BIG-IP® Systems Network Firewall Guide for ICSA Certification

Version 11.0

MAN-0392-00

Product Version

This manual applies to version 11.0 of the BIG-IP® product family.

Publication Date

This manual was published on November 3, 2011.

Legal Notices

Copyright

Trademarks

Export Regulation Notice

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

RF Interference Warning

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

FCC Compliance

This equipment has been tested and found to comply with the limits for a Class A digital device pursuant to Part 15 of FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This unit generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case the user, at his own expense, will be required to take whatever measures may be required to correct the interference.

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

Canadian Regulatory Compliance

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

Standards Compliance

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

Acknowledgments

This product includes software developed by Bill Paul.

This product includes software developed by Jonathan Stone.

This product includes software developed by Manuel Bouyer.

This product includes software developed by Paul Richards.

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

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

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

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

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

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

This product includes software developed by Adam Glass.

This product includes software developed by Christian E. Hopps.

This product includes software developed by Dean Huxley.

This product includes software developed by John Kohl.

This product includes software developed by Paul Kranenburg.

This product includes software developed by Terrence R. Lambert.

This product includes software developed by Philip A. Nelson.

This product includes software developed by Herb Peyerl.

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

This product includes software developed by Chris Provenzano.

This product includes software developed by Theo de Raadt.

This product includes software developed by David Muir Sharnoff.

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

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

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

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

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

This product includes software developed by Christos Zoulas.

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

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

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

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

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

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

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

This product includes the standard version of Perl software licensed under the Perl Artistic License (© 1997,

1998 Tom Christiansen and Nathan Torkington). All rights reserved. You may find the most current standard version of Perl at http://www.perl.com.

This product includes software developed by Jared Minch.

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

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

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

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

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

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

This product includes Hypersonic SQL

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

This product includes software developed by the Internet Software Consortium.

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

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

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

This product includes software developed by the Computer Systems Engineering Group at Lawrence Berkeley Laboratory. Copyright ©1990-1994 Regents of the University of California. All rights reserved.

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

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

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

3. All advertising materials mentioning features or use of this software must display the following acknowledgement:

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

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

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

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

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

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

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

Table of Contents

1
Logging

2
TCP and ICMP Packet Handling

3
Preventing IP Address Spoofing

I

Logging

- Packet logging on BIG-IP ports
- Configuring logging to show the ISO 4-digit date format
- Logging messages for manual system clock changes
- Logging exceptions

Packet logging on BIG-IP ports

For non-session packets dispatched by the network firewall, you can enable or disable logging of these packets on both TMM ports and administrative ports. You do this by configuring a BigDB variable.

In this context, ***non-session*** refers to any packet that does not belong to an existing connection. This includes not only packets used to first establish a connection, but also data packets that appear to belong to a non-existent connection.

Packets that appear to belong to a past or future connection are not considered to be non-session packets unless the BigDB variable **TM.ICSAStrictTCPForwarding** is enabled. For more information, see *TCP packet handling*, on page 2-1.

Examples of non-session packets are packets for certain types of inbound or outbound access requests that are evaluated against a defined access-control policy.

◆ Note

ICMP and ICMPv6 packets are referred to in log messages as non-session packets.

TMM ports

You can enable and disable packet logging for TMM ports, and you can view packet dispositions in log messages.

The location and name of the log file for TMM ports is **/var/log/ICSA**.

Enabling and disabling packet logging

For TMM ports, you configure the BigDB variable **TMM.LogNonSessionPackets**. This variable forces logging of all non-session packets from TMM ports.

To enable or disable packet logging on TMM ports, type one of the following at the system prompt:

```
tmsh modify sys db tmm.lognonsessionpackets value enable
tmsh modify sys db tmm.lognonsessionpackets value disable
```

Viewing packet dispositions in log messages

For TMM ports, the two types of dispositions that can appear in log messages are:

- **accepted**
  Indicates that the BIG-IP system has accepted and processed the packet.
- **rejected**
  Indicates that the BIG-IP system has rejected and discarded the packet.

The following sample log messages show dispositions for TCP, UDP, ICMP, and ICMPv6 packets respectively:

```
2011-09-07T16:51:46-07:00 tmm info tmm[8136]: 01070417: 134: ICSA: non-session TCP packet
accepted, source: 10.10.10.29 port: 33102, destination: 10.10.20.21 port: 80

2011-09-07T16:51:19-07:00 tmm info tmm[8136]: 01070417: 134: ICSA: non-session UDP packet
accepted, source: 10.10.10.29 port: 37818, destination: 10.10.20.21 port: 80

2011-09-07T17:00:44-07:00 tmm1 info tmm1[8137]: 01070417: 134: ICSA: non-session ICMP
packet accepted, source: 10.10.10.13, destination: 10.10.10.29, type code: Echo Reply

2011-09-07T16:49:36-07:00 tmm info tmm[8136]: 01070417: 134: ICSA: non-session ICMPv6
packet accepted, source: 10.10.10.29, destination: 10.10.20.21, type code: Unassigned
```

Administrative ports

You can enable and disable packet logging for the administrative port, and you can view packet dispositions in log messages.

