DEPLOYMENT GUIDE

Tuning the OneConnect Feature on the BIG-IP Local Traffic Manager

Version 1.0

Table of Contents

Introduction to OneConnect

Testing OneConnect

Appendices

Introduction to OneConnect

OneConnect™ is a feature of the BIG-IP LTM system that improves web application performance and decreases server load by reducing the concurrent connections and connection rate on back-end servers. This deployment guide explains the OneConnect feature, shows how to tune BIG-IP LTM parameters to get the maximum benefit, and includes specific test results.

OneConnect reuses TCP connections to each server for multiple clients. After the BIG-IP LTM has sent a request and received a complete response, these connections are put in a *Connection Reuse Pool*. When new clients create new TCP connections to the BIG-IP LTM, instead of creating new TCP connections to the servers, the BIG-IP LTM may pick existing connections from the Connection Reuse Pool.

When a connection to a server is first established, the kernel of the server operating system must allocate memory for the TCP connection state and data buffers, then notify the web server process. The web server process may have a thread ready to use, have to setup another thread, or even have to create a full copy of the main web server process. Regardless of implementation details, it is always computationally intensive to setup new connections compared to receiving requests on an already-open connection. This is why HTTP keep-alives were invented and made a standard part of HTTP/1.1. For each established connection, a web server consumes memory for the TCP connection itself (state/buffers stored in the kernel, likely only 8-64KB for an idle connection depending on memory pressure and previous use). More importantly, each connection consumes web server threads, and for most web applications, several megabytes of unique per-thread memory. As concurrency goes from tens to hundreds, or possibly hundreds to thousands (depending on application), the overhead of selecting which process/thread to run, and for how long, increases dramatically, reducing the effective CPU capacity of the server (known as context-switching overhead).

Fewer open connections and fewer connections opening/closing means lower resource consumption per server. Minimizing resource utilization per server enables server consolidation therefore allowing operational cost savings.

The following sections describe how to fine-tune BIG-IP parameters to lower per server concurrent connections and how to insulate any one server from dealing with a spike of requests. This includes details for tuning the OneConnect profile, the TCP profile itself, and the connection limits on individual pool members.

To provide feedback on this deployment guide or other F5 solution documents, contact us at *solutionsfeedback@f5.com*.

Recommended reading

We recommend reading the following solutions on Ask F5. Ask F5 requires a free user account.

- *SOL7208 Overview of the OneConnect profile*
- The BIG-IP LTM manual: ***BIG-IP Configuration Guide for BIG-IP Local Traffic Management***, specifically the section on "Configuring a OneConnect Profile" in the *Using Additional Profiles* chapter.

The reading the following solutions is optional, but may provide more background:

- *SOL5911 Managing connection reuse using OneConnect source mask*
- *SOL4816 Using the X-Forwarded-For HTTP header to preserve the original client IP address for traffic translated by a SNAT*
- *SOL7751 Overview of Clustered Multi-Processing (CMP)*
- *SOL8125 Overview of the TCP profile connection flow*
- *SOL3422 Overview of Content Spooling*

Product versions and revision history

Product and versions applicable to deployment guide:

Product Tested	Version
BIG-IP LTM	10.0. Note: While not specifically tested as of version 1.0 of this guide, this document will also apply to BIG-IP LTM version 9.x.

Revision history:

Document Version	Description
1.0	New deployment guide

Configuration overview

The following are notes about this configuration:

- To use OneConnect, TCP, HTTP and OneConnect profiles must be applied to a standard virtual server.
- Appropriate TCP profiles should be used on both the client- and server-side. For example, clients from Internet (or WAN) should use the **tcp-wan-optimized** profile. Servers, which are usually on the LAN, should use the **tcp-lan-optimized** profile.
- When using OneConnect, the web server may see the requests as originating from the source IP address of connections from Connection Reuse Pool. Note this may not be the original client IP address. If the web servers are required to log the original client IP address for requests,

you should enable the **Insert X-Forwarded For** option in the HTTP profile. (See **Configuring HTTP standard profile settings** in the ***Configuration Guide for BIG-IP Local Traffic Management*** and SOL4816 for more information).

- The ratio of many TCP connections on the client side to fewer on the server side can be substantial. As a result, with OneConnect, the total number of servers can be reduced dramatically. The exact ratio is specific to each deployment and depends on many factors, including client latency, server response delay and object size. These factors are discussed in detail below.
- At this time, the Least Connections load balancing method is not recommended as it provides a lower connection-reuse rate than other available methods. F5 is tracking this issue in CR120223
- If the servers being load balanced are only serving static content, and both clients and servers are on the same LAN, OneConnect may not provide a significant performance improvement. For most applications, however, OneConnect both reduces server resource consumption and increases the capacity to handle more users.

The following diagram shows how OneConnect can reduce server-side TCP connections by reusing the previously opened idle connections on the server for multiple clients. To simplify this example, there are only 2 clients (client 1 and client 2) and 1 server (server A). Assume there is no connection between any of the devices at the beginning.:

