

# DELL EMC ECS WITH F5

## Deployment Reference Guide

August 2017

## Revisions

Date	Description
August 2017	Initial release
October 2017	Corrected initial release date year to 2017 from 2016

The information in this publication is provided “as is.” Dell Inc. makes no representations or warranties of any kind with respect to the information in this publication, and specifically disclaims implied warranties of merchantability or fitness for a particular purpose.

Use, copying, and distribution of any software described in this publication requires an applicable software license.

Copyright © August 2017 Dell Inc. or its subsidiaries. All Rights Reserved. Dell, EMC, and other trademarks are trademarks of Dell Inc. or its subsidiaries. Other trademarks may be the property of their respective owners. Published in the USA [10/11/2017] [Whitepaper] [H16294]

Dell believes the information in this document is accurate as of its publication date. The information is subject to change without notice.

# Table of contents

Revisions .....	2
1 Introduction .....	4
1.1 Audience .....	4
1.2 Scope .....	4
2 ECS Overview .....	5
2.1 ECS Constructs .....	6
3 F5 Overview .....	8
3.1 F5 Networking Constructs .....	9
3.2 F5 Traffic Management Constructs .....	11
3.3 F5 Device Redundancy .....	13
4 ECS Configuration .....	15
5 F5 Configuration .....	19
5.1 F5 BIG-IP LTM .....	19
5.1.1 Example: S3 to Single VDC with Active/Standby LTM Pair .....	20
5.1.2 Example: LTM-terminated SSL Communication .....	30
5.1.3 Example: NFS via LTM .....	35
5.1.4 Example: Geo-affinity via iRule on LTM .....	43
5.2 F5 BIG-IP DNS .....	46
5.2.1 Example: Roaming Client Geo Environment .....	52
5.2.2 Example: Application Traffic Management for XOR Efficiency Gains .....	53
6 Best Practices .....	54
7 Conclusion .....	55
8 References .....	56
A Creating a Custom S3 Monitor .....	57
B BIG-IP DNS and LTM CLI Configuration Snippets .....	60
B.1 BIG-IP DNS .....	60
B.1.1 bigip.conf .....	60
B.1.2 bigip_base.conf .....	60
B.1.3 bigip_gtm.conf .....	61
B.2 BIG-IP LTM .....	64
B.2.1 bigip.conf .....	64
B.2.2 bigip_base.conf .....	67

# 1 Introduction

The explosive growth of unstructured data and cloud-native applications has created demand for scalable cloud storage infrastructure in the modern datacenter. Elastic Cloud Storage™ (ECS™) is the third generation object store by Dell EMC™ designed from the ground up to take advantage of modern cloud storage APIs and distributed data protection, providing active/active availability spanning multiple datacenters.

Managing application traffic both locally and globally can provide high availability (HA) and efficient use of ECS storage resources. HA is obtained by directing application traffic to known-to-be-available local or global storage resources. Optimal efficiency can be gained by balancing application load across local storage resources.

Organizations often choose F5® BIG-IP® DNS (formerly Global Traffic Manager™) and Local Traffic Manager™ (LTM®) products to manage client traffic between and within data centers. BIG-IP authoritatively resolves domain names such as s3.dell.com or nfs.emc.com. BIG-IP DNS systems return IP addresses with the intent to direct stateless client sessions to an ECS system at a particular data center based on monitoring the availability and performance of individual ECS data center locations, nodes, and client locations. LTMs can apply load balancing services based on availability, performance, and persistence, to proxy client traffic to an ECS node.

## 1.1 Audience

This document is targeted for customers interested in a reference deployment of ECS with F5.

## 1.2 Scope

This whitepaper is meant to be a reference guide for deploying F5 with ECS. An external load balancer (traffic manager) is highly recommended with ECS for applications that do not proactively monitor ECS node availability or natively manage traffic load to ECS nodes. Directing application traffic to ECS nodes using local DNS queries, as opposed to a traffic manager, can lead to failed connection attempts to unavailable nodes and unevenly distributed application load on ECS.

The ECS HDFS client, CAS SDK and ECS S3 API extensions are outside of the scope of this paper. The ECS HDFS client, which is required for Hadoop connectivity to ECS, handles load balancing natively. Similarly, the Centera Software Development Kit for CAS access to ECS has a built-in load balancer. The ECS S3 API also has extensions leveraged by certain ECS S3 client SDKs which allow for balancing load to ECS at the application level.

Dell EMC takes no responsibility for customer load balancing configurations. All customer networks are unique, with their own requirements. It's extremely important for customers to configure their load balancers according to their own circumstance. We only provide this paper as a guide. F5 or a qualified network administrator should be consulted before making any changes to your current load balancer configuration.

Related to the BIG-IP DNS and LTM, and outside this paper's scope, is the BIG-IP Application Acceleration Manager™ that provides encrypted, accelerated WAN optimization service. F5's BIG-IP Advanced Firewall Manager™ which provides network security and DDoS mitigation services, is also outside of the scope of this paper.

## 2 ECS Overview

ECS provides a complete software-defined strongly-consistent, indexed, cloud storage platform that supports the storage, manipulation, and analysis of unstructured data on a massive scale. Client access protocols include S3, with additional Dell EMC extensions to the S3 protocol, Dell EMC Atmos, Swift, Dell EMC CAS (Centera), NFS, and HDFS. Object access for S3, Atmos, and Swift is achieved via REST APIs. Objects are written, retrieved, updated and deleted via HTTP or HTTPS calls using REST verbs such as GET, POST, PUT, DELETE, and HEAD. For file access, ECS provides NFS version 3 natively and a Hadoop Compatible File System (HCFS).

ECS was built as a completely distributed system following the principle of cloud applications. In this model, all hardware nodes provide the core storage services. Without dedicated index or metadata master nodes the system has limitless capacity and scalability.

Service communication ports are integral in the F5 LTM configuration. See Table 1 below for a complete list of protocols used with ECS and their associated ports. In addition to managing traffic flow, port access is a critical piece to consider when firewalls are in the communication path. For more information on ECS ports refer to the ECS Security Configuration Guide at [https://support.emc.com/docu79370\\_ECS-3.0-Security-Configuration-Guide.pdf?language=en\\_US](https://support.emc.com/docu79370_ECS-3.0-Security-Configuration-Guide.pdf?language=en_US).

For a more thorough ECS overview, please review ECS Overview and Architecture whitepaper at <http://www.emc.com/collateral/white-papers/h14071-ecs-architectural-guide-wp.pdf>.

Table 1 - ECS protocols and associated ports

Protocol	Transport Protocol or Daemon Service	Port
S3	HTTP	9020
	HTTPS	9021
Atmos	HTTP	9022
	HTTPS	9023
Swift	HTTP	9024
	HTTPS	9025
NFS	portmap	111
	mountd, nfsd	2049
	lockd	10000

## 2.1 ECS Constructs

Understanding the main ECS constructs is necessary in managing application workflow and load balancing. This section details each of the upper-level ECS constructs.

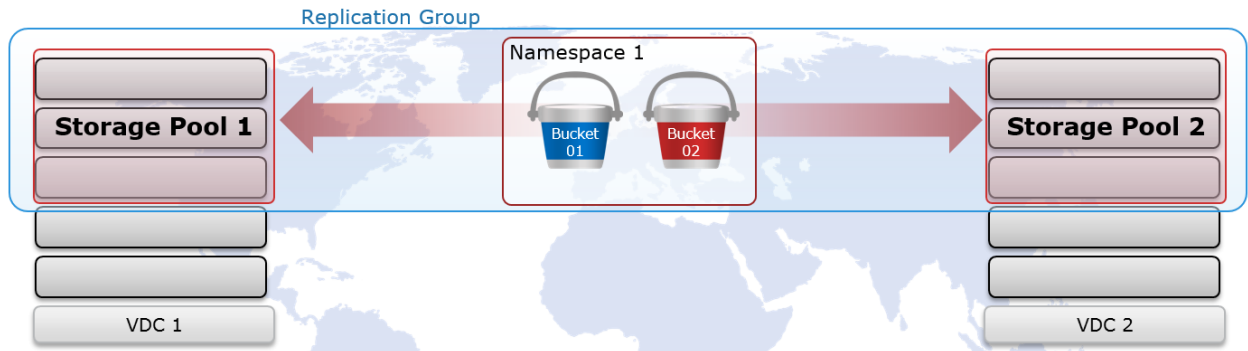


Figure 1 - ECS upper-level constructs

- **Storage pool** - The first step in provisioning a site is creating a storage pool. Storage pools form the basic building blocks of an ECS cluster. They are logical containers for some or all nodes at a site.

ECS storage pools identify which nodes will be used when storing object fragments for data protection at a site. Data protection at the storage pool level is rack, node, and drive aware. System metadata, user data and user metadata all coexist on the same disk infrastructure.

Storage pools provide a means to separate data on a cluster, if required. By using storage pools, organizations can organize storage resources based on business requirements. For example, if separation of data is required, storage can be partitioned into multiple different storage pools. Erasure coding (EC) is configured at the storage pool level. The two EC options on ECS are 12+4 or 10+2 (aka cold storage). EC configuration cannot be changed after storage pool creation.

Only one storage pool is required in a VDC. Generally, at most two storage pools should be created, one for each EC configuration, and only when necessary. Additional storage pools should only be implemented when there is a use case to do so, for example, to accommodate physical data separation requirements. This is because each storage pool has unique indexing requirements. As such, each storage pool adds overhead to the core ECS index structure.

A storage pool should have a minimum of five nodes and must have at least three or more nodes with more than 10% free space in order to allow writes.

- **Virtual Data Center (VDC)** - VDCs are the top-level ECS resources and are also generally referred to as a site or zone. They are logical constructs that represent the collection of ECS infrastructure you want to manage as a cohesive unit. A VDC is made up of one or more storage pools.

Between two and eight VDCs can be federated. Federation of VDCs centralizes and thereby simplifies many management tasks associated with administering ECS storage. In addition federation of sites allows for expanded data protection domains that include separate locations.

- **Replication Group** - Replication groups are logical constructs that define where data is protected and accessed. Replication groups can be local or global. Local replication groups protect objects within the same VDC against disk or node failures. Global replication groups span two or more federated VDCs and protect objects against disk, node, and site failures.

The strategy for defining replication groups depends on multiple factors including requirements for data resiliency, the cost of storage, and physical versus logical separation of data. As with storage pools, the minimum number of replication groups required should be implemented. At the core ECS indexing level, each storage pool and replication group pairing is tracked and adds significant overhead. It is best practice to create the absolute minimum number of replication groups required. Generally this is one replication group for each local VDC, if necessary, and one replication group that contains all sites. Deployments with more than two sites may consider additional replication groups, for example, in scenarios where only a subset of VDCs should participate in data replication, but, this decision should not be made lightly.

- **Namespace** - Namespaces enable ECS to handle multi-tenant operations. Each tenant is defined by a namespace and a set of users who can store and access objects within that namespace. Namespaces can represent a department within an enterprise, can be created for each unique enterprise or business unit, or can be created for each user. There is no limit to the number of namespaces that can be created from a performance perspective. Time to manage an ECS deployment, on the other hand, or, management overhead, may be a concern in creating and managing many namespaces.
- **Bucket** - Buckets are containers for object data. Each bucket is assigned to one replication group. Namespace users with the appropriate privileges can create buckets and objects within buckets for each object protocol using its API. Buckets can be configured to support NFS and HDFS. Within a namespace, it is possible to use buckets as a way of creating subtenants. It is not recommended to have more than 1000 buckets per namespace. Generally a bucket is created per application, workflow, or user.

### 3 F5 Overview

The F5 BIG-IP platform is a blend of software and hardware that forms the foundation of the current iteration of F5's Application Delivery Controller (ADC) technology. On top of the BIG-IP platform are a suite of products that provide a wide range of application services. Two of the BIG-IP products often used in building an organization's foundation for local and global traffic management include:

- **F5 BIG-IP Local Traffic Manager (LTM)** - Local application traffic load balancing based on a full-proxy architecture.
- **F5 BIG-IP DNS (formerly Global Traffic Manager (GTM))** - DNS services for application requests based on user, network, and cloud performance conditions.

F5 offers the products in both physical and virtual form. This paper makes no recommendation between choosing physical or virtual F5 systems. We do recommend that F5 documentation and product specifications be consulted to properly size the F5 systems. Sizing the BIG-IP systems should be based on the cumulative current and projected workload traffic bandwidth (MB/s), quantity of operations (Ops/sec) or transaction per second (TPS), and concurrent client sessions. Properly sized BIG-IP systems should not add any significant transactional overhead or limitation to workflows using ECS. The next few paragraphs briefly describe considerations when sizing ECS and associated traffic managers.

There are two components to each transaction with ECS storage, one, metadata operations, and two, data movement. Metadata operations include both reading and writing or updating the metadata associated with an object and we refer to these operations as transactional overhead. Every transaction on the system will have some form of metadata overhead and the size of an object along with the transaction type determine the associated level of resource utilization. Data movement is required for each transaction also and refers to the receipt or delivery of client data. We refer to this component in terms of throughput or bandwidth, generally in megabytes per second (MB/s).

To put this in to an equation, response time, or total time to complete a transaction, is the result of the time to perform the transaction's required metadata operations, or the transaction's transactional overhead, plus, the time it takes to move the data between client and through ECS, the transaction's data throughput.

Total response time = transactional overhead time + data movement time

For small objects transactional overhead is similar and is the largest, or limiting, factor to performance. Since every transaction has a metadata component, and the metadata operations have a minimum amount of time to complete, at objects with similar sizes that component, the transactional overhead, will be similar. On the other end of the spectrum, for large objects, the transactional overhead is minimal compared to the amount of time required to physically move the data between client and the ECS system. For larger objects the major factor in response time is the throughput which is why at certain large object sizes the throughput potential is similar.

Object size and thread counts, along with transaction type, are the primary factors which dictate performance on ECS. Because of this in order to size for workloads, object sizes and their transaction types along with related application thread counts should be cataloged for all workloads that will utilize the ECS system.

BIG-IP DNS and LTM each provide very different functions and can be used together or independently. When used together BIG-IP DNS can make decisions based on data received directly from LTMs. BIG-IP



DNS is recommended if you want to either globally distribute application load, or to ensure seamless failover during site outage conditions where static failover processes are not feasible or desirable. BIG-IP LTM can manage client traffic based on results of monitoring both network and application layers and is largely mandatory where performance and client connectivity is required.

With ECS, monitoring application availability to the data services across all ECS nodes is necessary. This is done using application level queries to the actual data services that handle transactions as opposed to relying only on lower network or transport queries which only report IP and port availability.

A major difference between BIG-IP DNS and LTM is that traffic flows through an LTM whereas the BIG-IP DNS only informs a client which IP to route to and does not handle client application data traffic.

## 3.1 F5 Networking Constructs

A general understanding of the basic BIG-IP network constructs is critical to a successful architecture and implementation. Here is a list of the most common of the BIG-IP networking constructs:

- **Virtual Local Area Network (VLAN)** - A VLAN is required and associated with each non-management network interface in use on a BIG-IP system.
- **Self IP** - F5 terminology for an IP address on a BIG-IP system that is associated with a VLAN, to access hosts in the VLAN. A self IP represents an address space as determined by the associated netmask.
- **Static self IP** - A minimum of one IP address is required for each VLAN in use in a BIG-IP system. Static self IP addresses are unit-specific in that they are unique to the assigned device. For example, if two BIG-IP devices are paired together for HA, each device's configuration will have at least one self IP for each VLAN. The static self IP configuration is not sync'd or shared between devices.
- **Floating self IP** - Traffic groups, which are logical containers for virtual servers, can float between BIG-IP LTM devices. Floating self IP addresses are created similarly to static addresses, except that they are assigned to a floating traffic group. With this, during failover the self IP floats between devices. The floating self IP addresses are shared between devices. Each traffic group that is part of an HA system uses a floating self IP address and the device actively serving the virtual servers on the traffic group hosts the floating self IP.
- **MAC Masquerading Address** - An optional virtual layer two Media Access Control (MAC) address, otherwise known as a MAC masquerade, which floats with a traffic group's floating self IP.

MAC masquerade addresses serve to minimize ARP communications or dropped packets during a failover event. A MAC masquerade address ensures that any traffic destined for the relevant traffic group reaches an available device after failover has occurred, because the MAC masquerade address floats to the available device along with the traffic group. Without a MAC masquerade address, on failover the sending host must relearn the MAC address for the newly-active device, either by sending an ARP request for the IP address for the traffic or by relying on the gratuitous ARP from the newly-active device to refresh its stale ARP entry.

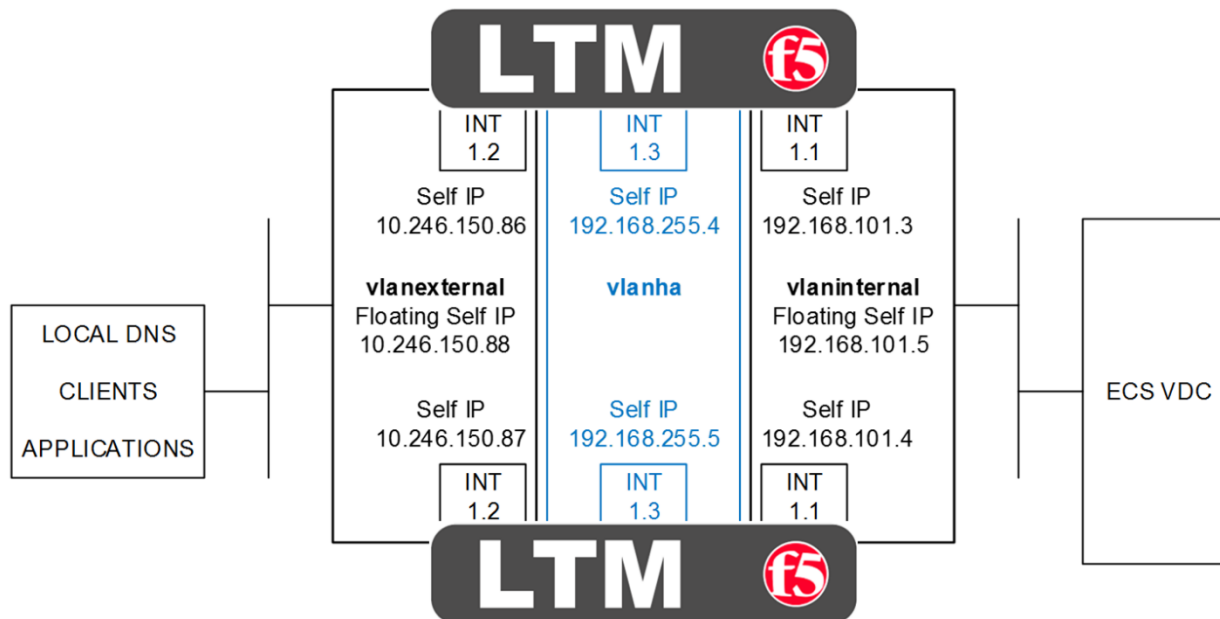


Figure 2 - HA pair of LTM with physical interfaces, VLAN and self IP addresses

Figure 2 above shows a HA pair of LTM. The physical interfaces as shown as 1.1, 1.2, and 1.3. In addition each device has a management interface which are not shown. Each interface has an associated VLAN and self IP address. Floating self IP addresses are created in redundant configurations as is a VLAN dedicated for HA. Not shown are any MAC masquerading addresses. MAC masquerading addresses are associated with traffic groups which are explained in the next section on F5 device redundancy.