The location and name of the log file for the administrative port is **/var/log/ICSA**.

Enabling and disabling packet logging

For the administrative port, you configure the BigDB variable **ICSA.ForceAdminPacketLogging**. This variable forces logging of all non-session packets from the administrative port.

To enable or disable packet logging on the administrative port, type one of the following at the system prompt:

```
tmsh modify sys db icsa.forceadminpacketlogging value enable
tmsh modify sys db icsa.forceadminpacketlogging value disable
```

Viewing packet dispositions in log messages

For the administrative port, a log message for a packet passing through the firewall indicates that the packet was either accepted or rejected. This log event disposition is indicated by the following strings in the log message:

- **SYN**
  Indicates that the first packet of the connection was permitted (accepted)
- **RST**
  Indicates that the first packet of the connection was denied (rejected)

The following is an example of a log message showing a disposition of RST (denied):

```
2010-12-10T14:50:25-08:00 bigip/1-bigip notice kernel: Packet Logging IN=eth0
OUT=MAC=00:01:d7:c5:6c:c1:00:01:e8:71:99:5a:08:00 SRC=192.168.42.79 DST=172.27.32.230
LEN=40 TOS=0x00 PREC=0x00 TTL=252 ID=38678 DF PROTO=TCP SPT=54706 DPT=443 WINDOW=0
RES=0x00 ACK RST URGP=0
```

Configuring logging to show the ISO 4-digit date format

You can affect the format of dates that appear in log messages. Specifically, you can enable or disable the display of extended dates (that is, the full 4-digit year) in all log files within the **/var/log** directory.

To show the ISO 4-digit date format, use one of the following Syslog commands:

```
tmsh modify sys syslog iso-date enabled
```

```
tmsh modify sys syslog iso-date disabled
```

Logging messages for manual system clock changes

When you manually change the system clock time using the Linux date command, the BIG-IP system logs these messages, in the file **/var/log/audit**. The system generates log messages for both successful and unsuccessful attempts.

The following is an example of the log message that is generated for a successful date change. This date change also includes the time from which the date was attempted to be set.

```
2010-11-30T15:00:00-08:00 local/bigip1 info date[9078]: 01070417:6: AUDIT - user root -
RAW: date change succeeded; attempted time set to 11301500 from 2010-11-30 15:14:28-08:00
```

Logging exceptions

The BIG-IP system does not generate log messages in these cases:

- When the BIG-IP system drops connections (including their packets) during flood conditions, that is, when the BIG-IP device is under heavy load.
- When raw IP Protocol 1 packets are sent with a small enough data payload to preclude an ICMP header.

2

TCP and ICMP Packet Handling

- Overview
- ICMP replay packet handling
- TCP packet handling

Overview

You can configure specific BigDB variables to ensure that the BIG-IP system handles ICMP replay packets and TCP packets in an ICSA-compliant manner.

ICMP replay packet handling

You can configure the way that the BIG-IP system handles ICMP replay packets by configuring the BigDB variable **TM.ICSAICMPReplay**. The possible values for this variable are:

- **allow**
  This value maintains the previous behavior of the BIG-IP system by allowing replayed ICMP errors to pass through the BIG-IP system. This is the default value.
- **discard**
  This value prevents all ICMP errors from passing through the BIG-IP system.
- **detect**
  This value allows the first ICMP error to pass through the BIG-IP system, while discarding any replayed ICMP errors.

For example, to ensure ICSA-compliant ICMP replay packet handling, you can use **tmsh** to configure the **TM.ICSAICMPReplay** variable as follows:

```
tmsh modify sys db tm.icsaicmpreplay value detect
```

TCP packet handling

You can configure the way that the BIG-IP system handles TCP packets by configuring the BigDB variable **TM.ICSAStrictTcpForwarding**. The possible values for this variable are:

- **enable**
  This value prevents packets with invalid TCP flags and invalid RST packets from passing through the BIG-IP system.
- **disable**
  This value maintains the previous behavior of the BIG-IP system. This is the default value.

For example, to ensure ICSA-compliant TCP packet handling, you can use **tmsh** to configure the variable **TM.ICSAStrictTcpForwarding** as follows:

```
tmsh modify sys db tm.icsastricttcpforwarding value enable
```

3

Preventing IP Address Spoofing

- Overview
- Method 1: Creating separate virtual servers
- Method 2: Creating separate packet filters

Overview

A forwarding (IP) type of virtual server can be at risk of accepting packets with spoofed IP addresses. This can occur when the virtual server is configured to accept inbound traffic for any destination IP address, on both the internal and external VLANs. (This configuration is also known as ***reverse path forwarding***, or ***RPF***.)

