F5 STUDY GUIDE

301a – F5 Certified Technology Specialist, LTM: Architect, Setup & Deploy

Eric Mitchell
Channel SE, East US and Federal
F5 Networks

Contents

Overview

Welcome to the 301a - LTM Specialist compiled Study Guide. The purpose of this guide is to help you prepare for the F5 301a - LTM Specialist exam. The contents of this document are based on the 301a - LTM Specialist Blueprint Guide.

This study guide provides students with some of the basic foundational knowledge required to pass the exam.

This study guide is a collection of information and therefore not a completely original work. The majority of the information is compiled from F5 sources that are located on Internet. All of the information locations are referenced at the top of each topic instead of in an Appendix of this document. This was done to help the reader access the reference the linked information easier without having to search through a formal appendix. This guide also references the same books as the exam Study Guide for each topic when applicable for consistency.

F5 Networks provides the 301a - LTM Specialist Study Guide as a study guide. The Resource Guide is a list of reading material that will help any student build a broad base of general knowledge that can assist in not only their exam success but in becoming a well rounded systems engineer. The Study Guide will be available to the candidate once they are qualified for the 301a - LTM Specialist exam.

Taking certified F5 LTM training, such as Administering BIG-IP v11 and Configuring BIG-IP LTM v11, will surely help with the topics of this exam but does not teach directly to the exam content. Hands on administrative experience with the BIG-IP platform licensed with LTM will reinforce many of the topics contained in the 301a - LTM Specialist exam.

The F5 Certified BIG-IP Administrator (F5-CA), which is made up of the 101 - Application Delivery Fundamentals and 201 - TMOS Administration exams, stand as a pre-requisite to this exam.

This guide was prepared by an F5 employee but is not an official F5 document and is not supported by F5 Networks.

Reading = Knowledge = Power

Printed References

These referenced books are important and should be considered basic reading material for this exam. If you have a newer copy of the material that is fine, be aware that the exam is based on the 11.2 version and content could have changed.

(Ref:1) Configuring BIG-IP Local Traffic Manager v11.2. v11.2.0 Edition. F5 Networks Training Course Manual.

(Ref:2) Administering BIG-IP v11.2. v11.2.0 Edition. F5 Networks Training Course Manual.

(Ref:3) Troubleshooting BIG-IP v11.2. v11.2.0 Edition. F5 Networks Training Course Manual.

(Ref:4) Developing iApps for BIG-IP v11.2. v11.2.0 Edition. F5 Networks Training Course Manual.

Introduction

F5 - 301a Local Traffic Manager Specialist Exam

The F5 BIG-IP Local Traffic Manager (LTM) increases an application's operational efficiency and ensures peak network performance by providing a flexible, high-performance application delivery system. With its application-centric perspective, LTM optimizes your network infrastructure to deliver availability, security, and performance for critical business applications. Although the Exam Blueprint is not written in a structure that presents topics in an educational order, it does provide all of the necessary building blocks. The Certified LTM Training classes from F5 will help with many of the scenario-based topics on the test. An LTM Specialist must be proficient with all aspects Architecture, Setup and Deployment of the LTM within a network.

Overview of SNAT Features

Traffic Management Shell

Although it is not mentioned in the blueprint as a requirement, a candidate should not focus only on the GUI interface for management of the LTM platform. Some test questions will refer to the command line interface (CLI) TMSH commands. You should take time to understand where in the CLI that common commands are issued so you can not only correctly answer the questions presented on the exam but also have enough knowledge of the CLI structure to eliminate bad commands from your question's answer choices.

Try building your vLab environment from command line to gain CLI proficiency.

SECTION 1 – ARCHITECT AN APPLICATION

Objective - 1.01 - Given an expected traffic volume, determine the appropriate SNAT configuration

1.01 – Explain when SNAT is required

Overview of SNAT Features

What is SNAT and when is it required?

A Secure Network Address Translation (SNAT) is a configuration object that maps the source client IP address in a request to a translation address defined on the BIG-IP device. When the BIG-IP system receives a request from a client, and if the client IP address in the request is defined in the origin address list for the SNAT, the BIG-IP system translates the source IP address of the incoming packet to the SNAT address.

A SNAT can be used by itself to pass traffic that is not destined for a virtual server. For example, you can use a SNAT object to pass certain traffic (such as DNS requests) from an internal network to an external network where your DNS server resides. A SNAT can also be used in conjunction with a virtual server to translate the source IP address of an incoming packet (with no SNAT configured, no source address translation takes place, and destination address translation takes place as separately configured in the Virtual Server properties). You can also use a SNAT to ensure that response traffic is returned through the BIG-IP system without requiring other outbound non-load balanced traffic to also route through the BIG-IP system, and without requiring any changes in the router or server's configuration. SNAT is also a critical component in one-armed configurations, preventing the server from responding directly to the client.

Port exhaustion or collisions may occur under heavy usage or special client traffic patterns. As a result, connections that cannot be translated due to lack of available ports on a given translation address may be dropped.

When a SNAT is configured on the BIG-IP system (either by itself or in conjunction with a virtual server), the source address of each connection is translated to a configured SNAT address, and the source port is mapped to a port currently available for that address. By default, the BIG-IP system attempts to preserve the source port, but if the port is already in use on the selected translation address, the system also translates the source port.

Each SNAT address, like any IP address, has only **65535** ports available. This is a limit of the TCP and User Datagram Protocol (UDP) protocols, since they use a 16-bit unsigned integer (thus ranging from 0 to 65535) to specify the source and destination ports. However, each SNAT address can potentially have to process more

than 65535 concurrent connections, as long as each socket pair is unique. A socket pair is defined by a 4-tuple structure consisting of the following elements:

- Source IP address
- Source port
- Destination IP address
- Destination port

For example, a given SNAT address can continue to use the same source port as long as the remote socket is unique, thus allowing the SNAT address to process more than 65535 concurrent connections.

For example:

SNAT address and port Remote socket

- `10.1.1.1:1234 ----------> 10.1.1.200:80`
- `10.1.1.1:1234 ----------> 10.1.1.201:80`
- `10.1.1.1:1234 ----------> 10.1.1.200:8080`
- `10.1.1.1:1234 ----------> 10.1.1.201:8080`

Note: When SNAT is used in conjunction with a virtual server that load balances connections to a pool; the remote socket is the IP address and port of the chosen pool member. Therefore, assuming a certain SNAT address is configured on only one virtual server, the SNAT address is able to process approximately 65535 concurrent connections for each pool member in the pool (each unique remote socket).

While the uniqueness of remote sockets depends entirely on your specific configuration and traffic, for simplicity you should think of 65535 concurrent connections as the maximum capacity for any given SNAT address. If you think more than 65535 connections may require translation, you should configure more SNAT addresses (for example, using a SNAT pool).

1.01 – Describe the benefit of using SNAT pools

Overview of SNAT Features

SNAT Pools

A SNAT pool represents a logical group of translation addresses that you configure on the BIG-IP system.

When a single IP address is used to SNAT traffic, it has a limit of 65535 ports that can be used for port mapping on the IP address. SNAT connections can fail if a large number of client requests are traversing a SNAT, which is using a single IP address. This will show up in the event logs on the BIG-IP as Port Exhaustion errors.

To mitigate port exhaustion, create SNAT pools or use SNAT Automap (with an appropriate number of self-IP addresses on the VLAN) to support the expected level of concurrent connections. Configuring a SNAT pool as the translation allows the SNAT function to map client connections to more than one IP address from the SNAT pool, thus increasing the total available ports likewise the supported client connections.

You can build a SNAT pool for a SNAT to use as the translation addresses and the BIG-IP will use an IP addresses from the pool in a Least Connections fashion.

Since the SNAT function is intelligent enough to know what address from the pool can be used for the address translation in each egress scenario; a SNAT pool can contain addresses from more than one egress network. This will allow you to build less SNAT pools by allowing you to mix the egress network addresses in one pool if you desire.

Objective - 1.02 - Given a scenario, determine the minimum profiles for an application

1.02 - Given a scenario, determine the minimum profiles for an application

Configuration Guide for Local Traffic Management: Understanding Profiles

Scenario Based Questions

To prepare for scenario based questions the candidate will need to complete hands-on configuration and testing of the configuration on the LTM. This will allow the candidate to better understand how different configurations can produce different results. All F5 exams use scenario-based questions that make the candidate apply what they know to a situation to determine the resulting outcome.

This topic is focused on assigning profiles to a virtual server configuration for the functionality of application using that virtual server. Understanding how why profiles are necessary and what requirements the applications have for the processing of the application traffic is the key to this topic. Experience with configuring virtual servers will give the candidate the ability to answer the questions on this topic.

The BIG-IP LTM can manage application-specific network traffic in a variety of ways, depending on the protocols and services being used. For each type of traffic that you want or need to manage, the LTM system contains configuration tools that you can use to intelligently control the behavior of that traffic. These tools are called profiles. A profile is a system-supplied configuration tool that enhances your capabilities for managing application-specific traffic. More specifically, a profile is an object that contains user-configurable settings, with default values, for controlling the behavior of a particular type of network traffic, such as HTTP connections. Using profiles enhances your control over managing network traffic, and makes traffic-management tasks easier and more efficient.

A virtual server can be set with a minimum of a layer for protocol profile and traffic will pass to the pool resource. Without profiles set to tell the virtual server how to process that type of traffic it is possible that some necessary functions will not be able to be completed.

1.02 - Explain security options available for the application

Virtual Server Security

A virtual server is essentially a listener that will be taking in and processing traffic on the BIG-IP platform. Some of the biggest security risks when configuring a virtual server are how it is listening, where it is listening and who can get to it. If you are configuring virtual server and not setting the necessary settings to restrict these areas of concern you are opening your self up to security risks.

How Is The Virtual Server Listening?

The broader you set a virtual server to listen the greater the risk of unintended inbound traffic. An application based virtual server should typically be configured to listen on the default port for the application. For example if you are configuring a virtual server for a new HTTP based website you would listen on port 80. If you listen on all ports (*), the virtual server will take in traffic destine for the virtual server on all 65535 ports of the IP address. And if the pool members for the virtual server are also listening on all ports (*), it will send traffic to the servers on the port it arrived on the virtual server.

If you need to listen on multiple ports for the same IP address you can approach this in two different ways. You can build a virtual server for each necessary port using the same IP address or you can build one virtual server on all ports and use an iRule to restrict the allowed inbound connections to your list of ports.

Where is the Virtual Server Listening?

When you configure a virtual server you tell the BIG-IP where you want it to listen for traffic destined for the IP address of the virtual server. This virtual server setting is the VLAN and Tunnel Traffic setting. By default the setting is set to All VLANs and Tunnels. Which means the BIG-IP will listen on all VLANs. You are probably thinking, ARP is only going to happen on the local subnet's VLAN, which is true. So what can it possibly mean to listen on all VLANs? When this setting is set to all VLANs it means that if traffic comes to BIG-IP destined for the virtual server address from a VLAN that is not the VLAN of the virtual server IP address, it will still take the traffic in on VLAN interface that it arrived on. BIG-IP is a default deny device but in setting the setting to All VLANS and Tunnels you have told the system to listen on all VLANs for traffic to the virtual server and allow it in.

Introduction to Packet Filtering

Packet Filters

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

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

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

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

You can also configure global packet filtering that applies to all packet filter rules that you create. The following sections describe how to use the Configuration utility to set global packet filtering options, as well as create and manage individual packet filters rules.

Introduction to iRules

iRules

You can use iRules to restrict traffic in almost anyway you can think of. You can set an iRule to keep connections from happening when coming from a certain IP address range or to a certain URI path in the HTTP request.

1.02 - Explain how to use LTM as a service proxy

Since the F5 BIG-IP platform is designed as a full-proxy architecture the LTM can act as a proxy for any service level connection.

You define the virtual server as a Standard virtual server that is listening on an IP address and port combination, which represents the application to the client. The virtual server should be configured with an appropriate layer-4 profile, any optional layer-7 protocol profiles you need and a pool for a resource. The LTM will then broker separate layer-4 connections for the client and server sides. The server side connections will be translated from the listening IP address and port combination of the virtual server to the IP address and port combination of the pool member that the connection will be sent to via the load-balancing algorithm of the pool.

The return traffic must flow through the BIG-IP to be correctly rewritten as it passes back to the client. The return traffic will be rewritten from the IP address and port combination of the pool member that received the inbound connection to the IP address and port combination of the virtual server that the client connected to when the connection was established.

Standard Virtual Server

Standard virtual server

The BIG-IP LTM TMOS operating system implements a full proxy architecture for virtual servers configured with a TCP profile. By assigning a custom TCP profile to the virtual server, you can configure the BIG-IP LTM system to maintain compatibility to disparate server operating systems in the data center. At the same time, the BIG-IP LTM system can leverage its TCP/IP stack on the client side of the connection to provide independent and optimized TCP connections to client systems.

In a full proxy architecture, the BIG-IP LTM system appears as a TCP peer to both the client and the server by associating two independent TCP connections with the end-to-end session. Although certain client information, such as the source IP address or source TCP port, may be re-used on the server side of the connection, the BIG-IP LTM system manages the two sessions independently, making itself transparent to the client and server.

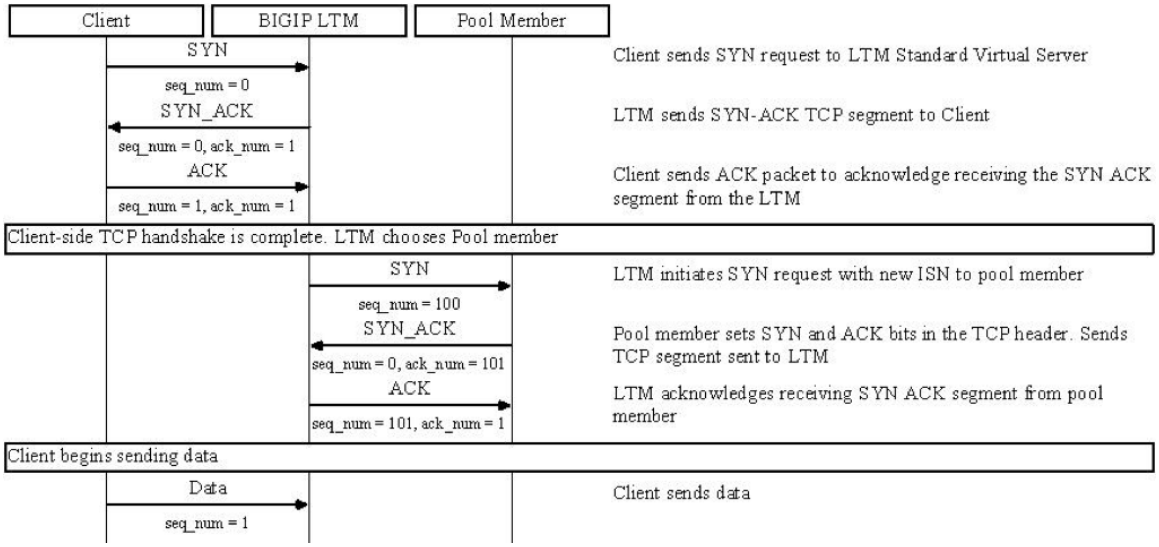
The Standard virtual server requires a TCP or UDP profile, and may optionally be configured with HTTP, FTP, or SSL profiles if Layer 7 or SSL processing is required.

The TCP connection setup behavior for a Standard virtual server varies depending on whether a TCP profile or a TCP and Layer 7 profile, such as HTTP, is associated with the virtual server.

Standard virtual server with a TCP profile

The TCP connection setup behavior for a Standard virtual server operates as follows: the three-way TCP handshake occurs on the client side of the connection before the BIG-IP LTM system initiates the TCP handshake on the server side of the connection.

A Standard virtual server processes connections using the full proxy architecture. The following TCP flow diagram illustrates the TCP handshake for a Standard virtual server with a TCP profile:



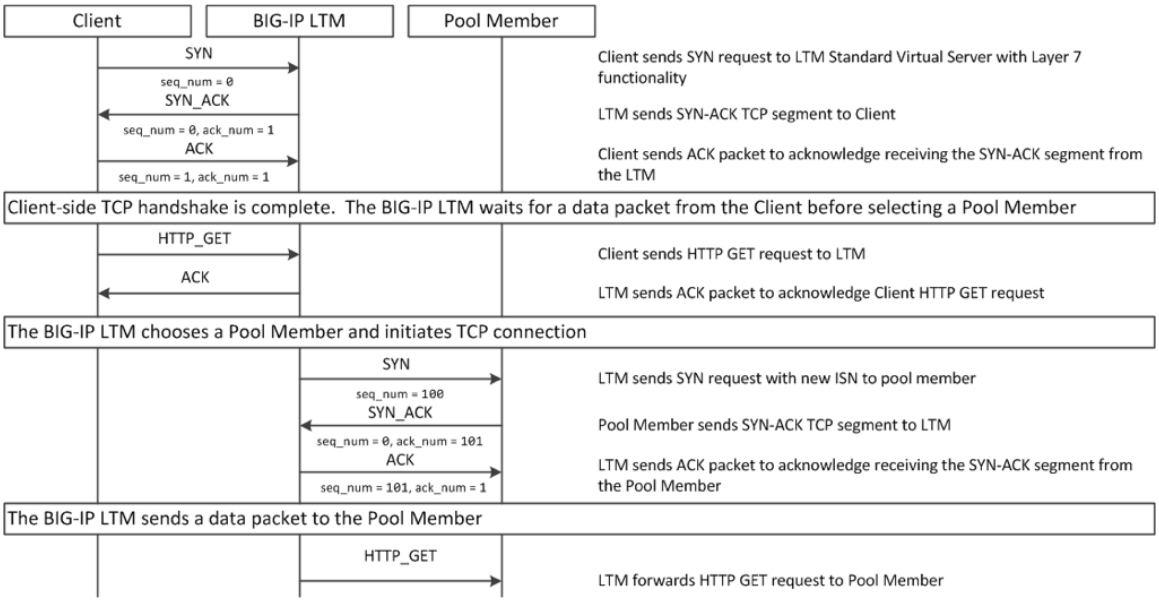
Standard virtual server with Layer 7 functionality

If a Standard virtual server is configured with Layer 7 functionality, such as an HTTP profile, the client must send at least one data packet before the server-side connection can be initiated by the BIG-IP LTM system.

Note: The BIG-IP LTM system may initiate the server-side connection prior to the first data packet for certain Layer 7 applications, such as FTP, in which case the user waits for a greeting banner before sending any data.

The TCP connection setup behavior for a Standard virtual server with Layer 7 functionality operates as follows: the three-way TCP handshake and initial data packet are processed on the client side of the connection before the BIG-IP LTM system initiates the TCP handshake on the server side of the connection.

A Standard virtual server with Layer 7 functionality processes connections using the full proxy architecture. The following TCP flow diagram illustrates the TCP handshake for a Standard virtual server with Layer 7 functionality:



1.02 - Describe how a given service is deployed on an LTM

Choosing Appropriate Profiles for HTTP Traffic

Processing HTTP traffic

The BIG-IP system allows you to process HTTP traffic using various profiles, including TCP+HTTP, FastHTTP, and FastL4. Each profile, or combination of profiles, offers distinct advantages, limitations, and features.

F5 recommends that you assess the needs of each HTTP virtual server individually, using the following information, to determine which profile, or profile combination, best meets the requirements for each virtual server.

> **Important:** The HTTP profile will work in all cases; however, the HTTP profile places BIG-IP in full Layer 7 inspection mode, which may be unnecessary when used on simple load balancing virtual servers. Thus, you should consider the other profile options provided in instances where the full Layer 7 engine is not necessary for a particular virtual server.

TCP+HTTP

Profiles: TCP+HTTP

Advantage: The HTTP profile can take full advantage of all of BIG-IP system's Layers 4 - 7 HTTP/HTTPS features.

When to use: The HTTP profile is used when any of the following features are required:

- IPv6 support
- TCPexpress and content spooling features reduce server load
- Full OneConnect functionality (including HTTP 1.0 transformations)
- Layer 7 persistence (cookie, hash, universal, and iRule)
- Full HTTP iRules logic
- Cache and Web Acceleration features
- HTTP Compression
- HTTP pipelining
- Virtual Server Authentication
- Redirect Rewriting
- SPDY protocol support (11.3.0 and later)

Limitations

- More CPU-intensive
- Memory utilization:
    - Cache / Web Acceleration

      The caching / web acceleration features provision user-defined memory for cache content for each virtual server that uses the given HTTP and Cache profiles.

    - Compression

      Larger buffer sizes can increase memory utilization when compressing large objects.

- TCP offloading/content spooling

  This can increase memory utilization in cases where either the client-side or the server-side of the connection is slower than the other. The BIG-IP system holds the data in the buffer until the slower side of the connection is able to retrieve it.

FastHTTP

Profile: FastHTTP

Advantage: Faster than HTTP profile

When to use: FastHTTP profile is recommended when it is not necessary to use persistence and or maintain source IP addresses. FastHTTP also adds a subset of OneConnect features to reduce the number of connections opened to the backend HTTP servers. The FastHTTP profile requires that the clients' source addresses are translated. If an explicit SNAT or SNAT pool is not specified, the appropriate self IP address is used.

> **Note:** Typically, server efficiency increases as the number of SNAT addresses that are available to the virtual server increases. At the same time, the increase in SNAT addresses that are available to the virtual server also decreases the likelihood that the virtual server will reach the point of ephemeral port exhaustion (65535 open connections per SNAT address).

Limitations

- Requires client source address translation
- Not compatible with persistence until version 10.0.0
- Limited iRules support L4 and are limited to a subset of HTTP header operations, and pool/pool member selection
- No compression
- No virtual server authentication
- No support for HTTP pipelining
- No TCP optimizations
- No IPv6 support

Note: FastHTTP is optimized for ideal traffic conditions, but may not be an appropriate profile to use when network conditions are less than optimal. For more information about the FastHTTP profile, refer to SOL8024: Overview of the FastHTTP profile.

FastL4

Profile: FastL4

Advantage: Accelerates packet processing

When to use: FastL4 is limited in functionality to socket level decisions (for example, src_ip:port dst_ip:port). Thus, you can use FastL4 only when socket level information for each connection is required for the virtual server.

Limitations

- No HTTP optimizations
- No TCP optimizations for server offloading
- SNAT/SNAT pools demote PVA acceleration setting level to Assisted
- iRules limited to L4 events, such as CLIENT_ACCEPTED and SERVER_CONNECTED
- No OneConnect
- Limited persistence options:
    - Source address
    - Destination address
    - Universal
    - Hash (BIG-IP 9.x only)
- No compression
- No Virtual Server Authentication
- No support for HTTP pipelining

Objective - 1.03 - Given an application configuration, determine which functions can be offloaded to the LTM device

1.03 - Explain how to offload HTTP servers for SSL, compression and caching

Offloading

One of the most prominent advantages to having a BIG-IP platform in your network is that it can offload functions from the server environment to improve their performance. SSL termination, HTTP compression and RAM Caching are a few of the primary functions

Each of these optimizations are configurations that are completed in profiles assigned to the virtual server.

SSL Profiles

SSL Offload

The primary way to control SSL network traffic on the BIG-IP platform is by configuring a Client or Server SSL profile:

- A Client profile is a type of traffic profile that enables Local Traffic Manager to accept and terminate any client requests that are sent by way of a fully SSL-encapsulated protocol. Local Traffic Manager supports SSL for both TCP and UDP protocols.
- A Server profile is a type of profile that enables Local Traffic Manager to initiate secure connections to a target web server.

To offloading of the overhead of processing SSL traffic from the server to the BIG-IP platform you will need to follow these high level steps:

1. Install a key/certificate pair on the BIG-IP system for terminating client-side secure connections.
2. Configure a client-side SSL profile using the new key/certificate pair.
3. Configure a virtual server to process the SSL traffic that uses the client-side SSL profile and a pool of the servers defined on HTTP. This virtual server will listen for HTTPS based traffic, terminate the SSL traffic and send the traffic to a pool resource that is listening for HTTP based traffic.

Compressing HTTP Responses

HTTP compression

An optional feature of the BIG-IP system is the system's ability to off-load HTTP compression tasks from the target server. All of the tasks that you need to configure HTTP compression, as well as the compression software itself, are centralized on the BIG-IP system. The primary way to enable HTTP compression is by configuring an HTTP Compression type of profile and then assigning the profile to a virtual server. This causes the system to compress HTTP content for any responses matching the values that you specify in the Request-URI or Content-Type settings of the HTTP Compression profile.

When you configure an HTTP Compression profile and assign it to a virtual server, the BIG-IP system reads the Accept-Encoding header of a client request and determines what content encoding method the client prefers. The BIG-IP system then removes the Accept-Encoding header from the request and passes the request to the server. Upon receiving the server response, the BIG-IP system inserts the Content-Encoding header, specifying either the gzip or deflate based on the compression method that the client specifies in the Accept-Encoding header.

Configuration

You should be familiar with how the configuration of HTTP Compression looks in the CLI Config as well as in the GUI.

To configure HTTP data compression, you need to create an HTTP compression type of profile, as well as a virtual server.

Creating a customized HTTP compression profile

If you need to adjust the compression settings to optimize compression for your environment, you can modify a custom HTTP compression profile.

1. On the Main tab, click Local Traffic > Profiles > Services > HTTP Compression. The HTTP Compression profile list screen opens.
2. Click Create. The New HTTP Compression Profile screen opens.
3. In the Name field, type a name for the profile.
4. From the Parent Profile list, select one of the following profiles:
    - httpcompression.
    - wan-optimized-compression.

5. Select the Custom check box. The fields in the Settings area become available for revision.
6. Modify the settings, as required.
7. Click Finished.

The modified HTTP compression profile is available in the HTTP Compression list screen.

Creating a virtual server for HTTP compression

You can create a virtual server that uses an HTTP profile with an HTTP compression profile to compress HTTP responses.

1. On the Main tab, click Local Traffic > Virtual Servers. The Virtual Server List screen displays a list of existing virtual servers.
2. Click the Create button. The New Virtual Server screen opens.
3. In the Name field, type a unique name for the virtual server.
4. Specify the Destination setting, using the Address field; type the IP address you want to use for the virtual server. The IP address you type must be available and not in the loopback network.
5. In the Service Port field, type 80, or select HTTP from the list.
6. Select http in the HTTP Profile list.
7. From the HTTP Compression Profile list, select one of the following profiles:
   - httpcompression
   - wan-optimized-compression
   - A customized profile
8. In the Resources area of the screen, from the Default Pool list, select a pool name.
9. Click Finished.

The virtual server with an HTTP profile configured with an HTTP compression profile appears in the Virtual Server list.

After you have created a custom HTTP Compression profile and a virtual server, you can test the configuration by attempting to pass HTTP traffic through the virtual server. Check to see that the BIG-IP system includes and excludes the responses that you specified in the custom profile, and that the system compresses the data as specified.

Profiles for Managing HTTP Traffic

Cacheing

To configure cacheing, you need to configure a Web Acceleration type of profile. These settings provide the ability to turn on the cache and fine-tune it for a specific implementation. Using a Web Acceleration type of profile, the system can store HTTP objects stored in memory that are reused by subsequent connections to reduce the amount of load on the back-end servers.

The default items stored by the cache are HTTP GET responses. However, you can specify URIs in the URI list if you want to cache POST and GET methods for a particular URI.

There are three types of Web Acceleration profiles that you can configure:

- A basic Web Acceleration profile
- An optimized acceleration profile
- An optimized caching profile

When to use the cache feature

The cache feature provides the ability to reduce the traffic load to back-end servers. This ability is useful if an object on a site is under high demand, if the site has a large quantity of static content, or if the objects on the site are compressed.

- **High-demand objects**

  This feature is useful if a site has periods of high demand for specific content. With the cache configured, the content server only has to serve the content to the BIG-IP system once per expiration period.

- **Static content**

  This feature is also useful if a site consists of a large quantity of static content such as CSS files, JavaScript files, or images and logos.

- **Content compression**

  For compressible data, the cache can store data for clients that can accept compressed data. When used in conjunction with the compression feature on the BIG-IP system, the cache takes stress off of the BIG-IP system and the content servers.

Items you can cache

The cache feature is fully compliant with the cache specifications described in RFC 2616, Hypertext Transfer Protocol – HTTP/1.1. This means you can configure the cache feature to cache the following content types:

- 200, 203, 206, 300, 301, and 410 responses
- Responses to GET methods, by default
- Other HTTP methods for URIs specified for inclusion in cached content, or specified in an iRule
- Content based on the User-Agent and Accept-Encoding values. The cache holds different content for Vary headers.

The items that the cache does not cache are:

- Private data specified by cache control headers
- HEAD, PUT, DELETE, TRACE, and CONNECT methods, by default

The caching mechanism

The default cache configuration caches only responses to HTTP GET methods. However, you can use the cache to cache other methods, too, including non-HTTP methods. You do this by specifying a URI in the URI Include or Pin list within a Web Acceleration profile, or by writing an iRule.

Objective - 1.04 - Given an iRule functionality, determine the profiles and configuration options necessary to implement the iRule.

1.04 - Explain how to create an HTTP configuration to handle an HTTP server error

HTTP Response
*links on DevCentral require member login

How to handle an HTTP server error

Configuring a virtual server on your BIG-IP platform to load balance the HTTP based traffic for your webservers can be a very simple configuration. But you realize that periodically a server returns an error and the clients are receiving a 404 error, and they are leaving your site for a competitor's site. You want to take an action on those errors to send your customers to a "Sorry Page".

If this were an issue of all of your servers be off line you could simply apply a custom HTTP profile to the virtual server and set the Fallback Host field with the URL to your Sorry Page. However this is happening intermittently on random server within the pool.

You could apply an iRule to your virtual server to send your customer to your Sorry Page when it sees the 404 error.

To do this, follow these steps:

1. Setup your Sorry Server to run the Sorry Page.
2. Write the iRule to meet your needs. The following is an example:

```
when HTTP_RESPONSE {
 if { [HTTP::status] contains “404”} {
 HTTP::redirect “http://www.mysorryserver.com/appsorrypage.html”
 }
}
```

3. Apply an HTTP profile (the default http profile will work) to the virtual server so that the virtual server will process the HTTP traffic allowing the iRule to work correctly.
4. Apply the new iRule to your virtual server.

You could do further rule work to track info about the server when the errors happen but it is not necessary to solve the problem.

Objective - 1.05 - Given an application configuration, determine the appropriate profile and persistence options

1.05 - Explain how to create an HTTP configuration for mobile clients

Protocol Profiles

Mobile Optimization

The BIG-IP system includes several pre-configured TCP profiles that you can use as is. In addition to the default TCP profile, the system includes TCP profiles that are pre-configured to optimize LAN and WAN traffic, as well as traffic for mobile users. You can use the pre-configured profiles as is, or you can create a custom profile based on a pre-configured profile and then adjust the values of the settings in the profiles to best suit your particular network environment.

The tcp-cell-optimized profile is a pre-configured profile type, for which the default values are set to give better performance to service providers' 3G and 4G customers. Specific options in the pre-configured profile are set to optimize traffic for most mobile users, and you can tune these settings to fit your network. For files that are smaller than 1 MB, this profile is generally better than the mptcp-mobile-optimized profile. For a more conservative profile, you can start with the tcp-mobile-optimized profile, and adjust from there.

> **Note:** Although the pre-configured settings produced the best results in the test lab, network conditions are extremely variable. For the best results, start with the default settings and then experiment to find out what works best in your network.

This list provides guidance for relevant settings

- Set the Proxy Buffer Low to the Proxy Buffer High value minus 64 KB. If the Proxy Buffer High is set to less than 64K, set this value at 32K.
- The size of the Send Buffer ranges from 64K to 350K, depending on network characteristics. If you enable the Rate Pace setting, the send buffer can handle over 128K, because rate pacing eliminates some of the burstiness that would otherwise exist. On a network with higher packet loss, smaller buffer sizes perform better than larger. The number of loss recoveries indicates whether this setting should be tuned higher or lower. Higher loss recoveries reduce the goodput.
- Setting the Keep Alive Interval depends on your fast dormancy goals. The default setting of 1800 seconds allows the phone to enter low power mode while keeping the flow alive on intermediary devices. To prevent the device from entering an idle state, lower this value to under 30 seconds.

- The Congestion Control setting includes delay-based and hybrid algorithms, which might better address TCP performance issues better than fully loss-based congestion control algorithms in mobile environments. The Illinois algorithm is more aggressive, and can perform better in some situations, particularly when object sizes are small. When objects are greater than 1 MB, goodput might decrease with Illinois. In a high loss network, Illinois produces lower goodput and higher retransmissions. The Woodside algorithm relies on timestamps to determine transmission. If timestamps are not available in your network, avoid using Woodside.

- For 4G LTE networks, specify the Packet Loss Ignore Rate as 0. For 3G networks, specify 2500. When the Packet Loss Ignore Rate is specified as more than 0, the number of retransmitted bytes and receives SACKs might increase dramatically.

- For the Packet Loss Ignore Burst setting, specify within the range of 6-12, if the Packet Loss Ignore Rate is set to a value greater than 0. A higher Packet Loss Ignore Burst value increases the chance of unnecessary retransmissions.

- For the Initial Congestion Window Size setting, round trips can be reduced when you increase the initial congestion window from 0 to 10 or 16.

- Enabling the Rate Pace setting can result in improved goodput. It reduces loss recovery across all congestion algorithms, except Illinois. The aggressive nature of Illinois results in multiple loss recoveries, even with rate pacing enabled.

A tcp-mobile-optimized profile is similar to a TCP profile, except that the default values of certain settings vary, in order to optimize the system for mobile traffic.

You can use the tcp-mobile-optimized profile as is, or you can create another custom profile, specifying the tcp-mobile-optimized profile as the parent profile.

1.05 - Explain how to create an HTTP configuration to optimize WAN connectivity

Managing Protocol Profiles

Optimize WAN Connectivity

You can use the tcp-wan-optimized profile to increase performance for environments where a link has lower bandwidth and/or higher latency. You can also implement WAN based Compression for HTTP traffic using the http compression profile.

The tcp-wan-optimized profile is a TCP-type profile. This profile is effectively a custom profile that Local Traffic Manager has already created for you, derived from the default tcp profile. This profile is useful for environments where a link has lower bandwidth and/or higher latency when paired with a faster link.

In cases where the BIG-IP system is load balancing traffic over a WAN link, you can enhance the performance of your wide-area TCP traffic by using the tcp-wan-optimized profile.

If the traffic profile is strictly WAN-based, and a standard virtual server with a TCP profile is required, you can configure your virtual server to use a tcp-wan-optimized profile to enhance WAN-based traffic. For example, in many cases, the client connects to the BIG-IP virtual server over a WAN link, which is generally slower than the connection between the BIG-IP system and the pool member servers. By configuring your virtual server to use the tcp-wan-optimized profile, the BIG-IP system can accept the data more quickly, allowing resources on the pool member servers to remain available. Also, use of this profile can increase the amount of data that the BIG-IP system buffers while waiting for a remote client to accept that data. Finally, you can increase network throughput by reducing the number of short TCP segments that the BIG-IP system sends on the network.

A tcp-wan-optimized profile is similar to a TCP profile, except that the default values of certain settings vary, in order to optimize the system for WAN-based traffic.

You can use the tcp-wan-optimized profile as is, or you can create another custom profile, specifying the tcp-wan-optimized profile as the parent profile.

1.05 - Determine when connection mirroring is required

Overview of Connection and Persistence Mirroring (11.x)

Connection Mirroring

The Connection Mirroring feature allows you to configure a BIG-IP system to duplicate connection information to the standby unit of a redundant pair. This setting provides higher reliability, but might affect system performance.

The BIG-IP systems are not stateful by default. In a BIG-IP redundant pair failover scenario, the redundant unit of the pair does not know the active connection states. F5 BIG-IP gives the administrator the ability to enable connection mirroring on a virtual server by virtual server basis.

Not all applications have to have their connection state know by the standby unit. Mainly applications that have long-term connections will need to have their connections mirrored.

For example, where long-term connections, such as FTP and Telnet, are good candidates for mirroring, mirroring short-term connections, such as HTTP and UDP, is not recommended as this causes a decrease in system performance. In addition, mirroring HTTP and UDP connections is typically not necessary, as those protocols allow for failure of individual requests without loss of the entire session.

Objective - 1.06 - Explain the steps necessary to configure AVR

1.06 - Explain the steps necessary to configure the AVR

BIG-IP Analytics: Implementations

Application Visibility and Reporting

Analytics (also called Application Visibility and Reporting (AVR)) is a module on the BIG-IP system that you can use to analyze the performance of web applications. It provides detailed metrics such as transactions per second, server and client latency, request and response throughput, and sessions. You can view metrics for applications, virtual servers, pool members, URLs, specific countries, and additional detailed statistics about application traffic running through the BIG-IP system.

Transaction counters for response codes, user agents, HTTP methods, countries, and IP addresses provide statistical analysis of the traffic that is going through the system. You can capture traffic for examination and have the system send alerts so you can troubleshoot problems and immediately react to sudden changes.

The Analytics module also provides remote logging capabilities so that your company can consolidate statistics gathered from multiple BIG-IP appliances onto syslog servers or SIEM devices, such as Splunk.

AVR Profile

An Analytics profile is a set of definitions that determines the circumstances under which the system gathers, logs, notifies, and graphically displays information regarding traffic to an application. The Analytics module requires that you select an Analytics profile for each application you want to monitor. You associate the Analytics profile with one or more virtual servers used by the application, or with an iApps application service. Each virtual server can have only one Analytics profile associated with it.

In the Analytics profile, you customize:

- What statistics to collect
- Where to collect data (locally, remotely, or both)
- Whether to capture the traffic itself
- Whether to send notifications

The BIG-IP system includes a default Analytics profile called analytics. It is a minimal profile that internally logs application statistics for server latency, throughput, response codes, and methods. You can modify the default profile, or create custom Analytics profiles for each application if you want to track different data for each one.

Charts shown on the Statistics > Analytics screens display the application data saved for all Analytics profiles associated with iApps application services or virtual servers on the system. You can filter the information, for example, by application or URL. You can also drill down into the specifics on the charts, and use the options to further refine the information in the charts.

Setting Up AVR

You can collect application statistics for one or more virtual servers or for an iApps application service. If virtual servers are already configured, you can specify them when setting up statistics collection. If you want to collect statistics for an iApps application service, you should first set up statistics collection, creating an Analytics profile, and then create the application service.

You need to provision the AVR module before you can set up local application statistics collection. You must have Adobe® Flash® Player installed on the computer where you plan to view Analytics statistics.

1.06 - Explain how to create an AVR profile and options

BIG-IP Analytics: Implementations

AVR profile and options

Setting up local application statistics collection

You need to provision the AVR module before you can set up local application statistics collection. You must have Adobe® Flash® Player installed on the computer where you plan to view Analytics statistics.

You can configure the BIG-IP system to collect specific application statistics locally.

1. On the Main tab, click Local Traffic > Profiles > Analytics.

   Tip: If Analytics is not listed, this indicates that Application Visibility and Reporting (AVR) is not provisioned, or you do not have rights to create profiles.

   The Analytics screen opens and lists all Analytics profiles that are on the system, including a default profile called analytics.

2. Click Create.

   The New Analytics Profile screen opens. By default, the settings are initially the same as in the default analytics profile.

3. In the Profile Name field, type a name for the Analytics profile.

4. For the Statistics Logging Type setting, verify that Internal is selected. If it is not, select the check box on the right first to activate the setting, then select Internal.

   Selecting Internal causes the system to store statistics locally, and you can view the charts on the system by clicking Overview > Statistics > Analytics.

5. Review the read-only Transaction Sampling Ratio value, which shows the current global (analytics) status of sampling for the system.