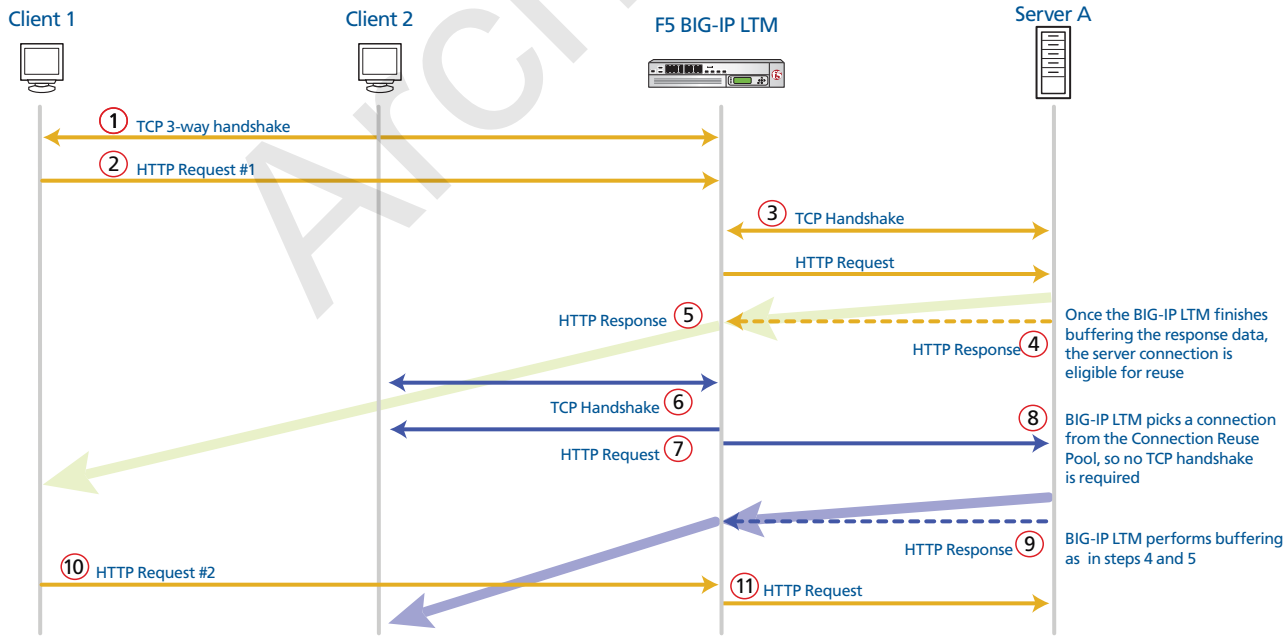

Figure 1 *OneConnect connection flow diagram*

1. Client 1 performs a 3-way TCP handshake to the BIG-IP LTM

2. Client 1 sends an HTTP request #1 to the BIG-IP LTM
3. Once the BIG-IP LTM receives a complete request from client (assume the BIG-IP selects Server A based on load balancing criteria), the BIG-IP performs a TCP handshake and forwards client's HTTP request to Server A.
4. Server A sends an HTTP response back to the BIG-IP LTM. Once the LTM buffers the complete response from server A, the server's connection is moved to Connection Reuse Pool and is left open. The thick arrow represents a larger amount of data compared to the HTTP request. The declining arrow represents delay time. The time to finish sending data back to the client may take longer because the client is usually from the Internet.
5. Step 5 starts at the same time as Step 4. The BIG-IP LTM spoon-feeds the HTTP response to client 1. Usually clients are from the Internet so it may take more time for them to receive a complete response.
6. While the BIG-IP LTM is spoon-feeding the HTTP response to Client 1, Client 2 performs a TCP 3-way handshake to the BIG-IP LTM.
7. Client 2 sends an HTTP request to the BIG-IP LTM. This step also occurs before step 5 has finished.
8. The BIG-IP LTM picks a connection from the Connection Reuse Pool, which in this case is the connection to Server A from Step 4. The BIG-IP LTM is reusing this previously idle connection, thus reducing the number of server-side TCP connections for multiple clients.
9. Server A sends the HTTP response back to the BIG-IP LTM. The BIG-IP performs the same buffering as in Steps 4 and 5.
10. Client 1 sends a second HTTP request to the BIG-IP LTM.
11. The BIG-IP LTM picks server A from the Connection Reuse Pool, like in Step 8.

Reducing concurrent server connections

To reduce concurrent server connections, you modify the P**roxy Buffer High/Low** option both in the client and server-side TCP profile (see *SOL3422* and *SOL8125* for more information).

Proxy Buffer High determines the buffer level at which the BIG-IP LTM stops receiving data from server. The BIG-IP LTM can buffer all or most of the response from a fast server (typically on the same LAN) before spoon-feeding the data to a slow client (typically clients have less resources and are not tuned for high speed, and are likely to be distant from the servers). Once a complete response has been buffered, the server connection can be reused for other clients. The higher the **Proxy Buffer High** value, the sooner the connection of a particular server moves into the Connection

Reuse Pool. Ideally, the **Proxy Buffer High** value should be higher than Maximum Object Size or as close to Maximum Object Size as possible. Note that the BIG-IP LTM consumes more memory as the **Proxy Buffer High** value increases.

Proxy Buffer Low specifies buffer level at which LTM starts receiving data again. The **Proxy Buffer Low** value should be slightly lower than Proxy Buffer High value. This allows the BIG-IP LTM to continue buffering sooner. One recommendation for Proxy Buffer Low is:

```
Proxy Buffer Low = Proxy Buffer High - (N x server-side Maximum Segment Size)
```

Where **N** is the desired number of maximum-size packets on the server-side. For example, if N=2 and the Maximum Segment Size (MSS) is 1500 (approximated), the BIG-IP LTM continues buffering as soon as the available buffer is greater or equal to 3,000 Bytes. This prevents a connection from being stalled and efficiently uses server-side bandwidth.

Proxy Buffer Low should not be lower than 75% of **Proxy Buffer High**.

