DEPLOYMENT GUIDE

A10 THUNDER ADC FOR EPIC SYSTEMS

(For A10 ACOS 2.7.x)

OVERVIEW

This document shows how A10 Thunder® ADC can be deployed with Epic Electronic Medical Record system. The tested solution is based on Thunder ADC device load balancing multiple Epic Web servers.

The deployment guide provides a detailed configuration guide on how to administer the Thunder ADC with Epic Systems. Since the Epic Systems has over 25 products that can be integrated, this deployment guide has selected some applications to be load balanced. However, the Epic Systems platform runs on the same web server architecture, hence the configuration across all products will be similar.

TABLE OF CONTENTS

DISCLAIMER

This document does not create any express or implied warranty about A10 Networks or about its products or services, including but not limited to fitness for a particular use and noninfringement. A10 Networks has made reasonable efforts to verify that the information contained herein is accurate, but A10 Networks assumes no responsibility for its use. All information is provided "as-is." The product specifications and features described in this publication are based on the latest information available; however, specifications are subject to change without notice, and certain features may not be available upon initial product release. Contact A10 Networks for current information regarding its products or services. A10 Networks' products and services are subject to A10 Networks' standard terms and conditions.

INTRODUCTION

Epic Systems is the leading Electronic Medical Record (EMR) system that offers the most intuitive, fast and user-friendly applications. The Epic Systems has expanded its features and capabilities from emergency department application (ASAP), scheduling application (Cadence), anesthesia information management system (Anesthesia) and more using a single interface to manage all systems via Epic Hyperspace. The Epic Systems is based on a client/server application and medical record information is hosted on a hierarchical database called Multi-user Multi-Programming System (MUMPS) database which provides a faster data insertion and retrieval. The Epic Systems utilizes a web server application and the A10 Thunder® ADC is used as an Application Delivery Controller (ADC) to provide advanced load balancing features including application optimization and acceleration such as RAM caching, TCP connection-reuse, health checks and more.

The Epic Systems creates a single medical record for each patient across all care settings which interconnect medical platforms and departments via Epic Hyperspace. The Hyperspace is a browser based application which enables medical staff to access the medical records securely.

DEPLOYMENT GUIDE PREREQUISITES

The deployment guide was tested based on the following:

- Thunder ADC requirements
  - The A10 Networks Thunder ADC must be running version 2.7.x or higher
- Epic Systems Requirements
  - Epic Version 2012IU2

ACCESSING THE THUNDER ADC

This section describes how to access the Thunder. The Thunder ADC can be accessed either from a Command Line Interface (CLI) or Graphical User Interface (GUI):

- CLI – Text-based interface in which you type commands on a command line. You can access the CLI directly through the serial console or over the network using either of the following protocols:
  - Secure protocol – Secure Shell (SSH) version 2
  - Unsecure protocol – Telnet (if enabled)
- GUI – Web-based interface in which you click to access configuration or management pages and type or select values to configure or manage the device. You can access the GUI using the following protocol:
  - Secure protocol – Hypertext Transfer Protocol over Secure Socket Layer (HTTPS)

Note*: HTTP requests are redirected to HTTPS by default on the Thunder device.*

- Default Username: “admin”
- Default password is “a10”.
- Default IP Address of the device is “172.31.31.31”

For detailed information on how to access the Thunder ADC, refer to document “*System Configuration and Administration Guide.*”

ARCHITECTURE OVERVIEW

The Epic Systems is composed of multiple backend components that run on application servers and database servers. The diagram below provides a high level network layout of how Epic components are interconnected across Epic Systems.

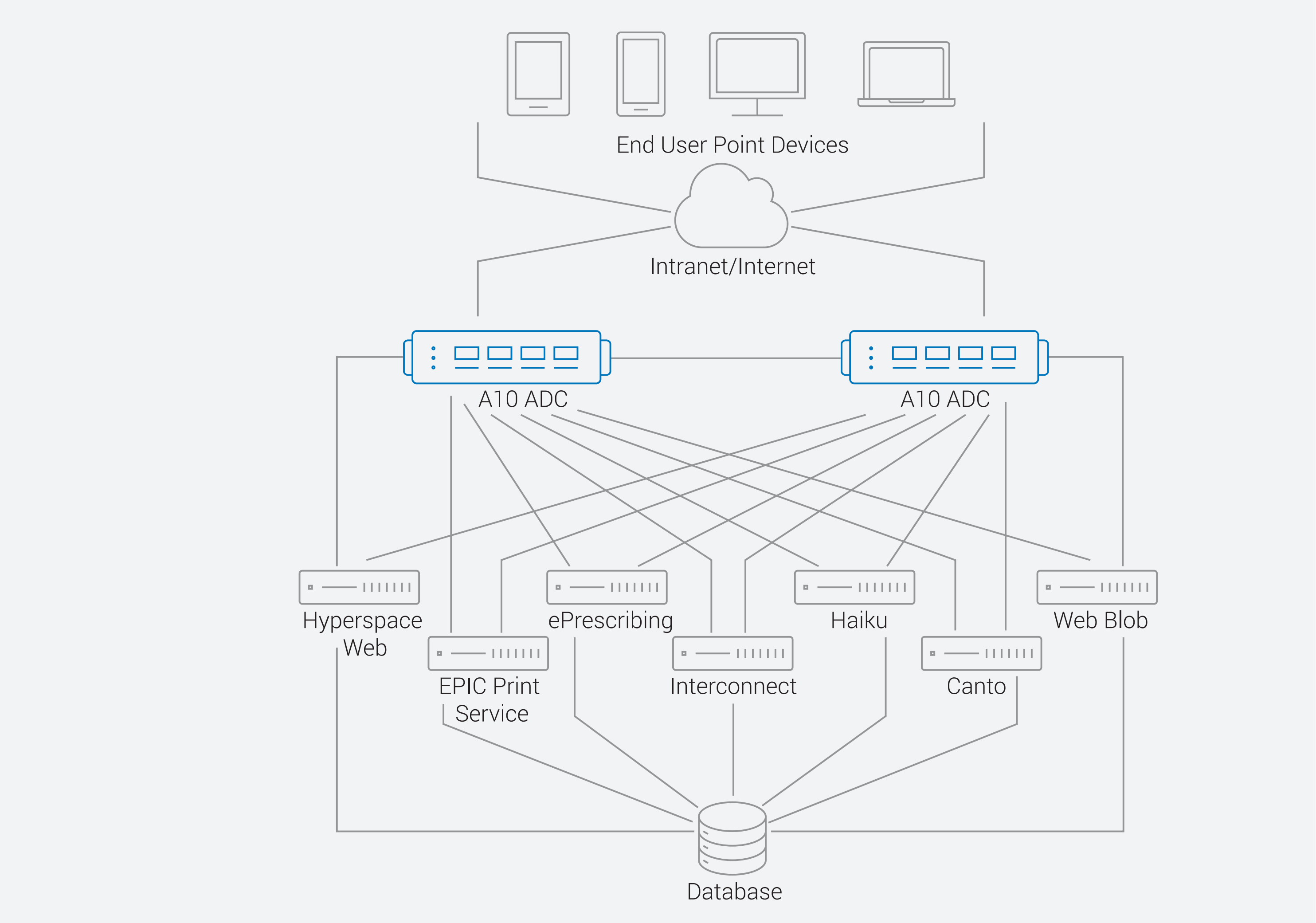


Figure 1: Epic Systems network topology

CONFIGURATION WORK FLOW

The configuration work flow will be as follows:

1. Define SLB configuration consisting of:
    - Real Servers
    - Health Monitors
    - Service Groups
    - Virtual Servers

2. Create and apply the following feature templates to SLB VIP:
   - SSL Offload
   - HTTP/HTTPS Compression
   - Cookie Persistence
   - TCP Connection Reuse
   - TCP Proxy
   - RAM Caching
   - IP Source NAT

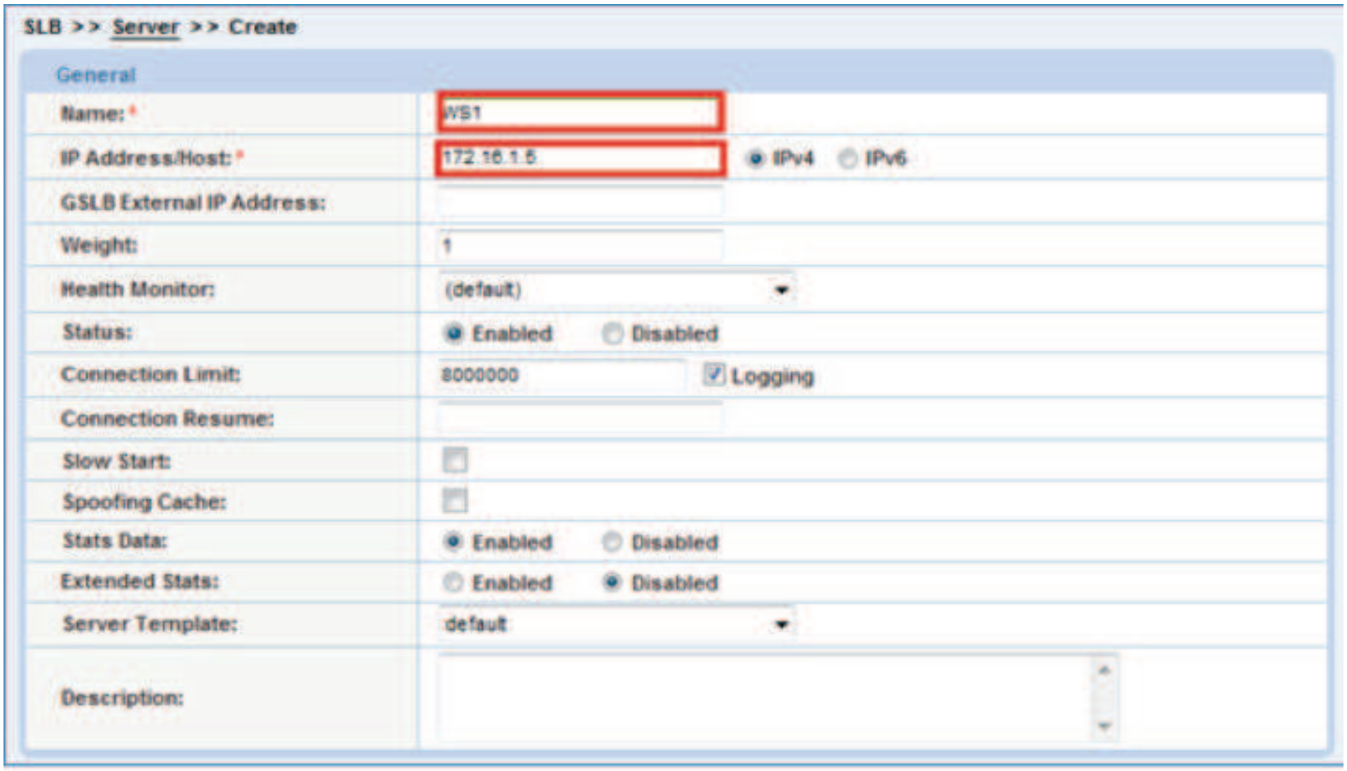
SLB CONFIGURATION

REAL SERVERS

This section demonstrates how to configure the Epic web servers in the Thunder ADC.

1. Navigate to **Config Mode > SLB > Service > Server.**
2. Click “**Add**” to add a new server.
3. Within the Server section, enter the following required information.
   - Name: “**WS1**”
   - IP address /Host: “**172.16.1.5**”

Note: Enter additional servers if necessary.



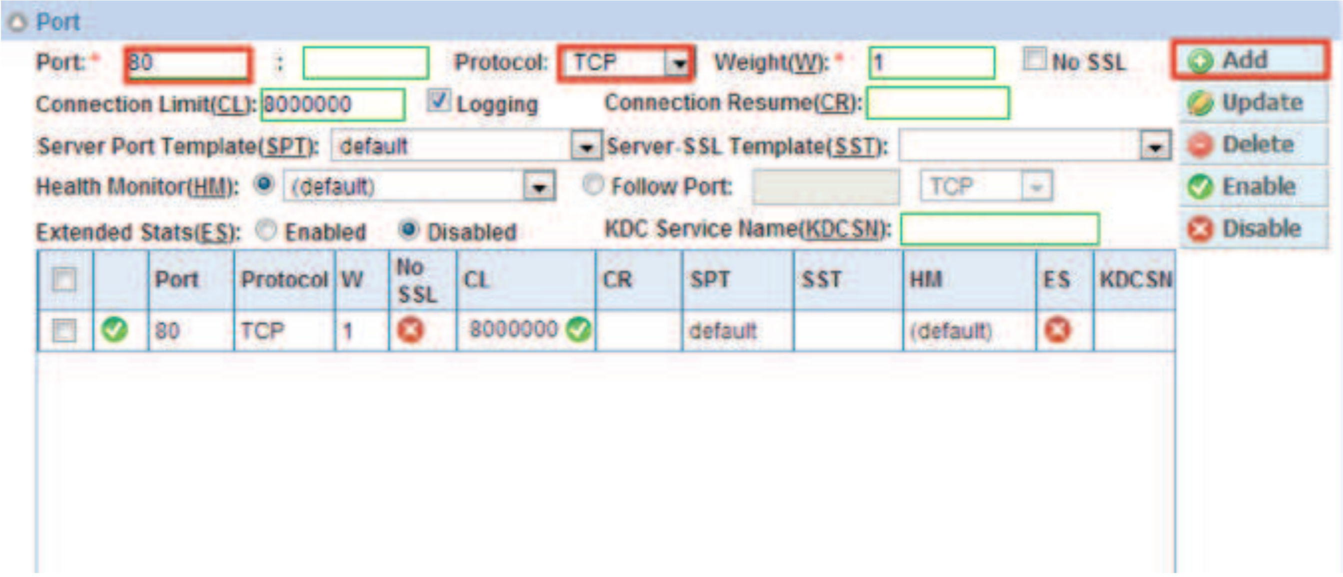
SLB >> Server >> Create

General

Name: *	WS1
IP Address/Host: *	172.16.1.5 ● IPv4 ○ IPv6
GSLB External IP Address:	
Weight:	1
Health Monitor:	(default)
Status:	● Enabled ○ Disabled
Connection Limit:	8000000 ☑ Logging
Connection Resume:	
Slow Start:	□
Spoofing Cache:	□
Stats Data:	● Enabled ○ Disabled
Extended Stats:	○ Enabled ● Disabled
Server Template:	default
Description:	

Figure 2: Real server configuration

4. To add ports to the server configuration, navigate to Config Mode > SLB > Service > Server > Port Section.
5. Enter Port, Protocol type and then click “Add”.
6. Click “OK” and “Save” configuration.



