3d** agents

Table 3.4 shows the protocols and corresponding ports used for iQuery communications between **big3d** agents and SNMP agents that run on host servers.

From	To	Protocol	From Port	To Port	Purpose
big3d agent	host SNMP agent	UDP	>1024	161	Ephemeral ports used to make SNMP queries for host statistics
host SNMP agent	big3d agent	UDP	161	>1024	Ephemeral ports used to receive host statistics using SNMP

Table 3.4 Communication protocols and ports between **big3d** agents and SNMP agents

Allowing iQuery communications to pass through firewalls

The payload information of an iQuery packet contains information that potentially requires translation when there is a firewall in the path between the **big3d** agent and the 3-DNS Controller. Only packet headers are translated by the firewall, payloads are not.

The iQuery translation option resolves this issue. With iQuery translation turned on, the iQuery packet stores the original IP address in the packet payload itself. When the packet passes through a firewall, the firewall translates the IP address in the packet header normally, but the IP address within the packet payload is preserved. The 3-DNS Controller reads the IP address out of the packet payload, rather than out of the packet header.

In the example configuration shown in Figure 3.1, a firewall separates the path between a BIG-IP Controller running a **big3d** agent and the 3-DNS Controller. The packet addresses are translated at the firewall. However, addresses within the iQuery payload are not translated, and they arrive at the BIG-IP Controller in their original states.

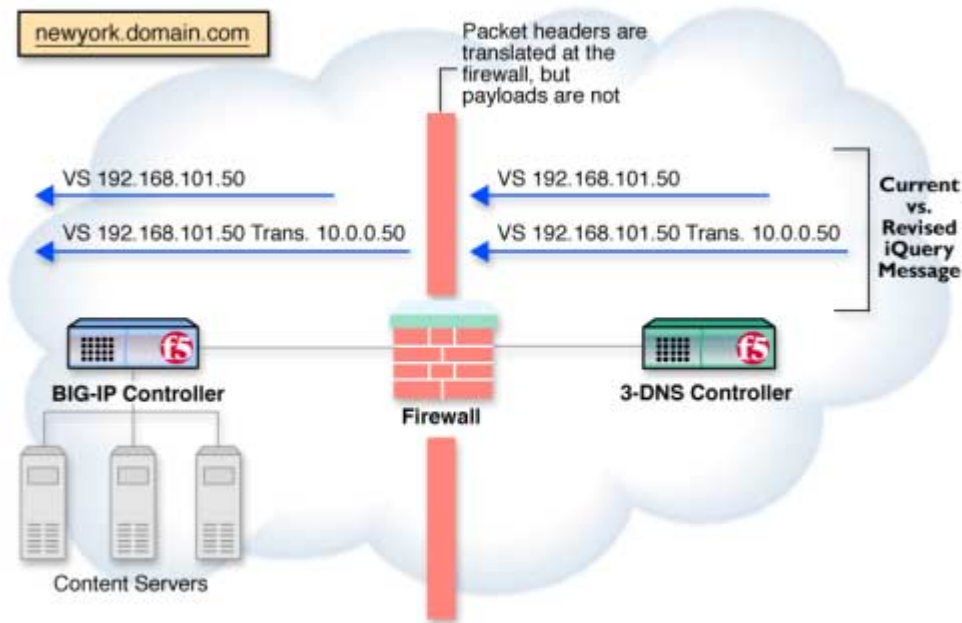


Figure 3.1 Translating packet address through the firewall

Communications between 3-DNS Controllers and other machines in the network

The following tables show the other ports and protocols that 3-DNS Controllers use for communication. Table 3.5 shows the ports that 3-DNS Controllers use for remote administrative connections to the 3-DNS web server.

From	To	Protocol	Port	Purpose
Configuration utility on a remote workstation	Crypto 3-DNS Controller	https over TCP	443	Connection to secure web server
Configuration utility on a remote workstation	Non-crypto 3-DNS Controller	http over TCP	80	Connection to standard web server

Table 3.5 Communications between 3-DNS Controllers and remote workstations

Table 3.6 shows the ports on which the 3-DNS Controller receives and responds to DNS resolution requests issued by LDNS servers.

From	To	Protocol	From Port	To Port	Purpose
LDNS	3-DNS Controller	UDP	53 or >1024	53	DNS resolution requests
3-DNS Controller	LDNS	UDP	53	53 or >1024	DNS resolution responses

Table 3.6 DNS communications on the 3-DNS Controller

Table 3.7 shows the protocols and ports that the **big3d** agent uses when collecting path data for local DNS servers.

From	To	Protocol	From Port	To Port	Purpose
big3d agent	LDNS	ICMP	N/A	N/A	Probing using ICMP pings
big3d agent	LDNS	TCP	2000-12000	53	Probing using TCP (Cisco routers should "allow establish")
LDNS	big3d agent	TCP	53	2000-12000	Probing using TCP (Cisco routers should "allow establish")
big3d agent	LDNS	UDP	2000-12000	33434	UDP probing and traceroute
LDNS	big3d agent	ICMP	N/A	N/A	Replies from ICMP, UDP pings, or traceroute probes

Table 3.7 Communications between **big3d** agents and local DNS servers

From	To	Protocol	From Port	To Port	Purpose
big3d agent	LDNS	dns_rev dns_dot	>1024	53	DNS version or DNS dot queries
LDNS	big3d agent	dns_rev dns_dot	53	>1024	DNS version or DNS dot responses

*Table 3.7 Communications between **big3d** agents and local DNS servers*

The **big3d** agent can run on a 3-DNS Controller, a BIG-IP Controller, an EDGE-FX Cache, or a GLOBAL-SITE Controller. If you run a **big3d** agent on a BIG-IP Controller and you set the SNMP prober factory count to **1** or higher, the **big3d** agent automatically opens UDP ports to allow for SNMP communications. If you do not want to open UDP ports for this purpose, you need to set the SNMP factory count to **0**.



4

Extended Content Verification (ECV)

- Working with the ECV Service Monitor

Working with the ECV Service Monitor

When you set up an extended content verification (ECV) service monitor for a wide IP, you can monitor not only the availability of a port or service on a server, but also the availability of a specific file on a particular server. An ECV service monitor verifies whether a specific file is available using the HTTP, HTTPS, or FTP network services.

An ECV service monitor can help you ensure that clients are getting what they are after, and that they will not get an error, whether they are looking for information, making an online purchase, or uploading software.

An ECV service monitor works in the following manner: if the file responds appropriately to the ECV query, the server where the file resides is marked as **up** and the client will be sent to that server. If the file does not respond as expected to the ECV query, the server where the file resides is marked as **down**, and the client will not be sent to that server.

To define ECV service monitors using the Configuration utility

1. In the navigation pane, click **Wide IPs**.
The Wide IP List screen opens.
2. In the Wide IP column, click the wide IP to which you want to add an ECV service monitor.
The Modify Wide IP screen opens.
3. Add the settings for the ECV near the bottom of the screen, and click **Update**. For more information on the ECV settings, click **Help** on the toolbar.

To define ECV service monitors from the command line

1. At the command prompt, type **3dnsmaint** to open the 3-DNS Maintenance menu.
2. On the 3-DNS Maintenance menu, choose **Edit 3-DNS Configuration** to open the **wideip.conf** file.
3. Use the syntax shown in Figure 4.1 to define an ECV service monitor.

You should place all ECV service monitor statements just before the wide IP pool definitions in the **wideip.conf** file.

◆ Note

*You can set up ECV monitors that use the **https** protocol on crypto 3-DNS Controllers.*

```
ecv {
  protocol      <http | https | ftp>
  filename      <"path and file name">
  scan_level    <all | first>
  user          [ <"user name"> ]
  hashed_password [ <"hashed version of server password"> ]
}
```

Figure 4.1 Syntax for defining ECV service monitors

Figure 4.2 shows a sample ECV statement that defines an ECV service monitor in the **wideip.conf** file.

```
ecv {
  protocol      http
  filename      "/home/user/readme.txt"
  scan_level    all
  user          "jones"
  hashed_password "22AECCCD9CA9C2CC8B"
}
```

Figure 4.2 Sample ECV service monitor definition



5

Load Balancing

- Working with load balancing modes
- Understanding load balancing
- Configuring load balancing
- Changing global variables that affect load balancing
- Troubleshooting manual configuration problems

Working with load balancing modes

Load balancing modes are used by the 3-DNS Controller to distribute the DNS name resolution requests, sent by local DNS servers, to the best available virtual server in your network. This chapter first describes how load balancing works on the 3-DNS Controller, explains the various static and dynamic load balancing modes, and then describes how to configure them.

Understanding load balancing

When the 3-DNS Controller receives a name resolution request from a local DNS server, the controller uses a load balancing mode to select the best available virtual server from a wide IP pool. Once the 3-DNS Controller selects the virtual server, it constructs the DNS answer (either an **A** record for each virtual server IP address, an **NS** record, or a **CNAME** record) and sends the answer back to the requesting client's local DNS server.

The 3-DNS Controller can choose a virtual server from a wide IP pool using either a static load balancing mode, which selects a server based on a pre-defined pattern, or a dynamic load balancing mode, which selects a server based on current performance.

The 3-DNS Controller uses load balancing modes in two situations:

- ◆ **Load balancing among multiple pools**

The 3-DNS Controller supports multiple pools. Configurations that contain two or more pools use a load balancing mode first to select a pool. Once the 3-DNS Controller selects a pool, the controller then uses a load balancing mode to choose a virtual server, within the selected pool. If the controller does not choose a virtual server in the first pool, it applies the load balancing mode to a different pool, either until it selects a virtual server, or all the pools are tried.

- ◆ **Load balancing within a pool**

Within each pool, you specify three different load balancing modes that the controller uses in sequential order: preferred method, alternate method, and fallback method. The *preferred* method is the first load balancing mode that the 3-DNS Controller uses for load balancing. If the preferred method fails, the controller then uses the alternate method for load balancing. If this load balancing mode fails, the controller uses the fallback load balancing mode. If the fallback method fails, the 3-DNS Controller returns the client to standard DNS for resolution.

Table 5.1 shows a complete list of supported load balancing modes, and indicates where you can use each mode in the 3-DNS Controller configuration. The following sections in this chapter describe how each load balancing mode works.

Load Balancing mode	Use for pool load balancing	Use for preferred method	Use for alternate method	Use for fallback method
Completion Rate		X		X
Global Availability	X	X	X	X
Hops		X		X
Kilobytes/Second		X		X
Least Connections		X		X
None		X	X	X
Packet Rate		X	X	X
Quality of Service		X		X
Random	X	X	X	X
Ratio	X	X	X	X
Return to DNS		X	X	X
Round Robin	X	X	X	X
Round Trip Time		X		X
Static Persist		X	X	X
Topology	X	X	X	X
VS Capacity		X	X	X

Table 5.1 Load balancing mode usage

Using static load balancing modes

Static load balancing modes distribute connections across the network according to predefined patterns, and take server availability into account. The 3-DNS Controller supports the following static load balancing modes:

- Static Persist
- Round Robin

- Ratio
- Random
- Global Availability
- Topology
- None
- Return to DNS

The None and Return to DNS load balancing modes are special modes that you can use to skip load balancing under certain conditions. The other static load balancing modes perform true load balancing as described in the following sections.

Static Persist mode

Static Persist mode provides static persistence of local DNS servers to virtual servers; it consistently maps an LDNS IP address to the same available virtual server for the duration of the session. This mode guarantees that certain transactions will be routed through a single transaction manager (for example, a BIG-IP Controller or other server array controller); this is beneficial for transaction-oriented traffic such as e-commerce shopping carts or online trading.

Round Robin mode

Round Robin mode distributes connections in a circular and sequential pattern among the virtual servers in a pool. Over time, each virtual server receives an equal number of connections.

Figure 5.1 shows a sample of the connection distribution pattern for Round Robin mode.



Figure 5.1 Round Robin mode

Ratio mode

Ratio mode distributes connections among a pool of virtual servers as a weighted Round Robin. For example, you can set up Ratio mode to send twice as many connections to a fast, new server, and only half as many connections to an older, slower server.

This load balancing mode requires that you define a ratio weight for each virtual server in a pool, or for each pool if you are using Ratio mode to load balance among multiple pools. The default ratio weight for a server or a pool is set to **1**.

Figure 5.2 shows a sample connection distribution for Ratio mode.



Figure 5.2 Ratio mode

Random mode

Random mode sends connections to virtual servers in a random, uniform distribution pattern. The Random mode is useful for certain test configurations.

Global Availability mode

Global Availability mode uses the virtual servers included in the pool in the order in which they are listed. For each connection request, this mode starts at the top of the list and sends the connection to the first available virtual server in the list. Global Availability mode moves to the next virtual server in the list only when the current virtual server is full or otherwise unavailable. Over time, the first virtual server in the list receives the most connections and the last virtual server in the list receives the least number of connections.

Topology mode

Topology allows you to direct or restrict traffic flow by entering network information into a topology statement in the configuration file. This allows you to develop proximity-based load balancing. For example, client requests in a particular geographic region can be directed to servers within

that same region. The 3-DNS Controller determines the proximity of servers by comparing location information derived from the DNS message to the topology records.

This load balancing mode requires you to do some advanced configuration planning, such as gathering the information you need to define the topology records that determine proximity of client local DNS servers to the various virtual servers. The 3-DNS Controller contains an IP classifier that accurately maps local DNS servers, so when you create topology records, you can refer to continents and countries, instead of IP subnets.

See Chapter 11, *Topology*, for detailed information about working with this and other topology features. For an example configuration using the Topology load balancing mode, see Chapter 3, *Configuring a Globally-Distributed Network*, in the **3-DNS Administrator Guide**.

None mode

The None load balancing mode is a special mode you can use if you want to skip the current load balancing method, or skip to the next pool in a multiple pool configuration. For example, if you set an alternate method to None in a pool, the 3-DNS Controller skips the alternate method and immediately tries the load balancing mode specified as the fallback method. If the fallback method is set to None, and you have multiple pools configured, the 3-DNS Controller uses the next available pool. If you do not have multiple pools configured, the controller returns the connection request to DNS for resolution.

This mode is most useful for multiple pool configurations. For example, you can temporarily remove a specific pool from service by setting each of the methods (preferred, alternate, and fallback) to None. (Note that you can also disable a pool from the Modify Wide IP Pools screen, in the Configuration utility.) You could also use the mode to limit each pool to a single load balancing mode. For example, you would set the preferred method in each pool to the desired load balancing mode, and then you would set both the alternate and fallback methods to None in each pool. If the preferred method fails, the None mode in both the alternate and fallback methods forces the 3-DNS Controller to go to the next pool for a load balancing answer.

Return to DNS mode

The Return to DNS mode is another special load balancing mode that you can use to immediately return connection requests to DNS for resolution. This mode is particularly useful if you want to temporarily remove a pool from service, or if you want to limit a pool in a single pool configuration to only one or two load balancing attempts.

Using dynamic load balancing modes

Dynamic load balancing modes distribute connections to servers that show the best current performance. The performance metrics taken into account depend on the particular dynamic mode you are using.

All dynamic load balancing modes make load balancing decisions based on the metrics collected by the **big3d** agents running in each data center. The **big3d** agents collect the information at set intervals that you define when you set the global timer variables. If you want to use the dynamic load balancing modes, you must run one or more **big3d** agents in each of your data centers, to collect the required metrics.

The 3-DNS Controller supports the following dynamic load balancing modes:

- Completion Rate
- Least Connections
- Packet Rate
- Round Trip Times (RTT)
- Hops
- Kilobytes/Second
- Quality of Service (QOS)
- VS Capacity

Completion Rate mode

Completion Rate mode selects a virtual server that currently maintains the least number of dropped or timed-out packets during a transaction between a data center and the client LDNS.

Figure 5.3 shows a sample connection distribution pattern for Completion Rate mode.

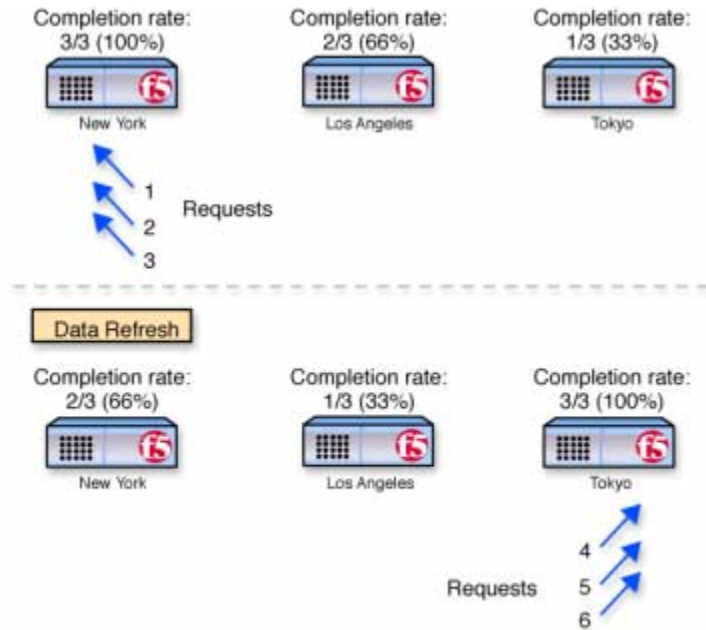


Figure 5.3 Completion Rate load balancing mode

Least Connections mode

Least Connections mode is used for load balancing virtual servers managed by BIG-IP Controllers. Least Connections mode simply selects a virtual server on the BIG-IP Controller that currently hosts the fewest connections.

Packet Rate mode

Packet Rate mode selects a virtual server that is currently processing the fewest number of packets per second.

Figure 5.4 shows a sample connection distribution for Packet Rate mode.

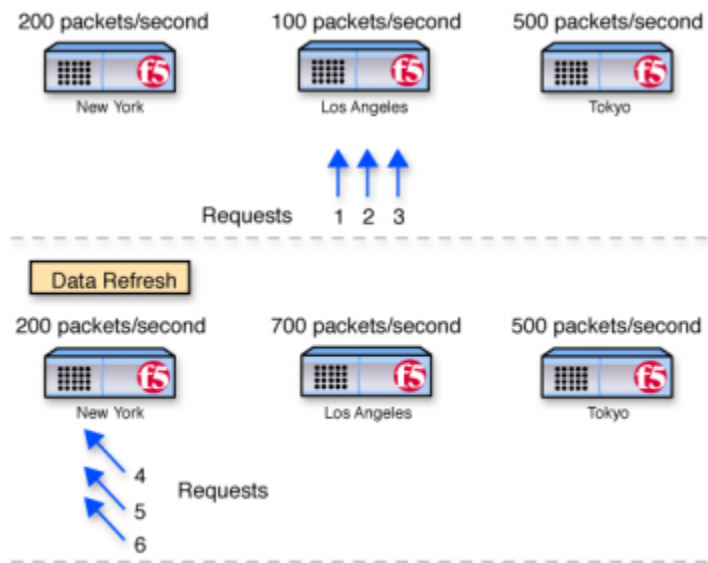


Figure 5.4 Packet Rate mode

Round Trip Times mode

Round Trip Times (RTT) mode selects the virtual server with the fastest measured round trip time between the data center and the client LDNS.

Figure 5.5 shows a sample connection distribution for Round Trip Times mode.

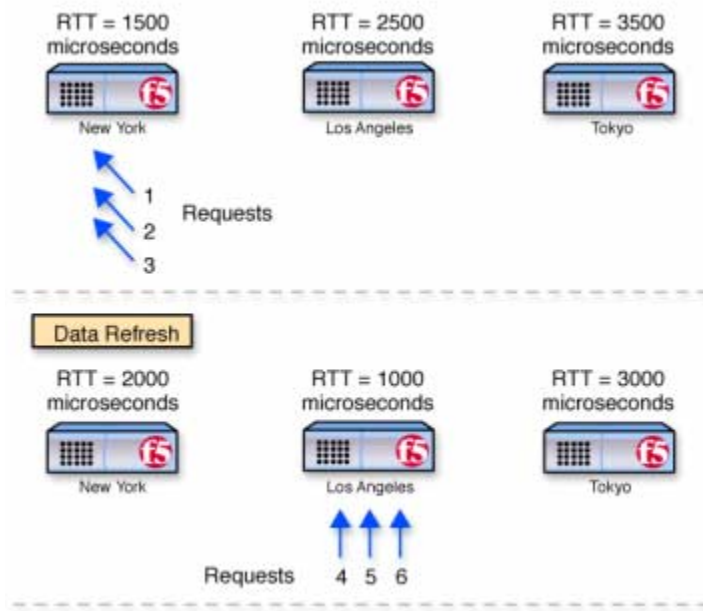


Figure 5.5 Round Trip Times mode

Hops mode

Hops mode is based on the **traceroute** utility, and it tracks the number of intermediate system transitions (hops) between the client LDNS and each data center. Hops mode selects a virtual server in the data center that has the fewest network hops from the LDNS.

Kilobyte/Second mode

Kilobytes/Second mode selects a virtual server that is currently processing the fewest number of kilobytes per second.

Quality of Service mode

Quality of Service mode uses the current performance information, calculates an overall score for each virtual server, and then distributes connections based on each virtual server's score. The performance factors that it takes into account include:

- Round trip time
- Hops
- Completion rate
- Packet rate

- Topology
- VS Capacity
- Kilobytes/Second

Quality of Service mode is a customizable load balancing mode. For simple configurations, you can easily use this mode with its default settings. For more advanced configurations, you can specify different weights for each performance factor in the equation.

You can also configure the Quality of Service load balancing mode to use the dynamic ratio feature. With the dynamic ratio feature turned on, the Quality of Service mode becomes similar to the Ratio mode where the connections are distributed in proportion to ratio weights assigned to each virtual server. The ratio weights are based on the QOS scores: the better the score, the higher percentage of connections the virtual server receives.

For details about customizing Quality of Service mode, see *Setting up Quality of Service (QOS) mode* in Chapter 7 of the **3-DNS Administrator Guide**.

VS Capacity mode

VS Capacity mode selects the virtual server that has the most nodes **up**. If more than one virtual server has the same quantity of nodes **up**, then the 3-DNS Controller load balances using the Random mode among those virtual servers.

Configuring load balancing

This section describes how to configure load balancing on the 3-DNS Controller. You configure load balancing at both the global and wide IP levels:

◆ Global

At the global level, you can configure default settings for the alternate and fallback load balancing methods. Then, if you do not specify alternate or fallback modes when defining a wide IP, the 3-DNS Controller uses the alternate and fallback methods you have configured at the global level. You can find instructions on how to configure global alternate and fallback methods in *Setting global alternate and fallback methods*, on page 5-16.

◆ Wide IP

When defining a wide IP, if you have multiple pools in your wide IP, you first specify which load balancing mode to use in selecting the pool in the wide IP. Next, you specify which preferred, alternate, and fallback load

balancing methods to use in selecting the virtual server within the selected pool. You can find instructions on how to configure these load balancing methods in the section, *Defining a wide IP*, on page 5-12.

Understanding wide IPs

After you configure the BIG-IP Controllers, EDGE-FX Caches, hosts, and the virtual servers they manage, you need to group the configured virtual servers into a wide IP. A **wide IP** is a mapping of a fully-qualified domain name (FQDN) to a set of virtual servers that host the domain content, such as a web site, an e-commerce site, or a CDN.

Before defining the first wide IP, you should do the following:

- ◆ Gather your configuration information for the BIG-IP Controller, EDGE-FX Cache, and host so you can easily see which virtual servers have the content you want to map to an FQDN. Then you can decide how to group virtual servers into pools.
- ◆ Decide which load balancing modes to use for each pool of virtual servers.

◆ Note

NameSurfer, an application included with the 3-DNS Controller, sets up DNS zone files so that wide IP definitions are properly linked to DNS. NameSurfer registers the virtual servers you add to wide IP pools as A records. No action is required on your part, as NameSurfer automatically handles this process. For more information on NameSurfer, see the online help that is included with it (available from the Configuration utility).

There may be situations (for example, e-commerce, and other sites with multiple services) where you need to configure a wide IP so that connections are not sent to a given address unless multiple ports or services are available. You configure this behavior after you define the wide IP. For details, see *Setting up load balancing for services that require multiple ports*, in Chapter 7 of the **3-DNS Administrator Guide**.

Understanding pools

A wide IP contains one or more pool definitions. A **pool** is a group of virtual servers that the 3-DNS Controller load balances. You can include all types of virtual servers (BIG-IP Controller, EDGE-FX Cache, and host) in a pool definition.

Defining a wide IP

After you determine which virtual servers you should place in which wide IP pools, you are ready to add the first wide IP to the configuration.

To define a wide IP using the Configuration utility

1. In the navigation pane, click **Wide IPs**.
The Wide IP List screen opens.
2. On the toolbar, click **Add Wide IP**.
The Add a New Wide IP screen opens.
3. Add the wide IP settings. For help on defining wide IPs, click **Help** on the toolbar.
The wide IP is added to your configuration.

Repeat this process for each wide IP you want to add.

To define a wide IP from the command line

1. At the command prompt, type **3dnsmaint** to open the 3-DNS Maintenance menu.
2. On the 3-DNS Maintenance menu, choose **Edit 3-DNS Configuration** to open the **wideip.conf** file.
3. Add a **wideip** statement.

Place the **wideip** statement after all **server** statements and before any **topology** statement.
4. Under the **wideip** statement, enter the wide IP address, port, and name information. Enclose the wide IP name in quotation marks.
5. Configure any options you want to set (such as the TTL, port list, or QOS coefficients) by entering the appropriate sub-statements.
6. Define the **pool** sub-statement. At the minimum, the **pool** sub-statement should include its name (enclosed in quotation marks) and the virtual servers it contains.
7. Define the load balancing modes you want to use by entering **preferred**, **alternate**, and **fallback** sub-statements.
8. Define the IP address, port, and ratio value for each virtual server that you want to include in this pool.