BIG-IP LTM, like with other Application Delivery Controllers, can be deployed in a variety of deployment architectures. Two commonly used network deployment architectures are:

1. **One-arm** - A one-arm deployment is where the LTM has only a single VLAN and interface configured for application traffic. The LTM both receives and forwards traffic using a single network interface. The LTM sits on the same subnet as the destination servers, in our case, the ECS nodes in a VDC. In this scenario applications direct their traffic to virtual servers on the LTM which use virtual IP addresses that are on the same subnet as the destination nodes. Traffic received by the LTM is forwarded to the target ECS node.
2. **Two-arm** - The examples we use throughout this paper are two-arm implementations. A two-arm deployment is where the LTM sits on the same subnet as the destination servers, as above, but also listens for the application traffic on an entirely different subnet. The LTM uses two interfaces, each on unique subnets, to receive and forward traffic. In this scenario two VLANs exist on the LTM, one for external, client-side traffic, and the other one for internal, server-side traffic. Figure 2 above is an example of a two-arm architecture.

Traffic routing between the client, LTM, and ECS nodes is important. Two primary options for routing exist for the pool members, in our case, ECS nodes:

1. The default route on the ECS nodes point to a **self IP address on the LTM**. With this, as the nodes return traffic to the clients, because the default route points to the LTM, traffic will route back to the clients the same way it came, through the LTM. In our lab setup we set the default route on the ECS nodes to point to the self IP address on the LTM. For Site 1, our site with a pair of LTMs configured for HA, we used the floating self IP address as the default gateway. For Site 2, the ECS nodes point to the self IP address of the single LTM.
2. The default route on the ECS nodes point to an **IP address on a device other than the LTM**, such as a router. This is often referred to as Direct Server Return routing. In this scenario application traffic is received by the node from the LTM, but due to the default route entry, is returned to the client bypassing the LTM. The routing is asymmetrical and it is possible that clients will reject the return traffic.

## 3.2 F5 Traffic Management Constructs

The key F5 BIG-IP traffic management logical constructs for the BIG-IP DNS and/or LTM are:

- **Server pool** - Server pools are logical sets of devices that are grouped together to receive and process traffic. Pools are used to efficiently distribute load across grouped server resources.
- **Virtual server** - Virtual servers are a traffic-management object that is represented in the BIG-IP system by a destination IP address and service port. Virtual servers require one or more pools.
- **Virtual address** - A virtual addresses is an IP address associated with a virtual server, commonly referred to as a VIP (virtual IP). In the BIG-IP system a virtual address cannot be explicitly created. VIPs are created by the BIG-IP system when a virtual server is created. Many virtual servers can share the same IP address.
- **Node** - A BIG-IP node is a logical object on the BIG-IP system that identifies the IP address or a fully-qualified domain name (FQDN) of a server that hosts applications. Nodes can be created explicitly or automatically when a pool member is added to a load balancing pool.
- **Pool member** - A pool member is a service on a node and is designated by an IP address and service (port).
- **Monitor** - Monitors are pre-configured (system provided) or user created associations with a virtual server, node, pool, or pool member to determine availability and performance levels. Health or performance monitors check status on an ongoing basis, at a set interval. Monitors allow intelligent decision making by BIG-IP LTM to direct traffic away from unavailable or unresponsive virtual servers, nodes, pools, or pool members. Multiple monitors can be associated with a virtual server, node, pool, or pool member.
- **Data Center** - BIG-IP DNS only. All of the BIG-IP DNS resources are associated with a data center. BIG-IP DNS consolidates the paths and metrics data collected from the servers, virtual servers, and links in the data center. BIG-IP DNS uses that data to conduct load balancing and route client requests to the best-performing resource.

- **Wide IP (WIP)** - BIG-IP DNS only. Wide IP addresses (WIP) are one of the main BIG-IP DNS configuration elements. WIP are identified by a fully qualified domain name (FQDN). Each WIP is configured with one or more pools (global). The global pool members are LTM-hosted virtual servers. BIG-IP DNS servers act as an authority for the FQDN WIP and return the IP address of the selected LTM virtual server to use.
- **Listener** - BIG-IP DNS only. A listener is a specialized virtual server that passively checks for DNS packets on port 53 and the IP address assigned to the listener. When a DNS query is sent to the IP address of the listener, BIG-IP DNS either handles the request locally or forwards the request to the appropriate resource.
- **Probe** - BIG-IP DNS only. A probe is an action taken by a BIG-IP system to acquire data from other network resources.

Figure 3 below shows the relationships between applications, virtual servers, server pools, pool members (ECS nodes), and health monitors.

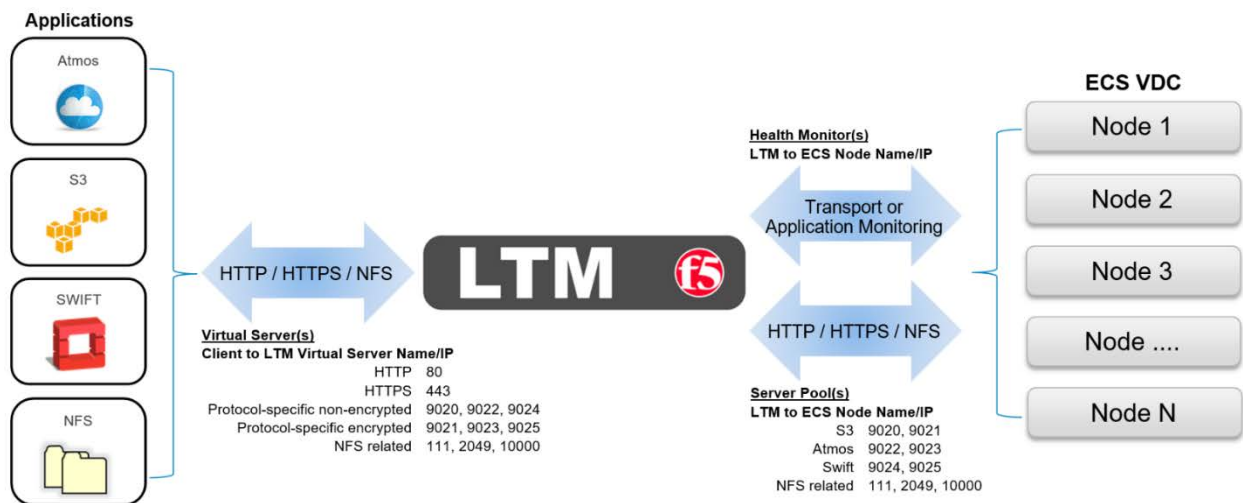


Figure 3 - Relationships between LTM, applications, virtual servers, server pools, pool members and health monitors

Figure 4 below shows the BIG-IP Wide IP (FQDN) to global pool member (LTM/site/virtual server) association.

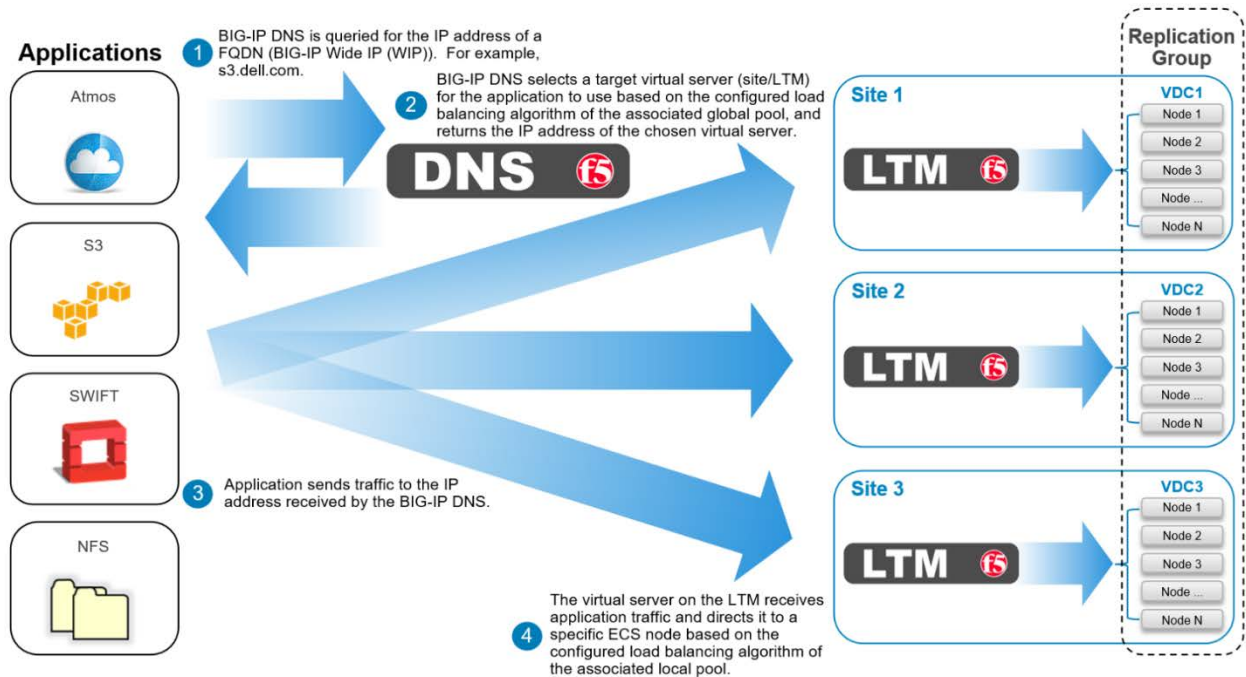


Figure 4 - BIG-IP Wide IP (FQDN) to global pool member (LTM/site/virtual server) association

### 3.3 F5 Device Redundancy

The BIG-IP DNS and LTM can be deployed as a standalone device or as a group of two or more identical devices for redundancy. In the single-device, or standalone scenario, one device handles all application traffic. The obvious downside with this is that if the device fails applications may experience a complete interruption in service. In a single-site, single-LTM deployment all application traffic will fail if the LTM becomes unavailable.

In multiple-site ECS deployments if each site contains a single LTM and a site's LTM fails, BIG-IP DNS can direct applications to use the LTM at an alternate site that is a member of the same replication group. So as long as an application can access storage at non-local sites within an acceptable level of performance, deploying a single LTM at each site may allow for significant cost savings. That is, so long as an application is tolerant of any increased latency caused by accessing data using a non-local site, purchasing two or more LTM for each site may not be necessary. Understanding the tradeoffs between implementing single or multiple devices, and the related application performance requirements, is important in developing a deployment best suited for your needs. Also an understanding of the concept of object owner on ECS, the access during outage (ADO) configuration and impact to object accessibility during temporary site outage (TSO) are all critical to consider when planning for multisite multi-access object namespace. Refer to the ECS Architectural and Overview whitepaper here for more details: <http://www.emc.com/collateral/white-papers/h14071-ecs-architectural-guide-wp.pdf>.

BIG-IP HA key constructs are:

- **Device trust** - Device trust establishes trust relationships between BIG-IP devices through mutual certificate-based authentication. A trust domain is a collection of BIG-IP devices that trust one another and is a prerequisite for creating a device group for ConfigSync and failover operations. The trust domain is represented by a special system-generated and system-managed device group named *device\_trust\_group*, which is used to synchronize trust domain information across all devices.
- **Device group** - A collection of BIG-IP devices that trust each other and can synchronize, and sometimes fail over, their BIG-IP configuration data. There are two types of device groups:
  1. A **Sync-Only** device group contains devices that synchronize configuration data, such as policy data, but do not synchronize failover objects.
  2. A **Sync-Failover** device group contains devices that synchronize configuration data and support traffic groups for failover purposes when a device becomes unavailable.
- **Traffic group** - A traffic group is a collection of related configuration objects (such as a self IP address, floating self IP address and MAC masquerading address) that run on a BIG-IP system and process a particular type of application traffic. When a BIG-IP system becomes unavailable, a floating traffic group can migrate to another device in a device group to ensure that the application traffic being handled by the traffic group continues to be processed with little to no interruption in service.

With two or more local devices, a logical BIG-IP device group can allow application traffic to be processed by more than one device. In a HA setup, during service interruption of a BIG-IP device, traffic can be routed to the remaining BIG-IP device(s) which allows applications to experience little if any interruption to service. The HA failover type discussed in this paper is sync-failover. This type of failover allows traffic to be routed to working BIG-IP devices during failure.

There are two redundancy modes available when using the sync-failover type of traffic group. They are:

1. **Active/Standby** - Two or more BIG-IP DNS or LTM devices belong to a device group. For the BIG-IP DNS, only one device actively listens and responds to DNS queries. The other device(s) are in standby mode and available to take over the active role in a failover event. For the LTM, one device in each floating traffic group, actively processes application traffic. The other devices associated with the floating traffic group are in standby mode and available to take over the active role in a failover event.
2. **Active/Active** - Two or more LTM devices belong to a device group. Both devices can actively process application traffic. This is accomplished by creating two or more BIG-IP floating traffic groups. Each traffic group can contain and processes traffic for one or more virtual servers.

It is key to understand that a single virtual IP address can be shared by many virtual servers, however, a virtual IP address may only be associated with one traffic group. If for example a pair of LTM are configured with many virtual servers that all share the same virtual IP address, all virtual servers can only be served by one LTM at a time regardless of the redundancy mode configured. This means in order to distribute application traffic to more than one LTM device, two or more unique virtual IP addresses must be in use between two or more virtual servers.

## 4 ECS Configuration

There is generally no special configuration required to support load balancing strategies within ECS. ECS is not aware of any BIG-IP DNS or LTM systems and is strictly concerned, and configured with, ECS node IP addresses, not, virtual addresses of any kind. Regardless of whether the data flow includes a traffic manager, each application that utilizes ECS will generally have access to one or more buckets within a namespace. Each bucket belongs to a replication group and it is the replication group which determines both the local and potentially global protection domain of its data as well as its accessibility. Local protection involves mirroring and erasure coding data inside disks, nodes, and racks that are contained in an ECS storage pool. Geo-protection is available in replication groups that are configured within two or more federated VDCs. They extend protection domains to include redundancy at the site level.

Buckets are configured for a single object API. A bucket can be an S3 bucket, an Atmos bucket, or a Swift bucket, and each bucket is accessed using the appropriate object API. Buckets can also be file enabled. Enabling a bucket for file access provides additional bucket configuration and allows application access to objects using NFS and/or HDFS.

Application workflow planning with ECS is generally broken down to the bucket level. The ports associated with each object access method, along with the node IP addresses for each member of the bucket's local and remote ECS storage pools, are the target for client application traffic. This information is what is required during LTM configuration. In ECS, data access is available via any node in any site that serves the bucket. In directing the application traffic to an F5 virtual address, instead of directly to an ECS node, load balancing decisions can be made which support HA and provide the potential for improved utility and performance of the ECS cluster.

The following tables provide the ECS configuration used for application access via S3 and NFS utilizing the BIG-IP DNS and LTM devices as described in this document. Note the configuration is sufficient for use by applications whether they connect directly to ECS nodes, they connect to ECS nodes via LTMs, and/or they are directed to LTMs via a BIG-IP DNS.

In our reference example, two five node ECS VDCs were deployed and federated using the ECS Community Edition 3.0 software on top of the CentOS 7.x operating system inside a VMWare ESXi lab environment. Virtual systems were used to ensure readers could successfully deploy a similar environment for testing and to gain hands-on experience with the products. A critical difference in using virtual ECS nodes, as opposed to ECS appliances, is that the primary and recommended method for monitoring an S3 service relies upon the underlying ECS fabric layer which is not in place in virtual systems. Because of this monitoring using the S3 Ping method is shown against physical ECS hardware in Appendix A, Creating a Custom S3 Monitor.



What follows are several tables with the ECS configuration used in our examples. Each table is preceded by a brief description.

Each site of the two sites has a single storage pool that contains all five of the site's ECS nodes.

Table 2 - ECS site's storage pools

Site 1 (federated with Site 2)	
<a href="https://192.168.101.11/#/vdc/provisioning/storagePools">https://192.168.101.11/#/vdc/provisioning/storagePools</a>	
Storage Pool (SP)	s1-ecs1-sp1
Name	ecs-1-1.kraft101.net 192.168.101.11 ecs-1-2.kraft101.net 192.168.101.12 ecs-1-3.kraft101.net 192.168.101.13 ecs-1-4.kraft101.net 192.168.101.14 ecs-1-5.kraft101.net 192.168.101.15
Nodes	
Site 2 (federated with Site 1)	
<a href="https://192.168.102.11/#/vdc/provisioning/storagePools">https://192.168.102.11/#/vdc/provisioning/storagePools</a>	
Storage Pool (SP)	s2-ecs1-sp1
Name	ecs-2-1.kraft102.net 192.168.102.11 ecs-2-2.kraft102.net 192.168.102.12 ecs-2-3.kraft102.net 192.168.102.13 ecs-2-4.kraft102.net 192.168.102.14 ecs-2-5.kraft102.net 192.168.102.15
Nodes	

The first VDC is created at Site 1 after the storage pools have been initialized. A VDC access key is copied from Site 2 and used to create the second VDC at Site 1 as well.

Table 3 - VDC name and endpoints

Site 1	
<a href="https://192.168.101.11/#/vdc/provisioning/virtualdatacenter">https://192.168.101.11/#/vdc/provisioning/virtualdatacenter</a>	
Virtual Data Center (VDC)	s1-ecs1-vdc1
Name	
Replication and Management Endpoints	192.168.101.11,192.168.101.12,192.168.101.13,192.168.101.14,192.168.101.15
Site 2	
Virtual Data Center (VDC)	s2-ecs1-vdc1
Name	
Replication and Management Endpoints	192.168.102.11,192.168.102.12,192.168.102.13,192.168.102.14,192.168.102.15

A replication group is created and populated with the two VDCs and their storage pools. Data stored using this replication group is protected both locally, at each site, and globally through replication to the second site. Applications can access all data in the replication group via any of the nodes in either of the VDC's associated storage pool.

Table 4 - Replication group

<a href="https://192.168.101.11/#/vdc/provisioning/replicationGroups/">https://192.168.101.11/#/vdc/provisioning/replicationGroups/</a>	
Replication Group (RG)	ecs-rg1-all-sites
Name	
VDC: SP	s1-ecs1-vdc1: s1-ecs1-sp1 s2-ecs1-vdc1: s2-ecs1-sp1



A namespace is created and associated with the replication group. This namespace will be used for S3 and NFS traffic.

Table 5 - Namespace

Namespace (NS)		<a href="https://192.168.101.11/#/vdc/provisioning/namespace">https://192.168.101.11/#/vdc/provisioning/namespace</a>
Name	webapp1	
Replication group	ecs-rg1-all-sites	
Access During Outage	Enabled	

An object user is required for accessing the namespace and created as per Table 6 below.