Furthermore, the number of the server's concurrent connections also depends on the OneConnect **Source Mask** option. The OneConnect **Source Mask** setting is applied to the source IP address of an incoming request to determine its eligibility for connection reuse. (See *SOL5911* for more information). The lower the **Source Mask**, the smaller the number of server connections. For example:

- A OneConnect profile with a source mask of 255.255.255.255 only aggregates connections originating from the same client IP address.
- A OneConnect profile with a source mask of 255.255.255.0 aggregates connections from client IP addresses sharing the same last octet.
- A OneConnect profile with a source mask of 0.0.0.0 shares reused connections across all client requests.

Lastly, the minimum number of connections to the back-end servers is equal to TMM processes multiplied by number of pool members. This is because client connections are distributed to multiple TMMs and each TMM has separate connections to back-end servers. For example, if you have 2,500 pool members, and are using a VIPRION with 16 TMMs, you are going to have at least 40,000 (16x2,500) connections to the servers.

Maintaining server concurrent connections at a desired value

The **Maximum Size** option in a OneConnect profile should be set to the desired number of server concurrent connections. **Maximum Size** specifies the maximum number of connections that the system holds in the Connection Reuse Pool.

When the amount traffic mandates a number of server connections that is higher than the Maximum Size value, the BIG-IP LTM may open extra TCP connections to servers, and close them once they are no longer needed. When the amount of traffic requires a number of server connections which

is lower than **Maximum Size**, the BIG-IP LTM maintains connections to servers at the peak value until **Maximum Age**, **Maximum Reuse**, or **Idle Timeout is** reached.

Higher **Maximum Age**, **Maximum Reuse**, or **Idle Timeout** values increase the connection-reuse rate, at the expense of redundant connections in the Connection Reuse Pool.

Maximum Size is per TMM (Traffic Management Microkernal). On Clustered Multi-Processing platforms (CMP, see *SOL7751*), the desired number is divided by number of TMMs.

```
Maximum Size  =  Desired number of server concurrent connections
                 ----------------------------------------------
                                    TMMs
```

In a future release, Maximum Size will be per system as opposed to per TMM. F5 Networks is tracking this request as CR120225.

Keeping server concurrent connections low

When the amount of traffic requires a number of server connections that is lower than the **Maximum Size** value, the BIG-IP LTM maintains connections to servers at peak value until **Maximum Age**, **Maximum Reuse**, or **Idle Timeout** values are reached. So after a traffic peak, Connection Reuse Pool may contain redundant connections. Lowering the value of **Maximum Age**, **Maximum Reuse**, or **Idle Timeout** cleans redundant connections in Connection Reuse Pool faster. (See *SOL7208* for a detailed description of these OneConnect parameters).

Protecting servers from a traffic spike

To protect servers from traffic spikes, the **Pool Member Connection Limit** should be set to the upper limit of each server. New client connections are reset if all servers reach the limit.

In Cluster Multi-Processing (CMP, see *SOL7751*), this connection limit is enforced per processor. For example, on a BIG-IP 6800 with two processors, if a pool member is configured with a connection limit of 100, those connections can be spread across the two available processors. Therefore, each TMM would have a limit of 50 connections.

However, with VIPRION each blade performs its own connection limit setting. For example, if a VIPRION with four blades has a connection limit of 100, each blade divides the 100 connection limit by four, and assigns 25 connections per TMM. Since each blade has four CPUs, which amounts to a total of 16 TMM, the total connection limit would be 400.

F5 Networks Product Development is tracking this connection limit issue on CMP platforms as CR93185. *SOL8457* provides more detail on this topic.

Testing OneConnect

In this section, we show the results of testing the various OneConnect parameters discussed in the first section of this guide. This demonstrates the actual reduction in server loading.

Test diagram

The following diagram is a logical representation of our testing environment.

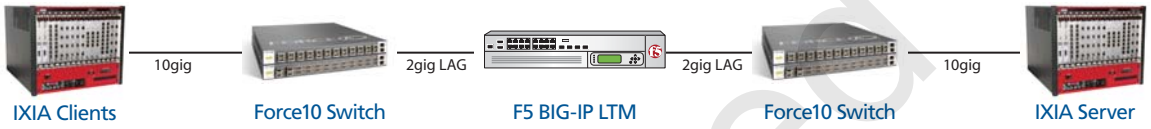

Figure 2 *Logical configuration diagram*

In Figure 1:

- Both the Ixia clients and server are on the same chassis.
- The client-side Force10 and server-side Force10 are the same switch
- The connection between Ixia and Force10 is 10-gig Ethernet
- The connection between the Force10 and the BIG-IP is 2 gig LAG

Test conditions

The following are the conditions of each of the tests:

- The Ixia client generates and maintains 24,000 concurrent connections to the BIG-IP LTM
- Ramp up time to the maximum number of concurrent connections is 250 seconds
- The client sends 100 HTTP requests for each TCP connection
- There are 24 back-end servers available for the connection pool
- The maximum object size is 262,144 bytes

Test Results

In this section, we show the results of testing the BIG-IP LTM using the tuning parameters discussed previously.

Test 1: Default test without using OneConnect

Without using a OneConnect profile, the BIG-IP LTM opens the exact same number of connections to the servers as it receives from the clients - 24,000 in this case. This is represented in Figure 3: the number of client connections and server connections track exactly.