Figure 5.6 shows the correct syntax for the **wideip** statement.

```

wideip {
  address <ip_addr>
  port <port_number> | <"service name">
  persist < yes | no >
  persist_ttl <number>
  name <"domain_name">
  [ alias <"alias_name"> ]
  [ ttl <number> ]
  [ port_list <port_number> <port_number> ... ]
  [ qos_coeff {
    rtt <n>
    completion_rate <n>
    packet_rate <n>
    topology <n>
    hops <n>
    vs_capacity <n>
    kbps <n>
  } ]
  [ pool_lbmode <rr | ratio | ga | random | topology> ]
  pool {
    name <"pool_name">
    [ limit {
      kbytes_per_second
      pkts_per_second <number>
      current_conns <number>
      cpu_usage <number>
      mem_avail <number>
      disk_avail <number>
    } ]
    [ ratio <pool_ratio> ]
    [ dynamic_ratio < yes | no > ]
    [ rr_ldns < yes | no > ]
    [ preferred < completion_rate | ga | hops | kbps | leastconn | packet_rate | qos |
random | ratio | return_to_dns | rr | rtt | topology | null | vs_capacity |
static_persist> ]
    [ alternate < ga | null | random | ratio | return_to_dns | rr | topology |
vs_capacity | static_persist> ]
    [ fallback <completion_rate | ga | hops | leastconn | null | packet_rate | qos |
random | ratio | return_to_dns | rr | rtt | topology | vs_capacity | static_persist> ]
    address <vs_addr>[:<port>] [ratio <weight>]
    address <vs_addr>[:<port>] [ratio <weight>]
    address <vs_addr>[:<port>] [ratio <weight>]
    ...
  }
}

```

Figure 5.6 Syntax for the **wideip** statement

An example of the wideip statement

Figure 5.7 shows a sample **wideip** statement. This statement defines a wide IP named **mx.wip.domain.com**, with an alias of **mail.wip.domain.com**. The wide IP contains two pools, with **pool_1** receiving three times as many requests as **pool_2**. The 3-DNS Controller attempts to resolve requests sent to **pool_1** using the Round Trip Times (RTT) mode. This mode sends connections to the virtual server in the pool that demonstrates the best round trip time between the virtual server and the client LDNS. If the 3-DNS Controller cannot resolve the request using the RTT mode, the controller distributes requests using the Random load balancing mode. The 3-DNS Controller distributes requests at a 2:1 ratio to the two virtual servers defined in **pool_2**, where the first listed virtual server receives twice as many connections as the second.

```
wideip {
  address      192.168.102.50
  service      "smtp"
  name         "mx.wip.domain.com"
  alias        "mail.wip.domain.com"
  pool_lbmode  ratio
  pool {
    name       "pool_1"
    ratio      3
    preferred  rtt
    alternate  random
    address    192.168.101.50
    address    192.168.102.50
    address    192.168.103.50
  }
  pool {
    name       "pool_2"
    ratio      1
    preferred  ratio
    address    192.168.104.50    ratio 2
    address    192.168.105.50    ratio 1
  }
}
```

Figure 5.7 Example syntax for defining a wide IP

Using the LDNS round robin wide IP attribute

LDNS round robin is an attribute that you can use in conjunction with any load balancing mode. The LDNS round robin attribute allows the 3-DNS Controller to return a list of available virtual servers, instead of a single virtual server. Certain browsers keep the answer returned by DNS servers. By enabling this attribute, the 3-DNS Controller returns a maximum of 16 virtual servers as the answer to a DNS resolution request. This provides browsers with alternate answers if a virtual server becomes unavailable.

Using the last resort pool designation

The last resort pool is an optional setting for a wide IP pool. The wide IP pool that you designate as the last resort pool, in the Configure Load Balancing for New Pool screen, is the virtual server pool that the 3-DNS Controller uses when all other pools have reached their thresholds or are unavailable for any reason. The 3-DNS Controller uses the last resort pool only when it tries, unsuccessfully, to load balance to all other configured pools.

When your network includes cache appliances hosting content from an origin site, you can designate the origin site as the last resort pool to handle requests if your cache virtual servers have reached their thresholds. You can also use the last resort pool to designate an overflow network so your origin servers remain available if network traffic spikes. You can only designate one last resort pool within a wide IP.

To designate a last resort pool using the Configuration utility

1. In the navigation pane, select **Wide IPs**.
The Wide IP List screen opens.
2. From the Pools column, select the pools for the wide IP for which you want to create a last resort pool.
The Modify Wide IP Pools screen opens.
3. From the Pool Name column, click the pool that you want to designate as the last resort pool.
The Modify Load Balancing for [pool name] screen opens.
4. Check the box next to **Last Resort Pool**, and click **Update**.

To designate a last resort pool from the command line

In the `wideip.conf` file, change the `last_resort` definition from `no` to `yes` for the pool that you want to designate as the last resort pool. Figure 5.8 shows an example of a last resort pool definition.

```
pool {
  name "origin"
  last_resort yes
  preferred kbps
  alternate rr
  fallback return_to_dns
  address 192.168.103.5
  address 192.168.103.6
  address 192.168.103.7
}
```

Figure 5.8 Example of a last resort pool definition

Changing global variables that affect load balancing

You can configure global variables that affect how load balancing is handled on a global basis for all wide IPs managed by the 3-DNS Controller. You can override these global settings for individual wide IPs as necessary.

Global variables that affect load balancing fall into two categories:

- Alternate and fallback load balancing methods
- TTL (time to live) and timer values

The default settings for these variables are adequate for most configurations. However, if you want to change any global variable, you should refer to the online help.

Setting global alternate and fallback methods

You can configure a load balancing method that all wide IPs can use in the event that their preferred method fails.

To configure global alternate and fallback load balancing methods using the Configuration utility

1. In the navigation pane, click **System**.
The System - General screen opens.
2. On the toolbar, click **Load Balancing**.
3. In the **Default Alternate** box, select the load balancing mode to use should a wide IP's preferred method fail.
4. In the **Default Fallback** box, specify the load balancing mode to use should the preferred and alternate methods fail.
If all methods fail, requests are returned to DNS.
5. Finish configuring the rest of the settings on the System - Load Balancing screen. (For help on configuring the load balancing settings, click **Help** on the toolbar.)
The global load balancing settings are added to your configuration.

To configure global alternate and fallback load balancing methods from the command line

1. At the command prompt, type **3dnsmaint** to open the 3-DNS Maintenance menu.
2. On the 3-DNS Maintenance menu, choose **Edit 3-DNS Configuration** to open the **wideip.conf** file.

3. Locate or add the **globals** statement. The **globals** statement should be at the top of the file.
4. Use the syntax shown in Figure 5.9 to define global alternate and fallback load balancing methods.

```
globals {
  [ default_alternate < ga | leastconn | null | packet_rate | random | ratio |
  return_to_dns | rr | topology | static_persist | vs_capacity > ]
  [ default_fallback < completion_rate | ga | hops | leastconn | null | packet_rate |
  qos | random | ratio | return_to_dns | rr | rtt | topology | static_persist |
  vs_capacity> ]
}
```

Figure 5.9 Configuring global alternate and fallback load balancing modes

Figure 5.10 shows a sample **globals** statement that defines global load balancing variables.

```
globals {
  default_alternate leastconn
  default_fallback rr
}
```

Figure 5.10 Sample syntax for setting global load balancing variables

Understanding TTL and timer values

Each 3-DNS object has an associated *time-to-live (TTL)* value. A TTL is the amount of time (measured in seconds) for which metrics information is considered valid. The timer values determine how often the 3-DNS Controller refreshes the information.

Table 5.2 describes each TTL value, as well as its default setting.

Parameter	Description	Default
Server TTL	Specifies the number of seconds that the 3-DNS Controller uses BIG-IP Controller and EDGE-FX Cache metrics information for name resolution and load balancing.	60
Host TTL	Specifies the number of seconds that the 3-DNS Controller uses generic host machine metrics information for name resolution and load balancing.	240

Table 5.2 TTL values and default settings

Parameter	Description	Default
3-DNS TTL	Specifies the number of seconds that the 3-DNS Controller considers performance data for the other 3-DNS Controllers to be valid.	60
Virtual server TTL	Specifies the number of seconds that the 3-DNS Controller uses virtual server information (data acquired from a BIG-IP Controller, EDGE-FX Cache, or other host machine about a virtual server) for name resolution and load balancing.	120
Hops TTL	Specifies the number of seconds that the 3-DNS Controller considers traceroute data to be valid.	604800 (seven days)
Path TTL	Specifies the number of seconds that the 3-DNS Controller uses path information for name resolution and load balancing.	2400
Default TTL	Specifies the default number of seconds that the 3-DNS Controller considers the wide IP A record to be valid. If you do not specify a wide IP TTL value when defining a wide IP, the wide IP definition uses the default_ttl value.	30

Table 5.2 *TTL values and default settings*

Each 3-DNS object also has a timer value. A timer value defines the frequency (measured in seconds) at which the 3-DNS Controller refreshes the metrics information it collects. In most cases, the default values for the TTL and timer parameters are adequate. However, if you make changes to any TTL or timer value, keep in mind that an object's TTL value must be greater than its timer value.

Table 5.3 describes each timer value, as well as its default setting.

Parameter	Description	Default
Server data refresh	Specifies the frequency (in seconds) at which the 3-DNS Controller refreshes BIG-IP Controller and EDGE-FX Cache information.	20
Host data refresh	Specifies the frequency (in seconds) at which the 3-DNS Controller refreshes other host machine information.	90
3-DNS data refresh	Specifies the frequency (in seconds) at which the 3-DNS Controller retrieves performance data for other 3-DNS Controllers in the sync group.	20
Virtual server data refresh	Specifies the frequency (in seconds) at which the 3-DNS Controller refreshes virtual server information.	30
ECV timer refresh	Specifies the frequency (in seconds) at which the 3-DNS Controller refreshes the ECV monitor.	90

Table 5.3 *Time values and default settings*

Parameter	Description	Default
Hops data refresh	Specifies the frequency (in seconds) at which the 3-DNS Controller retrieves traceroute data (traceroutes between each data center and each local DNS).	60
Path data refresh	Specifies the frequency (in seconds) at which the 3-DNS Controller refreshes path information (for example, round trip time or ping packet completion rate).	120
Remote nodes query	Specifies the frequency (in seconds) at which the 3-DNS Controller queries remote 3-DNS Controllers and BIG-IP Controllers.	60
3-DNS Sync Time Tolerance	Specifies the number of seconds that one 3-DNS Controller's time setting is allowed to be out of sync with another 3-DNS Controller's time setting. Note: If you are using NTP to synchronize the time of the 3-DNS Controller with a time server, leave the time tolerance at the default value of 10 . In the event that NTP fails, the 3-DNS Controller uses the time_tolerance variable to maintain synchronization.	10
Timer Sync State	Specifies the interval (in seconds) at which the 3-DNS Controller checks to see if it should change states (from Principal to Receiver or from Receiver to Principal).	30
Persist Cache	Specifies the interval (in seconds) at which the 3-DNS Controller archives the paths and metrics data.	3600

Table 5.3 Time values and default settings

To configure global TTL and timer values using the Configuration utility

1. In the navigation pane, click **System**.
The System - General screen opens.
2. To configure the default TTL for wide IPs, type a new value in the **Default TTL** box.
3. To configure other TTL and timer values, click **Timers and Task Intervals** on the toolbar.
The System - Timers & Task Intervals screen opens.
4. Add the TTL and timer values settings.

For help on configuring the TTL and timer values settings, click **Help** on the toolbar.

To configure global TTL and timer values from the command line

1. At the command prompt, type **3dnsmaint** to open the 3-DNS Maintenance menu.
2. On the 3-DNS Maintenance menu, choose **Edit 3-DNS Configuration** to open the **wideip.conf** file.
3. Locate or add the **globals** statement. The **globals** statement should be at the top of the file.
4. Use the syntax shown in Figure 5.11 to define global TTL and timer values.

```
globals {  
  [ timer_get_3dns_data <number> ]  
  [ timer_get_server_data <number> ]  
  [ timer_get_host_data <number> ]  
  [ timer_get_vs_data <number> ]  
  [ timer_get_ecv_data <number> ]  
  [ timer_get_path_data <number> ]  
  [ timer_get_trace_data <number> ]  
  [ timer_check_keep_alive <number> ]  
  [ timer_check_pending_q_timeouts <number> ]  
  [ timer_persist_cache <number> ]  
  [ timer_sync_state <number> ]  
  [ 3dns_ttl <number> ]  
  [ server_ttl <number> ]  
  [ host_ttl <number> ]  
  [ vs_ttl <number> ]  
  [ path_ttl <number> ]  
  [ trace_ttl <number> ]  
  [ default_ttl <number> ]  
}
```

Figure 5.11 Syntax for configuring global TTL and timer values

Troubleshooting manual configuration problems

Adding a wide IP requires careful planning and use of correct syntax. We recommend using the Configuration utility to create wide IPs and pools so that the correct syntax is generated automatically in the **wideip.conf** file. However, we have included the following recommendations to make it easier for you to spot and resolve any configuration problems if you choose to create your configuration by editing the **wideip.conf** file.

- ◆ **Configuration utility**

The Configuration utility contains Statistics screens that are useful in diagnosing problems, as they provide a snapshot of the 3-DNS Controller

network at any given time. To use them, expand the **Statistics** item in the navigation pane, then click either **Wide IPs** or **Summary** (and scroll until you see the **Wide IP** table).

The Configuration utility also contains the Network Map, which allows you to see the relationships between your data centers, servers, and virtual servers, and the wide IPs and pools you created with the virtual servers. For information on working with the Network Map, click **Help** on the toolbar.

◆ **wideip.conf syntax**

If you configure wide IPs from the command line, use the **3dparse** utility to verify **wideip.conf** syntax before you start **3dnsd**. To use this utility, type **3dparse** on the command line. For details on the **3dparse** utility, see the **3dparse** man page.

◆ **/var/log/messages**

If you encounter an error that you cannot trace, you can view the log file in the Configuration utility, or you can directly open the **/var/log/messages** file on your system. Using the UNIX **grep** utility, search for **3dnsd** (for example, **tail -100 /var/log/messages | grep 3dnsd**). This log file saves verbose error information, and should contain an explanation of the error.

◆ **BIND syntax**

If you are setting up the configuration from the command line, you may want to refer to one of the following BIND resources for help and background information:

- The O'Reilly & Associates book, *DNS and BIND*, Third Edition
- <http://www.isc.org/bind.html>



6

Network Map

- Introducing the Network Map

Introducing the Network Map

The 3-DNS Controller Network Map is a dynamic map that illustrates the physical and logical objects in your network. With the Network Map, you can:

- Visualize the overall structure of your 3-DNS Controller network
- Use the navigational tools to modify your network configuration
- View the enabled/disabled state of the various objects in your network

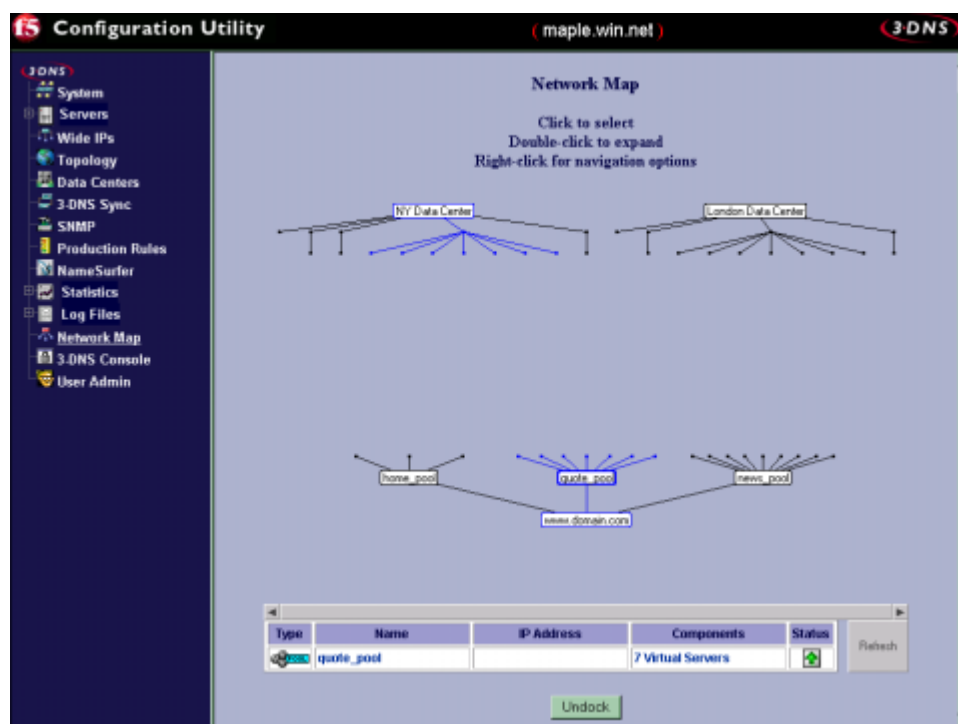


Figure 6.1 Example screen of the Network Map in the Configuration utility

In the Network Map, you can easily see how any component is related to the rest of the network, and how changes to the physical side of the network structure (for example, data centers or servers) can affect the logical side (for example, wide IPs or pools), and vice versa. As shown in Figure 6.1, the wide IP pool, **quote_pool**, is made up of virtual servers on a BIG-IP Controller in the data center, **NY Data Center**.

Working with the Network Map

The Network Map is a highly interactive screen. You can not only review and make changes to your 3-DNS Controller configuration, but you can also use the information table to quickly check whether an object is enabled or disabled. The following sections describe some of the tasks you can do in the Network Map.

Viewing the Network Map

You can view the Network Map only from the Configuration utility.

To view the Network Map using the Configuration utility

1. In the navigation pane, click **Network Map**.
The Network Map screen opens.
2. Click **Undock** if you want to open a popup screen of the Network Map. For more information on working with the Network Map, click **Help** on the toolbar.

Using the Network Map to review and modify the network configuration

The Network Map contains the following objects: data centers, servers, wide IPs, pools, virtual servers. You can double-click any object on the Network Map to expand the object. The relationship of that object to the rest of the network becomes readily apparent, as the components of that object are highlighted in blue throughout the map. For example, if you double-click a data center, the data center expands, displaying and highlighting all of the servers that reside in that data center. Toward the bottom of the map, all wide IPs that contain a virtual server that belongs to the servers in the selected data center are also highlighted. You can continue to double-click the objects to narrow your scope.

From the Network Map, you can also navigate to the screens where you configure the various objects. You do this by right-clicking the object name. A popup menu opens, displaying various options from which you can choose, depending on what part of that object you want to configure. For example, if you right-click a wide IP name, and from the popup menu select **Configure**, the Modify Wide IP screen opens, where you can modify the settings for the wide IP definition.

Using the information table on the Network Map

When you double-click any object on the Network Map, the information table at the bottom of the Network Map screen displays the following details about that object:

- Object type
- Object name
- Object IP address
- Any child objects for the highlighted object
- Object status

You can also refresh the Network Map by clicking the Refresh button next to the information table.



7

Production Rules

- Controlling network traffic patterns with production rules
- Setting up production rules in the Configuration utility
- Working with the production rules scripting language

Controlling network traffic patterns with production rules

Production rules are a policy-based management tool that you can use to dynamically change how the 3-DNS Controller distributes connections across the network. You can also use production rules to send system administrators notifications of specific events. Production rules are based on triggers, such as time of day, current traffic patterns, or current traffic volume. For example, you can configure a production rule that changes the load balancing mode to QOS during your peak business hours, and you can configure a production rule that notifies you when the number of name resolution requests exceeds a specific number.

You can create production rules that apply to the system in general, or you can create production rules for specific wide IPs.

If you want to configure basic production rules, we recommend that you use the Configuration utility. If you want to create custom production rules, you should review the following section, *Working with the production rules scripting language*, on page 7-5, which describes the scripting language you use to configure production rules from the command line. You may also want to contact a technical support engineer for additional assistance with complex configurations.

Setting up production rules in the Configuration utility

The Configuration utility uses a wizard-style format to help you set up production rules. The screen prompts that you see during the configuration process vary, depending on the items you select in each screen. However, to configure any production rule, you perform three basic steps:

- ◆ **Define the type of rule**

The two types of production rules are: global production rules, and wide IP production rules.

- ◆ **Define the rule trigger**

The two types of rule triggers are: a set time or time interval, and a specific system event.


- ◆ **Define the action taken**

The two basic types of rule actions are: to send user-definable messages to log files or email accounts, and to change specific load balancing settings.

The following sections discuss each production rule option in detail, and provide all of the information you need to complete the production rule using the wizard.

Viewing, adding, and deleting production rules

When you click **Production Rules** in the Configuration utility, the Production Rules wizard screen opens. The screen displays the list of existing global and wide IP production rules. You can add a new rule by clicking the **Add Production Rule** toolbar button, which starts the production rule wizard. The wizard prompts you to specify the various production rule options, and then allows you to review your selections before you save the production rule to the configuration.

Note that you can modify existing production rules by clicking the rule name in the list, and you can delete a production rule at any time by clicking the Delete button  next to the rule name.

Choosing the rule type

The first step in the production rule wizard is to choose whether the production rule is a global production rule or a wide IP production rule.

- ◆ **Global production rules**

Global production rules send messages to log files or to specific email accounts, based on a set time interval or on standard events. The standard events are listed and described in Table 7.2, on page 7-9.

- ◆ **Wide IP production rules**

Wide IP production rules are based either on the time of day, or on standard events. Wide IP production rules can change the current load balancing modes for the preferred, alternate, or fallback methods; they can reconfigure ratio settings for individual virtual servers; and they can reconfigure the coefficients for Quality of Service mode. Wide IP production rules can also send messages to log files or email accounts.

After you choose a rule type, the wizard prompts you to name the rule and allows you to add a brief description of the rule.

Defining time-based triggers

The next step in the wizard prompts you to choose a trigger for the production rule. You can set up two basic types of triggers: time-based triggers and event-based triggers. This section describes the options for the time-based triggers, and the following section describes options for the event-based triggers. Once you review the information for the type of trigger you want to set up, go to *Choosing the action taken*, on page 7-4.

Time-based triggers include two types: global production rules trigger on set time intervals, while wide IP production rules trigger at specific times on specific days. To set a time interval for a global production rule, you define the number of seconds that elapse between each action the production rule executes.

A wide IP production rule can trigger at a specific time of day, on a specific day of the week, on a specific date, or at a specific time on a specific date. The following procedures explain how to set up each type of time trigger for wide IP production rules.

To apply a time of day variable

1. From the Time Variable table, select **Time**.
2. In the **Start Time** box, specify the hour and minute you want the production rule action to begin.
3. In the **Stop Time** box, select the hour and minute you want the production rule action to stop.

Once you define the time of day that triggers the production rule, you continue with the wizard and begin to define the production rule action.

To apply a day of the week variable

1. From the Time Variable table, select **Day**. A table opens from which you select the day to start and stop the action.
2. From the **Start Day** box, select the day you want the production rule action to begin.
3. From the **Stop Day** box, select the day you want the production rule action to stop.

Once you define the day of the week that triggers the production rule, you continue with the wizard and begin to define the production rule action.

To apply a date variable

1. From the Time Variable table, select **Date**. A table opens from which you select the date to start and stop the action.
2. In the **Start Date** box, type the date you want the production rule action to begin (mm/dd/yyyy).
3. In the **Stop Date** box, type the date you want the production rule action to stop (mm/dd/yyyy).

Once you define the date that triggers the production rule, you continue with the wizard and begin to define the production rule action.

To apply a combined date and time variable

1. From the Time Variable table, select **Date/Time**.
Two tables open and you select the start and stop dates and times.
2. In the **Start Date** box, type the date you want the production rule action to begin (mm/dd/yyyy).
3. In the **Stop Date** box, type the date you want the production rule action to stop (mm/dd/yyyy).
4. In the **Start Time** box, specify the hour and minute you want the production rule action to begin.
5. In the **Stop Time** box, select the hour and minute you want the production rule action to stop.

Once you define the date and time that triggers the production rule, you continue with the wizard and begin to define the production rule action.

Defining event-based triggers

Both global production rules and wide IP production rules can be triggered by standard events, such as when a name resolution process begins. Wide IP production rules support two additional types of event-based triggers. You can set a wide IP production rule to trigger when a specific LDNS server makes a name resolution request, or to trigger when a user-specified number of name resolution requests are received by the 3-DNS Controller.

The standard events that can trigger both global and wide IP production rules are described in Table 7.2, on page 7-9.

Choosing the action taken

After you specify the production rule trigger, the wizard prompts you to choose the action that the production rule takes. Note that the actions that a production rule can take depend in part on whether the production rule is a global rule or a wide IP rule. For example, both global production rules and wide IP production rules can send user-defined messages to log files, or to specific email accounts, but only wide IP production rules can alter load balancing modes. The actions that you can choose for a production rule are:

- ◆ **Sending user-defined messages**
Both global and wide IP production rules can send user-defined messages to the **syslog** file, or to a specific email account.
- ◆ **Changing the load balancing mode settings**
Wide IP production rules can change load balancing mode settings for the wide IP. You can change the preferred, alternate, and fallback methods, and you can change QOS coefficient settings.
- ◆ **Changing virtual server ratios**
You can change virtual server ratios to alter the distribution load when the load balancing mode is set to Ratio.
- ◆ **Specifying a virtual server to return**
You can specify that the 3-DNS Controller returns a specific virtual server, rather than choosing a virtual server using load balancing.

Once you specify an action, the production rules wizard prompts you to review all of the production rule settings, and then saves the production rule to the configuration.

Working with the production rules scripting language

The production rules scripting language uses constructs and statements that are similar in syntax to Perl script and the C programming language. If you have a good working knowledge of Perl or C, you may want to create your own custom production rules. You can use the guidelines in this section in conjunction with the examples provided both here and in the sample **wideip.conf** file (installed on the 3-DNS Controller).

If you need to add custom production rules to your configuration, but you do not want to work out the implementation yourself, you can contact your vendor for assistance.

Inserting production rules in the wideip.conf file

Production rules are part of the **wideip.conf** file, and you can either insert them directly in the file, or you can store them in a separate file and include them by reference. If you want to use the Configuration utility to manage the 3-DNS Controller configuration, you must store production rules configured from the command line in a separate file, and include them by

reference. If you attempt to use custom production rules in a file that you edit using the Configuration utility, the production rule syntax may become corrupt.

◆ WARNING

*If you include custom production rules directly in the **wideip.conf** file, you must edit and maintain the **wideip.conf** file from the command line; you cannot use the Configuration utility for configuration administration.*

Executing and managing production rules

The **3dscrip** utility requires that production rules have the following attributes:

- Each production rule is uniquely identified by a label.
- Each production rule can be deleted using its label.
- All production rules at the global scope can be deleted.
- All production rules at the wide IP-pool scope can be deleted.
- Each production rule can be replaced.
- Each production rule can be annotated with a character string.