Table 6 - Object user

User		<a href="https://192.168.101.11/#/vdc/provisioning/users/object">https://192.168.101.11/#/vdc/provisioning/users/object</a>
Name	webapp1_user1	
NS	webapp1	
Object access	S3	
User key	Akc0GMp2W4jZyu/07A+HdRjLtamiRp2p8xp3at7b	

An NFS user and group are created as shown in Table 7 below. The NFS user is specifically tied to the object user and namespace created above in Table 6.

Table 7 - NFS user and group

File user/Group mapping		<a href="https://192.168.101.11/#/vdc/provisioning/file/ns1/userMapping/">https://192.168.101.11/#/vdc/provisioning/file/ns1/userMapping/</a>
User	Name: webapp1_user1, ID: 1000, Type: User, NS: webapp1	
Group	Name: webapp1_group1, ID: 1000, Type: Group, NS: webapp1	

A file-enabled S3 bucket is created inside the previously created namespace using the object user as bucket owner. The bucket is associated with the namespace and replication group. Table 8 below shows the bucket configuration.

Table 8 - S3 bucket configuration

Bucket		<a href="https://192.168.101.11/#/vdc/provisioning/buckets/">https://192.168.101.11/#/vdc/provisioning/buckets/</a>
Name	s3_webapp1	
NS: RG	webapp1: ecs-rg1-all-sites	
Bucket owner	webapp1_user1	
File system	Enabled	
Default bucket group	webapp1_group1	
Group file permissions	RWX	
Group directory permissions	RWX	
Access During Outage	Enable	

To allow for access to the bucket by NFS clients, a file export is created as per Table 9 below.

Table 9 - NFS export configuration

File export		https://10.10.10.101/#/vdc//provisioning/file//exports
Namespace	webapp1	
Bucket	s3_webapp1	
Export path	/webapp1/s3_webapp1	
Export host options	Host: * Summary: rw,async,authsys	

Table 10 below lists the DNS records for the ECS nodes. The required reverse lookup entries are not shown.

Table 10 - DNS records for ECS nodes and Site 1 and Site 2

DNS records (corresponding reverse entries not shown but are required)			
DNS record entry	Record type	Record data	Comments
ecs-1-1.kraft101.net	A	192.168.101.11	Public interface node 1 ecs1 site1
ecs-1-2.kraft101.net	A	192.168.101.12	Public interface node 2 ecs1 site1
ecs-1-3.kraft101.net	A	192.168.101.13	Public interface node 3 ecs1 site1
ecs-1-4.kraft101.net	A	192.168.101.14	Public interface node 4 ecs1 site1
ecs-1-5.kraft101.net	A	192.168.101.15	Public interface node 5 ecs1 site1
ecs-2-1.kraft102.net	A	192.168.102.11	Public interface node 1 ecs1 site2
ecs-2-2.kraft102.net	A	192.168.102.12	Public interface node 2 ecs1 site2
ecs-2-3.kraft102.net	A	192.168.102.13	Public interface node 3 ecs1 site2
ecs-2-4.kraft102.net	A	192.168.102.14	Public interface node 4 ecs1 site2
ecs-2-5.kraft102.net	A	192.168.102.15	Public interface node 5 ecs1 site2

## 5 F5 Configuration

Two configuration sections follow, the first for BIG-IP LTM and the second for BIG-IP DNS. Each section provides a couple examples along with the reference architecture. Context around related deployment options are also mentioned.

We make the assumption that whether BIG-IP devices are deployed in isolation or as a group, the general guidelines provided are the same. F5 documentation should be reviewed so that any differences between deploying single and redundant devices are understood.

All of the examples provided use the ECS VDCs as configured in the tables above. The examples provided were deployed in a lab environment using the ECS Community Edition, v3.0, and virtual F5 edition, version 13.0.0 (Build 2.0.1671 HF2). Unlike above in the ECS configuration section, steps are provided in the examples below on how to configure the F5 BIG-IP traffic management constructs. We do not provide any BIG-IP base configuration steps such as licensing, naming, NTP, and DNS.

For each site BIG-IP LTM is deployed and configured to balance the load to ECS nodes. In federated ECS deployments a BIG-IP DNS can be used to direct application traffic to the most suitable site.

### 5.1 F5 BIG-IP LTM



Figure 5 - LTM high-level overview

Local traffic managers, as seen in the middle of Figure 5 above, listen and receive client application traffic on virtual servers. Virtual servers are generally identified by a FQDN and port, for example, s3.site1.ecstme.org:9020. Virtual servers process traffic for either one service port, or all service ports. Virtual servers are backed by one or more server pools. The server pools have members assigned to them. The pool members consist of ECS nodes and identified by their hostname and port, for example, ecs-1-1:9020. A pool member, like a virtual server, can listen on either one port, or on all ports.

An LTM makes a decision for each application message it receives on a virtual server to determine which specific pool member in the virtual server's configured server pool(s) should receive the message. The LTM uses the configured load balancing algorithm to make this determination.

The LTM uses health monitors to keep an up-to-date status of all individual pool members and their relevant services. As a node or node's required service(s) become unavailable the LTM ceases to send application

traffic to the pool member and subsequent messages are forwarded to the next available pool member per the configured load balancing algorithm.

In this document examples are shown with virtual servers and server pool members that service single ports, S3 with Secure Socket Layer (SSL) at port 9021, for example. It is possible for a virtual server and related pool members to listen on all service ports. This would allow for all application traffic to point to one virtual server that services all the related ports. F5 BIG-IP systems have a configuration element called iRule. Using iRules allows administrators to detail rules, for example, to only process traffic on specific ports and to drop all other traffic. Using an iRule could allow a virtual server to listen on all ports but only process traffic for ports 111, 2049, and 10000, the NFS-related ports ECS uses. Readers are encouraged to research iRules to determine if they are worthwhile for use in their BIG-IP configuration.

NOTE: Local applications may use the S3-specific application ports, 9020 and 9021. For workflows over the Internet it is recommended to use ports 80 and 443 on the front end and ports 9020 and 9021 on the backend. This is because the Internet can handle these ports without problem. Using 9020 or 9021 may pose issues when used across the Internet.

### 5.1.1 Example: S3 to Single VDC with Active/Standby LTM Pair

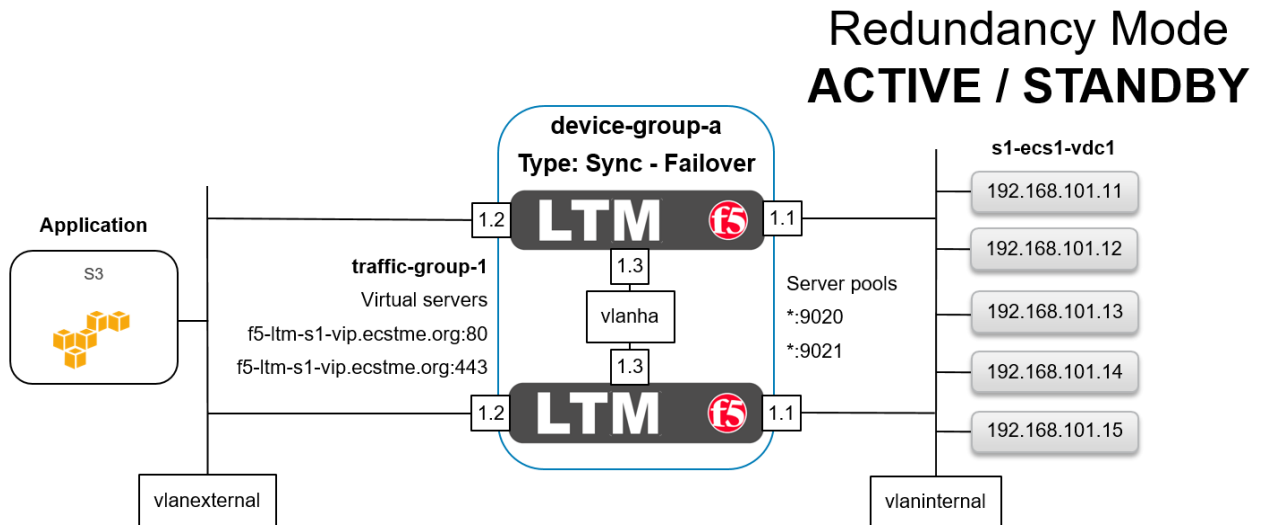


Figure 6 - LTM in active/standby redundancy mode

Figure 6 above shows the basic architecture for this example of an S3 application access to ECS via an Active/Standby HA pair of LTM at a single site. A client application is shown on the far left. In our case it resides in the same subnet as the LTM but it could be on any local or global network. Each of our LTM devices has four network interfaces. The first, not shown above, is used for management access. Interfaces 1.1 are assigned to a VLAN named *vlaninternal*. In general load balancing terms this is often also called the back-end or server-side network. Interfaces 1.2 are assigned to VLAN *vlanexternal*. Generally this is referred to as the front-end or client-side network. Interfaces 1.3 are assigned to VLAN *vlanha*. The HA VLAN is used for high availability functionality along with synchronization of the device's configuration and client connection mirroring. A BIG-IP device group, *device-group-a*, in our case, logically contains both devices. The device group's failover type, *sync-failover*, is configured with the redundancy mode set to Active/Standby. In Active/Standby redundancy mode, one device actively processes all application traffic at any given time. The

other device is idle and in standby mode waiting to take over processing application traffic if necessary. On the far right is our VDC at Site 1. All of the ECS nodes are each put in to two BIG-IP server pools as referenced in Figure 6 above using an asterisk. One server pool is configured with port 9020 and the other server pool for 9021. Each node is considered both a BIG-IP node and a BIG-IP pool member.

In between the client and LTM, a BIG-IP traffic group, *traffic-group-1*, in our example, is shown. It contains two virtual servers, one for each type of S3 traffic we provide access for in this example. Applications will point to the FQDN and appropriate port to reach the VDC.

In between the LTM and VDC two server pools are shown, one for each service port we required. Each of these server pools contain all VDC nodes. Not shown is the health monitoring. Monitoring will be discussed further down as the example continues.

Here is a general overview of the steps in this example:

1. Configure the first LTM.
2. Configure the second LTM and pair the devices for HA.

**Step 1:** Configure the first LTM.

GUI screenshots are used to demonstrate LTM configuration. Not shown are the initial F5 BIG-IP Virtual Edition basic setup of licensing and core infrastructure components such as NTP and DNS. Table 11 below shows these details used for each LTM device.

Table 11 - LTM base device details

Devices	System » Platform » Configuration
Hostname: Management IP	f5-ltm-s1-1.ecstme.org: 172.16.3.2
	f5-ltm-s1-2.ecstme.org: 172.16.3.3
	f5-ltm-s2-1.ecstme.org: 172.16.3.5
NTP	System » Configuration : Device : NTP
Time servers	10.254.140.21
DNS	System » Configuration : Device : DNS
DNS lookup servers list	10.246.150.22
DNS search domain list	localhost, ecstme.org

Figure 7 below shows three VLANs and associated network interfaces used. At this point in our example, and during single device configuration, the third interface and associated VLAN is configured but not used until later in the HA example. VLAN configuration is simple and consists of naming the VLAN and assigning a physical network interface, either tagged or untagged, as the resource. We use an untagged interface as each interface in our design only services one VLAN. None of the devices aside from the F5 BIG-IP DNS and LTM in our test environment use VLAN tagging. If an interface belongs to multiple VLANs it should be tagged. For more information on tagged versus untagged interfaces, and when to use each, refer to F5 documentation. In addition, refer to Dell EMC ECS Networking and Best Practices whitepaper here: <https://www.emc.com/collateral/white-paper/h15718-ecs-networking-bp-wp.pdf>.

Network » VLANs : VLAN List

VLAN List | VLAN Groups

Search Create...

<input checked="" type="checkbox"/>	Name	Application	Tag	Untagged Interfaces	Tagged Interfaces	Partition / Path
<input type="checkbox"/>	vlanexternal		4093	1.2		Common
<input type="checkbox"/>	vlanhs		4092	1.3		Common
<input type="checkbox"/>	vlaninternal		4094	1.1		Common

Delete...

Figure 7 - LTM VLANs with interfaces

Figure 8 below shows the non-management network interfaces on this LTM. As mentioned above, during single device deployment only two interfaces are required to carry application traffic in a typical two-arm architecture as described earlier in the F5 Networking Constructs section. The third, in our example, is configured in advance in preparation for HA.

Network » Interfaces : Interface List

Interface List | Interface Mirroring | LLDP | Statistics

Interfaces

<input checked="" type="checkbox"/>	Status	Name	MAC Address	Media Speed	VLAN Count	Trunk	Forwarding Mode
<input checked="" type="checkbox"/>	UP	1.1	00:50:56:8f:39:c4	10000	1		Forwarding
<input type="checkbox"/>	UP	1.2	00:50:56:8f:00:6c	10000	1		Forwarding
<input type="checkbox"/>	UP	1.3	00:50:56:8f:29:9a	10000	1		Forwarding

Enable Disable

Figure 8 - LTM non-management network interfaces with MAC addresses

Figure 9 below shows the self IP addresses required for this example. Each of the interfaces used for application traffic are assigned two self IP addresses. The local addresses are assigned to the interface and are assigned to the built-in traffic group named *traffic-group-local-only*. In preparation for HA a floating self IP address is also created for each interface carrying application traffic. In HA they will move between devices as needed in service to the active unit. The floating self IP addresses are associated with traffic group named *traffic-group-1*. Also shown in the graphic is the self IP address for HA. At this point, in a single device deployment configuration, only two self IP addresses are required for a two-arm architecture, one for each VLAN and interface that will handle application traffic. Later when we synchronize configuration between the LTM pair, the static self IP addresses will not be part of the settings shared between the two devices. That is, these self IP addresses are local only to the device they're created on.

Network » Self IPs

Self IP List

Search Create...

<input checked="" type="checkbox"/>	Name	Application	IP Address	Netmask	VLAN / Tunnel	Traffic Group	Partition / Path
<input type="checkbox"/>	selfip1.1floating		192.168.101.5	255.255.255.0	vlaninternal	traffic-group-1	Common
<input type="checkbox"/>	selfip1.1local		192.168.101.3	255.255.255.0	vlaninternal	traffic-group-local-only	Common
<input type="checkbox"/>	selfip1.2floating		10.246.150.88	255.255.255.0	vlanexternal	traffic-group-1	Common
<input type="checkbox"/>	selfip1.2local		10.246.150.88	255.255.255.0	vlanexternal	traffic-group-local-only	Common
<input type="checkbox"/>	selfip1.3		192.168.255.4	255.255.255.0	vlanha	traffic-group-local-only	Common

Delete...

Figure 9 - LTM static and floating self IP addresses

Figure 10 below begins to show the creation of one of the server pools.

Local Traffic » Pools : Pool List » New Pool...

Configuration: Advanced

Name	pool-all-nodes-9020
Description	
Health Monitors	<div style="display: flex; justify-content: space-between;"> <div style="border: 1px solid gray; padding: 5px;"> <p>Active</p> <p>/Common</p> <p>tcp</p> </div> <div style="border: 1px solid gray; padding: 5px;"> <p>Available</p> <p>https_443</p> <p>https_head_f5</p> <p>inband</p> <p>tcp_half_open</p> <p>udp</p> </div> </div>
Availability Requirement	All Health Monitor(s)
Allow SNAT	Yes
Allow NAT	Yes
Action On Service Down	Reject

Figure 10 - LTM server pool creation

The pool is named, assigned a health monitor, and configured with “Reject” as the desired “Action on Service Down” method. Built-in tcp health monitoring is used for the lab. All of the other settings in the upper ‘Configuration’ section for new pool creation dialog are default. When using F5 with a real hardware appliance the best practice is to use S3 Ping. This is explained further and demonstrated in Appendix A: Creating a Custom S3 Monitor.

The "Action on Service Down" setting controls LTM's connection management behavior when a monitor transitions a pool member to a DOWN state after it has been selected as the load balancing target for a connection. The best option is entirely dependent on that application's client and server implementations.

The possible options are:

- **None** - Default setting. The LTM will continue to send data on established connections as long as client is sending and server is responding.
- **Reject** - Most commonly used option. Used to explicitly close both sides of the connection when the server goes DOWN. This option often results in the quickest recovery of a failed connection since it forces the client side of the connection to close.
- **Drop** - The LTM will silently drop any new client data sent on established connections.
- **Reselect** - The LTM will choose another pool member if one is available and rebind the client connection to a new server side flow.

In the lower section of the new pool creation dialog the load balancing method is selected and pool members are assigned. Figure 11 below shows a list of all available load balancing methods. We use the often utilized least connections option. Consult F5 directly for the specific details and reasons why any of the available options may be desirable for a given workload.

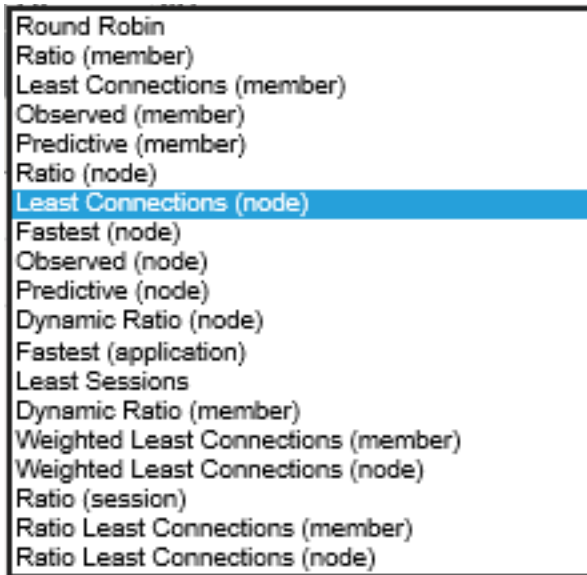


Figure 11 - LTM load balancing methods available



**Resources**

Load Balancing Method	Least Connections (node) ▼
Priority Group Activation	Disabled ▼
New Members	<input type="radio"/> New Node <input checked="" type="radio"/> New FQDN Node
	Node Name: <input type="text" value="ecs-1-5"/> (Optional)
	FQDN: <input type="text" value="ecs-1-5.kraft101.net"/> x
	Service Port: <input type="text" value="9020"/> <input type="button" value="Select..."/> ▼
	Auto Populate: <input type="text" value="Enabled"/> ▼
	<input type="button" value="Add"/>
<pre> R:1 P:0 C:0 ecs-1-1 ecs-1-1.kraft101.net :9020 R:1 P:0 C:0 ecs-1-2 ecs-1-2.kraft101.net :9020 R:1 P:0 C:0 ecs-1-3 ecs-1-3.kraft101.net :9020 R:1 P:0 C:0 ecs-1-4 ecs-1-4.kraft101.net :9020 R:1 P:0 C:0 ecs-1-5 ecs-1-5.kraft101.net :9020           </pre>	
<input type="button" value="Edit"/> <input type="button" value="Delete"/>	
<input type="button" value="Cancel"/> <input type="button" value="Repeat"/> <input type="button" value="Finished"/>	

Figure 12 - LTM server pool final configuration

Figure 12 above shows the final configuration selections with all pool members. The S3 with SSL pool is configured in the same manner using port 9021. The S3 with SSL pool allows for the encryption of traffic between the client and ECS node using an ECS-generated certificate. In the lab we use a self-signed certificate. In production environments it is recommended to use a certified signing authority.

