DevCentral Basics: Application Delivery Services

PRESENTED BY:
Jason Rahm, Sr. Strategic Architect
NETWORKING
Networking Concepts

- Physical/Virtual NICs
- VLANs and VLAN Groups
  - Untagged and Tagged Interfaces
- Self IPs (local / floating)
- Routes are just pathways, not permissions
- Auto Last Hop
- Route Domains
Overview of Networking

- **TMOS is a full proxy architecture**
  - Traffic must pass through BIG-IP to gain the benefits of TMOS

- **Routed mode (recommended)**
  - Real servers are on an internal network behind BIG-IP
  - BIG-IP is default gateway for the servers
  - The virtual servers are on an external network
  - Accessible by the clients

- **Source Network Address Translation (SNAT) Mode**
  - Also known as: One-Armed mode
  - Allows a BIG-IP to be inserted into existing networks without changing the existing IP address structure
How Routed Mode Works

The default gateway for the RED and BLUE servers is 1.1.1.254 on BIG-IP LTM.

HTTP request
DST: 2.2.2.2:80
SRC: 3.3.3.3
http_vs 2.2.2.2:80

HTTP response
DST: 3.3.3.3
SRC: 2.2.2.2:80

Unique TCP sessions

http_pool
RED 1.1.1.1:8080
BLUE 1.1.1.2:8080

HTTP request
DST: 1.1.1.1:8080
SRC: 3.3.3.3

http_vs 2.2.2.2:80

VLAN Internal
IP 1.1.1.254

VLAN External
IP 2.2.2.254

BIG-IP LTM chooses RED

Client 3.3.3.3

HTTP request
DST: 2.2.2.2:80
SRC: 3.3.3.3

HTTP response
DST: 3.3.3.3
SRC: 2.2.2.2:80

The default gateway for the RED and BLUE servers is 1.1.1.254 on BIG-IP LTM.
Why SNATs May Be Required

- The default gateway for the RED and BLUE servers is 1.1.1.254 on BIG-IP LTM.

The diagram illustrates the interaction between the client, BIG-IP LTM, and the servers. The client makes an HTTP request to 1.1.1.5 from 3.3.3.3, which is not the default gateway. BIG-IP LTM chooses the BLUE server from the http_pool RED 1.1.1.1:8080 BLUE 1.1.1.2:8080.

HTTP request:
- DST: 1.1.1.5
- SRC: 3.3.3.3

HTTP response:
- DST: 3.3.3.3
- SRC: 1.1.1.2

TCP Session is broken.
How SNAT Mode Works

The default gateway for the RED and BLUE servers is 1.1.1.254 on BIG-IP LTM.

Big-IP LTM chooses RED

Outside TCP session

Inside TCP session
LOAD BALANCING
Introduction to Load Balancing

- A load balancing method is an algorithm or formula used to determine which pool member to send traffic to.
- Load balancing is connection based.

- Static load balancing methods distribute connections in a fixed manner.
  - Round Robin (RR)
  - Ratio (Weighted Round Robin)
  - Distributes in a RR fashion for members/nodes whose ratio has not been met.

- Dynamic load balancing methods take into account one or more factors, such as the current connection count.

- It is important to experiment with different load balancing methods and select the one that offers the best performance in your particular environment.
Dynamic Load Balancing Methods

- **Least Connections**
  - Fewest L4 connections when load balancing decision is being made
  - Recommended when servers have similar capabilities
  - Very commonly used

- **Fastest**
  - Balances based upon the number of outstanding L7 requests and then L4 connections
  - Requires a L7 profile on the virtual server, else it’s just Least Connections
  - Recommended when servers have similar capabilities

- **Observed**
  - Calculates a ratio each second based on the number of L4 connections
  - Not recommended for large pools

*SOL6406 - Change in Behavior: Fastest, Observed, and Predictive load balancing modes*
Dynamic Load Balancing Methods

• **Predictive**
  - Calculates ratio base on the change between the previous connection counts and the current connection counts
  - Not recommended for large pools