Port: * 80 : Protocol: TCP Weight(W): * 1 □ No SSL

Connection Limit(CL): 8000000 ☑ Logging Connection Resume(CR):

Server Port Template(SPT): default Server SSL Template(SST):

Health Monitor(HM): ● (default) ○ Follow Port: TCP

Extended Stats(ES): ○ Enabled ● Disabled KDC Service Name(KDCSN):

□		Port	Protocol	W	No SSL	CL	CR	SPT	SST	HM	ES	KDCSN
□	✓	80	TCP	1	✗	8000000 ✓		default		(default)	✗	

Figure 3: Real server port configuration

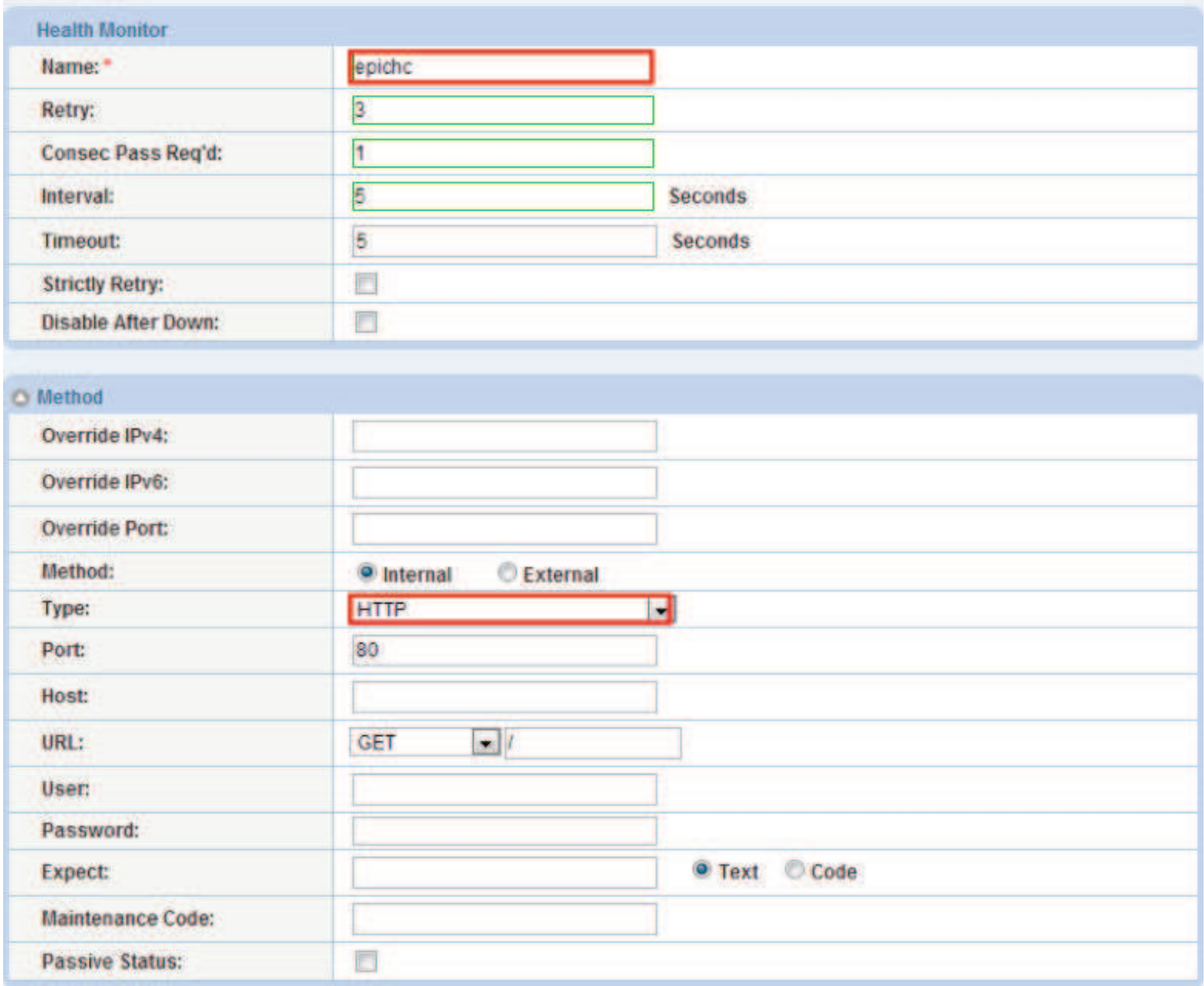
HEALTH MONITORS

The Thunder ADC can automatically initiate the health status checks of real servers and service ports. This provides clients assurance that all requests go to functional and available servers. If a server or a port does not respond appropriately to a health check, the server will be temporarily removed from the list of available servers. Once the server is restored and starts responding appropriately to the health checks, the server will be automatically added back to the list of available servers.

1. Navigate to Config Mode > SLB > Health Monitor > Health Monitor.
2. Health Monitor: Click the drop-down menu and select Create.
3. Enter the Health Monitor Name, “epichc”.
4. Under Method type, select “HTTP”.

   Note*: By default, Thunder ADC expects response code 200 (OK) with “HTTP” method. Please update “URL” or “Expect” section according to your environment.*

5. Click OK and then continue with the Service Group configuration.



Health Monitor

Name: *	epichc
Retry:	3
Consec Pass Req'd:	1
Interval:	5 Seconds
Timeout:	5 Seconds
Strictly Retry:	□
Disable After Down:	□