It is generally recommended however that SSL connections be terminated at the LTM. Use of SSL encryption during transport is CPU intensive and the F5 BIG-IP software is specially built to handle this processing efficiently. F5 hardware also has dedicated Application-Specific Integrated Circuits (ASIC) to further enhance efficiency in processing this traffic. SSL can be terminated at ECS, the LTM, or both. By terminating SSL at both places, a client will establish a secure connection to the F5 using an F5 generated certificate, and the LTM will establish a secure connection to the ECS node using the ECS certificate. Twice the amount of processing is required when encrypting each of the two connections individually. Traffic encryption options should be analyzed for each application workflow. For some workflows, for example when no personally identifiable information is transmitted, no encryption may be suitable. All traffic will pass over the wire in clear text fashion and no encryption-related CPU processing is required. For other workflows encryption on the external connection is appropriate but non-encrypted connectivity on the internal network may be fine. Some banks and other highly secure facilities use encryption for each of the two connections.

Figure 13 below shows both pools in the system. Although there are only five ECS nodes, ten members are shown for each pool. This is because the system lists both ephemeral and non-ephemeral members. Per the F5 documentation, during pool creation, when the default setting of “Auto Populate on” is used, the system creates an ephemeral node for each IP address returned as an answer to a DNS query. Also, when a DNS answer shows that the IP address of an ephemeral node doesn't exist anymore, the system deletes the ephemeral node.

These two pools can handle all of our S3 traffic needs. Traffic to pool members listening on port 9020 is not encrypted. Traffic to pool members listening on port 9021 is encrypted.

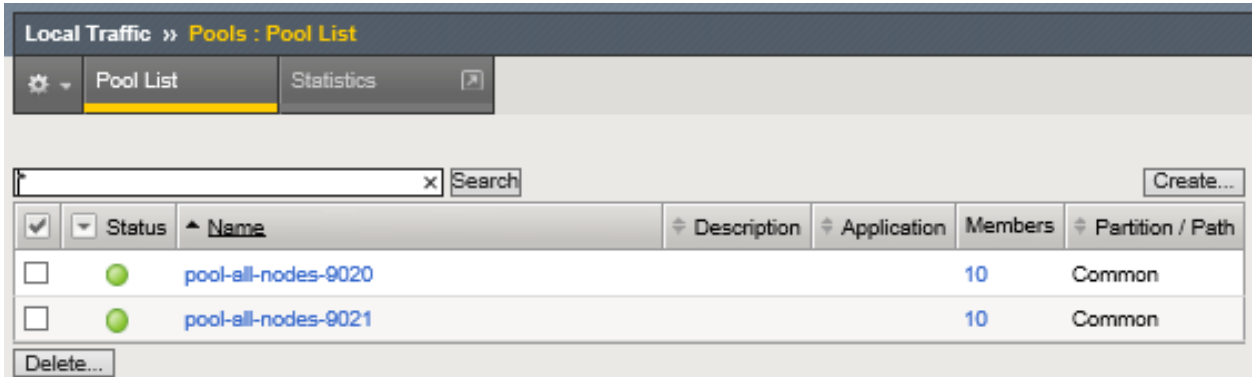


Figure 13 - LTM S3-related server pools

Next is the creation of a virtual server. Figure 14 below shows the assigned name, destination address, and service port for an S3 virtual server that serves non-encrypted traffic. A source address or network can optionally be added as well. The destination IP address shown is the Virtual IP address that will be created on the system. Client applications will point to this IP address. We optionally assign a name to the address in DNS to allow clients to use a FQDN.

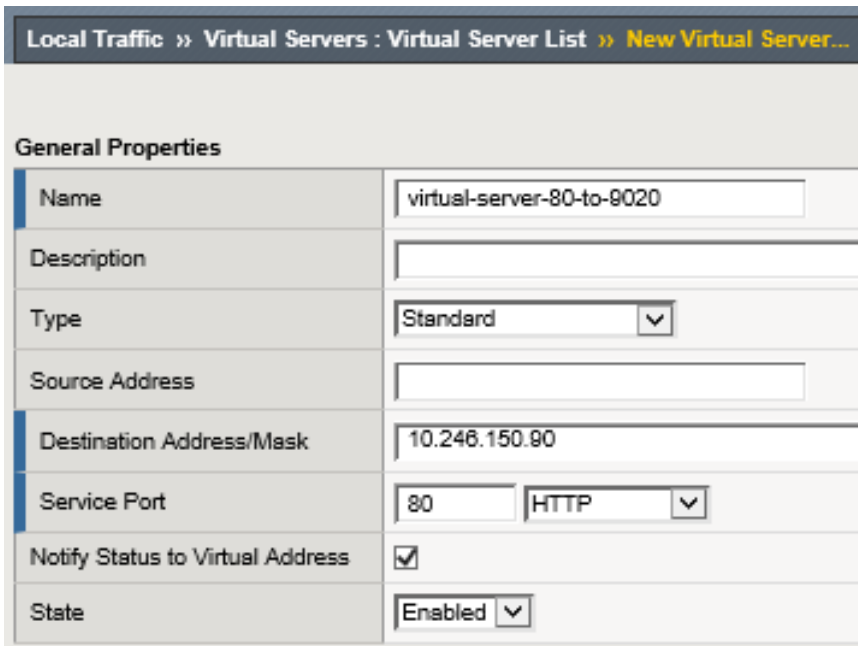


Figure 14 - LTM new virtual server creation dialog

The type of virtual server we use is Standard. Several other types exist and Figure 15 shows the list of options provided in the GUI. F5 documentation should be reviewed to understand the available virtual server types. We choose Standard however using another type of virtual server, such as Performance (HTTP), may be more suitable for production access.

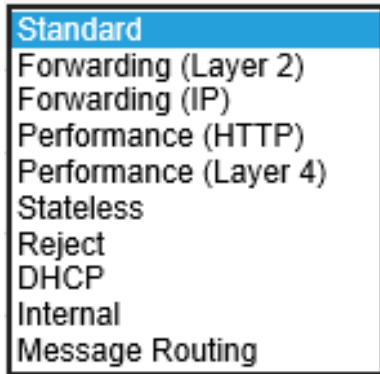


Figure 15 - LTM virtual server options

The remaining values are not shown. For this example, TCP is selected for protocol. Other protocol options exist and are UDP, SCTP, or All. We'll use "All" option when configuring NFS-related virtual servers in the NFS example. The pool we selected was *pool-all-nodes-9020*. We also selected a "Default Persistence Profile" of *source\_addr*. Figure 16 below shows a list of available values for this setting. F5 documentation should be consulted for details of each to see if they are more appropriate for a given workload. We selected and recommend *source\_addr* as the persistence profile because it keeps a client's application traffic pinned to the same pool member which allows for efficient use of ECS cache. We generally recommend *source\_addr* as the persistence profile for all virtual servers in use with ECS. Source address affinity persistence directs session requests to the same server based solely on the source IP address of a packet. In circumstances where many applications on a few number of clients connect to ECS through an LTM another persistence profile may be better suited because it could cause an uneven load distribution across ECS nodes. As with all configuration choices, be sure to understand the options available to make appropriate traffic management decisions for each workflow and architecture deployed.

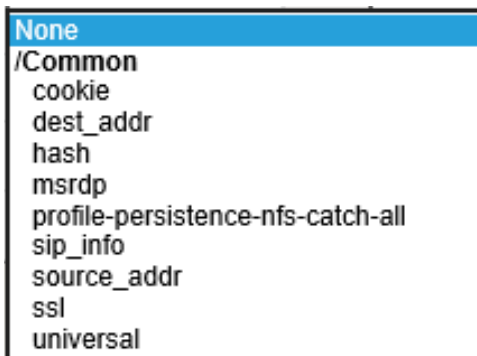


Figure 16 - LTM persistence profile options

Figure 17 below shows the virtual server for non-encrypted S3 traffic and some other virtual servers configured to this point. Three other virtual servers exist on this LTM and are not covered in our example. By default BIG-IP LTM has a default deny all policy. In order for traffic to traverse through an LTM a rule must explicitly exist. The topology used in our lab does not have a router in the mix so we created virtual servers to allow traffic to be forwarded as needed. The additional rules shown allow, for example, our two ECS VDCs to communicate with each other and for the underlying CentOS to access the Internet for application binaries and such.

Local Traffic » Virtual Servers : Virtual Server List

Virtual Server List | Virtual Address List | Statistics

Search [ ] [Create...]

<input type="checkbox"/>	Status	Name	Description	Application	Destination	Service Port	Type	Resources	Partition / Path
<input type="checkbox"/>		forwardingvserver101net			192.168.101.0/24	0 (Any)	Forwarding (IP)	Edit...	Common
<input type="checkbox"/>		forwardingvserver10net			10.246.150.0/24	0 (Any)	Forwarding (IP)	Edit...	Common
<input type="checkbox"/>		forwardingvservercatchall			Any IPv4	0 (Any)	Forwarding (IP)	Edit...	Common
<input type="checkbox"/>		virtual-server-80-to-9020			10.246.150.90	80 (HTTP)	Standard	Edit...	Common

Enable | Disable | Delete...

Figure 17 - Configured virtual servers

**Step 2:** Configure second LTM, pair the devices for HA.

We add a second LTM to the same network as the first and configure the network interfaces similar to the first, using unique self IP addresses. To create the pair, in the WebUI, Device Management is selected and Device Trust then Device Trust Members is chosen. The Add button is selected in the Peer and Subordinate Devices section to add the peer device. This was done from our first LTM set up, however, it can be accomplished via either device. The second device's management IP address, administrator username, and administrator password are entered. A retrieve Device Information button appears as shown below in Figure 18. Once clicked, some of the remote devices information is displayed. These actions are only required on one of the devices.

Device Management » Device Trust

Retrieve Device Credentials (Step 1 of 3)

Device Type	Peer
Device IP Address	172.16.101.4
Administrator Username	root
Administrator Password	••••••••

Cancel | Retrieve Device Information

Figure 18 - Retrieve device information button

After the two devices trust each other, a device group can be created. In our environment we called the device group *device-group-a*. The group type of Sync-Failover is chosen and both devices are added as members. The Network Failover box is checked. For now we will sync manually so none of the other boxes require check marks. A traffic group, *traffic-group-1*, is created automatically during this process.

Device Management, Devices, Device List, select (self), leads to Properties page along with Device Connectivity list menu. The list menu contains, ConfigSync, Failover Network and Mirroring. For the ConfigSync configuration we select our HA local address. For the Failover Network configuration we choose

HA for the unicast address. For Mirroring we chose HA IP address for the Primary, and *vlanexternal* address for the secondary. The connection state mirroring feature allows the standby unit to maintain all of the current connection and persistence information. If the active unit fails and the standby unit takes over, all connections continue, virtually uninterrupted.

With all of the required configuration we sync the first LTM to the group. This is done via Device Management, Overview, select self, choose sync device to group, and clicking Sync. Then we can sync the group to the second device.

Here is a table which provides a summary all configuration elements to this point.

Table 12 - LTM configuration elements

Device Group		Device Management » Device Groups	
Name	device-group-a		
Group type	Sync-Failover		
Sync type	Manual with incremental sync		
Members	f5-ltm-s1-1, f5-ltm-s1-2		
Traffic Group		Device Management » Traffic Groups	
Name	traffic-group-1		
Pools		Local Traffic » Pools	
Name	pool-all-nodes-9020		
Health monitors	tcp		
Action on service down	Reject		
Members	ecs-1-1:9020, ecs-1-2:9020, ecs-1-3:9020, ecs-1-4:9020, ecs-1-5:9020		
Virtual Servers		Local Traffic » Virtual Servers	
Name	virtual-server-80-to-9020		
Type	Standard		
Destination address	10.246.150.90		
Service port	9020		
Protocol	* All protocols		
Default pool	pool-all-nodes-9020		
Default persistence profile	source_addr		

At this point we have two LTM devices configured for HA at Site 1. In this Active/Standby scenario only one device handles application traffic at any time. Client connections and application sessions are mirrored to the standby device to allow failover with minimal interruption. Applications point to the single virtual server name or IP address. With *source\_addr* as the default persistence configuration chosen, connections from each client are directed by the LTM to a single ECS node. If that ECS node should fail the connections are reset and subsequent application traffic is pointed to the next indicated ECS node.

Table 13 below shows the LTM-related DNS entries.

Table 13 - LTM-related DNS entries

DNS record entry	Record type	Record data	Comments
f5-ltm-s1-1.ecstme.org	A	10.246.150.86	F5 LTM Site 1 (1 of 2) self IP address
f5-ltm-s1-2.ecstme.org	A	10.246.150.87	F5 LTM Site 1 (2 of 2) self IP address
f5-ltm-s1-floater.ecstme.org	A	10.246.150.88	F5 LTM Site 1 floating self IP address
f5-ltm-s1-virtual.ecstme.org	A	10.246.150.90	F5 LTM Site 1 virtual self IP address

Organizations may choose to consider a VDC unavailable if less than a required minimum number of nodes are up. A three node minimum may be chosen as it is the minimum number of nodes required for writes to complete. ECS customers decide on their own logic and configuration on how best to accomplish this.

One approach may be to configure a virtual server that can be queried with an associated iRule that determines the number of currently active pool members. With this a pool associated with a monitor that queries the virtual server can bring down a pool if there are not enough nodes available to service requests.

Another approach involves the creation of an object whose existence can be queried on each node. A monitor which queries for the object on each node can be customized to look for a minimum number of successful responses in order to consider the pool healthy.

### 5.1.2 Example: LTM-terminated SSL Communication

Three primary configuration options are available for encrypting client traffic to ECS. They are:

1. **ECS-terminated** SSL connectivity. End-to-end traffic encryption between client and ECS.
2. **LTM-terminated** SSL connectivity. Encrypted traffic between the client and LTM. No encryption between LTM and ECS.
3. **ECS-terminated and LTM-terminated** SSL connectivity. Traffic is encrypted twice, first between client and LTM, and second between LTM and ECS.

As previously mentioned, SSL termination is a CPU intensive task. LTM hardware has dedicated hardware processors specializing in SSL processing. LTM software also has mechanisms for efficient SSL processing. It is recommended, when appropriate, to terminate SSL on the LTM and offload encryption processing overhead off of the ECS storage. Each workflow should be assessed to determine if traffic requires encryption at any point in the communication path.

Generally storage administrators use SSL certificates signed by a trusted Certificate Authority (CA). A CA-signed or trusted certificate is highly recommended for production environments. For one, they can generally be validated by clients without any extra steps. Also some applications may generate an error message when encountering a self-signed certificate. In our example we generate and use a self-signed certificate.

Both the LTM and ECS software have mechanisms to produce the required SSL keys and certificates. Private keys remain on the LTM and/or ECS and clients must have a means to trust the device's certificate. This is one disadvantage to using self-signed certificates. A self-signed certificate is its own root certificate and as such client systems will not have it in their cache of known (and trusted) root certificates. Self-signed certificates must be installed in the certificate store of any machines that will access ECS.

The first step in configuring any of the three SSL connectivity scenarios described above is the generation of the required SSL keys and associated signed certificates. Each SSL key and certificate pair can then added to a custom SSL profile on the BIG-IP LTM. Custom SSL profiles are part of the virtual server configuration.

SSL connectivity between a client and LTM requires the creation of a key and certificate pair on the LTM. This pair is then added to an SSL profile and associated with the virtual server which hosts the associated traffic.

To encrypt traffic between the LTM and ECS nodes an ECS-generated SSL key and certificate pair are required and are added to a custom SSL profile. The same ECS-generated SSL key and signed certificate pair can also be used to communicate via encrypted traffic between the client and ECS directly. In this scenario all traffic on both legs are encrypted and the LTM simply forwards packets without providing any SSL services.

The simplest and most common use is when the client to LTM traffic is encrypted and LTM to ECS is not. In this scenario the LTM offloads the CPU-intensive SSL processing from ECS.

Here is a general overview of the steps we'll walk through in this example:

1. Create SSL key and self-signed certificate on the LTM.
2. Create a custom SSL profile and configure with the key/certificate pair.
3. Create a virtual server for LTM-terminated SSL connectivity to ECS.

**Step 1:** Create SSL key and self-signed certificate on the LTM.

A single self-signed certificate is used for all three LTM devices in our lab deployment. Subject Alternate Names (SAN) are populated during the creation of the certificate to include all possible combinations of names and addresses that may be used during client connectivity through the LTM to ECS. Using a single certificate pair for all devices isn't necessary. A key and certificate can be uniquely created for each BIG-IP device. Because self-signed certificates are used in our lab, additional client-side steps are required to allow the LTM to be trusted by the client. In production environments, or, when using widely recognized public signing authorities such as CA, there shouldn't be any need for special client-side accommodations to accept the granting authority.

Creating a self-signed digital certificate on an LTM device can be done at the command prompt using openssl, or, via the WebUI by clicking on the Create button located by navigating to the System, Certificate Management, Traffic Certificate Management, SSL Certificate List page.

A new SSL Certificate configuration page is presented and the Common Name (CN) and SAN fields are populated. The end result is a new self-signed certificate which is then exported and imported in to the other two LTM devices. Figure 19 shows the fields as used in the lab. The SSL key was also exported and imported to all three LTM under the same name for consistency, along with the new certificate.

System » Certificate Management : Traffic Certificate Management : SSL Certificate List » **ssl-certificate-ecs**

**General Properties**

Name	ssl-certificate-ecs.crt
Partition / Path	Common
Certificate Subject(s)	*.wip.f5-ecs.ecstme.org

**Certificate Properties**

Public Key Type	RSA
Public Key Size	2048 bits
Expires	Jun 27 2018 14:11:31 GMT
Version	3
Serial Number	236239891
Subject	Common Name: *.wip.f5-ecs.ecstme.org Organization: Division: Locality: State Or Province: Country: US
Issuer	Self
Email	
Subject Alternative Name	IP Address:10.246.150.91, IP Address:10.246.150.90, DNS:f5-ltm-s2-virtual.ecstme.org, DNS:swift.f5-ltm-s2-virtual.ecstme.org, DNS:atmos.f5-ltm-s2-virtual.ecstme.org, DNS:s3.f5-ltm-s2-virtual.ecstme.org, DNS:*s3.f5-ltm-s2-virtual.ecstme.org, DNS:f5-ltm-s2-virtual.ecstme.org, DNS:f5-ltm-s1-virtual.ecstme.org, DNS:swift.f5-ltm-s1-virtual.ecstme.org, DNS:atmos.f5-ltm-s1-virtual.ecstme.org, DNS:s3.f5-ltm-s1-virtual.ecstme.org, DNS:*s3.f5-ltm-s1-virtual.ecstme.org, DNS:f5-ltm-s1-virtual.ecstme.org, DNS:f5-ltm-s2-1.ecstme.org, DNS:f5-ltm-s1-2.ecstme.org, DNS:f5-ltm-s1-1.ecstme.org, DNS:wip.f5-ecs.ecstme.org, DNS:*wip.f5-ecs.ecstme.org

Figure 19 - Custom SSL certificate fields

At this point a newly-created self-signed certificate exists on all three LTM in the lab deployment. Clients will be able to negotiate an encrypted session to any of the lab LTMs via SSL using any of the names or IP addresses in the SAN field.