The **3dscrip** utility manages and executes production rules according the following guidelines:

- The **3dscrip** utility supports conditional execution of production rules using the **if** statement. You can use **if** statements in wide IP production rules, and in global production rules only if they are embedded within a **when** or an **every** statement.
- The **3dscrip** utility supports event-driven execution of production rules using the **when** statement. You can use the **when** statement only in global production rules.
- The **3dscrip** utility supports periodic execution of production rules using the **every** statement. You can use the **every** statement only in global production rules.

The following sections describe how to work with the **3dscrip** utility's components.

Working with the if statement

```
if(conditional-expression) { <action> ... } [ else { <action> ... } ]
```

The **if** statement is a standard statement that defines an event condition that triggers a production rule action. Typically you use **if** statements in wide IP production rules. An **if** statement must adhere to the following guidelines:

- The **if** statement can be specified in the scope of a wide IP **pool** statement.
- The **if** statement can be nested in another **if** statement.
- Multiple **if** statements can be specified in the same scope.
- The nesting of **if** statements is limited only by the memory capacity of the 3-DNS Controller.
- The precedence of logical, relational, and unary operators is the same as in ANSI-c.

If statement parameters and operators	Can contain or be one of these:
conditional-expression	A primitive-expression A primitive-expression followed by a relational-operator, followed by a primitive-expression A primitive-expression followed by an arithmetic-operator, followed by a primitive-expression Two conditional-expressions joined by a logical-operator
primitive-expression	A keyword which is evaluated when the conditional-expression is evaluated An intrinsic function which is evaluated when the conditional-expression is evaluated A literal value enclosed in full quotes A conditional-expression enclosed in parentheses A unary-operator followed by a conditional-expression enclosed in parentheses
logical-operators	Logical OR () Logical AND (&&)

Table 7.1 Components of the **if** statement

If statement parameters and operators	Can contain or be one of these:
relational-operators	Equality (==) Not equal (!=) Greater than (>) Greater than or equal to (>=) Less than (<) Less than or equal to (<=)
arithmetic-operator	modulus (mod)
unary operators	Unary negation (!) Unary minus (-)
keywords	day, time, date, datetime, ldns_ip, wip_ip, wip_name, wip_num_resolves, preferred, alternate, fallback, rtt, completion_rate, hops, packet_rate, topology
intrinsic functions	isLdnsInNet (Ip address, mask) isLdnsInAS (IP address, mask)

Table 7.1 Components of the if statement

Working with the when statement

```
when(event) { <action> ... }
```

The **when** statement is a standard statement that defines a specific event condition that triggers a production rule action. A **when** statement can be used only in global production rules, and it must adhere to the following guidelines:

- ◆ The **when** statement can be specified at the top scope of **wideip.conf**, after the **wide IP** definition(s) and before the **topology** statement.
- ◆ Multiple **when** statements can be specified in the same scope.
- ◆ Nesting of **when** statements is not allowed.

The production rule event triggers are described in Table 7.2.

Event triggers	Description
ResolveNameBegin	The production rule takes action each time the 3-DNS Controller receives a new resolution request.
ResolveNameEnd	The production rule takes action each time the 3-DNS Controller completes a name resolution.
FallbackToStatic	The production rule takes action each time the fallback load balancing method is used in a wide IP.
SIGINT	The production rule takes action each time the 3-DNS Controller receives a SIGINT command.
SIGHUP	The production rule takes action each time the 3-DNS Controller receives a SIGHUP command.
ReapPaths	The production rule takes action each time the 3-DNS Controller reaps obsolete path information.
CRC_Failure	The production rule takes action each time iQuery communication on the 3-DNS Controller experiences a CRC failure.
DownServer	The production rule takes action each time the 3-DNS Controller detects that another 3-DNS Controller, BIG-IP Controller, or host server becomes unavailable.
DownVS	The production rule takes action each time the 3-DNS Controller detects that a virtual server becomes unavailable.
DoneINT	The production rule takes action after the wideip.conf file is read on startup (a one-time event).
DoneConfigFile	The production rule takes action each time the 3-DNS Controller configuration is re-read (for example, when a 3ndc reload command is issued).

Table 7.2 Standard production rule event triggers

Working with the every statement

```
every(<seconds>) { <action> ... }
```

The **every** statement is a standard statement that defines a time interval at which the production rule action triggers, such as every 60 seconds. An **every** statement can be used only for a global production rule, and it must adhere to the following guidelines:

- The **every** statement can be specified at the top scope of the **wideip.conf** file, after the wide IP definition(s) and before the **topology** statement.
- Multiple **every** statements can be specified in the same scope.
- Nesting of **every** statements is not allowed.

Defining production rule actions

The production rules language supports the following actions. Not all actions apply to all production rule types. For example, the actions that change load balancing settings are valid only for wide IP production rules. Actions such as defining a log string can be used in either global production rules or wide IP production rules. Each action below specifies which production rule types can use it.

Production rule actions	Description	Production rule type
preferred <lbmode>	This action changes the preferred load balancing method in a wide IP.	Wide IP production rule only
alternate <lbmode>	This action changes the alternate load balancing method in a wide IP.	Wide IP production rule only
fallback <lbmode>	This action changes the fallback load balancing method in a wide IP.	Wide IP production rule only
log (<string>)	This action sends the specified string to the syslog utility, which writes the string to the syslog file.	Wide IP production rule Global production rule
log2mail (<string>)	This action sends the specified string to the Sendmail utility, which creates a mail message and forwards it to the administrative email account specified for Sendmail (see the log2mail man page for details about log2mail syntax).	Wide IP production rule Global production rule
vs (<ip>:<port>).ratio <n>	This action changes the ratio setting for a specific virtual server in a wide IP pool.	Wide IP production rule only
return_vs (<ip:port>)	This action skips the load balancing process and instead returns the specified virtual server to the requesting client. .	Wide IP production rule only

Table 7.3 Descriptions of production rule actions

Production rule examples

There are a variety of custom production rules that you may want to implement or expand on for your own network. Following are examples of these three custom production rules:

- Load balancing according to time of day
- Load balancing according to LDNS
- Hacker detection

Using production rules to load balance according to time of day

You can set up production rules ahead of time to deal with future needs and client demands for events. For example, say your company has a software distribution scheduled for release next Tuesday at 5:00 p.m. Pacific Standard Time. The new software will be available for download from the FTP sites at that time, and you expect that during the first week, traffic will be 10 times what it normally is, with frequent bursts during standard work hours, 7 a.m. to 6 p.m. However, the client base spans four time zones with an FTP server farm on the east coast in New York (**192.168.101.50**), and another on the west coast in Los Angeles (**192.168.102.50**). The 3-DNS Controller is located on the east coast and runs on Eastern Standard Time. You are willing to accept some network latency in return for guaranteed connections.

Figure 7.1 shows a sample production rule that handles the connections according to the anticipated load at specific times of the day.

```
wideip {
  address 192.168.101.50:21
  name "ftp.domain.com"
  pool {
    preferred ratio
    address 192.168.101.50 ratio 2
    address 192.168.102.50 ratio 1
    rule "ftp_balance"
      // Night time: qos
      if(time > "21:00" && time < "07:00") {
        preferred leastconn
      }
      else {
        preferred ratio
        // East Coast
        rule "east" if(time < "10:00") {
          vs.(192.168.101.50).ratio 3
          vs.(192.168.102.50).ratio 1
        }
        // Both coasts are at peak demand
        else {
          rule "both" if(time < "18:00") {
            vs.(192.168.101.50).ratio 1
            vs.(192.168.102.50).ratio 1
          }
          // West Coast
          else {
            vs.(192.168.101.50).ratio 1
            vs.(192.168.102.50).ratio 3
          }
        }
      }
  }
}
```

Figure 7.1 Load balancing by time of day

Using production rules to load balance according to LDNS

One interesting application of production rules is when you create a rule that triggers based on a specific LDNS server making a name resolution request. The following example is based on a web site published in three languages: English, Spanish, and Japanese. Suppose that the addresses in the network **10.10.0.0** are allocated to Japanese speakers, and the addresses in the

network **10.11.0.0** are allocated to Spanish speakers. The production rule shown in Figure 7.2 uses the address of the requesting LDNS to determine which virtual server should receive the connection.

```
wideip {
  address 192.168.101.50:80
  name "www.domain.com"
  pool {
    rule "Japanese" if(isLdnsInNet(10.10.0.0, 255.255.0.0)) {
      return_vs(192.168.103.50:80)
    }
    else {
      rule "Spanish" if(isLdnsInNet(10.11.0.0, 255.255.0.0)) {
        return_vs(192.168.102.50:80)
      }
      else { // assume English
        return_vs(192.168.101.50:80)
      }
    }
  }

  address 192.168.101.50 // English
  address 192.168.102.50 // Spanish
  address 192.168.103.50 // Japanese
}
}
```

Figure 7.2 Load balancing by IP address of LDNS

Using production rules for hacker detection

Another interesting example of triggering a production rule based on the requesting LDNS server is to take evasive action against known hackers attempting to access your system. The production rule shown in Figure 7.3 sends the hacker to a special server, rather than flat out rejecting the connection. As an alternative, you could change the rule to return a non-routable or non-existent address.

```
when(ResolveNameBegin) {
  rule "roach_motel" if(isLdnsInNet(10.20.30.4, 255.255.255.0)) {
    // Send this guy to our "roach motel" for hackers.
    // This address doesn't need to be listed in any wideip pool.
    // This address is reserved for us to watch hackers under the microscope.
    log2mail("Hacker $ldns_ip came back")
    return_vs(192.168.1.46:80)
  }
}
```

Figure 7.3 Sending a hacker to a specific server

A related example, shown in Figure 7.4, illustrates a production rule that deals with attacks against iQuery communications. The production rule would warn you if the 3-DNS Controller detected a hack attempt against the iQuery protocol, based on a communication failure.

```
Rule "iQuery_hacked" when(CRC_Failure) {  
    log2mail("Got CRC Failure")  
}
```

Figure 7.4 *Detecting an iQuery failure due to potential attack*



8

Resource Records

- Understanding resource records
- Types of resource records
- Additional resource record types

Understanding resource records

A *resource record* is an entry in a DNS database file and consists of a name, a TTL, a type, and data that is specific to the type. These resource records, in a hierarchical structure, make up the domain name system (DNS).

The standard resource record format, specified in RFC 1035, is as follows:

```
{name} {ttl} addr-class record type record-specific data
```

The resource record fields are defined as follows:

◆ **name**

The first field, **name**, is the name of the domain record and it must always start in column 1. For all resource records that are not the first in a file, the name may be left blank. When the name field is left blank, the record takes the previous resource record.

◆ **ttl**

The second field, **ttl** (time to live), is optional. This field specifies how long the resource record is stored by the LDNS. If this field is left blank, the default time to live value is specified in the start of authority (SOA) resource record (described later in this chapter).

◆ **address class**

The third field is the address class. Currently, only one class is supported: **IN**, for internet addresses and other internet information. Limited support is included for the **HS** class, which is for MIT/Athena "Hesiod" information.

◆ **record type**

The fourth field, record type, defines the type of this resource record, such as **A**, or **NS**.

◆ **other fields**

Additional fields may be present in a resource record, depending on its type.

Although case is preserved in names and data fields when loaded into the name server, comparisons and lookups in the name server database are not case-sensitive.

Types of resource records

There are many types of resource records currently in use. This section provides an overview of the most common resource record types, and lists other types of resource records. The six most common types of resource records are shown in Table 8.1.

Type	Description
A (Address)	Maps host names to IP addresses.
CNAME (Canonical Name)	Defines a host alias.
MX (Mail Exchange)	Identifies where to send mail for a given domain name.
NS (Name Server)	Identifies a domain's name servers.
PTR (Pointer)	Maps IP addresses to host names.
SOA (Start of Authority)	Indicates that a name server is the best source of information for a zone's data; defines default parameters for a zone.

Table 8.1 Common resource records

A (Address)

The Address record, or **A** record, lists the IP address for a given host machine name. The name field is the host's name, and the address is the network interface address. There should be one **A** record for each IP address of the machine.

Figure 8.1 shows an example of an **A** record.

{name}	{ttl}	addr-class	{type}	address
host1.domain.com		IN	A	128.32.0.4
		IN	A	10.0.0.78

Figure 8.1 Example of an A record

CNAME (Canonical Name)

The Canonical Name resource record, **CNAME**, specifies an alias or nickname for the official, or canonical, host name. This record must be the only one associated with the alias name. It is usually easier to supply one **A** record for a given address and use **CNAME** records to define alias host names for that address.

Figure 8.2 shows an example of a **CNAME** resource record:

alias	{ttl}	addr-class	{type}	Canonical name
wip.domain.com		IN	CNAME	host1.domain.com

Figure 8.2 Example of a CNAME record

MX (Mail Exchange)

The Mail Exchange resource record, **MX**, defines the mail system(s) for a given domain.

Figure 8.3 shows an example of an **MX** resource record.

name	{ttl}	addr-class	MX	pref	value	mail exchange
Munnari.OZ.AU.		IN	MX	0		Seismo.CSS.GOV.
*.IL.		IN	MX	0		RELAY.CS.NET.

Figure 8.3 Example of an MX record

NS (Name Server)

The name server resource record, **NS**, defines the name servers for a given domain, creating a delegation point and a subzone. The first name field specifies the zone that is serviced by the name server that is specified by the second name. Every zone needs at least two name servers.

Figure 8.4 shows an example of an **NS** resource record.

{name}	{ttl}	addr-class	NS	Name servers name
domain.com		IN	NS	host1.domain.com.
domain.com		IN	NS	host2.domain.com.

Figure 8.4 Example of an NS record

PTR (Pointer)

A name pointer resource record, **PTR**, associates a host name with a given IP address. These records are used for reverse name lookups.

The example of a **PTR** record shown in Figure 8.5 is used to set up reverse pointers for the special **IN-ADDR.ARPA** domain.

name	{ttl}	addr-class	PTR	real name
7.0		IN	PTR	monet.Berkeley.Edu.

Figure 8.5 Example of a PTR record

SOA (Start of Authority)

The start of authority resource record, **SOA**, starts every zone file and indicates that a name server is the best source of information for a particular zone. In other words, the **SOA** record indicates that a name server is authoritative for a zone. There must be exactly one **SOA** record per zone.

The following is an example of an **SOA** record.

name	{ttl}	addr-class	SOA	Origin	Person in charge
@		IN	SOA	ucbvax.Berkeley.Edu.	johndoe.berkeley.edu (
				1995122103	; Serial
				10800	; Refresh
				1800	; Retry
				3600000	; Expire
				259200)	; Minimum

Figure 8.6 Example of an SOA record

The specific fields in an **SOA** record are defined as follows:

- ◆ **Person in charge**
The email address for the person responsible for the name server, with the at character (@) changed to a dot (.). For example, **johndoe@berkeley.edu** becomes **johndoe.berkeley.edu**.
- ◆ **Serial number**
The version number of the data file; it must be a positive integer. You must increase this number whenever a change is made to the data.
- ◆ **Refresh**
The time interval between calls, in seconds, that the secondary name servers make to the primary name server to check if an update is necessary.

- ◆ **Retry**
The time interval, in seconds, that a secondary server waits before retrying a failed zone transfer.
- ◆ **Expire**
The maximum number of seconds that a secondary name server can use the data before it expires for lack of receiving a refresh.
- ◆ **Minimum**
The default number of seconds to be used for the time to live (TTL) field on resource records which do not specify a TTL in the zone file. It is also an enforced minimum on TTL if it is specified on a resource record in the zone.

Additional resource record types

Table 8.2 lists less common resource record types. For more information on these, see RFCs 1035, 1183, and 1664.

Type	Description
AAAA	IPv6 address
AFSDB	AFS database location
GPOS	Geographical position
HINFO	Host information
ISDN	Integrated services digital network address
KEY	Public key
KX	Key exchanger
LOC	Location information
MB	Mailbox domain name
MINFO	Mailbox or mail list information
NULL	A null RR
NSAP	Network service access point address
NSAP-PTR	(Obsolete)

Table 8.2 Less common resource records

Type	Description
NXT	Next domain
PX	Pointer to X.400/RFC822 information
RP	Responsible person
RT	Route through
SIG	Cryptographic signature
SRV	Server selection
TXT	Text strings
WKS	Well-known service description
X25	X25

Table 8.2 Less common resource records



9

Scripts

- Working with scripts

Working with scripts

The 3-DNS Controller is shipped with several scripts to simplify many configuration and maintenance tasks. This chapter provides information about the functionality of these scripts. If you plan on doing a scripted task from the command line, you should find this section helpful. Many scripts correspond to commands on the 3-DNS Maintenance menu.

◆ **Note**

Before you edit a script, make a backup copy of the original.

3dns_add script

The **3dns_add** script allows you to add a new 3-DNS Controller to an existing sync group in your network. The **3dns_add** script copies all configuration information from an existing 3-DNS Controller onto the new controller. For more details on using this script, please refer to Chapter 5, *Adding 3-DNS Controllers to the Network*, in the **3-DNS Administrator Guide**.

3dns_admin_start script

The **3dns_admin_start** script corresponds to the **Restart 3-DNS Configuration Utility** command on the 3-DNS Maintenance menu. This command restarts the 3-DNS web server, which hosts the Configuration utility.

3dns_dump script

The **3dns_dump** script saves the current state of the **3dnscd** cache to a new **/var/3dns/etc/wideip.conf** file.

3dns_sync_metrics script

The **3dns_sync_metrics** script corresponds to the **Synchronize Metrics Data** command on the 3-DNS Maintenance menu. You should use this script only when you are configuring a new 3-DNS Controller. This script prompts you to copy metrics data from a remote 3-DNS Controller to the local 3-DNS Controller.

3dns_web_config script

The **config_httpd** script corresponds to the **Reconfigure 3-DNS Configuration Utility** command on the 3-DNS Maintenance menu. This script lets you make configuration changes to the 3-DNS web server, which hosts the Configuration utility.

3dns_web_passwd script

The **3dns_web_passwd** script corresponds to the **Change/Add Users for 3-DNS Configuration Utility** command on the 3-DNS Maintenance menu. This script secures the 3-DNS web server using basic authentication. This script lets you provide restricted or administrative access to the 3-DNS web server for selected users only, and assigns passwords for those users. Users with restricted access have access to the statistics area only. Users with administrative access have access to all areas of the 3-DNS web server.

◆ Note

*The **3dns_web_passwd** script is run by the First-Time Boot utility. You can run this script again any time you need to provide access for another user.*

3dnsmaint script

The **3dnsmaint** script opens the 3-DNS Maintenance menu.

3dprint script

The **3dprint** script corresponds to the **Dump 3dnspd Statistics** command on the 3-DNS Maintenance Menu. This script lets you view these statistics screens on the command line:

◆ 3-DNS

Displays statistics about each 3-DNS Controller in your network; the statistics include such things as whether the controller is enabled or disabled, the number of packets per second traveling in and out of the 3-DNS Controller during the last sample period, and the name of the sync group to which each 3-DNS Controller belongs.

◆ BIG-IP

Displays statistics about all BIG-IP Controllers known to the 3-DNS Controller; the statistics include such things as the number of virtual servers each BIG-IP Controller manages, and the number of times the 3-DNS Controller resolves requests to those virtual servers.

- ◆ **Hosts**

Displays statistics about all hosts known to the 3-DNS Controller; the statistics include such things as the number of times that the 3-DNS Controller resolves requests to the host, and the number of virtual servers that the hosts manage.
- ◆ **Virtual Servers**

Displays statistics about BIG-IP Controllers and host virtual servers; the statistics include such things as the server state, and the number of times it has received resolution requests.
- ◆ **Paths**

Displays path statistics, such as round trip time, packet completion rate, the remaining time to live (TTL) before a path's metric data needs to be refreshed.
- ◆ **Local DNS**

Displays statistics collected for LDNS servers; the statistics include such things as the number of resolution requests received from a given server, and the current protocol used to probe the server.
- ◆ **Wide IPs**

Displays statistics about each wide IP defined on the 3-DNS Controller; the statistics include such things as load balancing information, and the remaining time to live (TTL) before the wide IP's metrics data needs to be refreshed.
- ◆ **Globals**

Displays statistics about the globals sub-statements; the statistics include such things as the current and default values for each of the globals sub-statements, and whether you have to restart **3dnsd** when you make changes to the parameters.
- ◆ **Summary**

Displays summary statistics, such as the 3-DNS Controller version, the total number of resolved requests, and the load balancing methods used to resolve requests.
- ◆ **Data Centers**

Displays statistics about the data centers, and their servers, in your network. The statistics include such things as the names of the data centers, the name or IP address of the servers in the data center, and whether the data center is enabled or disabled.
- ◆ **Sync Groups**

Displays statistics about each sync group in your network. The statistics include such things as the name of the sync group, whether **3dnsd** is running on each 3-DNS Controller, whether the **big3d** agent is running on each 3-DNS Controller, the name and IP address of the 3-DNS Controller, and whether the 3-DNS Controller is a principal or receiver.

3ndc script

The **3ndc** script starts the **3ndc** utility, which is described in the **3ndc** man page.

big3d_install script

The **big3d_install** script corresponds to the **Install and Start big3d** command on the 3-DNS Maintenance menu. This script installs and starts the appropriate version of the **big3d** agent on each BIG-IP Controller, EDGE-FX Cache, and GLOBAL-SITE Controller that the 3-DNS Controller knows about. This script is useful for 3-DNS Controller updates.

The **big3d_install** script performs the following procedure on each BIG-IP Controller, EDGE-FX Cache, of GLOBAL-SITE Controller:

1. Stops the running **big3d** agent process.
2. Uses a matrix file to determine which version of the **big3d** agent to copy to the BIG-IP Controller, EDGE-FX Cache, or GLOBAL-SITE Controller. The matrix file is a file that lists version numbers for all BIG-IP Controllers, EDGE-FX Caches, and GLOBAL-SITE Controllers known to the 3-DNS Controller and the version numbers of the **big3d** agent running on each BIG-IP Controller, EDGE-FX Cache, and GLOBAL-SITE Controller.
3. Adds the following to the end of the **/etc/rc.conf** file:

```
big3d_enabled="yes"
```
4. Starts **/usr/sbin/big3d**.

For configuration options, see the **big3d** man page.

big3d_restart script

The **big3d_restart** script corresponds to the **Restart big3d** command on the 3-DNS Maintenance menu. This script stops and restarts the **big3d** agent on each BIG-IP Controller, EDGE-FX Cache, and GLOBAL-SITE Controller known to the 3-DNS Controller.

big3d_version script

The **big3d_version** script corresponds to the **Check remote versions of big3d** command on the 3-DNS Maintenance menu. This script displays the version numbers for all BIG-IP Controllers, EDGE-FX Caches, and GLOBAL-SITE Controllers known to the 3-DNS Controller, as well as the version numbers of the **big3d** agent running on those devices.

config_ssh script

The **config_ssh** script corresponds to the **Configure SSH communication with remote devices** command on the 3-DNS Maintenance menu. All 3-DNS Controller scripts, and synchronization, require secure communications between controllers. Any time you add a new 3-DNS Controller, BIG-IP Controller, EDGE-FX Cache, or GLOBAL-SITE Controller to a network, you can run the **config_ssh** script, and if no **ssh** key exists on the controller, the script configures **ssh** access.

◆ **Note**

This script is not available in the non-crypto version of the 3-DNS Controller.

edit_lock script

The **edit_lock** script lets you safely edit a specified file that is synchronized between 3-DNS Controllers in a sync group. This script creates a temporary version of the original file, and this temporary file replaces the original file when you are finished editing it. If you do not use this script to edit a file, there is the danger that a partial file might be synchronized to other 3-DNS Controllers in the sync group.

To use this script, type the following, at the command line:

```
edit_lock <file name>
```

edit_wideip script

The **edit_wideip** script corresponds to the **Edit 3-DNS Configuration** command on the 3-DNS Maintenance menu. This script opens the **wideip.conf** file for editing, copies it to all other 3-DNS Controllers in the local 3-DNS Controller's sync group, and restarts the **3dnsd** utility.

install_key script

The **install_key** script corresponds to the **Generate and Copy iQuery Encryption Key** command on the 3-DNS Maintenance menu. This script starts the **F5makekey** program, and generates a seed key for encrypting communications between the 3-DNS Controllers and (if you have any in your network) BIG-IP Controllers, EDGE-FX Caches, or GLOBAL-SITE Controllers. The **install_key** script creates and distributes the iQuery key to all BIG-IP Controllers, EDGE-FX Caches, GLOBAL-SITE Controllers, and other 3-DNS Controllers in your network.

◆ **Note**

This script is not available on the non-crypto version of 3-DNS Controller.

To start the **F5makekey** program, type the following at the command line, in the **/usr/sbin** directory:

```
f5makekey
```

The seed value is located in **/etc/F5key.dat** and contains a random length (12-52) of random content (1-255), created by the **F5makekey** program. This array of values is used by MD-160, a one-way hash function, to generate a key (7 characters in length) for the Blowfish encryption algorithm.

syncd_checkpoint script

The **syncd_checkpoint** script creates a checkpoint file. A **checkpoint file** is a compressed tar file that contains an archive of the files that are synchronized.

You can run this script with or without arguments. If you run **syncd_checkpoint** without specifying arguments, the script creates the following default checkpoint file:

```
/var/3dns/staging/checkpoint/default.tar.gz
```

◆ **Note**

*All checkpoint file names have a **.tar.gz** suffix.*

The **syncd_checkpoint** script can take the following optional arguments:

```
syncd_checkpoint [-c <name>] [-i]
```

The options for **syncd_checkpoint** are defined in Table 9.1.

Option	Description
-c <name>	Creates a checkpoint file with the specified file name. You can also specify a non-default path for the file, unless the path starts with a slash (/). The default path for checkpoint files is /var/3dns/staging/checkpoint/ . The syncd_checkpoint script automatically appends a .tar.gz extension to the end of the file name.
-i	Runs the script in an interactive session, which means that you are prompted for a file name.