• **Weighted Least Connections**
  - Based on how close the number of connections are to meeting the connection limit for a pool member or node
  - Requires connection limits be set on pool member or node
  - Recommended when servers have different capabilities

• **Dynamic Ratio**
  - Dynamically weights servers based on the results of SNMP/WMI queries
  - Requires SNMP_DCA, SNMP_Base, or WMI pool monitoring
  - Recommended when custom calculations are needed
Nodes, Pool Members, and Pools

BIG-IP Platform

Pool = Group of Pool Members

192.168.1.100:80
192.168.1.101:80
192.168.1.102:80

Node = IP address
Pool Member = Node + Port
Load Balancing a Service (Member)

With each new client request, BIG-IP LTM verifies which pool member has the fewest active connections.

In this example, the HTTP pool is configured with the Least Connections (member) method.

BIG-IP LTM directs the request to the pool member with the least number of connections.

Current connection counts for each pool member are displayed in red.

<table>
<thead>
<tr>
<th>Pool</th>
<th>Member</th>
<th>IP Address</th>
<th>Port</th>
<th>Connections</th>
</tr>
</thead>
<tbody>
<tr>
<td>http_pool</td>
<td>172.20.10.1</td>
<td>172.20.10.1</td>
<td>80</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td>172.20.10.2</td>
<td>172.20.10.2</td>
<td>80</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>172.20.10.3</td>
<td>172.20.10.3</td>
<td>8080</td>
<td>36</td>
</tr>
<tr>
<td>secure_pool</td>
<td>172.20.10.2</td>
<td>172.20.10.2</td>
<td>443</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>172.20.10.3</td>
<td>172.20.10.3</td>
<td>443</td>
<td>22</td>
</tr>
</tbody>
</table>
Load Balancing an IP Address (Node)

In this example, the HTTP pool is configured with the **Least Connections (node)** method.

BIG-IP LTM directs the request to the node with the least number of connections.

With each new end-user request, BIG-IP LTM verifies which node has the fewest active connections.

With each new client request, BIG-IP LTM verifies which **node** has the fewest active connections.

This takes into account all services running on the node.

Current connection counts for each pool member are displayed in red.
MONITORS
Introduction to Monitors

- A monitor is a test
  - Of a specific application. For an expected response. Within a given time.

- All BIG-IP have two things in common
  - Interval - time between each check
  - Timeout - time required for a successful check to be received before BIG-IP marks the node as unavailable

- BIG-IP LTM can use composite monitors, so it can apply multiple checks
  - It can use all or some of the monitors to determine member status

- Monitors can also use reverse logic

- Monitors are served from the Self IP addresses
Types of Monitor Checks

- **Node checks**
  - Determines availability of all services for a particular node
  - For example, ICMP check determines if the node is pingable
  - When a check fails, the node is pulled from all pools it has membership in

- **Service checks**
  - Checks connectivity to services/ports
  - For example, HTTP check determines if port 80 can be opened

- **Content checks**
  - Queries the service and checks the contents of the query
  - For example, HTTP GET / determines if the page returns with correct content
  - Content checks can involve username and passwords

- **Path checks**
  - Are transparent monitors that check devices outside the pool

- **Interactive checks**
  - Custom scripts that interact with application
Monitors check the status of a pool member or node on an ongoing basis, at a set interval. If a pool member or node being monitored does not respond within the set interval, BIG-IP LTM marks it offline. BIG-IP LTM continues to direct traffic to the remaining pool members while continuing to monitor the offline pool member or node. When the pool member or node responds, BIG-IP LTM marks it as available and starts directing traffic to the pool member.
Monitor Status Reporting

- **Status is based on monitor response and object hierarchy**
  - The virtual server status is affected by the status of the pool
  - The pool status is affected by the status of pool members
  - A pool member is affected by the status of the node (server)

- **What happens when a monitor fails?**
  - If the timeout has been reached
  - The member becomes suspect
  - No NEW connections to the member
  - Existing connections are maintained
  - If there is a successful check before the timeout value is reached
  - New connections are sent to the member
  - If the monitor failures exceed the timeout value
  - Member is removed from the pool
  - Connections are reaped
## Monitor Status