**Step 2:** Create a custom SSL client profile and configure with the key/certificate pair.

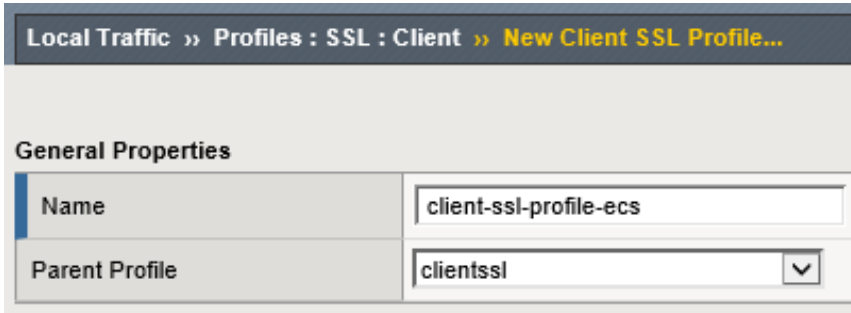
There are two types of SSL profiles for use in LTM virtual servers:

1. **Client SSL Profile.** A client SSL profile is used to enable SSL from the client to the virtual server.
2. **Server SSL Profile.** A server SSL profile is used to enable SSL from the LTM to the pool member.



In this example we create a new custom SSL client profile to allow for encrypted communication between the client and LTM. Similar steps can be used to create a custom SSL server profile which utilizes an SSL key and certificate generated for the ECS nodes. That custom SSL server profile could then be used to encrypt traffic between the LTM and ECS.

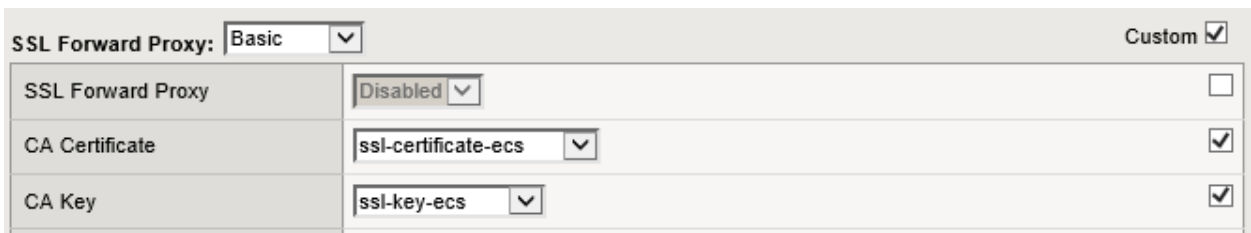
A new profile can be generated by clicking the Create button on the Local Traffic, Profiles: SSL: Client page. Profile configuration has four sections: General Properties, Configuration (Basic or Advanced), Client Authentication, and SSL Forward Proxy (Basic or Advanced). In the General Properties section a name is provided for the new profile along with the parent profile to be used which serves as a template. Figure 20 shows the values used for the example.



General Properties	
Name	client-ssl-profile-ecs
Parent Profile	clientssl

Figure 20 - LTM client SSL profile naming

Only two other values are edited for the example, specifying the `ssl-ecs-key.key` and the `ssl-certificate-ecs.crt` files as shown in Figure 21 below.



SSL Forward Proxy: Basic		Custom <input checked="" type="checkbox"/>
SSL Forward Proxy	Disabled	<input type="checkbox"/>
CA Certificate	ssl-certificate-ecs	<input checked="" type="checkbox"/>
CA Key	ssl-key-ecs	<input checked="" type="checkbox"/>

Figure 21 - CA certificate and key selection

The custom SSL client profile is generated on all three LTM in our reference deployment. Each LTM has a similar new custom SSL Client profile. This profile can now be associated with a virtual server.

**Step 3:** Create a virtual server for LTM-terminated SSL connectivity to ECS.

A second S3 virtual server is created in this example to demonstrate the scenario where an LTM offloads the SSL processing. Using this virtual server client application traffic will be encrypted between the clients and LTM and unencrypted between the LTM and ECS.

Similar to the first virtual server creation in the example above, Figure 22 below shows the General Properties section with values used.

General Properties	
Name	virtual-server-443-to-9020
Description	
Type	Standard
Source Address	
Destination Address/Mask	10.246.150.90
Service Port	443 HTTPS
Notify Status to Virtual Address	<input checked="" type="checkbox"/>
State	Enabled

Figure 22 - LTM virtual server general properties section

Only a single change is required in the configuration section, it is the addition of the SSL Profile (Client) value as shown in Figure 23 below.

	Selected	Available
SSL Profile (Client)	/Common client-ssl-profile-ecs	/Common clientssl clientssl-insecure-compatible clientssl-secure crypto-server-default-clientssl

Figure 23 - LTM virtual server with client SSL profile selection

The default pool and persistence profile are identical to those used during the creation of the virtual-server-80-to-9020 as shown in Figure 24 below.

Default Pool	pool-all-nodes-9020
Default Persistence Profile	source_addr

Figure 24 - LTM default pool and persistence profile selection

This virtual server can also be created on site 2's LTM. Site 1 configuration requires synchronization to the other LTM at this point.

Additional virtual servers can be created for S3 traffic. A SSL Server Profile can be created using ECS-generated key and certificate pair to allow encryption of traffic between the LTM and ECS nodes.

A common virtual server used is also port 9021-to-9021. With this, application traffic is encrypted between the client and ECS and the LTM performs no SSL processing, simply forwarding traffic to the ECS as-is. ECS nodes establish sessions directly with the clients. This requires the use of a pool with members that listen on port 9021.

Similar to the virtual servers created for S3 application traffic, virtual servers can be created for Atmos and Swift protocols using the appropriate ports.

### 5.1.3 Example: NFS via LTM

Using F5 BIG-IP LTM to provide HA during node failure is the only recommended deployment configuration with NFS workloads on ECS. It is not recommended to balance NFS traffic load across ECS nodes. This is because ECS nodes locally cache specific metadata attributes that NFS clients often query and read ahead (prefetch) NFS file data. These caching mechanisms allow fewer trips to disk which reduces system response time and generally improve sequential read throughput. Load balancing NFS traffic severely reduces the benefits of these ECS cache mechanisms.

A client application should be tied to a single ECS node for the duration of the session. Only during a failure should the connection between client and ECS be moved to another ECS node.

To configure an LTM to provide HA for NFS access to ECS, the basic configuration steps listed below should be performed. The steps can also be found with more detail at the following link: <https://support.f5.com/kb/en-us/solutions/public/6000/700/sol6749>. It is important to note the F5 link does not mention use of a health monitor during the creation of the NFS-specific pool. A pool configured without a monitor has no mechanism to trigger the LTM to direct clients to working nodes. One option for monitoring a pool serving NFS-related services is to create one or more custom monitors, up to one for each protocol and port combination required, and the requirement for one or more of them to be available on a node to receive traffic. This is the method chosen for use in our lab.

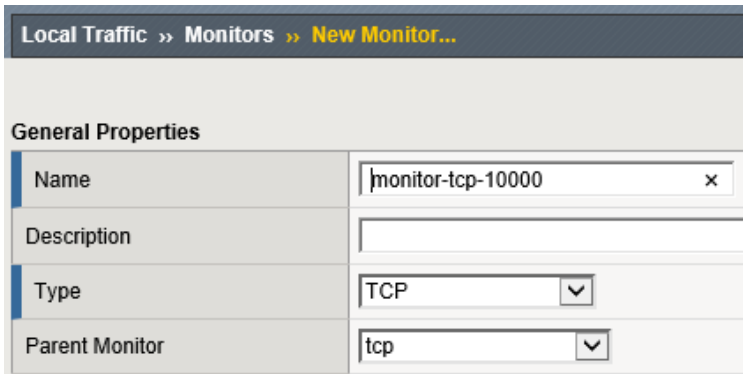
A virtual server configured for all service ports listens for any traffic bound for any port which is not configured elsewhere. In the examples provided in this paper we have created virtual servers configured for ports 80, 443, 9020, 9021. Optionally additional virtual servers could be created for the ports used with Atmos and Swift applications. This example demonstrates an NFS wild card (all ports) virtual server that listens on all other ports. Similarly the NFS wild card (all ports) pool will listen not just on the NFS-related ports but all ports as well. As mentioned above iRules can be put in place to further configure actions to take for traffic received on ports that either is or isn't explicitly serviced.

The overview of steps used in this example are as follows:

1. Create the NFS-related custom monitors.
2. Create the NFS server pool.
3. Create two NFS-related custom profiles.
4. Create the NFS virtual server.

**Step 1:** Create the NFS-related custom monitors.

For health monitoring six custom monitors are created, one for each pair of protocol and port. For example, one monitor is created for UDP traffic to port 10000. Another is created for TCP traffic for port 10000. All six monitors are associated with the pool and all are required to be available for a pool member to be considered for use by the virtual server. In production environments using this many monitors may be considered overkill as each monitor requires network communication between the LTM and ECS nodes. Figure 25 below shows the name and type selected during the creation of a custom monitor for traffic to port 10000 over TCP.

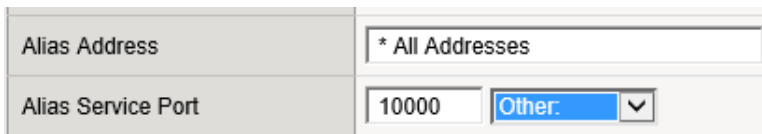


The screenshot shows the 'New Monitor...' configuration page in the LTM GUI. The breadcrumb trail is 'Local Traffic >> Monitors >> New Monitor...'. Under the 'General Properties' section, the following fields are visible:

Name	monitor-tcp-10000
Description	
Type	TCP
Parent Monitor	tcp

Figure 25 - LTM custom monitor creation general properties selection

Figure 26 below shows the alias address and service port settings used. This specific monitor, one of six, specifically checks for health of the Lock Manager service (port 10000) over TCP on the pool members.



The screenshot shows the 'Alias Address' and 'Alias Service Port' configuration fields. The 'Alias Address' is set to '\* All Addresses' and the 'Alias Service Port' is set to '10000' with the protocol dropdown set to 'Other:'.

Alias Address	* All Addresses
Alias Service Port	10000 Other:

Figure 26 - LTM custom monitor for Lock Manager server on port 10000

Figure 27 below shows all six custom NFS-related monitors created on the system. Two for each NFS-related service port (111, 2049, 10000). Each port has one monitor for traffic over TCP and one for traffic over UDP.

The screenshot shows the 'Local Traffic >> Monitors' configuration page. At the top, there is a 'Monitor List' tab with a settings icon. Below the tab is a search bar containing the text 'monitor', with 'Search' and 'Reset Search' buttons, and a 'Create...' button. The main area contains a table with the following columns: a checkbox, 'Name', 'Application', 'Type', and 'Partition / Path'. There are six rows of monitors listed:

<input checked="" type="checkbox"/>	▲ Name	↕ Application	▼ Type	↕ Partition / Path
<input type="checkbox"/>	monitor-tcp-10000		TCP	Common
<input type="checkbox"/>	monitor-tcp-111		TCP	Common
<input type="checkbox"/>	monitor-tcp-2049		TCP	Common
<input type="checkbox"/>	monitor-udp-10000		UDP	Common
<input type="checkbox"/>	monitor-udp-111		UDP	Common
<input type="checkbox"/>	monitor-udp-2049		UDP	Common

Figure 27 - LTM custom monitors

**Step 2:** Create the NFS server pool.

One pool is created for NFS traffic in our example. All ECS nodes are members of the pool and it is configured to listen on all ports. All six custom monitors are associated with the pool and the action on service down value is set to reject. Figure 28 below show the name, portion of the health monitors associated, availability requirement set to All, and action on service down set to reject.

Local Traffic » Pools : Pool List » **New Pool...**

Configuration: **Advanced** ▾

Name	pool-nfs												
Description													
Health Monitors	<table border="1"> <thead> <tr> <th>Active</th> <th>Available</th> </tr> </thead> <tbody> <tr> <td>/Common</td> <td>https_head_f5</td> </tr> <tr> <td>monitor-tcp-10000</td> <td>inband</td> </tr> <tr> <td>monitor-tcp-111</td> <td>tcp</td> </tr> <tr> <td>monitor-tcp-2049</td> <td>tcp_half_open</td> </tr> <tr> <td>monitor-udp-10000</td> <td>udp</td> </tr> </tbody> </table>	Active	Available	/Common	https_head_f5	monitor-tcp-10000	inband	monitor-tcp-111	tcp	monitor-tcp-2049	tcp_half_open	monitor-udp-10000	udp
Active	Available												
/Common	https_head_f5												
monitor-tcp-10000	inband												
monitor-tcp-111	tcp												
monitor-tcp-2049	tcp_half_open												
monitor-udp-10000	udp												
Availability Requirement	All ▾ Health Monitor(s)												
Allow SNAT	Yes ▾												
Allow NAT	Yes ▾												
Action On Service Down	Reject ▾												

Figure 28 - LTM new pool configuration dialog

Figure 29 below shows the Load Balancing Method (Least Connections) and member list for the pool.

**Resources**

Load Balancing Method	Least Connections (member) ▾
Priority Group Activation	Disabled ▾
New Members	<input type="radio"/> New Node <input type="radio"/> New FQDN Node <input checked="" type="radio"/> FQDN Node List
	Address: ecs-2-5 (ecs-2-5.kraft102.net) ▾
	Service Port: * * All Services ▾
	Auto Populate: Enabled ▾
	<b>Add</b>
	<pre>R:1 P:0 C:0 ecs-2-1 ecs-2-1.kraft102.net :*</pre> <pre>R:1 P:0 C:0 ecs-2-2 ecs-2-2.kraft102.net :*</pre> <pre>R:1 P:0 C:0 ecs-2-3 ecs-2-3.kraft102.net :*</pre> <pre>R:1 P:0 C:0 ecs-2-4 ecs-2-4.kraft102.net :*</pre> <pre>R:1 P:0 C:0 ecs-2-5 ecs-2-5.kraft102.net :*</pre>
<b>Edit</b> <b>Delete</b>	

Figure 29 - LTM pool load balancing and member selection

**Step 3:** Create two custom profiles.

A custom persistence profile is recommended for use by the NFS virtual server. Source Address Affinity as the Persistence Type is recommended along with a custom Timeout value of 86400 seconds to provide for a client to be persisted to the same NFS server for a 24 hour persistence period (adjust this value as appropriate for your environment). Figure 30 below shows the configuration used during the creation of the custom persistence profile.

Local Traffic » Profiles : Persistence » New Persistence Profile...	
<b>General Properties</b>	
Name	persistence-profile
Persistence Type	Source Address Affinity
Parent Profile	source_addr
<b>Configuration</b> <span style="float: right;">Custom <input type="checkbox"/></span>	
Match Across Services	<input type="checkbox"/>
Match Across Virtual Servers	<input type="checkbox"/>
Match Across Pools	<input type="checkbox"/>
Hash Algorithm	Default
Timeout	Specify... 86400 x seconds <input checked="" type="checkbox"/>
Prefix Length	None
Map Proxies	<input checked="" type="checkbox"/> Enabled
Override Connection Limit	<input type="checkbox"/>
Cancel Repeat Finished	

Figure 30 - Custom persistence profile creation

A custom FastL4 protocol profile with an Idle Timeout value of 3600 seconds is recommended to be used by the NFS virtual server. This allows for a connection to be idle for 1 hour before the BIG-IP system will close the connection (adjust this value as appropriate for your environment). Figure 31 below shows the configuration used during the creation of the custom protocol profile.

Local Traffic » Profiles : Protocol : Fast L4 » New Fast L4 Profile...

**General Properties**

Name	ofile-custom-fastl4-		
Parent Profile	fastL4	▼	

**Settings** Custom

Reset on Timeout	<input checked="" type="checkbox"/> Enabled	<input type="checkbox"/>	
Reassemble IP Fragments	<input type="checkbox"/>	<input type="checkbox"/>	
Idle Timeout	Specify... ▼	3600	seconds <input checked="" type="checkbox"/>

Figure 31 - Custom Fast L4 profile creation

**Step 4:** Create the NFS virtual servers.

A wildcard virtual server is configured as the NFS virtual server. Performance (Layer 4) as the Type. All ports are configured as the Service Port. Figure 32 below shows the General Properties configuration used for this example.

Local Traffic » Virtual Servers : Virtual Server List » New Virtual Server

**General Properties**

Name	virtual-server-nfs		
Description			
Type	Performance (Layer 4)	▼	
Source Address			
Destination Address/Mask	10.246.150.90		
Service Port	*	* All Ports	▼
Notify Status to Virtual Address	<input checked="" type="checkbox"/>		
State	Enabled ▼		

Figure 32 - LTM NFS virtual server general properties configuration

Within the Configuration section \*All Protocols as the Protocol and the newly-created custom FastL4 profile you created earlier as the Protocol Profile (Client). Figure 33 below shows the Configuration sections of the NFS-specific virtual server created.



Configuration: <b>Basic</b> ▼	
Protocol	* All Protocols ▼
Protocol Profile (Client)	protocol-profile-custom-fastl4-nfs ▼
HTTP Profile	None ▼
HTTP Proxy Connect Profile	None ▼
Traffic Acceleration Profile	None ▼
VLAN and Tunnel Traffic	All VLANs and Tunnels ▼
Source Address Translation	None ▼

Figure 33 - LTM NFS basic configuration selections

Within the Resources section select the NFS server pool created earlier as the Default Pool and the Persistence profile created earlier as the Default Persistence Profile. Figure 34 below shows the Resources section of the Resources section where the pool for NFS is set as the default pool and the default persistence profile is configured.

Resources	
iRules	<div style="display: flex; justify-content: space-between;"> <div style="border: 1px solid gray; width: 40%; height: 40px;"></div> <div style="border: 1px solid gray; padding: 5px;"> <p>Available</p> <p>/Common</p> <p>_sys_APM_ExchangeSupport_OA_BasicAuth</p> <p>_sys_APM_ExchangeSupport_OA_NtlmAuth</p> <p>_sys_APM_ExchangeSupport_helper</p> <p>_sys_APM_ExchangeSupport_main</p> </div> </div> <div style="margin-top: 5px;"> <span>&lt;&lt;</span> <span>&gt;&gt;</span> </div> <div style="margin-top: 5px;"> <span>Up</span> <span>Down</span> </div>
Default Pool	pool-nfs ▼
Default Persistence Profile	persistence-profile-custom-source_addr ▼
Fallback Persistence Profile	source_addr ▼

Cancel Repeat Finished

Figure 34 - LTM NFS virtual server default pool and persistence profile selection

Regardless of the LTM configuration an ECS NFS export can be mounted directly via any ECS node at either site. This is the beauty of using file-enabled buckets within a federated namespace - a truly global NFS mount point.