*Table 9.1 Optional arguments for the **syncd_checkpoint** script*

syncd_rollback script

The **syncd_rollback** script decompresses a checkpoint file, which contains an archive of all synchronized files. This has the effect of replacing the current files with the files archived in the checkpoint file.

The **syncd_rollback** script can take the following optional arguments:

```
syncd_rollback [-c] [-c <name>] [-r] [-u] [-i]
```

◆ Note

*When you run this script from the command line, you must use the **-r**, **-u**, or **-i** option.*

The options for **syncd_rollback** are defined in Table 9.2.

Option	Description
-c	Unrolls the most recently created checkpoint file, whether it is in the default location or elsewhere.
-c <name>	Unrolls the specified checkpoint file, whether it is in the default location or elsewhere. It is not necessary to end the name with .tar.gz , as this suffix is assumed.
-r	Restores archived files with their old timestamps. This means that if any of the synchronized files were updated on a remote 3-DNS Controller, the updated files will overwrite any older files contained in the checkpoint file.
-u	Restores archived files with updated timestamps with the current time. This means that the files in the checkpoint are synchronized to the remote 3-DNS Controllers and overwrite the existing files on the remote 3-DNS Controllers.
-i	Runs the script in an interactive session, which means that you are prompted for option information.

*Table 9.2 Optional arguments for the **syncd_rollback** script*

syncd_start script

The **syncd_start** script corresponds to the **Restart syncd** command on the 3-DNS Maintenance menu. This script restarts the **syncd** daemon if it is already running, or starts it if it is not. You can run this script with or without arguments. If you run **syncd_start** without specifying arguments, the script starts or restarts **syncd**. The **syncd_start** script can take the following optional arguments:

```
syncd_start [-c] [-c <name>] [-r] [-u] [-i]
```

◆ Note

*When you use the **-c** option, you must also use either the **-r** or **-u** option.*

The options for **syncd_start** are defined in Table 9.3.

Option	Description
-c	Before restarting syncd , unrolls the most recently created checkpoint file, whether it is in the default location or elsewhere.
-c <name>	Before restarting syncd , unrolls the specified checkpoint file, whether it is in the default location or elsewhere. It is not necessary to end the name with .tar.gz , as this suffix is assumed.
-r	Restores the archived files with their old timestamps. This means that if any of the synchronized files were updated on a remote 3-DNS Controller, the updated files overwrite the rolled back files.
-u	Restores the archived files with updated timestamps to the current time. This means that the files in the checkpoint file overwrite any updated files on remote 3-DNS Controllers.
-i	Runs the script in an interactive session, which means that you are prompted for option information.

*Table 9.3 Optional arguments for the **syncd_restart** script*

syncd_stop script

The **syncd_stop** script corresponds to the **Stop syncd** command on the 3-DNS Maintenance menu. This script stops the **syncd** daemon if it is running. You can run this script with or without arguments. If you run **syncd_stop** without specifying arguments, the script simply stops **syncd**. The **syncd_stop** script can take the following optional arguments:

```
syncd_stop [-c] [-c <name>] [-i]
```

The options for **syncd_stop** are defined In Table 9.4.

Option	Description
-c	Creates a checkpoint file in the default location before stopping syncd .
-c <name>	Creates a checkpoint file with the specified name and path before stopping syncd .
-i	Runs the script in an interactive session, which means that you are prompted for option information.

Table 9.4 *Optional arguments for the **syncd_stop** script*



10

SNMP

- Working with SNMP on the 3-DNS Controller
- Configuring SNMP on the 3-DNS Controller
- Configuring options for the checktrap.pl script
- Configuring the 3-DNS SNMP agent using the Configuration utility
- Configuring host SNMP settings on the 3-DNS Controller
- Configuring the SNMP agent on host servers

Working with SNMP on the 3-DNS Controller

The 3-DNS Controller ships with a customized simple network management protocol (SNMP) agent and management information base (MIB). This chapter describes the management and configuration tasks with which you can configure the 3-DNS SNMP agent.

The 3-DNS SNMP agent and MIB allow you to monitor the 3-DNS Controller by configuring traps for the SNMP agent or by polling the controller with your standard network management station. The 3-DNS SNMP agent has the following options to ensure secure management:

- Community names
- TCP wrappers
- View access control mechanism (VACM)

You can use the Configuration utility to configure the 3-DNS SNMP agent to send traps to your network management system. You can also set up custom traps by editing several configuration files.

WARNING

If you want to monitor the 3-DNS Controller using the SEE-IT Network Manager, you must configure the SNMP agent on the 3-DNS Controller.

Configuring SNMP on the 3-DNS Controller

To use SNMP on the 3-DNS Controller, you must complete the following tasks:

- ◆ Download the 3-DNS MIBs and load them into your network management station
- ◆ Modify the following configuration files:
 - **/etc/hosts.allow**
 - **/etc/snmpd.conf**
 - **/etc/snmptrap.conf**
 - **/etc/syslog.conf**
- ◆ Configure options for the **checktrap** script

Downloading the MIBs

The 3-DNS Controller includes a proprietary 3-DNS SNMP MIB. This MIB is specifically designed for use with the 3-DNS Controller. You can configure the SNMP settings in the Configuration utility or on the command line.

SNMP management software requires that you use the MIB files associated with the device. You can obtain three MIB files from the 3-DNS directory **/usr/local/share/snmp/mibs**, or you can download the files from the **Additional Software Downloads** section of the Configuration utility home screen. The files you need are:

- ◆ **3dns.my**
This is a vendor MIB that contains specific information for properties associated with specific functionality, such as load balancing.
- ◆ **rfc1611.my**
This is a DNS server MIB (RFC 1611) that provides standard management information.
- ◆ **UCD-SNMP-MIB.txt**
This is a MIB-II (RFC 1213) that contains specific management information for the UC-Davis SNMP agent.

For information about the objects defined in **3dns.my**, refer to the descriptions in the object identifier (OID) section of the MIB file. For information about the objects defined in **rfc1611.my**, refer to RFC 1611.

Understanding configuration file requirements

You need to make changes to several configuration files on the 3-DNS Controller before using the SNMP agent. Once you change these configuration files, you must restart the SNMP agent. The files are discussed in the following sections.

`/etc/hosts.deny`

The **/etc/hosts.deny** file must be present to deny, by default, all UDP connections to the SNMP agent. The contents of this file are as follows:

```
ALL : ALL
```

`/etc/hosts.allow`

The **/etc/hosts.allow** file specifies the hosts that are allowed to access the SNMP agent. You can configure access to the SNMP agent with the **/etc/hosts.allow** file in one of two ways:

- By typing in an IP address, or list of IP addresses, that are allowed to access the SNMP agent.

- By typing in a network address and mask to allow a range of addresses in a subnet to access the SNMP agent.

For a specific list of addresses, type in the list of addresses you want to allow access to the SNMP agent. Addresses in the list must be separated by blank space or by commas. The basic syntax is as follows:

```
daemon: <IP address> <IP address> <IP address>
```

For example, if you type the following line, the SNMP agent accepts connections from the specified IP addresses:

```
snmpd: 128.95.46.5 128.95.46.6 128.95.46.7
```

For a range of addresses, the basic syntax is as follows, where **daemon** is the name of the daemon, and **NETWORKADDRESS/MASK** specifies the network that is allowed access:

```
daemon: NETWORKADDRESS/MASK
```

For example, the following line sets the **snmpd** daemon to allow connections from the **128.95.46.0/255.255.255.0** address:

```
snmpd: 128.95.46.0/255.255.255.0
```

The previous example allows the 256 possible hosts from the network address **128.95.46.0** to access the SNMP daemon. You may also use the keyword **ALL** to allow access for all hosts or all daemons.

◆ Note

*If you prefer, instead of modifying this file from the command line, you can use the Configuration utility to specify the hosts that are allowed to access the SNMP agent. See **To set SNMP properties using the Configuration utility**, on page 10-7.*

/etc/snmpd.conf

The **/etc/snmpd.conf** file controls most aspects of the SNMP agent. This file is used to set up and configure certain traps, passwords, and general SNMP variable names.

A few of the necessary variables are listed below:

◆ System Contact Name

The System Contact is a MIB-II simple string variable defined by almost all SNMP boxes. It usually contains a user name and an email address. This is set by the **syscontact** key.

◆ Machine Location (string)

The Machine Location is a MIB-II variable that is supported by almost all boxes. It is a simple string that defines the location of the box. This is set by the **syslocation** key.

◆ **Community String**

The community string clear text password is used for basic SNMP security. This also maps to VACM groups, but for initial read-only access it is limited to only one group.

◆ **Trap Configuration**

Trap configuration is controlled by these entries in the `/etc/snmpd.conf` file:

- **trapsink <host>**

This sets the host to receive trap information. The `<host>` variable is an IP address.

- **trapport <port>**

This sets the port on which traps are sent. There must be one **trapport** line for each **trapsink** host.

- **trapcommunity <community string>**

This sets the community string (password) for sending traps. Once set, it also sends a trap upon startup: **coldStart(0)**.

- **authtrappable <integer>**

Set this variable to **1** so that traps can be sent for authentication warnings. Set the variable to **2** to disable it.

*Note: To change the trap port, be sure the **trapport** line precedes the **trapsink** line. If you use more than one **trapport** line, there must be one **trapport** line before each **trapsink** line. The same is true for **trapcommunity**; if you use more than one **trapcommunity** line, there must be one **trapcommunity** line before each **trapsink** line.*

◆ **System IP Setting**

You must set the system IP address using the **sysip** command; if this setting is not present, the **checktrap.pl** script fails to send all 3-DNS-specific traps. Use the following syntax to set the system IP address:

```
sysip <3-DNS IP address>
```

◆ **Note**

*If you prefer, instead of modifying this file from the command line, you can use the Configuration utility to set these SNMP properties. See **To set SNMP properties using the Configuration utility**, on page 10-7.*

/etc/snmptrap.conf

The configuration in **/etc/snmptrap.conf** determines which messages generate traps and what those traps are. The file includes OIDs, traps, and regular expression mappings. The configuration file specifies whether to send a specific trap based on a regular expression. An excerpt of the configuration file is shown in Figure 10.1.

```
# Default traps.
.1.3.6.1.4.1.3375.1.2.2.2.0.1 (SNMP_TRAP: VS.*?state change green.*?red) VIRTUAL SERVER
GREEN TO RED

.1.3.6.1.4.1.3375.1.2.2.2.0.2 (SNMP_TRAP: VS.*?state change red.*?green) VIRTUAL SERVER
RED TO GREEN

.1.3.6.1.4.1.3375.1.2.2.2.0.3 (SNMP_TRAP: SERVER.*?state change green.*?red) SERVER
GREEN TO RED

.1.3.6.1.4.1.3375.1.2.2.2.0.4 (SNMP_TRAP: SERVER.*?state change red.*?green) SERVER RED
TO GREEN

.1.3.6.1.4.1.3375.1.2.2.2.0.5 (SNMP_TRAP: iQuery message from big3d) CRC FAILURE
```

Figure 10.1 Excerpt from the */etc/snmptrap.conf* file

Some of the OIDs have been permanently mapped to specific 3-DNS Controller events. The OIDs that are permanently mapped for the 3-DNS Controller include:

- Virtual server green to red
- Virtual server red to green
- Server green to red
- Server red to green
- CRC failure
- Pool red to green
- Pool green to red
- 3-DNS Controller active to standby
- 3-DNS Controller standby to active

To see messages that are triggering an SNMP trap, look in the **var/3dns/log/3dns.log** file.

/etc/syslog.conf

To generate traps, you must configure **syslog** to send syslog lines to **checktrap.pl**. If the syslog lines match the specified regular expression in the **snmptrap.conf** file, the **checktrap.pl** script generates a valid SNMP

trap. The following line in the `/etc/syslog.conf` file causes the `syslog` utility to send the specified log output to the `checktrap.pl` script. The `checktrap.pl` script then compares the logged information to the `snmptrap.conf` file to determine if a trap should be generated.

```
local2.warning | exec /sbin/checktrap.pl.
```

◆ **Note**

If you uncomment this line, make sure you restart `syslogd`.

Configuring options for the `checktrap.pl` script

The `checktrap.pl` script reads a set of lines from standard input. The script checks each line against a set of regular expressions. If a line matches the regular expression, an SNMP trap is sent.

The following options are available for the `checktrap.pl` script.

◆ **SNMP configuration file**

This file contains the SNMP variables. The `checktrap.pl` script gets trap configuration information from this file. The default is `/etc/snmpd.conf`.

```
snmpd_conf_file=<snmp configuration file>
```

◆ **SNMP trap configuration file**

This file contains the regular expression to SNMP trap OID mappings. It also contains a description string that is added to the trap message. The default is `/etc/snmptrap.conf`.

```
trapd_conf_file=<snmp trap configuration file>
```

◆ **SNMP trap program**

This program sends the SNMP trap. This program should be the `snmptrap` program included with the 3-DNS Controller. The default is `/sbin/snmptrap`.

```
trap_program=<snmp trap program>
```

◆ **Date removal**

This option turns off automatic date removal. Normally, each input line is expected to begin with a date. Typically, this date is removed before the trap is sent. This option keeps the date information in the trap. If you do not add this option, the date is removed from the trap by default.

```
no_date_strip
```

◆ **Usage**

This option prints a usage string.

```
usage
```


Configuring the 3-DNS SNMP agent using the Configuration utility

You can use the Configuration utility to configure the following aspects of the 3-DNS SNMP agent:

- ◆ **Client access**
You can define a network address and netmask for a workstation from which SNMP requests are acceptable.
- ◆ **System information**
You can name a system contact, a machine location, and a community string.
- ◆ **Trap configuration**
You can enter a trap sink and a trap community.

To set SNMP properties using the Configuration utility

The Configuration utility provides sample SNMP settings for your reference. To use the 3-DNS SNMP MIB, you must replace these sample settings with settings appropriate to your environment and your specific SNMP management software.

1. In the navigation pane, click **SNMP**.
The SNMP Configuration screen opens.
2. Add the SNMP settings. For help on configuring the SNMP settings, click **Help** on the toolbar.

Configuring host SNMP settings on the 3-DNS Controller

After defining a host server, you need to configure its SNMP settings if you want to use SNMP host probing. Remember that you must first set up at least one SNMP prober factory on any 3-DNS Controller, BIG-IP Controller, or EDGE-FX Cache that runs the **big3d** agent.

The SNMP prober collects the following information:

- Memory utilization
- CPU utilization
- Disk space utilization
- Kilobytes/second
- Current connections
- Packet rate

The 3-DNS Controller gathers metrics for BIG-IP Controllers, EDGE-FX Caches, and several host servers. For information on the server types and the specific metrics that are collected, please refer to Table 10.1. To see the current performance of any of these server metrics, review the Metrics statistics screen.

Server Type or Operating System	Metrics collected:					
	Kilobytes/Second	Packets/Second	CPU	Memory	Disk	Current Connections
BIG-IP Controller	X	X				X
EDGE-FX Cache	X	X				X
Alteon Ace Director	X					X
BSD, UC Davis	X	X	X	X	X	X
CacheFlow	X	X	X			X
Cisco CSS series	X	X				X
Cisco LocalDirector	X	X				X
Cisco LocalDirector	X	X				X
Linux, UC Davis	X	X		X	X	X
Sun Solaris	X	X	X			X
Windows 2000 Server	X	X	X			X
Windows NT 4.0	X	X	X	X		X

Table 10.1 Server types and the metrics collected by the 3-DNS Controller

◆ **Note**

The Cisco LocalDirector metric shows new connections per second rather than current connections.

To configure host SNMP settings using the Configuration utility

1. In the navigation pane, expand the **Servers** item, and click **Host Servers**.
2. From the Host Server column, click a host server.
The Modify Host screen opens.
3. On the toolbar, click **SNMP Configuration**.
The Host SNMP Configuration screen opens.
4. Add the host SNMP settings. For help on configuring the host SNMP settings, click **Help** on the toolbar.

To configure host SNMP settings from the command line

1. At the command prompt, type **3dnsmaint** to open the 3-DNS Maintenance menu.
2. On the 3-DNS Maintenance menu, choose **Edit 3-DNS Configuration** to open the **wideip.conf** file.
3. Locate or add the host **server** statement.

All **server** statements should appear after the **sync_group** statement and before **wideip** statements.
4. Define the server type, address, name, prober, probe protocol, and port information as usual.
5. Add the **snmp** statement.
6. Define the virtual server information as usual.

Figure 10.2 shows the SNMP syntax for a host server in bold.

```
server {
  type host
  address <IP address>
  name <"host_name">
  probe_protocol <tcp | icmp>
  [ prober <IP address> ]
  port <port number> | service <"service name">
  [ snmp {
    agent <generic | ucd | solstice | ntserv | win2kserv | ciscold | ciscold2 | ciscold3
  | foundry | arrowpoint | alteon | cacheflow>
    port <port number>
    community <"community string">
    timeout <seconds>
    retries <number>
    version <SNMP version>
  } ]
  vs {
    address <virtual server IP address>
    port <port number> | service <"service name">
    [ probe_protocol <tcp | icmp> ]
  }
}
```

Figure 10.2 Configuring host SNMP settings

Configuring the SNMP agent on host servers

For host probing to work, you need to verify that the SNMP agent is properly configured on the host itself. We recommend that you refer to the documentation provided with your host SNMP software for complete configuration information.



||

Topology

- Working with Topology load balancing
- Setting up topology records
- Using the Topology load balancing mode in a wide IP
- Using the Topology load balancing mode in a pool
- Working with the topology statement in the wideip.conf file

Working with Topology load balancing

To use the Topology load balancing mode, you first set up topology records in a topology statement. Once you have defined a topology statement, you can set up Topology load balancing among pools in a wide IP, or within a pool. Note that if you do not create a topology statement, and you configure Topology as the load balancing mode, the 3-DNS Controller load balances requests using the Random mode.

The crypto 3-DNS Controller includes a database that maps IP addresses to geographic locations. With this database, the controller can use the geographic attributes of local DNS servers to direct traffic.

The following sections describe how to create a topology statement, and how to set up Topology load balancing.

Setting up topology records

A *topology record* has three elements: an LDNS server location endpoint, a virtual server location endpoint, and a relative weight. The location endpoints can be one of the following:

- IP subnet (CIDR definition)
- Wide IP pool (managed by the 3-DNS Controller)
- Data center (managed by the 3-DNS Controller)
- Country (based on top-level domain codes, as specified by IANA, the Internet Assigned Numbers Authority)
- Continent
- America Online (AOL) (for LDNS server location endpoints only)

The relative weight, or score, for the topology record allows the 3-DNS Controller to evaluate the best resolution option for a DNS request. The not (!) operator, when used in a topology record, indicates location endpoints not equal to that value.

A *topology statement* is composed of one or more topology records. Figure 11.1 is an example of a topology statement, with two topology records, as it appears in the Configuration utility.



Figure 11.1 Example of topology records in a topology statement

Here is an explanation of how to interpret the topology statement in the preceding example. A wide IP pool labeled **origin** manages the virtual servers that are returned for DNS resolution requests sent by LDNS servers located in North America. A wide IP pool labeled **cache_farm** manages the virtual servers that are returned for DNS resolution requests sent by LDNS servers located anywhere except North America. When the 3-DNS Controller receives a DNS resolution request from an LDNS server located in North America, it evaluates the first topology record and assigns a score of 100, because the LDNS server criteria matches. The controller then evaluates the next topology record, and assigns a score of 0 because the LDNS server criteria does not match. The controller then routes the DNS request to the wide IP pool **origin** for resolution, because that topology record has the highest score.

To add topology records using the Configuration utility

1. In the navigation pane, click **Topology**.
The Manage Topology Records screen opens.
2. Add the settings for the topology records, and click **Add**. For assistance with the settings on this screen, click **Help** on the toolbar.

To remove topology records using the Configuration utility

1. In the navigation pane, click **Topology**.
The Manage Topology Records screen opens.
2. Select the topology record that you want to remove from the Current Topology Records list, and click **Remove**.
The topology record is removed from the topology statement. For assistance with the settings on this screen, click **Help** on the toolbar.

Using the Topology load balancing mode in a wide IP

You can use the Topology load balancing mode to distribute traffic among wide IP pools. You must have at least two pools configured in the wide IP. You can use the Topology load balancing mode with pools to direct traffic to a specific data center in your network, to a third-party network, or to a content delivery network.

To set up topology to distribute traffic among wide IP pools using the Configuration utility

1. In the navigation pane, click **Wide IPs**.
The Wide IP List screen opens.
2. In the Wide IP column, click a wide IP name.
The Modify Wide IP screen opens.
3. In the **Pool LB Mode** box, select **Topology** as the load balancing mode for the wide IP.
4. Click **Update**.

To set up topology to distribute traffic among wide IP pools from the command line

1. At the command prompt, type **3dnsmaint** to open the 3-DNS Maintenance menu.
2. On the 3-DNS Maintenance menu, choose **Edit 3-DNS Configuration** to open the **wideip.conf** file.
3. Locate the **wideip** statement that you want to edit.
4. Define **topology** as the **pool_lbmode** load balancing mode for the wide IP.

Figure 11.2 shows a sample wide IP definition where Topology is the load balancing mode among the pools in this wide IP configuration.

```
wideip {
  address 192.168.44.1
  name    "www.domain.com"
  port    80
  pool_lbmode topology

  pool {
    name      "cache_farm"
    fallback  null
    address   192.168.44.10
    address   192.168.44.20
  }

  pool {
    name      "origin"
    address   172.168.11.10
    address   172.168.11.20
  }
}
```

Figure 11.2 Example syntax for Topology pool load balancing in a wide IP

Using the Topology load balancing mode in a pool

In addition to setting up the Topology load balancing mode to select a pool within a wide IP, you can also set up the Topology load balancing mode to select a virtual server within a pool. However, you must configure the topology records before the 3-DNS Controller can use the Topology load balancing mode within a pool. If you have no topology records in the topology statement, **Topology** does not appear as an option for the **Preferred**, **Alternate**, or **Fallback** load balancing modes for pools.

To set up topology load balancing within a pool using the Configuration utility

1. In the navigation pane, click **Wide IPs**.
The Wide IP List screen opens.
2. In the Pools column, click a pool.
The Modify Wide IP Pools screen opens.
3. In the Pool Name column, click a pool name.
The Modify Load Balancing for [pool name] screen opens.
4. In the **Preferred** box, select **Topology** as the load balancing mode for the pool.

5. Click **Update**.
The change is added to the configuration.

To set up topology load balancing in a pool from the command line

1. At the command prompt, type **3dnsmaint** to open the 3-DNS Maintenance menu.
2. On the 3-DNS Maintenance menu, choose **Edit 3-DNS Configuration** to open the **wideip.conf** file.
3. Locate the **wideip** statement you want to edit.
4. Define **topology** as the **preferred** load balancing mode for the pool.

The example in Figure 11.3 shows a sample wide IP definition where **topology** is the load balancing mode within a pool.

```
wideip {
  address 192.168.103.60
  port 80
  name "ntp.wip.domain.com"
  pool {
    name "poolA"
    preferred topology
    alternate rtt
    address 192.168.101.60 // New York
    address 192.168.102.60 // Los Angeles
    address 192.168.103.60 // Tokyo
  }
}
```

Figure 11.3 Example of Topology load balancing mode within a pool

Working with the topology statement in the wideip.conf file

The **topology** statement, in the **wideip.conf** file, can include the following variables to specify pools, data centers, continents, and countries, in addition to the traditional CIDR blocks, for both servers and local DNS servers.

Variable	Description
pool	Specify a wide-IP pool to score for load balancing. Note that pool names can be duplicated across wide IPs.
datacenter	Specify a data center to score for load balancing.
continent	Specify one of these continents for load balancing: " North America ", " South America ", " Europe ", " Asia ", " Australia ", " Africa ", or " Antarctica ".
country	Specify a country for load balancing using one of the two-letter country codes found in the file <code>/var/3dns/include/net.ccd</code> .
isp.AOL	For local DNS servers only, specify the Internet service provider, America Online (AOL).

Table 11.1 Variables used in the *topology* statement

To add a **topology** statement to the include file `/var/3dns/include/topology.inc`, follow the format of this example.

```

topology {
    // server          ldns          score
    "pool.origin"     cont."North America"  100
    "pool.cache_farm" !cont."North America"  100
}

```

Figure 11.4 Example of a *topology* statement

◆ **Note**

Use the **not (!)** notation in a *topology* statement to negate the meaning of an element, as shown in Figure 11.4.



12

Utilities

- Working with command line utilities
- Accessing 3-DNS Controller utilities documentation

Working with command line utilities

The 3-DNS Controller includes several command line utilities. These utilities allow you to configure the DNS and the various features of the 3-DNS Controller. You may also want to review Chapter 9, *Scripts*, for additional 3-DNS Controller configuration options.

Accessing 3-DNS Controller utilities documentation

You can access the most current documentation on 3-DNS Controller utilities by using the Configuration utility or by using the command line.

Accessing 3-DNS Controller documentation using the Configuration utility

You can view all the documentation for the 3-DNS Controller from the main screen of the Configuration utility, including the man pages for the utilities that are shipped with the controller.

To access documentation on 3-DNS Controller utilities using the Configuration utility

1. Log on to the Configuration utility.
2. From the Online Documentation section of the main 3-DNS Controller screen, click the **3-DNS Man Pages** link.
A screen containing an index of 3-DNS man pages opens.

Accessing 3-DNS Controller documentation from the command line

Using the command line, you can display a list of utilities that fall into a particular category, or else display the man page for a specific utility.

To display a list of utilities that fall into a particular category

To display a list of utilities that fall into a particular category, type the following command:

```
man -k <category>
```

For example, to get a list of utilities that pertain to DNS, type the following command, and a list of utilities that pertain to DNS appears.

```
man -k dns
```

To display documentation for a specific 3-DNS Controller utility

To display the man page for a specific utility, type the following command:

```
man <utility>
```

For example, if you type the following command, the **3dparse** man page appears:

```
man 3dparse
```



13

wideip.conf Configuration

- Overview of the wideip.conf file
- Working with statements
- Working with comments
- Understanding current values

Overview of the wideip.conf file

The 3-DNS Controller configuration file is called **/etc/wideip.conf**. The **wideip.conf** file describes a network's data centers, servers (3-DNS Controllers, BIG-IP Controllers, EDGE-FX Caches, GLOBAL-SITE Controllers, and hosts), and wide IPs. The **wideip.conf** file consists of two types of information: statements and comments.