   Learning from all transactions provides the most accurate statistical data but impacts performance. The system can perform traffic sampling; for example, sampling 1 of every 99 transactions; sampling is less precise but demands fewer resources. If you need to change the value, you can do it later by editing the default analytics profile.

   If using traffic sampling, the Traffic Capturing Logging Type setting and User Sessions metric option are not available.

6. In the Included Objects area, specify the virtual servers for which to capture application statistics:

   a. For the Virtual Servers setting, click Add.

      A popup lists the virtual servers that you can assign to the Analytics profile.

   b. From the Select Virtual Server popup list, select the virtual servers to include and click Done.

      Note: You need to have previously configured the virtual servers (with an HTTP profile) for them to appear in the list. Also, you can assign only one Analytics profile to a virtual server so the list shows only virtual servers that have not been assigned an Analytics profile.

   Special considerations apply if using Analytics on a BIG-IP system with both Application Security Manager and Access Policy Manager, where security settings (in Portal Access webtop or an iRule) redirect traffic from one virtual server to a second one. In this case, you need to attach the Analytics profile to the second virtual server to ensure that the charts show accurate statistics.

7. To the right of the Statistics Gathering Configuration area, select the Custom check box. The settings in the area become available for modification.

8. In the Statistics Gathering Configuration, for Collected Metrics, select the statistics you want the system to collect:

Option	Description
Server Latency	Tracks how long it takes to get data from the application server to the BIG-IP system (selected by default).
Page Load Time	Tracks how long it takes an application user to get a complete response from the application, including network latency and completed page processing. **Note:** End user response times and latencies can vary significantly based on geography and connection types.
Throughput	Saves information about HTTP request and response throughput (selected by default).
User Sessions	Stores the number of unique user sessions. For Timeout, type the number of minutes of user non-activity to allow before the system considers the session to be over. If using transaction sampling, this option is not available.

9. For Collected Entities, select the entities for which you want the system to collect statistics:

Option	Description
URLs	Collects the requested URLs.
Countries	Saves the name of the country where the request came from based on the client IP address.
Client IP Addresses	Saves the IP address where the request originated. The address saved also depends on whether the request has an XFF (X-forwarded-for) header and whether Trust XFF is selected.
Response Codes	Saves HTTP response codes that the server returned to requesters (selected by default).
User Agents	Saves information about browsers used when making the request.
Methods	Saves HTTP methods in requests (selected by default).

10. Click Finished.

11. If you need to adjust the Transaction Sampling Ratio value, click the default analytics profile on the Profiles: Analytics screen.

You can use the sampling ratio to fine-tune the tradeoff between more accurate data and a possible performance impact. The value set here applies to all Analytics profiles on the system.

- Select all to collect all of the traffic that is being monitored and produce the most accurate results; it also poses the risk of performance reduction.
- Select 1 of every n to sample every nth transaction; not all possible traffic is processed producing more generalized results, but performance is better.

Generally, it is best to use all when the BIG-IP system has low TPS, and use 1 of every n when it has high TPS (for example, select 1 of every 20 to sample every twentieth request).

If you enable sampling (by selecting a setting other than all), the User Sessions metric and **Traffic Capturing Logging Type** settings become unavailable.

The BIG-IP system collects statistics about the application traffic described by the Analytics profile. You can view the statistics by clicking Statistics > Analytics.

If you want to monitor statistics for an iApps application, create the iApp application service, enable Analytics on the template, and specify the Analytics profile you just created. The BIG-IP system then collects statistics for the application service, and the application name appears in the Analytics charts.

Objective - 1.07 - Given a set of reporting requirements, determine the AVR metrics and entities to collect

1.07 - Given a set of reporting requirements, determine the AVR metrics and entities to collect

AVR Metrics and Entities to Collect

As you are working with AVR in your vLab and looking at results of the metrics that you gather, you should be paying attention to what AVR allows you to collect like Server Latency, Page Load Time, Throughput and User Sessions. You should also know what each of these mean (defined in the last section). You should also be aware of what you can gather that information for, such as URLs, Countries, Client IP Addresses, Response Codes, User Agents and Methods. You should also know what each of those mean (defined in the last section).

1.07 - Explain the sizing implications of AVR on the LTM device

BIG-IP Analytics 11.2.0

AVR Sizing

Provisioning AVR can be as impactful as provisioning any other licensed module. AVR requires CPU and Memory resources to function. As you increase the use of AVR within the BIG-IP device it can continue to further impact system resources. If you intend to use AVR on your BIG-IP environment you should consider the resource impact when you are doing platform sizing, as if it were any other heavy impact licensable software for the system.

1.07 - Explain the logging and notifications options of AVR

Setting Up Application Statistics Collection

AVR

You can examine the statistics in the Analytics charts when Application Visibility and Reporting (AVR) is provisioned. Analytics charts display statistical information about traffic on your system, including the following details:

- Overview
- Transactions
- Latency
- Throughput
- Sessions

The system updates the Analytics statistics every five minutes (you can refresh the charts periodically to see the updates). The Analytics Overview provides a summary of the most frequent recent types of application traffic, such as the top virtual servers, top URLS, top pool members, and so on. You can customize the Analytics Overview so that it shows the specific type of data you are interested in. You can also export the reports to a PDF or CSV file, or send the reports to one or more email addresses.

> **Note:** The displayed Analytics statistics are rounded up to two digits, and might be slightly inaccurate.

Before you can look at the application statistics, you need to have created an Analytics profile so that the system is capturing the application statistics internally on the BIG-IP system. You must associate the Analytics profile with one or more virtual servers (in the Analytics profile or in the virtual server). If you created an iApp

application service, you can use the provided template to associate the virtual server. To view Analytics statistics properly, you must have Adobe Flash Player installed on the computer where you plan to view them.

1.07 - Explain the uses of the collected metrics and entities

Setting Up Application Statistics Collection

Uses of AVR

You can review charts that show statistical information about traffic to your web applications. The charts provide visibility into application behavior, user experience, transactions, and data center resource usage.

Collected Metrics

Option	Description
Server Latency	Tracks how long it takes to get data from the application server to the BIG-IP system (selected by default).
Page Load Time	Tracks how long it takes an application user to get a complete response from the application, including network latency and completed page processing. End user response times and latencies can vary significantly based on geography and connection types.
Throughput	Saves information about HTTP request and response throughput (selected by default).
User Sessions	Stores the number of unique user sessions. For Timeout, type the number of minutes of user non-activity to allow before the system considers the session to be over. If using transaction sampling, this option is not available.

Collected Entities

Option	Description
URLs	Collects the requested URLs.
Countries	Saves the name of the country where the request came from based on the client IP address.
Client IP Addresses	Saves the IP address where the request originated. The address saved also depends on whether the request has an XFF (X-forwarded-for) header and whether Trust XFF is selected.
Response Codes	Saves HTTP response codes that the server returned to requesters (selected by default).
User Agents	Saves information about browsers used when making the request.
Methods	Saves HTTP methods in requests (selected by default).

Objective - 1.08 - Given a scenario, determine the appropriate monitor type and parameters to use

1.08 - Explain how to create an application specific monitor

Implementing Health and Performance Monitoring

Application Specific Monitor

You can set up the BIG-IP system to monitor the health or performance of certain nodes or servers that are members of a load balancing pool. Monitors verify connections on pool members and nodes. A monitor can be either a health monitor or a performance monitor, designed to check the status of a pool, pool member, or node on an ongoing basis, at a set interval. If a pool member or node being checked does not respond within a specified timeout period, or the status of a pool member or node indicates that performance is degraded, the BIG-IP system can redirect the traffic to another pool member or node.

Some monitors are included as part of the BIG-IP system, while other monitors are user-created. Monitors that the BIG-IP system provides are called pre-configured monitors. User-created monitors are called custom monitors.

Before configuring and using monitors, it is helpful to understand some basic concepts regarding monitor types, monitor settings, and monitor implementation.

Monitor types

Every monitor, whether pre-configured or custom, is a certain type of monitor. Each type of monitor checks the status of a particular protocol, service, or application. For example, one type of monitor is HTTP. An HTTP type of monitor allows you to monitor the availability of the HTTP service on a pool, pool member, or node. A WMI type of monitor allows you to monitor the performance of a pool, pool member, or node that is running the Windows Management Instrumentation (WMI) software. An ICMP type of monitor simply determines whether the status of a node is up or down.

BIG-IP Local Traffic Manager: Monitors Reference

About application check monitors

An application check monitor interacts with servers by sending multiple commands and processing multiple responses.

An FTP monitor, for example, connects to a server, logs in by using a user ID and password, navigates to a specific directory, and then downloads a specific file to the /var/tmp directory. If the file is retrieved, the check is successful.

1. Local Traffic Manager opens a TCP connection to an IP address and port, and logs in to the server.
2. A specified directory is located and a specific file is requested.
3. The server sends the file to Local Traffic Manager.
4. Local Traffic Manager receives the file and closes the TCP connection.

About content check monitors

A content check monitor determines whether a service is available and whether the server is serving the appropriate content. This type of monitor opens a connection to an IP address and port, and then issues a command to the server. The response is compared to the monitor's receive rule. When a portion of the server's response matches the receive rule, the test is successful.

1. Local Traffic Manager opens a TCP connection to an IP address and port, and issues a command to the server.
2. The server sends a response.
3. Local Traffic Manager compares the response to the monitor's receive rule and closes the connection

Creating a custom HTTP monitor

Before creating a monitor, you must decide on a monitor type.

A custom HTTP monitor enables you to send a command to a server and examine that server's response, thus ensuring that it is serving appropriate content.

Note: An HTTP monitor can monitor Outlook® Web Access (OWA) in Microsoft® Exchange Server 2007 and Microsoft® SharePoint® 2007 web sites that require NT LAN Manager (NTLM) authentication. NTLM authentication requires a send string that complies with HTTP/1.1, a user name, and a password.

1. On the Main tab, click Local Traffic > Monitors. The Monitor List screen opens.
2. Type a name for the monitor in the Name field.
3. From the Type list, select HTTP.

   The screen refreshes, and displays the configuration options for the HTTP monitor type.
4. From the Import Settings list, select http.

   The new monitor inherits initial configuration values from the existing monitor.
5. In the Configuration area of the screen, select Advanced.

   This selection makes it possible for you to modify additional default settings.
6. Type a number in the Interval field that indicates, in seconds, how frequently the system issues the monitor check. The default is 5 seconds.
7. From the Up Interval list, do one of the following:
   - Accept the default, Disabled, if you do not want to use the up interval.
   - Select Enabled, and specify how often you want the system to verify the health of a resource that is up.
8. Type a number in the Time Until Up field that indicates the number of seconds to wait after a resource first responds correctly to the monitor before setting the resource to up.

   The default value is 0 (zero), which disables this option.
9. Type a number in the Timeout field that indicates, in seconds, how much time the target has to respond to the monitor check. The default is 30 seconds.

   If the target responds within the allotted time period, it is considered up. If the target does not respond within the time period, it is considered down.
10. Specify whether the system automatically enables the monitored resource, when the monitor check is successful, for Manual Resume.

This setting applies only when the monitored resource has failed to respond to a monitor check.

Option	Description
Yes	The system does nothing when the monitor check succeeds, and you must manually enable the monitored resource.
No	The system automatically re-enables the monitored resource after the next successful monitor check.

11. Type a text string in the Send String field that the monitor sends to the target resource. The default string is GET /\r\n. This string retrieves a default file from the web site.

    Important: Send string syntax depends upon the HTTP version. Please observe the following conventions.

Version	Convention
HTTP 0.9	"GET /\n" or "GET /\r\n".
HTTP 1.0	"GET / HTTP/1.0\r\n\r\n" or "GET /HTTP/1.0\n\n"
HTTP 1.1	"GET / HTTP/1.1\r\nHost: server.com\r\n\r\n" or "GET /HTTP/1.1\r\nHost: server.com\r\nConnection: close\r\n\r\n"

Type a fully qualified path name, for example, “GET /www/example/index.html\r\n”, if you want to retrieve a specific web site page.

12. Type a regular expression in the Receive String field that represents the text string that the monitor looks for in the returned resource.

    The most common receive expressions contain a text string that is included in an HTML file on your site. The text string can be regular text, HTML tags, or image names.

    Note: If you do not specify both a send string and a receive string, the monitor performs a simple service check and connect only.

13. Type a regular expression in the Receive Disable String field that represents the text string that the monitor looks for in the returned resource.

    Use a Receive String value together with a Receive Disable String value to match the value of a response from the origin web server and create one of three states for a pool member or node: Up (Enabled), when only Receive String matches the response; Up (Disabled), when only Receive Disable

String matches the response; or Down, when neither Receive String nor Receive Disable String matches the response.

Note: If you choose to set the Reverse setting to Yes, the Receive Disable String option becomes unavailable and the monitor marks the pool, pool member, or node Down when the test is successful.

14. Type a name in the User Name field.
15. Type a password in the Password field.
16. For the Reverse setting, do one of the following:
    - Accept the No default option.
    - Select the Yes option to make the Receive Disable String option unavailable and mark the pool, pool member, or node Down when the test is successful.
17. For the Transparent setting, do one of the following:
    - Accept the No default option.
    - Select the Yes option to use a path through the associated pool members or nodes to monitor the aliased destination.

The HTTP monitor is configured to monitor HTTP traffic.

1.08 - Given a desired outcome, determine where to apply health monitors

Configuration Guide for BIG-IP Local Traffic Management: 12 - Configuring Monitors

Applying Health Monitors

You must associate a monitor with the server or servers to be monitored. The server or servers can either be a pool, a pool member, or a node, depending on the monitor type.

Association types

You can associate a monitor with a server in any of these ways:

Monitor-to-pool association

This type of association associates a monitor with an entire load balancing pool. In this case, the monitor checks all members of the pool. For example, you can create an instance of the monitor http for every member of the pool my_pool, thus ensuring that all members of that pool are checked.

Monitor-to-pool member association

This type of association associates a monitor with an individual pool member, that is, an IP address and service. In this case, the monitor checks only that pool member and not any other members of the pool. For example, you can create an instance of the monitor http for pool member 10.10.10.10:80 of my_pool.

Monitor-to-node association

This type of association associates a monitor with a specific node. In this case, the monitor checks only the node itself, and not any services running on that node. For example, you can create an instance of the monitor ICMP for node 10.10.10.10. In this case, the monitor checks the specific node only, and not any services running on that node.

You can designate a monitor as the default monitor that you want Local Traffic Manager to associate with one or more nodes. In this case, any node to which you have not specifically assigned a monitor inherits the default monitor.

Some monitor types are designed for association with nodes only, and not pools or pool members. Other monitor types are intended for association with pools and pool members only, and not nodes.

Node-only monitors specify a destination address in the format of an IP address with no service port (for example, 10.10.10.2). Conversely, monitors that you can associate with nodes, pools, and pool members specify a destination address in the format of an IP address and service port (for example, 10.10.10.2:80). Therefore, when you use the Configuration utility to associate a monitor with a pool, pool member, or node, the utility displays only those pre-configured monitors that are designed for association with that server.

For example, you cannot associate the monitor ICMP with a pool or its members, since the ICMP monitor is designed to check the status of a node itself and not any service running on that node.

Monitor instances

When you associate a monitor with a server, Local Traffic Manager automatically creates an instance of that monitor for that server. A monitor association thus creates an instance of a monitor for each server that you specify. This means that you can have multiple instances of the same monitor running on your servers.

Because instances of monitors are not partitioned objects, a user can enable or disable an instance of a monitor without having permission to manage the associated pool or pool member.

For example, a user with the Manager role, who can access partition AppA only, can enable or disable monitor instances for a pool that resides in partition Common. However, that user cannot perform operations on the pool or pool members that are associated with the monitor. Although this is correct functionality, the user might not expect this behavior. You can prevent this unexpected behavior by ensuring that all pools and pool members associated with monitor instances reside in the same partition.

1.08 - Determine under which circumstances an external monitor is required

REF 1 p 19-6

External Monitor

An external monitor allows you to monitor services using your own programs. Your program tests services in any way you wish; the monitor need only know the name of the program. Once the BIG-IP system initiates the external program, it waits for any response set to standard out. If a response is seen the monitor is considered a success. If no response to scene prior to the timeout being reached the mantra has failed.

If the template health monitors that the BIG-IP platform executes directly will not work to monitor your application you can use an external monitor and call a script to check your application.

Objective - 1.09 - Given a set of parameters, predict an outcome of a monitor status on other LTM device objects

1.09 - Determine the effect of a monitor on the virtual server status

Monitors Concepts

Effect of Monitoring

Health monitoring with a BIG-IP allows you to monitor resources at many different levels. Monitors are assigned to resources in two areas of the configuration, at the node level and at the pool level. At the node level you can assign monitors to all nodes (Default Monitor) or to each node (Node Specific). At the pool level you can assign monitors to all pool members (Default Pool Monitor) or to each member (Member Specific).

If a monitor at the node level marks the node down, then pool member that uses the node IP address as its member IP address will automatically be marked down. This function works as a parent-child relationship between the node and the pool member. These monitors are typically network level monitors (ping, TCP half open)

When a pool member that is being monitored by a health monitor does not respond to a probe from the BIG-IP system within a specified timeout period, the system marks the pool member down and no longer load balances traffic to that pool member. If all of the pool members are marked off line and no pool members

are available to service the request then the pool is marked down and thus the virtual server is marked down. The status of a virtual server works as a parent-child relationship between the pool and the virtual server.

When the failing health monitor starts to succeed again and at least one pool member is able to respond, then pool will be marked available and thus the virtual server will also become available.

1.09 - Determine the effect of active versus inline monitors on the application status or on the LTM device

Monitors Concepts

Active Monitoring

Active monitoring checks the status of a pool member or node on an ongoing basis as specified. If a pool member or node does not respond within a specified timeout period, or the status of a node indicates that performance is degraded, the BIG-IP system can redirect the traffic to another pool member or node. There are many active monitors. Each active monitor checks the status of a particular protocol, service, or application. For example, one active monitor is HTTP. An HTTP monitor allows you to monitor the availability of the HTTP service on a pool, pool member, or node. A WMI monitor allows you to monitor the performance of a node that is running the Windows Management Instrumentation (WMI) software. Active monitors fall into two categories: Extended Content Verification (ECV) monitors for content checks, and Extended Application Verification (EAV) monitors for service checks, path checks, and application checks.

An active monitor can check for specific responses, and run with or without client traffic.

> **Note:** An active monitor also creates additional network traffic beyond the client request and server response and can be slow to mark a pool member as down.

Passive monitoring

Passive monitoring occurs as part of a client request. This kind of monitoring checks the health of a pool member based on a specified number of connection attempts or data request attempts that occur within a specified time period. If, after the specified number of attempts within the defined interval, the system cannot connect to the server or receive a response, or if the system receives a bad response, the system marks the pool member as down. There is only one passive monitor, called an Inband monitor.

A passive monitor creates no additional network traffic beyond the client request and server response. It can mark a pool member as down quickly, as long as there is some amount of network traffic.

> **Note:** A passive monitor cannot check for specific responses and can potentially be slow to mark a pool member as up.

Objective - 1.10 - Given a set of SSL requirements, determine the appropriate profile options to create or modify in the SSL profile

1.10 - Describe the difference between client and server SSL profiles

SSL Profiles

Differences between Client and Server SSL Profiles

With LTM, you can enable SSL traffic management for either client-side traffic or server-side traffic.

Client-side traffic refers to connections between a client system and the BIG-IP system. Server-side traffic refers to connections between the BIG-IP system and a target server system:

Client-side SSL traffic

When you enable the BIG-IP system to manage client-side SSL traffic, LTM terminates incoming SSL connections by decrypting the client request. LTM then sends the request, in clear text, to a target server. Next, LTM retrieves a clear-text response (such as a web page) and encrypts the request, before sending the web page back to the client. During the process of terminating an SSL connection, LTM can, as an option, perform all of the SSL certificate verification functions normally handled by the target web server.

Server-side SSL traffic

When you enable LTM to manage server-side SSL traffic, LTM enhances the security of your network by re-encrypting a decrypted request before sending it on to a target server. In addition to this re-encryption, LTM can, as an option, perform the same verification functions for server certificates that LTM can for client certificates.

1.10 - Describe the difference between client and server SSL processing

SSL Profiles

Differences

A Client profile is a type of traffic profile that enables Local Traffic Manager to accept and terminate any client requests that are sent by way of a fully SSL-encapsulated protocol. Local Traffic Manager supports SSL for both TCP and UDP protocols.

A Server profile is a type of profile that enables Local Traffic Manager to initiate secure connections to a target web server.

Objective - 1.11 - Given a set of application requirements, describe the steps necessary to configure SSL

1.11 - Describe the process to update expired SSL certificates

Renewing Certificate Authority Signed SSL Certificates Based on a New Private SSL Key (9.x - 10.x)

Update Expired SSL Certs

Some Certificate Authorities allow you to renew a certificate by requesting a new certificate using the Certificate Signing Request (CSR) on file, or by generating a new CSR. However, you may choose to generate a new private SSL key and then generate a new CSR from that new private SSL key.

> **Note:** To prevent any disruption to traffic or services, F5 recommends that you renew a certificate before the existing certificate expires.

Generating a new SSL key

To generate a new SSL key and prevent the existing SSL key and certificate from being overwritten, perform the following procedure:

1. Log in to the BIG-IP LTM Configuration utility.
2. Select Local Traffic.
3. Select SSL Certificates.
4. Select Create.
5. Select Certificate Authority from the Issuer drop-down menu.
6. Enter a name for the SSL key file, and append a number or date at the end of the name.

   Note: By naming the file in this format, you can have the same file name as the previous file and the system treats it as a separate file.

   For example, you can change www.mysite.local to use the syntax of one of the following SSL key name examples:

   www.mysite.local2

   www.mysite.localMMDDYYYY

7. Enter the Certificate properties information as they are listed in the previous SSL key.

   Note: To view the information that is listed in the existing SSL key, select the name and certificate from the SSL Certificates List.

8. The new SSL key information should appear similar to the following example:

   Name: www.mysite.local2

   Common Name: www.mysite.local

   Organization: F5 Networks

   Division: AskF5

   Locality: Seattle

   State Or Province: WA

   Country: US

9. Click Finished.
10. Send the generated CSR to the CA to obtain a new certificate.
11. Import the new certificate by selecting Import.
12. Select Certificate from the drop-down menu.
13. Enter a name for the certificate using the same name you gave to the SSL key created in step 6.

    Note: Using the same name for both the key and certificate does not cause any problems as the two files are saved with different extensions.

14. Select Upload File or Paste Text.
15. Click Finished.

Updating the SSL profile with the new SSL key and certificate

Note: Once the new key and certificate are installed on the BIG-IP LTM system, F5 recommends that you update the new key and certificate during a maintenance window to prevent any disruption to SSL traffic.

To update the SSL profile, perform the following procedure:

1. Log in to the BIG-IP LTM Configuration utility.
2. Select Local Traffic.

3. Select Profiles.
4. Select SSL.
5. Choose Client or Server.
6. Select the profile for which you are going to switch the SSL key and certificate.
7. Select the new SSL key and certificate from the appropriate drop-down menu.
8. Click Finished.

   Note: Existing connections will continue to use the old SSL certificate until the connections complete, are renegotiated, or TMM is restarted.

1.11 - Describe the steps to incorporate client authentication to the SSL process

SSL Profiles Part 8: Client Authentication

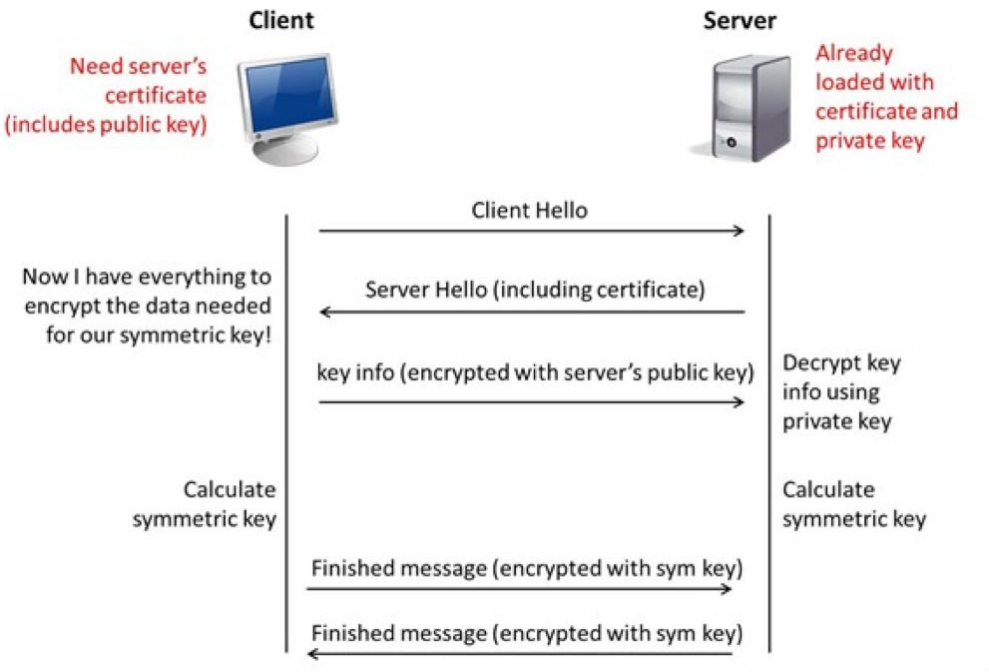
Client Authentication

In a TLS handshake, the client and the server exchange several messages that ultimately result in an encrypted channel for secure communication. During this handshake, the client authenticates the server's identity by verifying the server certificate (for more on the TLS handshake, see article 1 of this series). Although the client always authenticates the server's identity, the server is not required to authenticate the client's identity. However, there are some situations that call for the server to authenticate the client.

Client authentication is a feature that lets you authenticate users that are accessing a server. In client authentication, a certificate is passed from the client to the server and is verified by the server. Client authentication allow you to rest assured that the person represented by the certificate is the person you expect. Many companies want to ensure that only authorized users can gain access to the services and content they provide. As more personal and access-controlled information moves online, client authentication becomes more of a reality and a necessity.

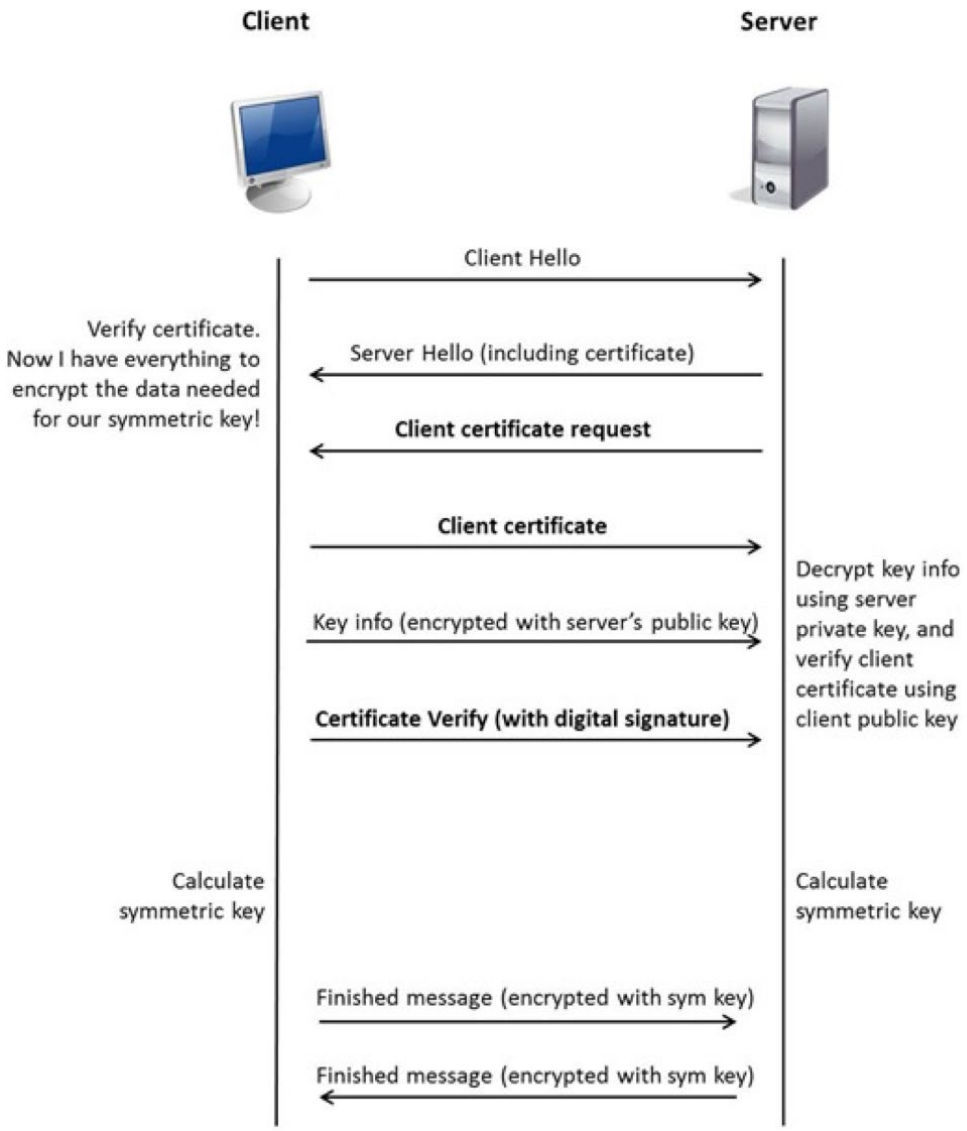
How Does Client Authentication Work?

Before we jump into client authentication, let's make sure we understand server authentication. During the TLS handshake, the client authenticates the identity of the server by verifying the server's certificate and using the server's public key to encrypt data that will be used to compute the shared symmetric key. The server can only generate the symmetric key used in the TLS session if it can decrypt that data with its private key. The following diagram shows an abbreviated version of the TLS handshake that highlights some of these concepts.



Ultimately, the client and server need to use a symmetric key to encrypt all communication during their TLS session. In order to calculate that key, the server shares its certificate with the client (the certificate includes the server's public key), and the client sends a random string of data to the server (encrypted with the server's public key). Now that the client and server each have the random string of data, they can each calculate (independently) the symmetric key that will be used to encrypt all remaining communication for the duration of that specific TLS session. In fact, the client and server both send a "Finished' message at the end of the handshake...and that message is encrypted with the symmetric key that they have both calculated on their own. So, if all that stuff works and they can both read each other's "Finished" message, then the server has been authenticated by the client and they proceed along with smiles on their collective faces (encrypted smiles, of course).

You'll notice in the diagram above that the server sent its certificate to the client, but the client never sent its certificate to the server. When client authentication is used, the server still sends its certificate to the client, but it also sends a "Certificate Request" message to the client. This lets the client know that it needs to get its certificate ready because the next message from the client to the server (during the handshake) will need to include the client certificate. The following diagram shows the added steps needed during the TLS handshake for client authentication.



So, you can see that when client authentication is enabled, the public and private keys are still used to encrypt and decrypt critical information that leads to the shared symmetric key. In addition to the public and private keys being used for authentication, the client and server both send certificates and each verifies the certificate of the other. This certificate verification is also part of the authentication process for both the client and the server. The certificate verification process includes four important checks. If any of these checks do not return a valid response, the certificate verification fails (which makes the TLS handshake fail) and the session will terminate.

These checks are as follows:

1. Check digital signature
2. Check certificate chain
3. Check expiration date and validity period
4. Check certificate revocation status

Here's how the client and server accomplish each of the checks for client authentication:

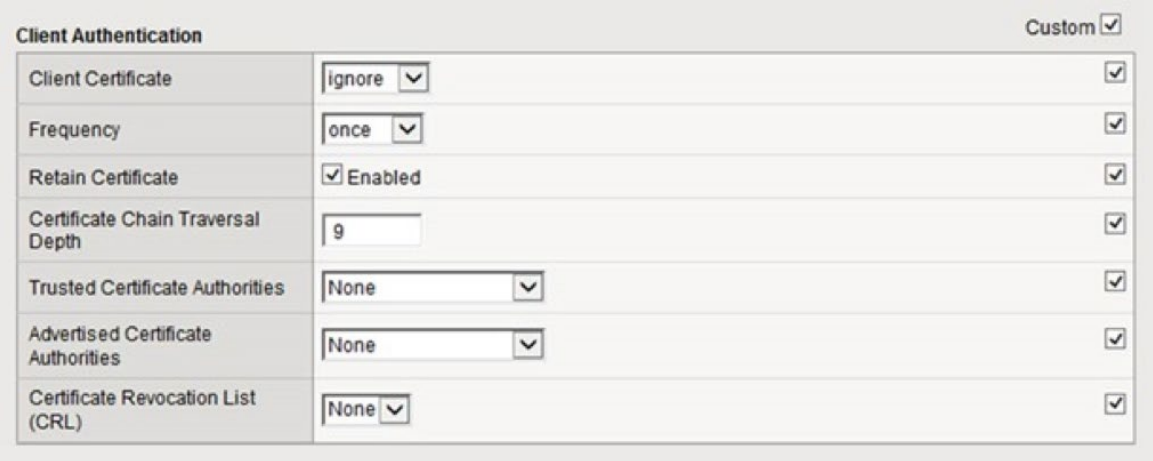
1. **Digital Signature:** The client sends a "Certificate Verify" message that contains a digitally signed copy of the previous handshake message. This message is signed using the client certificate's private key. The server can validate the message digest of the digital signature by using the client's public key (which is found in the client certificate). Once the digital signature is validated, the server knows that public key belonging to the client matches the private key used to create the signature.
2. **Certificate Chain:** The server maintains a list of trusted CAs, and this list determines which certificates the server will accept. The server will use the public key from the CA certificate (which it has in its list of trusted CAs) to validate the CA's digital signature on the certificate being presented. If the message digest has changed or if the public key doesn't correspond to the CA's private key used to sign the certificate, the verification fails and the handshake terminates.
3. **Expiration Date and Validity Period:** The server compares the current date to the validity period listed in the certificate. If the expiration date has not passed and the current date is within the period, everything is good. If it's not, then the verification fails and the handshake terminates.
4. **Certificate Revocation Status:** The server compares the client certificate to the list of revoked certificates on the system. If the client certificate is on the list, the verification fails and the handshake terminates.

As you can see, a bunch of stuff has to happen in just the right way for the Client-Authenticated TLS handshake to finalize correctly. If any piece is not setup correctly the communication flow will fail. Something as simple as not including a necessary Chain Certificate will cause the clients browser to pop a warning for trust issues, even if you are using the correct certificates. But, all this is in place for your own protection. After all, you want to make sure that no one else can steal your identity and impersonate you on a critically important website!

BIG-IP Configuration

Now that we've established the foundation for client authentication in a TLS handshake, let's figure out how the BIG-IP is set up to handle this feature. The following screenshot shows the user interface for configuring Client Authentication.

To get here, navigate to Local Traffic > Profiles > SSL > Client.



The Client Certificate drop down menu has three settings: Ignore (default), Require, and Request. The "Ignore" setting specifies that the system will ignore any certificate presented and will not authenticate the client before establishing the SSL session. This effectively turns off client authentication. The "Require" setting enforces client authentication. When this setting is enabled, the BIG-IP will request a client certificate and attempt to verify it. An SSL session is established only if a valid client certificate from a trusted CA is presented. Finally, the "Request" setting enables optional client authentication. When this setting is enabled, the BIG-IP will request a client certificate and attempt to verify it. However, an SSL session will be established regardless of whether or not a valid client certificate from a trusted CA is presented. The Request option is often used in conjunction with iRules in order to provide selective access depending on the certificate that is presented. For example: let's say you would like to allow clients who present a certificate from a trusted CA to gain access to the application while clients who do not provide the required certificate be redirected to a page detailing the access requirements. If you are not using iRules to enforce a different outcome based on the certificate details, there is no significant benefit to using the "Request" setting versus the default "Ignore" setting. In both cases, an SSL session will be established regardless of the certificate presented.

Frequency specifies the frequency of client authentication for an SSL session. This menu offers two options: Once (default) and Always. The "Once" setting specifies that the system will authenticate the client only once for an SSL session. The "Always" setting specifies that the system will authenticate the client once when the SSL session is established as well as each time that session is reused.

The Retain Certificate box is checked by default. When checked, the client certificate is retained for the SSL session.

Certificate Chain Traversal Depth specifies the maximum number of certificates that can be traversed in a client certificate chain. The default for this setting is 9. Remember that "Certificate Chain" part of the verification checks? This setting is where you configure the depth that you allow the server to dig for a trusted CA. For more on certificate chains, see article 2 of this SSL series.

Trusted Certificate Authorities setting is used to specify the BIG-IP's Trusted Certificate Authorities store. These are the CAs that the BIG-IP trusts when it verifies a client certificate that is presented during client authentication. The default value for the Trusted Certificate Authorities setting is None, indicating that no CAs are trusted. Don't forget...if the BIG-IP Client Certificate menu is set to Require but the Trusted Certificate Authorities is set to None, clients will not be able to establish SSL sessions with the virtual server. The drop down list in this setting includes the name of all the SSL certificates installed in the BIG-IP's /config/ssl/ssl.crt directory. A newly-installed BIG-IP system will include the following certificates: default certificate and ca-bundle certificate. The default certificate is a self-signed server certificate used when testing SSL profiles. This certificate is not appropriate for use as a Trusted Certificate Authorities certificate bundle. The ca-bundle certificate is a bundle of CA certificates from most of the well-known PKIs around the world. This certificate may be appropriate for use as a Trusted Certificate Authorities certificate bundle. However, if this bundle is specified as the Trusted Certificate Authorities certificate store, any valid client certificate that is signed by one of the popular Root CAs included in the default ca-bundle.crt will be authenticated. This provides some level of identification, but it provides very little access control since almost any valid client certificate could be authenticated.

If you want to trust only certificates signed by a specific CA or set of CAs, you should create and install a bundle containing the certificates of the CAs whose certificates you trust. The bundle must also include the entire chain of CA certificates necessary to establish a chain of trust. Once you create this new certificate bundle, you can select it in the Trusted Certificate Authorities drop down menu.

The Advertised Certificate Authorities setting is used to specify the CAs that the BIG-IP advertises as trusted when soliciting a client certificate for client authentication. The default value for the Advertised Certificate Authorities setting is None, indicating that no CAs are advertised. When set to None, no list of trusted CAs is sent to a client with the certificate request. If the Client Certificate menu is set to Require or Request, you can configure the Advertised Certificate Authorities setting to send clients a list of CAs that the server is likely to trust. Like the Trusted Certificate Authorities list, the Advertised Certificate Authorities drop down list includes the name of all the SSL certificates installed in the BIG-IP /config/ssl/ssl.crt directory. A newly-installed BIG-IP system includes the following certificates: default certificate and ca-bundle certificate. The default certificate is a self-signed server certificate used for testing SSL profiles. This certificate is not appropriate for use as an Advertised Certificate Authorities certificate bundle. The ca-bundle certificate is a bundle of CA certificates from most of the well-known PKIs around the world. This certificate may be appropriate for use as an Advertised Certificate Authorities certificate bundle.

If you want to advertise only a specific CA or set of CAs, you should create and install a bundle containing the certificates of the CA to advertise. Once you create this new certificate bundle, you can select it in the Advertised Certificate Authorities setting drop down menu.

You are allowed to configure the Advertised Certificate Authorities setting to send a different list of CAs than that specified for the Trusted Certificate Authorities. This allows greater control over the configuration information shared with unknown clients. You might not want to reveal the entire list of trusted CAs to a client that does not automatically present a valid client certificate from a trusted CA. Finally, you should avoid specifying a bundle that contains a large number of certificates when you configure the Advertised Certificate Authorities setting. This will cut down on the number of certificates exchanged during a client SSL handshake. The maximum size allowed by the BIG-IP for native SSL handshake messages is 14,304 bytes. Most handshakes don't result in large message lengths, but if the SSL handshake is negotiating a native cipher and the total length of all messages in the handshake exceeds the 14,304 byte threshold, the handshake will fail.