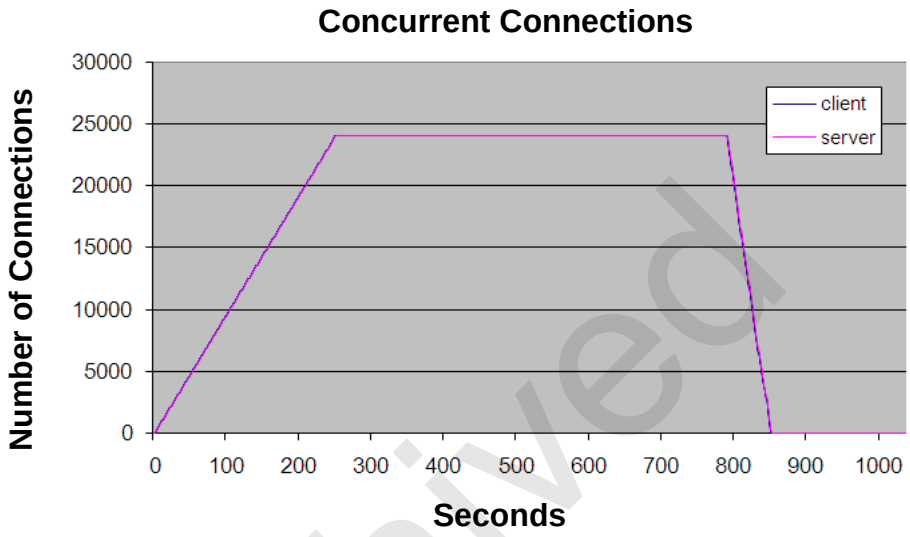

Figure 3 *Concurrent results without using OneConnect*

Test 2: Applying the OneConnect profile

In this test, we apply the OneConnect profile. The following are notes about this test:

- For the TCP profiles, we applied the default **tcp-wan-optimized** (client) and **tcp-lan-optimized** (server) to the virtual server.
- We recommend using a high value for OneConnect **Maximum Size** and **Maximum Reuse** settings to ensure number of connections to the servers is not limited by these parameters. In our test, both values are set to **10,000**.
- After applying the OneConnect profile, the BIG-IP LTM reduces the number of connections to the servers to less than 10%. Figure 4 shows the reduction in server side connections.

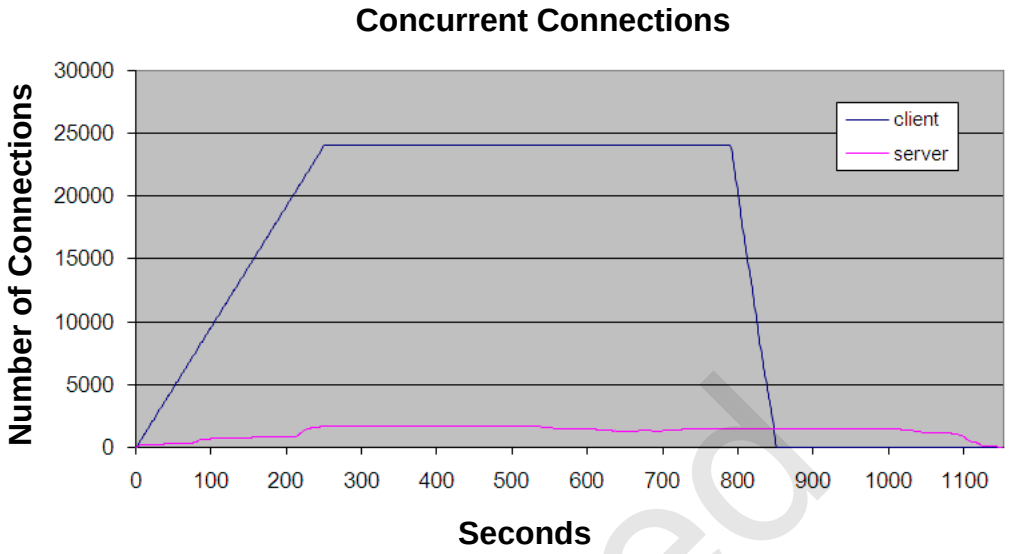

Figure 4 *Test results using the OneConnect profile*

Figure 5 shows only the open connections of the server. We use this graph to compare to results from the next sections.

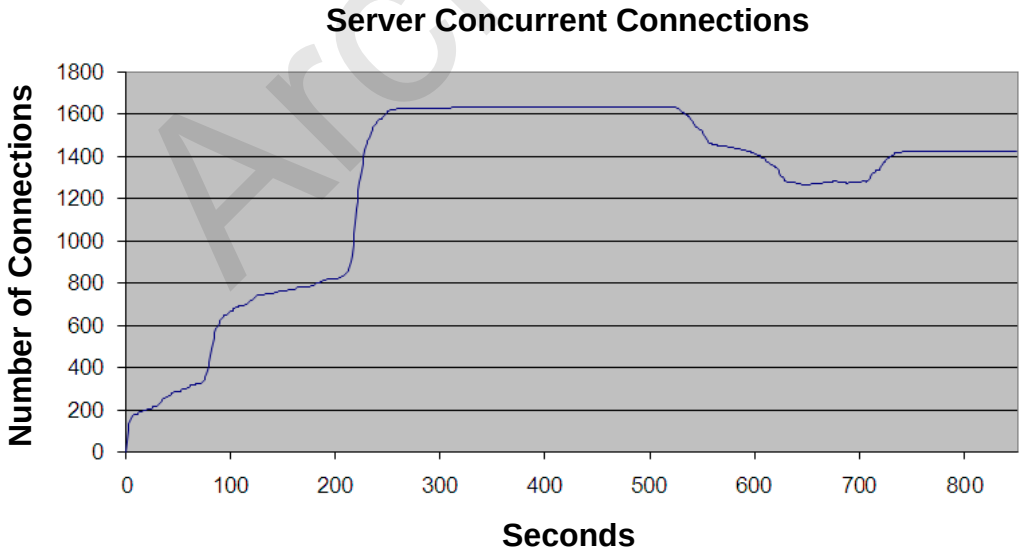

Figure 5 *Concurrent connections on the server*

Test 3: Increasing the Proxy Buffer High/Low settings

In this test, we show the results of increasing the **Proxy Buffer High** and **Proxy Buffer Low** values, which reduces the number of concurrent connections on the server. These settings are found in the TCP profiles. The following are notes about this test:

- We set the **TCP Proxy Buffer High** value to 280,000 bytes; this should be a higher number than Maximum Object Size (maximum object size is 262,144 bytes)
- We set the **TCP Proxy Buffer Low** value to 277000 bytes. This is approximately 2 MSS (2 x 1500) lower than TCP Proxy Buffer High
- Figure 6 shows that servers' concurrent connection is reduced by 66% when compared to the previous test.

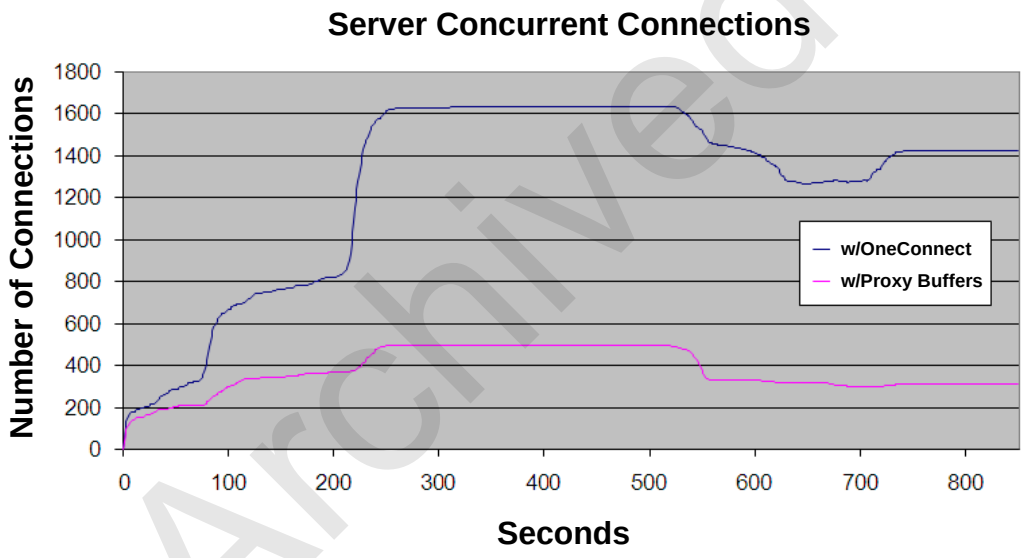

Figure 6 *Concurrent connections on the server after tuning the proxy buffers*

Test 4: Setting the OneConnect Maximum Size value

In this test, we set the OneConnect **Maximum Size** value to maintain the open connections on the server at optimal level. Notes about this test:

- According to the server concurrent connection graph there is a peak around 250 seconds.
- After the peak, the BIG-IP LTM maintains the server connections until it reaches the timeout of 300 seconds.
- We set our desired number of server concurrent connections to 384.
- The device under test has 2 TMMs. So the Maximum Size per TMM is 384 divided by 2, or 192.

- You can see in Figure 7 that after the peak, number of server concurrent connections is back to the desired value.

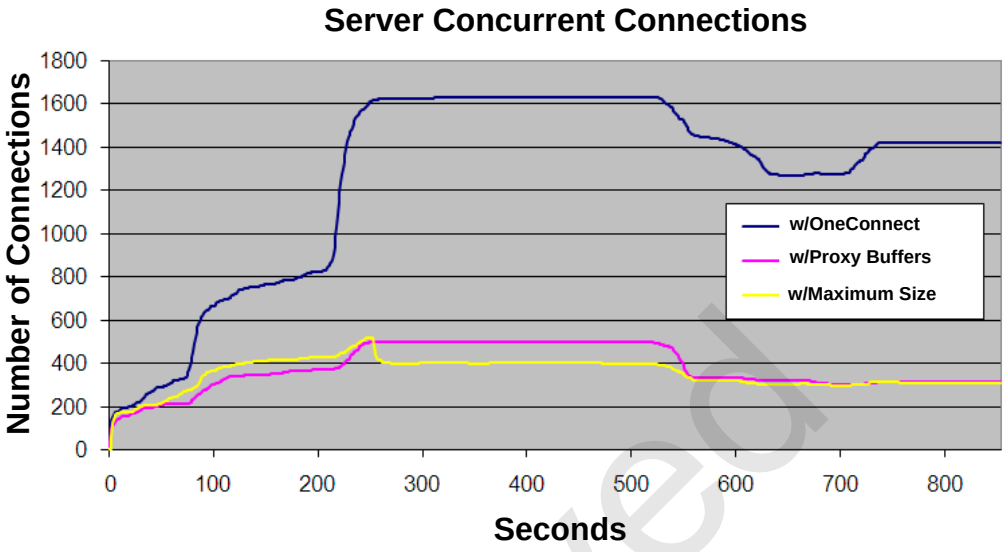

Figure 7 *Results after setting the OneConnect Maximum Size value*

Test 5: Lowering the OneConnect Max Reuse value

In this test, we show how lowering the OneConnect **Maximum Reuse** value results in a smaller Connection Reuse Pool.

Notes about this test:

- We decreased the OneConnect **Maximum Reuse** value to 1,000
- Figure 8 shows redundant connections in Connection Reuse Pool are cleaned up faster.