Your **wideip.conf** file must include at least the following definitions.

- A **datacenter** statement
- At least one **server** statement defining a 3-DNS Controller
- At least one virtual server, which is defined as part of a BIG-IP Controller, EDGE-FX Cache, or host **server** statement
- A **wideip** statement, for load balancing

If a **wideip.conf** file lacks complete definitions, one of the following happens:

- If the file cannot be parsed, **3dnsd** does not start.
- If the file can be parsed, 3-DNS Controllers revert to standard DNS behavior.

To open the wideip.conf file

1. At the command line, type **3dnsmaint**.
The 3-DNS Maintenance menu opens.
2. On the 3-DNS Maintenance menu, select **Edit 3-DNS Configuration**.

WARNING

*We do not recommend opening the **wideip.conf** file in a text editor. Instead, use the **Edit 3-DNS Configuration** command on the 3-DNS Maintenance menu. This command allows you to edit and save the configuration file. This command also parses the configuration file and alerts you to any syntax errors.*

Using include files

Include files are files that contain configuration information about one aspect of your network, and are listed in the main configuration file (**wideip.conf**). For example, you can have one include file that defines the BIG-IP Controllers in your network, and another include file that defines all wide IPs. Both files are listed in the **wideip.conf** file in place of the actual **server** and **wideip** statements.

Using include files reduces the size of the **wideip.conf** file and makes it easier to manage your configuration. Include files are automatically created and implemented whenever one of the following occurs:

- You configure your network setup using the Configuration utility.
- You perform any type of dumping operation. By default, dumping operations are **on**. To turn dumping **off**, set the **global** sub-statement **timer_persist_cache** to **0**.

◆ **Note**

When include files are generated, any comments you incorporated are deleted.

Syntax for include files

The following syntax is used when incorporating include files into a **wideip.conf** file.

```
include root_in "/config/3dns/include"
include root_out "/config/3dns/include"
include global <"file_name.inc">
include server <"file_name.inc">
include bigip <"file_name.inc">
include host <"file_name.inc">
include 3dns <"file_name.inc">
include datacenter <"file_name.inc">
include sync_group <"file_name.inc">
include wideip <"file_name.inc">
include 3dscrip <"file_name.inc">
include topology <"file_name.inc">
include geoloc <"netIana.inc">
include cdn <"cdn.inc">
include ldns <"ldns.inc">
include region <"file_name.inc">
include manifest <"file_name.inc">
```

Figure 13.1 Syntax for include files

Definitions of include statements

Table 13.1 lists the include statements, their descriptions, and their default file names.

Parameter	Description	Default
include root_in	Specifies the root directory from which to get include files. Enclose all file names in quotation marks.	"/config/3dns/include"
include root_out	Specifies the root directory in which to dump include files.	"/config/3dns/include"
include global	Specifies the name of the file that contains the globals statement.	"globals.inc"
include server	Specifies the name of the file that contains server statements.	"servers.inc"
include bigip	Specifies the name of the file that contains BIG-IP Controller server statements.	"bigip.inc"
include host	Specifies the name of the file that contains host server statements.	"host.inc"
include 3dns	Specifies the name of the file that contains 3-DNS Controller server statements.	"3dns.inc"
include datacenter	Specifies the name of the file that contains datacenter statements.	"datacenters.inc"
include sync_group	Specifies the name of the file that contains sync_group statements.	"sync_groups.inc"
include wideip	Specifies the name of the file that contains wideip statements.	"wideip.inc"
include 3dscrip	Specifies the name of the file that contains production rule configuration.	"prodrules.inc"
include topology	Specifies the name of the file that contains the topology statement.	"topology.inc"
include geoloc	Specifies the name of the file that contains the IP geo-classification database. It is important that you do not edit this statement.	"netlana.inc"
include ldns	Specifies the name of the file that contains information about local DNS servers and path information.	"ldns.inc"
include region	Specifies the name of the file that contains any region definitions statements.	"region.inc"
include cdn	Specifies the name of the file that contains any content delivery network (CDN) provider statements.	"cdn.inc"
include manifest	Specifies the name of the file that the Configuration utility uses to manage any production rules generated by the utility. It is important that you do not edit this statement.	"/config/3dns/ .prodruledb/manifest"

Table 13.1 Include file descriptions

Working with statements

A top-level 3-DNS Controller statement begins with a keyword, and may be followed either by a value or by a block of sub-statements enclosed in brackets (`{ }`).

The 3-DNS platform supports the following top-level statements.

- ◆ **include**
The **include** statement lists any include files that are configured on the 3-DNS Controller.
- ◆ **globals**
The **globals** statement defines system-level settings for any 3-DNS Controller configuration options and sets defaults for other statements.
- ◆ **server**
The **server** statement defines a 3-DNS Controller, a BIG-IP Controller, an EDGE-FX Cache, a GLOBAL-SITE Controller, or a host machine.
- ◆ **datacenter**
The **datacenter** statement defines the group of 3-DNS Controllers, BIG-IP Controllers, EDGE-FX Caches, GLOBAL-SITE Controllers, and hosts that reside in a single physical location.
- ◆ **sync_group**
The **sync_group** statement defines the group of 3-DNS Controllers that synchronize their configuration settings and metrics data.
- ◆ **wideip**
The **wideip** statement defines a wide IP. Wide IPs map a domain name to a load balancing mode and a set of virtual servers (on BIG-IP Controllers, EDGE-FX Caches, and other host machines).
- ◆ **topology**
The **topology** statement contains the topology records that facilitate the topology load balancing mode (on its own and as part of the Quality of Service mode). Note that while the topology statement is the preferred location for topology configuration information.

Syntax rules

Keep the following rules in mind when creating and editing statements in the **wideip.conf** file.

- ◆ **Statement order**
Statements should appear in this order in the **wideip.conf** file:
 - **globals**
 - most **include** statements (except the **include ldns** statement)
 - **server**

- **datacenter**
- **sync_group**
- **wideip**
- **include ldns**
- ◆ **Port specification**
When you define 3-DNS Controllers or hosts, the port specification must follow the **probe_protocol** sub-statement. When you define a BIG-IP Controller, EDGE-FX Cache, or virtual server (on a BIG-IP Controller, an EDGE-FX Cache, or a host), the port specification must immediately follow the address specification, and can take any of the following forms:

```
address <ip_addr>:<port>
address <ip_addr>
port <port>
address <ip_addr>
service <"wks">
```

In the above example, <"wks"> stands for well-known service and is a quoted string representing the name of a WKS defined in the `/etc/services` file. For example, you can use **"http"** instead of **80** to represent the HTTP protocol.

- ◆ **cur_values**
You may notice several **cur_values** in the **wideip.conf** file; do not edit them unless you are instructed to do so by your vendor's technical support. For more information, see *Understanding current values*, on page 13-47.

Typography in syntax examples

Certain characters are used to indicate whether a parameter is mandatory or optional, or whether you can use one parameter or another.

- ◆ **Mandatory parameters**
Angle brackets (< >) enclose mandatory parameters where you must type the data associated with a command.
- ◆ **Optional parameters**
Brackets ([]) enclose optional parameters.
- ◆ **Brackets**
Brackets ({ }) include the options available in a statement or sub-statement.
- ◆ **Choice of parameters**
A vertical bar (|) between two values means that either value is acceptable.

The globals statement

The **globals** statement sets up global options to be used by the 3-DNS Controller, and must appear before any other statements in the **wideip.conf** file. Each **globals** sub-statement has a default setting, and you do not need to edit the **globals** statement unless you want to change a default setting. If the 3-DNS Controller does not find a **globals** statement in the configuration file, the 3-DNS Controller uses a **globals** block, with each option set to its default.

If you use a **globals** sub-statement more than once, the 3-DNS Controller uses the last listed value and does not generate an error message. For example, if your **globals** statement contains the lines shown in the following figure, the 3-DNS Controller uses the value **50**.

```
globals {  
    bigip_ttl 100  
    bigip_ttl 50  
}
```

Figure 13.2 Multiple globals sub-statements

Syntax for the `globals` statement

The **globals** statement supports the following sub-statements. When you define a **globals** statement, you need only include those sub-statements that you want to change from the default.

```
globals {
  [ time_tolerance <number> ]
  [ encryption < yes | no > ]
  [ encryption_key_file <string> ]
  [ check_static_depends < yes | no > ]
  [ check_dynamic_depends < yes | no > ]
  [ default_persist_ttl < <number> s | m | h | d | w | m | y > ]
  [ default_probe_limit <number> ]
  [ persist_ldns < yes | no > ]
  [ persist_mask <ip address> ]
  [ bleed_requests < yes | no > ]
  [ persist_nonwipnames < yes | no > ]
  [ timer_get_3dns_data <number> ]
  [ timer_get_server_data <number> ]
  [ timer_get_host_data <number> ]
  [ timer_get_vs_data <number> ]
  [ timer_get_ecv_data <number> ]
  [ timer_get_path_data <number> ]
  [ timer_get_trace_data <number> ]
  [ timer_check_keep_alive <number> ]
  [ timer_persist_cache <number> ]
  [ timer_sync_state <number> ]
  [ dc_prefix <string> ]
  [ dns_ttl <number> ]
  [ 3dns_ttl <number> ]
  [ bigip_ttl <number> ]
  [ edgefx_ttl <number> ]
  [ host_ttl <number> ]
  [ vs_ttl <number> ]
  [ path_ttl <number> ]
  [ trace_ttl <number> ]
  [ default_ttl <number> ]
  [ rtt_timeout <number> ]
  [ rtt_sample_count <number> ]
  [ rtt_packet_length <number> ]
  [ rx_buf_size <number> ]
  [ tx_buf_size <number> ]
  [ dump_region < yes | no > ]
  [ dump_topology < yes | no > ]
  [ filter_ip < cidr block | ip address > ]
  [ filter_mask <ip address> ]
}
```

Figure 13.3 Syntax for the `globals` statement

```

[ qos_coeff_rtt <number> ]
[ qos_coeff_completion_rate <number> ]
[ qos_coeff_packet_rate <number> ]
[ qos_coeff_topology <number> ]
[ qos_coeff_hops <number> ]
[ qos_coeff_vs_capacity ]
[ qos_coeff_kbps <number>]
[ qos_factor_rtt <number> ]
[ qos_factor_completion_rate <number> ]
[ qos_factor_packet_rate <number> ]
[ qos_factor_topology <number> ]
[ qos_factor_hops <number> ]
[ qos_factor_vs_capacity <number> ]
[ qos_factor_kbps <number>
[ default_alternate < ga | null | random | ratio | static_persist |
packet_rate | leastconn | return_to_dns | rr | topology | vs_capacity
| kbps > ]
[ default_fallback < completion_rate | ga | hops | leastconn |
null | packet_rate | qos | random | ratio | return_to_dns |
rr | rtt | topology | vs_capacity | static_persist | kbps > ]
[ fb_respect_depends < yes | no > ]
[ fb_respect_acl < yes | no > ]
[ aol_aware < yes | no > ]
[ path_duration <number> ]
[ ldns_duration <number> ]
[ prober <ip_addr> ]
[ resolver_tx_buf_size <number> ]
[ resolver_rx_buf_size <number> ]
[ use_alternate_iq_port < yes | no > ]
[ multiplex_iq < yes | no > ]
[ paths_never_die < yes | no > ]
[ paths_noclobber < yes | no > ]
[ rtt_probe_dynamic < yes | no > ]
[ rtt_port_discovery < yes | no > ]
[ rtt_autorecover_discovery < yes | no > ]
[ rtt_discovery_method < short | wks | full > ]
[ discovery_portlist {
    method < short | wks | all >
    [ portlist { <number> <number> ... } ]
    [ randomize < yes | no > ]
} ]
[ rtt_allow_probe < yes | no > ]
[ rtt_allow_hops < yes | no > ]
[ rtt_allow_frag < yes | no > ]
[ rtt_probe_protocol < dns_rev | dns_dot | udp | tcp | icmp >
[ datasize_system <number> ]
[ datasize_reap_pct <number> ]
[ default_iquery_protocol < udp | tcp > ]
[ traceroute_port <number> ]
[ do_dynamic < yes | no > ]
}

```

Figure 13.3 Syntax for the *globals* statement

Figure 13.4 shows an example of a valid **globals** statement.

```
globals {
  prober 192.168.101.2    // Default prober is New York 3-DNS Controller
  encryption yes         // Encrypt iQuery
  paths_noclobber yes    // Don't overwrite metrics with zeroed results
  path_ttl 2400          // Extend the life of path metrics
  rtt_probe_dynamic yes  // Switch probing method if current fails
}
```

Figure 13.4 Example syntax for the **globals** statement

Definition of globals sub-statements

The **globals** sub-statements and their parameters are described in the following sections.

Synchronization

The synchronization sub-statement specifies how the current 3-DNS Controller handles synchronizing its database with the other 3-DNS Controllers in the network.

Parameter	Description	Default
time_tolerance	Specifies the variation of time allowed (in seconds) when comparing time stamps on files. The syncd daemon allows for slight variation in time stamps when it compares files during the synchronization process. If the difference between the two time stamps falls within the time_tolerance setting, the daemon considers the files to be the same and does not overwrite one with the other.	10

Table 13.2 Synchronization sub-statement

Encryption

The encryption sub-statements specify whether the communication between the 3-DNS Controller and a BIG-IP Controller is encrypted.

Parameter	Description	Default
encryption	Specifies whether to enable encryption for iQuery events.	no
encryption_key_file	Specifies the location and name of the iQuery encryption key file.	"/etc/F5key.dat"

Table 13.3 Encryption sub-statements

Dependencies

The dependencies sub-statements specifies whether the 3-DNS Controller checks the availability of virtual servers or paths before the controller sends a connection to a virtual server.

Parameter	Description	Default
check_static_depends	Specifies whether to check the availability of virtual servers on BIG-IP Controllers, EDGE-FX Caches, and hosts. Change this option to no if you want to test your configuration. Setting this option to no forces the virtual servers to have green (up) status indicators on the Virtual Server Statistics screen in the Configuration utility.	yes
check_dynamic_depends	Specifies that the 3-DNS Controller checks the availability of a path before it uses the path for load balancing. Changing this option to no overrides the path_ttl and whether the last probe attempt was successful.	yes

Table 13.4 *Dependencies sub-statement*

LDNS persistence

Dynamic load balancing modes depend on path information to resolve requests. The value for **persist_ldns** must be set to **yes** (the default) so that the 3-DNS Controller stores and uses path information. If you use only static load balancing modes, you can set **persist_ldns** to **no** to conserve memory.

Parameter	Description	Default
persist_ldns	Specifies whether the 3-DNS Controller records in its cache the IP addresses of all LDNS machines that make resolution requests.	yes

Table 13.5 *LDNS persistence sub-statement*

Load balancing persistence

The load balancing persistence sub-statements define how the 3-DNS Controller load balances persistent connections.

Parameter	Description	Default
default_persist_ttl	Specifies the length of time the 3-DNS Controller retains persistent connections information before the information is purged.	3600

Table 13.6 *Load balancing persistence sub-statements*

Parameter	Description	Default
persist_mask	Specifies the significant bits of an LDNS IP address to use with the static_persist load balancing mode.	0xFFFFFFFF
bleed_requests	Specifies whether load-balanced persistent connections are allowed to remain connected, until the TTL expires, when you disable a pool. When set to no , the connections are terminated immediately when the pool is disabled. This variable affects the persist setting in the load balancing sub-statement. See page 13-38 for more information.	yes

Table 13.6 Load balancing persistence sub-statements

Periodic task intervals

The periodic task interval sub-statements define the frequency at which the 3-DNS Controller refreshes the metrics information it collects.

Parameter	Description	Default
timer_get_3dns_data	Specifies how often the 3-DNS Controller retrieves performance data for other 3-DNS Controllers in the sync group. You can enter a value between 1 and 4294967295 seconds.	20
timer_get_server_data	Specifies how often the 3-DNS Controller refreshes BIG-IP Controller and EDGE-FX Cache information. You can enter a value between 1 and 4294967295 seconds.	20
timer_get_host_data	Specifies how often the 3-DNS Controller refreshes other host machine information. You can enter a value between 1 and 4294967295 seconds.	90
timer_get_vs_data	Specifies how often the 3-DNS Controller refreshes virtual server information. You can enter a value between 1 and 4294967295 seconds.	30
timer_get_path_data	Specifies how often the 3-DNS Controller refreshes path information (for example, round trip time or ping packet completion rate). You can enter a value between 1 and 4294967295 seconds.	120
timer_get_ecv_data	Specifies how often the 3-DNS Controller refreshes ECV information. You can enter a value between 5 and 4294967295 seconds.	90
timer_get_trace_data	Specifies how often the 3-DNS Controller retrieves traceroute data (traceroutes between each data center and each LDNS). You can enter a value between 1 and 4294967295 seconds.	60

Table 13.7 Periodic task interval sub-statements

Parameter	Description	Default
timer_check_keep_alive	Specifies how often the 3-DNS Controller queries remote 3-DNS Controllers and BIG-IP Controllers. This value determines how often 3dnsd sends hello packets to each big3d agent in its configuration. You can enter a value between 1 and 4294967295 seconds.	60
timer_persist_cache	Specifies how often the 3-DNS Controller writes the wideip.conf file from memory. You can enter a value between 1 and 4294967295 seconds.	300

Table 13.7 Periodic task interval sub-statements

Data time-outs

The data time-out sub-statements set the amount of time for which metrics information is considered valid. After a time-out is reached, the 3-DNS Controller refreshes the information.

Parameter	Description	Default
default_ttl	Specifies the default number of seconds that the 3-DNS Controller considers the wide IP A record to be valid. If you do not specify a wide IP TTL value when defining a wide IP pool, the wide IP definition uses the default_ttl value.	30
3dns_ttl	Specifies the number of seconds that the 3-DNS Controller considers performance data for the other 3-DNS Controllers to be valid.	60
bigip_ttl	Specifies the number of seconds that the 3-DNS Controller can use BIG-IP Controller information for name resolution and load balancing. You can enter a value between 1 and 4294967295 . The following relationship should be maintained: bigip_ttl > timer_get_server_data . A 2:1 ratio is the optimal setting for this relationship.	60
edgefx_ttl	Specifies the number of seconds that the 3-DNS Controller can use EDGE-FX Cache information for name resolution and load balancing. You can enter a value between 1 and 4294967295 . The following relationship should be maintained: edge_ttl > timer_get_server_data . A 2:1 ratio is the optimal setting for this relationship.	60
host_ttl	Specifies the number of seconds that the 3-DNS Controller can use generic host machine information for name resolution and load balancing. You can enter a value between 1 and 4294967295 . The following relationship should be maintained: host_ttl > timer_get_host_data .	240
vs_ttl	Specifies the number of seconds that the 3-DNS Controller can use virtual server information (data acquired from a BIG-IP Controller or other host machine about a virtual server) for name resolution and load balancing. You can enter a value between 1 and 4294967295 . The following relationship should be maintained: vs_ttl > timer_get_vs_data .	120

Table 13.8 Data time-outs sub-statements

Parameter	Description	Default
path_ttl	Specifies the number of seconds that the 3-DNS Controller can use path information for name resolution and load balancing. You can enter a value between 1 and 4294967295 . The following relationship should be maintained: path_ttl > timer_get_vs_data .	2400
trace_ttl	Specifies the amount of time (in seconds) that the 3-DNS Controller considers traceroute data to be valid. You can enter a value between 1 and 4294967295 .	604800 (seven days)

Table 13.8 *Data time-outs sub-statements*

Metrics collection

The metrics collection sub-statements define how the 3-DNS Controller collects path information.

Parameter	Description	Default
rtt_timeout	Specifies how long the big3d agent waits for a probe. You can enter a value between 1 and 4294967295 seconds.	5
rtt_sample_count	Specifies the number of packets to send from the BIG-IP Controller to the LDNS to determine the path information between those two machines. You can type a value between 1 and 25 .	3
rtt_packet_length	Specifies the length of packets, in bytes, to send from the BIG-IP Controller to the LDNS to determine the path information between those two machines. You can type a value between 64 and 500 ; the default value for this setting is 64 .	64
rtt_probe_protocol	Determines which protocols the 3-DNS Controller uses to probe LDNS servers to calculate RTT times, and in what order the protocols are used. You can specify the icmp , udp , tcp , dns_dot , or dns_rev protocols.	icmp

Table 13.9 *Metrics collection sub-statements*

Resource limits

The resource limits sub-statements define the amount of memory on the 3-DNS Controller that is allocated to sending and receiving metrics information.

Parameter	Description	Default
rx_buf_size	Specifies the maximum amount of socket buffer data memory the server can use when receiving data. You can enter a value between 8192 and 65536 .	49152
tx_buf_size	Specifies the maximum amount of socket buffer data memory the server can use when transmitting data. You can enter a value between 8192 and 65536 .	49152

Table 13.10 Resource limits sub-statements

QOS values

The Quality of Service load balancing mode distributes connections based on a path evaluation score. Using the equation below, the Quality of Service mode compares paths between the LDNS and each virtual server included in the **wideip** statement. The 3-DNS Controller load balances each new connection to the virtual server associated with the best (highest) path score.

```
score_path =
[(qos_coeff_packet_rate) * (1 / score_packet_rate)] +
(qos_coeff_rtt) * (1 / score_rtt)] +
[(qos_coeff_completion_rate) * (score_completion_rate)] +
[(qos_coeff_topology) * (score_topology)] +
[(qos_coeff_hops) * (score_hops)] +
[(qos_coeff_vs_capacity) * (score_vs_capacity)] +
[(qos_coeff_kbps) * (score_kbps)]
```

Figure 13.5 QOS equation

The coefficients for the score computation are defined as **globals**, but you can override them within a **wideip** statement.

Parameter	Description	Default
qos_coeff_rtt	Specifies the relative weighting for round trip time when the load balancing mode is set to Quality of Service. You can enter a value between 0 and 4294967295 .	20
qos_coeff_completion_rate	Specifies the relative weighting for ping packet completion rate when the load balancing mode is set to Quality of Service. You can enter a value between 0 and 4294967295 .	5

Table 13.11 QOS values sub-statements

Parameter	Description	Default
qos_coeff_packet_rate	Specifies the relative weighting for BIG-IP Controller packet rate when the load balancing mode is set to Quality of Service. You can enter a value between 0 and 4294967295 .	3
qos_coeff_topology	Specifies the relative weighting for topology when the load balancing mode is set to Quality of Service. You can enter a value between 0 and 4294967295 .	0
qos_coeff_hops	Specifies the relative weighting for hops when the load balancing mode is set to Quality of Service. You can enter a value between 0 and 4294967295 .	0
qos_coeff_vs_capacity	Specifies the relative weighting for virtual server capacity when the load balancing mode is set to Quality of Service. You can enter a value between 0 and 4294967295 .	0
qos_coeff_kbps	Specifies the relative weighting for kilobytes per second when the load balancing mode is set to Quality of Service. You can enter a value between 0 and 4294967295 .	3
qos_factor_rtt	Specifies the factor used to normalize raw round trip time values when computing the QOS score.	10000
qos_factor_completion_rate	Specifies the factor used to normalize raw completion rate values when computing the QOS score.	10000
qos_factor_packet_rate	Specifies the factor used to normalize raw packet rate values when computing the QOS score.	10000
qos_factor_topology	Specifies the factor used to normalize raw topology values when computing the QOS score.	10
qos_factor_hops	Specifies the factor used to normalize raw hops values when computing the QOS score.	25
qos_factor_vs_capacity	Specifies the factor used to normalize raw virtual server capacity values when computing the QOS score.	1
qos_factor_kbps	Specifies the factor used to normalize raw kilobytes per second values when computing the QOS score.	1

Table 13.11 QOS values sub-statements

Load balancing

The load balancing sub-statement defines the alternate and fallback load balancing modes.

Parameter	Description	Default
default_alternate	Defines the default alternate load balancing mode used when the preferred load balancing mode does not provide a resolution. You can override this setting in the wideip statement.	rr
default_fallback	Defines the default fallback load balancing mode used when the preferred and alternate load balancing modes do not provide a resolution. You can override this setting in the wideip statement.	return_to_dns
fb_respect_depends	Determines whether the 3-DNS Controller respects virtual server status when load balancing switches to the specified fallback mode.	no
fb_respect_acl	Determines whether the 3-DNS Controller imposes topology access control when load balancing switches to the specified fallback mode.	no
aol_aware	Determines whether the 3-DNS Controller recognizes local DNS servers that belong to the Internet service provider, America Online (AOL).	no

Table 13.12 *Load balancing sub-statements*

Prober

The prober sub-statement defines the IP address of the machine that pings a host system to verify whether it is available. Typically, you use the IP address of the 3-DNS Controller itself, but you can use other network servers.

Parameter	Description	Default
prober	Specifies the default prober for host status, usually the 3-DNS Controller IP address. Using this sub-statement is not necessary if the 3-DNS Controller only manages the BIG-IP Controller. When this option is set to 0 , the 3-DNS Controller's IP address is the implied value. This sub-statement can be overridden within the server statement.	0.0.0.0

Table 13.13 *Prober sub-statement*

◆ WARNING

*You must define a prober if the 3-DNS Controller manages virtual servers on hosts. If you do not define a default prober in the **globals** statement, or **probers** in the **host** statements for all hosts, you will encounter validation errors.*

Buffer size

The buffer size sub-statements specify the maximum amount of UDP data that the 3-DNS Controller can receive, and also specify the maximum amount of TCP data that the 3-DNS Controller can send.

Parameter	Description	Default
resolver_rx_buf_size	Specifies the UDP receive buffer size. The value is overridden only if it is larger than the one first assigned by the kernel.	98304
resolver_tx_buf_size	Specifies the TCP send buffer size.	24576

Table 13.14 Buffer size sub-statements

Reaping

The 3-DNS Controller stores dynamic LDNS and network path data in memory. The amount of data that can be held in memory at any given time is based on the amount of memory in the 3-DNS Controller. **Reaping** is the process of finding the least-used data in memory and deleting it.

The default reaping values are adequate for most configurations. Contact your technical support representative if you want to make changes to them.