Method

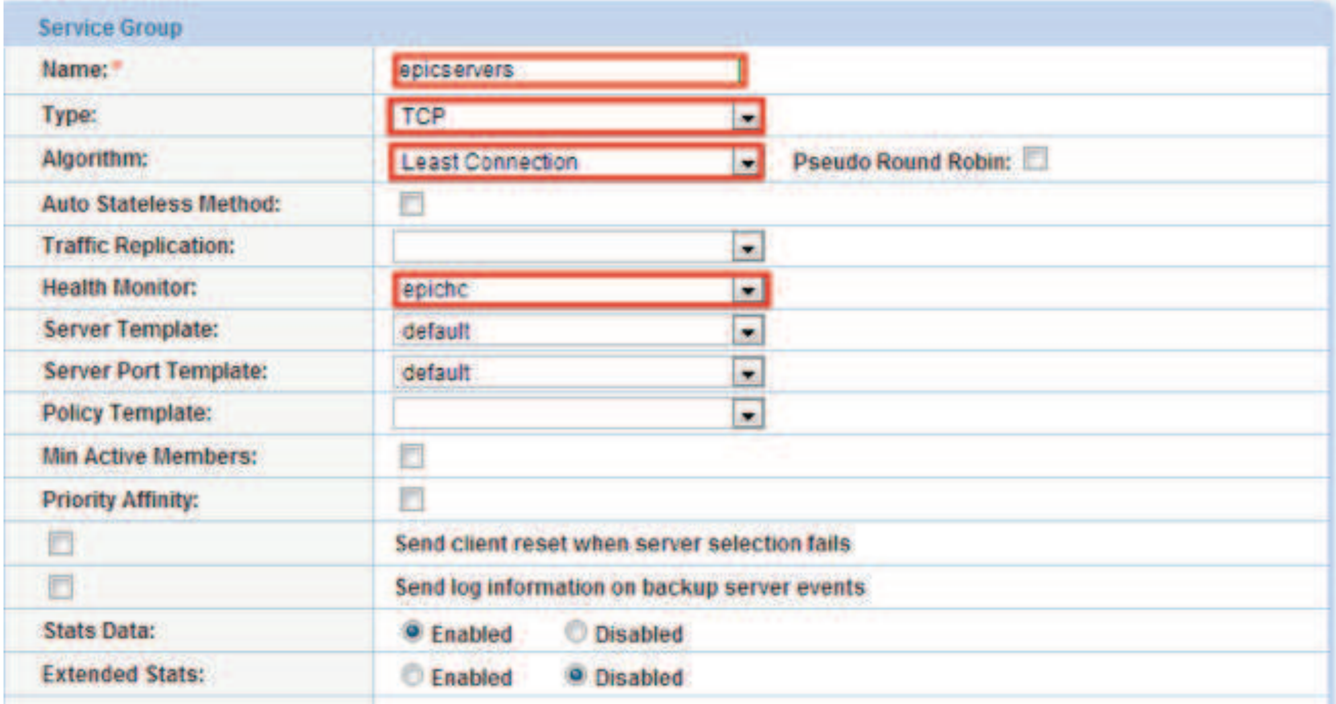
Override IPv4:	
Override IPv6:	
Override Port:	
Method:	● Internal ○ External
Type:	HTTP
Port:	80
Host:	
URL:	GET /
User:	
Password:	
Expect:	● Text ○ Code
Maintenance Code:	
Passive Status:	□

Figure 4: Health monitor configuration

SERVICE GROUP

This section demonstrates how to configure the Epic web servers in a service group. A service group contains a set of real servers from which the Thunder ADC can select to service client requests. A service group supports multiple Epic real servers as one logical server.

1. Navigate to **Config Mode > SLB > Service > Service Group.**
2. Click "**Add**" to add a new service group.
3. Within the Server Group section, enter the following required information:
   - Name: "**epicservers**"
   - Type: Select "**TCP**" from the drop-down menu.
   - Algorithm: "**LeastConnection**" from the drop-down menu.
   - Health Monitor: Select "**epichc**"



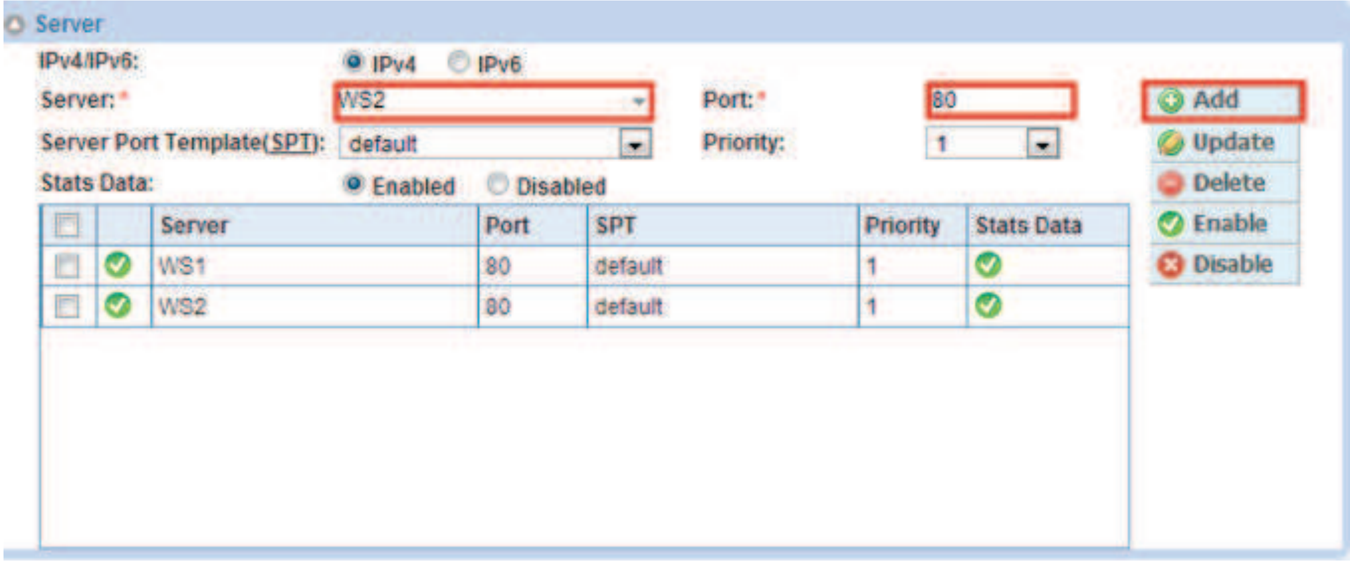
Service Group

Name: *	epicservers	
Type:	TCP	
Algorithm:	Least Connection	Pseudo Round Robin: □
Auto Stateless Method:	□	
Traffic Replication:		
Health Monitor:	epichc	
Server Template:	default	
Server Port Template:	default	
Policy Template:		
Min Active Members:	□	
Priority Affinity:	□	
□	Send client reset when server selection fails	
□	Send log information on backup server events	
Stats Data:	● Enabled ○ Disabled	
Extended Stats:	○ Enabled ● Disabled	

Figure 5: Service group configuration

4. From the Server section of the window, add one or more servers from the server drop-down list:
   Server: Select "**WS1**" from the drop-down menu
   Port: Enter "**80**"
5. Click "**Add**" and enter all the available Epic web servers.

The server names **WS1** and **WS2** are entered, each with port **80**.