Once the NFS monitors, pools, and virtual servers are configured on the LTM at both sites in our example deployment, we can also mount the ECS NFS exports via the LTM virtual server IP addresses. Listed below

are a few commands we ran on our client. If the exports are mounted at both sites via the LTM and direct to ECS, performing a listing across the directories will show identical contents as expected.

Below are a few of the **mount** commands used on the client. The NFS export configured on ECS can be mounted using any of the ECS node IP or virtual server IP addresses or their associated DNS names. Listing the directory contents of all four mounts will reveal the same contents.

```
sudo mount -o"vers=3" 192.168.101.11:/webappl/s3_webappl ~/f5-101-nfs-via-ecs/  
sudo mount -o"vers=3" ecs-2-3.kraft102.net:/webappl/s3_webappl ~/f5-102-nfs-via-  
ecs/  
sudo mount -o"vers=3" f5-ltm-s1-virtual.ecstme.org:/webappl/s3_webappl ~/f5-101-  
nfs-via-ltm/  
sudo mount -o"vers=3" 10.246.150.91:/webappl/s3_webappl ~/f5-102-nfs-via-ltm/
```

Below is the output from the **showmount** command. This command shows mount information for an NFS server.

Output of all commands are identical as expected. A single NFS export is created in our federated namespace. That export is available on any node in either of the sites. The export can be mounted by clients using either an ECS node IP address or via the LTM using one of the virtual server IP addresses.

```
ckraft@kraft-mint18 ~ $ showmount -e ecs-1-1.kraft101.net  
Export list for ecs-1-1.kraft101.net:  
/webappl/s3_webappl *  
ckraft@kraft-mint18 ~ $ showmount -e ecs-2-1.kraft102.net  
Export list for ecs-2-1.kraft102.net:  
/webappl/s3_webappl *  
ckraft@kraft-mint18 ~ $ showmount -e 10.246.150.90  
Export list for 10.246.150.90:  
/webappl/s3_webappl *  
ckraft@kraft-mint18 ~ $ showmount -e 10.246.150.91  
Export list for 10.246.150.91:  
/webappl/s3_webappl *
```

Below is the output from the *rpcinfo* command. This command is used to report Remote Procedure Call (RPC) information.

Output of both commands are identical as expected. The first query used the IP address of an ECS node and the second query used the IP address of one of the LTM virtual servers.

```
ckraft@kraft-mint18 ~ $ rpcinfo -p 192.168.101.11
```

program	vers	proto	port	service
100000	4	tcp	111	portmapper
100000	3	tcp	111	portmapper
100000	2	tcp	111	portmapper
100000	4	udp	111	portmapper
100000	3	udp	111	portmapper
100000	2	udp	111	portmapper
100005	3	tcp	2049	mountd
100005	3	udp	2049	mountd
100003	3	tcp	2049	nfs
100024	1	tcp	2049	status
100021	4	tcp	10000	nlockmgr
100021	4	udp	10000	nlockmgr

```
ckraft@kraft-mint18 ~ $ rpcinfo -p 10.246.150.91
```

program	vers	proto	port	service
100000	4	tcp	111	portmapper
100000	3	tcp	111	portmapper
100000	2	tcp	111	portmapper
100000	4	udp	111	portmapper
100000	3	udp	111	portmapper
100000	2	udp	111	portmapper
100005	3	tcp	2049	mountd
100005	3	udp	2049	mountd
100003	3	tcp	2049	nfs
100024	1	tcp	2049	status
100021	4	tcp	10000	nlockmgr
100021	4	udp	10000	nlockmgr

#### 5.1.4 Example: Geo-affinity via iRule on LTM

Some organizations may find it useful to configure their LTM with access to both local and remote ECS nodes. Applications generally experience the best performance when communicating with a local ECS. For applications that can tolerate additional latencies that come with remote site communication however, ECS can provide increased storage efficiency using XOR.

To take advantage of the storage efficiencies gained on ECS by XOR data must be written evenly across three or more sites. While writing data evenly across multiple sites leads to increased storage efficiency, reading data in a similar fashion may lead to increased WAN overhead and storage inefficiencies due to caching of remote data. This is because in order for ECS to provide data that is spread out across multiple sites in a strongly consistent manner, it maintains a record of each object's owner. The object owner is a VDC which serves as the definitive source and ultimate authority for changes to an object. When an object is

read from a non-owner site, ECS must communicate with the owner site across the WAN to determine the latest version of the object. By using the LTM to direct applications to the site where an object was originally written, WAN traffic can be minimized and caching of ECS objects at non-owning sites eliminated or dramatically minimized. This results in higher performance for application workflow and minimal caching of remote data.

Globally balancing writes across ECS sites in a basic round robin fashion will lead to the highest XOR efficiency. However, applying the same basic round robin algorithm to reads would mean requests would most often be sent to a different site to where an object was written. This is where a geo-affinity algorithm is beneficial.

The geo-affinity algorithm provided in this example was validated using the S3 API. It results in the following behavior:

- Equivalent amount of data written to all sites. This leads to lower data protection overhead with XOR.
- Objects are always read from site where it was originally written. This leads to lower WAN traffic, higher performance, and no caching needed for remote data.

As shown in figures below, the rule performs a hash of the object's path as seen in the URL to determine which pool to send the request to.

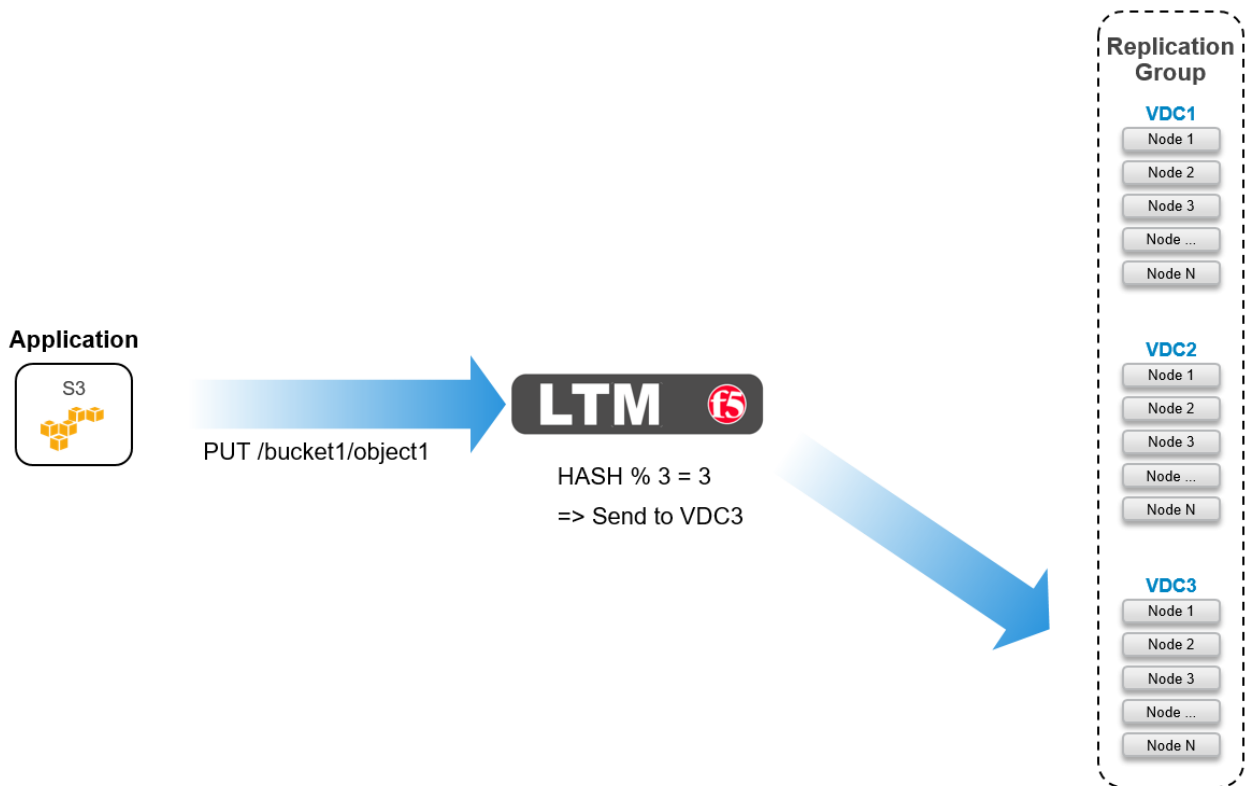


Figure 35 - Geo-affinity on LTM for PUT

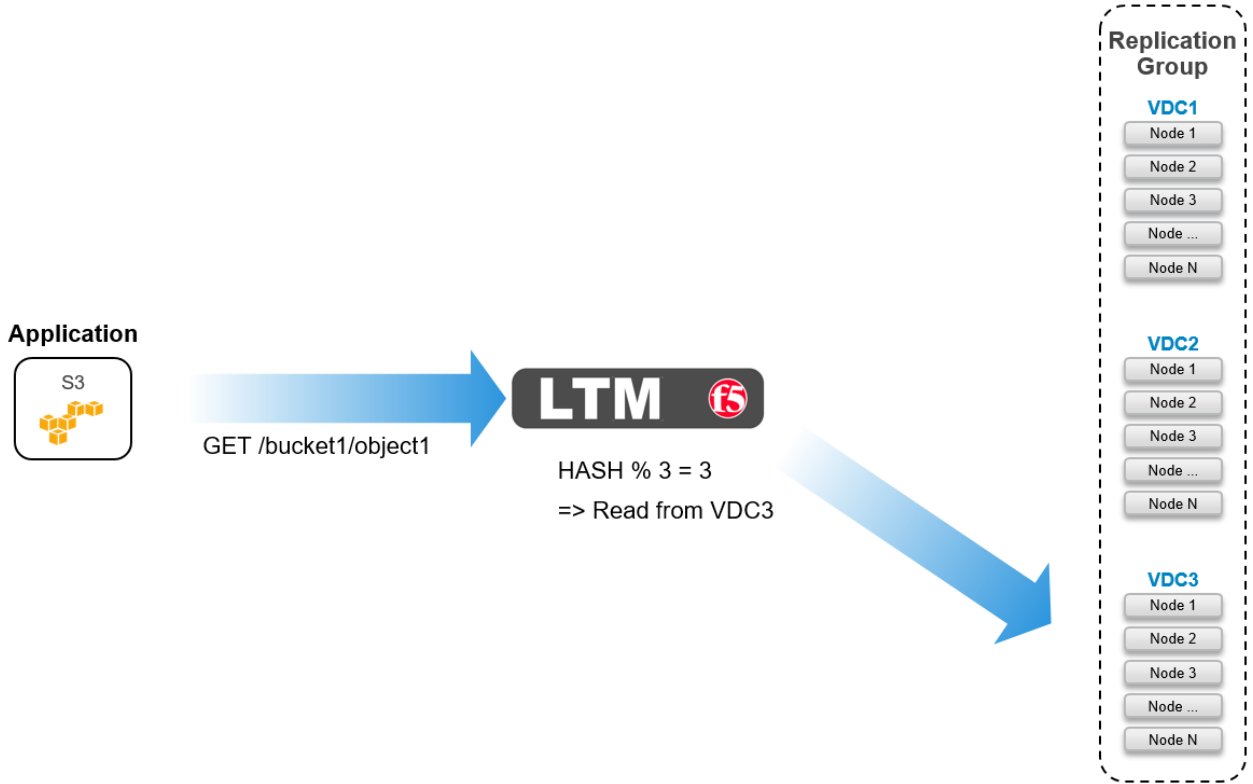


Figure 36 - Geo-affinity on LTM for GET

Below is a code sample which shows an iRule configured on an LTM with three pools, one local and two remote to the LTM.

```
when HTTP_REQUEST {
    if { [HTTP::path] ne {} and [HTTP::path] starts_with {} }{
        if { [getfield [HTTP::host] {:} 1] ends_with "<my base url>" or
[getfield [HTTP::host] {:} 1] ends_with ".s3.amazonaws.com" }{
            set path [string range [HTTP::path] 1 end]
        }
        else {
            set pathList [split [HTTP::path] {/}]
            if { [llength $pathList] > 2 } {
                set path [string range [join [lreplace $pathList 1 1] {/}] 1
end]
            }
        }
    }
    if { [info exists path] and $path ne {} } {
        binary scan [sha1 $path] H6 hash
        switch -- [expr 0x$hash % 3 ] {
            0 { pool /Common/pl-stor-ecs-vdc1-9020 }
            1 { pool /Common/pl-stor-ecs-vdc2-9020 }
            2 { pool /Common/pl-stor-ecs-vdc3-9020 }
        }
    }
}
```

## 5.2 F5 BIG-IP DNS

F5's BIG-IP DNS can provide all of an organization's DNS-related services. This paper focuses on how BIG-IP DNS can provide Global Server Load Balancing (GSLB). GSLB enables geographically distributed applications to scale and perform efficiently. It also allows applications to use a single endpoint regardless of their location. Figure 37 below shows a deployment example where BIG-IP DNS's delegation mode is used. In the deployment example, two BIG-IP DNS, one at each of two sites, are configured to provide authoritative DNS answers for a subdomain named *f5-ecs*. This is a subdomain of the primary domain named *ecstme.org* which is used in a lab for this example.

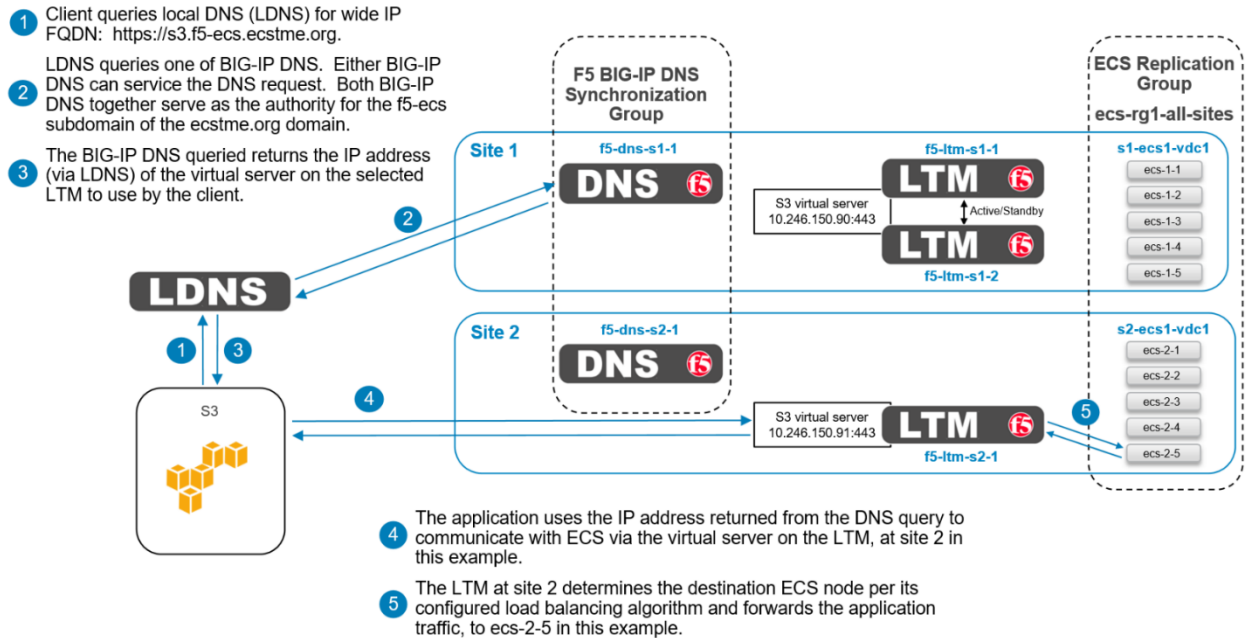


Figure 37 - BIG-IP DNS high-level delegation mode overview

Introducing a BIG-IP DNS system in to an existing DNS architecture, as we are in this example, can be minimally disruptive because it requires little interaction with the existing authoritative architecture. A local DNS (LDNS) server is solely responsible for the domain ecstme.org and an associated /24 IP address space. The LDNS server is configured to delegate authority for a sub-domain named *f5-ecs.ecstme.org*. This is done by creating an NS record in the *ecstme.org* domain.

Each of the two BIG-IP DNS systems in this example requires a DNS A record in the primary domain. The server's A records are associated with the NS record for the delegated namespace, *f5-ecs.ecstme.org*. With this setup, any DNS query inside the *f5-ecs.ecstme.org* sub-domain is forwarded to one of the two BIG-IP DNS systems for resolution.

Table 14 below show the required DNS entries for this example.

Table 14

DNS record entry	Record type	Record data	Comments
f5-dns-s1-1.ecstme.org	A	10.246.150.83	BIG-IP DNS self IP address Site 1
f5-dns-s2-1.ecstme.org	A	10.246.150.84	BIG-IP DNS self IP address Site 2
f5-ecs.ecstme.org	NS	f5-dns-s1-1.ecstme.org	BIG-IP DNS name server record Site 1
f5-ecs.ecstme.org	NS	f5-dns-s2-1.ecstme.org	BIG-IP DNS name server record Site 2

CNAME records can optionally be created to redirect DNS requests to the BIG-IP DNS for resolution. Our example deployment does not use CNAMEs. Some organizations prohibit the use of CNAME records as they can complicate DNS administration in large scale environments. Also the use of CNAME records for heavily used names cause an extra redirect and lookup in the DNS resolution path which can result in undesirable latency.

A main benefit for using a BIG-IP DNS system is that it allows one namespace to be used by clients regardless of the number of end sites. For example, if three VDCs exist at geographically separate locations, and each VDC is served by one or more BIG-IP LTM, all users regardless of their location can use a single name, for example, <https://s3.f5-ecs.ecstme.org>. The BIG-IP DNS directs clients to the recommended, and available, ECS node by returning the IP address of the virtual server on the LTM the client should use.

In HA scenarios with BIG-IP DNS devices, only the Active/Standby type is available. That is if two BIG-IP DNS devices are to be deployed at a site in a redundant pair, they must be in the Active/Standby type of configuration. Globally however multiple BIG-IP DNS systems can all actively receive DNS requests from local DNS systems by participating in a synchronization group. This is how the two BIG-IP DNS systems are deployed in this example. One BIG-IP DNS system is deployed at each site. Both BIG-IP DNS systems are in a common synchronization group. The primary domain, [ecstme.org](https://ecstme.org), upon receiving a query for an address inside the [f5-ecs](https://f5-ecs) subdomain, will forward that query to one of the two BIG-IP DNS systems. Each of the BIG-IP DNS systems serve as authority for the [f5-ecs](https://f5-ecs) subdomain. Members of the synchronization group communicate with each other to ensure they're all in agreement with the latest information.

The first step in BIG-IP DNS configuration involves creating data centers. Two data centers are created on each BIG-IP DNS instance in the deployment example, one for each site. Figure 38 below shows the data centers as created on one of the BIG-IP DNS systems.