<table>
<thead>
<tr>
<th>Status</th>
<th>Status Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Available</strong></td>
<td>General: Child • Monitor successful</td>
</tr>
<tr>
<td></td>
<td>General: Parent • At least one child is Green</td>
</tr>
<tr>
<td></td>
<td>Child –Node • Most recent monitor successful</td>
</tr>
<tr>
<td></td>
<td>Pool Member • Most recent monitor successful</td>
</tr>
<tr>
<td></td>
<td><strong>Pool</strong> • <strong>At least one pool member is available</strong></td>
</tr>
<tr>
<td></td>
<td>Virtual Server • At least one pool is available</td>
</tr>
<tr>
<td><strong>Unknown</strong></td>
<td>General: Child • No associated monitor (or timeout of first check not reached)</td>
</tr>
<tr>
<td></td>
<td>General: Parent • All child objects are unknown (blue)</td>
</tr>
<tr>
<td></td>
<td>Node • No associated monitor (or timeout of first check not reached and not successful)</td>
</tr>
<tr>
<td></td>
<td>Pool Member • No associated monitor (or timeout of first check not reached and not successful)</td>
</tr>
<tr>
<td></td>
<td>Pool • All pool members are unknown (blue)</td>
</tr>
<tr>
<td></td>
<td>Virtual Server • All pools are unknown (blue)</td>
</tr>
<tr>
<td><strong>Offline</strong></td>
<td>General: Child • Monitor failed</td>
</tr>
<tr>
<td></td>
<td>General: Parent • At least one child red AND no green or yellow children available</td>
</tr>
<tr>
<td></td>
<td>Node • Most recent monitor failed (no successful checks within timeout period)</td>
</tr>
<tr>
<td></td>
<td>Pool Member • Most recent monitor failed (no successful checks within timeout period)</td>
</tr>
<tr>
<td></td>
<td>Pool • One or more members are offline and no members are available</td>
</tr>
<tr>
<td></td>
<td>Virtual Server • One or more pools offline and no members available</td>
</tr>
</tbody>
</table>
Inband (Passive) Monitors

- Inband monitors use client request to see if the pool member:
  - Connects (SYN-SYN/ACK-ACK)
  - If there is a L7 profile on the virtual server (e.g. HTTP), checks for a response to the L7 request

- Inband monitors require pools and virtual servers that:
  - Are Standard or Performance (Layer 4)
  - Use the TCP or SCTP protocol profile

- Inband monitors are effective in marking a pool member down
  - Not as efficient in marking a member up
  - Member remains uncheck for a period of time

- Active monitors are more effective in marking members up

- Active and Inband monitors can be combined
  - For effective monitoring with a lower overhead monitoring
  - Monitors at a different interval if the pool member is available
Profiles

- Are a configuration tool that aids you in managing application traffic

- A profile defines how a virtual server processes packets it receives
  - Based on which profiles are assigned to the virtual server
  - Based upon the profile’s configured parameters
  - The same profile can be associated with one or more virtual servers

- Different profile types, different traffic processing capabilities
  - Protocol profiles, such as, TCP and UDP
  - SSL profiles, for client-side and server-side certificates and keys
  - Service (L7) profiles, such as, HTTP, FTP, DNS
  - And many more……

- Profiles have a parent/child relationship
  - Changes to a parent profile are passed down to the child profile(s)
Modify the `http_fallback` profile, which changes the site for all internal virtual servers that use the `http_fallback` profile.

That same change propagates to the `child` profile `http_xforward`, so all external virtual servers now fallback to the new fallback host.

Create a new profile called `http_fallback` by using the default `http` profile as the `parent` profile.

Create a new profile called `http_xforward` by using `http_fallback` as the `parent` profile and then configure it to include the “X-Forwarded For” HTTP header.

- For all virtual servers, if a pool fails, you want to redirect to the same fallback host.
- For external facing virtual servers insert the “X-Forwarded For” HTTP header to maintain the original client IP for logging.