A forwarding (IP) virtual server configured in this way cannot detect spoofed source packets because the system does not distinguish between an internal network and an external network,

To prevent IP address spoofing in this configuration, you can use either of two approaches:

- You can create a separate virtual server per protocol per VLAN. With this approach, you are allowing inbound traffic, destined for specific networks, on specific VLANs.
- You can create a separate packet filter per VLAN. With this approach, you are rejecting inbound traffic, from specific networks, on specific VLANs.

Method 1: Creating separate virtual servers

To prevent IP address spoofing when using an IP forwarding virtual server, you can configure two separate virtual servers per protocol, one for each traffic direction.

In this configuration, each virtual server has a specific destination network and only accepts traffic that originates from either an external VLAN (inbound traffic) or an internal VLAN (outbound traffic), but not both.

◆ Note

A consequence of choosing the virtual server approach is that for each protocol, you must create a separate virtual server specifying the particular destination network that you want to allow traffic to reach.

Figure 3.1, on page 3-2, shows a sample configuration of two IP forwarding virtual servers for the HTTP protocol.

```
ltm virtual /Common/HTTP_inbound {
    destination /Common/10.10.20.0:http
    ip-forward
    ip-protocol tcp
    mask 255.255.255.0
    profiles {
        /Common/fastL4 { }
    }
    translate-address disabled
    translate-port disabled
    vlans {
        /Common/external
    }
    vlans-enabled
}
ltm virtual /Common/HTTP_outbound {
    destination /Common/10.10.10.0:http
    ip-forward
    ip-protocol tcp
    mask 255.255.255.0
    profiles {
        /Common/fastL4 { }
    }
    translate-address disabled
    translate-port disabled
    vlans {
        /Common/internal
    }
    vlans-enabled
}
```

Figure 3.1 *Sample virtual servers for preventing IP address spoofing*

This example shows that instead of configuring a single virtual server as a wildcard virtual server (with a destination IP address and netmask of **0.0.0.0**), you can configure two virtual servers that each define a specific destination network, for either outbound or inbound traffic. This limits the IP addresses to which each virtual server will forward traffic.

You can create a virtual server configuration similar to the example by using either the BIG-IP Configuration utility or **tmsh.**

To configure a virtual server to accept traffic on VLAN external

1. On the Main tab, expand **Local Traffic**, and click **Virtual Servers**.
2. In Name column, locate and click the name of the relevant IP forwarding virtual server.
   The settings for the virtual server appear.
3. Locate the **Destination** setting.
4. For the **Type** setting, enable **Network**.

5. In the **Address** box, change the IP address to a network IP address that represents the range of internal IP addresses that the virtual server will accept.
6. In the **Mask** box, change the netmask to a netmask for the IP address you specified in the **Address** box.
7. From the **VLAN and Tunnel Traffic** list, select **Enabled on**.
8. In the **Available** box, select the name of an external VLAN and using the Move button, move the name to the **Selected** box.
9. At the bottom of the screen, click **Update**.

To configure a virtual server to accept traffic on VLAN internal

1. On the Main tab, expand **Local Traffic**, and click **Virtual Servers**.
2. In Name column, locate and click the name of the relevant IP forwarding virtual server.
   The settings for the virtual server appear.
3. Locate the **Destination** setting.
4. For the **Type** setting, enable **Network**.
5. In the **Address** box, change the IP address to a network IP address that represents the range of external IP addresses that the virtual server will accept.
6. In the **Mask** box, change the netmask to a netmask for the IP address you specified in the **Address** box.
7. From the **VLAN and Tunnel Traffic** list, select **Enabled on**.
8. In the **Available** box, select the name of an internal VLAN and using the Move button, move the name to the **Selected** box.
9. Click **Update**.

Method 2: Creating separate packet filters

An alternate way of preventing IP address spoofing when using an IP forwarding virtual server is to configure two separate packet filters, one for the external VLAN and one for the internal VLAN.

In this configuration, each packet filter rejects packets from one or more specific networks. Thus, one packet filter prevents packets with spoofed internal source addresses from being accepted by the external VLAN, and the other packet filter prevents packets with spoofed external source addresses from being accepted by the internal VLAN.

Using this method, you can still use a single wildcard virtual server that is configured to allow traffic from all VLANs.

◆ Note

A consequence of using the packet filter approach is that you must specify every source network from which to block traffic, for both inbound and outbound traffic.

Figure 3.2 shows a sample configuration of two packet filters, one for each VLAN. In this example, each packet filter rejects packets from a single source network.

```
net packet-filter /Common/reject_external_on_internal {
    action reject
    order 10
    rule "( src net 10.10.10.0/24 )"
    vlan /Common/internal
}
net packet-filter /Common/reject_internal_on_external {
    action reject
    order 5
    rule "( src net 10.10.20.0/24 )"
    vlan /Common/external
}
```

Figure 3.2 *Sample packet filters for preventing IP address spoofing*