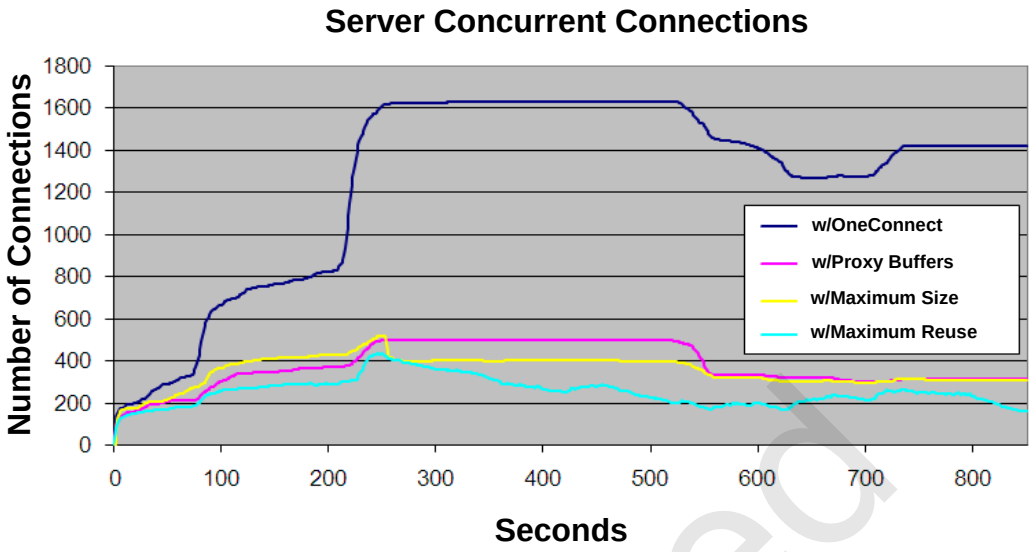

Figure 8 *Lowering the Maximum Reuse value*

Another benefit of Oneconnect is that it reduces latency by eliminating the time taken to repeatedly open connections. Figure 9 shows the TTFB (Time To First Byte) comparison between Test 1 and Test 5. At steady state, OneConnect reduces TTFB by about 15%. However, this reduction may not be noticeable when comparing TTLB (Time To Last Byte) if the TTLB is very high. In this case, TTFB is reduced by 20 milliseconds while TTLB is around 3 seconds; so it is not noticeable when comparing TTLB.

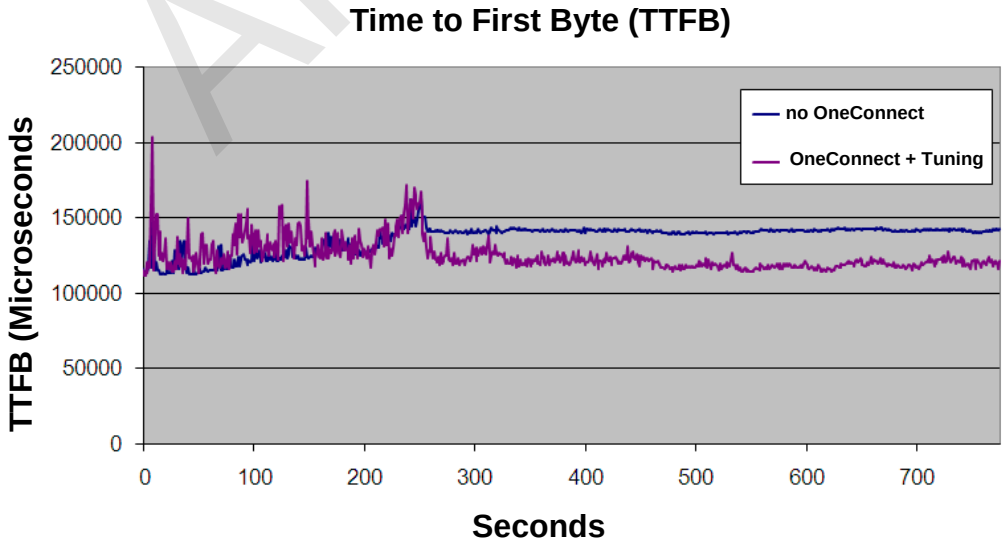

Figure 9 *Time to First Byte comparison*

Test 6: Protecting servers from a traffic spike

In this section, we show how the BIG-IP LTM can protect the server from a traffic spike using the **Pool Member Connection Limit**. The following are notes about this test:

- To simulate a traffic spike, the ramp up time is reduced to 125 seconds. From the No Limit line in Figure 10, the traffic spike at around 83 seconds causes the BIG-IP LTM to open more than 700 connections to the servers. When the limit is set, the LTM limits connections to servers as shown in Limit line in Figure 10.
- In this case, the **Pool Member Connection Limit** is set to 24 per server. Total connection should be 24 (per server) x 24 servers, or 576. However, the Limit line in Figure 10 shows 528 connections during peak. This is an expected behavior. When OneConnect is applied, the actual limit per TMM is minus by 1. This is because one connection has been reserved for internal use.

  For example, our **Pool Member Connection Limit** is set to 24. The device under test has 2 TMMs, so the Limit per TMM is 24/2 or 12. The actual limit will be 11 per TMM or 22 per system. Hence, the actual limit for 24 servers is 22 (per server) x 24 servers or 528.