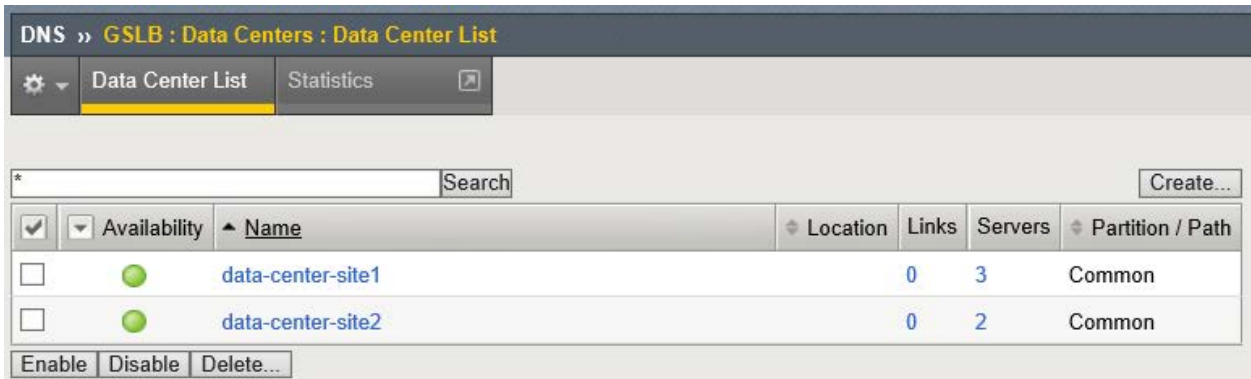


Figure 38 - BIG-IP DNS configured data centers

Creation of data centers is simple and only requires naming of the instance. Optionally a prober preference can be set. In the deployment example, the prober preference was set to 'Inside Data Center' for the local-to-the-BIG-IP-DNS data center and 'Outside Data Center' for the data center created for the other site.

Data centers are populated with server objects. Server objects represent each of the BIG-IP systems on the network. It is possible to create server objects for third party systems as well. BIG-IP DNS server objects are created for each of the BIG-IP DNS and LTM systems in the deployment example. Self IP addresses of the BIG-IP systems are used during creation, not the management IP addresses. More than one IP address can be used depending upon how the server interacts with the rest of the network.

Figure 39 below show the configuration used for the BIG-IP DNS at site 2. Along with naming the instance, it is assigned to the appropriate data center and assigned the *bigip* health monitor.



DNS » GSLB : Servers : Server List » Servers : f5-dns-s2-1

Properties
  Devices
  Virtual Servers
  Links
  Statistics

### General Properties

Name	f5-dns-s2-1
Partition / Path	/Common
Product	BIG-IP System
Data Center	data-center-site2
Prober Preference	Inherit From Data Center
Prober Fallback	Inherit From Data Center
State	Enabled

Configuration: Basic

	Selected	Available
Health Monitors	/Common bigip	/Common gateway_icmp gtp http http_head_f5

Figure 39 - BIG-IP DNS server creation

Initially only the BIG-IP DNS system where the data center and server objects were created will show as available in the servers list in the WebUI. Access to the command line interface of this system is required for the next step where the *big3d\_install* script is run. Running this script upgrades all BIG-IP system's *big3d* agents and instructs the other systems to authenticate through the exchange of SSL certificates.

Example of the command run in our lab environment:

```
root@(f5-dns-s1-1)(cfg-sync In Sync)(Standby)(/Common)(tmsh)# run gtm
big3d_install 10.246.150.84 10.246.150.86 10.246.150.87 10.246.150.89
```

At this point in the configuration all BIG-IP systems can communicate with each other. The BIG-IP DNS systems can now use the other BIG-IP systems when load balancing DNS queries, and can acquire statistics and status information for the virtual servers these systems manage.

DNS » GSLB : Servers : Server List									
Server List		Trusted Server Certificates			Statistics				
* <input type="text"/> Search <input type="button" value="Create..."/>									
<input type="checkbox"/>	Status	Name	Devices	Address	Data Center	Virtual Servers	Product	Partition / Path	
<input type="checkbox"/>	<span style="color: green;">●</span>	server-dns-s1-1	1	10.246.150.83	data-center-ecstme	0	BIG-IP System	Common	
<input type="checkbox"/>	<span style="color: green;">●</span>	server-dns-s2-1	1	10.246.150.84	data-center-ecstme	0	BIG-IP System	Common	
<input type="checkbox"/>	<span style="color: green;">●</span>	server-ltm-s1-1	1	10.246.150.86	data-center-ecstme	1	BIG-IP System	Common	
<input type="checkbox"/>	<span style="color: green;">●</span>	server-ltm-s1-2	1	10.246.150.87	data-center-ecstme	1	BIG-IP System	Common	
<input type="checkbox"/>	<span style="color: green;">●</span>	server-ltm-s2-1	1	10.246.150.89	data-center-ecstme	0	BIG-IP System	Common	
<input type="button" value="Enable"/> <input type="button" value="Disable"/> <input type="button" value="Delete..."/>									

Figure 40 - BIG-IP DNS server list

For the DNS-related configuration synchronization between BIG-IP DNS devices to occur, a synchronization group must be created and enabled. This is done on one of the BIG-IP DNS systems. Next, the other BIG-IP DNS system is configured to join the synchronization group. This is completed using the *gtm\_add* script. The script is ran using a command line interface on the second BIG-IP DNS.

All BIG-IP DNS systems in the same BIG-IP DNS synchronization group have the same rank, exchange heartbeat messages, and share probing responsibility.

Below is output of the *gtm\_add* script run in the deployment environment:

```
[root@f5-dns-s2-1:Standby:In Sync] config # tmsh
root@(f5-dns-s2-1)(cfg-sync In Sync)(Standby)(/Common)(tmos)# run gtm gtm_add
10.246.150.83
WARNING: Running this script will wipe out the current configuration
files (bigip_gtm.conf, named.conf and named zone files) on the BIG-IP GTM
Controller on which this script is run. The configuration will be
replaced with the configuration of the remote BIG-IP GTM Controller
in the specified sync group
The local BIG-IP GTM MUST already be added in the configuration of the
other GTM.

NOTE: The current master key of this BIG-IP will be changed to match the
master key of the remote BIG-IP GTM.
The BIG-IP config will be saved via:
    tmsh save sys config
after the master key is changed.
Are you absolutely sure you want to do this? [y/n] y

==> Running 'bigstart shutdown gtmd' on the local system
==> Running 'bigstart shutdown zrd' on the local system
==> Running 'bigstart shutdown named' on the local system
    Retrieving remote and installing local BIG-IP's SSL certs ...
Enter root password if prompted
The authenticity of host '10.246.150.83 (10.246.150.83)' can't be established.
RSA key fingerprint is 5d:9b:7c:2a:0d:da:4e:17:64:16:1e:1a:32:56:e7:f5.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.246.150.83' (RSA) to the list of known hosts.
Password:
Rekeying Master Key...
Saving running configuration...
    /config/bigip.conf
    /config/bigip_base.conf
    /config/bigip_user.conf
Saving Ethernet mapping...done
Verifying iQuery connection to 10.246.150.83. This may take up to 30 seconds

Retrieving remote GTM configuration...

Retrieving remote DNS/named configuration...

Restarting gtmd
Restarting named
Restarting zrd

==> Done <==
```

## 5.2.1 Example: Roaming Client Geo Environment

BIG-IP DNS can be configured to direct traffic on the basis of the source of the DNS query. This allows applications to use common, global FQDNs, and have their traffic directed to the closest or best-performing destination resource.

Figure 41 shows an application physically located near Site 1. BIG-IP DNS can use the source address of the LDNS server that is resolving the FQDN on behalf of the client to determine which site is most appropriate for the application to be directed to.

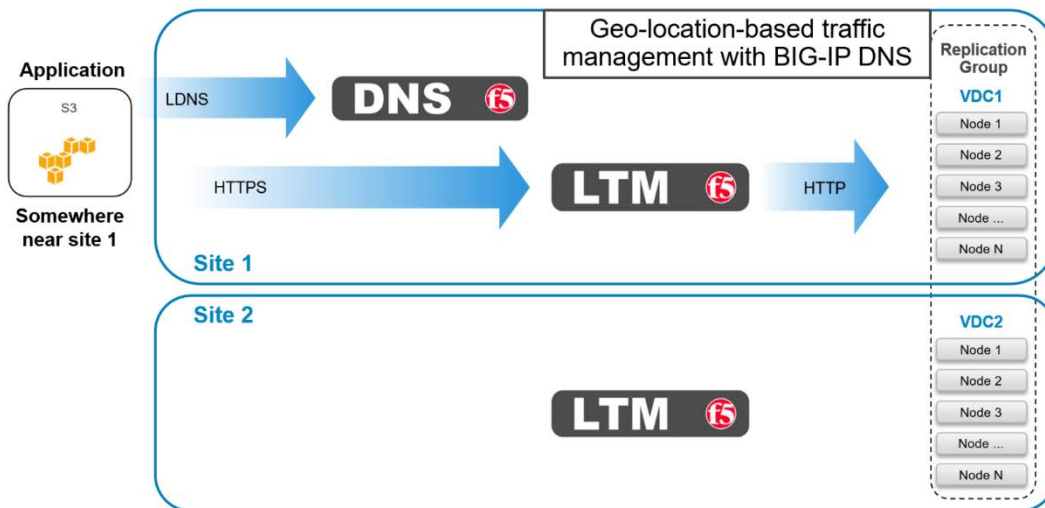


Figure 41 - BIG-IP DNS roaming client near Site 1

Figure 42 shows an application physically near Site 2. BIG-IP DNS will resolve the DNS query to a virtual server located on the LTM at Site 2. This should provide the application with the path of the lowest latency.

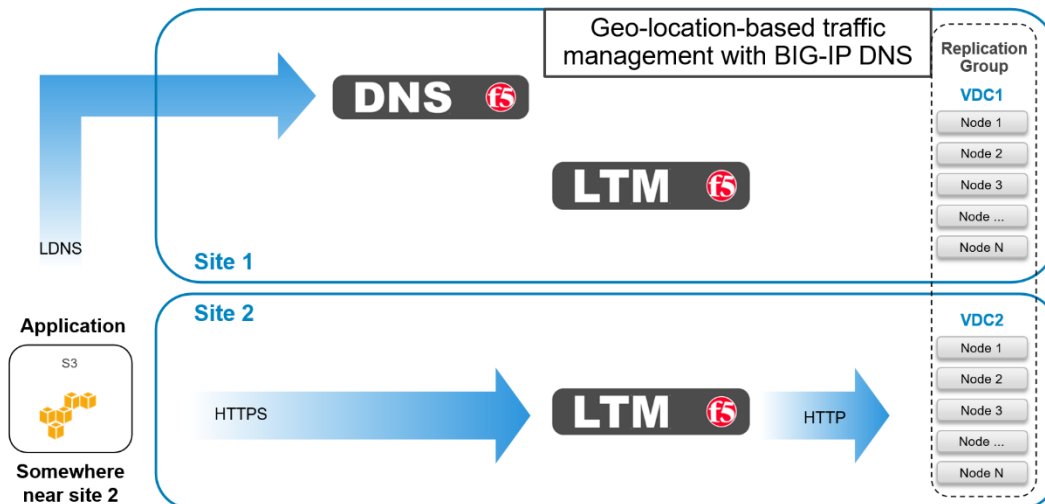


Figure 42 - BIG-IP DNS roaming client near Site 2

## 5.2.2 Example: Application Traffic Management for XOR Efficiency Gains

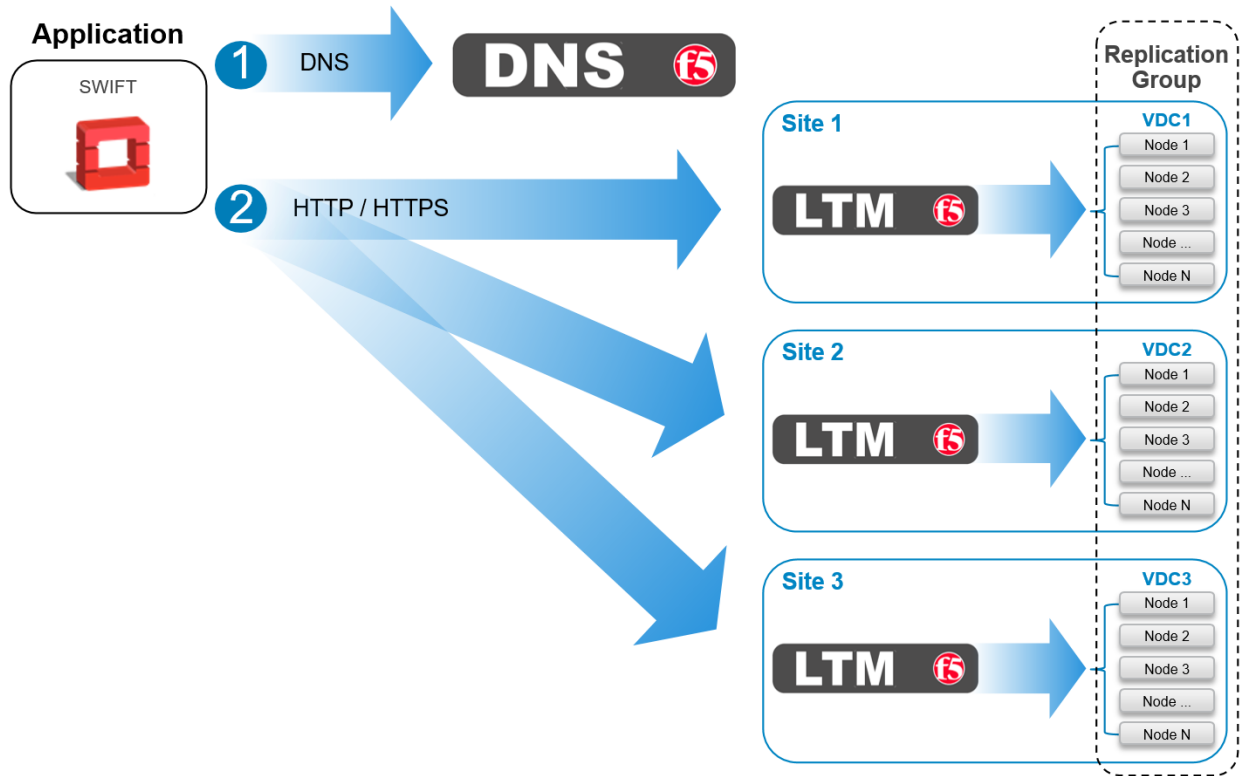


Figure 43 - XOR efficiency gains using BIG-IP DNS

Data must be written across all sites in order to take advantage of the storage efficiencies gained on ECS by XOR. For workflows that are able to tolerate latencies associated with this traffic pattern, BIG-IP DNS can be configured to balance write traffic equally between sites in a round robin fashion.

## 6 Best Practices

Utilizing a load balancer with ECS is highly recommended. Highlights of some the best practices when deploying with ECS include:

- Do not use an LTM for CAS traffic. The Centera SDK has a built-in load balancer and cannot function without direct access to all nodes.
- Traffic management is best utilized for data traffic.
- Terminate SSL connections on LTM when possible to reduce the load on the ECS nodes.
- If SSL termination is required on ECS nodes itself, then use Layer 4 (TCP) to pass through the SSL traffic to ECS nodes for handling. The certificates would need to be installed on the ECS nodes and not on the LTM.
- Use two or more LTMs at each site. Using a single LTM at a site creates a single point of failure. In multisite federated environments where BIG-IP DNS devices provide site redundancy, a single LTM may be sufficient so long as applications can tolerate increased latency during a site failure. If application traffic cannot be directed to alternate sites during failure, a single LTM deployment can result in complete data unavailability.
- For NFS, use only the high available functionality of the load balancer. That is, for NFS workloads configure LTM to keep client sessions terminated on a single ECS node. Only during node failure should an NFS session be torn down and established on another ECS node. Balancing NFS traffic across ECS nodes, even during read only operations, is inefficient because it wouldn't take advantage of the ECS caching mechanism.
- When deploying three or more ECS sites, and when performance considerations allow, employ a global load balancing mechanism to distribute load across sites to take advantage of ECS XOR storage efficiency. Also important to optimize the local object read hit rate in a global deployment.
- Use source-address based session persistence. For sites serving a large amount of traffic from a small number of end-points, this may result in an uneven distribution of load. If this is an issue, restrict session persistence to only "PUT" and "POST" operations, if possible. Session persistence should only be disabled as a last resort.
- Service providers and others accepting connections from the public Internet should use the "TCP WAN Optimized" profile. This enables TCP window scaling and other RFC 1323 throughput-improving options, adjusts various buffer sizes upward, and sets other defaults that are appropriate for WAN traffic. Service providers should also consider raising TCP send and receive buffer sizes and proxy buffer high and low water marks even further to at least 128KB or 256KB. This allows a better throughput from end users on links with high bandwidth delay factors (for example, high bandwidth and high latency).

## 7

### Conclusion

The F5 BIG-IP DNS and LTM products can be useful in enhancing ECS high availability and performance with traffic management at the local and global level. Managing application traffic to ECS intelligently minimizes impacts due to node or site failures. In addition balancing traffic load across ECS nodes allows for efficient use of the ECS software and underlying physical resources.

## 8

# References

### ECS Product Documentation

- ECS product documentation at support site or the community links:  
[https://support.emc.com/products/37254\\_ECS-Appliance-/Documentation/](https://support.emc.com/products/37254_ECS-Appliance-/Documentation/)  
<https://community.emc.com/docs/DOC-53956>
- ECS Architecture and Overview  
<http://www.emc.com/collateral/white-papers/h14071-ecs-architectural-guide-wp.pdf>
- ECS Networking and Best Practices  
<http://www.emc.com/collateral/white-paper/h15718-ecs-networking-bp-wp.pdf>

### F5 Product Documentation

- F5 Virtual Edition and Supported Hypervisors Matrix  
[https://support.f5.com/kb/en-us/products/big-ip\\_ltm/manuals/product/ve-supported-hypervisor-matrix.html](https://support.f5.com/kb/en-us/products/big-ip_ltm/manuals/product/ve-supported-hypervisor-matrix.html)
- Link to F5 BIG-IP Product Manuals and Release Notes  
<https://support.f5.com/csp/tech-documents>
- F5 Local Traffic Manager and Global Traffic Managers Operations Guide  
[https://support.f5.com/kb/en-us/products/big-ip\\_ltm/manuals/product/f5-ltm-gtm-operations-guide-1-0.html](https://support.f5.com/kb/en-us/products/big-ip_ltm/manuals/product/f5-ltm-gtm-operations-guide-1-0.html)
- BIG-IP System: Essentials  
[https://support.f5.com/kb/en-us/products/big-ip\\_ltm/manuals/product/bigip-system-essentials-12-1-1.html](https://support.f5.com/kb/en-us/products/big-ip_ltm/manuals/product/bigip-system-essentials-12-1-1.html)
- BIG-IP Local Traffic Management: Basics  
[https://support.f5.com/kb/en-us/products/big-ip\\_ltm/manuals/product/ltm-basics-12-1-0.html](https://support.f5.com/kb/en-us/products/big-ip_ltm/manuals/product/ltm-basics-12-1-0.html)
- BIG-IP System: SSL Administration  
[https://support.f5.com/kb/en-us/products/big-ip\\_ltm/manuals/product/bigip-system-ssl-administration-12-1-1.html](https://support.f5.com/kb/en-us/products/big-ip_ltm/manuals/product/bigip-system-ssl-administration-12-1-1.html)
- BIG-IP DNS: Implementations  
<https://support.f5.com/kb/en-us/products/big-ip-dns/manuals/product/bigip-dns-implementations-13-0-0.html>
- BIG-IP DNS Services: Implementations  
[https://support.f5.com/kb/en-us/products/big-ip\\_ltm/manuals/product/bigip-dns-services-implementations-13-0-0.html](https://support.f5.com/kb/en-us/products/big-ip_ltm/manuals/product/bigip-dns-services-implementations-13-0-0.html)
- BIG-IP DNS: Load Balancing  
<https://support.f5.com/kb/en-us/products/big-ip-dns/manuals/product/bigip-dns-load-balancing-12-1-0.html>
- BIG-IP Local Traffic Manager and DNS Operations Guide  
[https://support.f5.com/kb/en-us/products/big-ip\\_ltm/manuals/product/f5-ltm-dns-operations-guide.html](https://support.f5.com/kb/en-us/products/big-ip_ltm/manuals/product/f5-ltm-dns-operations-guide.html)



# A Creating a Custom S3 Monitor

Use of the S3 Ping operation is recommended in monitoring ECS S3 service port availability on ECS software running on dedicated ECS hardware. This operation is documented inside the Dell EMC ECS REST API Reference Guide which can be found at <https://www.emc.com/techpubs/api/ecs/v3-0-0-0/index.htm>.