For all virtual servers, if a pool fails, you want to redirect to the same fallback host.

For external facing virtual servers insert the “X-Forwarded For” HTTP header to maintain the original client IP for logging.

Associate `http_fallback` with the internal virtual servers, and…

Associate `http_fallback` with the internal virtual servers, and…

…`http_xforward` with the external virtual servers.

…`http_xforward` with the external virtual servers.

Profiles

http (parent)

http_fallback (parent/child)

http_xforward (child)
TCP Profile

- Default (often) not the best choice
- Basic TCP protocol validation performed
- Built-in and custom optimizations possible
- Timers, memory, connection setup, congestion control, etc.
- Can set client- and server-side independently

300ms .5% loss
60ms .01% loss
<1ms .0001% loss
Client- and Server-SSL Profiles

- **Clientssl** – Presents the certificate to the client
- **Serverssl** – Receives the certificate from the server
HTTP Profile

- Basic protocol validation when applied
- Required profile for traffic through other modules like AAM, ASM
- Highly customizable web application experience
**Persistence Profiles**

- **Active**
- Tracked in memory tables
- **Passive**
- External or calculated

<table>
<thead>
<tr>
<th>Persistence Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source IP</td>
</tr>
<tr>
<td>192.168.101.23</td>
</tr>
</tbody>
</table>

**SRC: 192.168.101.23**

**Pool Member:**
192.168.100.103:80
Working with Profiles

- Best practice is to not modify default profiles
- Always create a custom profile, even if you change nothing

- Some profiles conflict with each other
  - BIG-IP will notify you of conflicts, most are obvious, for example, UDP/HTTP

- Some profiles require other profile, for example:
  - Using the Stream profile to replace strings in HTTP, would required HTTP profile

- Profiles can have an impact on system resources, for example:
  - HTTP Compression profiles may impact CPU, if not hardware accelerated
  - Web Acceleration profiles will impact memory usage, since content is stored in RAM
VIRTUAL SERVERS
AND SNATS
BIG-IP LTM Components: Virtual Servers

A virtual server is an IP address and service (port) combination that listens for client requests.

Each virtual server will uniquely process client requests that match its IP address and port.

Each virtual server then directs the traffic, usually to an application pool.

The virtual server translates the destination IP address and port to the selected pool member.

NOTE: BIG-IP LTM is a default deny device; the virtual server is the most common way to allow client requests to pass through.

NOTE: Multiple virtual servers can reference the same pools, pool members, and/or nodes.
Virtual Servers

- One of the most important configuration components
- Determines what traffic is to pass
- Where the traffic goes
- How it is viewed/manipulated/validated (mostly via profiles)
- So in the last slides we saw virtual server basics (in and out) ….
But there is so much more....

- And this is just the basic menu...
- Layer 4-7 profiles
- Restrictions on traffic
- Source Address Translation
- Destination IP and Port Translation in Advanced
How Does a BIG-IP Handle Inbound Traffic

• But virtual server isn’t the only listener

• Listeners are
  • Self IPs (Port Lockdown: None)
  • SNATs (Source initiated)
  • NATs (Two-way)
  • Virtual Servers
Packet Processing Priority

1. Existing connection in connection table
2. Packet filter rule
3. Virtual server
4. SNAT
5. NAT
6. Self-IP
7. Drop
Standard Virtual Server Packet Flow

Proxy

Load balancing algorithms

HTTP

SSL

TCP Express

VS listener

IPv4
IPv6

RAM Cache

iRules

iRules

iRules

iRules

iRules

HTTP

SSL

TCP Express

IPv4
IPv6

iRules

iRules

iRules

iRules

iRules
Forwarding Virtual Servers

Example: Web administrators required SSH, TFTP, Webmin, HTTP, and HTTPS access to individual backend Apache servers.