Parameter	Description	Default
datasize_system	Specifies the amount of RAM that the 3-DNS Controller reserves for system usage, such as non-3-DNS specific processes.	if 64 MB RAM, default is 32; otherwise default is 64
datasize_reap_pct	Specifies what percentage of memory that the 3-DNS Controller frees up during the reap process.	15
path_duration	Specifies the number of seconds that a path remains cached after its last access. You can type a value between 60 and 2147483648 .	604800 (7 days)
ldns_duration	Specifies the number of seconds that an inactive LDNS remains cached. Each time an LDNS makes a request, the clock starts again. You can type a value between 60 and 2147483648 .	2419200 (28 days)

Table 13.15 Reaping sub-statements

iQuery port options

The iQuery port options determine which port (or ports) the 3-DNS Controller uses to send and receive iQuery traffic.

Parameter	Description	Default
use_alternate_iq_port	Determines whether the 3-DNS Controller runs iQuery traffic on port 245 (the port used in older configurations), or on port 4353 , the iQuery port registered with IANA. The default setting, yes , uses port 4353 . To use port 245 , change this setting to no .	yes
multiplex_iq	Determines whether the 3-DNS Controller uses the ephemeral ports for iQuery traffic returned from the big3d agent. The default setting forces iQuery traffic to use a single port defined by use_alternate_iq_port for all incoming iQuery traffic.	yes

Table 13.16 *iQuery port options sub-statements*

Probing

The 3-DNS Controller uses probing to collect path metrics. The 3-DNS Controller then uses the metrics to make traffic distribution and load balancing decisions.

Parameter	Description	Default
default_probe_limit	Specifies a limit on the number of times the 3-DNS Controller probes a path. With the default setting, there is no limit on path probes.	0
paths_never_die	Specifies that dynamic load balancing modes can use path data even after the TTL for the path data has expired. We recommend that you change this setting to yes , which has the effect of requiring that the 3-DNS Controller always uses path data even if the path's TTL expires.	no
paths_noclobber	Specifies whether the 3-DNS Controller overwrites existing path data with blank data when a path probe fails. With the default setting, the 3-DNS Controller does not overwrite existing path data with blank data when a path probe fails. Unlike paths_never_die , this parameter has no effect on path_ttl .	yes
check_dynamic_depends	Specifies that the 3-DNS Controller checks the availability of a path before it uses the path for load balancing. Changing this option to no overrides the path_ttl and whether the last probe attempt was successful. You can use this parameter in conjunction with paths_noclobber . This parameter does not prevent the refreshing of path metrics.	yes
rtt_port_discovery	Determines whether the 3-DNS Controller uses the discovery factory to find an alternate port to be used by the probing factory, if probing on port 53 fails.	no

Table 13.17 *Probing sub-statements*

Parameter	Description	Default
rtt_autorecover_discovery	Specifies whether to move collected LDNS information from the Needs Discovery probing state to the Needs Probe state if rtt_port_discovery is set to no . Setting this parameter to no means that if probing fails for a specific LDNS for any reason, the LDNS is ineligible for future probing attempts.	yes
rtt_discovery_method	Determines which ports to scan. The default, short , causes the 3-DNS Controller to scan a pre-defined list of ports, and then scans port 53 . Other acceptable values are wks (well-known services), and all .	short
rtt_probe_dynamic	Determines whether the 3-DNS Controller scans the ports specified in rtt_discovery_method in the same order every time, or in a random order. The default setting causes the controller to scan the ports in the same order every time.	no
rtt_allow_probes	Specifies that the 3-DNS Controller issues probe requests for path metrics to local DNS servers. You can change this setting to no to turn off path probing.	yes
rtt_allow_hops	Specifies that the 3-DNS Controller should collect hops metrics when probing paths.	yes
rtt_allow_frag	Specifies that the 3-DNS Controller should break each probe packet into smaller packets when probing paths.	no
probe_protocol	Specifies the protocol that the 3-DNS Controller uses to probe local DNS servers. The default is icmp . The other available protocols are: dns_rev , dns_dot , udp , and tcp	icmp

Table 13.17 Probing sub-statements

The server statement

The **server** statement defines the characteristics associated with a particular 3-DNS Controller, BIG-IP Controller, EDGE-FX Cache, GLOBAL-SITE Controller, or host.

◆ Note

*The **server** statement replaces the **bigip** and **host** statements used in earlier versions of 3-DNS Controller. Although this version of 3-DNS Controller provides backward compatibility with the earlier **bigip** and **host** statements, we recommend that you use the newer syntax.*

A **server** statement contains the following information:

- The type of server: 3-DNS Controller, BIG-IP Controller, EDGE-FX Cache, GLOBAL-SITE Controller, or host
- The IP address and host name of the server

- If the server is a BIG-IP Controller, EDGE-FX Cache, or host, the set of virtual servers that is available on it
- Dynamically collected information about the server, its virtual servers and ports, and the paths between the server and LDNS

Because available sub-statements vary by server type, the syntax and examples for each type are listed separately. All sub-statements are defined in the table starting on page 13-27.

Syntax for the server statement (3-DNS Controller)

The following **server** statement syntax applies to 3-DNS Controllers only. Note that this **server** statement does not define virtual servers; the purpose of defining a 3-DNS Controller is to set up the **big3d** agent to obtain path probing information.

```
server {
  type 3dns
  address <IP address>
  name <"3dns_name">
  iquery_protocol [ udp | tcp ]
  [ remote {
    secure <yes | no>
    user <"user name">
  } ]
  [ interface {
    address <NIC IP address>
    address <NIC IP address>
  } ]
  [ factories {
    prober <number>
    discovery <number>
    snmp <number>
    hops <number>
  } ]
  [ prober <IP address> ]
  probe_protocol < icmp | udp | tcp | dns_rev | dns_dot >
  port <port to probe>
  [ discovery_portlist {
    method < short | wks | all >
    [portlist { <number> <number> ... }
    [randomize < yes | no >
  } ]
} ]
}
```

Figure 13.6 Server statement syntax for defining a 3-DNS Controller

Figure 13.7 shows an example of the syntax to use in defining a 3-DNS Controller.

```
// New York
server {
    type 3dns
    address 192.168.101.2
    name "3dns-newyork"
    iquery_protocol udp
    remote {
        secure no
        user "root"
    }
    probe_protocol icmp
    port 53
}
```

Figure 13.7 Example syntax for defining a 3-DNS Controller

Syntax for the server statement (BIG-IP Controller)

The following **server** statement syntax applies to BIG-IP Controllers and their virtual servers only.

```
server {
    type bigip
    address <IP address>
    name <"bigip_name">
    iquery_protocol [ udp | tcp ]
    [ vsmetrics < yes | no > ]
    [ remote {
        secure <yes | no>
        user <"user name">
    } ]
    [ interface {
        address <NIC IP address>
        address <NIC IP address>
    } ]
    [ limit {
        [ kbytes_per_sec <number> ]
        [ pkts_per_sec <number> ]
        [ current_conns <number> ]
    } ]
    [ probe_protocol < icmp | ucp | tcp | dns_dot | dns_rev > ]
    [ ratio <number> ]
    [ factories {
        prober <number>
        discovery <number>
        snmp <number>
        hops <number>
    } ]
}
```

Figure 13.8 Server statement syntax for defining a BIG-IP Controller

```
vs {
  address <virtual server IP address>
  port <port number> | service <"service name">
  [ limit {
    [ kbytes_per_sec <number> ]
    [ pkts_per_sec <number> ]
    [ current_conns <number> ]
  } ]
  [ depends_on {
    <IP address>:<port number> //example 10.10.10.10:443
  } ]
  [ translate {
    <IP address>:<port number>
  } ]
  probe_protocol <tcp | udp>
}
```

Figure 13.8 Server statement syntax for defining a BIG-IP Controller

Figure 13.9 shows an example of the syntax to use in defining a BIG-IP Controller.

```
server {
    type            bigip
    address         192.168.101.40
    name            "bigip-newyork"
    iquery_protocol udp

    remote {
        secure      yes
        user         "administrator"
    }

    # Tell 3-DNS about the 2 interfaces on a BIG-IP HA
    Controller
    interface {
        address     192.168.101.41
        address     192.168.101.42
    }

    # Change the number of factories doing the work at big3d
    factories {
        prober      6
        discovery   1
        snmp        1
        hops        2
    }

    vs {
        address     192.168.101.50
        service     "http"
        translate {
            address  10.0.0.50
            port     80
        }
    }

    vs {
        address     192.168.101.50:25 // smtp
        translate {
            address  10.0.0.50:25
        }
    }
}
```

Figure 13.9 Example syntax for defining a BIG-IP Controller

Syntax for the server statement (EDGE-FX Cache)

This **server** statement syntax applies to EDGE-FX Caches only.

```
server {
  type edgefx
  address <IP address>
  name <"edgefx_name">
  [ limit {
    [ kbytes_per_sec <number> ]
    [ pkts_per_sec <number> ]
    [ current_conns <number> ]
    [ cpu_avail <number> ]
    [ disk_avail <number> ]
    [ mem_avail <number> ]
  } ]
  iquery_protocol [ udp | tcp ]
  [ remote {
    secure <yes | no>
    user <"user name">
  } ]
  [ ratio <number> ]
  [ factories {
    prober <number>
    discovery <number>
    hops <number>
    snmp <l> { //required
      agent edgefx
      version 2
      community <"public">
    }
  } ]
  vs {
    address <virtual server IP address>
    port <port number> | service <"service name">
    [ limit {
      [ cpu_avail <number> ]
      [ disk_avail <number> ]
      [ mem_avail <number> ]
      [ kbytes_per_sec <number> ]
      [ pkts_per_sec <number> ]
      [ current_conns <number> ]
    } ]
  }
}
```

Figure 13.10 Example syntax for defining an EDGE-FX Cache

Syntax for the server statement (GLOBAL-SITE Controller)

The following **server** statement syntax applies to GLOBAL-SITE Controllers only.

```
server {
  type gsite
  address <IP address>
  name <"gsite_name">
  iquery_protocol [ udp | tcp ]
  [ remote {
    secure <yes | no>
    user <"user name">
  } ]
  [ probe_protocol < icmp | ucp | tcp | dns_dot | dns_rev > ]
  [ factories {
    prober <number>
    discovery <number>
    hops <number>
    snmp <number>
  } ]
}
```

Figure 13.11 Example syntax for defining a GLOBAL-SITE Controller

Syntax for the server statement (host)

The following **server** statement syntax applies to hosts only. Note that the **snmp** sub-statement is necessary only if you want the **big3d** agent to use an SNMP agent on the host to collect additional metrics information. For more information on configuring these settings, see Chapter 10, *SNMP*.


```
server {
    type host
    address <IP address>
    name <"host_name">
    probe_protocol <tcp | icmp | dns_rev | dns_dot>
    [ prober <IP address> ]
    port <port number> | service <"service name">
    [ snmp {
        agent <generic | ucd | solstice | ntserve | win2kserve |
        ciscold | ciscold2 | ciscold3 | foundry | arrowpoint | alteon
        | cacheflow>
        port <port number>
        community <"community string">
        timeout <seconds>
        retries <number>
        version <SNMP version>
    } ]
    [ limit {
        [ kbytes_per_sec <number> ]
        [ pkts_per_sec <number> ]
        [ current_conns <number> ]
        [ cpu_avail <number> ]
        [ disk_avail <number> ]
        [ mem_avail <number> ]
    } ]
    vs {
        address <virtual server IP address>
        port <port number> | service <"service name">
        [ probe_protocol <tcp | icmp | dns_rev | dns_dot> ]
    }
    [ limit {
        [ kbytes_per_sec <number> ]
        [ pkts_per_sec <number> ]
        [ current_conns <number> ]
        [ cpu_avail <number> ]
        [ disk_avail <number> ]
        [ mem_avail <number> ]
    } ]
}
```

Figure 13.12 Server statement syntax for defining a host

Figure 13.13 shows an example of the syntax to use in defining a host.

```
server {
    type          host
    address       192.168.104.40
    name         "host-tokyo"
    probe_protocol icmp
    port         53
    snmp {
        agent     ucd
        community "public"
        version   1
    }
    vs {
        address    192.168.104.50:25
        limit {
            kbytes_per_second 15000
        }
    }
    vs {
        address    192.168.104.50:80
        limit {
            kbytes_per_second 15000
        }
    }
}
```

Figure 13.13 Example syntax for defining a host

Definition of server sub-statements

The **server** statement supports the following sub-statements. Note that available sub-statements vary by server type.

Address information

The address information sub-statements specify the name, address, and type of each server. Depending on the type of server you are configuring, you may need to specify a probe protocol, prober IP address, and port number.

Table 13.18 lists the parameters of the address information sub-statement.

Parameter	Description
type	Indicates whether the specified server is a 3-DNS Controller, BIG-IP Controller, EDGE-FX Cache, GLOBAL-SITE Controller, or host.
address	Specifies the IP address of the 3-DNS Controller, BIG-IP Controller, EDGE-FX Cache, GLOBAL-SITE Controller, or host.
name	Specifies the name of the 3-DNS Controller, BIG-IP Controller, EDGE-FX Cache, GLOBAL-SITE Controller, or host. You must enclose all names in quotation marks.

Table 13.18 Address information sub-statements

Parameter	Description
iQuery_protocol	Specifies the iQuery transport option, TCP or UDP.
probe_protocol	Specifies the protocol method to use for probing this host: ICMP, UDP, or TCP. Applies to 3-DNS Controllers and hosts. Note that UDP is not supported on hosts.
prober	Specifies the IP address of the machine probing the host. This IP address points to a BIG-IP Controller, a 3-DNS Controller, a GLOBAL-SITE Controller, or an EDGE-FX Cache that runs the big3d agent. The big3d agent actually probes the host and virtual servers to verify whether the host or a particular virtual server is currently available to accept connections. If you omit this parameter, the 3-DNS Controller uses the prober <ip_addr> parameter defined in the globals statement. This applies to hosts only.
port	Specifies the port used to probe this host if the probe_protocol parameter is set to TCP. This applies to hosts only.

Table 13.18 Address information sub-statements

Limit settings

Using the **limit** sub-statement, you can manage the physical and throughput resources of your BIG-IP Controllers, EDGE-FX Caches, hosts, and their respective virtual servers. If you omit this sub-statement, the 3-DNS Controller does not use resource thresholds to monitor the availability of the BIG-IP Controllers, EDGE-FX Caches, hosts, and their respective virtual servers.

Parameter	Description
limits	Indicates the start of the limits definition. Applies to BIG-IP Controllers and their virtual servers, EDGE-FX Caches and their virtual servers, and hosts and their virtual servers.
cpu_avail	Specifies, in percentage, how much CPU processing must remain available on the server or virtual server. The cpu_avail parameter applies to hosts and EDGE-FX Caches only.
mem_avail	Specifies, in kilobytes, how much memory must remain available on the server or virtual server. The mem_avail parameter applies to hosts and EDGE-FX Caches only.
disk_avail	Specifies, in kilobytes, how much disk space must remain available on the server or virtual server. The disk_avail parameter applies to hosts and EDGE-FX Caches only.
kbytes_per_sec	Specifies, in kilobytes per second, the maximum allowable throughput rate for the server or virtual server.
pkts_per_sec	Specifies, in packets per second, the maximum allowable data transfer rate for the server or virtual server.
current_conn	Specifies the maximum number of current connections for the server or virtual server.

Table 13.19 Limit sub-statement

Remote connections

It is necessary to use the **remote** sub-statement only if you want to specify a different login name or specifically use SSH or RSH on 3-DNS Controllers, BIG-IP Controllers, or EDGE-FX Caches.

Parameter	Description
remote	Indicates the start of a remote sub-statement. Applies to 3-DNS Controllers, BIG-IP Controllers, GLOBAL-SITE Controllers, and EDGE-FX Caches.
secure	Specifies whether to use SSH (secure shell) or RSH (remote shell) for remote connections. The default for crypto controllers is yes , which specifies that SSH is used. Non-crypto controller versions must use RSH instead. Applies to 3-DNS Controllers, BIG-IP Controllers, and EDGE-FX Caches.
user	Specifies the "superuser" name that is used to allow a remote user to log on to the controller. Enclose this name in quotation marks. If you omit this parameter, the default, " root ", is used. Applies to 3-DNS Controllers, BIG-IP Controllers, GLOBAL-SITE Controllers, and EDGE-FX Caches.

Table 13.20 Remote connections sub-statements

Hardware redundancy

If you have hardware-redundant 3-DNS Controllers and BIG-IP Controllers, you must configure the **interface** sub-statement for the 3-DNS Controller to work properly with BIG-IP Controllers in Active-Active mode. This sub-statement is also required in using the standby BIG-IP Controller or 3-DNS Controller for probing.

Parameter	Description
interface	Indicates the start of the interface sub-statement.
address	Specifies the IP address of both network interface cards, on separate lines. Applies to 3-DNS Controllers and BIG-IP Controllers.

Table 13.21 Hardware redundancy sub-statements

Factories

With 3-DNS Controllers, BIG-IP Controllers, GLOBAL-SITE Controllers, and EDGE-FX Caches, you can change the number and types of probing factories by using the **factories** sub-statement. If you omit this sub-statement, the defaults are used. For more information on probing, see Chapter 3, *The big3d Agent*.

Parameter	Description
factories	Indicates the start of the factories definition. Applies to 3-DNS Controllers, BIG-IP Controllers, EDGE-FX Caches, and GLOBAL-SITE Controllers.
prober	Specifies the number of prober factories to use.
discovery	Specifies the number of discovery factories to use.
snmp	Specifies the number of SNMP factories to use. Note that you must use an SNMP factory to collect metrics from an EDGE-FX Cache.
hops	Specifies the number of hops factories to use.

Table 13.22 *Factories sub-statements*

SNMP settings

The **snmp** sub-statement is valid for hosts and EDGE-FX Caches only. This sub-statement instructs the **big3d** agent to use an SNMP agent on the host or cache to collect additional metrics information.

If you need help configuring the SNMP agent on the EDGE-FX Cache, refer to the *EDGE-FX Administrator Guide*. If you need help configuring the SNMP agent on the host, refer to the documentation provided with the host.

Parameter	Description
snmp	Specifies the start of an SNMP definition. Applies to hosts and EDGE-FX Caches only.
agent	Specifies the SNMP agent type. If you omit this parameter, the big3d agent uses the generic SNMP agent. Applies to hosts and EDGE-FX Caches only.
port	Specifies the port the SNMP agent runs on. Applies to hosts and EDGE-FX Caches only.
community	Specifies the password for basic SNMP security and for grouping SNMP hosts. Enclose this string in quotation marks. Applies to hosts and EDGE-FX Caches only.

Table 13.23 *SNMP sub-statements*

Parameter	Description
timeout	Specifies the amount of time (in seconds) for the timeout. The default is appropriate in most cases. If you are contacting a host through a very slow network, you can try increasing the timeout and retries values to improve performance. However, the problem with increasing these values is that a host that is down may hold up other SNMP responses for an excessive amount of time. Applies to hosts only.
retries	Specifies the number of times requests should be retried. The default is appropriate in most cases. If you are contacting a host through a very slow network, you can try increasing the timeout and retries values to improve performance. However, the problem with increasing these values is that a host that is down may hold up other SNMP responses for an excessive amount of time. Applies to hosts only.
version	Specifies the SNMP agent version number. Applies to hosts only.

Table 13.23 *SNMP sub-statements*

Virtual server definitions

Part of defining a BIG-IP Controller, EDGE-FX Cache, or host is defining the virtual servers that the server manages. After you define a virtual server here (including specifying the address and port), you can use this virtual server in a **wideip** definition.

Parameter	Description
vs	Indicates the start of a virtual server definition.
address	Specifies the IP address of the virtual server. Note that the virtual server's address must be listed first, before port or service values.
port or service	Specifies the virtual server's port number or service name. You can add the port number, preceded by a colon, on the same line as the virtual server's address, or you can enter it on the next line. You can use the service name if it is a WKS (well known service) and you enclose it in quotation marks.
limit	Specifies resource thresholds for the virtual server. Note that if a virtual server reaches a limit, the virtual server is marked as unavailable for load balancing.
depends_on	Specifies the IP address and port of other virtual servers that must also be available for load balancing (up status) before the 3-DNS Controller uses this virtual server for load balancing.
probe_protocol	Specifies the protocol to use for probing this virtual server: ICMP or TCP.
translate	Specifies that iQuery packets sent to the BIG-IP Controller include translated IP addresses (required if the packets must pass through a firewall). When you use this keyword, you must then include address and port/service information for the translated IP addresses. Applies to BIG-IP Controllers only.

Table 13.24 *Virtual server definitions*

The datacenter statement

A **datacenter** statement defines the group of 3-DNS Controllers, BIG-IP Controllers, EDGE-FX Caches, GLOBAL-SITE Controllers, and hosts that reside in a single physical location.

Syntax for the datacenter statement

The **datacenter** statement uses the following syntax.

```
datacenter {
  name <"data center name">
  [ location <"location info"> ]
  [ contact <"contact info"> ]
  [ 3dns <IP address | name> ]
  [ bigip <IP address | name> ]
  [ host <IP address | name> ]
  [ edgefx <IP address | name> ]
  [ gsite <IP address | name> ]
}
```

Figure 13.14 Syntax for the *datacenter* statement

Figure 13.15 shows an example of a valid **datacenter** statement.

```
datacenter {
  name "New York"
  location "NYC"
  contact "3DNS_Admin"
  3dns 192.168.101.2
  bigip 192.168.101.40
  edgefx 192.168.101.50
  host 192.168.105.40
  gsite 192.168.101.70
}
```

Figure 13.15 Example syntax for the *datacenter* statement

Definition of datacenter sub-statements

The **datacenter** sub-statements specify a name for the data center and the machines it contains.

Parameter	Description
name	Specifies the name of this data center. The name must be enclosed in quotation marks.
location	Specifies the location of the data center. This name must be enclosed in quotation marks. This sub-statement is not required, but this information can be useful if problems later arise or changes are required.

Table 13.25 Data center sub-statements

Parameter	Description
contact	Identifies the administrator of the data center. This name must be enclosed in quotation marks. This sub-statement is not required, but this information can be useful if problems later arise or changes are required.
3dns	Specifies the IP address of a 3-DNS Controller in this data center.
bigip	Specifies the IP address of a BIG-IP Controller in this data center.
edgefx	Specifies the IP address of an EDGE-FX Cache in this data center.
gsite	Specifies the IP address of a GLOBAL-SITE Controller in this data center.
host	Specifies the IP address of a host in this data center.

Table 13.25 Data center sub-statements

The sync_group statement

The **sync_group** statement defines the group of 3-DNS Controllers that synchronize their configuration settings and metrics data. You configure this statement in the **wideip.conf** file of the principal 3-DNS Controller.

Syntax for the sync_group statement

The **sync_group** statement uses the following syntax.

```
sync_group {
    name <"name">
    3dns <ip_address | "domain_name">
    [ 3dns <ip_address | "domain_name"> ]
}
```

Figure 13.16 Syntax for the **sync_group** statement

Note that the **sync_group** statement does not support location or contact sub-statements.

Figure 13.17 shows an example of a valid **sync_group** statement.

```
sync_group {
    name "sync"
    3dns 192.168.101.2 // New York - this is the principal controller
    3dns 192.168.102.2 // Los Angeles - this is a receiver controller
    3dns 192.168.103.2 // Madrid - this is also a receiver controller
}
```

Figure 13.17 Example syntax for the **sync_group** statement

Definition of sync_group sub-statements

The **sync_group** sub-statements define the members of the sync group.

Parameter	Description
name	Specifies the name of this sync group.
3dns	Specifies the IP address or domain name (enclosed in quotation marks) of a 3-DNS Controller in the group. First list the IP address of the principal controller. Then list all other 3-DNS Controllers, in the order that they should become a principal controller, if the previously listed principal 3-DNS Controller fails. Note that there can only be one principal controller at any time.

Table 13.26 Sync_group sub-statements

The wide IP statement

The **wideip** statement defines a wide IP. A wide IP maps a domain name to a load balancing mode and a set of virtual servers (on BIG-IP Controllers, EDGE-FX Caches, and/or other host machines).

Syntax for the wideip statement

The **wideip** statement uses the following syntax.

```

wideip {
    address <ip_address>
    port <port_number> | <"service name">
    [ ttl <number> ]
    [ persist < yes | no > ]
    [ persist_ttl <number> ]
    name <"domain_name">
    [ alias <"alias_name"> ]
    [ port_list <port_number> <port_number> ... ]
    [ qos_coeff {
        rtt <number>
        hops <number>
        completion_rate <number>
        packet_rate <number>
        vs_capacity <number>
        topology <number>
        kbps <number>
    } ]
    [ pool_lbmode <rr | ratio | ga | random | topology> ]
    [ ecv {
        protocol <none | ftp | http | https>
        file_name <string>
        user <string>
        password <string>
        hashed_password <string>
        scan_level <none | all | first>
        transfer_amount <number>
        connection_timeout <number>
        transfer_timeout <number>
    } ]
    pool {
        name <"pool_name">
        [ ttl <number> ]
        [ ratio <number> ]
        [ last_resort <yes | no> ]
        [ check_static_depends < yes | no > ]
        [ check_dynamic_depends < yes | no > ]
        [ limit {
            [ kbytes_per_sec <number> ]
            [ pkts_per_sec <number> ]
            [ current_conns <number> ]
            [ cpu_avail <number> ]
            [ disk_avail <number> ]
            [ mem_avail <number> ]
        } ]
    }
    [ type <A | NS | CDN> ]
    [ CDN <string> ]
    [ CNAME <canonical name> ]
    [ ZNAME <zone name> ]

```

Figure 13.18 Syntax for the *wideip* statement

```

[ ratio <pool_ratio> ]
[ dynamic_ratio < yes | no > ]
[ rr_ldns < yes | no > ]
[ rr_ldns_limit <number> ]
[ preferred < completion_rate | ga | hops | leastconn |
packet_rate | qos | random | ratio | return_to_dns | rr | rtt
| topology | vs_capacity | null | static_persist | kbps>]
[ alternate < ga | null | random | ratio | return_to_dns |
rr | topology | packet_rate| leastconn | vs_capacity |
static_persist> ]
[ fallback < completion_rate | ga | hops | leastconn |
packet_rate | qos | random | ratio | return_to_dns | rr | rtt
| topology | vs_capacity | null | static_persist | kbps> ]
    address <vs IP address:port> [ratio <weight>]
  }
}

```

Figure 13.18 Syntax for the *wideip* statement

Figure 13.19 shows an example of a valid **wideip** statement.

```

wideip {
  address          192.168.102.50
  service          "http"
  name             "http.wip.domain.com"
  alias            "store.wip.domain.com"
                  "*.wip.domain.com"
                  "http.wip.domain.???"
  pool_lbmode     ratio
  pool {
    name          "pool_1"
    ratio         3
    limit {
      kbytes_per_second 10000
    }
    preferred     rtt
    alternate     random
    address       192.168.101.50
    address       192.168.102.50
    address       192.168.103.50
  }
  pool {
    name          "pool_2"
    ratio         1
    limit {
      kbytes_per_second 10000
    }
    preferred     ratio
    address       192.168.104.50   ratio 2
    address       192.168.105.50   ratio 1
  }
}

```

Figure 13.19 Example syntax for the *wideip* statement

Definition of wideip sub-statements

The **wideip** sub-statements define groups of virtual servers to be load balanced, and they assign load balancing characteristics, such as the load balancing mode, to each group.