The Certificate Revocation List (CRL) setting allows you to specify a CRL that the BIG-IP will use to check revocation status of a certificate prior to authenticating a client. If you want to use a CRL, you must upload it to the /config/ssl/ssl.crl directory on the BIG-IP. The name of the CRL file may then be entered in the CRL setting dialog box. Note that this box will offer no drop down menu options until you upload a CRL file to the BIG-IP. Since CRLs can quickly become outdated, you should use either OCSP or CRLDP profiles for more robust and current verification functionality.

Objective - 1.12 - Given a set of application requirements, determine the appropriate virtual server type to use

1.12 - Given a set of application requirements, determine the appropriate virtual server type to use

Overview of BIG-IP Virtual Server Types (11.x)

You should expect to see many questions that involve knowing what type of virtual server should be used in different scenarios, as well as, if a type were used what the outcome would be. The following table lists the virtual server type options and defines each virtual server type:

Virtual server type	Description of virtual server type
Standard	A Standard virtual server directs client traffic to a load balancing pool and is the most basic type of virtual server. It is a general purpose virtual server that does everything not expressly provided by the other type of virtual servers.
Forwarding (Layer 2)	A Forwarding (Layer 2) virtual server typically shares the same IP address as a node in an associated VLAN. A Forwarding (Layer 2) virtual server is used in conjunction with a VLAN group.
Forwarding (IP)	A Forwarding (IP) virtual server forwards packets directly to the destination IP address specified in the client request. A Forwarding (IP) virtual server has no pool members to load balance.
Performance (Layer 4)	A Performance (Layer 4) virtual server has a FastL4 profile associated with it. A Performance (Layer 4) virtual server increases the speed at which the virtual server processes packets.
Performance (HTTP)	A Performance (HTTP) virtual server has a FastHTTP profile associated with it. The Performance (HTTP) virtual server and related profile increase the speed at which the virtual server processes HTTP requests.
Stateless	A Stateless virtual server improves the performance of UDP traffic in specific scenarios.
Reject	A Reject virtual server rejects any traffic destined for the virtual server IP address.
DHCP Relay	A DHCP Relay virtual server relays DHCP client requests for an IP address to one or more DHCP servers, and provides DHCP server responses with an available IP address for the client. (11.1.0 and later)

1.12 - Explain the security implications of adding service and/or protocol profiles to a virtual server

Profiles and security

Newer versions of TMOS have added features that will do protocol level inspection to work in conjunction with profiles to provide security for the applications you Adding protocol and service profiles to a virtual server help to provide additional level of security for the traffic flowing through the BIG-IP platform. Each profile tells the virtual server how to process the traffic according to the settings defined in the profile. This can cause traffic, that is not following the protocol RFC or that is falling out of the guides of the profile settings, to cause log level events or even break the flows.

1.12 - Differentiate between client side and server side settings

Client side and Server side

The concept of Client side and server side is important for most administrators to understand. It concept is straight forward but very important to remember that in a full proxy environment we need to think about the connections from client to server as multiple connections and thus the actions that need to be taken or processing that has to happen may need to be done in different ways on either side of the proxy.

A few of the profile settings in the configuration of a virtual server provide the option to use a separate client side and server side profile. This gives the administrator the ability to process traffic different on each side of the proxied connection for protocol level traffic and for SSL termination and encryption. For example you could set a TCP profile for a virtual server that is WAN optimized for the client side and LAN optimized for the server side. This is better known as TCP express and is a very powerful function that the BIG-IP can perform.

Objective - 1.13 - Given a set of application requirements, determine the appropriate virtual server configuration settings

1.13 - Describe which steps are necessary to complete prior to creating the virtual server

Virtual Servers

Configuring Virtual Servers

When creating virtual server objects in the config utility or the GUI you will need to do a few tasks prior to jumping into building the virtual server. It is true that you can create a simple virtual server in the GUI without doing anything prior because the GUI will allow you to build the Pool and Node objects on the fly inside the virtual server creation task. If you are creating a new virtual server in the config utility you will have to create all necessary configuration objects that the virtual server will need to use for its creation.

It is always a good idea to have an established object naming convention figured out prior to configuring any objects as well as a good understanding of IP addresses to be used for the virtual server creation.

The objects you should create are as follows:

1. Health monitors for nodes and pool members
2. SNAT pools

3. Any necessary profiles
4. Any necessary iRules
5. Nodes that will be used in the pool members
6. Pool and pool members

1.13 - Describe the security options when creating a virtual server (i.e., VLAN limitation, route domains, packet filters, iRules)

Virtual Server Security

A virtual server is essentially a listener that will be taking in and processing traffic on the BIG-IP platform. Some of the biggest security risks when configuring a virtual server are how it is listening, where it is listening and who can get to it. If you are configuring a virtual server and not setting the necessary settings to restrict these areas of concern you are opening your self up to security risks.

How Is The Virtual Server Listening?

The broader you set a virtual server to listen the greater the risk of unintended inbound traffic. An application based virtual server should typically be configured to listen on the default port for the application. For example if you are configuring a virtual server for a new HTTP based website you would listen on port 80. If you listen on all ports (*), the virtual server will take in traffic destine for the virtual server on all 65535 ports of the IP address. And if the pool members for the virtual server are also listening on all ports (*), it will send traffic to the servers on the port it arrived on the virtual server.

If you need to listen on multiple ports for the same IP address you can approach this in two different ways. You can build a virtual server for each necessary port using the same IP address or you can build one virtual server on all ports and use an iRule to restrict the allowed inbound connections to your list of ports.

Where is the Virtual Server Listening?

When you configure a virtual server you tell the BIG-IP where you want it to listen for traffic destined for the IP address of the virtual server. This virtual server setting is the VLAN and Tunnel Traffic setting. By default the setting is set to All VLANs and Tunnels. Which means the BIG-IP will listen on all VLANs. You are probably thinking, ARP is only going to happen on the local subnet's VLAN, which is true. So what can it possibly mean to listen on all VLANs? When this setting is set to all VLANs it means that if traffic comes to BIG-IP destined for the virtual server address from a VLAN that is not the VLAN of the virtual server IP address, it will still take the traffic in on VLAN interface that it arrived on. BIG-IP is a default deny device but in setting the setting to All VLANS and Tunnels you have told the system to listen on all VLANs for traffic to the virtual server and allow it in.

Working with Route Domains

Route Domains

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

Because route domains segment network traffic, you can assign the same IP address or subnet to multiple nodes on a network, provided that each instance of the IP address resides in a separate routing domain in the BIG-IP system. This feature will allow you to isolate traffic and let upstream devices such as firewalls apply policy to the connections between systems.

Introduction to Packet Filtering

Packet Filters

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

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

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

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

You can also configure global packet filtering that applies to all packet filter rules that you create. The following sections describe how to use the Configuration utility to set global packet filtering options, as well as create and manage individual packet filters rules.

Introduction to iRules

iRules

You can use iRules to restrict traffic in almost anyway you can think of. You can set an iRule to keep connections from happening when coming from a certain IP address range or to a certain URI path in the HTTP request.

Objective - 1.14 - Explain the matching order of multiple virtual servers

1.14 - Explain the matching order of multiple virtual servers

Order of Precedence for Virtual Server Matching (9.x - 11.2.1)

Virtual Server Matching Order

The BIG-IP system determines the order of precedence applied to new inbound connections using an algorithm that places a higher precedence on the address netmask and a lesser emphasis on the port. BIG-IP LTM sets virtual server precedence according to the following criteria:

- The first precedent of the algorithm chooses the virtual server that has the longest subnet match for the incoming connection.
- If the number of bits in the subnet mask match, the algorithm chooses the virtual server that has a port match.
- If no port match is found, the algorithm uses the wildcard server (if a wildcard virtual server is defined).
- A wildcard address has a netmask length of zero; thus, it has a lower precedence than any matching virtual server with a defined address.

This algorithm results in the following order of precedence:

- <address>:<port>
- <address>:*
- <network>:<port>
- <network>:*
- *:<port>
- *:*

Example of VIP precedence behavior

For example, for a BIG-IP system with the following VIPs configured on the inbound VLAN:

`10.0.0.0/8:80`

`10.10.0.0/16:80`

`10.10.10.10/32:80`

`20.0.0.0/8:*`

`20.0.0.0/8:80`

`*:80` (alternatively noted as `0.0.0.0/0:80`)

`*:*` (alternatively noted as `any:any, 0.0.0.0/0:any`)

The following table illustrates how inbound destination addresses map to the configured VIPs:

Inbound destination address	VIP
`10.10.10.10:80`	`10.10.10.10/32:80` - address match and port match
`10.10.10.11:80`	`10.10.0.0/16:80` - most specific address match and port match
`10.1.10.10:80`	`10.0.0.0/8:80` - most specific address match and port match
`20.0.0.0:80`	`20.0.0.0/8:80` - most specific address match and port match
`20.0.0.0:443`	`20.0.0.0/8:*` - most specific address match with wildcard port
`1.1.1.1:443`	`*:*` - wildcard address and wildcard port

Objective - 1.15 - Given a scenario, determine the appropriate load balancing method(s)

1.15 - Identify the behavior of the application to be load balanced

Application behavior

The key to choosing which load balancing method to use for an application, is to understand the behavior of the application. What understanding the behavior really means is, to understand how users access and use the application. Every application will have somewhat different client access behaviors. You will need to gather as much information as you can to gain an understanding of how the application is used. Sometimes this may mean using the application yourself.

If the application is used the same by every user, and the amount of data transmitted and length of the connection time to the application is the same for every user; then Round Robin will likely work well for the application. If you have disparate hardware all servicing the same application you may want to use a ratio based algorithm.

But as the transmitted size of the content and length of the session varies more you will need to move to dynamic algorithms. Dynamic load balancing methods take into account one or more dynamic factors, such as current connection count or even server system performance. Because each application is unique, and distribution of load can depend on a number of different factors, it is recommend that you experiment with different load balancing methods, and select the one that offers the best performance in your particular application.

1.15 - Differentiate different load balancing methods

Pools

Load Balancing Methods

There are many different types of load balancing algorithms that can be used to decide how connections are distributed across a group of servers that are hosting an application or service. All of these can be grouped into two different types of algorithms, static and dynamic.

Some examples of static load balancing algorithms are Round-Robin and Ratio. These types of algorithms do not take any environmental information into consideration and simply do what they are defined to do for connection distribution. Round-Robin may work well for short-lived, simple connections that return about the same amount of data in all responses. Ratio is typically used when the servers in the group are not all of equal capacity, or licensing levels differ per server and an uneven load should go to one server over the other in the group.

Some examples of dynamic load balancing algorithms are least connections and fastest. These types of algorithms can be affected by environmental information and use that information to make a better server choice. Least Connections looks at current connection counts at Layer 4 to the server and choses the server with the least connections. Fastest looks at the outstanding Layer 7 request and choses the server with the lowest amount.

Local Traffic Manager load balancing methods

Method	Description	When to use
Round Robin	This is the default load balancing method. Round Robin mode passes each new connection request to the next server in line, eventually distributing connections evenly across the array of machines being load balanced.	Round Robin mode works well in most configurations, especially if the equipment that you are load balancing is roughly equal in processing speed and memory.
Ratio (member) Ratio (node)	Local Traffic Manager distributes connections among pool members or nodes in a static rotation according to ratio weights that you define. In this case, the number of connections that each system receives over time is proportionate to the ratio weight you defined for each pool member or node. You set a ratio weight when you create each pool member or node.	These are static load balancing methods, basing distribution on user-specified ratio weights that are proportional to the capacity of the servers.
Dynamic Ratio (member) Dynamic Ratio (node)	The Dynamic Ratio methods select a server based on various aspects of real-time server performance analysis. These methods are similar to the Ratio methods, except that with Dynamic Ratio methods, the ratio weights are system-generated, and the values of the ratio weights are not static. These methods are based on continuous monitoring of the servers, and the ratio weights are therefore continually changing. **Note:** *To implement Dynamic Ratio load balancing, you must first install and configure the necessary server software for these systems, and then install the appropriate performance monitor.*	The Dynamic Ratio methods are used specifically for load balancing traffic to RealNetworks® RealSystem® Server platforms, Windows® platforms equipped with Windows Management Instrumentation (WMI), or any server equipped with an SNMP agent such as the UC Davis SNMP agent or Windows 2000 Server SNMP agent.

Method	Description	When to use
Fastest (node) Fastest (application)	The Fastest methods select a server based on the least number of current sessions. These methods require that you assign both a Layer 7 and a TCP type of profile to the virtual server. **Note:** If the OneConnectTM feature is enabled, the Least Connections methods do not include idle connections in the calculations when selecting a pool member or node. The Least Connections methods use only active connections in their calculations.	The Fastest methods are useful in environments where nodes are distributed across separate logical networks.
Least Connections (member) Least Connections (node)	The Least Connections methods are relatively simple in that Local Traffic Manager passes a new connection to the pool member or node that has the least number of active connections. **Note:** If the OneConnect feature is enabled, the Least Connections methods do not include idle connections in the calculations when selecting a pool member or node. The Least Connections methods use only active connections in their calculations.	The Least Connections methods function best in environments where the servers have similar capabilities. Otherwise, some amount of latency can occur. For example, consider the case where a pool has two servers of differing capacities, A and B. Server A has 95 active connections with a connection limit of 100, while server B has 96 active connections with a much larger connection limit of 500. In this case, the Least Connections method selects server A, the server with the lowest number of active connections, even though the server is close to reaching capacity. If you have servers with varying capacities, consider using the Weighted Least Connections methods instead.

Method	Description	When to use
Weighted Least Connections (member) Weighted Least Connections (node)	Like the Least Connections methods, these load balancing methods select pool members or nodes based on the number of active connections. However, the Weighted Least Connections methods also base their selections on server capacity. The Weighted Least Connections (member) method specifies that the system uses the value you specify in Connection Limit to establish a proportional algorithm for each pool member. The system bases the load balancing decision on that proportion and the number of current connections to that pool member. For example, member_a has 20 connections and its connection limit is 100, so it is at 20% of capacity. Similarly, member_b has 20 connections and its connection limit is 200, so it is at 10% of capacity. In this case, the system select selects member_b. This algorithm requires all pool members to have a non-zero connection limit specified. The Weighted Least Connections (node) method specifies that the system uses the value you specify in the node's Connection Limit setting and the number of current connections to a node to establish a proportional algorithm. This algorithm requires all nodes used by pool members to have a non-zero connection limit specified. If all servers have equal capacity, these load balancing methods behave in the same way as the Least Connections methods. **Note:** If the OneConnect feature is enabled, the Weighted Least Connections methods do not include idle connections in the calculations when selecting a pool member or node. The Weighted Least Connections methods use only active connections in their calculations.	Weighted Least Connections methods work best in environments where the servers have differing capacities. For example, if two servers have the same number of active connections but one server has more capacity than the other, Local Traffic Manager calculates the percentage of capacity being used on each server and uses that percentage in its calculations.
Observed (member) Observed (node)	With the Observed methods, nodes are ranked based on the number of connections. The Observed methods track the number of Layer 4 connections to each node over time and create a ratio for load balancing.	The need for the Observed methods is rare, and they are not recommended for large pools.

Method	Description	When to use
Predictive (member) Predictive (node)	The Predictive methods use the ranking methods used by the Observed methods, where servers are rated according to the number of current connections. However, with the Predictive methods, Local Traffic Manager analyzes the trend of the ranking over time, determining whether a nodes performance is currently improving or declining. The servers with performance rankings that are currently improving, rather than declining, receive a higher proportion of the connections.	The need for the Predictive methods is rare, and they are not recommend for large pools.
Least Sessions	The Least Sessions method selects the server that currently has the least number of entries in the persistence table. Use of this load balancing method requires that the virtual server reference a type of profile that tracks persistence connections, such as the Source Address Affinity or Universal profile type. **Note:** The Least Sessions methods are incompatible with cookie persistence.	The Least Sessions method works best in environments where the servers or other equipment that you are load balancing have similar capabilities.

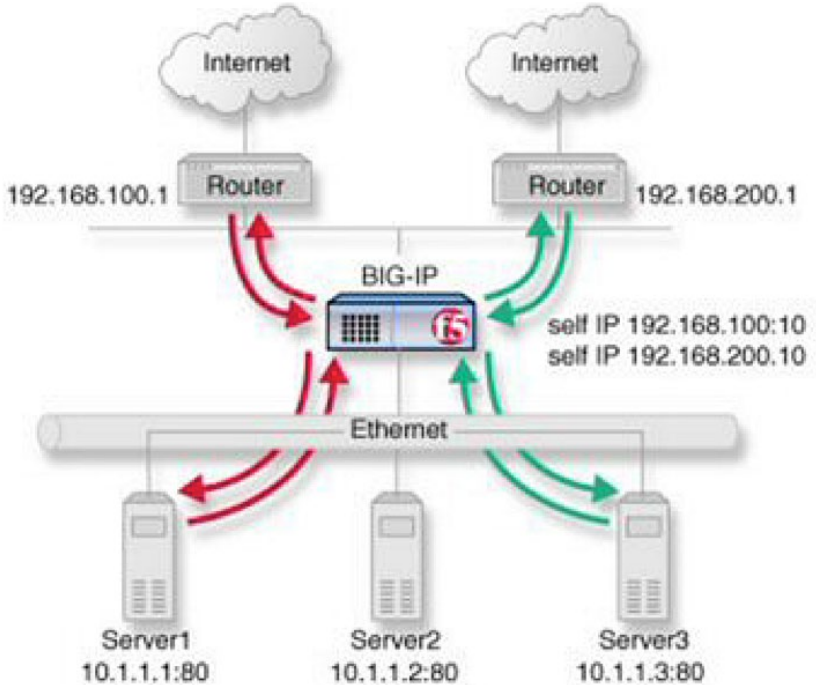
1.15 - Explain how to perform outbound load balancing

Load Balancing ISPs

Outbound Load Balancing

You might find that as your network grows, or network traffic increases, you require an additional connection to the Internet. You can use this configuration to add an Internet connection to your existing network. The following illustration shows a network configured with two Internet connections.

Illustration of ISP load balancing



ISP load balancing (Outbound Load Balancing)

Task summary for ISP load balancing

Creating a pool of outbound routers

You can a create load balancing pool, which is a logical set of devices, such as web servers, that you group together to receive and process traffic, to efficiently distribute the load on your resources. Using this procedure, create one pool to load balance the routers.

1. On the Main tab, click Local Traffic > Pools. The Pool List screen opens.
2. Click Create. The New Pool screen opens.
3. In the Name field, type a unique name for the pool.
4. For the Health Monitors setting, in the Available list, select a monitor type, and click << to move the monitor to the Active list.
5. From the Load Balancing Method list, select how the system distributes traffic to members of this pool. The default is Round Robin.

6. For the Priority Group Activation setting, specify how to handle priority groups:
    - Select Disabled to disable priority groups. This is the default option.
    - Select Less than, and in the Available Members field, type the minimum number of members that must remain available in each priority group in order for traffic to remain confined to that group.
7. Using the New Members setting, add each resource that you want to include in the pool:
    - Either type an IP address in the Address field, or select a node address from the Node List.
    - Type a port number in the Service Port field, or select a service name from the list.
    - To specify a priority group, type a priority number in the Priority field.
    - Click Add.
8. Click Finished.

The load balancing pool appears in the Pools list.

Creating a virtual server for outbound traffic for routers

You must create a virtual server to load balance outbound connections. The default pool that you assign as a resource in this procedure is the pool of routers.

1. On the Main tab, click Local Traffic > Virtual Servers. The Virtual Server List screen displays a list of existing virtual servers.
2. Click the Create button. The New Virtual Server screen opens.
3. In the Name field, type a unique name for the virtual server.
4. Specify the Destination setting, using the Address field; type the IP address you want to use for the virtual server. The IP address you type must be available and not in the loopback network.
5. In the Resources area of the screen, from the Default Pool list, select a pool name.
6. Click Finished.

The virtual server is configured to load balance outbound connections to the routers.

Creating self IP addresses an external VLAN

You must assign two self IP addresses to the external VLAN.

1. On the Main tab, click Network > Self IPs. The Self IPs screen opens.

2. Click Create. The New Self IP screen opens.
3. In the IP Address field, type an IP address. This IP address should represent the network of the router. The system accepts IP addresses in both the IPv4 and IPv6 formats.
4. In the Netmask field, type the network mask for the specified IP address.
5. Select External from the VLAN list.
6. Click Repeat.
7. In the IP Address field, type an IP address. This IP address should represent the address space of the VLAN that you specify with the VLAN/Tunnel setting. The system accepts IP addresses in both the IPv4 and IPv6 formats.
8. Click Finished. The screen refreshes, and displays the new self IP address in the list.

The self IP address is assigned to the external VLAN.

Enabling SNAT automap for internal and external VLANs

You can configure SNAT automapping on the BIG-IP system for internal and external VLANs.

1. On the Main tab, click Local Traffic > SNATs. The SNAT List screen displays a list of existing SNATs.
2. Click Create.
3. Name the new SNAT.
4. From the Translation list, select automap.
5. For the VLAN List setting, in the Available field, select external and external, and using the Move button, move the VLANs to the Selected field.
6. Click Finished.

SNAT automapping on the BIG-IP system is configured for internal and external VLANs.

1.15 - Explain CARP persistence

Overview of the CARP hash algorithm

CARP Persistence

The hash persistence profile can perform a pool member selection using a stateless hash algorithm based on the Cache Array Routing Protocol (CARP). Additionally, starting in BIG-IP 10.2.0, the source address affinity

and destination address affinity persistence profiles can also perform pool member selection using a stateless hash algorithm based on CARP. The CARP algorithm was originally described in an IETF draft, which is available at the following location:

Cache Array Routing Protocol v1.0

The CARP algorithm provides the following advantages over stateful persistence methods:

- No memory storage required

  The CARP algorithm uses a stateless selection; therefore, it does not use the persistence table, which is stored in memory.

  Note: Since the CARP algorithm does not store persistence records, commands such as tmsh show / ltm persistence persist-records and bigpipe persist show all cannot be used to monitor CARP persistence.

- No timeout

  Since the CARP algorithm does not use the persistence table, there is no timeout associated.

- Failover without mirroring

  Since the CARP persistence method uses a stateless election, no mirroring of persistence data is required in order to maintain persistence after a failover event.

- Automatic redistribution

  Since the CARP algorithm is based on available pool members, the algorithm automatically adjusts as pool members become available or unavailable, and has the benefit of reducing the percentage of cache misses compared with the default hash algorithm. For example, if you have a pool of four HTTP cache servers, and one server goes offline, the CARP algorithm will redistribute those requests to the remaining servers. Therefore, only connections that were going to that one server will be redistributed. Additionally, when that server is brought back online, only those requests that were originally sent to that server will be directed back. In the case of the default hash algorithm, all requests will be randomly redistributed across the remaining servers each time a server transitions from offline to online.

CARP is an excellent choice for load balancing a pool of HTTP cache proxies, when used in combination with the HTTP and OneConnect profiles. However, the CARP algorithm is not limited to HTTP traffic. Other applications, which are compatible with stateless persistence, may also benefit from the advantages listed above.

Objective - 1.16 - Explain the effect of LTM device configuration parameters on load balancing decisions

1.16 - Differentiate between members and nodes

Nodes

Nodes vs Members

A node is a logical object on the BIG-IP Local Traffic Manager system that identifies the IP address of a physical resource on the network. You can explicitly create a node, or you can instruct Local Traffic Manager to automatically create one when you add a pool member to a load balancing pool.

The difference between a node and a pool member is that a node is designated by the devices IP address only (10.10.10.10), while designation of a pool member includes an IP address and a service (such as 10.10.10:80).

A primary feature of nodes is their association with health monitors. Like pool members, nodes can be associated with health monitors as a way to determine server status. However, a health monitor for a pool member reports the status of a service running on the device, whereas a health monitor associated with a node reports status of the device itself.

For example, if an ICMP health monitor is associated with node 10.10.10.10, which corresponds to pool member 10.10.10.10:80, and the monitor reports the node as being in a down state, then the monitor also reports the pool member as being down. Conversely, if the monitor reports the node as being in an up state, then the monitor reports the pool member as being either up or down, depending on the status of the service running on it.

Nodes are the basis for creating a load balancing pool. For any server that you want to be part of a load balancing pool, you must first create a node, that is, designate that server as a node. After designating the server as node, you can add the node to a pool as a pool member. You can also associate a health monitor with the node, to report the status of that server.

1.16 - Explain the effect of the load balancing method on the LTM platform

The Ratio Load balancing method is more CPU intensive than the Round Robin load balancing method

Load Balancing Method Effect on LTM

The function of load balancing traffic based on the processing of an algorithm against the available pool members will uses system resources. Each algorithm may use a different amount of system resources. Static load Balancing methods will tend to use fewer resources than a more advanced dynamic algorithm. However if a more advanced method is needed to distribute the application connections across the servers then that is what should be used. The additional overhead will likely be nominal to the overall system performance. If you are using a method that could be done with a less complex algorithm then you should step it down to that algorithm.

For Example:

The Ratio load balancing method uses an algorithm that is more CPU intensive than the algorithm used by the Round Robin load balancing method. For this reason, you should use the Ratio load balancing method only when different node or pool member weights are required. For example, you should modify a pool that distributes connections among four pool members in a static rotation according to a 1:1:1:1 ratio to use the Round Robin load balancing method. In large configurations, this method can significantly reduce the CPU load.

1.16 - Explain the effect of CMP on load balancing methods

Overview of Clustered Multiprocessing (11.0.0 - 11.2.x)

CMP Effects Load Balancing Methods

CMP should not be confused with Symmetric Multi-Processing (SMP). SMP architecture is used in multiple operating systems. SMP operates by allowing operating systems and software applications that are optimized for SMP to use the multiple processors available to the operating system. SMP performs this operation by spreading multiple threads across multiple processors, which allows for faster processing and more efficient use of system resources, as multiple threads can be processed simultaneously instead of waiting in a queue to be processed. CMP uses a similar approach to leverage multiple processing units by spawning a separate instance of the TMM process on each processing unit available to the system. While SMP may be used for any process, CMP processing is available only to the BIG-IP TMM process for the sole purpose of providing more dedicated resources to manage load balanced traffic. With multiple TMM instances simultaneously processing traffic, system performance is enhanced, and traffic management capacity is expanded.

The CMP feature is automatically enabled on CMP-capable platforms, ensuring that all instances of TMM are available to process application traffic as follows:

- 2 TMM processes running on a dual-CPU BIG-IP 8400 system
- 4 TMM processes running on a dual-core, dual-CPU BIG-IP 8800 system
- 32 TMM processes running on a VIPRION platform with four 2100 blades

Load balancing behavior on CMP enabled virtual servers

Connections on a CMP enabled virtual server are distributed among the available TMM processes. The load balancing algorithm, specified within the pool associated with the CMP enabled virtual server, is applied independently in each TMM. Since each TMM handles load balancing independently from the other TMMs, distribution across the pool members may appear to be incorrect when compared with a non-CMP enabled virtual server using the same load balancing algorithm.

Consider the following example configuration:

Virtual Server: `172.16.10.10:80`
Pool with 4 members: `10.0.0.1:80`
`10.0.0.2:80`
`10.0.0.3:80`
`10.0.0.4:80`

Pool Load Balancing Method: Round Robin

Scenario 1: Virtual server without CMP enabled

Four connections are made to the virtual server. The BIG-IP system load balances the four individual connections to the four pool members based on the Round Robin load balancing algorithm:

--Connection 1-->| | --Connection 1--> `10.0.0.1:80`
--Connection 2-->| -> BIG-IP Virtual Server -> | --Connection 2--> `10.0.0.2:80`
--Connection 3-->| | --Connection 3--> `10.0.0.3:80`
--Connection 4-->| | --Connection 4--> `10.0.0.4:80`

Scenario 2: Virtual server with CMP enabled on a BIG-IP 8800

Four connections are made to the virtual server, unlike the first scenario where CMP was disabled, the BIG-IP distributes the connections across the multiple TMM processes. The BIG-IP 8800 with CMP enabled can use four TMM processes. Since each TMM handles load balancing independently of the other TMM processes, it is possible that all four connections are directed to the same pool member.

```
--Connection 1--> |                                | --Connection 1--> TMM0 --> 10.0.0.1:80
--Connection 2--> | -> BIG-IP Virtual Server ->    | --Connection 2--> TMM1 --> 10.0.0.1:80
--Connection 3--> |                                | --Connection 3--> TMM2 --> 10.0.0.1:80
--Connection 4--> |                                | --Connection 4--> TMM3 --> 10.0.0.1:80
```

The CMP feature is designed to speed up connection handling by distributing connections across multiple TMM processes. While initially this behavior may appear to favor one or several servers, over time the load will be distributed equally across all servers.

1.16 - Explain the effect of OneConnect/MBLB on load balancing

Managing Connection Reuse Using OneConnect Source Mask

OneConnect

The BIG-IP OneConnect feature can increase network throughput by efficiently managing connections created between the BIG-IP system and back end nodes. OneConnect allows the BIG-IP system to minimize the number of server-side TCP connections by making existing idle connections available for reuse by other clients. The OneConnect source mask setting manages connection reuse, and is applied to the server-side source IP address of a request to determine its eligibility for connection reuse.

Overview of the OneConnect Mask

OneConnect applies a mask (much like applying an independent subnet mask) to client source IP addresses on server-side connections. This mask determines the availability of an existing idle TCP connection for the request on the selected destination server. The following list displays the idle TCP port reuse behavior. To simplify things, regarding these explanations, F5 assumes that a previous TCP request has established a connection on the destination server, the connection is idle, and the same destination server has been selected for a subsequent client request.

OneConnect Mask	OneConnect masking behavior
255.255.255.255	The entire client IP address is evaluated. A request from the same address reuses an established idle TCP connection.
255.255.255.0	Octets 1-3 are evaluated client addresses matching this subset and reuse an existing idle TCP connection.
255.255.0.0	Client addresses are evaluated based on the first and second octets when selecting an idle TCP connection for reuse.
255.0.0.0	Only the first octet of a client IP address is evaluated when selecting an idle TCP connection for reuse.
0.0.0.0	The all zeros mask (the default setting) looks for any open idle TCP connection on the destination server. The main difference is OneConnect does not attempt to group the request based on octets matched, but uses open idle TCP connections in a highly efficient manner.

Effects of modifying the source mask

The following three scenarios describe the effects of using different source masks:

Using a OneConnect profile with a 0.0.0.0 source mask

A OneConnect profile with a source mask of `0.0.0.0` shares idle connections across all client requests in the following manner:

- Client A with source IP address `10.10.10.10` connects to a virtual server.
- The BIG-IP system load-balances the connection and applies the source mask to the request on the server-side flow, finds no suitable connection for reuse, and creates a TCP connection to server A in the pool.
- Client B with source IP address `10.20.20.20` connects to the same virtual server.
- The BIG-IP system load-balances the connection, applies the source mask to the server-side flow, and finds an eligible idle TCP connection.
- The BIG-IP system aggregates the request from client B over the existing TCP connection created for client A.

Using a OneConnect profile with a 255.255.255.0 source mask

A OneConnect profile with a source mask of `255.255.255.0` aggregates connections from client IP addresses sharing the same last octet in the following manner:

- Client A with a source IP address of `10.10.10.10` connects to a virtual server.
- The BIG-IP system load-balances the connection and applies the source mask to the request on the server-side flow, finds no suitable connection for reuse, and creates a TCP connection to server A in the pool.
- Client B with a source IP address of `10.10.10.100` connects to the same virtual server.
- The BIG-IP system load-balances the connection, applies the source mask to the server-side flow, and finds an eligible idle TCP connection.
- The BIG-IP system aggregates the request from client B over the existing TCP connection created for client A.
- Client C with source IP address `10.10.20.10` connects to the same virtual server.
- The BIG-IP system load-balances the connection, applies the source mask to the server-side flow, and finds no suitable connection for reuse.
- The BIG-IP system creates a new TCP connection to the selected pool member.

Using a OneConnect profile with a 255.255.255.255 source mask

A OneConnect profile with a source mask of `255.255.255.255` will only aggregate connections originating from the same server-side client IP address.

1.16 - Explain how monitors and load balancing methods interact

Causes of Uneven Traffic Distribution Across BIG-IP Pool Members

Monitors and Load Balancing Methods

The BIG-IP system is designed to distribute client requests to load balancing pools composed of multiple servers. Factors such as the BIG-IP configuration, server performance, and network-related issues determine the pool member to which the BIG-IP system sends the connection, and whether connections are evenly distributed across BIG-IP pool members. For example, a virtual server referencing a Round Robin pool will distribute connections across BIG-IP pool members evenly over time. However, if the same virtual server also

references a BIG-IP configuration object that affects traffic distribution such as a OneConnect profile or an iRule, connections may not be evenly distributed, as expected.

Factors affecting traffic distribution across pool members are discussed below.

Load balancing methods

The load balancing algorithm is the primary mechanism that determines how connections are distributed across pool members. You can define static or dynamic load balancing methods for a pool. Certain load balancing methods are designed to distribute requests evenly across pool members, and other load balancing methods are designed to favor higher performing servers, possibly resulting in uneven traffic distribution across pool members.

Static load balancing methods

Certain static load balancing methods are designed to distribute traffic evenly across pool members. For example, the Round Robin load balancing method causes the BIG-IP system to send each incoming request to the next available member of the pool, thereby distributing requests evenly across the servers in the pool. However, when a static load balancing method such as Round Robin is used along with a BIG-IP configuration object that affects load distribution, such as a OneConnect profile or a persistence profile, traffic may not be evenly distributed across BIG-IP pool members as expected.

Dynamic load balancing methods

Dynamic load balancing methods typically favor higher performing servers, and may result in uneven traffic distribution across pool members. Dynamic load balancing methods are designed to work with servers that differ in processing speed and memory. For example, when a dynamic load balancing method such as the Observed method is defined for a pool, higher performing servers will process more connections over time than lower performing servers. As a result, connection statistics for the higher performing servers will exceed those for lower performing severs.

SECTION 2 - SET-UP, ADMINISTER, AND SECURE LTM DEVICES

Objective - 2.01 Distinguish between the management interface configuration and application traffic interface configuration

2.01 - Explain the requirements for management of the LTM devices

Overview of the Management Port

LTM Management

The BIG-IP can be managed through either the TMM switch interfaces or the MGMT interface. However, F5 recommends that you use the management port.

The TMM switch ports are the interfaces that the BIG-IP system uses to send and receive load-balanced traffic.

The system uses the MGMT interface to perform system management functions. The MGMT interface is intended for administrative traffic and cannot be used for load-balanced traffic. Additionally, since no access controls can be applied on the MGMT interface, F5 recommends that you limit network access through the MGMT interface to trusted traffic. For security reasons, the MGMT interface should be connected to only a secure, management-only network, such as one that uses an RFC1918 private IP address space. If you do not have a trusted and secure management network, F5 recommends that you do not use the MGMT interface, and that you grant administrative access through the TMM switch interfaces or the local serial console.

2.01 - Explain the requirements for the application traffic traversing the LTM devices

BIG-IP Local Traffic Manager: Implementations

Requirements for Application Traffic to Traverse the LTM

The BIG-IP system must be connected to a client accessible network as well as the server network. These networks may be one in the same or separate networks in your environment. The BIG-IP should have a default gateway configured. The client will access the application via a virtual server configured on the client side network. The virtual server will pass traffic to a pool member configured in the pool associated to the virtual server. This will send the traffic to the server defined as the pool member. The server will receive the traffic and process it as necessary. The server will respond to the client of the connection. The traffic will need to pass back through the BIG-IP to be processed back to the client correctly. If the server has a default path back to the client that does not traverse the BIG-IP platform the communication will fail. If this is the case a SNAT of the clients source IP address will correct this issue. If the servers default path to the client is through the BIG-IP the system will handle rewriting the packet back to the client correctly.

2.01 - Explain how to configure management connectivity options: AOM, serial console, USB & Management Ethernet Port

Performing a Clean Installation of BIG-IP 11.x or Enterprise Manager 3.x

USB

On rare occasions, you may be required to perform a clean installation of BIG-IP 11.x. During a clean installation, all mass-storage devices are wiped, therefore restoring the BIG-IP system to its factory defaults. In addition, a clean installation allows you to reinstall a BIG-IP unit that no longer boots from any of its boot locations.

You should choose an installation method based on the equipment available to you, and whether you have physical access to the system that requires reinstalling. Choose one of the following installation methods, which are listed in order of preference:

USB DVD-ROM drive

> **Note:** F5 recommends that you perform a clean installation by using a USB DVD-ROM drive, as this is the simplest and most reliable of all the installation methods.

USB thumb drive

Burn the product ISO image to a DVD.

Image the USB thumb drive using the product ISO image file.

Installing the software

1. Connect to the BIG-IP system serial console.
2. Depending on the choice you made in the previous procedure, perform one of the following actions:
   - Connect the USB DVD-ROM drive to the F5 system and load the disc you burned with the product ISO image.
   - Connect the USB thumb drive to the F5 system..
3. Reboot the BIG-IP system. If the F5 system cannot reboot, power cycle the BIG-IP system.

   Note: Upon completion of this step, regardless of the installation method, the BIG-IP system boots into the Maintenance Operating System (MOS).
4. The MOS asks you to specify the type of terminal you are using. If you do not know what to specify, press Enter. The default setting (vt100) is fine in most cases.
5. If you have booted the F5 system from a USB device, the system may display a manufacturing installation dialog.
6. Press Ctrl+C to exit the dialog.
7. Continue with step 8 if you have booted the F5 system from a USB removable media containing a BIG-IP 11.0.0 image, but you want to perform a custom installation using the diskinit and image2disk utilities. Otherwise, to reinstall the system according to the manufacturing installation plan displayed in the output, press Enter and skip to step 10.
8. To wipe all mass-storage devices inside the BIG-IP system, type the following command:

   ```
   diskinit --style volumes
   ```

   Important: Do not omit the --style option; if you omit it, the system wipes the drives but does not reformat them.
9. The diskinit utility asks whether you want to proceed wiping the drives. To continue, type y and press Enter. Otherwise, type n and press Enter.

Important: Confirming this operation destroys all data on the system. Do not proceed with this step if you have data that needs to be recovered from the system. Using the MOS, you may be able to manually mount a partition or volume and recover such data.

10. Install the software using one of the following methods:

    - If you are using a USB DVD-ROM drive or a USB thumb drive, use the following command:
      `image2disk --format=volumes --nosaveconfig --nosavelicense`
    - If you are using a PXE server, use the following command syntax:
      `image2disk --format=volumes --nosaveconfig --nosavelicense http://<SERVER_IP>/<PATH>`

      For example, to install BIG-IP 11.x on HD1.1 using the http server configured in the previous procedure, type the following command:
      `image2disk --format=volumes --nosaveconfig --nosavelicense http://192.168.1.1/SOL13117`

    Note: BIG-IP 11.x cannot be installed on a CompactFlash media drive; you must use boot locations on the system's hard drive.

    Note: You must specify the --nosaveconfig option, as the system does not have a configuration to save.

    Note: If you are using a USB DVD-ROM drive or a USB thumb drive, you do not need to specify an installation repository, as the image2disk utility automatically finds and defaults to /cdserver.

    Note: For more information about the image2disk utility, refer to the Help screen by using the image2disk --h command.

11. Once the installation has completed, disconnect any removable media from the BIG-IP system.

12. To restart the system, type the following command:
    `reboot`

    The system boots from the location you have just reinstalled.