Server

IPv4/IPv6: ● IPv4 ○ IPv6

Server: * WS2 Port: * 80

Server Port Template(SPT): default Priority: 1

Stats Data: ● Enabled ○ Disabled

□		Server	Port	SPT	Priority	Stats Data
□	✓	WS1	80	default	1	✓
□	✓	WS2	80	default	1	✓

Figure 6: Service group server configuration

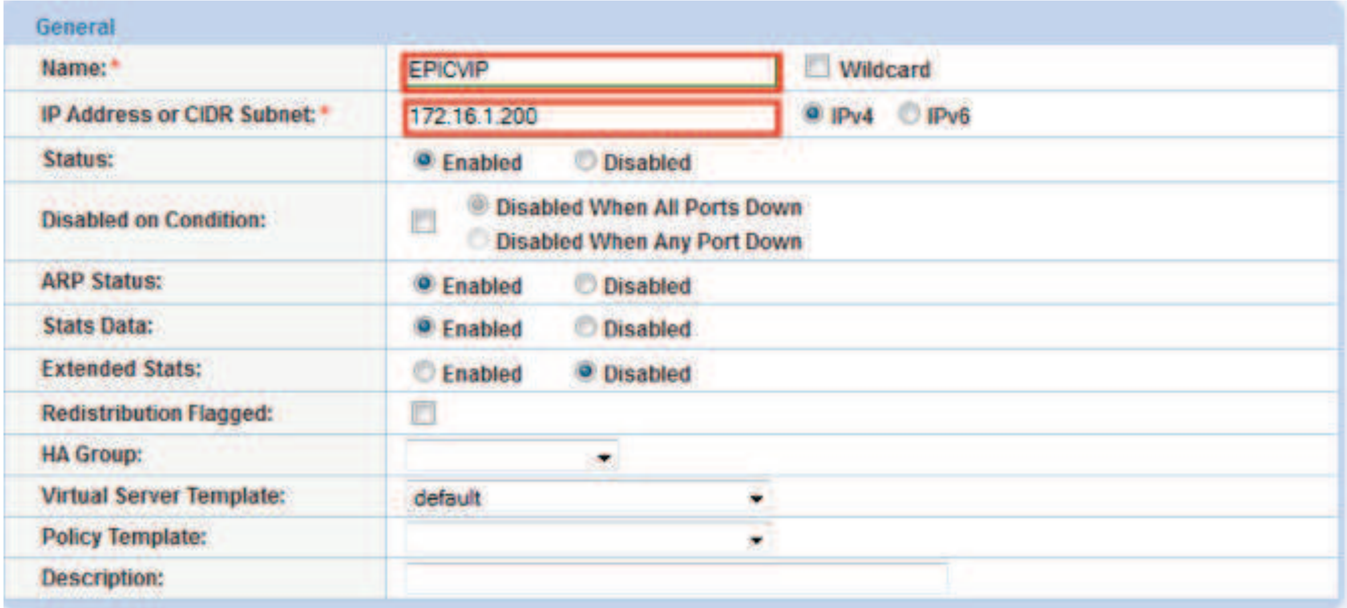
6. Once completed click "**OK**" and "**Save**" configuration

   Note: It is best practice that each Epic application must be in a service group. For example, if you have multiple Haiku servers, those servers should be provisioned to be in the same service group.

VIRTUAL SERVER

This section demonstrates how to configure the VIP with the Thunder ADC. Each Epic component will have its own VIP and you will be using HTTPS (443) for all application except Epic Print Service (EPS). Refer to the note below regarding EPS configuration.

1. Navigate to Config Mode > SLB > Service > Virtual Server
2. Within the **General** section, enter the following required information:
   - Name: "EPICVIP"
   - IP Address or CIDR Subnet: *172.16.1.200*

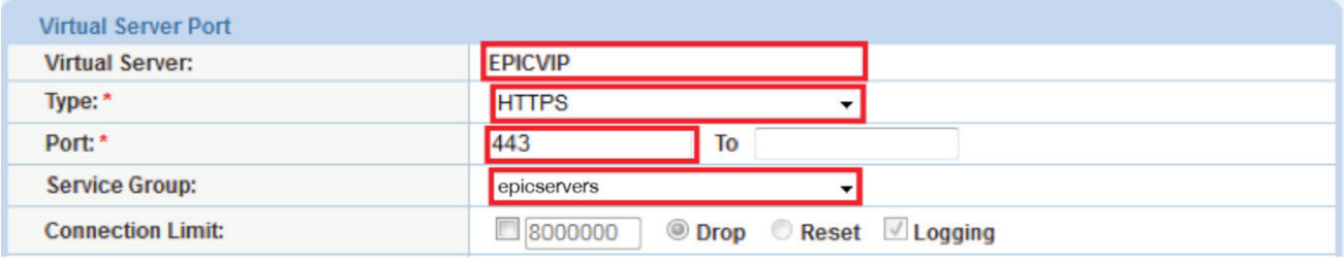


General

Name: *	EPICVIP □ Wildcard
IP Address or CIDR Subnet: *	172.16.1.200 ● IPv4 ○ IPv6
Status:	● Enabled ○ Disabled
Disabled on Condition:	□ ○ Disabled When All Ports Down ○ Disabled When Any Port Down
ARP Status:	● Enabled ○ Disabled
Stats Data:	● Enabled ○ Disabled
Extended Stats:	○ Enabled ● Disabled
Redistribution Flagged:	□
HA Group:	
Virtual Server Template:	default
Policy Template:	
Description:	

Figure 7: Virtual server or VIP configuration

3. In the **Port** section:
   - Click "**Add**".
   - Enter the Virtual Server Port information.
     1. Type: From the drop down menu select "**HTTPS**"
     2. Port: "**443**"
     3. Service Group: From the drop down menu select: "**epicservers**" to bind the virtual server to the real servers.

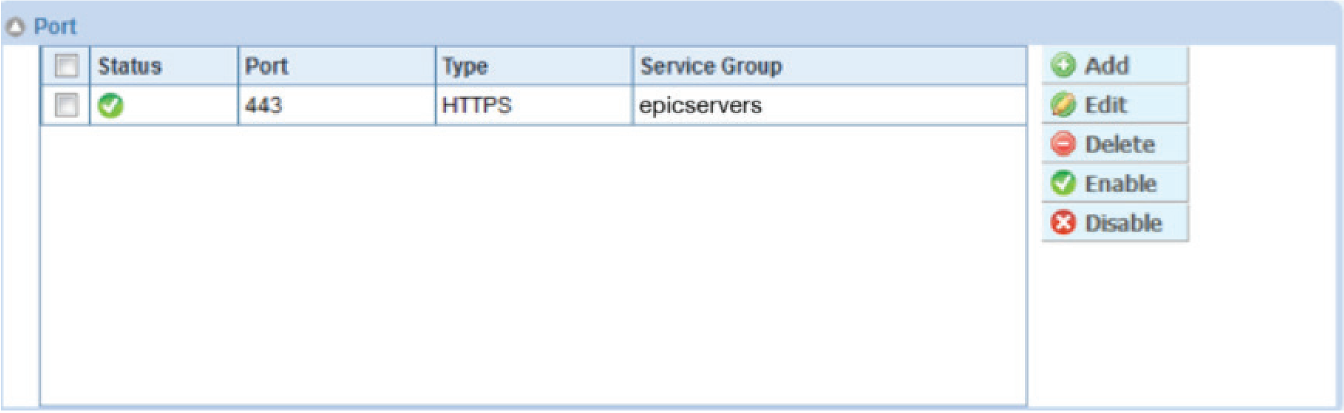


Virtual Server Port

Virtual Server:	EPICVIP
Type: *	HTTPS
Port: *	443 To
Service Group:	epicservers
Connection Limit:	□ 8000000 ● Drop ○ Reset ☑ Logging

Figure 8: Virtual server port configuration

4. Click "**OK**" and then click "**Save**" to store your configuration changes.



Port

□	Status	Port	Type	Service Group
□	✓	443	HTTPS	epicservers

Figure 9: Virtual port lists

5. Click "**OK**" and "**Save**" configuration.

Note: To configure Epic Print Service (EPS) you have to use port 21 for virtual server port (front end) and server port (back end). No acceleration or optimization needed except IP Source NAT maybe required depending on the topology your implementation. See sample configuration from the Appendix section.