Address information

The address information sub-statements specify the IP address, name, and alias of the wide IP. They also specify the pool of virtual servers that the wide IP load balances.

Parameter	Description
address	Specifies the IP address registered with InterNIC that corresponds to the wide IP name. This IP address is also listed as the A record in the zone file for the domain.
port or service	Specifies the default port number or service name for the wide IP. You can use the service name if it is a well known service (WKS) and you enclose it in quotation marks.
name	Specifies the fully qualified domain name for the wide IP (for example, " www.wip.domain.com "). You must enclose all names in quotation marks. Note that you can use two wildcard characters, the asterisk (*) and the question mark (?), in wide IP names. The asterisk (*) can represent multiple characters, and the question mark (?) can represent a single character. Any of the following examples are valid for the name or alias parameter in a wideip statement: " www*.com ", " *.domain.com ", " *.domain.??? ", and so on.
alias	Specifies an alternate name for the wide IP. You must enclose all names in quotation marks. Alias names are treated the same as the domain name. Note that you can use two wildcard characters, the asterisk (*) and the question mark (?), in wide IP names. The asterisk (*) can represent multiple characters, and the question mark (?) can represent a single character. Any of the following examples are valid for the name or alias parameter in a wideip statement: " www*.com ", " *.domain.com ", " *.domain.??? ", and so on. You can specify an unlimited number of alias names for each wide IP.

Table 13.27 Address information sub-statements

Load balancing sub-statements

The load balancing sub-statements denote the general load balancing attributes for all pools in the **wideip.conf** file.

Parameter	Description
ttl	Specifies the amount of time (in seconds) that the A record is used by the LDNS after resolving the wide IP. This is the TTL associated with the A record as specified by RFC 1035.
persist	Specifies whether to maintain a persistent connection between an LDNS and a particular virtual server in the wide IP (rather than load-balancing the connection to any available virtual server). Note that the variables bleed_requests and default_persist_ttl , in the globals statement, affect this setting. See page 13-10 for more information.
persist_ttl	Specifies the number of seconds to maintain a persistent connection between an LDNS and a particular virtual server in this wide IP; this setting is valid only if you have configured the persist parameter.
port_list	Specifies a list of ports that must be available before the 3-DNS Controller can send connections to the specified address.
qos_coeff	Specifies the relative weighting for each load balancing method in calculating the Quality of Service mode. Before you adjust any QOS coefficients, you may want to review Chapter 7, <i>Additional Load Balancing Options</i> , in the 3-DNS Administrator Guide .
pool_lbmode	Specifies the load balancing mode to use to balance requests over all pools.
ecv	Specifies an extended content verification (ECV) monitor for a virtual server in a pool.
protocol	Specifies the protocol to use for the ECV. You can use only http , https , or ftp . Use only with the ecv sub-statement.
file_name	Specifies the name of the object to retrieve. Use only with the ecv sub-statement.
user	Specifies the user name that you use to log in to the service. Use only with the ecv sub-statement.
password	Specifies the password that corresponds to the user account. Use only with the ecv sub-statement.
hashed_password	Specifies the password in encrypted characters. Use only with the ecv sub-statement.
scan_level	Specifies whether you want to scan just through the configured wide IP names, or through the wide IP names and aliases. Use only with the ecv sub-statement. Note that if you use wildcard characters in the wide IP name or alias parameters, those names and aliases are ignored by the ECV scans.
transfer_amount	Specifies the number of bytes to transfer. Use only with the ecv sub-statement.

Table 13.28 *Load balancing sub-statements*

Parameter	Description
transfer_timeout	Specifies the maximum amount of time the file information transfer should take. Use only with the ecv sub-statement.
connection_timeout	Specifies the maximum amount of time to connect to a service. Use only with the ecv sub-statement.

Table 13.28 *Load balancing sub-statements*

Pool sub-statements

The **pool** sub-statements define the virtual servers, and the load balancing modes within the pool, that the 3-DNS Controller uses to respond to DNS requests. Note that you can have one or more pools in a wide IP definition.

Parameter	Description
pool	Indicates the start of the pool definition for this wide IP. A pool is a set of virtual servers defined and owned by a BIG-IP Controller, EDGE-FX Cache, or host machine.
name	As part of a pool definition, defines the name of the pool. All names must be enclosed in quotation marks.
ttd	Specifies the amount of time (in seconds) that the A record is used by the LDNS after resolving the wide IP. This is the TTL associated with the A record as specified by RFC 1035.
ratio	As part of a pool definition, ratio specifies the default weighting to use, with respect to other pool types, when the pool_lbmode is ratio .
last_resort	Specifies whether the 3-DNS Controller directs LDNS requests to this pool when no other pools in the wide IP successfully respond to the request. The default setting is no .
check_static_depends	Specifies whether the 3-DNS Controller checks availability before returning a virtual server in the pool. (Note that this parameter does not affect the status of the virtual server on the Virtual Server Statistics screen, in the Configuration utility, while the global variable of the same name does affect the status.)
check_dynamic_depends	Specifies whether the 3-DNS Controller checks paths before returning a virtual server in the pool.
type	Specifies the type of pool. The default is A . However, you can specify NS if you want to redirect LDNS requests to other name servers. You can also use CDN to redirect LDNS requests to a CDN provider in the cdn.inc file.
cdn	Specifies the name of the CDN provider to redirect LDNS requests to, as listed in the cdn.inc file. Use this attribute only if you specify the pool type as CDN .

Table 13.29 *Pool sub-statements*

Parameter	Description
<code>cname</code>	Specifies the canonical name (cname) for the pool. Use this attribute with the pool type NS or CDN to redirect LDNS requests to a name server in another network, or to a CDN provider in the cdn.inc file. Enclose the cname in quotation marks.
<code>zname</code>	Specifies the zone name (zname), or domain name, for the CDN pool. Use this attribute with the pool type CDN . Enclose the zname in quotation marks.
<code>dynamic_ratio</code>	Specifies whether the 3-DNS Controller treats QOS scores as ratios, and uses each server in proportion to the ratio determined by the QOS calculation. The default is no .
<code>rr_ldns</code>	Specifies whether the 3-DNS Controller returns a list of available virtual servers available for load balancing to a client and stores the list in the browser cache. The default is no , which specifies that the 3-DNS Controller returns only one A record per query.
<code>rr_ldns_limit</code>	The maximum number of A records to return when rr_ldns is set to yes . You can enter a value between 0 and 16 . The default is 0 , which specifies that the 3-DNS Controller returns the IP addresses of all (up to 16) available virtual servers.
<code>preferred</code>	Specifies the load balancing mode to use for the specified pool. Each acceptable value is described in the next table. The default is rr (Round Robin).
<code>alternate</code>	Specifies the load balancing mode to use for the specified pool if the preferred mode fails. The default is rr (Round Robin). Also see the description of default_alterate , a globals sub-statement, on page 13-9.
<code>fallback</code>	Specifies the load balancing mode to use for the specified pool if the alternate mode fails. If the fallback mode fails, the 3-DNS Controller returns the request to DNS. The default is return_to_dns . Also see the description of default_fallback , a globals sub-statement, on page 13-9.
<code>address</code>	As part of a pool definition, address specifies the IP address of each virtual server in the pool. You can use the same virtual server in multiple pools, but not within the same pool.
<code>port</code>	Specifies a specific port to use for the specified virtual server. This sub-statement is optional. A port specified here overrides the wide IP's port setting. If a port is not specified here, the wide IP's port value is assumed.
<code>ratio</code>	As part of a virtual server's address specification, ratio defines the default weighting to use with respect to all virtual servers in this pool when the Ratio load balancing mode is employed. The default is 1.

Table 13.29 *Pool sub-statements*

Load balancing modes

The load balancing sub-statements specify the load balancing modes to use for the wide IP in this order:

- The 3-DNS Controller attempts to load balance requests using the **preferred** mode.

- If the **preferred** mode fails, the 3-DNS Controller tries the **alternate** mode.
- If the **alternate** mode fails, the 3-DNS Controller tries the **fallback** mode.
- If the **fallback** mode fails, the request is returned to DNS. DNS attempts to resolve the request based on the contents of the zone files.

As noted in Table 13.30, not all modes are valid for the **alternate** sub-statement. Also note that the **alternate** and **fallback** sub-statements accept two additional values, **return_to_dns** and **null**.

If you do not specify a load balancing mode within a pool, the wide IP uses the default load balancing mode defined in the **globals** statement (see page 13-6).

Parameter	Description
completion_rate	Sends each new connection to the server that has the fewest number of dropped packets. Valid in a preferred or fallback sub-statement.
global_availability (ga)	Distributes connections to a list of servers, always sending a connection to the first available server in the list.
hops	Sends each new connection to the server that has the fewest number of network hops between the server and the client LDNS. Valid in a preferred or fallback sub-statement.
leastconn	Sends each new connection to the server that currently hosts the fewest current connections.
null	Bypasses the current load balancing method and forces the 3-DNS Controller to use the next load balancing method or, if it has cycled through all load balancing sub-statements for the pool, to the next pool. Valid in an alternate or fallback sub-statement.
packet_rate	Sends each new connection to the server that is managed by a BIG-IP Controller currently handling the least amount of network traffic (determined by the fewest number of packets currently processed by the controller).
qos	Takes these performance factors into account when determining how to distribute connections: hops, packet rate, completion rate, round trip time, kbps, virtual server capacity, and topology. You can configure how much emphasis to place on each performance factor, or you can configure the Quality of Service mode to treat all factors as being equally important. Valid in a preferred or fallback sub-statement.
random	Distributes each new connection to a server chosen at random from the wide IP set of virtual servers.
ratio	Distributes new connections across servers in proportion to a user-defined ratio.

Table 13.30 Load balancing mode sub-statements

Parameter	Description
return_to_dns	Returns the resolution request to DNS, preventing the 3-DNS Controller from using the next load balancing method or using the next available pool.
rr	Distributes connections evenly across all servers, passing each new connection to the next server in line.
rtt	Sends each new connection to the server that demonstrates the fastest round trip time between the server and the client LDNS. Valid in a preferred or fallback sub-statement.
topology	Distributes connections based on the proximity of an LDNS to a particular data center.
static_persist	Distributes connections to a virtual server based on IP address only. The 3-DNS Controller always returns the same virtual server, if the virtual server is available.
kpbs	Distributes connections to the virtual server with the lowest kilobytes per second throughput rate.

Table 13.30 Load balancing mode sub-statements

Use the following equation to configure the Quality of Service load balancing mode:

$$A (1/\text{packet rate}) + B (1/\text{rtt}) + C (\text{completion rate}) + D (\text{topology}) + E (1/\text{hops}) + F (1/\text{kpbs}) + G (\text{vs_capacity})$$

The topology statement

The **topology** statement implements a form of wide-area IP filtering, based on the geographic attributes of the DNS message. For example, you can specify that requesting LDNS clients in North America are allowed access to data centers in North America, but not allowed access to data centers in South America.

By including a **topology** statement in your **wideip.conf** file, you can use the topology load balancing mode, both on its own and as part of the Quality of Service mode.

For more information on using the Topology load balancing mode, see Chapter 3, *Configuring a Globally Distributed Network*, and Chapter 4, *Configuring a Content Delivery Network*, in the **3-DNS Administrator Guide**. For more information on topology in general, see Chapter 11, *Topology*.

Syntax for the topology statement

The **topology** statement uses the following syntax.

```

topology {
    longest_match <yes | no>

    // server          ldns          score
    "pool.origin"     cont."North America"    100
    "pool.cache_farm" !cont."North America" 100
}

```

Figure 13.20 Syntax for the **topology** statement

Definition of topology sub-statements

The topology sub-statements define the topology records that the 3-DNS Controller uses for Topology load balancing.

Parameter	Description
longest_match	In cases where there are topology records that match a particular IP address, longest_match specifies whether the 3-DNS Controller selects the record that is most specific, and thus has the longest match. When longest_match is set to yes , the topology records are sorted according to the longest match criteria any time the wideip.conf file is dumped.
server	Specifies the location of the virtual servers.
ldns	Specifies the location of the LDNS making the name resolution request.
pool	Specifies a wide-IP pool for load balancing. Note that pool names can be duplicated across wide IPs. Use this for server in a topology record.
datacenter	Specifies a data center for load balancing. Use this for server in a topology record.
continent	Specifies one of these continents for load balancing: " North America ", " South America ", " Europe ", " Asia ", " Australia ", " Africa ", or " Antarctica ". Use this for ldns in a topology record.
country	Specifies a country for load balancing using one of the two-letter country codes found in the file /var/3dns/include/net.ccdb . Use this for ldns in a topology record.
isp.AOL	For local DNS servers only, specifies the Internet service provider, America Online (AOL).
!	The not (!) operator negates the meaning of an element in a topology record.
score	Specifies the relative weight, or score, for the topology record, which allows the 3-DNS Controller to evaluate the best resolution option for a DNS request.

Table 13.31 Topology sub-statements

Access control lists

You can now create access control lists (ACLs) that contain a group of LDNS IP addresses whose paths the 3-DNS Controller will not probe. The three different types of ACLs are:

- Prober
- Hops
- Discovery

Syntax for the access control lists

The access control lists use the following syntax.

```
actions {
  NO_RELAY
  delete rdb ACL region "probe_acl"
  delete rdb ACL region "hops_acl"
  delete rdb ACL region "discovery_acl"
}
region_db ACL {
  region {
    name "probe_acl"
    <ldns cidr>
    <ldns cidr>
  }
  region {
    name "hops_acl"
    region "probe_acl"
    <ldns cidr>
    <ldns cidr>
  }
  region {
    name "discovery_acl"
    <ldns cidr>
    <ldns cidr>
    <ldns cidr>
  }
}
```

Figure 13.21 Syntax for the access control lists

Definition of the access control list sub-statements

The access control list sub-statements define local DNS servers that should not be probed.

Parameter	Description
actions	Include, but do not modify this sub-statement.
region_db ACL	Specifies that ACLs are being created.
region	Specifies groups of CIDRs by probe type.
name	Specifies the name of the ACL.
probe_acl	The 3-DNS Controller restricts any big3d agent from probing the defined group of LDNS servers.
hops_acl	The 3-DNS Controller restricts any big3d agent from tracerouting the defined group of LDNS servers
discovery_acl	The 3-DNS Controller restricts any big3d agents from performing port discovery on the defined group of LDNS servers

Table 13.32 Access control list sub-statements

◆ Note

For more information on ACLs, refer to Chapter 2, Access Control Lists.

Working with comments

You can insert comments anywhere you would otherwise see white space in the 3-DNS Controller configuration file.

Syntax

Note that the comment syntax depends on the environment in which you use the configuration file.

```
/* This is a 3-DNS comment as in C */  
// This is a 3-DNS comment as in C++  
# This is a 3-DNS comment as in common UNIX shells and Perl
```

Figure 13.22 Syntax for comments

Definition and usage

The format for comments varies by programming language; each format is described below. To avoid comment nesting problems, we recommend that you use only one comment style in your **wideip.conf** file. However, all styles may be used in a single **wideip.conf** file.

C style comments

C style comments start with the slash character, followed by the asterisk character (`/*`), and end with the asterisk character, followed with the slash character (`*/`). Because the comment is completely delimited with these characters, a comment can span multiple lines.

Note that C style comments cannot be nested. For example, the following syntax is not valid because the entire comment ends with the first `*/`.

```
/* This is the start of a comment.  
   This is still part of the comment.  
/* This is an incorrect attempt to nest a comment. */  
   This is no longer in any comment. */
```

Figure 13.23 Syntax for C style comments

C++ style comments

C++ style comments start with two slash characters (`//`) and are no longer than one line in length. To have one logical comment span multiple lines, each line must start with the `//` pair.

```
// This is the start of a comment. The next line  
// is a new comment line, even though it is  
// logically part of the previous comment.
```

Figure 13.24 Syntax for C++ style comments

Shell style comments

Shell style (also known as Perl style) comments start with the number character (#) and are no longer than one line in length.

```
# This is the start of a comment. The next line
# is a new comment line, even though it is logically
# part of the previous comment.
```

Figure 13.25 Syntax for shell style comments

Understanding current values

You may notice several current values in the **wideip.conf** file. Current values are preceded by the **cur_** prefix in the **wideip.conf** file. The purpose of current values is to pre-load the database with previously collected statistics and metrics. The collected statistics and metrics are useful if you want to quickly restart a 3-DNS Controller without a temporary loss of intelligence.

You may notice current values associated with **server**, **vs**, **path**, or **wideip** definitions. (You can also view current values by clicking the Conf button



in the Configuration utility.) The current values parameters show the real-time status of the servers, virtual servers, local DNS server paths, and wide IPs that make up your configuration. Examples of current values for each type of definition follow.

◆ WARNING

Do not edit the current values statements unless you are a very experienced 3-DNS Controller user, or you are instructed to do so by your vendor.

Server definition current values

Server definitions may contain several current values, as shown in Figure 13.26.

```
// New York BIG-IP Controller
server {
  type bigip
  address 192.168.101.40
  cur_ok 1 //Up
  cur_packet_rate 6
  cur_packet_in 1872
  cur_packet_out 1812
  cur_uptime 3615 //60 mins 15 secs
  [virtual server definitions]
}
```

Figure 13.26 Example of current values in a server definition

The current values parameters that are shown in Figure 13.26 are defined in Table 13.33. Note that you may see more current values than those listed here.

Parameter	Description
cur_ok	Indicates the state of the specified server. The options are: 1 (up), 2 (down), 3 (waiting), 4 (alert), and 5 (panic).
cur_packet_rate	Indicates the number of packets per second sent during the last sample period.
cur_packet_in	Indicates the number of packets that the server has received.
cur_packet_out	Indicates the number of packets the server has sent.
cur_uptime	Indicates the length of time that the server has been running since the last reboot.

Table 13.33 Description of current values in a server definition

Virtual server definition current values

Virtual server definitions may contain several current values, as shown in Figure 13.27.

```
vs {
  address 192.168.102.50:80 //http
  [ depends_on {
    address 109.168.102.50:20 //ftp-data
    address 192.168.102.50:443 //https
  } ]
  limit { /* none */ }
  probe_protocol tcp
  cur_state 1 // green
  cur_nodes_up 3
  cur_connections 0
  ...
  cur_picks 0
  cur_refreshes 41
}
```

Figure 13.27 Example of current values in a virtual server definition

The current values parameters that are shown in Figure 13.27 are defined in Table 13.34. Note that you may see more current values than those listed here.

Parameter	Description
cur_state	Indicates the availability of the virtual server to receive connection requests. The options are: 1 (green - available), 2 (red - down), 3 (blue - unknown), 4 (yellow - unavailable)
cur_nodes_up	Indicates the number of active servers serving the specified virtual server.
cur_connections	Indicates the number of connections to the specified virtual server.
cur_picks	Indicates the number of times the specified virtual server was returned by the 3-DNS Controller.
cur_refreshes	Indicates the number of times the server and connection counts were refreshed with new data.

Table 13.34 Description of current values in a virtual server definition

Local DNS server paths current values

Path definitions for local DNS servers may contain several current values, as shown in Figure 13.28.

```
path {
  address 10.25.50.100 // LDNS
  cur_rtt 102382
  cur_completion_rate 10000
  cur_picks 239
  cur_accesses 302
}
```

Figure 13.28 Example of current values in a path definition

The current values parameters that are shown in Figure 13.28 are defined in Table 13.35. Note that you may see more current values than those listed here.

Parameter	Description
cur_rtt	Indicates the round trip time (RTT), which is a calculation of the time (in microseconds) that the specified machine takes to respond to a probe issued by the 3-DNS Controller.
cur_completion_rate	Indicates the percentage of completed packets versus lost packets, using this equation: [1 - (packets received / sent)] X 10000.
cur_picks	Indicates the number of times this path's data resulted in the virtual server being chosen for a connection. This only applies if a wide IP is doing dynamic load balancing (using path data).
cur_accesses	Indicates the number of times this path was considered when performing dynamic load balancing.

Table 13.35 Description of current values in a path definition

Wide IP definition current values

Wide IP definitions may contain several current values, as shown in Figure 13.29.

```
wideip {
  address 192.168.102.70
  name "www.domain.com"
  port 80
  cur_preferred 143982
  cur_alternate 108090
  cur_fallback 130094
  cur_returned_to_dns 23872
  [pool definitions]
}
```

Figure 13.29 Example of current values in a wide IP definition

The current values parameters that are shown in Figure 13.29 are defined in Table 13.36. Note that you may see more current values than those listed here.

Parameter	Description
cur_preferred	Indicates the number of times the specified wide IP was resolved by the preferred load balancing mode.
cur_alternate	Indicates the number of times the specified wide IP was resolved by the alternate load balancing mode.
cur_fallback	Indicates the number of times the specified wide IP was resolved by the fallback load balancing mode.
cur_returned_to_dns	Indicates the number of times the specified wide IP did not find a suitable virtual server to return using the preferred , alternate , or fallback load balancing modes. In this situation, the 3-DNS Controller returns the wide IP key (fallback address) as specified in the zone file.

Table 13.36 Description of current values in a wide IP definition

◆ Tip

To find out how many times the 3-DNS Controller has received resolution requests for a wide IP, add the values for **cur_preferred**, **cur_alternate**, and **cur_fallback**.



Glossary

3-DNS Distributed Traffic Controller

The 3-DNS Distributed Traffic Controller is a wide area load distribution solution that intelligently allocates Internet and intranet service requests across geographically distributed network servers. The 3-DNS Distributed Traffic Controller is also called the 3-DNS Controller.

3-DNS Maintenance menu

The 3-DNS Maintenance menu is a command line utility that you use to configure the 3-DNS Controller.

3-DNS web server

The 3-DNS web server is a standard web server that hosts the Configuration utility on the 3-DNS Controller.

A record

The **A** record is the ADDRESS resource record that a 3-DNS Controller returns to a local DNS server in response to a name resolution request. The **A** record contains a variety of information, including one or more IP addresses that resolve to the requested domain name.

access control list (ACL)

An access control list is a list of local DNS server IP addresses that are excluded from path probing, hops, or port discovery queries.

active unit

In a redundant system, an active unit is a controller that currently load balances name resolution requests. If the active unit in the redundant system fails, the standby unit assumes control and begins to load balance requests.

alternate method

The alternate method specifies the load balancing mode that the 3-DNS Controller uses to pick a virtual server if the preferred method fails. See also *fallback method*, *preferred method*.

big3d agent

The **big3d** agent is a monitoring agent that collects metrics information about server performance and network paths between a data center and a specific local DNS server. The 3-DNS Controller uses the information collected by the **big3d** agent for dynamic load balancing.

BIND (Berkeley Internet Name Domain)

BIND is the most common implementation of the Domain Name System (DNS). BIND provides a system for matching domain names to IP addresses. For more information, refer to <http://www.isc.org/products/BIND>.

CNAME record

A canonical name (CNAME) record acts as an alias to another domain name. A canonical name and its alias can belong to different zones so the **CNAME** record must always be entered as a fully qualified domain name. **CNAME** records are useful for setting up logical names for network services so that they can be easily relocated to different physical hosts.

completion rate

The completion rate is the percentage of packets that a server successfully returns during a given session.

Completion Rate mode

The Completion Rate mode is a dynamic load balancing mode that distributes connections based on which network path drops the fewest packets, or allows the fewest number of packets to time out.

Configuration utility

The Configuration utility is the browser-based application that you use to configure the 3-DNS Controller.

content delivery network (CDN)

A content delivery network (CDN) is an architecture of Web-based network components that helps dramatically reduce the wide-area network latency between a client and the content they wish to access. A CDN includes some or all of the following network components: wide-area traffic managers, Internet service providers, content server clusters, caches, and origin content providers.

CDN switching

CDN switching is the functionality of the 3-DNS Controller that allows a user to redirect traffic to a third-party network, or transparently switch traffic to a CDN. The two features of the 3-DNS Controller that make CDN switching possible are geographic redirection and the pool type CDN.

data center

A data center is a physical location that houses one or more 3-DNS Controllers, BIG-IP Controllers, EDGE-FX Caches, GLOBAL-SITE Controllers, or host machines.

data center server

A data center server is any server recognized in the 3-DNS Controller configuration. A data center server can be any of the following: a 3-DNS Controller, a BIG-IP Controller, an EDGE-FX Cache, a GLOBAL-SITE Controller, or a host.

discovery factory

A discovery factory is a tool managed by the **big3d** agent that checks for alternate ports to ping when trying to collect path data for a local DNS server.

domain name

A domain name is a unique name that is associated with one or more IP addresses. Domain names are used in URLs to identify particular Web pages. For example, in the URL **http://www.f5.com/index.html**, the domain name is **f5.com**.

dynamic load balancing modes

Dynamic load balancing modes base the distribution of name resolution requests to virtual servers on live data, such as current server performance and current connection load.

dynamic site content

Dynamic site content is a type of site content that is automatically generated each time a user accesses the site. Examples are current stock quotes or weather satellite images.