Connecting a Serial Terminal to a BIG-IP System

Serial Console

You can administer a BIG-IP system by using a null modem cable to connect a management system that runs a terminal emulator program to the BIG-IP serial port. To connect to the BIG-IP system using the serial port, you must have a DB9 null modem cable and a VT100-capable terminal emulator available on the management system.

To configure a serial terminal console for the BIG-IP system, perform the following procedure:

1. Connect the null modem cable to the console port on the BIG-IP system.
2. Connect the null modem cable to a serial port on the management system with the terminal emulator.
3. Configure the serial terminal emulator settings according to the following table:

Setting	Value
Bits per second [baud]	19200
Data bits	8
Parity	None
Stop bit	1
Flow control	None

4. Turn on the BIG-IP system.

When the BIG-IP system starts up with the console working correctly, the system start-up sequence displays, and then the sequence completes with a BIG-IP system login prompt. If garbled text displays on the console, you may be required to change the baud of the serial console port using the LCD panel on the BIG-IP system.

Configuring and Displaying the Management IP Address for the BIG-IP System

Management Ethernet Port

The management port on a BIG-IP system provides administrative access to the system out-of-band of the application traffic. This allows you to restrict administrative access to an internal secure network. You can display and configure the management IP address for the BIG-IP system using the Configuration utility, the command line, and the LCD panel.

Configuring the management IP address using the Configuration utility, command line, or LCD panel

You can configure the management IP address using the Configuration utility, the tmsh utility, the config command, or the LCD panel. To do so, perform one of the following procedures:

> **Impact of procedure:** Changing the management IP address will disconnect you from the BIG-IP system if you are connected through the management port.

Configuring the management IP address using the Configuration utility

1. Log in to the Configuration utility.
2. Navigate to System > Platform.
3. In the Management Port section, configure the IP address, network mask, and management route.
4. To save the changes, click Update.

Configuring the management IP address using the tmsh utility

1. Log in to the Traffic Management Shell (tmsh) by typing the following command:
   ```
   tmsh
   ```
2. To configure the management IP address, use the following syntax:
   ```
   create /sys management-ip [ip address/netmask]
   or
   create /sys management-ip [ip addres/prefixlen]
   ```
   For example:
   ```
   create /sys management-ip 192.168.1.245/255.255.255.0
   or
   create /sys management-ip 192.168.1.245/24
   ```
3. To configure a default management gateway, use the following syntax:
   ```
   create /sys management-route default gateway <gateway ip address>
   ```
   For example:
   ```
   create /sys management-route default gateway 192.168.1.254
   ```
4. Save the changes by typing the following command:
   ```
   save /sys config partitions all
   ```

Configuring the management IP address using the config command

1. Log in to the command line of the BIG-IP system.
2. Enter the F5 Management Port Setup Utility by typing the following command:
   ```
   config
   ```
3. To configure the management port, type the appropriate IP address, netmask, and management route in the screens that follow.

Configuring the management IP address using the LCD panel

1. Press the X button to activate Menu mode for the LCD.
2. Use the arrow keys to select System, and press the Check button.
3. To select Management, press the Check button.
4. To select Mgmt IP, press the Check button.
5. Enter your management IP address using the arrow keys, and press the Check button.
6. Use the arrow keys to select Mgmt Mask, and press the Check button.
7. Enter the netmask using the arrow keys, and press the Check button.
8. Use the arrow keys to select Mgmt Gateway, and press the Check button.
9. Enter your default route using the arrow keys, and press the Check button.

   If you do not have a default route, enter 0.0.0.0.
10. Use the arrow keys to select Commit, and press the Check button.
11. To select OK, press the Check button.

Configuring AOM Network Access (2000, 4000, 5000, 7000, and 10000 Series Platforms)

AOM

Always-On Management (AOM) is a separate subsystem that provides lights-out management for the BIG-IP system by using the 10/100/1000 Ethernet management port over secure shell (SSH), or by using the serial console.

AOM allows you to manage BIG-IP platforms using SSH (most platforms) or the serial console, even if the Host subsystem is turned off. The BIG-IP Host subsystem and the AOM subsystem operate independently. If AOM is reset or fails, the BIG-IP Host subsystem continues to operate and there is no interruption to load-balanced traffic. AOM is always turned on when power is supplied to the platform. If the BIG-IP Host subsystem stops responding, you can use the AOM Command Menu to reset it.

Configuring AOM network access

To configure AOM so that it can be accessed over the network, perform the following procedure:

Impact of procedure: Performing the following procedure should not have a negative impact on your system.

1. Connect the serial console to the CONSOLE port.

2. Display the AOM command menu by typing the following key sequence:
   `Esc (`

   The AOM command menu displays as follows:

   AOM Command Menu:

   B --- Set console baud rate

   I --- Display platform information

   P --- Power on/off host subsystem

   R --- Reset host subsystem

   N --- Configure AOM network

   S --- Configure SSH Server

   A --- Reset AOM

   E --- Error report

   Q --- Quit menu and return to console

3. To configure network access, press the N key.

   The AOM management network configurator screen appears.

4. Complete the network configurator screens.

   Important: The AOM IP address must be different than the BIG-IP management address, but on the same IP subnet.

5. To disable the network configuration, re-run the N ---Configure AOM network option, and enter `0.0.0.0` for the IP address.

Objective - 2.02 Given a network diagram, determine the appropriate network and system settings (i.e., VLANs, selfIPs, trunks, routes, NTP servers, DNS servers, SNMP receivers and syslog servers)

2.02 - Explain the requirements for self IPs (including port lockdown)

Self IP Addresses

Self IPs

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

> **Note:** Only users with either the Administrator or Resource Administrator user role can create and manage self IP addresses.
>
> **Note:** A self IP address can be in either IPv4 or IPv6 format.

IP address

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

Netmask

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

VLAN/Tunnel assignment

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

- Assigning a self IP address to a VLAN

  The self IP address that you assign to a VLAN should represent an address space that includes the self IP addresses of the hosts that the VLAN contains. For example, if the address of one destination server

in a VLAN is `10.0.0.1` and the address of another server in the VLAN is 10.0.0.2, you could assign a self IP address of `10.0.0.100`, with a netmask of `255.255.0.0`, to the VLAN.

- Assigning a self IP address to a VLAN group

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

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

Port lockdown

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

- For UDP, the allowed protocols and services are: DNS (53), SNMP (161), RIP (520)
- For TCP, the allowed protocols and services are: SSH (22), DNS (53), SNMP (161), HTTPS (443), 4353 (iQuery)

If you do not want to use the default setting (Allow Default), you can configure port lockdown to allow either all UDP and TCP protocols and services (Allow All), no UDP protocols and services (Allow None), or only those that you specify (Allow Custom).

Traffic groups

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

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

If you want the self IP address to be a static (non-floating) IP address (used mostly for standalone devices), you can assign a non-floating traffic group to the self IP address. A non-floating traffic group causes the self IP address to become a non-floating self IP address. An example of a non-floating self IP address is the address

that you assign to the default VLAN named HA, which is used strictly to process failover communications between BIG-IP devices, instead of processing application traffic.

2.02 - Explain routing requirements for management and application traffic (including route domains and IPv6)

Overview of Management Interface Routing (11.x)

The Traffic Management Microkernel (TMM) controls all of the BIG-IP switch ports (TMM interfaces), and the underlying Linux operating system controls the BIG-IP management interface. The management interface processes only administrative traffic. The TMM interfaces process both application traffic and administrative traffic.

Traffic type

The BIG-IP system can process the following traffic types:

Application traffic

TMM processes inbound application traffic that arrives on a TMM switch interface and is destined for a BIG-IP self IP address or a virtual server address.

Administrative traffic

BIG-IP administrative traffic can be defined as follows:

- Inbound administrative connections

  Inbound connections sent to the BIG-IP management IP address that arrive on the management interface are processed by the Linux operating system. Inbound connections sent to the BIG-IP self IP addresses that arrive on a TMM interface are processed by TMM. If the self IP address is configured to allow a connection to the destination service port, TMM hands the connection off to the Linux operating system, which then processes the connection request.

- Outbound administrative connections

  Outbound connections sent from the BIG-IP system by administrative applications (SNMP, SMTP, SSH, NTP, etc.) are processed by the Linux operating system. These connections may use either the management address or a self IP address as the source address. The BIG-IP system compares the destination address to the routing table to determine the interface through which the BIG-IP system routes the traffic.

Note: This behavior applies to only unsolicited outbound traffic: traffic that is not in response to a request originated by a remote host. A response to a request originated by a remote host is returned to the last MAC address traversed by the inbound request.

Note: You can configure a health monitor to send probes using the management network. However, F5 strongly discourages this configuration because the management network is not intended for production traffic. F5 recommends that the pool members/nodes reside on a network that is reachable through TMM interfaces so that health monitor probes are sent through TMM interfaces.

BIG-IP routing tables

The BIG-IP routing table consists of the following routing subtables:

Management routes

Management routes are routes that the BIG-IP system uses to forward traffic through the management interface. For traffic sourced from the management address, the system prefers management routes over TMM routes, and uses the most specific matching management route. If no management route is defined or matched, the system uses the most specific matching TMM route.

TMM routes

TMM switch routes are routes that the BIG-IP system uses to forward traffic through the TMM switch interfaces instead of through the management interface. Routes in the TMM subtable are defined with a lower metric than routes in the management subtable. Traffic sourced from a TMM (self IP) address will always use the most specific matching TMM route. Traffic sourced from a TMM address will never use a management route. When TMM is not running, the TMM addresses are not available, and all TMM routes are removed. As a result, when TMM is not running, all outbound administrative traffic uses the most specific matching management route.

F5 recommends that you add static routes for management traffic whose destination does not match the directly-connected management network. This configuration is useful when you handle SNMP traffic that is directed to an SNMP Manager that resides on another network, which is accessible only through the management network or other network services that are hosted on networks that are not accessible by way of the TMM interfaces.

The BIG-IP system can act as a full proxy and termination point providing application availability, bi-directional address translation, authentication and security services, and DNS services that cross IPv4 and IPv6 network boundaries.

2.02 - Explain the effect of system time on LTM devices

Verifying NTP Peer Server Communications

Time

Having the correct system time set on your BIG-IP devices is critical for many different administrative functions. Time stamping for logging is all based on system time. SSL certificates could have issues with the expiration dates. In HA environments if the system time is not set correctly between the units in the HA configuration the systems may not be able to sync configs.

When the BIG-IP system clock is not showing the correct timezone, or the date and time is not synchronized correctly, this could be caused by incorrect NTP configuration or a communication issue with a valid NTP peer server. Remember that even if you have the NTP settings correct in the BIG-IP system it may not be able to reach the NTP if there is an up stream Firewall or other network restrictions.

Network Time Protocol (NTP)

NTP is a protocol for synchronizing the clocks of computer systems over the network. On BIG-IP systems, accurate timestamps are essential to guarantee the correct behavior of a number of features. While in most cases it is sufficient to configure a couple of time servers that the BIG-IP system will use to update its system time, it is also possible to define more advanced NTP configurations on the BIG-IP system.

Objective - 2.03 Given a network diagram, determine the appropriate physical connectivity

2.03 - Explain physical network connectivity options of LTM devices

Manual Chapter: BIG-IP Platform Properties

Networking Device

Depending on the model of BIP-IP platform you are working with, you will find different counts and types of network interfaces. The image below describes the different interfaces on the typical BIG-IP platform. Interface count will vary per model.

1. Management port
2. USB ports
3. Console port

4. Serial (hard-wired) failover port
5. 10/100/1000 interfaces
6. SFP ports
7. Indicator LEDs
8. LCD display
9. LCD control buttons

Physical Connections

Management Port

Every BIG-IP system has a management port. The management port is a special interface that the BIG-IP system uses to receive or send certain types of administrative traffic. You cannot use the management port for normal traffic that is slated for load balancing. Instead, the BIG-IP system uses the TMM switch interfaces for that type of traffic. TMM switch interfaces are those interfaces controlled by the Traffic Management Microkernel (TMM) service.

Configuring the management port of a BIG-IP system means assigning an IP address to the port, supplying a netmask for the IP address, and specifying an IP address for the BIG-IP system to use as a default route. The IP address that you assign to the management port must be on a different network than the self IP addresses that you assign to VLANs. Note that specifying a default route for the management port is only necessary if you intend to manage the BIG-IP system from a node on a different subnet.

> **Note:** The IP address for the management port must be in IPv4 format.

TMM Switch Ports

Auto MDI/MDIX functionality is retained when you manually configure an interface to use specific speed and duplex settings. Therefore, you can use either a straight-through cable or a crossover cable when media settings are forced, and you will be able to successfully link to either DTE or DCE devices.

The following specifications are for the available Copper Gigabit Ethernet modules and the platforms that support those modules.

1000Base-T Copper Ethernet Transceiver SFP module specifications

- Connector type: RJ45
- Maximum operating distance: 100 meters (328 feet)
- Cable specifications: Minimum Cat5; Cat5e or Cat6 recommended

There are many available SFP - Fiber Gigabit Ethernet modules for the different BIG-IP hardware platforms.

A full list can be found at the following link: Specifications of the Fiber Gigabit Ethernet SFP, XFP, SFP+ and QSFP+ Module Ports on BIG-IP System Platforms

Objective - 2.04 Explain how to configure remote authentication and multiple administration roles on the LTM device

2.04 - Explain the relationship between route domains, user roles and administrative partitions

BIG-IP User Accounts

User access relationships

Every User ID configured on the BIG-IP system is tied to either one administrative partition (i.e. typical user), or all administrative partitions (i.e. administrator). And the partition assignment defines where that user can do the functions of their role level (Guest thru Administrator). This means we can control where and what each user can affect when they are working in the BIG-IP platform by how we define each users role and partition access.

We also gain further control of user access to resources through route domains. Each administrative partition is assigned to a route domain. And since the user is tied to the partition they are as well.

2.04 - Explain the mapping between remote users and remote role groups

BIG-IP User Accounts

Understanding default remote-account authorization

The Authentication screen that you used to specify the type of remote authentication server also includes some default authorization values (for the Role, Partition Access, and Terminal Access settings). Therefore, if you do not explicitly configure these authorization settings for an individual BIG-IP system user account, the BIG-IP system assigns the default values to that account. This ensures that all remote user accounts have valid authorization settings assigned to them.

The default values for the Role, Partition Access, and Terminal Access settings are as follows:

Role–No Access

Partition Access–All

Terminal Access–Disabled

When you use these default values for a user account, the user account appears in the list of BIG-IP user accounts as Other External Users.

You can change the values that the BIG-IP system automatically uses as the default values for the Role, Partition Access, and Terminal Access settings.

To change the default authorization properties for remote user accounts, you configure the Role, Partition Access, and Terminal Access settings on the same Authentication screen that you used to specify the type of remote authentication server you are using.

Important: For the Other External Users user account, you can modify the Role, Partition Access, and Terminal Access settings only when your current partition on the BIG-IP system is set to Common. If you attempt to modify these settings when your current partition is other than Common, the system displays an error message.

Note: You can sometimes inadvertently affect your own user account or all user accounts, if the BIG-IP system is configured to perform remote user authentication, and you or another system administrator changes the default role or partition assigned to all external user accounts:

- If you log on to the BIG-IP system using one of these remotely-authenticated Administrator accounts, and you or another Administrator user modifies the default role of all external accounts from Administrator to a lesser role, the system modifies the user role of your own account to the lesser role. However, the change to your own account does not actually occur until you log off and log on again to the BIG-IP system, thus allowing you to complete any tasks you initiated that still require the Administrator role.

- Similarly, your user account can be affected if the BIG-IP system is configured to perform remote user authentication, and the default partition assigned to all external user accounts is a specific partition. In this case, if you are logged on to the BIG-IP system through the command line using one of the remotely-authenticated accounts, and another user who is logged on through the Configuration utility modifies the default partition for external users, the BIG-IP system immediately logs you off when you attempt to issue another command.

  Important: If a BIG-IP system administrator changes the user role or partition assignment (or both) for any remote user account, the BIG-IP system logs off all users immediately. (A remote user account in this case refers to Other External Users.)

2.04 - Explain the options for partition access and terminal access

BIG-IP User Accounts

Partition Access

A user role defines the access level that a user has for each object in the users assigned partition. An access level refers to the type of task that a user can perform on an object. Possible access levels are:

- Write

  Grants full access, that is, the ability to create, modify, enable and disable, and delete an object.

- Update

  Grants the ability to modify, enable, and disable an object.

- Enable/disable

  Grants the ability to enable or disable an object.

- Read

  Grants the ability to view an object.

Terminal Access

Specifies the level of access to the BIG-IP system command line interface. Possible values are: Disabled and Advanced shell.

Users with the Administrator or Resource Administrator role assigned to their accounts can have advanced shell access, that is, permission to use all BIG-IP system command line utilities, as well as any Linux commands.

Objective - 2.05 Given a scenario, determine an appropriate high availability configuration (i.e., failsafe, failover and timers)

2.05 - Compare and contrast network and serial failover

Comparison of Hardwired Failover and Network Failover Features

Network Failover

Network failover is based on heartbeat detectionwhere the system sends heartbeat packets over the internal network.

The system uses the primary and secondary failover addresses to send network failover heartbeat packets. For more information about the BIG-IP mirroring and network failover transport protocols, refer to the following articles:

Service Port and Protocol Used for BIG-IP Network Failover
Transport Protocol Used for BIG-IP Connection and Persistence Mirroring

The BIG-IP system considers the peer down after the Failover.NetTimeoutSec timeout value is exceeded. The default value of Failover.NetTimeoutSec is three seconds, after which the standby unit attempts to switch to an active state. The following database entry represents the default settings for the failover time configuration:

Failover.NetTimeoutSec = 3

Device Service Clustering (DSC) was introduced in BIG-IP 11.0.0 and allows many new features such as synchronization and failover between two or more devices. Network failover provides communication between devices for synchronization, failover, and mirroring and is required for the following deployments:

- Sync-Failover device groups containing three or more devices
- Active-active configurations between two BIG-IP platforms
- BIG-IP VIPRION platforms
- BIG-IP Virtual Edition

An active-active pair must communicate over the network to indicate the objects and resources they service. Otherwise, if network communications fail, the two systems may attempt to service the same traffic management objects, which could result in duplicate IP addresses on the network.

A broken network may cause BIG-IP systems to enter into active-active mode. To avoid this issue, F5 recommends that you dedicate one interface on each system to perform only failover communications and, when possible, directly connect these two interfaces with an Ethernet cable to avoid network problems that could cause the systems to go into an active-active state.

> **Important:** When you directly connect two BIG-IP systems with an Ethernet cable, do not change the speed and duplex settings of the interfaces involved in the connection. If you do, depending on the BIG-IP software version, you may be required to use a crossover cable. For more information, refer to SOL9787: Auto MDI/MDIX behavior for BIG-IP platforms.

If you configure a BIG-IP high-availability pair to use network failover, and the hardwired failover cable also connects the two units, hardwired failover always has precedence; if network failover traffic is compromised, the two units do not fail over because the hardwired failover cable still connects them.

Hardwired Failover

Hardwired failover is also based on heartbeat detection, where one BIG-IP system continuously sends voltage to another. If a response does not initiate from one BIG-IP system, failover to the peer occurs in less than one second. When BIG-IP redundant devices connect using a hardwired failover cable, the system automatically enables hardwired failover.

The maximum hardwired cable length is 50 feet. Network failover is an option if the distance between two BIG-IP systems exceeds the acceptable length for a hardwired failover cable.

> **Note:** For information about the failover cable wiring pinouts, refer to Pinouts for the Failover Cable used with BIG-IP Platforms.

Hardwired failover can only successfully be deployed between two physical devices. In this deployment, hardwired failover can provide faster failover response times than network failover. However, peer state may be reported incorrectly when using hardwired failover alone.

Hardwired failover is only a heartbeat and carries no status information. Communication over the network is necessary for certain features to function properly. For example, Traffic Management Microkernel (TMM) uses the network to synchronize packets and flow state updates to peers for connection mirroring. To enable proper state reporting and mirroring, F5 recommends that you configure network failover in addition to hardwired failover.

2.05 - Explain use cases for MAC masquerading

Configuring MAC Masquerade (11.x)

MAC Masquerading

Using MAC masquerading will reduce ARP convergence issues within the BIG-IP LAN environments when a failover event happens.

To optimize the flow of traffic during failover events, you can configure MAC masquerade addresses for any defined traffic groups on the BIG-IP system. A MAC masquerade address is a unique, floating MAC address that you create. You can assign one MAC masquerade address to each traffic group on a BIG-IP device. By assigning a MAC masquerade address to a traffic group, you associate that address with any floating IP addresses associated with the traffic group. By configuring a MAC masquerade address for each traffic group, a single Virtual Local Area Network (VLAN) can potentially carry traffic and services for multiple traffic groups, with each service having its own MAC masquerade address.

2.05 - Determine when it is appropriate to use more than two LTM devices in a device service cluster

BIG-IP Redundant Systems Configuration Guide

Device Service Cluster

The TMOS within the BIG-IP system includes an underlying architecture that makes it possible for you to create a redundant system configuration, known as device service clustering (DSC), for multiple BIG-IP devices on a network. This redundant system architecture provides both synchronization of BIG-IP configuration data and high availability at user-defined levels of granularity. More specifically, you can configure a BIG-IP device on a network to:

- Synchronize some or all of its configuration data among any number of BIG-IP devices on a network
- Fail over to one of many available devices
- Mirror connections to a peer device to prevent interruption in service during failover

If you have two BIG-IP devices only, you can create either an active/standby or an active-active configuration. With more than two devices, you can create a configuration in which multiple devices are active and can fail over to one of many, if necessary.

Device Service Clusters are intended for environments that need to run multiple active BIG-IP units to handle capacity and still maintain the ability to fail traffic to additional BIG-IP resources.

By setting up a redundant system configuration, you ensure that BIG-IP configuration objects are synchronized and can fail over at useful levels of granularity to appropriate BIG-IP devices on the network. You also ensure that failover from one device to another, when enabled, occurs seamlessly, with minimal interruption in application delivery.

2.05 - Explain the functionality of HA groups

Understanding Fast Failover

HA Groups

The BIG-IP system includes a feature known as fast failover. Fast failover is a feature that is based on the concept of an HA group. An HA group is a set of trunks, pools, or clusters (or any combination of these) that you want the BIG-IP system to use to calculate an overall health score for a device in a redundant system configuration. A health score is based on the number of members that are currently available for any trunks, pools, and clusters in the HA group, combined with a weight that you assign to each trunk, pool, and cluster. The device that has the best overall score at any given time becomes or remains the active device.

> **Note:** To use the fast failover feature, you must first create a redundant system configuration. The fast failover feature is designed for a redundant configuration that contains a maximum of two devices in a device group, with one active traffic group.
>
> **Note:** Only VIPRION systems can have a cluster as an object in an HA group. For all other platforms, HA group members consist of pools and trunks only.

An HA group is typically configured to fail over based on trunk health in particular. Trunk configurations are not synchronized between units, which means that the number of trunk members on the two units often differs whenever a trunk loses or gains members. The HA group feature makes it possible for failover to occur based on changes to trunk health instead of on system or VLAN failure.

Only one HA group can exist on the BIG-IP system. By default, the HA group feature is disabled.

To summarize, when you configure the HA group, the process of one BIG-IP device failing over to the other based on HA scores is noticeably faster than if failover occurs due to a hardware or daemon failure.

2.05 - Compare and contrast failover unicast and multicast

BIG-IP Redundant Systems Configuration Guide

Failover Unicast and Multicast

The unicast failover configuration uses a self-IP address and TMM switch port to communicate failover packets between each BIG-IP appliance. For appliance platforms, specifying two unicast addresses should suffice.

For VIPRION platforms, you should enable multicast and retain the default multicast address that the BIG-IP system provides. The multicast failover entry uses the management port to communicate failover packets between each VIPRION system. As an alternative to configuring the multicast failover option, you can define a unicast mesh using the management port for each VIPRION system.

Objective - 2.06 Given a scenario, describe the steps necessary to set up a device group, traffic group and HA group

2.06 - Explain how to set up sync-only and sync-failover device service cluster

BIG-IP Redundant Systems Configuration Guide

Working With Sync-Failover Device Groups

One of the types of device groups that you can create is a Sync-Failover type of device group. A Sync-Failover device group contains devices that synchronize configuration data and fail over to one another when a device becomes unavailable. A maximum of eight devices is supported in a Sync-Failover device group.

A device in a trust domain can belong to one Sync-Failover device group only.

For devices in this type of device group, the BIG-IP system uses both the device group and the traffic group attributes of a folder to make decisions about which devices to target for synchronizing the contents of the folder, and which objects to include in failover.

In the simplest configuration, you can use the BIG-IP Configuration utility to:

1. Create a Sync-Failover device group containing all of local BIG-IP devices.
2. Assign the device group to the root folder as the default device group.
3. Assign the default traffic group, traffic-group-1, to the root folder as the default traffic group.

The result is that all folders inherit the default device group and the default traffic group as their device group and traffic group attribute values, causing all BIG-IP configuration data on a BIG-IP device to be synchronized to all devices in that device group, and the objects in traffic-group-1 to fail over to another member of the device group when a device becomes unavailable.

Creating a Sync-Failover device group

This task establishes failover capability between two or more BIG-IP devices. If the active device in a Sync-Failover device group becomes unavailable, the configuration objects fail over to another member of the device group and traffic processing is unaffected. You can perform this task on any authority device within the local trust domain.

1. On the Main tab, click Device Management > Device Groups. The Device Groups screen displays a list of existing device groups.
2. On the Device Group List screen, click Create.
3. Type a name for the device group, select the device group type Sync-Failover, and type a description for the device group.
4. In the Configuration area of the screen, select a host name from the Available list for each BIG-IP device that you want to include in the device group. Use the Move button to move the host name to the Selected list.

   The Available list shows any devices that are members of the device's local trust domain but not currently members of a Sync-Failover device group. A device can be a member of one Sync-Failover group only.
5. For Network Failover, select the Enabled check box.
6. Click Finished.

You now have a Sync-Failover type of device group containing BIG-IP devices as members.

Configuring failover settings on a device group

You use this procedure to configure some failover settings for a specific device group.

1. On the Main tab, click Device Management > Device Groups. The Device Groups screen displays a list of existing device groups.
2. In the Group Name column, click the name of a device group.
3. On the menu bar, click Failover.
4. In the Link Down Time on Failover field, use the default value of 0.0, or specify a new value.

   This setting specifies the amount of time in seconds that interfaces for any VLANs on external devices are down when a traffic group fails over and goes to the standby state. Specifying a value other than 0.0 for this setting causes other vendor switches to use the specified time to learn the MAC address of the new active device.

Also stated well in SOL5241 as follows:

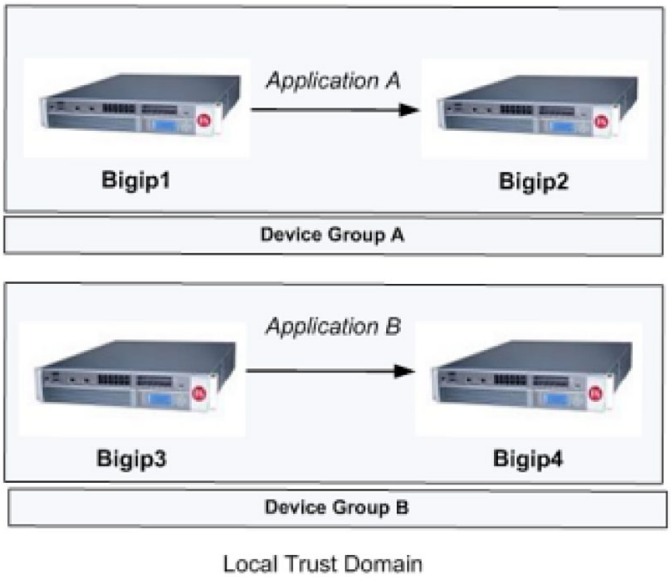
The Link Down Time on Failover setting is used in environments where the upstream devices' ARP cache restricts updates until a specific period of time has passed or if the device on the interface is no longer available. When the Link Down Time on Failover setting is enabled and a failover occurs, the failing unit will mark any interfaces for the external VLAN as DOWN. As a result, the upstream device will flush its ARP cache and allow the newly active unit's ARP information to be accepted by the upstream device.

5. Click Save Changes.

Sample Sync-Failover configuration

You can use a Sync-Failover device group in a variety of ways. This sample configuration shows two separate Sync-Failover device groups in the local trust domain. Device group A is a standard active/standby configuration. Only BIG-IP1 normally processes traffic for application A. This means that BIG-IP1 and BIG-IP2 synchronize their configurations, and BIG-IP1 fails over to BIG-IP2 if BIG-IP1 becomes unavailable. BIG-IP1 cannot fail over to BIG-IP3 or BIG-IP4 because those devices are in a separate device group.

Device group B is also a standard active/standby configuration, in which BIG-IP3 normally processes traffic for application B. This means that BIG-IP3 and BIG-IP4 synchronize their configurations, and BIG-IP3 fails over to BIG-IP4 if BIG-IP3 becomes unavailable. BIG-IP3 cannot fail over to BIG-IP1 or BIG-IP2 because those devices are in a separate device group.



Example illustration of a Sync-Failover device group

Working with Sync-Only device groups

One of the types of device groups that you can create is a Sync-Only device group. A Sync-Only device group contains devices that synchronize configuration data with one another, but their configuration data does not fail over to other members of the device group. A maximum of 32 devices is supported in a Sync-Only device group.

A device in a trust domain can be a member of more than one Sync-Only device group. A device can also be a member of both a Sync-Failover group and a Sync-Only group.

A typical use of a Sync-Only device group is one in which you configure a device to synchronize the contents of a specific folder to a different device group than to the device group to which the other folders are synchronized.

Creating a Sync-Only device group

Follow these steps to create a Sync-Only type of device group. You can perform this task on any BIG-IP device within the local trust domain.

1. On the Main tab, click Device Management > Device Groups. The Device Groups screen displays a list of existing device groups.
2. On the Device Group List screen, click Create.
3. Type a name for the device group, select the device group type Sync-Only, and type a description for the device group.
4. Select an IP address and host name from the Available list for each BIG-IP device that you want to include in the device group. Use the Move button to move the host name to the Includes list.

   The list shows any devices that are members of the device's local trust domain.
5. For Automatic Sync, select the Enabled check box.
6. Click Finished.

You now have a Sync-Only type of device group containing BIG-IP devices as members.

Enabling and disabling Automatic Sync

For Sync-Only device groups, you can choose to either automatically or manually synchronize configuration data in a device group.

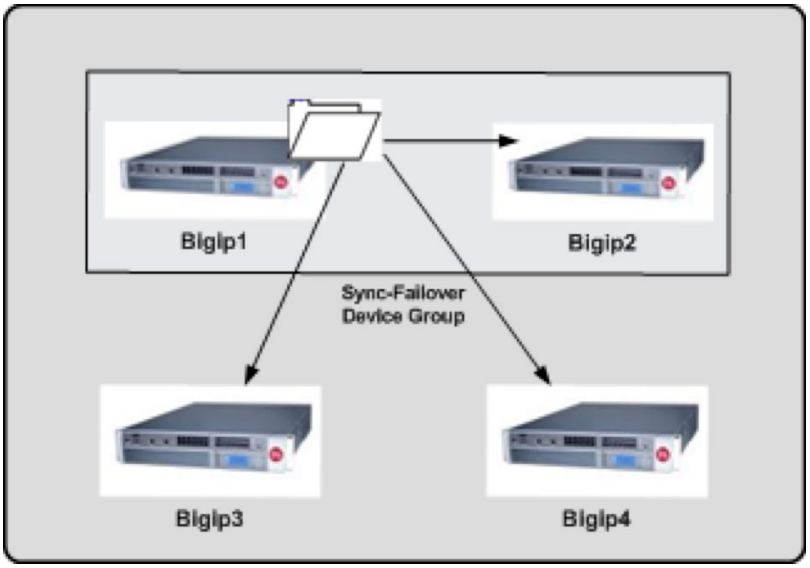
> **Note:** For Sync-Failover device groups, the BIG-IP system supports manual synchronization only.

You can use the BIG-IP Configuration utility to enable or disable automatic synchronization. When enabled, this feature causes any BIG-IP device in the device group to synchronize its configuration data to the other members of the device group whenever that data changes.

1. On the Main tab, click Device Management > Device Groups. The Device Groups screen displays a list of existing device groups.
2. In the Group Name column, click the name of the relevant device group.
3. On the menu bar, click ConfigSync.
4. For Automatic Sync, clear or select the Enabled check box.
5. Click Update.

Sample Sync-Only configuration

The most common reason to use a Sync-Only device group is to synchronize a specific folder containing policy data that you want to share across all BIG-IP devices in a local trust domain, while setting up a Sync-Failover device group to fail over the remaining configuration objects to a subset of devices in the domain. In this configuration, you are using a Sync-Only device group attribute on the policy folder to override the inherited Sync-Failover device group attribute. Note that in this configuration, BIG-IP1 and BIG-IP2 are members of both the Sync-Only and the Sync-Failover groups.



Sync-Only Device Group

To implement this configuration, follow this process.

1. Create a Sync-Only device group on the local device, adding all devices in the local trust domain as members.
2. Create a Sync-Failover device group on the local device, adding a subset of devices as members.
3. On the folder containing the policy data, use tmsh to set the value of the device group attribute to the name of the Sync-Only device group.
4. On the root folder, retain the default Sync-Failover device group assignment.

This section uses GUI level instructions to explain the setup and configuration of HA functionalities. You should be very comfortable with how the configuration looks in the CLI as well through hands-on work in your vLabs.

2.06 - Explain how to configure HA groups

BIG-IP Redundant Systems Configuration Guide

HA Group Configuration

To configure the BIG-IP system so that failover can occur based on an HA score, you must specify values for the properties of an HA group. The system makes it possible for you to configure one HA group only; you cannot create additional HA groups. Once you have configured HA group properties, the BIG-IP system uses that configuration to calculate an overall HA score for each device in the redundant system configuration.

1. On the Main tab, click System > High Availability.
2. On the menu bar, click HA Group.
3. In the HA Group Properties area of the screen, in the HA Group Name field, type a name for the HA group.
4. Verify that the Enable check box is selected.
5. In the Active Bonus field, specify an integer the represents the amount by which you want the system to increase the overall score of the active device. The purpose of the active bonus is to prevent failover when minor or frequent changes occur to the configuration of a pool, trunk, or cluster.
6. For the Pools setting, in the Available box, click a pool name and use the Move button to move the pool name to the Selected box. This populates the table that appears along the bottom of the screen with information about the pool.

7. For the Trunks setting, in the Available box, click a trunk name and use the Move button to move the trunk name to the Selected box. This populates the table that appears along the bottom of the screen with information about the trunk.
8. For the Clusters setting (VIPRION platforms only), in the Available box, click a cluster name and use the Move button to move the cluster name to the Selected box.
9. In the table displayed along the bottom of the screen, for the Threshold setting, for each pool or trunk in the HA group, optionally specify an integer for a threshold value.
10. For the Weight setting, for each pool or trunk in the HA group, specify an integer for the weight. The allowed weight for an HA group object ranges from 10 through 100. This value is required.
11. Click Create.

You now have an HA group that the BIG-IP system can use to calculate an HA score for failover. This section uses GUI level instructions to explain the setup and configuration of HA functionalities. You should be very comfortable with how the configuration looks in the CLI as well through hands-on work in your vLabs.

2.06 - Explain how to assign virtual servers to traffic groups

Traffic Group Assignment

When a virtual server is created it will inherit the default traffic group of the partition/path that it is created in. However there can be more than one traffic group in a partition.

If you want to move the virtual server object to a different traffic group:

1. On the Main tab, click Local Traffic > Virtual Server > Virtual Address List.
2. Click on the address of the virtual server you wan to modify.
3. In the Traffic Group field change the selection box to the Traffic Group you wish to use.
4. Click Update.

Objective - 2.07 Predict the behavior of an LTM device group or traffic groups in a given failure scenario

2.07 Predict the behavior of an LTM device group or traffic groups in a given failure scenario

Troubleshooting ConfigSync and Device Service Clustering Issues (11.x)

Scenario Based Questions

To prepare for scenario based questions the candidate will need to complete hands-on configuration and testing of the configuration on the LTM. This will allow the candidate to better understand how different configurations can produce different results. All F5 exams use scenario-based questions that make the candidate apply what they know to a situation to determine the resulting outcome.

This topic is focused on predicting behaviors during failovers between BIG-IP systems. Understanding how device groups and traffic groups behave is the key to this topic. Experience with failing over HA systems will give the candidate the ability to answer the questions on this topic.

F5 introduced the Device Service Clustering (DSC) architecture in BIG-IP 11.x. DSC provides the framework for ConfigSync, and other high-availability features, including the following components:

Device trust and trust domains

Device trust establishes trust relationships between BIG-IP devices through certificate-based authentication. Each device generates a device ID key and Secure Socket Layer (SSL) certificate upon upgrade or installation. A trust domain is a collection of BIG-IP devices that trust each other, and can synchronize and fail over their BIG-IP configuration data, as well as regularly exchange status and failover messages.

When the local BIG-IP device attempts to join a device trust with a remote BIG-IP device, the following applies:

If the local BIG-IP device is added as a peer authority device, the remote BIG-IP device presents a certificate signing request (CSR) to the local device, which then signs the CSR and returns the certificate along with its CA certificate and key.

If the local BIG-IP device is added as a subordinate (non-authority) device, the remote BIG-IP device presents a CSR to the local device, which then signs the CSR and returns the certificate. The CA certificate and key are not presented to the remote BIG-IP device. The subordinate device is unable to request other devices to join the device trust.

Device groups

A device group is a collection of BIG-IP devices that reside in the same trust domain and are configured to securely synchronize their BIG-IP configuration and failover when needed. Device groups can initiate a ConfigSync operation from the device group member with the desired configuration change. You can create two types of device groups:

A Sync-Failover device group contains devices that synchronize configuration data and support traffic groups for failover purposes.

A Sync-Only device group contains devices that synchronize configuration data, but do not synchronize failover objects and do not fail over to other members of the device group.

Traffic groups

A traffic group represents a collection of related configuration objects that are configured on a BIG-IP device. When a BIG-IP device becomes unavailable, a traffic group can float to another device in a device group.

Folders

A folder is a container for BIG-IP configuration objects. You can use folders to set up synchronization and failover of configuration data in a device group. You can sync all configuration data on a BIG-IP device, or you can sync and fail over objects within a specific folder only.

Centralized Management Infrastructure (CMI) communication channel

The BIG-IP system uses SSL certificates to establish a trust relationship between devices. In a device trust, BIG-IP devices can act as certificate signing authorities, peer authorities, or subordinate non-authorities. When acting as a certificate signing authority, the BIG-IP device signs x509 certificates for another BIG-IP device that is in the local trust domain. The BIG-IP device for which a certificate signing authority device signs its certificate is known as a subordinate non-authority device. The BIG-IP system uses the following certificates to establish a secure communication channel:

Objective - 2.08 Determine the effect of LTM features and/or modules on LTM device performance and/or memory

2.08 - Determine the effect of iRules on performance

iRules Optimization 101 - #05 - Evaluating iRule Performance

Effect of iRules on Performance

This is a classic case of “It Depends”. Since iRules are written individually to solve specific issues or do specific functions necessary for a particular scenario, there is not a fixed sheet of performance numbers showing how an iRule will impact performance. iRules do get compiled into byte code, and can run at wire speed, but it really depends on what you’re doing. Many times there is more than one way to write an iRule and one method may work more efficiently than another.

That said there are ways to see how an iRule is performing by collecting and interpreting runtime statistics by inserting a timing command into event declarations to see over all CPU usage when under load. This tool will help you to create an iRule that is performing the best on your system.