The S3 Ping operation is dependent upon the fabric layer inside the ECS software. The fabric layer of the ECS software stack provides clustering and system health among other things. It is responsible for keeping required services up and running and managing resources such as disks, containers, and the network. It tracks and reacts to environmental changes such as failure detection and provides alerts related to system health. The S3 Ping operation uses the fabric layer to determine the state of the node's maintenance mode. For additional information about the ECS fabric layer refer to the ECS Overview and Architecture whitepaper at <http://www.emc.com/collateral/white-papers/h14071-ecs-architectural-guide-wp.pdf>.

The ECS Community Edition is a virtual instance of the ECS software and as such has no underlying fabric layer. It is because of this that the example LTM S3 pool configurations provided in the main section of this paper use standard HTTP and HTTPS built-in monitors.

Figure 44 below shows the general properties section of a custom S3 monitor.

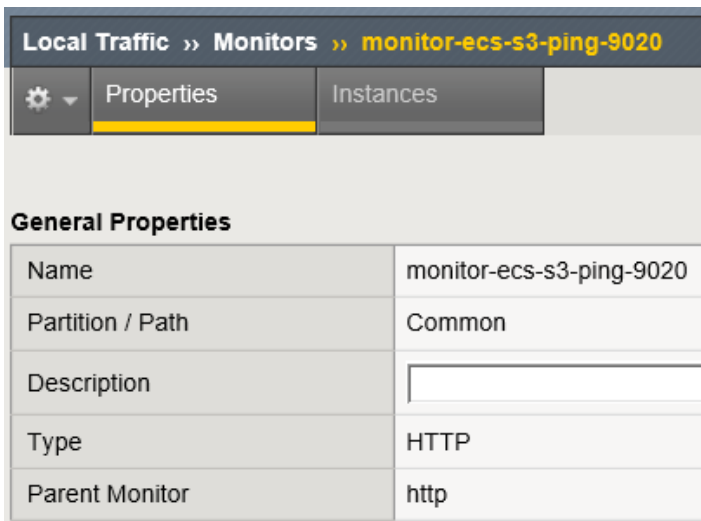


Figure 44 - Custom S3 monitor general properties section

Figure 45 below shows the strings portion of the custom S3 monitor.

**Note:** Regarding the send string configured in the configuration section of the new monitor, 'GET /?ping HTTP/1.1\r\nHost:f5\r\n\r\n', the value of 'f5' that is associated with 'Host' is a dummy value. This dummy value is needed and is used for the GET all.

Configuration: Basic <input type="button" value="v"/>	
Interval	<input type="text" value="5"/> seconds
Timeout	<input type="text" value="16"/> seconds
Send String	<pre>GET /?ping HTTP/1.1\r\nHost:f5\r\n\r\n</pre>
Receive String	<pre>&lt;Name&gt;MAINTENANCE_MODE&lt;/Name&gt;&lt;Status&gt;OFF&lt;/Status&gt;</pre>
Receive Disable String	<pre>&lt;Name&gt;MAINTENANCE_MODE&lt;/Name&gt;&lt;Status&gt;ON&lt;/Status&gt;</pre>

Figure 45 - Strings fields of custom S3 monitor

Figure 46 below shows the final part of the configuration section for the custom S3 monitor.

User Name	<input type="text" value="admin"/>
Password	<input type="password" value="••••••••"/>
Reverse	<input type="radio"/> Yes <input checked="" type="radio"/> No
Transparent	<input type="radio"/> Yes <input checked="" type="radio"/> No
Alias Address	<input type="text" value="* All Addresses"/>
Alias Service Port	<input type="text" value="9020"/>
Adaptive	<input type="checkbox"/> Enabled
<input type="button" value="Update"/> <input type="button" value="Delete"/>	

Figure 46 - Basic configuration section (continued) of custom S3 monitor

Figure 47 below shows the custom S3 monitor created above in use as the health monitor for a pool of ECS nodes.

Local Traffic » Pools : Pool List » **New Pool...**

Configuration: Basic ▾

Name	pool-u300-9020	
Description		
Health Monitors	Active	Available
	<input type="checkbox"/> /Common <input checked="" type="checkbox"/> monitor-ecs-s3-ping-9020	<input type="checkbox"/> inband <input type="checkbox"/> monitor-ecs-s3-ping-9021 <input type="checkbox"/> monitor-tcp-10000 <input type="checkbox"/> monitor-tcp-111 <input type="checkbox"/> monitor-tcp-2049

**Resources**

Load Balancing Method	Least Connections (member) ▾
Priority Group Activation	Disabled ▾
New Members	<input checked="" type="radio"/> New Node <input type="radio"/> New FQDN Node <input type="radio"/> FQDN Node List
	Node Name: <input type="text"/> (Optional)
	Address: <input type="text" value="10.246.150.178"/>
	Service Port: <input type="text" value="9020"/> <input type="text" value="Select..."/> ▾
	<input type="button" value="Add"/>
	<input type="text" value="R:1 P:0 C:0 10.246.150.175 10.246.150.175 :9020"/> <input type="text" value="R:1 P:0 C:0 10.246.150.176 10.246.150.176 :9020"/> <input type="text" value="R:1 P:0 C:0 10.246.150.177 10.246.150.177 :9020"/> <input type="text" value="R:1 P:0 C:0 10.246.150.178 10.246.150.178 :9020"/>
<input type="button" value="Edit"/> <input type="button" value="Delete"/>	

Figure 47 - LTM pool using custom S3 monitor

## B BIG-IP DNS and LTM CLI Configuration Snippets

Provided for reference, the following are sample configuration from the primary relevant BIG-IP configuration files.

### B.1 BIG-IP DNS

List of primary configuration files:

```
[root@f5-dns-s1-1:Active:Standalone] config # ls -la bigip_base.conf bigip.conf
bigip_gtm.conf
-rw-r-----. 1 root root 7954 2017-07-08 00:46 bigip_base.conf
-rw-r-----. 1 root root 3832 2017-07-08 00:46 bigip.conf
-rw-r-----. 1 root root 7883 2017-07-08 00:47 bigip_gtm.conf
```

#### B.1.1 bigip.conf

##### Listener

```
ltm virtual /Common/listener-tcp-ipv4 {
    destination /Common/10.246.150.83:53
    ip-protocol tcp
    mask 255.255.255.255
    profiles {
        /Common/dns { }
        /Common/tcp { }
    }
    source 0.0.0.0/0
    translate-address disabled
    translate-port disabled
}
```

#### B.1.2 bigip\_base.conf

##### Self IP

```
net self /Common/selfip1.1local {
    address 10.246.150.83/24
    allow-service all
    traffic-group /Common/traffic-group-local-only
    vlan /Common/vlanexternal
}
```

## VLAN

```
net vlan /Common/vlanexternal {
  interfaces {
    1.1 { }
  }
  tag 4094
}
```

## B.1.3 bigip\_gtm.conf

### Data Center

```
gtm datacenter /Common/data-center-site1 { }
gtm datacenter /Common/data-center-site2 {
  prober-preference outside-datacenter
}
```

### Server, BIG-IP DNS

```
gtm server /Common/f5-dns-s1-1 {
  datacenter /Common/data-center-site1
  devices {
    f5-dns-s1-1 {
      addresses {
        10.246.150.83 { }
      }
    }
  }
  link-discovery enabled-no-delete
  monitor /Common/bigip
  product bigip
  virtual-server-discovery enabled-no-delete
  virtual-servers {
    /Common/listener-tcp-ipv4 {
      destination 10.246.150.85:53
    }
    /Common/listener-udp-ipv4 {
      destination 10.246.150.85:53
    }
    /Common/listener-udp-ipv6 {
      destination fe80::250:56ff:fe8f:3948.53
    }
  }
}
```

## Server, BIG-IP LTM

```
gtm server /Common/f5-ltm-s2-1 {
  datacenter /Common/data-center-site2
  devices {
    f5-ltm-s2-1 {
      addresses {
        10.246.150.89 { }
      }
    }
  }
  link-discovery enabled-no-delete
  monitor /Common/bigip
  product bigip
  virtual-server-discovery enabled-no-delete
  virtual-servers {
    /Common/virtual-server-80-to-9020 {
      destination 10.246.150.91:80
    }
    /Common/virtual-server-443-to-9020 {
      destination 10.246.150.91:443
    }
    /Common/virtual-server-9021-to-9021 {
      destination 10.246.150.91:9021
    }
    /Common/virtual-server-nfs {
      destination 10.246.150.91:0
    }
  }
}
```

## Pool

```
gtm pool a /Common/pool-nfs {
  load-balancing-mode topology
  members {
    /Common/f5-ltm-s1-1:/Common/virtual-server-nfs {
      member-order 0
    }
    /Common/f5-ltm-s1-2:/Common/virtual-server-nfs {
      member-order 1
    }
    /Common/f5-ltm-s2-1:/Common/virtual-server-nfs {
      member-order 2
    }
  }
}
```

## WIP

```
gtm wideip a /Common/nfs.f5-ecs.ecstme.org {
  persistence enabled
  pool-lb-mode topology
  pools {
    /Common/pool-nfs {
      order 0
    }
  }
}
gtm wideip a /Common/s3-9021.f5-ecs.ecstme.org {
  aliases {
    *.s3-9021.f5-ecs.ecstme.org
  }
  persistence enabled
  pool-lb-mode topology
  pools {
    /Common/pool-s3-port9021 {
      order 0
    }
  }
}
gtm wideip a /Common/s3-http.f5-ecs.ecstme.org {
  aliases {
    *.s3-http.f5-ecs.ecstme.org
  }
  persistence enabled
  pool-lb-mode topology
  pools {
    /Common/pool-s3-port80 {
      order 0
    }
  }
}
gtm wideip a /Common/s3.f5-ecs.ecstme.org {
  aliases {
    *.s3.f5-ecs.ecstme.org
  }
  persistence enabled
  pool-lb-mode topology
  pools {
    /Common/pool-s3-port443 {
      order 0
    }
  }
}
```

## B.2 BIG-IP LTM

List of primary configuration files:

```
[root@f5-ltm-s1-1:Active:In Sync] config # ls -la bigip_base.conf bigip.conf
-rw-r-----. 1 root root 11345 2017-07-13 13:38 bigip_base.conf
-rw-r-----. 1 root root 12740 2017-07-13 13:38 bigip.conf
```

### B.2.1 bigip.conf

#### Node

```
ltm node /Common/ecs-1-1 {
    fqdn {
        autopopulate enabled
        name ecs-1-1.kraft101.net
    }
}
```

#### Virtual Server

```
ltm virtual /Common/virtual-server-80-to-9020 {
    destination /Common/10.246.150.90:80
    ip-protocol tcp
    mask 255.255.255.255
    persist {
        /Common/source_addr {
            default yes
        }
    }
    pool /Common/pool-all-nodes-9020
    profiles {
        /Common/tcp { }
    }
    source 0.0.0.0/0
    translate-address enabled
    translate-port enabled
}
```



## Pool

```
ltm pool /Common/pool-all-nodes-9020 {
  load-balancing-mode least-connections-node
  members {
    /Common/ecs-1-1:9020 {
      fqdn {
        autopopulate enabled
        name ecs-1-1.kraft101.net
      }
    }
    /Common/ecs-1-2:9020 {
      fqdn {
        autopopulate enabled
        name ecs-1-2.kraft101.net
      }
    }
    /Common/ecs-1-3:9020 {
      fqdn {
        autopopulate enabled
        name ecs-1-3.kraft101.net
      }
    }
    /Common/ecs-1-4:9020 {
      fqdn {
        autopopulate enabled
        name ecs-1-4.kraft101.net
      }
    }
    /Common/ecs-1-5:9020 {
      fqdn {
        autopopulate enabled
        name ecs-1-5.kraft101.net
      }
    }
  }
  monitor /Common/tcp
  service-down-action reset
}
```

## Monitor

```
ltm monitor http /Common/monitor-ecs-s3-ping-9020 {
  adaptive disabled
  defaults-from /Common/http
  destination *:9020
  interval 5
  ip-dscp 0
  password $M$lI$FWlBV3+GAmVjOk/qdhnFNw==
  recv <Name>MAINTENANCE_MODE</Name><Status>OFF</Status>
  recv-disable <Name>MAINTENANCE_MODE</Name><Status>ON</Status>
  send "GET /?ping HTTP/1.1\r\nHost:f5\r\n\r\n"
  time-until-up 0
  timeout 16
  username admin
}
```

## Profile, client-ssl

```
ltm profile client-ssl /Common/client-ssl-profile-ecs {
  app-service none
  cert /Common/default.crt
  cert-key-chain {
    default {
      cert /Common/default.crt
      key /Common/default.key
    }
  }
  chain none
  defaults-from /Common/clientssl
  inherit-certkeychain true
  key /Common/default.key
  passphrase none
  proxy-ca-cert /Common/ssl-certificate-ecs.crt
  proxy-ca-key /Common/ssl-key-ecs.key
}
```

## Profile, fastl4

```
ltm profile fastl4 /Common/protocol-profile-custom-fastl4-nfs {
  app-service none
  defaults-from /Common/fastL4
  idle-timeout 3600
}
```

## Routes

```
net route /Common/default {
    gw 10.246.150.1
    mtu 1500
    network default
}
net route /Common/route102net {
    gw 10.246.150.89
    mtu 1500
    network 192.168.102.0/24
}
```

## B.2.2 bigip\_base.conf

### Self IP

```
net self /Common/selfip1.2local {
    address 10.246.150.86/24
    allow-service {
        default
    }
    traffic-group /Common/traffic-group-local-only
    vlan /Common/vlanexternal
}
net self /Common/selfip1.1floating {
    address 192.168.101.5/24
    allow-service {
        default
    }
    traffic-group /Common/traffic-group-1
    vlan /Common/vlaninternal
}
```

### VLAN

```
net vlan /Common/vlanha {
    interfaces {
        1.3 { }
    }
    tag 4092
}
```

## Devices

```
cm device /Common/f5-ltm-s1-1.ecstme.org {
    active-modules { "BIG-IP, VE Trial|EZHDWMD-WHJVHIY|Rate Shaping|External
Interface and Network HSM, VE|SDN Services, VE|SSL, Forward Proxy, VE|Max
Compression, VE|BIG-IP VE, Multicast Routing|SSL, VE|DNS (1K QPS), VE|Routing
Bundle, VE|AFM, VE|ASM, VE|Crypto Offload, VE, Tier 1 (25M - 200M)|DNSSEC|Anti-
Virus Checks|Base Endpoint Security Checks|Firewall Checks|Network Access|Secure
Virtual Keyboard|APM, Web Application|Machine Certificate Checks|Protected
Workspace|Remote Desktop|App Tunnel|CGN, BIG-IP VE, AFM ONLY|PSM, VE" }
    base-mac 00:50:56:8f:79:26
    build 2.0.1671
    cert /Common/dtdi.crt
    chassis-id 420fdac5-59ea-53b5-763185f6e91c
    configsync-ip 192.168.255.4
    edition "Hotfix HF2"
    hostname f5-ltm-s1-1.ecstme.org
    key /Common/dtdi.key
    management-ip 172.16.3.2
    marketing-name "BIG-IP Virtual Edition"
    mirror-ip 192.168.255.4
    mirror-secondary-ip 10.246.150.86
    platform-id Z100
    product BIG-IP
    self-device true
    time-zone America/Toronto
    unicast-address {
        {
            effective-ip management-ip
            effective-port 1026
            ip management-ip
        }
        {
            effective-ip 192.168.255.4
            effective-port 1026
            ip 192.168.255.4
        }
    }
    version 13.0.0
}
```

```

cm device /Common/f5-ltm-s1-2.ecstme.org {
    active-modules { "BIG-IP, VE Trial|VEWOCES-VNHDLKP|Rate Shaping|External
Interface and Network HSM, VE|SDN Services, VE|SSL, Forward Proxy, VE|Max
Compression, VE|BIG-IP VE, Multicast Routing|SSL, VE|DNS (1K QPS), VE|Routing
Bundle, VE|AFM, VE|ASM, VE|Crypto Offload, VE, Tier 1 (25M - 200M)|DNSSEC|Anti-
Virus Checks|Base Endpoint Security Checks|Firewall Checks|Network Access|Secure
Virtual Keyboard|APM, Web Application|Machine Certificate Checks|Protected
Workspace|Remote Desktop|App Tunnel|CGN, BIG-IP VE, AFM ONLY|PSM, VE" }
    base-mac 00:50:56:8f:57:85
    build 2.0.1671
    chassis-id 420f0d1e-817c-ad57-214b40151423
    configsync-ip 192.168.255.5
    edition "Hotfix HF2"
    hostname f5-ltm-s1-2.ecstme.org
    management-ip 172.16.3.3
    marketing-name "BIG-IP Virtual Edition"
    mirror-ip 192.168.255.5
    mirror-secondary-ip 10.246.150.87
    platform-id Z100
    product BIG-IP
    time-zone America/Toronto
    unicast-address {
        {
            effective-ip management-ip
            effective-port 1026
            ip management-ip
        }
        {
            effective-ip 192.168.255.5
            effective-port 1026
            ip 192.168.255.5
        }
    }
    version 13.0.0
}

```

### Device group

```

cm device-group /Common/device-group-a {
    devices {
        /Common/f5-ltm-s1-1.ecstme.org { }
        /Common/f5-ltm-s1-2.ecstme.org { }
    }
    type sync-failover
}

```

## Device trust

```
cm device-group /Common/device_trust_group {
    auto-sync enabled
    devices {
        /Common/f5-ltm-s1-1.ecstme.org { }
        /Common/f5-ltm-s1-2.ecstme.org { }
    }
    hidden true
    network-failover disabled
}
```

## Mirroring

```
sys state-mirroring {
    addr 192.168.255.4
    secondary-addr 10.246.150.86
}
```