Name: forwarding_vs
Source Address: 3.3.3.0/24
Destination Address/Mask: 2.2.2.0/24
Service Port: *
Protocol: *

Unique TCP sessions
## Virtual Server Precedence

- **What happens if a packet matches virtual A in 1 AND virtual B in 3, and I disable virtual A?**
- The packet is rejected
- Set db variable `tm.continuematching` to true to change that default behavior

<table>
<thead>
<tr>
<th>Order</th>
<th>Destination</th>
<th>Source</th>
<th>Service Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&lt;host address&gt;</td>
<td>&lt;host address&gt;</td>
<td>&lt;port&gt;</td>
</tr>
<tr>
<td>2</td>
<td>&lt;host address&gt;</td>
<td>&lt;host address&gt;</td>
<td>*</td>
</tr>
<tr>
<td>3</td>
<td>&lt;host address&gt;</td>
<td>&lt;network address&gt;</td>
<td>&lt;port&gt;</td>
</tr>
<tr>
<td>4</td>
<td>&lt;host address&gt;</td>
<td>&lt;network address&gt;</td>
<td>*</td>
</tr>
<tr>
<td>5</td>
<td>&lt;host address&gt;</td>
<td>*</td>
<td>&lt;port&gt;</td>
</tr>
<tr>
<td>6</td>
<td>&lt;host address&gt;</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>7</td>
<td>&lt;network address&gt;</td>
<td>&lt;host address&gt;</td>
<td>&lt;port&gt;</td>
</tr>
<tr>
<td>8</td>
<td>&lt;network address&gt;</td>
<td>&lt;host address&gt;</td>
<td>*</td>
</tr>
<tr>
<td>9</td>
<td>&lt;network address&gt;</td>
<td>&lt;network address&gt;</td>
<td>&lt;port&gt;</td>
</tr>
<tr>
<td>10</td>
<td>&lt;network address&gt;</td>
<td>&lt;network address&gt;</td>
<td>*</td>
</tr>
<tr>
<td>11</td>
<td>&lt;network address&gt;</td>
<td>*</td>
<td>&lt;port&gt;</td>
</tr>
<tr>
<td>12</td>
<td>&lt;network address&gt;</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>13</td>
<td>*</td>
<td></td>
<td>Repeat 7-12</td>
</tr>
</tbody>
</table>
NATs and SNATs

- NATs are source and destination listeners
- NAT relationship is 1:1
- NAT source can only match a host
- NAT takes precedence over SNAT if both source match
- SNATs are source listeners only
- SNAT relationship is 1:1 and/or many:1
- SNATs can be specified, auto-mapped, or pooled
- Cannot initiate a connection to a SNAT
HIGH AVAILABILITY
Device Service Clusters (DSCs)

- Manual syncing
- Sync common configuration items across Device Groups
- 2 Active / 1 Standby
- Ability to scale beyond the pair
- Simplify deployment of Active-Active
- Higher device utilization
- 5 Active
Sync-Only Device Groups

- Allows flexible membership
- Different hardware platforms
- Different license/modules

- Can be configured to auto-sync objects
  - Certificates
  - CRL
  - Data groups
  - External monitors
  - iApps
  - iRules
  - Policies
  - Profiles

- Max of 32 Sync-Only groups are supported

- Device trust uses built-in sync-only group “device_trust_group”
  - Auto-sync enabled
  - Adding devices to trust-domain auto-adds to device_trust_group
Sync-Failover Device Groups

- **V11.0-11.3 requires like devices**
  - Same hardware/VE
  - Same license/provisioning
  - Not strictly enforced

- **V11.4**
  - Allows for different devices
  - But not in mirrored environments
  - Still requires same licensing/provisioning

- **Logical grouping of HA devices**
  - F5 provides N+M redundancy
  - N Active units + M standby units
  - Mirroring requires only two devices be a part of the group

- **A device can only be part of one sync-failover group**
Traffic Groups

- A collection of listeners to failover

- Create traffic groups and assign applications to the group

- Assign the traffic groups to cluster members

- If a cluster member has no active traffic groups, it is in standby

- If a device fails, the traffic group migrates to another BIG-IP in the cluster
Thank You