Collecting Statistics

To generate & collect runtime statistics, you can insert the command “timing on” into your iRule. When you run traffic through your iRule with timing enabled, LTM will keep track of how many CPU cycles are spent evaluating each iRule event. You can enable rule timing for the entire iRule, or only for specific events.

To enable timing for the entire iRule, insert the “timing on” command at the top of the rule before the first “when EVENT_NAME” clause.

With the timing command in place, each time the rule is evaluated, LTM will collect the timing information for the requested events.

To get a decent average for each of the events, you’ll want to run at least a couple thousand iterations of the iRule under the anticipated production load.

Viewing Statistics

The statistics for your iRule (as measured in CPU cycles) may be viewed at the command line or console by running

```
tmsh show ltm rule rule_name all
```

The output includes totals for executions, failures & aborts along with minimum, average & maximum cycles consumed for each event since stats were last cleared.

```
------------------------------------------------

Ltm::Rule rule_name

------------------------------------------------

Executions

Total       729

Failures      0

Aborts        0

CPU Cycles on Executing

Average    3959

Maximum   53936

Minimum    3693
```

Evaluating statistics

"Average cycles reported" is the most useful metric of real-world performance, assuming a large representative load sample was evaluated.

The "maximum cycles reported" is often very large since it includes some one-time and periodic system overhead. (More on that below.)

Here's a spreadsheet (iRules Runtime Calculator) that will calculate percentage of CPU load per iteration once you populate it with your clock speed and the statistics gathered with the "timing" command. (Clock speed can be found by running 'cat /proc/cpuinfo' at the command line.)

Caveats

Timing is intended to be used only as an optimization/debug tool, and does have a small impact on performance; so don't leave it turned on indefinitely.

Timing functionality seems to exhibit a 70 - 100 cycle margin of error.

Use average cycles for most analyses. Maximum cycles is not always an accurate indicator of actual iRule performance, as the very first call a newly edited iRule includes the cycles consumed for compile-time

optimizations, which will be reflected in an inflated maximum cycles value. The simple solution to this is to wait until the first time the rule is hit, then reset the statistics.

However, maximum cycles is also somewhat inflated by OS scheduling overhead incurred at least once per tick, so the max value is often overstated even if stats are cleared after compilation.

Constructing CMP-compatible iRules

Global Variable Impact

iRules use global variables to make variable data that is created in one context, that is available to other connections, virtual servers, and Traffic Management Microkernel (TMM) instances. If a virtual server references an iRule that uses a global variable that is not Clustered Multiprocessing (CMP) compatible, the virtual server will be ineligible for CMP processing. In most cases, it is good to retain the benefits of CMP processing when using iRules. This document expands on the various ways to represent global variable data, making it available to other connections, other virtual servers, and other TMM instances.

In many cases, variable data used in an iRule is required to be available only within the scope of the current connection. The use of Tcl local variables satisfies this requirement, and does not affect CMP compatibility.

In other cases, variable data must be available globally, that is, outside the context of a connection. The most common requirement people have is to capture data from one connection, then to reference that data from subsequent connections that are part of the same session. This requirement can be further refined to include both multiple connections traversing the same TMM instance, such as would be seen on a non-CMP-enabled system or virtual server, and also multiple related connections on CMP-enabled virtual servers, which may traverse different TMM instances.

Another common use for global variables is to share data among multiple iRules that run on the same BIG-IP system. For example, to set and enforce a cumulative concurrent connection limit, an iRule would need to both set a globally accessible limit value, and also allow each iRule instance to update a separate globally-accessible counter value.

The use of global variables can force the BIG-IP system to automatically disable CMP processing, which is known as demotion. Demotion of a virtual server limits processing of that virtual server to only one CPU core. This can adversely affect performance on multi-core BIG-IP systems, as only a fraction of the available CPU resources are available for each demoted virtual server. In addition, CMP demotion can create an internal communication bottleneck for virtual servers that are WebAccelerator-enabled or ASM-enabled.

The following sections explain each of three popular methods for sharing iRules-derived data globally, including the CMP compatibility of each method.

Using Tcl global variables

Tcl global variables are not actually global on a CMP-enabled BIG-IP system, since the global variables are not shared among TMM instances. Tcl global variables are accessible globally only within the local TMM instance (meaning that each TMM instance would need to set and update separately its own copy of the variable and the value of the variable). As a result, the TMM process running on one processor is not able to access the contents of the same Tcl global variable that was set by a different TMM process, even if both TMM processes are handling connections for the same virtual server. Because of this limitation, the use of a Tcl global variable in an iRule automatically demotes from CMP any virtual server to which it is applied. This avoids the confusion that would otherwise result from accessing and updating multiple instances of the same "global" variable. Because the virtual server will be automatically demoted from CMP, you should restrict the use of Tcl global variables to iRules that will be applied to virtual servers that do not depend on CMP processing.

Using static global variables

If you must share static data (data that will never be modified by the iRule itself) across CMP-enabled virtual servers, you can use a static global variable. A static global variable stores data globally to the entire BIG-IP system, and is set within each TMM instance each time the iRule is initialized. The value of a static global variable is assumed not to change unless the iRule is re-initialized. As a result, static global variables must be set within the RULE_INIT event. Static global variables set within the RULE_INIT event are propagated to all TMM instances each time the iRule is initialized: when the iRule is loaded at system startup, when the configuration is re-loaded, or when the iRule is modified from within the BIG-IP Configuration utility and saved.

> **Important:** While it is possible to use the set command to modify a static global variable within the iRule and outside of the RULE_INIT event, such modifications will not be propagated to each TMM instance; they will be visible to only the TMM process on which the modification was made, resulting in inconsistent values for the static global variable across TMM instances. As a result, F5 strongly recommends that you do not update the value of any static global variable within the iRule.

Using the session table to store global variables

If you must share non-static global data across CMP-enabled virtual servers, you can use the session table to store and reference the data. Session table data is shared among all TMM instances. Using the session table imposes considerable operational overhead, but the preservation of CMP processing for the virtual server typically far outweighs any such impact.

You can use the table command to manipulate the session table. For details, refer to the DevCentral article linked in the Supplemental Information section below.

Recommendations

As you can see, there are several different options for using global variables, or the equivalent functionality, in session tables. Each of these options has advantages and disadvantages in their use. Typically these decisions are made on performance and ease of implementation.

In summary:

Tcl global variables

You should restrict the use of Tcl global variables to iRules that will be applied to virtual servers that do not depend on CMP processing.

Static global variables

The use of static global variables is recommended for sharing static data (data that will not be updated by any iRule) among TMM instances that are used by CMP-enabled virtual servers, or for sharing static data among multiple iRules without affecting the CMP status of any virtual server to which it is applied.

Session table

The use of the session table is recommended for sharing dynamic global variable data (data that will be updated within the iRule) among CMP-enabled virtual servers.

2.08 - Determine the effect of RAM cache on performance and memory

Understanding RAM Caching

Effect of RAM Cache on Performance

The largest effect of using the RAM Cache feature on the BIG-IP system is system memory utilization. There is a finite amount of RAM in every system and using any amount of that RAM for caching HTTP objects can impact performance and even limit provisioning additional licensing options.

RAM Cache

A RAM Cache is a cache of HTTP objects stored in the BIG-IP system's RAM that are reused by subsequent connections to reduce the amount of load on the back-end servers.

When to use the RAM Cache

The RAM Cache feature provides the ability to reduce the traffic load to back-end servers. This ability is useful if an object on a site is under high demand, if the site has a large quantity of static content, or if the objects on the site are compressed.

- High demand objects

  This feature is useful if a site has periods of high demand for specific content. With RAM Cache configured, the content server only has to serve the content to the BIG-IP system once per expiration period.

- Static content

  This feature is also useful if a site consists of a large quantity of static content such as CSS, javascript, or images and logos.

- Content compression

  For compressible data, the RAM Cache can store data for clients that can accept compressed data. When used in conjunction with the compression feature on the BIG-IP system, the RAM Cache takes stress off of the BIG-IP system and the content servers.

Items you can cache

The RAM Cache feature is fully compliant with the cache specifications described in RFC 2616, Hypertext Transfer Protocol -- HTTP/1.1. This means you can configure RAM Cache to cache the following content types:

- 200, 203, 206, 300, 301, and 410 responses
- Responses to GET methods by default.
- Other HTTP methods for URIs specified in the URI Include list or specified in an iRule.
- Content based on the User-Agent and Accept-Encoding values. The RAM Cache holds different content for Vary headers.

The items that the RAM Cache does not cache are:

- Private data specified by cache control headers
- By default, the RAM Cache does not cache HEAD, PUT, DELETE, TRACE, and CONNECT methods.

Understanding the RAM Cache mechanism

The default RAM Cache configuration caches only the HTTP GET methods. You can use the RAM Cache to cache both the GET and other methods, including non-HTTP methods, by specifying a URI in the URI Include list or writing an iRule.

2.08 - Determine the effect of compression on performance

The HTTP Compression Profile Type

Effect of Compression on Performance

The function of data compression is highly CPU intensive. The largest effect of using the RAM Cache feature on the BIG-IP system is system memory utilization. There is a finite amount of RAM in every system and using any amount of that RAM for caching HTTP objects can impact performance and even limit provisioning additional licensing options.

HTTP Compression

An optional feature is the BIG-IP systems ability to off-load HTTP compression tasks from the target server. All of the tasks needed to configure HTTP compression in Local Traffic Manager, as well as the compression software itself, are centralized on the BIG-IP system.

gzip compression levels

A gzip compression level defines the extent to which data is compressed, as well as the compression rate. You can set the gzip level in the range of 1 through 9. The higher the gzip level, the better the quality of the compression, and therefore the more resources the system must use to reach that specified quality. Setting a gzip level yields these results:

- A lower number causes data to be less compressed but at a higher performance rate. Thus, a value of 1 causes the least compression but the fastest performance.
- A higher number causes data to be more compressed but at a slower performance rate. Thus, a value of 9 (the highest possible value) causes the most compression, but the slowest performance.

  Warning: Selecting any value other than 1 - Least Compression (Fastest) can degrade system performance.

For example, you might set the gzip compression level to 9 if you are utilizing Local Traffic Manager cache feature to store response data. The reason for this is that the stored data in the cache is continually re-used in responses, and therefore you want the quality of the compression of that data to be very high.

As the traffic flow on the BIG-IP system increases, the system automatically decreases the compression quality from the gzip compression level that you set in the profile. When the gzip compression level decreases to the point where the hardware compression provider is capable of providing the specified compression level, the system uses the hardware compression providers rather than the software compression providers to compress the HTTP server responses.

Tip: You can change the way that Local Traffic Manager uses gzip levels to compress data by configuring the compression strategy. The compression strategy determines the particular compression provider (hardware or software) that the system uses for HTTP responses. The available strategies are: Speed (the default strategy), Size, Ratio, and Adaptive.

Memory levels for gzip compression

You can define the number of kilobytes of memory that Local Traffic Manager uses to compress data when using the gzip or deflate compression method. The memory level is a power-of-2 integer, in bytes, ranging from 1 to 256.

Generally, a higher value causes Local Traffic Manager to use more memory, but results in a faster and higher compression ratio. Conversely, a lower value causes Local Traffic Manager to use less memory, but results in a slower and lower compression ratio.

2.08 - Determine the effect of modules on performance and memory

Resource Provisioning

Effect of Modules On Performance

Enabling additional software on any F5 hardware platform will increase the utilization of the hardware resources of the unit. As you provision the software modules in TMOS the Resource Provisioning screen will show the administrator how much CPU, Disk and Memory is being used by each module. And if provisioning an additional module requires more resources than are available on the system, the system will not allow the provisioning of the module.

Resource Provisioning is a management feature to help support the installation and configuration of many modules available with BIG-IP. Provisioning gives you some control over the resources, both CPU and RAM, which are allocated to each licensed module. You may want, for example, to minimize the resources available to GTM on a system licensed for LTM and GTM. Since all models have some reliance on both management (Linux) and local traffic features, they will always be provisioned. Other modules must be manually provisioned. When you provision the modules, you can choose between four levels of resources. A fifth level may be allowed on certain modules. Dedicated, Nominal, Minimum and None are available for all modules and Lite is a fifth level available for trials only.

You can manage the provisioning of system memory, disk space, and CPU usage among licensed modules on the BIG-IP system.

There are five available resource allocation settings for modules.

None/Disabled

Specifies that a module is not provisioned. A module that is not provisioned does not run.

Dedicated

Specifies that the system allocates all CPU, memory, and disk resources to one module. When you select this option, the system sets all other modules to None (Disabled).

Nominal

Specifies that, when first enabled, a module gets the least amount of resources required. Then, after all modules are enabled, the module gets additional resources from the portion of remaining resources.

Minimum

Specifies that when the module is enabled, it gets the least amount of resources required. No additional resources are ever allocated to the module.

Lite

Lite is available for selected modules granting limited features for trials.

Provisioning licensed modules in BIG-IP through 11.2.1

1. Log in to the BIG-IP Configuration utility.
2. Click System.
3. Click Resource Provisioning.
4. In the Resource Provisioning (Licensed Modules) section, from the drop-down menu, select Minimum or Nominal for each licensed module.
5. After making the necessary provisioning changes, click System, click Configuration, click Device, and then click the Reboot button to restart the system.
6. When prompted, click OK to confirm the restart operation.

Objective - 2.09 Determine the effect of traffic flow on LTM device performance and/or utilization

2.09 - Explain how to use traffic groups to maximize capacity

Understanding Traffic Groups

Traffic Groups

A traffic group is a collection of related configuration objects that run on a BIG-IP device. Together, these objects process a particular type of traffic on that device. When a BIG-IP device becomes unavailable, a traffic group floats (that is, fails over) to another device in a device group to ensure that application traffic continues to be processed with little to no interruption in service. In general, a traffic group ensures that when a device becomes unavailable, all of the failover objects in the traffic group fail over to any one of the devices in the device group, based on the number of active traffic groups on each device.

A traffic group is initially active on the device on which you create it, until the traffic group fails over to another device. For example, if you initially create three traffic groups on Device A, these traffic groups remain active on Device A until one or more traffic groups fail over to another device. If you want to balance the traffic group load among all devices in the device group, you can intentionally cause a traffic group to fail over to another device. You do this using the Force to Standby option of the Configuration utility.

> **Important:** Although a specific traffic group can be active on only one device in a device group, the traffic group actually resides and is in a standby state on all other device group members, due to configuration synchronization.

Only certain types of configuration objects can belong to a traffic group. Examples of traffic group objects are self IP addresses and virtual IP addresses.

An example of a set of objects in a traffic group is an iApps application service. If a device with this traffic group is a member of a device group, and the device becomes unavailable, the traffic group floats to another member of the device group, and that member becomes the device that processes the application traffic.

When a traffic group fails over to another device in the device group, the device that the system selects is normally the device with the least number of active traffic groups. When you initially create the traffic group on a device, however, you specify the device in the group that you prefer that traffic group to run on in the event that the available devices have an equal number of active traffic groups (that is, no device has fewer active traffic groups than another). Note that, in general, the system considers the most available device in a device group to be the device that contains the fewest active traffic groups at any given time.

Maximizing Capacity

I could not find any specific documentation on maximizing capacity with Traffic Groups and the resource guide does not have any references to content.

I believe that they are trying to emphasize the fact that with the creations of traffic groups you can move the servicing of the traffic group resources from one device to another and thus balance loads or maximize capacity across devices.

Objective - 2.10 Determine the effect of virtual server settings on LTM device performance and/or utilization

2.10 - Determine the effect of connection mirroring on performance

Overview of Connection and Persistence Mirroring (11.x)

Connection Mirroring Performance Implications

The connection and persistence mirroring feature allows you to configure a BIG-IP system to duplicate connection and persistence information to the standby unit of a redundant pair. This setting provides higher reliability, but might affect system performance.

The BIG-IP device service clustering (DSC) architecture allows you to create a redundant system configuration for multiple BIG-IP devices on a network. System redundancy includes the ability to mirror connection and persistence information to a peer device to prevent interruption in service during failover. Traffic Management Microkernel (TMM) manages the state mirroring mechanism, and connection and persistence data is synchronized to the standby unit with every packet or flow state update. The standby unit decapsulates the packets and adds them to the connection table.

This feature can add CPU overhead to the system and can also cause network congestion depending on the system configuration.

Recommendations

When configuring mirroring on the BIG-IP system, F5 recommends that you consider the following factors:

Note: Only FastL4 and SNAT connections are re-mirrored after failback.

- Enable connection and persistence mirroring when a BIG-IP failover would cause the user's session to be lost or significantly disrupted

For example, where long-term connections, such as FTP and Telnet, are good candidates for mirroring, mirroring short-term connections, such as HTTP and UDP, is not recommended as this causes a decrease in system performance. In addition, mirroring HTTP and UDP connections is typically not necessary, as those protocols allow for failure of individual requests without loss of the entire session.

- Configure a dedicated VLAN and dedicated interfaces to process mirroring traffic

  The TMM process manages the BIG-IP LTM state mirroring mechanism, and connection data is synchronized to the standby unit with every packet or flow state update. In some mirroring configurations, this behavior may generate a significant amount of traffic. Using a shared VLAN and shared interfaces for both mirroring and production traffic reduces the overall link capacity for either type of traffic. Due to high traffic volumes, production traffic and mirroring traffic may interfere, potentially causing latency in mirrored connections or interrupting the network mirror connection between the two BIG-IP devices. If the network mirror connection is interrupted, it can cause loss of mirror information and interfere with the ability of the peer device to take over connections in the event of a failover.

- Directly cable network mirroring interfaces

  You can directly cable network mirroring interfaces on the BIG-IP systems in the failover pair, and F5 highly recommends that you do this when configuring a dedicated VLAN for mirroring. Configuring the pair in this way removes the need to allocate additional ports on surrounding switches, and removes the possibility of switch failure and switch-induced latency. Interfaces used for mirroring should be dedicated to the mirroring VLAN. Tagged interfaces shared with other VLANs could become saturated by traffic on other VLANs.

- Configure both primary and secondary mirroring addresses

  This would allow an alternate mirroring path and ensure reliable mirroring in the event of equipment or cable failure.

Objective - 2.11 Describe how to deploy vCMP guests and how the resources are distributed

BIG-IP vCMP Hosts and Guests Configuration Options

VCMP

Virtual Clustered Multiprocessing (vCMP) is a BIG-IP feature that allows you to run multiple instances of the BIG-IP software (vCMP guests) on a single hardware platform.

Typically, the initial BIG-IP configuration involves setting a host name, configuring Virtual Local Area Networks (VLANs) and self IPs, setting up High Availability (HA), and then configuring traffic objects such as pools, virtual servers, secure network address translations (SNATs), and other options made available by various BIG-IP modules. The vCMP feature changes the initial configuration model with the vCMP host and vCMP guest relationship, and understanding where to configure these options will ensure that the vCMP systems function properly. The following statements provide a brief description of the vCMP host and vCMP guest:

- The vCMP host is the system-wide hypervisor that makes it possible for you to create, view, and manage all vCMP guests on the system. A vCMP host allocates system resources to vCMP guests as needed.
- A vCMP guest is an object that you create on the vCMP host system for the purpose of running one or more BIG-IP modules. A vCMP guest consists of a Traffic Management Operating System (TMOS) instance, plus one or more BIG-IP modules. Each vCMP guest has its own share of hardware resources that the vCMP host allocates to it, effectively making each vCMP guest function like a separate BIG-IP device.

2.11 - Identify platforms that support vCMP

vCMP Host and Compatible Guest Version Matrix

Version 11.2 Hardware Platforms for vCMP

This list of supported hardware for vCMP was a short list in TMOS version 11.2, which this exam is currently based on.

The following is a list of available vCMP hardware for TMOS version 11.2:

- VIPRION 2400 with B2100 series blades
- VIPRION 4400(J100) with B4200 series blades
- VIPRION 4400(J100)/4480(J102) with B4300 series blades

  Note: F5 introduced support for the VIPRION B4300 blade in the 4400(J100) or 4480(J102) 4-slot chassis in BIG-IP 11.2.0. Support for the VIPRION B4300 blade in the 4800(S100) 8-slot chassis was not available until BIG-IP 11.4.0.

2.11 - Identify the limitations of vCMP

BIG-IP vCMP Hosts and Guests Configuration Options

Limitations of vCMP

When configuring the vCMP feature, you should consider the following factors:

- Once you provision the vCMP feature, you cannot provision any BIG-IP modules, such as BIG-IP LTM, on the vCMP host. Moreover, if you have already provisioned any BIG-IP modules on the BIG-IP system before you provision the vCMP feature, those modules are de-provisioned when you provision the vCMP feature. This situation, in turn, interrupts any application traffic that the system is currently processing.

- When you are logged in to the vCMP host, do not attempt to configure BIG-IP module features, such as virtual servers, pools, and profiles. You should use the vCMP host only to create and manage vCMP guests and to perform Layer 2 (L2) network configuration. If you attempt to configure BIG-IP modules while you are logged in to the vCMP host, the system can produce unexpected results. Always log in to the relevant vCMP guest before you configure BIG-IP module features.

Redundancy considerations:

- The self IP addresses that you specify per vCMP guest for configuration synchronization (ConfigSync) and failover should be the self IP addresses configured on the vCMP guest (not the vCMP host). Similarly, the management IP address that you specify per vCMP guest for device trust and failover should be the cluster IP address of the vCMP guest.

- For Sync-Failover device groups, each device group member must run on a chassis separate from the other members. The maximum supported size of a Sync-Failover device group is eight members.

  Important: Device group members should be vCMP guests, not vCMP hosts. Configuring redundancy between or among vCMP hosts could produce unexpected results.

- When you initially log in to the system to configure a vCMP guest, you access the vCMP host using its management IP address.

- When performing resource-intensive actions, such as upgrading software or installing hotfixes for a vCMP guest, it is important to maintain resources on the vCMP host system.

2.11 - Describe the effect of licensing and/or provisioning on the vCMP host and vCMP guest

Introduction to the vCMP System

BIG-IP license considerations for vCMP

If vCMP was a purchased licensed component on the BIG-IP system, then you can provision the vCMP feature and create guests with one or more BIG-IP system modules provisioned. For example, if you purchased a BIG-IP 5200v licensed with LTM, GTM, ASM and vCMP you can provision vCMP on the BIG-IP 5200v and each guest has access to the LTM, GTM and ASM license.

Note the following considerations:

- Each guest inherits the license of the vCMP host.
- The host license must include all BIG-IP modules that are to be provisioned across all guest instances. Examples of BIG-IP modules are BIG-IP Local Traffic Manager and BIG-IP Global Traffic Manager.
- The license allows you to deploy the maximum number of guests that the platform allows.
- If the license includes the appliance mode feature, you cannot enable appliance mode for individual guests; when licensed, appliance mode applies to all guests and cannot be disabled.

You activate the BIG-IP system license when you initially set up the system.

vCMP provisioning

To enable the vCMP feature, you perform two levels of provisioning. First, you provision the vCMP feature as a whole. When you do this, the BIG-IP system, by default, dedicates most of the disk space to running the vCMP feature, and in the process, creates the host portion of the vCMP system. Second, once you have configured the host to create the guests, each guest administrator logs in to the relevant guest and provisions the required BIG-IP modules. In this way, each guest can run a different combination of modules. For example, one guest can run LTM only, while a second guest can run LTM and ASM.

> **Important:** Once you provision the vCMP feature, you cannot provision any BIG-IP modules, such as BIG-IP LTM, on the vCMP host. Moreover, if any BIG-IP modules are already provisioned on the system before you provision the vCMP feature, those modules are de-provisioned when you provision the vCMP feature. This, in turn, interrupts any application traffic currently being processed.
>
> **Note:** The reserved disk space protects against any possible resizing of the file system.

2.11 - Describe how to deploy vCMP guests

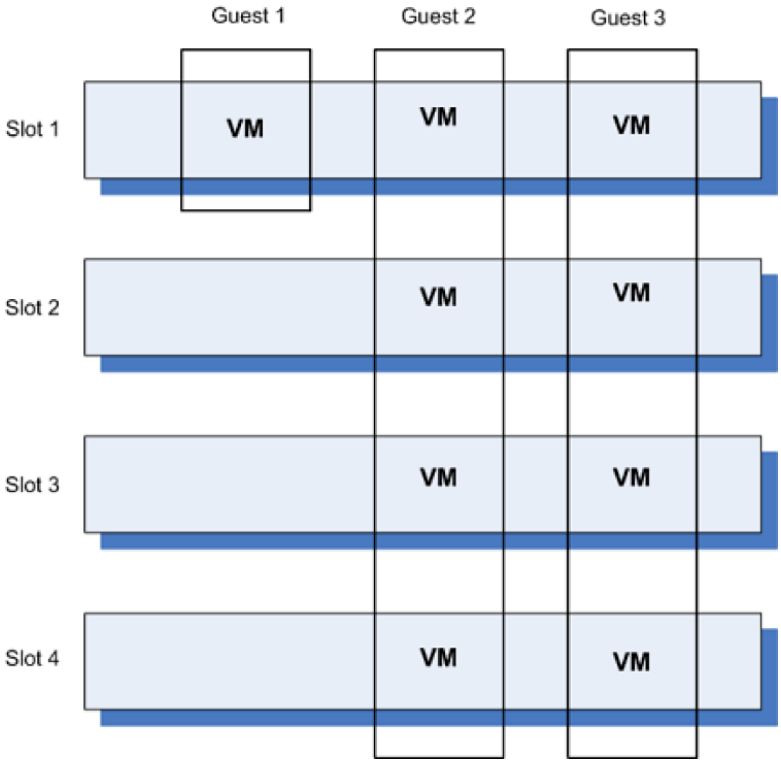
Initial vCMP Setup

Creating a vCMP guest

To create a vCMP guest, you need a VIPRION chassis system configured with a floating cluster management IP address, some base network objects such as trunks and VLANs, and you must license and provision the system to run the vCMP feature.

A guest can run on one available slot or all available slots of a chassis.

This illustration shows three guests running on a BIG-IP system. Guest 1 runs on a single slot only. Guest 2 and Guest 3 each run on all available slots.



You create a vCMP guest when you want to configure and run one or more BIG-IP modules as though the modules were running together on their own BIG-IP device. For example, you can create a guest that runs BIG-IP Local Traffic Manager and BIG-IP Global Traffic Manager.

Note: This procedure creates a guest in Bridged mode.

Note: When creating a guest, if you see an error message such as "Insufficient disk space on /shared/vmdisks. Need 24354M additional space.", you must delete existing unattached virtual disks until you have freed up that amount of disk space.

1. Use a browser to log in to the VIPRION chassis's management IP address. This logs you in to the floating IP address for the cluster.
2. On the Main tab, click vCMP > Guest List.
3. Click Create.
4. From the Properties list, select Advanced.
5. In the Name field, type a name for the guest.
6. In the Host Name field, type the host name of the BIG-IP system. Assign a fully-qualified domain name (FQDN). If you assign a name that is not an FQDN, the system might display an error message. If you leave this field blank, the system assigns the name localhost.localdomain.
7. From the Number of Slots list, select either Single Slot or All Slots. This causes the guest to reside on one slot or to span all slots. Note that once you configure a guest to span all slots, you cannot change this value later to Single Slot, unless you first change the state of the guest to Configured. Also note that if you decide to reconfigure an all slot guest to a single slot guest, you cannot specify on which available single slot the guest will reside.
8. From the Management Network list, select Bridged.
9. For the Cluster IP Address setting, fill in the required information:
10. In the IP Address field, type a unique management IP address that you want to assign to the guest. You use this IP address to access the guest when you want to manage a module running within the guest.
11. In the Network Mask field, type the network mask for the cluster IP address.
12. In the Management Route field, type a gateway address for the cluster IP address.
13. From the Initial Image list, select an ISO image file for installing TMOS software and the BIG-IP license onto the guest's virtual disk. The license associated with the selected image provides access to the correct BIG-IP modules.
14. In the Virtual Disk list, retain the default value of None. The BIG-IP system creates a virtual disk with a default name (the guest name plus the string .img, such as guestA.img). Note that if an unattached virtual disk file with that default name already exists, the system displays a message, and you must

manually attach the virtual disk. You can do this using the tmsh command line interface, or use the Configuration utility to view and select from a list of available unattached virtual disks.

15. For the VLAN List setting, select both an internal and an external VLAN name from the Available list, and use the Move button to move the VLAN names to the Selected list.
16. From the Requested State list, select Provisioned. This allocates all necessary resources to the guest, such as CPU cores, virtual disk, and so on.
17. Click Finish.

After clicking Finished, wait while the system installs the selected ISO image onto the guest's virtual disk. When this process is complete, you can deploy the guest.

> **Note:** You can also skip the Provisioned state and instead go straight to the Deployed state if you are confident of your guest configuration. Provisioning first and then deploying makes it more straightforward to make changes to the slots to which your guests are allocated if you find you need to make changes.

Setting a vCMP guest to the Deployed state

Until you deploy a vCMP guest, your vCMP VIPRION has no medium for provisioning and running the BIG-IP modules that you can use to process traffic.

1. Ensure that you are still logged in to the vCMP host using the BIG-IP system's cluster IP address.
2. On the Main tab, click vCMP > Guest List.
3. In the Name column, click the name of the vCMP guest that you want to deploy.
4. From the Requested State list, select either Provisioned or Deployed.
5. Click Update.

After moving a vCMP guest to the Deployed state, wait while the guest boots and becomes accessible. Then, you can log into the vCMP guest to provision specific BIG-IP modules.

Provisioning a BIG-IP module within a guest

Before you can access a guest to provision BIG-IP modules, the vCMP guest must be in the Deployed state.

You determine which BIG-IP modules run within a guest by provisioning the modules. For example, if you want guestA to run LTM and GTM, log into guestA and provision it with LTM and GTM. If you want guestB to run LTM and ASM, log into guestB and provision it with BIG-IP LTM and BIG-IP ASM. Bear in mind that guests

inherit the licenses of the vCMP host on which they were created, so any BIG-IP modules that you want to provision on a guest must be included in the license you installed with the vCMP host.

Note: This procedure applies to guests in Bridged mode only. Guests in isolated mode can be accessed only using vconsole and tmsh.

1. Use a browser and the management IP address that you configured for the guest to log in to the guest. If the system prompts you to run the Setup Utility, do not. Instead, complete this task to produce an initial configuration better suited for a vCMP guest. The BIG-IP Configuration utility opens so that you can configure the guest.
2. On the Main tab, click System > Resource Provisioning.
3. In the Resource Provisioning (Licensed Modules) area, from the Local Traffic (LTM) list, select Minimal, Nominal, or Dedicated, depending on your needs.
4. Click Update.

After provisioning the module from within the guest, create self IP addresses and assign a vCMP host VLAN to each one. The vCMP host VLANs that you assign to these self IP addresses are the VLANs you created before creating the guest.

Creating self IP addresses for VLANs

You need at least one VLAN or VLAN group configured before you create a self IP address.

Self IP addresses enable the BIG-IP system, and other devices on the network, to route application traffic through the associated VLAN or VLAN group. Repeat the steps in this task for each VLAN.

1. On the Main tab, click Network > Self IPs. The Self IPs screen opens.
2. Click Create. The New Self IP screen opens.
3. In the Name field, type a unique name that readily identifies the VLAN to which it will associate for the self IP. Name the self IP for the internal VLAN Internal, name the external VLAN External, and name the HA VLAN HA.
4. In the IP Address field, type an IP address. This IP address must be within the address space that corresponds to the VLAN for which it is created (Internal, External or HA). The system accepts IP addresses in both the IPv4 and IPv6 formats.
5. In the Netmask field, type the network mask for the specified IP address.

6. From the VLAN/Tunnel list, select the VLAN to associate with this self IP address:
    - For the internal network, select the VLAN that is associated with an internal interface or trunk.
    - For the external network, select the VLAN that is associated with an external interface or trunk.
    - For the HA network, select the VLAN that is associated with an internal interface or trunk.
7. From the Port Lockdown list, select Allow Default.
8. Repeat the last 4 steps, but this time specify an address from your external network in step 4 and select the VLAN named external in step 6.
9. Repeat steps 3 through 7 one more time, but this time specify an address on your internal network in step 4 and select the VLAN named HA in step 6.
10. Click Finished. The screen refreshes, and displays the new self IP address in the list.

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

Overview: Verifying initial vCMP configuration

Verifying your vCMP configuration confirms that the setup performed up to this point is functioning properly. Once you establish that the vCMP configuration is correct, you will likely need to create a profile, pools, and virtual server that are tailored to your network topology before your guest can begin processing LTM traffic.

2.11 - Explain how resources are assigned to vCMP guests (e.g., SSL, memory, CPU, disk)

Understanding vCMP Guests

The vCMP host (hypervisor) allocates hardware resources to each BIG-IP vCMP guest instance.

As you create the vCMP Guest you define the number of slots, which inherently defines the CPU and memory of the guest. You also define the size of the virtual disk. On systems that include SSL and compression hardware processors, the vCMP feature shares these hardware resources among all guests on the system.

About CPU allocation

The following table lists the possible combinations of vCPU and memory allocation for a vCMP guest on various platforms:

Platform	Available vCPU per slot/appliance	Possible vCPU allocation per guest per slot/ appliance	Approximate memory allocated per guest, per slot/appliance based on vCPU allocation (GB)	Maximum number of guests per slot/ appliance
VIPRION B2100 Blade	8	2, 4, 8	3 - 13	4
VIPRION B4200 Blade	8	2, 4	3 - 6	4
VIPRION B4300 Blade	24	2, 4, 6, 12	3 - 21	6

For single-slot guests, when the system allocates CPU cores to a guest, the system determines the best slot for the guest to run on. The system selects the slot with the most unallocated CPU cores. For all-slot guests, the system allocates CPU cores from every available slot.

Guest type	CPU core allocation
Single slot	The system allocates one or more CPU cores to the guest.
All slot	The system allocates two CPU cores from each available slot. For example, if three slots are available, the system allocates two CPU cores from each slot, totaling six CPU cores for that guest. The maximum number of CPU cores that the system can allocate to a guest is eight.

The number of CPU cores that the BIG-IP system assigns to each guest depends on whether you configure the guest to run on a single slot or on all available slots of the system:

The BIG-IP system uses a sequential pattern to determine the chassis slot and CPU cores to which single-slot guests deploy. You control to which slot your guest migrates by knowing this pattern and making sure that the slot to which you want the guest to deploy is the next open resource. You open a slot by disabling its guests; you fill a slot by deploying a temporary guest as placeholder. The table lists the order in which cores and slots are allocated to deploying guests.

Slot #	CPU cores 0 and 1	CPU cores 2 and 3	CPU cores 4 and 5	CPU cores 6 and 7
Slot 1	Fills first	Fills fifth	Fills ninth	Fills thirteenth
Slot 2	Fills second	Fills sixth	Fills tenth	Fills fourteenth
Slot 3	Fills third	Fills seventh	Fills eleventh	Fills fifteenth
Slot 4	Fills fourth	Fills eighth	Fills twelfth	Fills sixteenth

About physical memory allocation

The BIG-IP system allocates a portion of the total system memory to each guest.

About virtual disks allocation

A virtual disk is a portion of the total disk space on the BIG-IP system that the system allocates to a vCMP guest. The system allocates one virtual disk to each slot on which the guest resides. Although each virtual disk for a guest has a fixed, maximum size limit, the actual size of a virtual disk is the amount of space that the guest actually uses on that slot.

The maximum size limit for a guest is 100GB, and the typical footprint of a new guest (when viewed from the host) is around 5GB.

You cannot explicitly create virtual disks; instead, the BIG-IP system creates virtual disks whenever you set the state of a guest to Provisioned and the guest does not already have an attached virtual disk.

About hardware processors allocation

On systems that include SSL and compression hardware processors, the vCMP feature shares these hardware resources among all guests on the system.

Objective - 2.12 Determine the appropriate LTM device security configuration to protect against a security threat

2.12 - Explain the implications of SNAT and NAT on network promiscuity

In a typical network, a host's network adapter only receives frames that are meant for it. If the Host's network adapter supports promiscuous mode, then placing it in promiscuous mode will allow it to receive all frames passed on the switch that are allowed under the VLAN policy for the associated port group. This can be useful for intrusion detection monitoring or if a sniffer needs to analyze all traffic on the network segment.

When the BIG-IP platform performs SNAT or NAT functions to the network traffic that is traversing the system it rewrites the destination IP or source IP address of the traffic depending on the function performed. This can make troubleshooting captures of network traffic difficult since with SNAT all traffic seems to have a source IP address of the BIG-IP system, or with NAT the destination IP address is not the same on each side of the communications of the BIG-IP.

2.12 - Explain the implications of forwarding virtual servers on the environment security

Overview of IP Forwarding Virtual Servers

Forwarding Virtual Servers

There are two different types of forwarding virtual server, the Layer2 forwarding and IP forwarding.

An IP forwarding virtual server accepts traffic that matches the virtual server address and forwards it to the destination IP address that is specified in the request rather than load balancing the traffic to a pool. Address translation is disabled when you create an IP forwarding virtual server, leaving the destination address in the packet unchanged. When creating an IP forwarding virtual server, as with all virtual servers, you can create either a host IP forwarding virtual server, which forwards traffic for a single host address, or a network IP forwarding virtual server, which forwards traffic for a subnet.

Layer 2 (L2) forwarding virtual servers are similar to IP forwarding virtual servers because they do not have pool members to load balance. Therefore, when the BIG-IP LTM system evaluates the packet for processing, the system looks only at the destination IP address.

When creating an L2 forwarding virtual server, it is not possible to specify the destination of the packet by associating a pool with the virtual server. The BIG-IP LTM system will forward packets based on the destination L2 Media Access Control (MAC) address and refer to the forwarding MAC address table.

Forwarding Virtual Servers Security Concerns

Since the forwarding virtual server is not processing traffic to a specific pool resource traffic can be destined for any IP address in the subnet that the BIG-IP system is listening on. This may allow traffic to systems that you did not intend to pass.

2.12 - Describe how to disable services

Overview of IP Forwarding Virtual Servers

The Big-IP platform is a default deny network platform. You have to configure the BIG-IP to listen for traffic whether that is for management or for production traffic. Some listeners need to be built to pass more than one type of traffic. And many times the configuration will allow either one port or all ports. If you want to restrict it to a short list of ports, an iRule can be used that will allow the selective list of ports.

Simply sending a TCP reset to a port scan can tell an intruder a lot about your environment. By default, the TM.RejectUnmatched BigDB variable is set to true, and the BIG-IP system sends a TCP RST packet in response to a non-SYN packet that matches a virtual server address and port or self IP address and port, but does not match an established connection. The BIG-IP system also sends a TCP RST packet in response to a packet that matches a virtual server address, or self IP address, but specifies an invalid port. The TCP RST packet is sent on the client side of the connection, and the source IP address of the reset is the relevant BIG-IP LTM object address or self IP address for which the packet was destined. If TM.RejectUnmatched is set to false, the system silently drops unmatched packets.

Disabling Services On FVS

When you configure a forwarding virtual server you can set the listening port to a single specific port or all ports. Typically when you are forwarding traffic you will do all ports so all traffic passes, but you can restrict it to only a certain type.

Creating an IP forwarding virtual server

1. Log in to the Configuration utility.
2. Navigate to Local Traffic > Virtual Servers.
3. Click Create.

4. Enter a Name for the virtual server.
5. From the Type menu, select Forwarding (IP).
6. If the destination is a single host, select Host, or if the destination is a network, select Network.
7. Enter the IP address for the virtual server (enter a Netmask if the destination is a network).
8. Enter a Service Port number, or Select a service from the adjacent menu (type an asterisk character to match all ports).
9. Click Finished.