Extended Content Verification (ECV)

On the 3-DNS Controller, ECV is a service monitor that checks the availability of actual content, (such as a file or an image) on a server, rather than just checking the availability of a port or service, such as HTTP on port 80.

external interface

An external interface is the network interface that can be accessed across a wide-area network (WAN). See also *internal interface*.

fail-over

Fail-over is the process whereby a standby unit in a redundant system takes over when a software failure or hardware failure is detected on the active unit.

fail-over cable

The fail-over cable is the cable that directly connects the two controller units in a hardware-based redundant system.

fallback method

The fallback method is the third method in a load balancing hierarchy that the 3-DNS Controller uses to load balance a resolution request. The 3-DNS Controller uses the fallback method only when the load balancing modes specified for the preferred and alternate methods fail. Unlike the preferred method and the alternate method, the fallback method uses neither server nor virtual server availability for load balancing calculations. See also *preferred method, alternate method*.

FDDI (Fiber Distributed Data Interface)

FDDI is a multi-mode protocol for transmitting data on optical-fiber cables at speeds up to 100 Mbps.

First-Time Boot utility

The First-Time Boot utility is a utility that takes you through the initial system configuration process. The First-Time Boot utility runs automatically when you turn on a controller for the first time.

Global Availability mode

Global Availability is a static load balancing mode that bases connection distribution on a particular server order, always sending a connection to the first available server in the list. This mode differs from Round Robin mode in that it searches for an available server always starting with the first server in the list, while Round Robin mode searches for an available server starting with the next server in the list (with respect to the server selected for the previous connection request).

hops factory

A hops factory is a type of factory run by the **big3d** agent that collects hops data about network paths.

host

A host is a network server that manages one or more virtual servers that the 3-DNS Controller uses for load balancing.

ICMP (Internet Control Message Protocol)

ICMP is an Internet communications protocol used to determine information about routes to destination addresses, such as virtual servers managed by BIG-IP Controllers and 3-DNS Controllers.

internal interface

An internal interface is a network interface that can be accessed from a local-area network (LAN). See also *external interface*.

iQuery

The iQuery protocol is used to exchange information between 3-DNS Controllers, BIG-IP Controllers, EDGE-FX Caches, and GLOBAL-SITE Controllers. The iQuery protocol is officially registered with IANA for port 4353, and works on UDP and TCP connections.

Kilobytes/Second mode

The Kilobytes/Second mode is a dynamic load balancing mode that distributes connections based on which available server currently processes the fewest kilobytes per second.

Least Connections mode

The Least Connections mode is a dynamic load balancing mode that bases connection distribution on which server currently manages the fewest open connections.

load balancing methods

Load balancing methods are the settings that specify the hierarchical order in which the 3-DNS Controller uses three load balancing modes. The preferred method specifies the first load balancing mode that the 3-DNS Controller tries, the alternate method specifies the next load balancing mode to try if the preferred method fails, and the fallback method specifies the last load balancing mode to use if both the preferred and the alternate methods fail.

load balancing mode

A load balancing mode is the way in which the 3-DNS Controller determines how to distribute connections across an array.

local DNS

A local DNS is a server that makes name resolution requests on behalf of a client. With respect to the 3-DNS Controller, local DNS servers are the source of name resolution requests. Also referred to as LDNS.

metrics information

Metrics information is the data that is typically collected about the paths between BIG-IP Controllers, EDGE-FX Caches or GLOBAL-SITE Controllers, and local DNS servers. Metrics information is also collected about the performance and availability of virtual servers. Metrics information is used for load balancing, and it can include statistics such as round trip time, packet rate, and packet loss.

MindTerm SSH

MindTerm SSH is the third-party application on 3-DNS Controllers that uses SSH for secure remote communications. SSH encrypts all network traffic (including passwords) to effectively eliminate eavesdropping, connection hijacking, and other network-level attacks. SSH also provides secure tunneling capabilities and a variety of authentication methods.

name resolution

Name resolution is the process by which a name server matches a domain name request to an IP address, and sends the information to the client requesting the resolution.

name server

A name server is a server that maintains a DNS database, and resolves domain name requests to IP addresses using that database.

named

The **named** daemon manages domain name server software.

NameSurfer

NameSurfer is the third-party application on 3-DNS Controllers that automatically manages DNS zone files, synchronizing them with the configuration on the controller. NameSurfer automatically updates any configuration changes that you make using the Configuration utility. NameSurfer also provides a graphical user interface for DNS zone file management.

Network Time Protocol (NTP)

Network Time Protocol functions over the Internet to synchronize system clocks to Universal Coordinated Time. NTP provides a mechanism to set and maintain clock synchronization within milliseconds.

NS record

A name server (NS) record is used to define a set of authoritative name servers for a DNS zone. A name server is considered authoritative for some given zone when it has a complete set of data for the zone, allowing it to answer queries about the zone on its own, without needing to consult another name server.

packet rate

The packet rate is the number of data packets per second processed by a server.

Packet Rate mode

The Packet Rate mode is a dynamic load balancing mode that distributes connections based on which available server currently processes the fewest packets per second.

path

A path is a logical network route between a data center server and a local DNS server.

path probing

Path probing is the collection of metrics data, such as round trip time and packet rate, for a given path between a requesting LDNS server and a data center server.

persistence

On a 3-DNS Controller, persistence is a series of related requests received from the same local DNS server for the same wide IP name. When persistence is turned on, a 3-DNS Controller sends all requests from a particular local DNS server for a specific wide IP to the same virtual server, instead of load balancing the requests.

picks

Picks represent the number of times a particular virtual server is selected to receive a load balanced connection.

pool

A pool is a group of virtual servers managed by a BIG-IP Controller, an EDGE-FX Cache, or a host. The 3-DNS Controller load balances among pools (using the Pool LB Mode), as well as among individual virtual servers.

pool ratio

A pool ratio is a ratio weight applied to pools in a wide IP. If the Pool LB mode is set to Ratio, the 3-DNS Controller uses each pool for load balancing in proportion to the weight defined for the pool.

preferred method

The preferred method specifies the first load balancing mode that the 3-DNS Controller uses to load balance a resolution request. See also *alternate method*, *fallback method*.

principal 3-DNS Controller

A 3-DNS Controller that initiates metrics collection by the **big3d** agents and distributes the metrics to other members of a sync group. See also *receiver 3-DNS Controller*.

production rule

A production rule, on the 3-DNS Controller, can change system behavior under specific operating conditions. For example, a production rule can switch load balancing modes or can reroute network traffic to a specific set of servers. Production rules are based on triggers such as time of day or current network traffic load.

probe protocol

The probe protocol is the specific protocol used to probe a given path and collect metrics information for the path. The probe protocols available on the 3-DNS Controller are: ICMP, DNS_REV, DNS_DOT, UDP, and TCP. The probe protocols that are available change based on the data center server type.

prober

A prober is a specific thread of the **big3d** agent that is used for path probing of a given set of paths.

prober factory

A prober factory is a utility that collects metrics data, such as round trip time and packet rate, for a given path between a requesting LDNS and a data center server. Prober factories are managed by the **big3d** agent, which reports the path probing metrics to the 3-DNS Controller. Prober factories can run on BIG-IP Controllers, EDGE-FX Caches, and GLOBAL-SITE Controllers.

Quality of Service load balancing mode

The Quality of Service load balancing mode is a dynamic load balancing mode that bases connection distribution on a configurable combination of the packet rate, completion rate, round trip time, hops, virtual server capacity, kilobytes per second, and topology information.

QOS equation

The QOS equation is the equation on which the Quality of Service load balancing mode is based. The equation calculates a score for a given path between a data center server and a local DNS server. The Quality of Service mode distributes connections based on the best path score for an available data center server. You can apply weights to the factors in the equation, such as round trip time and completion rate.

ratio

A ratio is the parameter in a virtual server statement that assigns a weight to the virtual server for load balancing purposes.

Ratio mode

The Ratio load balancing mode is a static load balancing mode that distributes connections across an pool of virtual servers in proportion to the ratio weight assigned to each individual virtual server.

receiver 3-DNS Controller

A receiver 3-DNS Controller is a controller, in a sync group, that receives metrics data that are broadcast from **big3d** agents, but does not initiate metrics collection. See also *principal 3-DNS Controller*.

redundant system

A redundant system is a pair of controllers that are configured for fail-over. In a redundant system, one controller runs as the active unit and the other controller runs as the standby unit. If the active unit fails, the standby unit takes over and manages resolution requests.

remote administrative IP address

A remote administrative IP address is an IP address from which a controller allows shell connections, such as SSH, RSH, or Telnet.

resolver

The resolver is the client part of the Domain Name System. The resolver translates a program's request for host name information into a query to a name server, and translates the response into an answer to the program's request. See also *name server*.

resource record

resource record is a record in a DNS database that stores data associated with domain names. A resource record typically includes a domain name, a TTL, a record type, and data specific to that record type. See also *A record*, *CNAME record*, *NS record*.

reverse domains

A type of DNS resolution request that matches a given IP address to a domain name. The more common type of DNS resolution request starts with a given domain name and matches that to an IP address.

root name server

A root name server is a master DNS server that maintains a complete DNS database. There are approximately 13 root name servers in the world that manage the DNS database for the World Wide Web.

Round Robin mode

Round Robin mode is a static load balancing mode that bases connection distribution on a set server order. Round Robin mode sends a connection request to the next available server in the order.

round trip time (RTT)

Round trip time is the calculation of the time (in microseconds) that a local DNS server takes to respond to a ping issued by the **big3d** agent running on a data center server. The 3-DNS Controller takes RTT values into account when it uses dynamic load balancing modes.

Round Trip Time mode

Round Trip Time is a dynamic load balancing mode that bases connection distribution on which virtual server has the fastest measured round trip time between the data center server and the local DNS server.

secondary DNS

The secondary DNS is a name server that retrieves DNS data from the name server that is authoritative for the DNS zone.

site content

Site content is data (including text, images, audio, and video feeds) that is accessible to clients who connect to a given site. See also *dynamic site content*, *static site content*.

SNMP (Simple Network Management Protocol)

SNMP is the Internet standard protocol, defined in STD 15, RFC 1157, that was developed to manage nodes on an IP network.

sod (switch over daemon)

The **sod** daemon controls the fail-over process in a redundant system.

SSH

SSH is a protocol for secure remote login and other secure network services over a non-secure network.

standby unit

A standby unit is a controller in a redundant system that is always prepared to become the active unit if the active unit fails.

static load balancing modes

Static load balancing modes base the distribution of name resolution requests to virtual servers on a pre-defined list of criteria and server and virtual server availability; they do not take current server performance or current connection load into account.

static site content

Static site content is a type of site content that is stored in HTML pages, and changes only when an administrator edits the HTML document itself.

subdomain

A subdomain is a sub-section of a higher level domain. For example, **.com** is a high level domain, and **F5.com** is a subdomain within the **.com** domain.

sub-statement

A sub-statement is a logical section within a statement that defines a particular element in the statement. A sub-statement begins with the sub-statement name followed by an open brace ({) and ends with a closed brace (}). Everything between those braces is part of the sub-statement. Sub-statements typically define a group of related variables, such as the calculation coefficients used in Quality of Service load balancing.

sync group

A sync group is a group of 3-DNS Controllers that share system configurations and path metrics for data center servers and virtual servers. Sync groups have one principal 3-DNS Controller, and may contain one or

more receiver controllers. The receiver controllers obtain their configuration information from the principal controller. See also *principal 3-DNS Controller*, *receiver 3-DNS Controller*.

time tolerance value

The time tolerance value is the number of seconds that one 3-DNS Controller's clock is allowed to differ in comparison to another 3-DNS Controller's clock, without the two clocks being considered out of sync.

Topology mode

The Topology mode is a static load balancing mode that bases the distribution of name resolution requests on the weighted scores for topology records. Topology records are used by the Topology load balancing mode to redirect DNS queries to the closest virtual server, geographically, based on location information derived from the DNS query message.

topology record

A topology record specifies a score for a local DNS server location endpoint and a virtual server location endpoint.

topology score

The topology score is the weight assigned to a topology record when the 3-DNS Controller is filtering the topology records to find the best virtual server match for a DNS query.

topology statement

A topology statement is a collection of topology records.

traceroute

Traceroute is the utility that the hops factory uses to calculate the total number of network hops between a local DNS server and a specific data center.

TTL (Time to Live)

The TTL is the number of seconds for which a specific DNS record or metric is considered to be valid. When a TTL expires, the server usually must refresh the information before using it again.

unavailable

The **unavailable** is a status used for data center servers and virtual servers. When a data center server or virtual server is **unavailable**, the 3-DNS Controller does not use it for load balancing.

unknown

The **unknown** status is used for data center servers and virtual servers. When a data center server or virtual server is new to the 3-DNS Controller and does not yet have metrics information, the 3-DNS Controller marks its status as **unknown**. The 3-DNS Controller can use unknown servers for load balancing, but if the load balancing mode is dynamic, the 3-DNS Controller uses default metrics information for the unknown server until it receives live metrics data.

up

The **up** status is used for data center servers and virtual servers. When a data center server or virtual server is **up**, the data center server or virtual server is available to respond to name resolution requests.

virtual server

A virtual server is a specific combination of a virtual IP address and virtual port, and is associated with a content site that is managed by a BIG-IP Controller, EDGE-FX Cache, or host server.

watchdog timer card

The watchdog timer card is a hardware device that monitors the 3-DNS Controller for hardware failure.

wide IP

A wide IP is a collection of one or more domain names that maps to one or more groups of virtual servers managed either by BIG-IP Controllers, EDGE-FX Caches, or by host servers. The 3-DNS Controller load balances name resolution requests across the virtual servers that are defined in the wide IP that is associated with the requested domain name.

WKS (well-known services)

Well-known services are protocols on ports 0 through 1023 that are widely used for certain types of data. Some examples of some well-known services (and their corresponding ports) are: HTTP (port 80), HTTPS (port 443), and FTP (port 20).

WKS record

A WKS record is a DNS resource record that describes the services usually provided by a particular protocol on a specific port.

zone

In DNS terms, a zone is a subset of DNS records for one or more domains.

zone file

In DNS terms, a zone file is a database set of domains with one or many domain names, designated mail servers, a list of other name servers that can answer resolution requests, and a set of zone attributes, which are contained in an SOA record.



Index

/etc/hosts.allow file 10-2
 /etc/snmptrap.conf file 10-5
 /etc/syslog.conf file 10-5

3-DNS Controller scripts 9-1
 3-DNS guides, types of 1-1
 3-DNS Maintenance menu
 and scripts 9-1
 3-DNS web server
 configuring 9-2
 configuring passwords 9-2
 configuring users 9-2
 3dns_add script 9-1
 3dns_admin_start 9-1
 3dns_dump script 9-1
 3dns_sync_metrics script 9-1
 3dns_web_passwd script 9-2
 3dnsmaint script 9-2
 3dprint script 9-2
 3dscrip, managing production rules 7-6
 3ndc script 9-4

A

A record, about 8-2
 access control lists
 about 2-1
 defining 2-1
 examples 2-2
 syntax 13-44
 ACL
 see access control list 2-1
 actions supported by production rules 7-10
 additional 3-DNS Controllers, adding 9-1
 Ask F5 knowledge base 1-4

B

big3d agent
 about 3-1
 and dynamic load balancing 5-6
 and factory types 3-3
 configuring 3-1
 default settings 3-1
 installing on BIG-IP Controllers 3-1, 3-2
 installing on EDGE-FX Caches 3-2
 installing on GLOBAL-SITE Controller 3-2
 restarting 9-4
 stopping 9-4
 updating BIG-IP Controllers 9-4

 updating EDGE-FX Caches 9-4
 updating GLOBAL-SITE Controllers 9-4
 viewing version numbers 9-5
 big3d_install script 9-4
 big3d_restart script 9-4
 big3d_version script 9-5
 BIG-IP Controller
 collecting metrics 10-8
 installing big3d agent 3-1, 3-2
 updating big3d agent 9-4
 viewing big3d version 9-5
 BIND resources 5-21
 buffer size 13-17

C

Change/Add Users for 3-DNS Configuration Utility
 command 9-2
 Check remote versions of big3d command 9-5
 checkpoint file 9-6
 checktrap.pl script 10-6
 configuring 10-6
 CNAME record, about 8-3
 command line
 conventions 1-3
 viewing statistics 9-2
 command line utilities 12-1
 comments
 3-DNS Controller 13-45
 3-DNS Controller syntax 13-46
 syntax in wideip.conf 13-46
 config_httpd script 9-2
 config_ssh script 9-5
 configuration file syntax
 for 3-DNS Controller 13-20
 for BIG-IP Controller 13-21
 for comments 13-46
 for datacenter statement 13-32
 for EDGE-FX Cache 13-24
 for GLOBAL-SITE Controller 13-25
 for hosts 13-25
 for topology statement 13-43
 for wideip statement 13-35
 configuration files
 on the 3-DNS Controller 13-1
 requirements overview 13-1
 working with statements 13-4
 Configuration utility
 and custom production rules 7-6
 setting up production rules 7-1
 Configure SSH communication with remote devices
 command 9-5

- custom production rules
 - and local DNS servers 7-12
 - and the Configuration utility 7-6
 - examples 7-11

D

- data center servers
 - collecting metrics 10-8
- Dump 3dnscd Statistics command 9-2
- dynamic load balancing
 - about 5-6
 - and big3d agents 5-6
- Dynamic load balancing modes
 - and path information 13-10
- dynamic load balancing modes
 - types 5-6
- dynamic ratio
 - and Quality of Service mode 5-10

E

- ECV 4-1
- ECV service monitors 4-1
- EDGE-FX Cache
 - collecting metrics 10-8
 - installing big3d agent 3-2
 - updating big3d agent 9-4
 - viewing big3d version 9-5
- Edit 3-DNS Configuration command 9-5
- edit_lock script 9-5
- edit_wideip script 9-5
- encryption 9-6
- event-based triggers
 - defining 7-4
- every statement
 - guidelines 7-9
 - in production rules 7-9
- examples of syntax
 - for datacenter statement 13-32
 - for globals statement 13-9
 - for server statement (3-DNS) 13-21
 - for server statement (BIG-IP) 13-23
 - for server statement (EDGE-FX) 13-24
 - for server statement (GLOBAL-SITE) 13-25
 - for server statement (host type) 13-27
 - for sync_group statement 13-33
 - for wideip statement 13-36
- Extended Content Verification. See ECV

F

- F5makekey script 9-6
- factories
 - default settings 3-3
- factories, types of
 - discovery 3-3
 - hops 3-3
 - permanent 3-3
 - probing 3-3
 - SNMP 3-3
- file monitors 4-1
- firewalls
 - configuring for 3-7

G

- Generate and Copy iQuery Encryption Key command 9-6
- global production rules 7-2
- global timers
 - configuring 5-19
- global variables
 - configuring load balancing 5-16
 - configuring timers 5-20
 - configuring TTL 5-20
- globals statement
 - about 13-6
 - load balancing variables 5-17
- GLOBAL-SITE Controller
 - installing big3d agent 3-2
 - updating big3d agent 9-4
 - viewing big3d version 9-5

H

- hacker detection 7-13
- help, finding 1-3
- host servers
 - collecting metrics 10-8
 - configuring SNMP 10-7, 10-9

I

- if statement
 - guidelines 7-7
 - in production rules 7-7
- include files 13-1
- Install and Start big3d command 3-2
- install_key script 9-6
- iQuery
 - configuring for firewalls 3-7
 - encrypting 9-6

iQuery key
 distributing to other F5 servers 9-6
 iQuery port 13-18

K

knowledge base, Ask F5 1-4

L

last resort pool
 about 5-15
 configuring 5-15
 configuring an overflow network 5-15

LDNS

 load balancing 7-12

LDNS round robin

 about 5-14

load balancing

 and persistence 13-10
 configuring 5-10
 configuring at the global level 5-10
 configuring at the wide IP level 5-10
 configuring global variables 5-16
 using production rules 7-11, 7-12

load balancing according to LDNS 7-12

load balancing modes

 completion rate 5-6
 Hops 5-9
 Kilobytes/Second 5-9
 least connections 5-7
 None 5-5
 packet rate 5-7
 Quality of Service 5-9
 random 5-4
 ratio 5-4
 return to DNS 5-5
 round robin 5-3
 round trip times 5-8
 static persist 5-3
 VS Capacity 5-10

load balancing sub-statement 13-16

M

man pages 12-1

management tool

 production rules 7-1

manual configurations

 troubleshooting 5-20
 troubleshooting syntax errors 5-21
 understanding error messages 5-21
 verifying wideip.conf syntax 5-21, 13-1

Map, Network 6-1

memory allocation 13-14, 13-17

metrics

 and probing 13-18
 collecting path information 13-13
 setting durations 13-12
 synchronizing 9-1
 types of 10-8
 updating 13-11

metrics collection 10-8

 about TTL and timers 5-17

 setting TTL and timer values 5-17

monitors, file 4-1

MX record 8-3

N

Network Map 6-1

 and objects 6-2

 configuring the network 6-2

 viewing 6-2

network traffic

 controlling 7-1

network, viewing layout 6-1

NS record 8-3

O

overflow network

 and last resort pool 5-15

P

path information 13-10

 using probing 13-18

periodic task intervals 13-11

persistence 13-10

pools 5-11, 13-39

prober sub-statement 13-16

probing 13-18

 and SNMP 10-7

 server types 10-8

 types of metrics 10-8

probing exclusion lists

 see access control lists 13-44

production rules 7-1

 according to LDNS 7-12

 according to time of day 7-11

 adding 7-2

 applying a combined date and time variable 7-4

 applying a date variable 7-3

 applying day of the week variable 7-3

 applying time of day variable 7-3

- choosing rule types 7-2
- configuring custom 7-5
- configuring in wideip.conf file 7-5
- defining custom 7-5
- defining global 7-2
- defining triggers 7-2, 7-4
- defining wide IP 7-2
- deleting 7-2
- detecting hackers 7-13
- examples of custom 7-11
- executing 7-6
- getting help 7-5
- inserting in wideip.conf file 7-5
- managing with 3dscrip utility 7-6
- managing with Configuration utility 7-2
- types of actions 7-10
- understanding 3dscrip guidelines 7-6
- using Configuration utility 7-1
- using every statement 7-9
- using if statement 7-7
- using scripting language 7-5
- using when statement 7-8
- viewing in Configuration utility 7-2

protection from hackers

- using production rules 7-13

PTR record 8-4

Q

QOS coefficients 13-14

QOS equation 13-14

Quality of Service mode 13-14

- customizing 5-10
- using dynamic ratio 5-10

R

Random load balancing mode 5-4

Ratio load balancing mode 5-4

reaping 13-17

Reconfigure 3-DNS Configuration Utility command 9-2

release notes 1-3

requirements

- configuration files 13-1

resource records

- A 8-2
- CNAME 8-3
- less common types 8-5
- MX 8-3
- NS 8-3
- PTR 8-4
- SOA 8-4

Restart 3-DNS Configuration Utility command 9-1

Restart big3d command 9-4

Round Robin load balancing mode 5-3

S

scripting language

- setting up production rules 7-5

scripts 9-1

- 3dns_admin_start 9-1
- 3dns_dump 9-1
- 3dns_metrics_sync 9-1
- 3dns_web_passwd 9-2
- 3dnsmaint 9-2
- 3dprint 9-2
- 3ndc 9-4
- big3d_install 9-4
- big3d_restart 9-4
- big3d_version 9-5
- checktrap.pl 10-6
- config_httpd 9-2
- edit_lock 9-5
- edit_wideip 9-5
- F5makekey 9-6
- install_key 9-6
- syncd_checkpoint 9-6
- syncd_rollback 9-7
- syncd_start 9-8
- syncd_stop 9-8

server statement 13-19

- 3-DNS Controller 13-20
- BIG-IP Controller 13-21
- EDGE-FX Cache 13-24
- GLOBAL-SITE Controller 13-25
- hosts 13-25

service monitors, ECV 4-1

SNMP

3-DNS Controller OIDs 10-5

and probing 10-7

client access 10-3

generating traps 10-5

in the Configuration utility 10-7

MIB 10-2

trap configuration 10-4

SNMP agent

allowing host access 10-2

configuration file requirements 10-2

configuring 10-2, 10-3

configuring hosts 10-10

denying UPD connections 10-2

generating traps 10-5

in the Configuration utility 10-7

SNMP prober factory 10-7

-
- SNMP trap logs 10-5
 - SOA record 8-4
 - ssh key
 - generating 9-5
 - statements
 - 3-DNS Controller 13-4
 - globals 13-6
 - server 13-19
 - sync_group 13-33
 - topology 13-42
 - wideip 13-34
 - statistics, viewing 9-2
 - sync groups
 - archiving synchronized files 9-6
 - editing synced files 9-5
 - restoring archived files 9-7
 - syncd_checkpoint script 9-6
 - syncd_rollback script 9-7
 - syncd_start script 9-8
 - syncd_stop 9-8
 - syncd_stop script 9-8
 - Synchronize Metrics Data command 9-1
 - synchronized files
 - and checkpoint files 9-6
 - archiving 9-6
 - restoring from archive 9-7
 - syntax
 - and editing rules 13-4
 - for comments 13-46
 - for datacenter statement 13-32
 - for globals statement 13-7
 - for topology statement 13-43
 - for wideip statement 13-35
 - syslog utility 10-5
- T**
- technical support resources 1-3
 - time of day load balancing 7-11
 - timer values
 - about 5-18
 - and metrics collection 5-18
 - configuring 5-19
 - Topology
 - using topology records 11-1
 - Topology load balancing mode
 - about 11-1
 - configuring in pools 11-4
 - configuring in wide IPs 11-3
 - in a pool 11-1
 - in wide IPs 11-1
 - topology records
 - about 11-1
 - in topology statements 11-1
 - variables 11-5
 - topology statement 13-42
 - variables 11-5
 - triggers
 - defining 7-2
 - event-based 7-4
 - time-based 7-2
 - TTL values
 - about 5-17
 - and metrics collection 5-17
 - configuring 5-19
- U**
- utilities 12-1
 - syslog 10-5
 - viewing man pages 12-1
- V**
- view of network 6-1
 - virtual servers
 - checking availability 13-10
- W**
- when statement
 - guidelines 7-8
 - in production rules 7-8
 - wide IP production rules 7-2
 - wide IPs
 - about 5-11
 - and DNS zone file management 5-11
 - configuring 5-12
 - syntax 5-14
 - using a last resort pool 5-15
 - wideip statement 13-34
 - wideip.conf file
 - about 13-1
 - adding production rules 7-5
 - configuration requirements 13-1
 - configuring production rules 7-5
 - editing 9-5, 13-1
 - syntax editing rules 13-4
-