FEATURE TEMPLATES

This section describes how to define and apply one or more the following feature templates:

- SSL Offload
- HTTP/HTTPS Compression
- Cookie Persistence
- TCP Connection Reuse
- TCP Proxy
- RAM Caching
- IP Source NAT

After configuring the feature templates apply them to the Virtual Server.

SSL OFFLOAD

SSL Offload acts as an acceleration feature by removing the burden of processing SSL traffic from the Epic web servers. Instead of having the Epic servers handling these transactions, the Thunder ADC Series decrypts and encrypts all HTTPS traffic and forwards the traffic to the Epic Server over HTTP (unsecured).

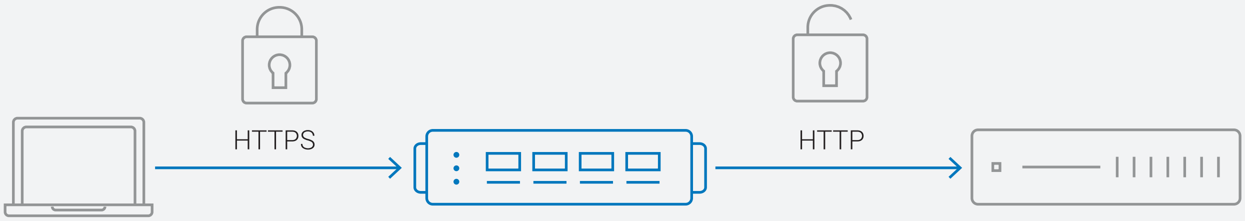


Figure 10: SSL Offload overview

To configure SSL Offload, following configurations are required:

- Use HTTP for the communication between Epic web servers and Thunder ADC.
- Use HTTPS on Virtual IP for the communication between clients and Thunder ADC
- Import existing Epic web server SSL cert or create self-signed CA on the Thunder ADC.
- Create SSL template and associate VIP with the SSL template

The list of ciphers supported by Thunder ADC can be found on A10 Support Portal in the document A10 Thunder SSL Cipher Suites List (ACOS 2.7.x – 4.1.0-Px).

IMPORT OR GENERATE CERTIFICATE

1. Navigate to **Config Mode > SLB > SSL Management > Certificate.**
2. There are two options to configure when installing an SSL template from the Thunder ADC either:

   Option 1: Generate a Self-Signed CA from the Thunder ADC

   Option 2: Import an SSL Certificate and Key:

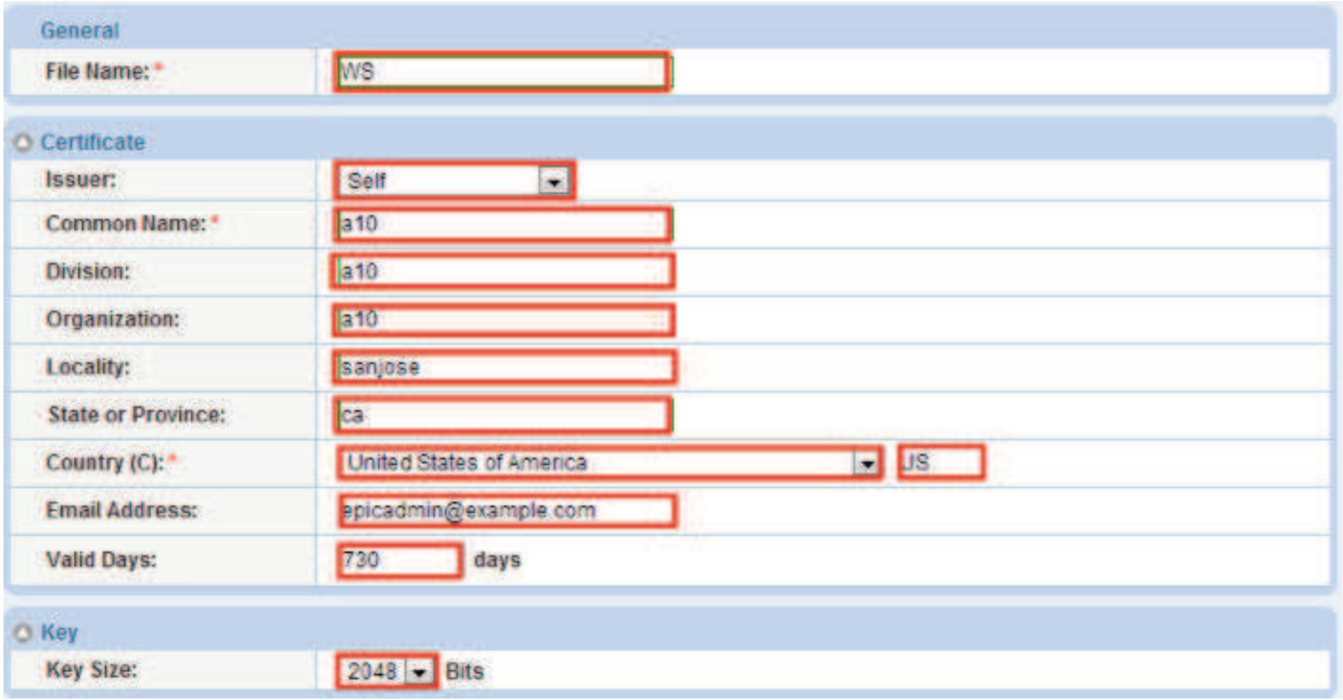
   Export existing CA certificate from Epic web servers and import to Thunder ADC.

OPTION 1: GENERATE A SELF-SIGNED CA FROM THE THUNDER

1. Click **Create** to add a new SSL certificate from the SSL Management.
2. Enter the File Name of the certificate: “**WS**”.
3. From the Issuer: Select “**Self**“ from the from the drop-down menu, and then enter the following values:
   - i. Common Name: “**WS**”
   - ii. Division: “**a10**”
   - iii. Organization: “**a10**”
   - iv. Locality: ”**sanjose**”
   - v. State or Province: “**ca**”
   - vi. Country: “**USA**”
   - vii. Email Address: “**epicadmin@example.com**”
   - viii. Valid Days: “**730**” (Default)
   - ix. Key Size (Bits): “**2048**”

Note: The Thunder ADC supports 1028-, 2048-, 4096-bit SSL key. The higher bit SSL key size, the more CPU processing will be required. The Thunder ADC SSL models handle the SSL transaction dedicated security processors.