2.12 - Describe how to disable ARP

Working with Address Resolution Protocol

Address Resolution Protocol on the BIG-IP system

The BIG-IP system is a multi-layer network device, and as such, needs to perform routing functions. To do this, the BIG-IP system must be able to find destination MAC addresses on the network, based on known IP addresses. The way that the BIG-IP system does this is by supporting Address Resolution Protocol (ARP), an industry-standard Layer 3 protocol. Settings for ARP behaviors can be found on the Main tab, click Network > ARP > Options. You can also see and manage the dynamic and static ARP entries in ARP cache from the console or the GUI.

Virtual Servers

Disabling ARP For A Virtual Server's Address

If you want to control how the BIG-IP handles ARP for a virtual server you can make a change to the configuration of the virtual address of the virtual server.

What is a virtual address?

A virtual address is the IP address with which you associate a virtual server. For example, if a virtual server's IP address and service are 10.10.10.2:80, then the IP address 10.10.10.2 is a virtual address.

You can create a many-to-one relationship between virtual servers and a virtual address. For example, you can create the three virtual servers 10.10.10.2:80, 10.10.10.2:443, and 10.10.10.2:161 for the same virtual address, 10.10.10.2.

You can enable and disable a virtual address. When you disable a virtual address, none of the virtual servers associated with that address can receive incoming network traffic.

A virtual address is created indirectly when you create a virtual server. When this happens, Local Traffic Manager internally associates the virtual address with a MAC address. This in turn causes the BIG-IP system to respond to Address Resolution Protocol (ARP) requests for the virtual address, and to send gratuitous ARP requests and responses with respect to the virtual address. As an option, you can disable ARP activity for virtual addresses, in the rare case that ARP activity affects system performance. This most likely occurs only when you have a large number of virtual addresses defined on the system.

2.12 - Explain how to set up logging for security events on the LTM device

DDoS Attack Prevention in LTM

BIG-IP Local Traffic Manager helps protect against network DoS and DDoS threats. When using LTM, you can protect against network DoS attacks and increase end-user application performance with accurate triggers and controls. In BIG-IP LTM, there are a couple of changes you can make in tightening the configuration and monitoring messages to ensure the LTM helps protect against DoS and DDoS attacks.

1. Lower the default TCP connection timeouts in the TCP profile.
2. Lower the Reaper percents from low 85 / high 95 to low 75 / high 90.
    a. This means fewer connections are held open, but means the LTM will be more aggressive cleaning out idle connections during a TCP connection flood.
3. Analyze the typical and maximum HTTP header size, including cookies that should legitimately be seen.
    a. The default maximum on LTM is 32k.
    b. This should be lowered if your average is 4k and max possible is 8k.
    c. In this example, setting the max header size to 16 should adequately ensure no false positives (resulting in rejected connections), while helping to ensure a number of HTTP header based DoS attacks are better handled.

Monitor /var/log/ltm for messages such as:

- Sweeper imitated - this means the reapers have kicked in due to high TCP connection counts and high memory utilization
- ICMP messages limited to 250 - Usually a ping or form of ICMP attack encountered and being mitigated

- SYNcookie activated - SYN flood attack encountered
- HTTP header size exceeding 32k length - often from SlowLoris or similar HTTP header attack

Once configured, BIG-IP LTM's approach to network DoS and DDoS attacks is an attack mitigation configuration that protects core infrastructure when an attack occurs.

2.12 - Explain how route domains can be used to enforce network segmentation

Working with Route Domains

What is a route domain?

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

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

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

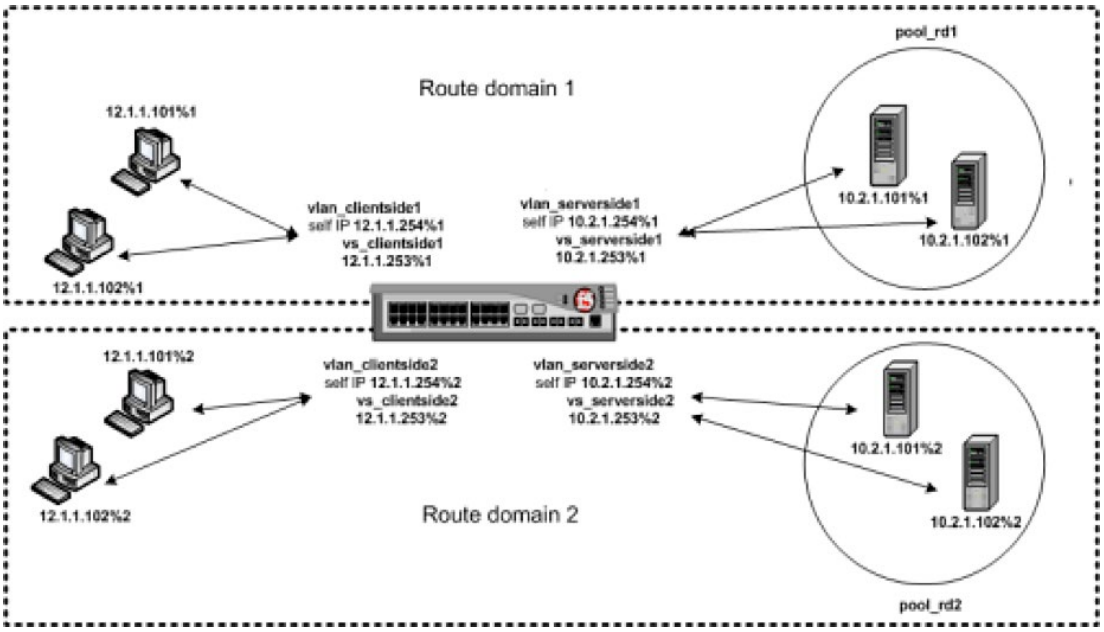
Benefits of route domains

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

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

Sample route domain deployment

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



A sample route domain deployment

SECTION 3 – DEPLOY APPLICATIONS

Objective - 3.01 Describe how to deploy and modify applications using existing and/or updated iApp application templates

DevCentral iApps

*links on DevCentral require member login

iApps

iApps is the BIG-IP system framework for deploying services-based, template-driven configurations on BIG-IP systems running TMOS 11.0.0 and later. It consists of three components: Templates, Application Services, and Analytics. An iApps Template is where the application is described and the objects (required and optional) are defined through presentation and implementation language. An iApps Application Service is the deployment process of an iApps Template which bundles the entire configuration options for a particular application together. You would have an iApps Application Service for SharePoint, for example. iApps Analytics include performance metrics on a per-application and location basis.

Benefits of using iApps:

- User-customizable
- Easy editing of configurations and cleanup
- Reentrancy
- Configuration encapsulation
- Cradle-to-grave configuration management
- Strictness protects against accidental changes to the configuration
- Operational tasks and health status for App objects displayed on App-specific component view (see right)
- Copy/Import/Export capability

iApp Components

The iApps framework consists of two main components, application services and templates.

Application Services

iApps application services use templates to guide users through configuring new BIG-IP system configurations. An application service lets an authorized user easily and consistently deploy complex BIG-IP system configurations just by completing the information required by the associated template. Every application service is attached to a specific configuration and cannot be copied the way that iApps templates can.

Templates

iApps templates create configuration-specific forms used by application services to guide authorized users through complex system configurations. The templates provide programmatic, visual layout and help information. Each new application service uses one of the templates to create a screen with fields and help that guide the user through the configuration process and creates the configuration when finished.

iApps templates allow users to customize by either modifying an existing template or creating one from scratch. Users can create scratch-built templates using either the iApps Templates screen or any text-editing software.

Strict Updates Setting

When you are working in the Application Services properties screen, and select the Advanced view, the Strict Updates field is shown. Selecting Strict Updates protects against accidental changes to an application service's configuration. The Strict Updates setting is on by default when an application service is created.

> **Note:** Unless you have a specific reason to turn off strict updates, F5 recommends that you leave the setting on.

When the strict updates setting is enabled, users can control only objects that are exposed through the templates. Template reentrancy, covered in the template authoring, is not recommended if strict updates is turned off.

> **Note:** Even with strict updates enabled, it is possible to enable and disable some objects using interfaces (such as tmsh or the Configuration Utility) other than the reentrant template. These objects include:

- nodes
- pool members
- virtual addresses
- virtual servers

Deploying an application service

The following procedure covers the minimum steps needed to deploy a configuration using an iApps application service.

1. On the Main tab, expand iApp, and click Application Services.
2. Click Create.
3. In the Name field, type the name for your application service.
4. From the Template List menu, select a template for your application, and wait for the screen to automatically refresh.
5. Configure remaining settings as needed.
6. At the bottom of the screen click Finished to save your changes.
7. Wait for the application properties to load.

8. (Optional) In the Description field, enter information to describe this application service and click Update.

Your application service is now deployed on the BIG-IP system.

Modifying an application service

The following procedure tells how to modify an existing application service.

1. On the Main tab, expand iApp, and click Application Services.
2. From the Application Service List, select an application service to view.
3. Click the Reconfigure tab. The screen displays the settings for the application service.
4. Click the Components tab and use the components tree to view the components that belong to the application service.
5. Edit the fields that require modification and then click Finished to save your changes.

The system saves the application service modifications and they are ready to use.

3.01 - Identify the appropriate application template to use to deploy the application

DevCentral iApps
*Login required

Appropriate Application Template

In versions 11.0 - 11.3, the following iApps Templates were shipped with BIG-IP:

- Citrix Presentation Server 4.5
- Citrix XenApp 5.0, 5.1 and 6.0
- Diameter
- DNS Load Balancing
- HTTP
- IP Forwarding
- LDAP

- Microsoft Exchange 2010 and 2010 SP1
- Microsoft Exchange OWA 2007
- Microsoft IIS 7 and 7.5
- Microsoft Lync 2010
- Microsoft OCS 2007 R2
- Microsoft Sharepoint 2010
- Microsoft Sharepoint 2007
- nPath
- Oracle Application Server 10g (and SSO version 10g Release 2 - v10.1.2.0.2)
- Oracle EBS 12
- Oracle PeopleSoft 9
- Oracle WebLogic Server 10.3 (BEA WebLogic 5.1 and 8.1)
- Radius
- SAP Enterprise Portal 6.0, mySAP ERP 2005
- SAP ERP Central Component 6.0, mySAP ERP 2005
- VMware View 2.1, 3.0.1, 4.0 and 4.5

There are many application specific templates that you can use that can match the application for which you are trying to build out configuration. Some of the templates are more generic to fit a broader set of applications, for example the HTTP template. If there is not a template that fits your application or you want one specifically for your application you can try to find a template from the F5 DevCentral website or you can create your own.

3.01 - Describe how to locate, retrieve and import new and updated application templates

iApp Codeshare

A good size list of iApps templates come preloaded in the version of TMOS you are running. As you upgrade to newer versions of TMOS more current iApps templates will be in the OS build.

There are also supported templates available for download. See the iApps Codeshare site in the link for this section for an additional list of F5 templates by application. You can also find community-contributed templates on this site.

To retrieve and install the new template use the following steps:

Installing New iApp Templates

- Download ZIP file containing the new iApp Templates and extract the *.tmpl file(s)
- Log on to the BIG-IP system web-based Configuration utility.
- On the Main tab, expand iApp, and then click Templates.
- Click the Import button on the right side of the screen.
- Click a check in the Overwrite Existing Templates box.
- Click the Browse button, and then browse to the location you saved the iApp file.
- Click the Upload button. The iApp is now available for use.

3.01 - Identify use cases for deploying the application templates

DevCentral iApps
*Login required

iApps Use Cases

Use iApps to automate the way you add virtual servers so that you don't have to go through the same manual steps every time you add a new application. Or build a custom iApps to manage your iRules inventory.

iApps gives the administrators of BIG-IP the ability to deploy applications they are not familiar with by using the templates that are structured around best practice deployments. This can create configuration consistency when multiple administrators are working in the same system building configurations for similar applications. iApps can also cut down on deployment time for new applications. The possibilities are nearly endless since you have the ability to create your own templates.

Objective - 3.02 Given application requirements, determine the appropriate profiles and profile settings to use

3.02 - Describe the connections between profiles and virtual servers

Introduction to Profiles

Profile Summary:

Profiles are a configuration object containing a logical group of settings for controlling the behavior of a particular type of network traffic that, when assigned to a virtual server, define how that traffic should be processed by the virtual server as it is passing through the virtual server.

Profiles

Profiles are a configuration object that you can use to affect the behavior of certain types of network traffic. More specifically, a profile is an object that contains settings with values, for controlling the behavior of a particular type of network traffic, such as HTTP connections. Profiles also provide a way for you to enable connection and session persistence, and to manage client application authentication.

By default, Local Traffic Manager provides you with a set of profiles that you can use as is. These default profiles contain various settings with default values that define the behavior of different types of traffic. If you want to change those values to better suit the needs of your network environment, you can create a custom profile. A custom profile is a profile derived from a default profile and contains values that you specify.

Connection Between Profiles and Virtual Servers

Once you have created a profile for a specific type of traffic, you implement the profile by associating that profile with one or more virtual servers.

You associate a profile with a virtual server by configuring the virtual server to reference the profile. Whenever the virtual server receives that type of traffic, Local Traffic Manager applies the profile settings to that traffic, thereby controlling its behavior. Thus, profiles not only define capabilities per network traffic type, but also ensure that those capabilities are available for a virtual server.

Because certain kinds of traffic use multiple protocols and services, users often create multiple profiles and associate them with a single virtual server.

For example, a client application might use the TCP, SSL, and HTTP protocols and services to send a request. This type of traffic would therefore require three profiles, based on the three profile types TCP, Client SSL, and HTTP.

Each virtual server lists the names of the profiles currently associated with that virtual server. You can add or remove profiles from the profile list, using the Configuration utility. Note that Local Traffic Manager has specific requirements regarding the combinations of profile types allowed for a given virtual server.

In directing traffic, if a virtual server requires a specific type of profile that does not appear in its profile list, Local Traffic Manager uses the relevant default profile, automatically adding the profile to the profile list. For example, if a client application sends traffic over TCP, SSL, and HTTP, and you have assigned SSL and HTTP profiles only, Local Traffic Manager automatically adds the default profile tcp to its profile list.

At a minimum, a virtual server must reference a profile, and that profile must be associated with a UDP, FastL4, Fast HTTP, or TCP profile type. Thus, if you have not associated a profile with the virtual server, Local Traffic Manager adds a UDP, FastL4, Fast HTTP, or TCP default profile to the profile list.

The default profile that Local Traffic Manager chooses depends on the configuration of the virtual servers protocol setting. For example, if the protocol setting is set to UDP, Local Traffic Manager adds the udp profile to its profile list.

3.02 - Describe profile inheritance

Introduction to Profiles

Profile Inheritance

Custom profiles have a parent-child relationship with the parent profile used to build the custom profile. When you make a change to any profile that is used as a parent profile to another custom profile, the change you make will also change in the custom profile if it is not a modified value. Meaning that if that value is already set as a custom value in the custom profile it will remain the custom value and not inherit the changed parent value.

Custom and Parent Profiles

A custom profile is a profile that is derived from a parent profile that you specify. A parent profile is a profile from which your custom profile inherits its settings and their default values.

When creating a custom profile, you have the option of changing one or more setting values that the profile inherited from the parent profile. In this way, you can pick and choose which setting values you would like to change and which ones you would like to retain. An advantage to creating a custom profile is that by doing so, you preserve the setting values of the parent profile.

> **Note:** If you do not specify a parent profile when you create a custom profile, Local Traffic Manager automatically assigns a related default profile as the parent profile. For example, if you create a custom HTTP type of profile, the default parent profile is the default profile http.

If you do not want to use a default profile as is or change its settings, you can create a custom profile. Creating a custom profile and associating it with a virtual server allows you to implement your own specific set of traffic-management policies.

When you create a custom profile, the profile is a child profile and automatically inherits the setting values of a parent profile that you specify. However, you can change any of the values in the child profile to better suit your needs.

If you do not specify a parent profile, Local Traffic Manager uses the default profile that matches the type of profile you are creating.

> **Important:** When you create a custom profile, the BIG-IP system places the profile into your current administrative partition.

> **Important:** Within the Configuration utility, each profile creation screen contains a check box to the right of each profile setting. When you check a box for a setting and then specify a value for that setting, the profile then retains that value, even if you change the corresponding value in the parent profile later. Thus, checking the box for a setting ensures that the parent profile never overwrites that value through inheritance.

Once you have created a custom profile, you can adjust the settings of your custom profile later if necessary. If you have already associated the profile with a virtual server, you do not need to perform that task again.

Using the default profile as the parent profile

A typical profile that you can specify as a parent profile when you create a custom profile is a default profile. For example, if you create a custom TCP-type profile called my_tcp_profile, you can use the default profile tcp as the parent profile. In this case, Local Traffic Manager automatically creates the profile my_tcp_profile so that it contains the same settings and default values as the default profile tcp. The new custom profile thus inherits its settings and values from its parent profile. You can then retain or change the inherited setting values in the custom profile to suit your needs.

Using a custom profile as the parent profile

When creating a custom profile, you can specify another custom profile, rather than the default profile, as the parent profile. The only restriction is that the custom profile that you specify as the parent must be of the same profile type as the profile you are deriving from the parent. Once you have created the new custom profile, its settings and default values are automatically inherited from the custom profile that you specified as the parent.

For example, if you create a profile called my_tcp_profile2, you can specify the custom profile my_tcp_profile as its parent. The result is that the default setting values of profile my_tcp_profile2 are those of its parent profile my_tcp_profile.

If you subsequently modify the settings of the parent profile (my_tcp_profile), Local Traffic Manager automatically propagates those changes to the new custom profile.

For example, if you create the custom profile my_tcp_profile and use it as a parent profile to create the custom profile my_tcp_profile2, any changes you make later to the parent profile my_tcp_profile are automatically propagated to profile my_tcp_profile2. Conversely, if you modify any of the settings in the new custom profile (in our example, my_tcp_profile2), the new custom profile does not inherit values from the parent profile for those particular settings that you modified.

3.02 - Explain how to configure the different SSL profile settings

SSL profiles

There are two different types of SSL profiles for the BIG-IP system. There is a Client SSL profile which enables the BIG-IP system to accept and terminate client requests that are sent using a fully SSL-encapsulated protocol and provides a number of configurable settings for managing client-side Secure Socket Layer (SSL) connections. There is also a Server SSL profile which enables the BIG-IP system to initiate secure connections to your SSL servers by using a fully SSL-encapsulated protocol and providing configurable settings for managing server-side SSL connections.

You can use the default SSL client and Server

The following sections will describe in depth the different settings in each SSL profile type.

Overview of the Client SSL profile (11.x)

Client SSL Profile

General Properties

Setting	Description
Name	The **Name** setting is required. To create a Client SSL profile, you must specify a unique name for the profile.
Parent Profile	This setting specifies an existing profile to use as the parent profile. A profile inherits settings from its parent, unless you override the setting by selecting its **Custom** box and modifying the value. The default is **clientssl** profile.

Configuration

This section describes the most commonly used SSL settings for a Client SSL profile, for example, the certificate and key to send to SSL clients for certificate exchange.

Setting	Description
Certificate	The **Certificate** setting (**Certificate Key Chain** in BIG-IP 11.5.0 and later) is required. By default, the Client SSL profile uses a self-signed certificate, named **default.crt**. However, this field is almost always customized to reference a certificate that is specific to the site to which the profile will be applied. The SSL certificate must be in PEM format and must be imported to the BIG-IP system with the corresponding key before they can be referenced by an SSL profile. For information about importing an SSL certificate and key using the Configuration utility, refer to SOL14620: Managing SSL certificates for BIG-IP systems. For information about importing an SSL certificate and key using the Traffic Management Shell (tmsh) utility, refer to SOL14031: Importing the SSL certificate and key using the Traffic Management Shell. For information about verifying the certificate format, refer to SOL13349: Verifying SSL certificate and key pairs from the command line (11.x) After importing the SSL certificate and matching key to the BIG-IP system, choose the appropriate certificate from the Certificate setting.
Key	The **Key** setting is required. By default, the Client SSL profile uses the built-in key, named **default.key**, which matches **default.crt**. You must choose the key that matches the configured certificate and the key must be in PEM format. After importing the SSL certificate and matching key to the BIG-IP system, choose the appropriate key from the **Key** setting.
Passphrase	The **Passphrase** setting is optional. It is only required if the key is passphrase-protected. There is no default value for this setting. If your key is passphrase-protected, enter the required passphrase.
Chain	The **Chain** setting is optional. This setting is used to specify a certificate bundle or chain the client can use to establish a trust relationship with a server that presents a certificate signed by an untrusted Certificate Authority (CA). The default value for the Chain setting is **None**, indicating that no chain certificate will be presented to the client with the server SSL certificate. This setting lists the name of all the SSL certificates installed in the BIG-IP system's SSL certificate store. If you are using certificates signed by an Intermediate CA, F5 recommends that you create and install a bundle that contains the certificates for all of the CAs in the chain between the certificate configured in the SSL profile and a root CA whose certificate is trusted by the expected client base. The new certificate bundle can then be selected in the **Chain** setting. For information about creating and installing a custom certificate bundle, refer to SOL13302: Configuring the BIG-IP system to use an SSL chain certificate (11.x). **Note:** Regardless of the **Chain** setting, if the **Trusted Certificate Authorities** setting is configured, the certificate bundle contained in the configured **Trusted Certificate Authorities** file is presented.

Setting	Description
Ciphers	The Ciphers setting is optional. By default, the Client SSL profile uses the DEFAULT cipher string. In most cases, the DEFAULT cipher string is appropriate, but can be customized as necessary to meet the security and performance needs of your site. For information about configuring the SSL cipher for an SSL profile, refer to SOL13171: Configuring the cipher strength for SSL profiles (11.x).
Options	When enabled (**Options List**), references the **Options List** setting, which industry standard SSL options and workarounds use for handling SSL processing. The default setting is **All Options Disabled**.
Options List	The **Options List** setting provides selection from a set of industry standard SSL options and workarounds for handling SSL processing.
Proxy SSL	The Proxy SSL setting was introduced in BIG-IP 11.0.0. By default, the Proxy SSL setting is disabled (cleared). When enabled, the client is allowed to directly authenticate with the server, and the server can authenticate with the client, based on the client certificate presented. In a typical setup, with the BIG-IP system in the middle, the client and server cannot communicate directly to authenticate each other. The Proxy SSL setting requires both a Client SSL profile and a Server SSL profile, and must be enabled in both profiles. For information about the Proxy SSL setting, refer to the following resources: 1. SOL13385: Overview of the Proxy SSL feature 2. The Implementing Proxy SSL on a Single BIG-IP system chapter of the BIG-IP LTM Implementations guide.
ModSSL Methods	**ModSSL Methods** enables or disables ModSSL method emulation. Enable this option when OpenSSL methods are inadequate, for example, when you want to use SSL compression over TLSv1. By default, this setting is disabled (cleared).
Cache Size	The **Cache Size** setting specifies the maximum number of SSL sessions allowed in the SSL session cache. The default value for **Cache Size** is **262144** sessions. For information about the SSL **Cache Size** settings, refer to SOL6767: Overview of the BIG-IP SSL session cache profile settings.
Cache Timeout	The **Cache Timeout** setting specifies the number of seconds that SSL sessions are allowed to remain in the SSL session cache before being removed. The default value for **Cache Timeout** is **3600** seconds. The range of values configurable for Cache Timeout is between **0** and **86400** seconds inclusive. **Note:** Longer cache time-out periods can increase the risk of SSL session hijacking.
Alert Timeout	The **Alert Timeout** setting specifies the duration that the system tries to close an SSL connection by transmitting an alert or initiating an unclean shutdown before resetting the connection. The default value for BIG-IP 11.2.0 and later is **10** seconds. The default value for 11.0.0 through 11.1.0 is **60** seconds. Select **Indefinite** to specify that the connection should not be reset after transmitting an alert or initiating an unclean shutdown.

Setting	Description
Handshake Timeout	The **Handshake Timeout** setting specifies the number of seconds that the system tries to establish an SSL connection before terminating the operation. The default value for BIG-IP 11.2.0 and later is **10** seconds. The default value for 11.0.0 through 11.1.0 is **60** seconds. Selecting **Indefinite** specifies that the system continues trying to establish a connection for an unlimited time.
Renegotiation	The **Renegotiation** setting can be configured to control whether the virtual server allows midstream session renegotiation. When **Renegotiation** is enabled, the BIG-IP system processes mid-stream SSL renegotiation requests. When disabled, the system terminates the connection, or ignores the request, depending on system configuration. By default, this setting is enabled (selected).
Renegotiation Period	The amount of time in seconds from the initial connection before the system the SSL session. Indefinite will not renegotiate the SSL session and is the default setting.
Renegotiation Size	The amount of application data in megabytes from the initial connection before the system renegotiates the SSL session. Indefinite will not renegotiate the SSL session and is the default setting.
Renegotiate Max Record Delay	The number of SSL records allowed during the SSL renegotiation before the system terminates the connection. Indefinite allows an unlimited number and is the default setting. The Renegotiation Max Record Delay option was introduced in BIG-IP version 11.6.0.
Secure Renegotiation	The BIG-IP SSL profiles support the TLS Renegotiation Indication Extension, which allows the user to specify the method of secure renegotiation for SSL connections. The default value for the Client SSL profile is **Require**. The values for the **Secure Renegotiation** setting in the Client SSL profile are as follows: 1. **Request:** Specifies that the system requests secure renegotiation of SSL connections. 2. **Require:** Specifies that the system requires secure renegotiation of SSL connections. In this mode, the system permits initial SSL handshakes from clients, but terminates renegotiations from clients that do not support secure renegotiation. 3. **Require Strict:** Specifies that the system requires strict, secure renegotiation of SSL connections. In this mode, the system denies initial SSL handshakes from clients that do not support secure renegotiation.
Server Name	Starting in BIG-IP 11.1.0, the BIG-IP SSL profiles support the TLS Server Named Indication (SNI) Extension, which allows the BIG-IP system to select the appropriate SSL profile based on the TLS SNI information provided by the client. The **Server Name** setting specifies the fully qualified DNS hostname of the server (or a wildcard string containing the asterisk '*' character to match multiple names) used in the TLS SNI connection. There is no default value for this setting. For information about configuring the TLS SNI feature on the BIG-IP system, refer to SOL13452: Configuring a virtual server to serve multiple HTTPS sites using TLS Server Name Indication feature.

Setting	Description
Default SSL Profile for SNI	When enabled, this setting indicates that the profile should be used as the default SSL profile when there is no match to the server name, or when the client does not support TLS SNI extension. By default, this setting is disabled (cleared). For information about configuring the TLS SNI feature on the BIG-IP system, refer to SOL13452: Configuring a virtual server to serve multiple HTTPS sites using TLS Server Name Indication feature.
Require Peer SNI Support	When enabled, this setting requires that the client must support the TLS SNI extension; otherwise, the BIG-IP system disconnects the client connection with a fatal alert. By default, this setting is disabled (cleared).
Unclean Shutdown	The SSL protocol performs a clean shutdown of an active TLS/SSL connection by sending a close notify alert to the peer system. The **Unclean Shutdown** setting allows the BIG-IP system to perform an unclean shutdown of SSL connections by closing the underlying TCP connection without sending the SSL close notify alerts. By default, this setting is enabled (selected) and is useful for certain browsers that handle SSL shutdown alerts differently. For example, some versions of Internet Explorer require SSL shutdown alerts from the server while other versions do not, and the SSL profile cannot always detect this requirement. **Important:** If you disable (clear) the **Unclean Shutdown** setting, some browsers may display blank pages or errors when connecting to the virtual server.
Strict Resume	The **Strict Resume** setting enables or disables the resumption of SSL sessions after an unclean shutdown. By default, this setting is disabled (cleared).
Non-SSL Connections	Enables or disables acceptance of non-SSL connections. By default, the acceptance of non-SSL sessions is disabled (cleared).

Client Authentication

The Client Authentication section of the Client SSL profile is specific to client certificate authentication. Some applications require clients to establish their identity to the server before proceeding with the SSL session. Client certificate authentication uses the following sequence of events:

- The client requests an SSL connection.
- The SSL server presents its SSL certificate, and any configured chain certificate bundle, to the client.
- The SSL client uses the CA certificates stored in its Trusted Device Certificate store, and the supplied certificate chain, if necessary, to authenticate the server.
- The SSL server requests a client certificate, advertising a list of preferred CAs if configured to do so.
- The SSL client presents its SSL certificate.
- The SSL server uses its configured, trusted CA certificate bundle, to authenticate the client.

Setting	Description
Client Certificate	The **Client Certificate** setting is required. This setting is used to enable and disable client certificate authentication. The possible options for the **Client Certificate** setting are: 3. **Ignore**: The **Ignore** setting is the default setting. It disables Client Certificate Authentication. The BIG-IP system ignores any certificate presented and does not authenticate the client before establishing the SSL session. 4. **Request**: The **Request** setting enables optional Client Certificate Authentication. The BIG-IP system will request a client certificate and attempt to verify it. However, an SSL session is established regardless of whether or not a valid client certificate from a trusted CA was presented. The **Request** setting is often used in conjunction with iRules to provide selective access depending on the certificate presented. For example, this option would be useful if you would like to allow clients who present a certificate from the configured trusted CA to gain access to the application, while clients who do not provide the required certificate are redirected to a page that details the access requirements. However, if you are not using iRules to enforce a different outcome, depending on the certificate details, there is no functional benefit to using the **Request** setting instead of the default Ignore setting. In both cases, an SSL session is established, regardless of the certificate presented, and the connection is proxied to the default pool. 5. **Require**: The Require setting enforces Client Certificate Authentication. The BIG-IP system will request a client certificate and attempt to verify it. An SSL session is established only if a valid client certificate from a trusted CA was presented. The Require setting is used to restrict access to only clients that present a valid certificate from a trusted CA. **Note:** The **Auto** setting was removed in BIG-IP 11.0.0.
Frequency	The **Frequency** setting specifies the frequency of client authentication for an SSL session. The default value for this setting is **once**.
Certificate Chain Traversal Depth	The **Certificate Chain Traversal Depth** setting specifies the maximum number of certificates to be traversed in a client certificate chain. The default value is **9**.

Setting	Description

Trusted Certificate Authorities	The **Trusted Certificate Authorities** setting is required only if the BIG-IP system performs Client Certificate Authentication. This setting is used to specify the BIG-IP system's Trusted Certificate Authorities store (the CAs that the BIG-IP system trusts when the system verifies a client certificate that is presented during Client Certificate Authentication). The default value for the **Trusted Certificate Authorities** setting is **None**, which indicates that no CAs are trusted. The None value is only appropriate if Client Certificate Authentication is not desired. Unless it is performing Client Certificate Authentication, the SSL server does not need to trust any CA. If the BIG-IP Client Certificate Mode is set to Require, but Trusted Certificate Authorities is set to None, clients cannot establish SSL sessions with the virtual server. This setting lists the name of all the SSL certificates installed on the BIG-IP system.
	
	The **ca-bundle** certificate may be appropriate for use as a Trusted Certificate Authorities certificate bundle. However, if this bundle is specified as the Trusted Certificate Authorities certificate store, any valid client certificate that is signed by one of the popular Root CAs included in the default **ca-bundle.crt** is authenticated. This provides some level of identification, but very little access control because almost any valid client certificate could be authenticated. However, it is more common when configuring client certificate authentication to accept client certificates from one, or a select few, PKIs or private CAs.
	
	If you want to trust only certificates signed by a specific CA or set of CAs, F5 recommends that you create and install a bundle that contains trustworthy CA certificates. The new certificate bundle can then be selected in the **Trusted Certificate Authorities** setting. For information about creating a custom certificate bundle, refer to SOL13302: Configuring the BIG-IP system to use an SSL chain certificate (11.x).
	
	The bundle must also include the entire chain of CA certificates necessary to establish a chain of trust, as described in the **Chain** setting. To support multiple PKI hierarchies, this bundle can contain CA certificates from several different PKIs. The bundle does not need to contain CA certificates from the PKI that signed the server SSL certificate, unless client SSL certificates from that PKI must be validated by the BIG-IP system. However, in practice, Client Certificate Authentication is most commonly used with Private PKIs, and the **Trusted Certificate Authorities** setting often contains only a certificate or chain from the PKI that signed the server certificate.
	
	You can use the **openssl** command to verify the client certificate against the Trusted Certificate Authority bundle prior to importing it onto the BIG-IP system. For example, the following **openssl** command verifies the client certificate, **client.crt**, against the Trusted Certificate Authority bundle:
	`openssl verify -purpose sslclient -CAfile /path/to/trusted-ca-bundle.crt /path/to/client.crt`
	
	If the chain of trust can be established for the server certificate using the specified chain, the command returns output, similar to the following example:
	`client.crt: OK`
	
	Important: Configuring the **Trusted Certificate Authorities** setting has no effect on client validation of the SSL server certificate that the BIG-IP system presents upon connection to an SSL virtual server. It is the SSL client's responsibility to verify the validity of the SSL server's certificate using its own Trusted Certificate store. However, the certificate bundle contained in the configured **Trusted Certificate Authorities** file is presented with the server SSL certificate, regardless of the Chain setting

Setting	Description
Advertised Certificate Authorities	The **Advertised Certificate Authorities** setting is optional. It is used to specify the CAs that the BIG-IP system advertises as trusted when soliciting a client certificate for client certificate authentication. If the **Client Certificate** setting is configured to **Require** or **Request**, you can configure the **Advertised Certificate Authorities** setting to send clients a list of CAs that the server is likely to trust. The default value for the **Advertised Certificate Authorities** setting is **None**, indicating that no CAs are advertised. When set to **None**, no list of trusted CAs is sent to a client with the certificate request. This setting lists the name of all the SSL certificates installed on the BIG-IP system. If you want to advertise only a specific CA, or set of CAs, F5 recommends that you create and install a bundle that contains the certificates of the CA to advertise. You can then select the new certificate bundle in the **Advertised Certificate Authorities** setting. For information about creating a custom certificate bundle, refer to SOL13302: Configuring the BIG-IP to use an SSL chain certificate (11.x). You can configure the **Advertised Certificate Authorities** setting to send a different list of CAs than that specified for the Trusted Certificate Authorities. This allows greater control over the configuration information shared with unknown clients. You might not want to reveal the entire list of trusted CAs to a client that does not automatically present a valid client certificate from a trusted CA. Although the two settings can be configured differently, in most cases, you should configure the **Advertised Certificate Authorities** setting to use the same certificate bundle as the **Trusted Certificate Authorities** setting. **Important:** Avoid specifying a bundle that contains many certificates when configuring the **Advertised Certificate Authorities** setting. This minimizes the number of certificates that must be exchanged during a client SSL handshake. The maximum size allowed by the BIG-IP system for native SSL handshake messages is 14,304 bytes. Although typical handshakes do not result in excessive message length, if the SSL handshake is negotiating a native cipher, and the total length of all messages in the handshake exceeds this byte threshold, the handshake will fail.
Certificate Revocation List (CRL)	The **Certificate Revocation List (CRL)** setting allows you to specify a CRL that the BIG-IP system should use to check revocation status of a certificate prior to authenticating a client. If you want to use a CRL, you must import it to the BIG-IP system. The name of the CRL file can then be entered in the **Certificate Revocation List (CRL)** setting dialog box. For information about importing an SSL CRL file, refer to SOL14620: Managing SSL certificates for BIG-IP systems. Since CRLs can quickly become outdated, F5 recommends that you use either OCSP or CRLDP profiles for more robust and current verification functionality.

Overview of the Client SSL Profile (11.x)

Server SSL Profile

General Properties

Setting	Description
Name	The Name setting is required. To create a Server SSL profile, you must specify a unique name for the profile. For more information about the profile name requirements, refer to the following articles: 6. SOL6869: Reserved words that should not be used in BIG-IP configurations 7. SOL13209: BIG-IP configuration object names must begin with an alphabetic character
Parent Profile	This setting specifies an existing profile to use as the parent profile. A profile inherits settings from its parent, unless you override the setting by selecting its **Custom** box and modifying the value. The default is **serverssl** profile.

Configuration

This section describes the most common SSL settings for a Server SSL profile, for example, the certificate and key to send to SSL servers for certificate exchange.

Setting	Description
Certificate	The **Certificate** setting is optional. The default value for this setting is None. When you apply a Server SSL profile to a virtual server, the BIG-IP system acts as an SSL client. If you do not intend for the BIG-IP system to present its client certificate on behalf of clients traversing the virtual server, select **None**. If you expect the BIG-IP system to present a client certificate, import the certificate and matching key to the BIG-IP system, and then choose the appropriate certificate from the menu. For information about importing an SSL certificate and key by using the Configuration utility, refer to SOL14620: Managing SSL certificates for BIG-IP systems. For information about importing an SSL certificate and key using the Traffic Management Shell (tmsh) utility, refer to SOL14031: Importing the SSL certificate and key using the Traffic Management Shell. For information about verifying the certificate format, refer to SOL13349: Verifying SSL certificate and key pairs from the command line (11.x)
Key	The **Key** setting is required only for configured certificates. If you have configured a certificate, you must choose the key that matches the configured certificate. The default value for this setting is **None**.

Setting	Description
Passphrase	The **Passphrase** setting is optional. It is only required if the key is passphrase-protected. There is no default value for this setting. If a key is specified and the key is passphrase protected, enter the required passphrase. When no passphrase is configured, the **Passphrase** field displays an eight-asterisk mask (********), giving the appearance that an eight-character password has been configured. The passphrase is encrypted before it is saved in the **bigip.conf** file.
Chain	The **Chain** setting is optional. This setting is used to specify a certificate bundle or chain that the server can use to establish a trust relationship with a client that presents a certificate signed by an untrusted Certificate Authority (CA). The default value for the Chain setting is **None**, indicating that the BIG-IP system will not present a chain certificate with its client SSL certificate. This setting lists the name of all the SSL certificates installed in the BIG-IP system's SSL certificate store. If the certificate configured in the Server SSL profile is signed by an Intermediate CA, F5 recommends that you create and install a bundle that contains the certificates of all the CAs in the chain between the certificate configured in the Server SSL profile and a root CA whose certificate is trusted by your SSL servers. The new certificate bundle can then be selected in the Chain setting. For information about creating and installing a custom certificate bundle, refer to SOL13302: Configuring the BIG-IP system to use an SSL chain certificate (11.x).
Ciphers	The **Ciphers** setting is optional. By default, the Server SSL profile uses the **DEFAULT** cipher string. In most cases, the default setting is appropriate, but can be customized, as necessary, to meet the security and performance needs of your site. The SSL server selects the cipher used in a particular connection from the ciphers presented by the SSL client. When using Server SSL, the BIG-IP system acts as an SSL client. Although the server decides which cipher to use, you can gain some control by customizing the ciphers presented by the client. For information about configuring the SSL cipher for an SSL profile, refer to SOL13171: Configuring the cipher strength for SSL profiles (11.x).
Options	When enabled, (**Options List**) references the **Options List** setting, which industry standard SSL options and workarounds for handling SSL processing. The default setting is **All Options Disabled**.
Options List	The **Options List** setting provides selection from a set of industry standard SSL options and workarounds for handling SSL processing.
Proxy SSL	The **Proxy SSL** setting was introduced in BIG-IP 11.0.0. By default, the **Proxy SSL** setting is disabled (cleared). When enabled, the client is allowed to directly authenticate with the server, and the server can authenticate with the client, based on the client certificate presented. In a typical setup with the BIG-IP system in the middle, the client and server cannot communicate directly to authenticate each other. The **Proxy SSL** setting requires both a Client SSL profile and a Server SSL profile, and must be enabled in both profiles. For information about the **Proxy SSL** setting, refer to the following resources: 8. SOL13385: Overview of the Proxy SSL feature 9. The Implementing Proxy SSL on a Single BIG-IP system chapter of the BIG-IP LTM Implementations guide.