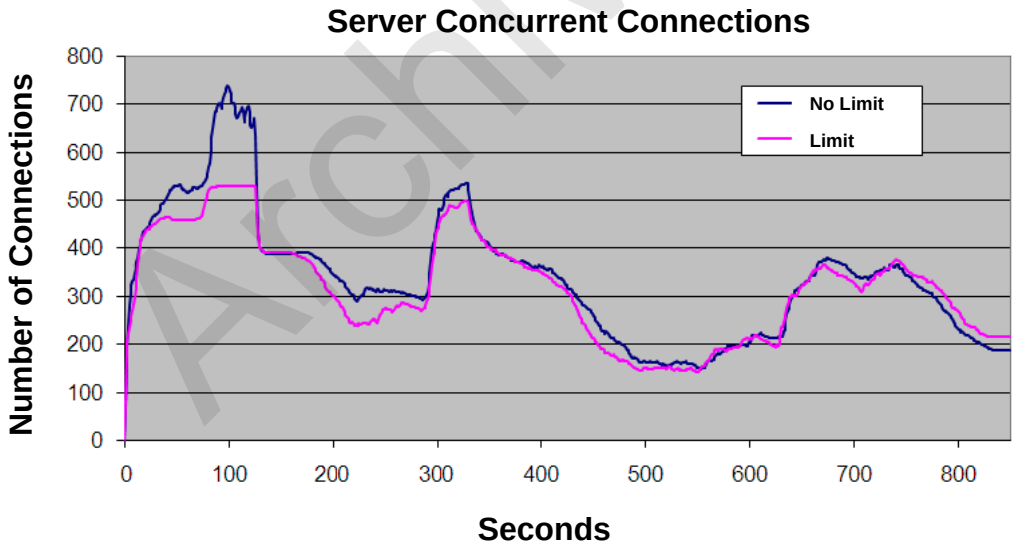

Figure 10

Conclusion

In this document, we tuned the OneConnect feature in the following ways:

- Reducing the concurrent connections per server by increasing Proxy Buffer High and Low
- Maintaining server concurrent connections at a specific level by setting the Maximum Size
- Keeping the server concurrent connections low by reducing Maximum Reuse (or Idle Timeout or Max Age)
- Protecting servers from spike by applying a pool member limit.

Appendix A: Detailed test configuration

Device under test:

- BIG-IP 6800
- LTM version 10.0.0 HF1 build 5460

Switch:

- Force10 C300

Client:

- Ixia 10/100/1000 ASM XMV 12X (10G Aggregated)
- 24 source IP
- 100 ms latency (round-trip)
- 24,000 concurrent connections
- Each TCP connection generates 100 HTTP requests
- There is a 60 seconds pause (think time) after every 10 HTTP requests

Server:

- Ixia 10/100/1000 ASM XMV 12X (10G Aggregated)
- Simulated 24 servers
- 10 ms server response delay

Web pages: randomly generated 100 pages which includes

- 48 x 1024 bytes pages
- 13 x 16384 bytes pages
- 19 x 65535 bytes pages
- 13 x 131072 bytes pages
- 7 x 262144 bytes pages

Test timeline:

Tests 1 - 5:

- Take 250 seconds, to ramp up from 0 to 24,000 client concurrent connections
- Maintain 24,000 concurrent connections for 540 seconds
- Takes 60 seconds to ramp down to 0
- Maintain at 0 concurrent connections for 350 seconds (to ensure the BIG-IP LTM disconnects the connections to the servers after a 300 second default timeout).

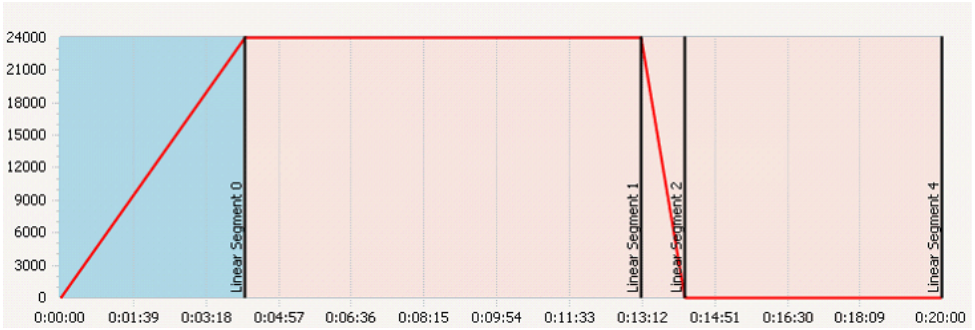

Figure 11 *Tests 1-5 timeline*

Test 6:

- Ramp up time is reduced from 250 to 125 seconds

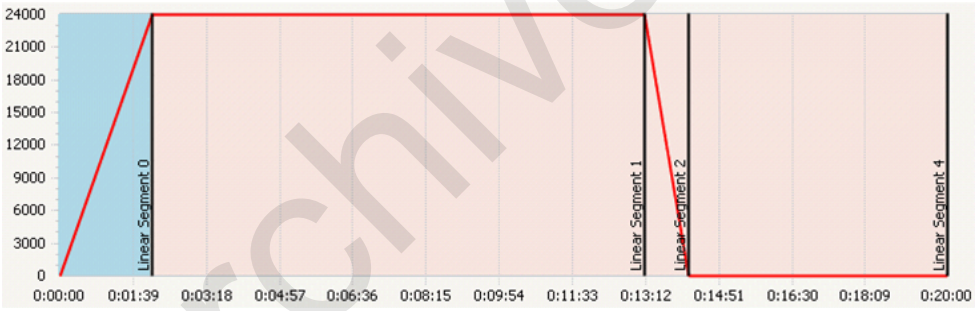

Figure 12 *Test 6 timeline*

Appendix B - BIG-IP Configuration

Test 1:

```
# IP address and VLAN configuration are omitted
b pool p members     10.110.0.101:http     10.110.0.102:http     10.110.0.103:http \
      10.110.0.104:http     10.110.0.105:http     10.110.0.106:http \
      10.110.0.107:http     10.110.0.108:http     10.110.0.109:http \
      10.110.0.110:http     10.110.0.111:http     10.110.0.112:http \
      10.110.0.113:http     10.110.0.114:http     10.110.0.115:http \
      10.110.0.116:http     10.110.0.117:http     10.110.0.118:http \
      10.110.0.119:http     10.110.0.120:http     10.110.0.121:http \
      10.110.0.122:http     10.110.0.123:http     10.110.0.124:http
b snatpool s members       10.110.1.1      10.110.1.2      10.110.1.3    \
      10.110.1.4      10.110.1.5      10.110.1.6    \
      10.110.1.7      10.110.1.8      10.110.1.9    \
      10.110.1.10      10.110.1.11      10.110.1.12    \
      10.110.1.13      10.110.1.14      10.110.1.15    \
      10.110.1.16      10.110.1.17      10.110.1.18    \
      10.110.1.19      10.110.1.20      10.110.1.21    \
      10.110.1.22      10.110.1.23      10.110.1.24
b profile tcp mytcp-lan defaults from tcp-lan-optimized
b profile tcp mytcp-wan defaults from tcp-wan-optimized
b virtual v snatpool s pool p destination 10.109.0.100:http ip protocol tcp  \
      profiles mytcp-lan serverside mytcp-wan clientside
```

Test 2:

```
b profile http myhttp defaults from http
b profile oneconnect myoneconnect max size 10000 max reuse 10000
b virtual v profile myhttp myoneconnect add
```

Test 3:

```
b profile tcp mytcp-lan proxy buffer high 280K
b profile tcp mytcp-lan proxy buffer low 277K
```

Test 4:

```
b profile oneconnect myoneconnect max size 192
```

Test 5:

```
b profile oneconnect myoneconnect max reuse 1000
```

Test 6:

```
b pool p members all limit 24
```

Appendix C - Ixia HTTPclient command list

```
<?xml version="1.0" ?>
<ActionList>
   <Profiles/>
   <Actions>
     <action value="GET,DUT1:80,/#1024.html?client=000_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=001_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#131072.html?client=002_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#65535.html?client=003_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=004_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=005_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#65535.html?client=006_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=007_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#65535.html?client=008_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=009_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="{Think},,,None,60000-60000,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=010_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=011_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#131072.html?client=012_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=013_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#65535.html?client=014_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#131072.html?client=015_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=016_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#262144.html?client=017_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#131072.html?client=018_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=019_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="{Think},,,None,60000-60000,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=020_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=021_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
```

```
    <action value="GET,DUT1:80,/#1024.html?client=022_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#16384.html?client=023_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#131072.html?client=024_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=025_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=026_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=027_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=028_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=029_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="{Think},,,None,60000-60000,,-1
"/>
    <action value="GET,DUT1:80,/#262144.html?client=030_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#16384.html?client=031_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=032_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#131072.html?client=033_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#131072.html?client=034_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#262144.html?client=035_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=036_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#65535.html?client=037_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#262144.html?client=038_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#131072.html?client=039_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="{Think},,,None,60000-60000,,-1
"/>
    <action value="GET,DUT1:80,/#65535.html?client=040_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#65535.html?client=041_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#131072.html?client=042_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#131072.html?client=043_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=044_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=045_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#65535.html?client=046_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=047_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
```

```
     <action value="GET,DUT1:80,/#16384.html?client=048_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#131072.html?client=049_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="{Think},,,None,60000-60000,,-1
"/>
     <action value="GET,DUT1:80,/#16384.html?client=050_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#65535.html?client=051_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#16384.html?client=052_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=053_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=054_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#65535.html?client=055_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#65535.html?client=056_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#65535.html?client=057_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=058_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#65535.html?client=059_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="{Think},,,None,60000-60000,,-1
"/>
     <action value="GET,DUT1:80,/#131072.html?client=060_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=061_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=062_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#16384.html?client=063_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=064_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=065_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=066_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=067_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#65535.html?client=068_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=069_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="{Think},,,None,60000-60000,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=070_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#1024.html?client=071_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
     <action value="GET,DUT1:80,/#65535.html?client=072_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
```

```
    <action value="GET,DUT1:80,/#1024.html?client=073_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=074_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=075_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#65535.html?client=076_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=077_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#262144.html?client=078_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=079_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="{Think},,,None,60000-60000,,-1
"/>
    <action value="GET,DUT1:80,/#16384.html?client=080_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#16384.html?client=081_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#262144.html?client=082_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#65535.html?client=083_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=084_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#16384.html?client=085_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#262144.html?client=086_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#16384.html?client=087_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#16384.html?client=088_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=089_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="{Think},,,None,60000-60000,,-1
"/>
    <action value="GET,DUT1:80,/#131072.html?client=090_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#65535.html?client=091_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=092_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#65535.html?client=093_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=094_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=095_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#16384.html?client=096_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#16384.html?client=097_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
    <action value="GET,DUT1:80,/#1024.html?client=098_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
```

```
     <action value="GET,DUT1:80,/#1024.html?client=099_($port-id)_($sourceip)_($sourceport),None,,,-1
"/>
   </Actions>
</ActionList>
```

Appendix IV - Ixia Ixload configuration

For Tests 1 - 4:

http://www.f5.com/solutions/deployment-guides/files/step0-4.rxf

For Test 6:

http://www.f5.com/solutions/deployment-guides/files/step5.rxf