4. Click “**OK**” and “**Save**” configuration.



General

File Name:*	WS

Certificate

Issuer:	Self
Common Name:*	a10
Division:	a10
Organization:	a10
Locality:	sanjose
State or Province:	ca
Country (C):*	United States of America US
Email Address:	epicadmin@example.com
Valid Days:	730 days

Key

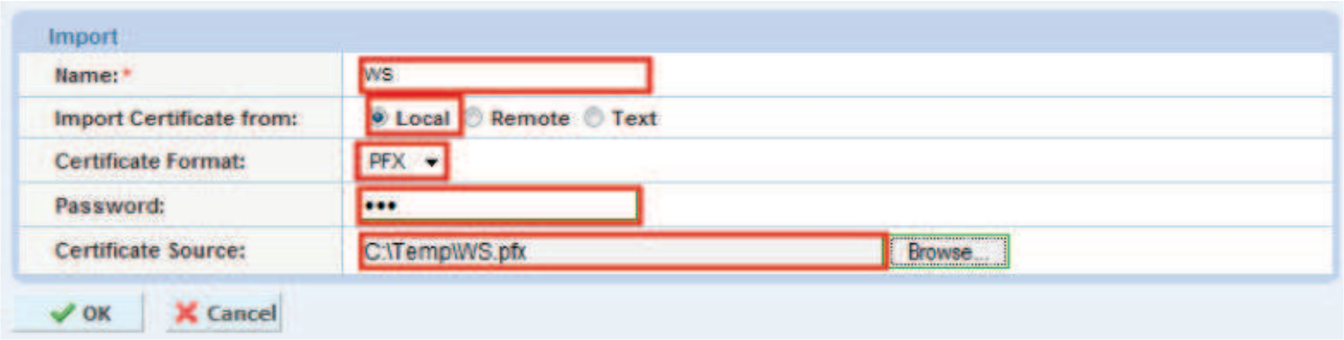
Key Size:	2048 Bits

Figure 11: Client SSL certificate creation

OPTION 2: IMPORT SSL CERTIFICATE AND KEY

1. Click “**Import**” to add a new SSL certificate from the SSL Management.
2. Enter a name for the certificate “**WS**”.
3. Select “**Local**” from **Import Certificate from**: (depends where the certificate is originating from).
4. Enter Certificate Password (if applicable).
5. Enter Certificate Source (if applicable).
6. Click “**OK**” and “**Save**” your configuration.

Note: If you are importing a CA-signed certificate for which you used the Thunder device to generate the CSR, you do not need to import the key. The key is automatically generated on the Thunder device when you generate the CSR.



Import

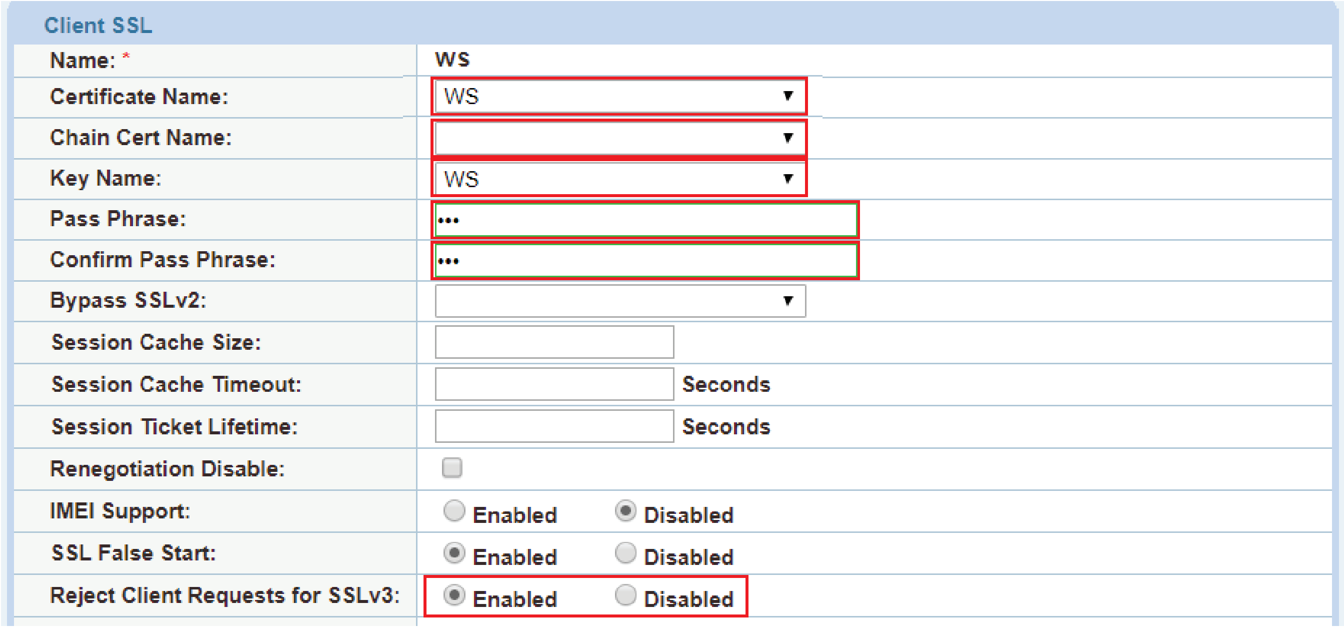
Name:*	WS
Import Certificate from:	● Local ○ Remote ○ Text
Certificate Format:	PFX
Password:	•••
Certificate Source:	C:\Temp\WS.pfx

Figure 12: Import SSL certificate

CREATE CLIENT SSL TEMPLATE

This section describes how to configure a client SSL template.

1. Navigate to Config Mode > SLB > SSL > Template > SSL > Client SSL.
2. Click “Add”.
3. Enter Name: “Client SSL-WS”.
4. Enter Certificate Name: “WS”.
5. Enter Key Name: “WS”.
6. Enter Pass Phrase: “example”.
7. Enter Confirm Pass Phrase: “example”.
8. Enable the option “Reject Client Requests for SSLv3”.



Client SSL

Name: *	WS
Certificate Name:	WS ▼
Chain Cert Name:	▼
Key Name:	WS ▼
Pass Phrase:	•••
Confirm Pass Phrase:	•••
Bypass SSLv2:	▼
Session Cache Size:	
Session Cache Timeout:	Seconds
Session Ticket Lifetime:	Seconds
Renegotiation Disable:	□
IMEI Support:	○ Enabled ● Disabled
SSL False Start:	● Enabled ○ Disabled
Reject Client Requests for SSLv3:	● Enabled ○ Disabled

Figure 13: Client SSL

9. Click “OK” and “Save” configuration.

HTTP/HTTPS COMPRESSION

Compression is a bandwidth optimization feature that condenses the HTTP objects that are sent from a web server. The purpose of compression is to transmit the requested data more efficiently (fewer amounts of data transmitted) and faster response times observed at the client side.