Setting	Description
ModSSL Methods	The **ModSSL Methods** setting enables or disables ModSSL method emulation. Enable this option when OpenSSL methods are inadequate, for example, when you want to use SSL compression over TLSv1. By default, this setting is disabled (cleared).
Cache Size	The **Cache Size** setting specifies the maximum number of SSL sessions allowed in the SSL session cache. The default value for the **Cache Size** setting is **262144** sessions. For information about the SSL **Cache Size** settings, refer to SOL6767: Overview of the BIG-IP SSL session cache profile settings.
Cache Timeout	The **Cache Timeout** setting specifies the number of seconds that SSL sessions are allowed to remain in the SSL session cache before being removed. The default value for the **Cache Timeout** setting is **3600** seconds. The range of values configurable for the **Cache Timeout** setting is between 0 and 86400 seconds, inclusive.
Alert Timeout	The **Alert Timeout** setting specifies the duration that the system tries to close an SSL connection by transmitting an alert or initiating an unclean shutdown before resetting the connection. The default value for this setting in BIG-IP 11.2.0 and later is **10** seconds. The default value for this setting in BIG-IP 11.0.0 through 11.1.0 is **60** seconds. You can select **Indefinite** to specify that the connection should not be reset after transmitting an alert or initiating an unclean shutdown.
Handshake Timeout	The **Handshake Timeout** setting specifies the number of seconds that the system tries to establish an SSL connection before terminating the operation. The default value for this setting in BIG-IP 11.2.0 and later is **10** seconds. The default value for this setting in BIG-IP 11.0.0 through 11.1.0 is **60** seconds. You can select **Indefinite** to specify that the system should continue to try and establish a connection for an unlimited time.
Renegotiation	You can configure the **Renegotiation** setting to control whether the virtual server allows midstream session renegotiation. When **Renegotiation** is enabled, the BIG-IP system processes mid-stream SSL renegotiation requests. When **Renegotiation** is disabled, the system terminates the connection, or ignores the request, depending on the system configuration. By default, this setting is enabled (selected).
Renegotiation Period	You can configure the **Renegotiate Period** setting to control the amount of time, in seconds, that the system waits before renegotiating the SSL session. By default, this setting is set to **Indefinite**. When this setting is set to **Indefinite**, the system does not renegotiate SSL sessions based on a specified time interval.
Renegotiation Size	The **Renegotiate Size** setting controls the amount of data exchange, in megabytes, before the system renegotiates the SSL session. By default, this setting is set to **Indefinite**. When this setting is set to **Indefinite**, the system does not renegotiate SSL sessions based on the amount of exchanged data.

Setting	Description
Secure Renegotiation	The BIG-IP SSL profiles support the TLS Renegotiation Indication Extension, which allows the user to specify the method of secure renegotiation for SSL connections. The default value for the Server SSL profile is **Require Strict**. The values for the **Secure Renegotiation** setting in the Server SSL profile are as follows: 4. **Request**: Specifies that the system requests secure renegotiation of SSL connections. 5. **Require**: Specifies that the system requires secure renegotiation of SSL connections. In this mode, SSL connections initiated from the system to an unpatched server fail when renegotiation is enabled. 6. **Require Strict**: Specifies that the system requires strict, secure renegotiation of SSL connections. In this mode, SSL connections that are initiated from the system to an unpatched server fail when renegotiation is enabled. Within the context of the Server SSL profile, there is no behavioral difference between the **Require** and **Require Strict** settings. In either mode, initial SSL connections from the BIG-IP system to unpatched servers fail.
Server Name	Starting in BIG-IP 11.1.0, the BIG-IP SSL profiles support the TLS Server Named Indication (SNI) Extension, which allows the BIG-IP system to send ClientHello messages with SNI extension. The **Server Name** setting specifies the fully qualified DNS hostname of the server (or a wildcard string containing the asterisk '*' character to match multiple names) used in the TLS SNI connection. There is no default value for this setting.
Default SSL Profile for SNI	When enabled, this setting indicates that the system should use the profile as the default SSL profile for connecting to the server. By default, this setting is disabled (cleared).
Require Peer SNI support	When enabled, this setting requires that the server must support the TLS SNI extension; otherwise, the BIG-IP system disconnects the SSL connection with a fatal alert. By default, this setting is disabled (cleared).
Unclean Shutdown	The SSL protocol performs a clean shutdown of an active TLS/SSL connection by sending a close notify alert to the peer system. The **Unclean Shutdown** setting allows the BIG-IP system to perform an unclean shutdown of SSL connections by closing the underlying TCP connection without sending the SSL close notify alerts. By default, this setting is enabled (selected).
Strict Resume	The **Strict Resume** setting enables or disables the resumption of SSL sessions after an unclean shutdown. By default, this setting is disabled (cleared).

Server Authentication

The Server Authentication section of the Server SSL profile provides configurable settings for handling server authentication before proceeding with the SSL session.

Setting	Description
Server Certificate	The **Server Certificate** setting specifies how the system handles server certificates. The possible values for the **Server Certificate** setting are: 10. **Ignore**: The **Ignore** setting is the default setting. The BIG-IP system ignores certificates from the server and never authenticates the server. 11. **Require**: The **Require** setting enforces server authentication. The BIG-IP system requires the server to present a valid certificate before establishing the SSL session. If you select **Require** as the **Server Certificate** setting, you must also specify a value in the **Authenticate Name** setting. A blank **Authenticate Name** setting indicates that all servers are authenticated, even though you have specified **Require** as the **Server Certificate** setting.
Frequency	The **Frequency** setting specifies the frequency of server authentication for an SSL session. The default value for this setting is **Once**, which causes the system to authenticate the server for an SSL session only once. When you configure this setting to **Always**, the system authenticates the server for an SSL session and every subsequent reuse of the SSL session.
Certificate Chain Traversal Depth	The **Certificate Chain Traversal Depth** setting specifies the maximum number of certificates that the system traverses in a server certificate chain. The default value is **9**.
Authenticate Name	This setting specifies a Common Name (CN) that is embedded in a server certificate. The system authenticates a server based on the specified CN. There is no default value for this setting.

Setting	Description
Trusted Certificate Authorities	The T**rusted Certificate Authorities** setting is optional. The system uses this setting to specify the CAs that the BIG-IP system trusts when verifying a server certificate. The default value for this setting is **None**, which causes the system to accept a server certificate signed by any CA. If you select **Require** for the **Server Certificate** setting, you must specify a CA from the **Trusted Certificate Authorities** setting. The selected CA will be trusted by the system when verifying a server certificate. The **ca-bundle** certificate may be appropriate for use as a Trusted Certificate Authorities certificate bundle. However, if this bundle is specified as the Trusted Certificate Authorities certificate store, any valid server certificate that is signed by one of the popular root CAs included in the default **ca-bundle.crt** is authenticated. This provides some level of identification, but very little access control because almost any valid server certificate could be authenticated. If you want to trust only a server certificate that has been signed by a private PKI or set of private PKIs, F5 recommends that you create and install a custom certificate bundle that contains the private PKI certificates, including the CA that directly signed your servers' certificates. You can then select the new certificate bundle in the **Trusted Certificate Authorities** setting. For information about creating a custom certificate bundle, refer to SOL13302: Configuring the BIG-IP system to use an SSL chain certificate (11.x). **Important:** Configuring the **Trusted Certificate Authorities** setting has no effect on server validation of the SSL server certificate that the BIG-IP system presents when connecting to an SSL server. The SSL server is responsible for verifying the validity of the SSL client's certificate using its own Trusted Certificate store. If you need to specify a chain of trust to support the SSL server's verification of the BIG-IP system's client certificate, refer to your SSL server documentation for configuration details.
Certificate Revocation List (CRL)	The **Certificate Revocation List (CRL)** setting allows you to specify a CRL that the BIG-IP system should use to check revocation status of a certificate before the system authenticates a server. If you want to use a CRL, you must import it to the BIG-IP system. You can then select the name of the CRL file from the **Certificate Revocation List (CRL)** setting. For information about importing an SSL CRL file, refer to SOL14620: Managing SSL certificates for BIG-IP systems. Because CRLs can quickly become outdated, F5 recommends that you use either OCSP or CRLDP profiles for more robust and current verification functionality.

3.02 - Explain the effect of changing protocol settings

The Fast L4 Profile Type

Protocol Settings

Protocol profiles support parameters concerning timeouts in connection management.

Making changes to these profiles directly affects the behaviors connections at layer 4 for the associated virtual server. If the protocol profile is currently being used and you make a change to the profile it can break the existing layer 4 connections using that profile. Making a change to a parent protocol profile can not only affect the connections using that profile where it is assigned but also any connections using a child of that parent profile.

All virtual servers have at least one protocol profile associated with them. While the HTTP Class is a protocol profile for configuration purposes, it cannot be the sole profile of any virtual server and must always be combined with both the TCP and HTTP profile.

The protocol profiles types are:

- Fast L4
- Fast HTTP
- HTTP Class
- TCP
- UDP
- SCTP

For each protocol profile type, Local Traffic Manager provides a pre-configured profile with default settings. In most cases, you can use these default profiles as is. If you want to change these settings, you can configure protocol profile settings when you create a profile, or after profile creation by modifying the profiles settings.

You can see all of the settings for each of the protocol profiles types below. It is quite long and understanding the description will help you to understand the impact.

The Fast L4 profile type

The purpose of a Fast L4 profile is to help you manage Layer 4 traffic more efficiently. When you assign a Fast L4 profile to a virtual server, the Packet Velocity® ASIC (PVA) hardware acceleration within the BIG-IP system can process some or all of the Layer 4 traffic passing through the system. By offloading Layer 4 processing to

the PVA hardware acceleration, the BIG-IP system can increase performance and throughput for basic routing functions (Layer 4) and application switching (Layer 7).

You can use a Fast L4 profile with these types of virtual servers: Performance (Layer 4), Forwarding (Layer 2), and Forwarding (IP).

This table lists and describes the settings of a Fast L4 profile.

Settings of a Fast L4 profile

Setting	Description	Default Value
Name	This setting specifies a unique name for the profile.	No default value
Parent Profile	This setting specifies the profile that you want to use as the parent profile. Your new profile inherits all non-custom settings and values from the parent profile specified.	fastL4
Reset on Timeout	If this setting is enabled and a TCP connection exceeds the timeout value for idle connections, the BIG-IP system sends a reset in addition to deleting the connection.	Enabled
Reassemble IP Fragments	If this setting is enabled, the BIG-IP system reassembles IP fragments.	Disabled
Idle Timeout	This setting specifies the number of seconds that a connection is idle before the connection is eligible for deletion. For background information on setting idle timeout values, see Chapter 1, Introducing BIG-IP Local Traffic Manager.	300
TCP Handshake Timeout	**Specify**: Specifies the acceptable duration for a TCP handshake, that is, the maximum idle time between a client SYN and a client ACK. If the TCP handshake takes longer than the timeout, the system automatically closes the connection. **Disabled**: Specifies that the system does not apply a timeout to a TCP handshake. **Indefinite**: Specifies that the acceptable duration for a TCP handshake is indefinite.	5
Max Segment Size Override	Overrides the maximum segment size (MSS), which is 1460. Possible values are: **Disabled**: Specifies that you want the maximum segment size to remain at 1460. **Specify**: Permits you to override the maximum segment size (1460) by specifying a number. Note that specifying a 0 value is equivalent to retaining the default value (**Disabled**).	Disabled

Setting	Description	Default Value
PVA Acceleration	This setting specifies the maximum acceleration mode that you prefer the system to use. Note that depending on the virtual server configuration, the system might or might not accelerate traffic in this mode. Possible values are **Full**, **Assisted**, or **None**. Additional information on this setting follows this table.	Full
IP ToS to Client	This setting specifies the Type of Service level that the BIG-IP system assigns to IP packets when sending them to clients.	Pass Through
IP ToS to Server	This setting specifies the Type of Service level that the BIG-IP system assigns to IP packets when sending them to servers	Pass Through
Link QoS to Client	This setting specifies the Quality of Service level that the BIG-IP system assigns to IP packets when sending them to clients.	Pass Through
Link QoS to Server	This setting specifies the Quality of Service level that the BIG-IP system assigns to IP packets when sending them to servers.	Pass Through
TCP Timestamp Mode	Specifies the action that the BIG-IP system should take on TCP timestamps. Possible values are: **Preserve**, **Strip**, and **Rewrite**.	Preserve
TCP Window Scale Mode	Specifies the action that the BIG-IP system should take on TCP windows. Possible values are: **Preserve** and **Strip**.	Preserve
Generate Internal Sequence Numbers	Enables the BIG-IP system to generate its own sequence numbers for SYN packets, according to RFC 1948.	Disabled
Strip Sack OK	Enables the BIG-IP system to block a TCP SackOK option from passing to the server on an initiating SYN.	Disabled
RTT from Client	Specifies that the BIG-IP system should use TCP timestamp options to measure the round-trip time to the client.	Disabled
RTT from Server	Specifies that the BIG-IP system should use TCP timestamp options to measure the round-trip time to the server.	Disabled
Loose Initiation	Specifies, when checked (enabled), that the system initializes a connection when it receives any TCP packet, rather that requiring a SYN packet for connection initiation. The default is disabled. We recommend that if you enable the **Loose Initiation** setting, you also enable the **Loose Close** setting. **Important:** Enabling loose initiation can permit stray packets to pass through the system. This can pose a security risk and reduce system performance.	Disabled
Loose Close	Specifies, when checked (enabled), that the system closes a loosely-initiated connection when the system receives the first FIN packet from either the client or the server.	Disabled

Setting	Description	Default Value
TCP Close Timeout	Specifies the length of time in seconds that a connection can remain idle before deletion, once the system receives a CLOSE packet for that connection. The **TCP Close Timeout** value must be less than the **Idle Timeout** value. Also, for the **TCP Close Timeout** value to be valid, you must have the **Loose Close** setting enabled.	5
TCP Keep Alive Interval	Specifies the keep-alive probe interval, in seconds.	Disabled
Hardware SYN Cookie Protection	Enables or disables hardware SYN cookie protection when PVA10 is present on the system. This feature is available on certain hardware platforms only.	Disabled
Software SYN Cookie Protection	Enables or disables software SYN cookie protection when PVA10 is not present on the system.	Disabled

PVA hardware acceleration

Once you implement a Fast L4 profile, Local Traffic Manager automatically selects the most efficient PVA hardware acceleration mode for Layer 4 traffic. Possible modes are Full, Assisted, and None.

The particular hardware acceleration mode that Local Traffic Manager selects depends on these factors:

- The Fast L4 profile settings

  The mode that the BIG-IP selects is influenced by the way that you configure the settings of the Fast L4 profile.

- The virtual server configuratioThe mode that Local Traffic Manager selects is influenced by the specific features that you assigned to the virtual server (such as pools, SNAT pools, and iRules).

- A monitor assigned to associated nodes

  For full PVA acceleration, you must assign monitors to the relevant nodes.

- The value of the PVA Acceleration setting

  The PVA Acceleration setting in the Fast L4 profile defines the maximum amount of hardware acceleration that you want to allow, for Layer 4 traffic passing through the virtual server. Therefore, if you set the value to:

  Full: The system can set hardware acceleration to any of the three modes (Full, Assisted, or None), depending on the virtual server configuration. This is the default value.

Assisted: The system can set hardware acceleration to either Assisted or None mode, depending on the virtual server configuration.

None: The system does not perform hardware acceleration.

Depending on the current mode to which hardware acceleration is automatically set, Local Traffic Manager accelerates Layer 4 traffic as described in the following table.

Effect of PVA hardware acceleration mode on Layer 4 traffic

Hardware Acceleration Mode	Result
Full	The hardware acceleration processes all Layer 4 traffic. Layer 4 traffic is not managed through the use of BIG-IP software features. In this case, Local Traffic Manager treats client-side and server-side packets as part of the same connection. An example of using hardware acceleration in Full mode is when you want to load balance Layer 4 traffic to two servers, using the Round Robin load balancing method, with no session persistence or iRules.
Assisted	Local Traffic Manager load balances all SYN packets, while the hardware acceleration assists with the remaining packets, including the tearing down of connections.
None	The hardware acceleration does not process any Layer 4 traffic. Instead, the BIG-IP application manages all Layer 4 traffic.

The Fast HTTP profile type

The Fast HTTP profile is a configuration tool designed to speed up certain types of HTTP connections. This profile combines selected features from the TCP, HTTP, and OneConnect profiles into a single profile that is optimized for the best possible network performance. When you associate this profile with a virtual server, the virtual server processes traffic packet-by-packet, and at a significantly higher speed.

You might consider using a Fast HTTP profile when:

- You do not need features such as remote server authentication, SSL traffic management, and TCP optimizations, nor HTTP features such as data compression, pipelining, and RAM Cache.
- You do not need to maintain source IP addresses.
- You want to reduce the number of connections that are opened to the destination servers.
- The destination servers support connection persistence, that is, HTTP/1.1, or HTTP/1.0 with Keep-Alive headers. Note that IIS servers support connection persistence by default.

- You need basic iRule support only (such as limited Layer 4 support and limited HTTP header operations). For example, you can use the iRule events CLIENT_ACCEPTED, SERVER_CONNECTED, and HTTP_REQUEST.

A significant benefit of using a Fast HTTP profile is the way in which the profile supports connection persistence. Using a Fast HTTP profile ensures that for client requests, Local Traffic Manager can transform or add an HTTP Connection header to keep connections open. Using the profile also ensures that Local Traffic Manager pools any open server-side connections. This support for connection persistence can greatly reduce the load on destination servers by removing much of the overhead caused by the opening and closing of connections.

> **Note:** The Fast HTTP profile is incompatible with all other profile types. Also, you cannot use this profile type in conjunction with VLAN groups, or with the IPv6 address format.

You can use the default fasthttp profile as is, or create a custom Fast HTTP profile. The following table lists and describes the settings of the Fast HTTP profile.

Settings of a Fast HTTP profile

Setting	Description	Default Value
Name	Specifies a unique name for the profile.	No default value
Parent Profile	Specifies the profile that you want to use as the parent profile. Your new profile inherits all non-custom settings and values from the parent profile specified.	fasthttp
Reset on Timeout	Specifies, when checked (enabled), that the system sends a TCP RESET packet when a connection times out, and deletes the connection.	Enabled (Checked)
Idle Timeout	This setting specifies the number of seconds that a connection is idle before the connection flow is eligible for deletion because it has no traffic. Possible values are: **Specify**, **Immediate**, and **Indefinite**. For background information on setting idle timeout values, see Chapter 1, Introducing BIG-IP Local Traffic Manager.	300
Maximum Segment Size Override	Specifies a maximum segment size (MSS) override for server-side connections. The default setting is **0**, which corresponds to an MSS of **1460**. To override this size, you can specify any integer between **536** and **1460**.	0
Client Close Timeout	Specifies the number of seconds after which the system closes a client connection, when the system either receives a client FIN packet or sends a FIN packet to the client. This setting overrides the **Idle Timeout** setting. Possible values are: **Specify**, **Immediate**, and **Indefinite**. For more information, see the online help.	5

Setting	Description	Default Value
Server Close Timeout	Specifies the number of seconds after which the system closes a client connection, when the system either receives a server FIN packet or sends a FIN packet to the server. This setting overrides the **Idle Timeout** setting. Possible values are: **Specify**, **Immediate**, and **Indefinite**. For more information, see the online help.	5
Unclean Shutdown	Specifies how the system handles closing connections. Possible values are: **Disabled**, **Enabled**, and **Fast**. For more information, see the online help.	Disabled
Force HTTP 1.0 Response	Specifies, when checked (enabled), that the server sends responses to clients in the HTTP/1.0 format. This effectively disables client chunking and pipelining.	Disabled (Cleared)
Maximum Pool Size	Specifies the maximum number of connections a load balancing pool can accept. A setting of 0 specifies that there is no maximum; that is, a pool can accept an unlimited number of connections.	2048
Minimum Pool Size	Specifies the minimum number of connections that a load balancing pool can accept. A setting of 0 specifies that there is no minimum.	0
Ramp-Up Increment	Specifies the increment in which the system makes additional connections available, when all available connections are in use.	4
Maximum Reuse	Specifies the maximum number of times that the system can re-use a current connection.	0
Idle Timeout Override	Specifies the number of seconds after which a server-side connection in a pool is eligible for deletion, when the connection has no traffic. This setting overrides the **Idle Timeout** setting. Possible values are: **Specify**, **Disabled**, and **Indefinite**. For more information, see the online help.	Disabled
Replenish	Specifies whether the BIG-IP system should maintain a steady-state maximum number of back-end connections. If you disable this setting, the system does not keep a steady-state maximum of connections to the back end, unless the number of connections to the pool drops below the value specified in the **Minimum Pool Size** setting.	Enabled (Checked)
Parse Requests	Specifies, when checked (enabled), that the system parses the HTTP data in the connection stream. Note that if you are using a Fast HTTP profile for non-HTTP traffic, you should disable this setting to shield against dynamic denial-of-service (DDOS) attacks.	Enabled (Checked)
Maximum Header Size	Specifies the maximum amount of HTTP header data that the system buffers before making a load balancing decision.	32768

Setting	Description	Default Value
Maximum Requests	Specifies the maximum number of requests that the system allows for a single client-side connection. When the specified limit is reached, the final response contains a **Connection: close** header is followed by the closing of the connection. The default setting of 0 means that the system allows an infinite number of requests per client-side connection.	0
Insert XForwarded For	Specifies whether the system inserts the **XForwarded For**: header in an HTTP request with the client IP address, to use with connection pooling. Possible settings are **Enabled** and **Disabled**. For more information, see the online help.	Disabled
Request Header Insert	Specifies a string that the system inserts as a header in an HTTP request. If the header exists already, the system does not replace it.	No default value

The HTTP Class profile type

An HTTP Class profile is a configuration tool that you can use to classify HTTP traffic. When you classify traffic, you forward traffic to a destination based on an examination of traffic headers or content. Use of an HTTP Class profile is an efficient way for Local Traffic Manager to classify traffic based on criteria that you specify. Although you can perform these same traffic-classification functions using the iRules feature, using an HTTP Class profile simplifies this process.

The destination you specify can be either a load balancing pool or a URL. To classify HTTP traffic, you configure an HTTP Class profile to specify strings that match a list type. The list types that you can use for string matching are:

- Host names
- URIs
- Headers
- Cookies

The string that you can match to one of these lists can be either a pattern string or a regular expression.

Once Local Traffic Manager matches the string to the corresponding list type, the system can send the traffic to a pool that you specify. Alternatively, you can create an HTTP Class profile that forwards a client request from the targeted HTTP virtual server to an HTTPS virtual server instead of to a pool.

The following table lists and describes the settings of an HTTP Class profile.

Settings of an HTTP Class profile

Setting	Description	Default Value
Name	Specifies a unique name for the profile.	No default value
Parent Profile	Specifies the profile that you want to use as the parent profile. Your new profile inherits all non-custom settings and values from the parent profile specified.	httpclass
Application Security	Specifies that you want a virtual server to forward traffic to the Application Security ManagerTM application. In this case, the HTTP Class profile is the equivalent of an Application Security Manager application security class. This setting appears only when Application Security Manager is licensed on the BIG-IP system.	Disabled (Cleared)
WebAccelerator	Specifies that you want a virtual server to forward traffic to the WebAcceleratorTM application. This setting appears only when WebAccelerator is licensed on the BIG-IP system.	Disabled (Cleared)
Hosts	Specifies whether the host names used as criteria for routing HTTP requests constitute all hosts or individual hosts that you specify. A value of **Match All** directs the system to forward HTTP requests from all hosts. A value of **Match Only** directs the system to forward HTTP requests based on only those hosts you specify.	Match All
Host List	Specifies individual host names to be used as criteria for routing HTTP requests. Using the **Entry Type** list, you must also identify each host name as either a pattern string or a regular expression. This setting appears only when the value of **Hosts** is **Match Only**. **Note:** When you use pattern strings, this list type is case-sensitive. For more information, see The HTTP Class profile type.	No default value
URI Paths	Specifies whether the URIs used as criteria for routing HTTP requests constitute all URIs or individual URIs that you specify. A value of **Match All** directs the system to forward HTTP requests from all URIs. A value of **Match Only** directs the system to forward HTTP requests based on only those URIs you specify.	Match All
URI List	Specifies individual URI paths to be used as criteria for routing HTTP requests. Using the Entry Type list, you must also identify each URI as either a pattern string or a regular expression. This setting appears only when the value of **URI Paths** is **Match Only**. **Note:** When you use pattern strings, this list type is case-sensitive. For more information, see The HTTP Class profile type.	No default value

Setting	Description	Default Value
Headers	Specifies whether the headers and their values, used as criteria for routing HTTP requests constitute all headers or individual headers that you specify. A value of **Match All** directs the system to forward HTTP requests based on all headers. A value of **Match Only** directs the system to forward HTTP requests based on only those headers you specify.	Match All
Header List	Specifies individual headers and their values that the BIG-IP system uses as criteria for routing HTTP requests. Using the Entry Type list, you must also identify each header as either a pattern string or a regular expression. This setting appears only when the value of **Headers** is **Match Only**.	No default value
Cookies	Specifies whether cookies used as criteria for routing those requests constitute all cookies or individual cookies that you specify. A value of **Match All** directs the system to forward HTTP requests based on all cookies. A value of **Match Only** directs the system to forward HTTP requests based on only those cookies you specify.	Match All
Cookie List	Specifies individual cookies to be used as criteria for routing HTTP requests. Using the **Entry Type** list, you must also identify each cookie as either a pattern string or a regular expression. This setting appears only when the value of **Cookies** is **Match Only**.	No default value
Send To	Specifies the destination for HTTP traffic. Possible values are **None**, **Pool**, or **Redirect To**.	None
Pool	Specifies the name of the pool to which you want to send classified traffic. This setting appears only when the value of the **Send To** setting is **Pool**.	None
Redirect To Location	Specifies the URI to which the system should send the traffic. You use this setting when you want the profile to redirect the client request from an HTTP virtual server to an HTTPS virtual server, instead of to a pool. For example, you can create an HTTP virtual server with the URL **http://siterequest/**, to listen on port **80**. You can then assign an HTTP Class profile to the virtual server, to redirect client requests to the HTTPS virtual server, **https://siterequest/**. Note that the string you specify can be a Tcl expression, such as **https://[HTTP::host][HTTP::uri]**.	No default value
Rewrite URI	Specifies the Tcl expression that the system uses to rewrite the request URI that is forwarded to the server without sending an HTTP redirect to the client. Note that if you use static text for this setting instead of a Tcl expression, the system maps the specified URI for every incoming request. Also, you cannot use this setting if the value of the **Send To** setting is **Redirect To**.	No default value

The TCP profile type

TCP profiles are configuration tools that help you to manage TCP network traffic. Many of the configuration settings of TCP profiles are standard SYSCTL types of settings, while others are unique to Local Traffic Manager.

TCP profiles are important because they are required for implementing certain types of other profiles. For example, by implementing TCP, HTTP, and OneConnect profiles, along with a persistence profile and a remote authentication profile, you can take advantage of these traffic management features:

- Content spooling, to reduce server load
- OneConnect, to pool server-side connections
- Layer 7 session persistence, such as hash or cookie persistence
- iRules for managing HTTP traffic
- HTTP RAM Cache
- HTTP data compression
- HTTP pipelining
- Application authentication using a remote server
- Rewriting of HTTP redirections

Local Traffic Manager includes three specific TCP profiles:

- tcp

This is the default TCP profile.

- tcp-lan-optimized

The tcp-lan-optimized profile is a TCP-type profile. This profile is effectively a custom profile that Local Traffic Manager has already created for you, derived from the default tcp profile. This profile is useful for environments where a link has higher bandwidth and/or lower latency when paired with a slower link.

- tcp-wan-optimized

The tcp-wan-optimized profile is a TCP-type profile. This profile is effectively a custom profile that Local Traffic Manager has already created for you, derived from the default tcp profile. This profile is useful for environments where a link has lower bandwidth and/or higher latency when paired with a faster link.

The following table lists and describes the settings of the default tcp profile.

Settings of a TCP profile

Setting	Description	Default Value
Name	Specifies a unique name for the profile.	No default value
Parent Profile	Specifies the profile that you want to use as the parent profile. Your new profile inherits all non-custom settings and values from the parent profile specified.	tcp
Reset on Timeout	If this setting is enabled and a TCP connection exceeds the timeout value for idle connections, sends a reset in addition to deleting the connection.	Enabled (Checked)
Time Wait Recycle	Recycles the connection when a SYN packet is received in a TIME-WAIT state.	Enabled (Checked)
Delayed ACKs	If this setting is enabled, allows coalescing of multiple acknowledgement (ACK) responses.	Enabled (Checked)
Proxy Maximum Segment	Advertises the same maximum segment to the server as was negotiated with the client.	Disabled (Cleared)
Proxy Options	Advertises an option (such as timestamps) to the server only if it was negotiated with the client.	Disabled (Cleared)
Proxy Buffer Low	Specifies the proxy buffer level at which the receive window was opened.	4096
Proxy Buffer High	Specifies the proxy buffer level at which the receive window was closed.	16384
Idle Timeout	Specifies the number of seconds that a connection is idle before the connection is eligible for deletion. For background information on setting idle timeout values, see Chapter 1, Introducing BIG-IP Local Traffic Manager.	300

Setting	Description	Default Value
Zero Window Timeout	Specifies the length of time, in milliseconds, that the TCP connection can receive zero-length window probes before the system closes the connection. The timer starts when an effective window size becomes zero, and stops when the window size becomes greater than zero. If the timer elapses, the connection is terminated. This setting is useful for handling slow clients with small buffers, such as cell phones. Possible values are: **Specify**: Specifies a number of milliseconds that the TCP connection can receive zero-length window probes before the system closes the connection. **Indefinite**: Specifies that the system does not delete TCP connections based on zero-length window.	20000
Time Wait	Specifies the number of milliseconds that a connection is in a TIME-WAIT state before entering the CLOSED state.	2000
FIN Wait	Specifies the number of seconds that a connection is in the FIN-WAIT or CLOSING state before quitting. A value of 0 represents a term of forever (or until the metrics of the FIN state).	5
Close Wait	Specifies the number of seconds that a connection remains in a LAST-ACK state before quitting. A value of 0 represents a term of forever (or until the metrics of the FIN state).	5
Send Buffer	Causes the BIG-IP system to send the buffer size, which is specified in bytes.	32768
Receive Window	Causes the BIG-IP system to receive the window size, which is specified in bytes.	32768
Keep Alive Interval	Causes the BIG-IP system to keep alive the probe interval, which is specified in seconds.	1800
Maximum SYN Retransmissions	Specifies the maximum number of retransmissions of SYN segments that the BIG-IP system allows.	3
Maximum Segment Retransmissions	Specifies the maximum number of retransmissions of data segments that the BIG-IP system allows.	8
IP ToS	Specifies the Type of Service level that the BIG-IP system assigns to TCP packets when sending them to clients.	0
Link QoS	Specifies the Quality of Service level that the BIG-IP system assigns to TCP packets when sending them to clients.	0

Setting	Description	Default Value
Selective ACKs	Specifies, when checked (enabled), that the system processes data using selective ACKs whenever possible, to improve system performance. Enabling this setting improves packet flow in a lossy network because the system can acknowledge successfully received packets out of order. This is a negotiated option and is automatically disabled if not supported by a peer. **Note:** F5 recommends that you use the default value.	Enabled (Checked)
Extended Congestion Notification	Specifies, when checked (enabled), that the system uses the TCP flags **CWR** (congestion window reduction) and **ECE** (ECN-Echo) to notify its peer of congestion and congestion counter-measures. **Note:** F5 recommends that you use the default setting. When enabled, this setting can interfere with overall congestion calculations. The setting also allows for potential security issues, whereby an intermediate device can stimulate poor performance by spoofing **CWR** packets.	Disabled (Cleared)
Extensions for High Performance (RFC 1323)	Specifies, when checked (enabled), that the system uses the timestamp and window scaling extensions for TCP (as specified in RFC 1323) to enhance high-speed network performance. These options are used to help calculate the round trip time, as well as the available resources on a peer. They are fundamentally linked with congestion control. Also, these options are normally negotiated, and you should not need to disable them unless a network device or peer does not implement them correctly.	Enabled (Checked)
Limited Transmit Recovery	Specifies, when checked (enabled), that the system uses limited transmit recovery revisions for fast retransmits (as specified in RFC 3042), to reduce the recovery time for connections on a lossy network. Enabling this setting allows TCP to temporarily stretch the congestion window when first receiving a duplicate ACK packet. This in turn allows for faster retransmissions and a quicker recovery from the small congestion window. With this setting enabled, the aggressive transmit behavior is limited to the recovery period.	Enabled (Checked)

Setting	Description	Default Value
Slow Start	Specifies, when checked (enabled), that the system uses larger initial window sizes (as specified in RFC 3390) to help reduce round trip times. The setting ramps up the amount of data transmitted to a peer over a period of time. Enabling this setting avoids sudden and excessive congestion on the link. Also, the congestion metrics cache might provide historical data about the peer, allowing the slow start to be jump started. If you disable this setting, the system initializes the congestion window to the maximum window scale and attempts to transmit as much data as possible until congestion occurs. Consequently, in networks with unlimited bandwidth (such as directly-connected local peers), more data can initially be transmitted.	Enabled (Checked)
Deferred Accept	Specifies, when checked (enabled), that the system defers allocation of the connection chain context until the system has received the payload from the client. Enabling this setting is useful in dealing with 3-way handshake denial-of-service attacks.	Disabled (Cleared)
Verified Accept	When enabled, verifies that a server is available to accept the connection (by actually sending the server a SYN) before responding to the client's SYN with a SYN-ACK. (Normally, the BIG-IP system accepts the client's connection before selecting a server with which to communicate.)	Disabled (Cleared)
Bandwidth Delay	Specifies, when checked (enabled), that the system attempts to calculate the optimal bandwidth to use to the client, based on throughput and round-trip time, without exceeding the available bandwidth.	Enabled (Checked)
Nagles Algorithm	Specifies, when checked (enabled), that the system applies Nagle's algorithm to reduce the number of short segments on the network. When the system receives packets that are less than the maximum segment size (MSS), the packets are coalesced until the peer has sent the ACK packet for the previous segment. This helps to reduce congestion by creating fewer packets on the network. Note that enabling this setting for interactive protocols such as Telnet might cause degradation on high-latency networks.	Enabled (Checked)
Acknowledge on Push	Specifies, when enabled, significantly improved performance to Windows® and Mac OS peers who are writing out on a very small send buffer.	Disabled (Cleared)
MD5 Signature	Specifies, when enabled, to use RFC2385 TCP-MD5 signatures to protect TCP traffic against intermediate tampering.	Disabled (Cleared)
MD5 Signature Passphrase	Specifies, when enabled, a plaintext passphrase which may be between 1 and 80 characters in length, and is used in a shared-secret scheme to implement the spoof-prevention parts of RFC2385.	No default value

Setting	Description	Default Value
Congestion Control	Specifies the congestion control mechanism that the BIG-IP system is to use. Possible values are: **None**—No congestion control algorithm implemented. With you choose this value, any congestion will result in lost packets and potentially long recovery stalls during large data transfers. **High Speed**—A more aggressive, loss-based algorithm. This algorithm improves on the behavior of the New Reno algorithm by progressively switching from the New Reno algorithm to the Scalable algorithm, based on the size of the congestion window. This allows the algorithm to make more aggressive changes when the window is small and make more conservative changes when the window is already large. **New Reno**—A modification to the Reno algorithm that responds to partial acknowledgements when selective acknowledgements (SACKs) are unavailable. This algorithm sends missing data and exits the recovery period more aggressively than does the Reno algorithm. The New Reno algorithm produces reasonable results for scaling the window in mixed environments. **Reno**—An implementation of the TCP Fast Recovery algorithm, based on the implementation in the BSD Reno release. During the slow-start period, this algorithm initially increases the congestion window exponentially. **Scalable**—A TCP algorithm modification that adds a scalable, delay-based and loss-based component into the Reno algorithm. This algorithm improves on the behavior of the New Reno algorithm. The algorithm is more tolerant of partial losses; it cuts back and increases the congestion window more conservatively.	High Speed
Congestion Metrics Cache	Specifies, when checked (enabled), that the system uses a cache for storing congestion metrics. Subsequently, because these metrics are already known and cached, the initial slow-start ramp for previously-encountered peers improves.	Enabled (Checked)
Appropriate Byte Counting (RFC 3465)	Increases the congestion window by basing the increase amount on the number of previously unacknowledged bytes that each ACK covers. **Note:** F5 recommends that you use the default setting. When this setting is disabled, in situations with lost ACK packets, the congestion window remains small for a longer period of time.	Enabled (Checked)
D-SACK (RFC 2883)	Specifies the use of the **Selective ACKs** (SACK) option to acknowledge duplicate segments. If a peer does not send duplicate segments, the system disables SACK processing altogether. Note that when enabled, this setting requires more processing, to always populate the SACK with all duplicate segments.	Disabled (Cleared)

Setting	Description	Default Value
Packet Lost Ignore Rate	Specifies the threshold of packets lost per million at which the system performs congestion control. Valid values range from 0 to 1,000,000. The default is 0, meaning the system performs congestion control if any packet loss occurs. If you set the ignore rate to 10 and packet loss for a TCP connection is greater than 10 per million, congestion control occurs.	0
Packet Lost Ignore Burst	Specifies the probability of performing congestion control when multiple packets are lost, even if the value of the **Packet Lost Ignore Rate setting** was not exceeded. Valid values range from **0** to **4,294,967,295**. A value of **0** means that the system performs congestion control if any packets are lost. Higher values decrease the chance of performing congestion control.	0
Initial Congestion Window Size	Specifies the initial congestion window size for connections to this destination. Actual window size is this value multiplied by the MSS (Maximum Segment Size) for the same connection. The default is 0 (zero), meaning that the system uses the values specified in RFC2414. Valid values range from 0 to 16.	0
Initial Receive Window Size	Specifies the initial receive window size for connections to this destination. Actual window size is this value multiplied by the MSS (Maximum Segment Size) for the same connection. The default is 0 (zero), meaning that the system uses the Slow Start value. Valid values range from 0 to 16.	0
Initial Retransmission Timeout Base Multiplier for SYN Retransmission	Specifies the initial RTO (Retransmission TimeOut) base multiplier for SYN retransmissions. The default is 0 (zero). This value is modified by the exponential backoff table, which selects the interval for subsequent retransmissions.	0
Delay Window Control	Specifies that the system will use an estimate of queueing delay as a measure of congestion to control, in addition to the normal loss-based control, the amount of data sent.	Disabled

The UDP profile type

The UDP profile is a configuration tool for managing UDP network traffic.

Because the BIG-IP system supports the OpenSSL implementation of datagram Transport Layer Security (TLS), you can optionally assign both a UDP and a Client SSL profile to certain types of virtual servers.

The following table lists and describes the settings of a UDP profile.

Settings of a UDP profile

Setting	Description	Default Value
Name	This setting specifies a unique name for the profile.	No default value
Parent Profile	This setting specifies the profile that you want to use as the parent profile. Your new profile inherits all non-custom settings and values from the parent profile specified.	udp
Idle Timeout	This setting specifies the number of seconds that a connection is idle before the connection flow is eligible for deletion. For background information on setting idle timeout values, see Chapter 1, Introducing BIG-IP Local Traffic Manager.	60
IP ToS	This setting specifies the Type of Service level that the BIG-IP system assigns to UDP packets when sending them to clients.	0
Link QoS	This setting specifies the Quality of Service level that the BIG-IP system assigns to UDP packets when sending them to clients.	0
Datagram LB	This setting specifies, when checked (enabled), that the system load balances UDP traffic packet-by-packet.	Disabled (Unchecked)
Allow No Payload	This setting specifies, when checked (enabled), that the system passes datagrams that contain header information, but no essential data.	Disabled (Unchecked)
PVA Acceleration	This setting specifies the maximum acceleration mode that you prefer the system to use. Note that depending on the virtual server configuration, the system might or might not accelerate traffic in this mode. Possible values are **Full**, **Assisted**, or **None**. Additional information on this setting follows this table.	Full

The SCTP profile type

Local Traffic Manager includes a profile type that you can use to manage Stream Control Transmission Protocol (SCTP) traffic. Stream Control Transmission Protocol (SCTP) is a general-purpose, industry-standard transport protocol, designed for message-oriented applications that transport signaling data. The design of

SCTP includes appropriate congestion avoidance behavior, as well as resistance to flooding and masquerade attacks.

Unlike TCP, SCTP includes the ability to support several streams within a connection. While a TCP stream refers to a sequence of bytes, an SCTP stream represents a sequence of messages.

You can use SCTP as the transport protocol for applications that require monitoring and detection of session loss. For such applications, the SCTP mechanisms to detect session failure actively monitor the connectivity of a session.

The following table lists and describes the settings of an SCTP profile.

Settings of an SCTP profile

Setting	Description	Default Value
Name	Specifies a unique name for the profile.	No default value
Parent Profile	Specifies the profile that you want to use as the parent profile. Your new profile inherits all non-custom settings and values from the parent profile specified.	tcp
Receive Ordered	If this setting is enabled, the system delivers messages to an upper layer, in order.	Enabled (Checked)
Send Partial	If this setting is enabled, the system accepts a partial amount of application data.	Enabled (Checked)
TCP Shutdown	If this setting is enabled, SCTP instances emulate TCP closing. After receiving a SHUTDOWN message from an upper-layer user process, an SCTP instance initiates a graceful shutdown, by sending a SHUTDOWN chunk.	Enabled (Checked)
Reset on Timeout	If this setting is enabled and an SCTP connection exceeds the timeout value for idle connections, the system sends a reset in addition to deleting the connection.	Enabled (Checked)
Out Streams	Specifies the number of outbound streams that you want the chunk to request.	2
In Streams	Specifies the number of inbound streams that you want the chunk to request.	2
Send Buffer	Causes the BIG-IP system to send the buffer size, in bytes.	65536
Receive Window	Specifies the number of bytes that a sender can transmit without receiving an acknowledgment (ACK).	65535

Setting	Description	Default Value
Transmit Chunks	Specifies the number of transmit chunks allowed in buffer.	256
Receive Chunks	Specifies the number of receive chunks allowed in buffer.	256
Cookie Expiration	Specifies the valid duration of a cookie, in seconds.	60
Maximum Initial Retransmit Limit	Specifies the maximum number of times that the system attempts to establish a connection.	4
Maximum Association Retransmit Limit	Specifies the maximum number of times that the system attempts to send data.	8
Proxy Buffer Low	Specifies the proxy buffer level at which the system opens the receive window.	4096
Proxy Buffer High	Specifies the proxy buffer level at which the system closes the receive window.	16384
Idle Timeout	Specifies the number of seconds that a connection is idle before the connection is eligible for deletion. For background information on setting idle timeout values, see Chapter 1, Introducing BIG-IP Local Traffic Manager.	300
Heartbeat Interval	Specifies the number of seconds to wait before sending a heartbeat chunk.	30
IP ToS to Peer	Specifies the Type of Service level that the BIG-IP system assigns to SCTP packets when sending them to a client.	0
Link QoS to Peer	Specifies the Quality of Service level that the BIG-IP system assigns to SCTP packets when sending them to a client.	0
Secret	Specifies the internal secret string used to calculate the key-hash method authentication code (HMAC) for cookie verification.	default

3.02 - Explain the use cases for the fast protocols (e.g. fastL4, fastHTTP)

Protocol Profiles

Fast L4

The purpose of a Fast L4 profile is to help you manage Layer 4 traffic more efficiently. When you assign a Fast L4 profile to a virtual server, the Packet Velocity® ASIC (PVA) hardware acceleration within the BIG-IP system can process some or all of the Layer 4 traffic passing through the system. By offloading Layer 4 processing to the PVA hardware acceleration, the BIG-IP system can increase performance and throughput for basic routing functions (Layer 4) and application switching (Layer 7).

You can use a Fast L4 profile with these types of virtual servers: Performance (Layer 4), Forwarding (Layer 2), and Forwarding (IP).

This profile will typically be used when there is no need to process the traffic above Layer 4.

Fast HTTP

The Fast HTTP profile is a configuration tool designed to speed up certain types of HTTP connections. This profile combines selected features from the TCP, HTTP, and OneConnect profiles into a single profile that is optimized for the best possible network performance. When you associate this profile with a virtual server, the virtual server processes traffic packet-by-packet, and at a significantly higher speed.

You might consider using a Fast HTTP profile when:

- You do not need features such as remote server authentication, SSL traffic management, and TCP optimizations, nor HTTP features such as data compression, pipelining, and RAM Cache.
- You do not need to maintain source IP addresses.
- You want to reduce the number of connections that are opened to the destination servers.
- The destination servers support connection persistence, that is, HTTP/1.1, or HTTP/1.0 with Keep-Alive headers. Note that IIS servers support connection persistence by default.
- You need basic iRule support only (such as limited Layer 4 support and limited HTTP header operations). For example, you can use the iRule events CLIENT_ACCEPTED, SERVER_CONNECTED, and HTTP_REQUEST.

A significant benefit of using a Fast HTTP profile is the way in which the profile supports connection persistence. Using a Fast HTTP profile ensures that for client requests, Local Traffic Manager can transform or

add an HTTP Connection header to keep connections open. Using the profile also ensures that Local Traffic Manager pools any open server-side connections. This support for connection persistence can greatly reduce the load on destination servers by removing much of the overhead caused by the opening and closing of connections.

> **Note:** The Fast HTTP profile is incompatible with all other profile types. Also, you cannot use this profile type in conjunction with VLAN groups, or with the IPv6 address format.

When writing iRules, you can specify a number of events and commands that the Fast HTTP profile supports.

You can use the default fasthttp profile as is, or create a custom Fast HTTP profile.

3.02 - Explain the persistence overrides

Introduction to Session Persistence Profiles

Persistence Overrides

Persistence is the nemesis of load balancing! When any type of persistence is needed it overrides the function of load balancing. When persistence is configured, the first inbound connection is load balanced to the best pool member resource according to the algorithm and availability status. All following client requests are then directed to the same pool member throughout the life of a session or during subsequent sessions.

Persistence can even be configured to Override Connection Limits in the profile settings. This setting says that the system will allow pool member connection limits to be overridden for persisted clients. Per-virtual connection limits remain hard limits and are not overridden.

3.02 - Describe the use of HTTP classes and profiles

Overview of the HTTP Class Profile

HTTP Classes

In BIG-IP 9.4.x through 11.3.0, the HTTP Class profile provides a way to classify HTTP traffic and perform an action, such as sending the traffic to a load-balancing pool, rewriting the URI, or forwarding traffic to a selected module, such as ASM.

> **Note:** Starting in BIG-IP 11.4.0, the HTTP Class profile is replaced by the Local Traffic Policies feature.

An HTTP Class profile allows you to sort selected HTTP traffic and perform an action based on the profile configuration. For example, you can send the traffic to a destination, such as a load balancing pool or rewrite the URI.

In addition, you can use the HTTP Class profile to forward traffic to a selected module such as ASM, or WebAccelerator (9.4.x - 10.x).

Beginning in BIG-IP WebAccelerator 11.0.0, web acceleration is enabled through the Web Acceleration profile.

To classify HTTP traffic, you configure an HTTP Class profile to specify strings that match a list type. The string that you can match to one of these lists can be either a pattern string or a regular expression. The list types are case-sensitive for pattern strings. For example, the system treats the pattern string www.f5.com differently from the pattern string www.F5.com. You can override this case-sensitivity by using the Linux regexp command.

F5 recommends using HTTP Class profiles when it is possible to classify HTTP traffic using simple strings or regex patterns. For more complex operations, you may need to use an iRule.

3.02 - Describe the link between iRules and statistics, iRules and stream, and iRule events and profiles

The Statistics Profile Type

iRules and Statistics

The Statistics profile provides user-defined statistical counters. Each profile contains 32 settings (Field1 through Field32), which define named counters. Using a cl-based iRule command, you can use the names to manipulate the counters while processing traffic.

For example, you can create a profile named my_stats, which assigns the counters tot_users, cur_users, and max_users to the profile settings Field1, Field2, and Field3 respectively. You can then write an iRule named track_users, and then assign the my_stats profile and the track_users iRule to a virtual server named stats-1.

Example of Statistics profile counters used in an iRule

```
profile stats my_stats {
defaults from stats
field1 tot_users
field2 cur_users
field3 max_users
}

rule track_users {
when CLIENT_ACCEPTED {
STATS::incr my_stats tot_users
STATS::setmax my_stats max_users [STATS::incr my_stats cur_users]
}
}

virtual stats-1 {
destination 10.10.55.66:http
ip protocol tcp
profile http my_stats tcp
pool pool1
rule track_users
}
```

In this example, the counter tot_users counts the total number of connections, the counter cur_users counts the current number of connections, and the counter max_users retains the largest value of the counter cur_users.

The Statistics Profile Type

iRules and Stream

Using an iRule to apply a Stream profile only when desired traffic is seen, can be a powerful way to use Stream profiles.

For example, when you want to scrub content; instead of collecting the HTTP response payloads, the iRule can uses the stream filter to replace the strings inline. This should be more efficient than buffering the payload with the HTTP::collect command.

Another Example, you may need to only look for a particular string within XML content and replace it with a different string. The string you are looking for may appear in multiple locations, not just in the XML content. Using a Stream profile to replace the string would cause the string to be replaced everywhere in the content

not just in the XML content. Using an iRule to run when it sees XML content and then apply the stream will give you the power you need to solve the issue.

iRules

iRules and Profiles

When you are writing an iRule, you might want that iRule to recognize the value of a particular profile setting so that it can make a more-informed traffic management decision. Fortunately, the iRules feature includes a command that is specifically designed to read the value of profile settings that you specify within the iRule.

Not only can iRules read the values of profile settings, but they can also override values for certain settings. This means that you can apply configuration values to individual connections that differ from the values Local Traffic Manager applies to most connections passing through a virtual server.

The Profile Command

The iRules feature includes a command called PROFILE. When you specify the PROFILE command in an iRule and name a profile type and setting, the iRule reads the value of that particular profile setting. To do this, the iRule finds the named profile type that is assigned to the virtual server and reads the value of the setting that you specified in the PROFILE command sequence. The iRule can then use this information to manage traffic.

For example, you can specify the command `PROFILE::tcp idle_timeout` within your iRule. Local Traffic Manager then finds the TCP profile that is assigned to the virtual server (for example, my_tcp) and queries for the value that you assigned to the Idle Timeout setting.

> **Note:** If an iRule references a profile, Local Traffic Manager processes this type of iRule last, regardless of its order in the list of iRules assigned to a virtual server.

Commands That Override Profile Settings

Some of the iRule commands for querying and manipulating header and content data have equivalent settings within various profiles. When you use those commands in an iRule, and an event triggers that iRule, Local Traffic Manager overrides the values of those profile settings, using the value specified within the iRule instead.

For example, an HTTP profile might specify a certain buffer size to use for compressing HTTP data, but you might want to specify a different buffer size for a particular type of HTTP connection. In this case, you can include the command HTTP::compress_buffer_size in your iRule, specifying a different value than the value in the profile.

3.02 - Describe the link between iRules and persistence

Hash Persistence

iRules and Persistence

You can assign a persistence profile from within an iRule as you can most profiles. But there is a persistence method that specifically relies on an iRule to fuction. This type of persistence is known as Universal Persistence.

Universal Persistence

Included in Local Traffic Managers Universal Inspection Engine (UIE) is a set of functions that you can specify within BIG-IP system iRules to direct traffic in more granular ways. Using these iRule functions, you can write expressions that direct traffic based on content data, or direct traffic to a specific member of a pool.

Universal persistence takes this iRules feature one step further, by allowing you to use the iRule persist uie command to implement persistence for sessions based on content data, or based on connections to a specific member of a pool. Universal persistence does this by defining some sequence of bytes to use as a session identifier.

To use iRule expressions for persistence, a universal persistence profile includes a setting that specifies the name of the iRule containing the expression.

Sample iRule for universal persistence

```
rule my_persist_irule {
when HTTP_REQUEST { persist uie [HTTP::header myheader] }
}
```

Unlike hash persistence, which uses a hash of the data as the persistence key, universal persistence uses the data itself as the persistence key.

> **Note:** F5 Networks recommends that you configure a OneConnect profile in addition to the Universal profile, to ensure that Local Traffic Manager load balances HTTP requests correctly.

3.02 - Describe hashing persistence methods

Hash Persistence

Hash persistence

Hash persistence allows you to create a persistence hash based on an existing iRule that uses the persist iRule command. Using hash persistence is the same as using universal persistence, except that with hash persistence, the resulting persistence key is a hash of the data, rather than the data itself.

Sample iRule for hash persistence

```
rule my_persist_irule {
when HTTP_REQUEST { persist hash [HTTP::header myheader] }
}
```

Note that if you use hash persistence and Local Traffic Manager cannot find an entry in the persistence table for a connection, and the system has not yet chosen a pool member due to fallback persistence, then the system uses the hash value, rather than the specified load balancing method, to select the pool member.

For example, if the persistence table contains no entry for the hash value 2356372769, and the number of active nodes in the pool remains the same, then a session with that hash value for persistence is always persisted to node 10.10.10.190 (assuming that the node is active).

Cookie Hash Method

If you specify the Cookie Hash method, the hash method consistently maps a cookie value to a specific node. When the client returns to the site, Local Traffic Manager uses the cookie information to return the client to a given node. With this method, the web server must generate the cookie; Local Traffic Manager does not create the cookie automatically as it does when you use the HTTP Cookie Insert method.

3.02 - Describe the cookie persistence options

Cookie Persistence

Cookie Persistence

You can set up Local Traffic Manager to use HTTP cookie persistence. Cookie persistence uses an HTTP cookie stored on a client's computer to allow the client to reconnect to the same pool member previously visited at a web site.

There are four methods of cookie persistence available:

- HTTP Cookie Insert method
- HTTP Cookie Rewrite method
- HTTP Cookie Passive method
- Cookie Hash method

The method you choose to use affects how Local Traffic Manager returns the cookie when returning the cookie to the client.

HTTP Cookie Insert method

If you specify HTTP Cookie Insert method within the profile, the information about the server to which the client connects is inserted in the header of the HTTP response from the server as a cookie. The cookie is named BIGipServer<pool_name>, and it includes the address and port of the server handling the connection. The expiration date for the cookie is set based on the timeout configured on the BIG-IP system. HTTP Cookie Insert is the default value for the Cookie Method setting.

> **Tip:** You can assign this type of profile to a Performance (HTTP) type of virtual server.

HTTP Cookie Rewrite method

If you specify HTTP Cookie Rewrite method, Local Traffic Manager intercepts a Set-Cookie header, named BIGipCookie, sent from the server to the client, and overwrites the name and value of the cookie. The new cookie is named BIGipServer<pool_name> and it includes the address and port of the server handling the connection.

> **Important:** We recommend that you use this method instead of the HTTP Cookie Passive method whenever possible.

The HTTP Cookie Rewrite method requires you to set up the cookie created by the server. For the HTTP Cookie Rewrite method to succeed, there needs to be a blank cookie coming from the web server for Local Traffic Manager to rewrite. With Apache variants, the cookie can be added to every web page header by adding the following entry to the httpd.conf file:

```
Header add Set-Cookie BIGipCookie=00000000000000000000000000...
```

(The cookie must contain a total of 120 zeros.)

> **Note:** For backward compatibility, the blank cookie can contain only 75 zeros. However, cookies of this size do not allow you to use iRules and persistence together.

HTTP Cookie Passive method

If you specify the HTTP Cookie Passive method, Local Traffic Manager does not insert or search for blank Set-Cookie headers in the response from the server. This method does not try to set up the cookie. With this method, the server provides the cookie, formatted with the correct server information and timeout.

> **Important:** We recommend that you use the HTTP Cookie Rewrite method instead of the HTTP Cookie Passive method whenever possible.

For the HTTP Cookie Passive method to succeed, there needs to be a cookie coming from the web server with the appropriate server information in the cookie. Using the Configuration utility, you generate a template for the cookie string, with encoding automatically added, and then edit the template to create the actual cookie.

For example, the following string is a generated cookie template with the encoding automatically added, where [pool name] is the name of the pool that contains the server, 336260299 is the encoded server address, and 20480 is the encoded port:

Set-Cookie:BIGipServer[poolname]=336268299.20480.0000; expires=Sat, 01-Jan-2002 00:00:00 GMT; path=/

Cookie Hash method

If you specify the Cookie Hash method, the hash method consistently maps a cookie value to a specific node. When the client returns to the site, Local Traffic Manager uses the cookie information to return the client to a given node. With this method, the web server must generate the cookie; Local Traffic Manager does not create the cookie automatically as it does when you use the HTTP Cookie Insert method.

Cookie profile settings

To implement cookie persistence, you can either use the default cookie profile, or create a custom profile.

Settings of a Cookie persistence profile

Setting	Description	Default Value
Name	Specifies a unique name for the profile. This setting is required.	No default value
Persistence Type	Specifies the type of persistence. This setting is required.	Cookie
Cookie Method	Specifies the type of cookie processing that the BIG-IP system is to use.	HTTP Cookie Insert
Cookie Name	Specifies the name of the cookie that the BIG-IP system should look for or insert.	This value is autogenerated based on the pool name.
Expiration	Sets the expiration time of the cookie. Applies to the HTTP Cookie Insert and HTTP Cookie Rewrite methods only. When using the default (checked), the system uses the expiration time specified in the session cookie.	Enabled (Checked)
Hash Offset	With respect to Cookie persistence, this setting applies to the **Cookie Hash** method only.	0
Hash Length	With respect to Cookie persistence, this setting applies to the **Cookie Hash** method only.	0
Timeout	This setting applies to the **Cookie Hash** method only. The setting specifies the duration, in seconds, of a persistence entry.	180
Mirror Persistence	Specifies, when enabled (checked), that if the active unit goes into the standby mode, the system mirrors any persistence records to its peer. With respect to Cookie profiles, this setting applies to the **Cookie Hash** method only.	Disabled (Cleared)
Match Across Services	Specifies that all persistent connections from a client IP address that go to the same virtual IP address also go to the same node. With respect to Cookie profiles, this setting applies to the **Cookie Hash** method only.	Disabled (Cleared)
Match Across Virtual Servers	Specifies that all persistent connections from the same client IP address go to the same node. With respect to Cookie profiles, this setting applies to the **Cookie Hash** method only.	Disabled (Cleared)
Match Across Pools	Specifies that the BIG-IP system can use any pool that contains this persistence entry. With respect to Cookie profiles, this setting applies to the **Cookie Hash** method only.	Disabled (Cleared)
Override Connection Limit	Specifies, when checked (enabled), that the system allows you to specify that pool member connection limits are overridden for persisted clients. Per-virtual connection limits remain hard limits and are not overridden.	Disabled (Cleared)

3.02 - Determine which profiles are appropriate for a given application

Using Profiles with iRules

Profiles

The BIG-IP LTM system can manage application-specific network traffic in a variety of ways, depending on the protocols and services being used. For example, you can configure the LTM system to compress HTTP response data, or you can configure the system to authenticate SSL client certificates before passing requests on to a target server.

For each type of traffic that you want to manage, the LTM system contains configuration tools that you can use to intelligently control the behavior of that traffic. These tools are called profiles. A profile is a system-supplied configuration tool that enhances your capabilities for managing application-specific traffic. More specifically, a profile is an object that contains user-configurable settings, with default values, for controlling the behavior of a particular type of network traffic, such as HTTP connections. Using profiles enhances your control over managing network traffic, and makes traffic-management tasks easier and more efficient.

Appropriate Profiles

You can set up a virtual server to process traffic with simply an unmodified client and server side Protocol Profile and a pool for the destination. And the biggest decision you had to make was is the traffic TCP, UDP or SCTP based. But when the application you are working with needs advanced processing up to the application layer you will need a profile to make that happen.

As an administrator you will need to understand the type of application traffic with which you are working. So you can choose the matching protocol profiles that will allow the traffic to be processed correctly. The settings of those profiles will give you the ability to understand what may need to be done to the traffic as the BIG-IP LTM is processing it. The processing of the application level traffic is controlled by the associated application level profiles applied to the virtual server. Even if the processing you need to do will be completed with an iRule the events in the iRule may only be allowed if the correct profiles are applied.

So the short of it is, understand what protocols your applications are using and what functions your applications need to have done and pick those correlating profiles.

3.02 - Determine when an iRule is preferred over a profile or vice versa

iRule vs Profile Processing

iRule vs Profile

It would seem that manipulating packet contents is so easily accomplished by using an iRule that it is always the way to go. However, the code for TMM profiles is compiled in the TMM kernel, and has undergone strenuous testing & optimization to be as fast & reliable as possible, so if you can use a built-in profile option instead of an iRule, it will be more efficient.

The functions that are built into most profiles are the necessary or most common functions. If you need to do more than a profile can do for the traffic processing then an iRule is the way to go.

3.02 - Explain how to manipulate the packet contents using profiles

Manipulating Packet Contents Using Profiles

There are a few profiles that can manipulate packet contents for their purposes.

Ref: 1, Other Profiles.

The Stream Profile

The stream profile performs a “search and replace” for all occurrences of the string in the TCP data string, both requests and responses. If you wish to limit substitution to certain situations, you must define an iRule with appropriate STREAM commands. When a virtual server has a stream profile, the search and replace operation can be performed on both client requests (changing requests on the way to the servers) and server responses (changing responses on the way to the client). The stream profile performs search and replace on a packet-by-packet basis. While the TCP profile will adjust the IP header length, the stream profile will not adjust the HTTP Content-Length header. Also went an HTTP profile is associated with the virtual server, the stream profile processes the HTTP data portion of the packet, not HTTP headers

Note that list types are case-sensitive for pattern strings. For example, the system treats the pattern string www.f5.com differently from the pattern string www.F5.com. You can override this case sensitivity by using the Linux regexp command.

Profiles for Managing HTTP Traffic

The HTTP Profile

The HTTP profile has a setting that will insert an XForwarded-For header into an HTTP request, to use with connection pooling. This feature adds the IP address of the client as the value of the XForwarded-For header.

The setting is used to send the original client IP address to the web server so the server can populate its event logs with the original client IP address when traffic coming to the server is SNAT'd.

Cookie Persistence

The Cookie Persistence Profile

If you specify HTTP Cookie Insert method within the profile, the information about the server to which the client connects is inserted in the header of the HTTP response from the server as a cookie. The cookie is named BIGipServer<pool_name>, and it includes the address and port of the server handling the connection. The expiration date for the cookie is set based on the timeout configured on the BIG-IP system. HTTP Cookie Insert is the default value for the Cookie Method setting.

Each of these profile's functions or settings could be done within an iRule, but it is accomplishable without writing any code on the BIG-IP platform using these profiles.

Objective - 3.03 Determine the effect of traffic flow on LTM device performance and/or utilization

3.03 - Describe the effect of priority groups on load balancing

Pools

Effect of priority groups on load balancing

Priority Groups are designed to dynamically allocate additional pool members to the pool, as resources drop below the minimum active member setting. With priority group activation configured load balancing algorithms may not seem to behave as expected. Pool members that are not activated by the priority group will not receive any traffic until they become active, due to a higher priority group member failure. And as lower priority groups are brought online they are added to the pool algorithm as net new resources.

Depending on the algorithm that you are using for load balancing the load across the pool this can cause issues when a server is being added. For example: If least connections was the algorithm for the pool and the priority group "at least" setting was 2. If the pool lost members and was down to one remaining, the priority

group would enable the next lowest group of servers. This could cause the new servers that are now entering the pool with zero layer 4 connections to receive all new connections until those incoming servers have caught up to the exiting server's level of layer 4 connections. This may over whelm a server or make the pool seem as if it is not load balancing the traffic across the servers.

You can use the setting Slow Ramp Time to alleviate the risk of a server being over whelmed with all of the new connections in this scenario.

3.03 - Explain the effects of SNAT settings on pools

SNAT implementation

A SNAT is a configuration object (processing all traffic based on source VLAN and origin address) or is a setting within the configuration of each virtual server (processing all traffic passing through the virtual server) or a SNAT can be intelligently applied within an iRule applied to a virtual server. However there are settings to permit SNAT functions in other areas of the configuration. Within the settings of a pool configuration

Pools

SNATs and NATs

When configuring a pool, you can specifically disable any secure network address translations (SNATs) or network address translations (NATs) for any connections that use that pool. By default, these settings are enabled. You can change this setting on an existing pool by displaying the Properties screen for that pool.

One case in which you might want to configure a pool to disable SNAT or NAT connections is when you want the pool to disable SNAT or NAT connections for a specific service. In this case, you could create a separate pool to handle all connections for that service, and then disable the SNAT or NAT for that pool.

Allow SNAT setting

You can configure a pool so that SNATs are automatically enabled or disabled for any connections using that pool.

This setting's default value is Yes.

3.03 - Explain how persistence settings can override connection limits

Introduction to Session Persistence Profiles

It is understood that persistence will keep a users session on the same pool member according to the configuration of the type of persistence method that is being used. And if connection limits were set for that

pool member; it would also be understood that if that pool member reached its limit, no further connections would be allowed. This behavior would break the users session. To solve this type of problem the Override Connection Limit setting can be enabled in the persistence profile.

Override Connection Limit setting

This setting can be enabled in the persistence profile.

It specifies, when checked (enabled), that the system allows you to specify that pool member connection limits are overridden for persisted clients. Per-virtual connection limits remain hard limits and are not overridden.

This setting is not enabled by default.

3.03 - Describe the relationship between monitors and state

Troubleshooting Health Monitors

Both Monitors and State settings will determine if a node or pool member is accessible.

A monitor is an important BIG-IP feature that verifies connections to pool members or nodes. A health monitor is designed to report the status of a pool, pool member, or node on an ongoing basis, at a set interval. When a health monitor marks a pool, pool member, or node down, the BIG-IP system stops sending traffic to the device.

A failing or misconfigured health monitor may cause traffic management issues similar, but not limited, to the following:

- Connections to the virtual server are interrupted or fail.
- Web pages or applications fail to load or execute.
- Certain pool members or nodes receive more connections than others.

The previously mentioned symptoms may indicate that a health monitor is marking a pool, pool member, or node down indefinitely, or that a monitor is repeatedly marking a pool member or node down and then back up (often referred to as a bouncing pool member or node). For example, if a misconfigured health monitor constantly marks pool members down and then back up, connections to the virtual server may be interrupted or fail altogether. You will then need to determine whether the monitor is misconfigured, the device or application is failing, or some other factor is occurring that is causing the monitor to fail (such as network-related issue). The troubleshooting steps you take will depend on the monitor type and the observed symptoms.

3.03 - Describe the functionality of Action On Service Down

Overview of the Action On Service Down Feature

Action On Service Down

The Action On Service Down feature allows the BIG-IP system to choose another pool member and rebind the client connection to a new server connection if the target pool member becomes unavailable.

When a pool member fails to respond, as configured, to a health monitor, the system marks that pool member down, and continues to monitor it to determine when the member becomes available again. While a pool member is marked down, the system does not send any new connections to that pool member.

The Action On Service Down feature specifies how the system should respond to already-established connections when the target pool member becomes unavailable.

The available settings for this feature as follows:

None:

The BIG-IP system takes no action on existing connections, and removes the connection table entry based on the associated profile's idle timeout value. The BIG-IP system sends a TCP Reset (RST) or ICMP Unreachable once idle timeout is reached. This is the default setting.

This is the best option for most common scenarios, as this allows for endpoints to resume gracefully on their own. This may be a good choice for clients that transfer large amounts of data, as the pool member may recover itself before the connection is reset, allowing the large transfer to continue.

Reject:

The BIG-IP system sends RST or ICMP messages to reset active connections and removes them from the BIG-IP connection table.

This may be a good choice for clients that need to be notified of pool member state changes sooner than the configured idle timeout period for that virtual server. Once the target pool member is deemed unavailable, the BIG-IP system immediately alerts the client by resetting the connection, causing the client to attempt a new connection.

Drop:

The BIG-IP system silently removes the connection table entry.

You should carefully consider this option, as the client receives no feedback from the BIG-IP system regarding the connection state. However, this option works well for short-lived, connectionless protocols, such as UDP. For example, DNS queries.

Reselect:

The BIG-IP system manages established client connections by moving them to an alternate pool member without a connection teardown or setup.

This option is only appropriate for:

- Virtual servers with address and port translation disabled

  Note: This is default for FastL4 type virtual servers, such as network or wildcard forwarding.

- Transparent pool members, such as firewalls, routers, proxy servers, and cache servers

  Note: Transparent devices can forward packets to destinations without regard for the state of the connection.

- UDP virtual servers

  Note: When choosing the Reselect option for Action on Service Down, the BIG-IP system does not reform existing TCP connections, but continues to forward existing connections. If the back-end pool members are not transparent devices, and the virtual server has address translation enabled, all existing TCP connections sent to a pool member will likely reset due to the pool member having no record of these ongoing connections in its connection table. This is analogous to choosing the Reset action, except the pool members will be resetting the connections instead of the BIG-IP system. F5 highly recommends choosing a different Action on Service Down option, if you do not meet the above criteria for the Reselect option.

  Note: Services, such as HTTP require that the system establish a transport layer connection before transmitting HTTP messages. This is commonly referred to as a 3-way handshake, and is used by the client or server to establish communication options and to track requests or responses. When a server receives a request from a client without having established the transport layer connection, normal behavior is for the server to reject the connection by sending a TCP response with the RST flag set. For more information, refer to Internet Engineering Task Force (RFC 793), section Reset Generation. This link takes you to a resource outside of AskF5. The third party could remove the document without our knowledge.

3.03 - Describe the functionality of Priority Group Activation

Pools

Priority Group Activation

With the Priority Group Activation feature, you can specify the minimum number of members that must remain available in each priority group in order for traffic to remain confined to that group. This feature is used in tandem with the Priority Group feature for individual pool members.

If the number of available members assigned to the highest priority group drops below the minimum number that you specify, Local Traffic Manager distributes traffic to the next highest priority group, and so on. As members become available again in the higher group the Local Traffic Manager will disable the lower priority group as necessary.

The configuration shown in Figure 3.03a has three priority groups, 3, 2, and 1, with the Priority Group Activation value (shown as min active members) set to 2.

Figure 3.03a Sample pool configuration for priority load balancing

```
pool my_pool {
lb_mode fastest
min active members 2
member 10.12.10.7:80 priority 3
member 10.12.10.8:80 priority 3
member 10.12.10.9:80 priority 3
member 10.12.10.4:80 priority 2
member 10.12.10.5:80 priority 2
member 10.12.10.6:80 priority 2
member 10.12.10.1:80 priority 1
member 10.12.10.2:80 priority 1
member 10.12.10.3:80 priority 1
}
```

Connections are first distributed to all pool members with priority 3 (the highest priority group). If fewer than two priority 3 members are available, traffic is directed to the priority 2 members as well. If both the priority 3 group and the priority 2 group have fewer than two members available, traffic is directed to the priority 1 group. Local Traffic Manager continuously monitors the priority groups, and each time a higher priority group once again has the minimum number of available members, Local Traffic Manager limits traffic to that group.

3.03 - Describe the persistence across pools and services (e.g., Match Across Services, Match Across vs Match Across Pools)

Match Across Options for Session Persistence

Match Across To Solve Deeper Persistence Issues

The Match Across options specify that, regardless of the type of persistence you are implementing, you can specify the criteria that the BIG-IP system uses to send all requests from a client to the same pool member. The criteria are based on the virtual servers that are hosting the client connection.

Match Across Services

The Match Across Services option is used in the following two configurations:

- Configurations that have multiple virtual servers with the same IP address but have different services specified.
- Configurations that have pool members sharing the same address but have different services specified.

  Important: The Match Across Services option uses only the node IP address to find a persistence match in pools other than the one for which the persistence record was written. This deviation from the normal persistence matching behavior is required to accommodate the intended use cases for the feature to match even when the service port does not. Because of this lack of granularity, a pool containing multiple members with the same node address may result in inconsistent load balancing behavior. For this reason, F5 recommends that pools associated with virtual servers that are configured to use the Match Across Services option should not contain multiple members using the same node address.

A typical use of the Match Across Services feature is for combined HTTP/HTTPS support for the same site. Commerce sites are typically configured to allow customers to view and select merchandise using HTTP, but then the site switches to HTTPS when the customer begins the checkout process. The Match Across Services option is useful in this configuration as it allows the session information to be shared between the virtual servers and ensures that the client is directed to the same pool member.

The example, the configuration below shows that clients are load balanced to pool member `172.16.1.2:http`, and an entry is created in the persistence table when they first connect to virtual server `192.168.0.10:http`.

If the same clients connect to virtual server `192.168.0.10:https`, the BIG-IP system uses the persistence session information that was established with the initial connection, and directs the request to pool member `172.16.1.2:https`.

If the same clients connect to virtual server `192.168.0.20:http`, the request is load balanced according to the method specified by the pool, and a new persistence session is entered in the persistence table for tracking.

> **Note:** This behavior occurs because the third virtual server does not share the same address as the other two that are configured.

If the client connects to a different virtual server that does not utilize persistence, that connection will be load balanced according to the load balancing option specified by the pool for that virtual server.

The following configuration shows how a request is directed with the Match Across Services option enabled:

```
HTTP Virtual Server: 192.168.0.10:http
Type of Persistence Used: Source Address Affinity and
Match Across Services enabled
HTTP Pool Name: http_pool
HTTP Pool Members: 172.16.1.1:http
  172.16.1.2:http
  172.16.1.3:http

HTTPS Virtual Server: 192.168.0.10:https
Type of Persistence Used: Source Address Affinity and
Match Across Services enabled
HTTPS Pool Name: https_pool
HTTPS Pool Members: 172.16.1.1:https
  172.16.1.2:https
  172.16.1.3:https

HTTP Virtual Server: 192.168.0.20:http
Type of Persistence Used: Source Address Affinity and
Match Across Services enabled
HTTP Pool Name: http2_pool
HTTP Pool Members: 172.16.1.1:8443
  172.16.1.2:8443
  172.16.1.3:8443
```

Match Across Virtual Servers

Match Across Virtual Servers is similar to Match Across Services, but it does not require the virtual servers to share the same IP address. This configuration allows clients to access different virtual servers, regardless of their IP address, and still access the same pool member.

The example configuration below shows that clients are load balanced to pool member `172.16.1.2:http`, and an entry is created in the persistence table when they first connect to virtual server `192.168.0.10:http`.

If the same clients connect to virtual server `192.168.0.10:https`, the BIG-IP system uses the persistence session information that was established with the initial connection to virtual server `192.168.0.10:http`, and directs the request to pool member `172.16.1.2:https`.

If the same clients connect to virtual server `192.168.0.20:http`, the BIG-IP uses the persistence session information that was established with the initial connection to virtual server `192.168.0.10:http` and directs the request to pool member `172.16.1.2:8443`.

> **Note:** This behavior occurs because the pool members used by virtual server `192.168.0.20:http` have the same node IP as those specified in the http_pool used by virtual server `192.168.0.10:http`.

If the client connects to a different virtual server that does not use persistence, that connection will be load balanced according to the load balancing option specified by the pool for that virtual server.

The following configuration shows how a request is directed when the Match Across Virtual Servers option is enabled:

```
HTTP Virtual Server: 192.168.0.10:http
Type of Persistence Used: Source Address Affinity and
Match Across Virtuals enabled
HTTP Pool Name: http_pool
HTTP Pool Members: 172.16.1.1:http
  172.16.1.2:http
  172.16.1.3:http

HTTPS Virtual Server: 192.168.0.10:https
Type of Persistence Used: Source Address Affinity and
Match Across Virtuals enabled
HTTPS Pool Name: https_pool
HTTPS Pool Members: 172.16.1.1:https
  172.16.1.2:https
  172.16.1.3:https

HTTP Virtual Server: 192.168.0.20:http
Type of Persistence Used: Source Address Affinity and
Match Across Virtuals enabled
HTTP Pool Name: http2_pool
HTTP Pool Members: 172.16.1.1:8443
  172.16.1.2:8443
  172.16.1.3:8443
```

Match Across Pools

The Match Across Pools option allows the BIG-IP system to use any pool that contains a persistence record for that specific client. You must proceed cautiously when using this option, as it can direct a client's request to a pool that is not specified by the virtual server.

3.03 - Describe how connection limits are affected by node, pool and virtual server settings

Connection limits configured on pool members or nodes for a CMP system are enforced per TMM instance

Pool member connection limits can be affected by settings in multiple areas of the configuration. Connection limits on a pool member can be affected by having connection limits set on the associated node. If there is connection limits set on the node and that setting is lower than the connection limit of the associated pool member, then the pool member can never reach its full limit. Likewise if the node is defined in multiple pools all connections to the node from the other pools count toward the node level connection limit which can restrict the number of connections to the pool member with connection limits set.

Virtual servers can have connection limits and persistence configured where each could affect connection limit settings at the pool level. The corresponding persistence profile can be set to allow "Override Connection Limits" described in a previous section. Connection limits on the virtual server can affect connection levels to the pool member if the virtual server setting is lower than the setting at the pool member level.

Vice versa, the connection limits at the pool level can affect the connection limits set in the virtual server configuration. If there are connection limits set on the pool members of a virtual servers associated pool, and the total of the pool members connection limits are less than the virtual server's connection limits then the virtual server can never reach its full limit.

Connection limits configured on pool members or nodes for a CMP system are enforced per TMM instance

The BIG-IP system divides the configured connection limit by the number of Traffic Management Microkernels (TMMs) that are running on the system, and forces the calculated limit to round down to the nearest whole number, per TMM. For example, when a pool member or node is configured with a connection limit of 10 on a CMP system with four running TMMs, the BIG-IP system divides the configured limit of 10 by four TMMs, and enforces the calculated connection limit of two on each running TMM. This is expected behavior.

> **Note:** If you use a pool member or node that is configured with very low connection limits, the system may appear to enforce lower than expected connection limits. In addition, F5 does not recommend that you configure a connection limit to a number lower than the number of TMM instances supported by the platform; doing so may result in unpredictable connection counts.

3.03 - Describe how priority groups are affected by connection limits

Pools

When a connection limit is set on a pool member, the pool member can receive that specific count of concurrent connections as a maximum. If the pool member has reached the connection limit and is temporarily unavailable, it will not count as an unavailable pool member that could trigger the priority group to activate the next group of pool members. Thus the pool could run out of available connections and still never activate an additional priority group in the pool.

Conclusion

This document is intended as a study guide for the F5 301a – LTM Specialist: Architect, Set-Up & Deploy exam. This study guide is not an all-inclusive document that will guarantee a passing grade on the exam. It is intended to be a living doc and any feedback or material that you feel should be included, to help exam takers better prepare, can be sent to channeleng@f5.com.

Thank you for using this study guide to prepare the 301a – LTM Specialist exam and good luck with your certification goals.

Thanks,
Eric Mitchell
Channel FSE, East US and Federal

F5 Networks, Inc. 401 Elliott Avenue West, Seattle, WA 98119 888-882-4447 f5.com

Americas info@f5.com
Asia-Pacific apacinfo@f5.com
Europe/Middle-East/Africa emeainfo@f5.com
Japan f5j-info@f5.com