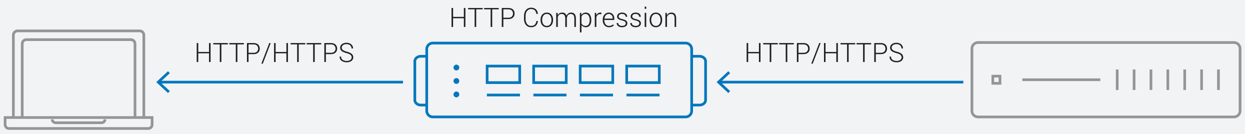
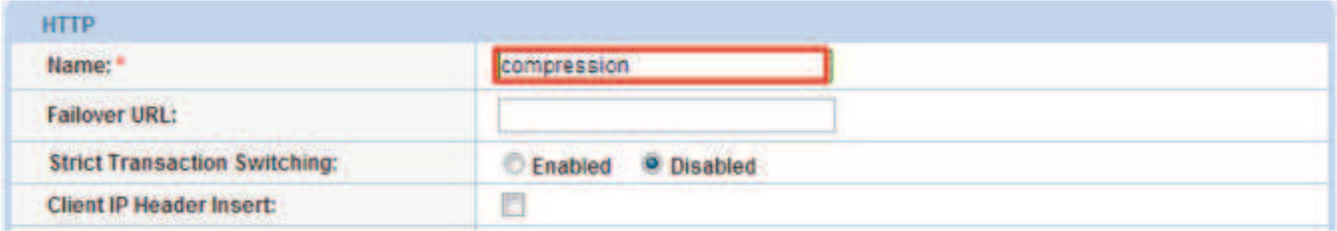


Figure 14: HTTP compression overview

CREATE HTTP/HTTPS COMPRESSION TEMPLATE

1. Navigate to **Config Mode** > **SLB** > **Template** > **Application** > **HTTP**.
2. Click "**Add**".
3. Enter Name: "**compression**".

Note: Compression is disabled by default. When compression is enabled, the following options will have these default values:

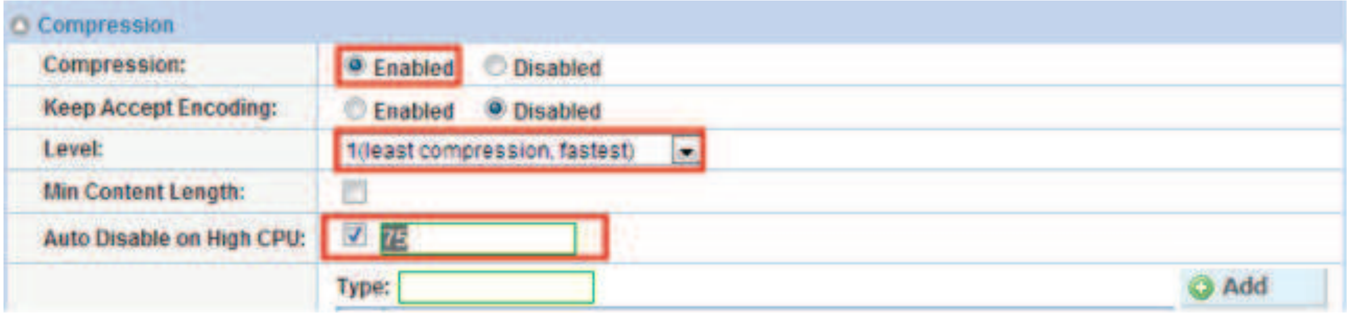


HTTP	
Name: *	compression
Failover URL:	
Strict Transaction Switching:	○ Enabled ● Disabled
Client IP Header Insert:	□

Figure 15: HTTP compression template

4. Click the "**Compression**" drop-down menu and enter the compression options.
5. Enter the desired compression options from the template:
   - Select "**Enable**" compression
   - Level: Select from the drop-down menu "**Level 1 (Least Level Compression)**"
   - Select **Auto Disable on High CPU** option and enter "**75**" (CPU percentage)

Note: The auto disable on high CPU option is an optional feature within the compression parameters. Administrators can select the CPU percentage that they are willing to use for the compression to terminate.



Compression	
Compression:	● Enabled ○ Disabled
Keep Accept Encoding:	○ Enabled ● Disabled
Level:	1(least compression, fastest)
Min Content Length:	□
Auto Disable on High CPU:	☑ 75
	Type: Add

Figure 16: Compression configuration column

6. Click "**OK**" and then click "**Save**" to store your configuration changes.

COOKIE PERSISTENCE

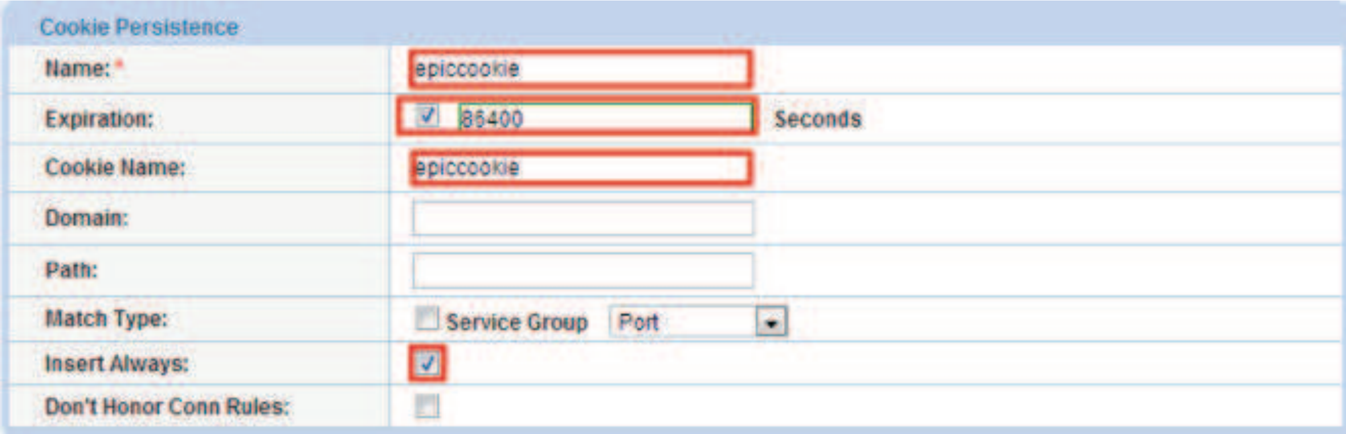
Cookie persistence enables you to insert a cookie into server responses to clients, to direct clients to the same service group, real server, or real service port for subsequent request for this service. The advantage of cookie persistence within the Epic solutions is to direct all requests to the same Epic backend server that was recently visited as long as the expiry time has not been exceeded.

CREATE COOKIE PERSISTENCE TEMPLATE

To enable cookie persistence the template must be created first, as follows:

1. Navigate to **Config mode** > **SLB** > **Template** > **Persistent** > **Cookie Persistence**.
2. Click "**Add**" to add a new cookie persistence template.
3. Select the Expiration radio button and enter "**86400**" in the **Seconds** field.

4. Cookie Name: "**epiccookie**".



Cookie Persistence

Name:*	epiccookie
Expiration:	☑ 86400 Seconds
Cookie Name:	epiccookie
Domain:	
Path:	
Match Type:	□ Service Group Port
Insert Always:	☑
Don't Honor Conn Rules:	□

Figure 17: Cookie persistence template

6. Click "**OK**" and then click "**Save**" to store your configuration changes.

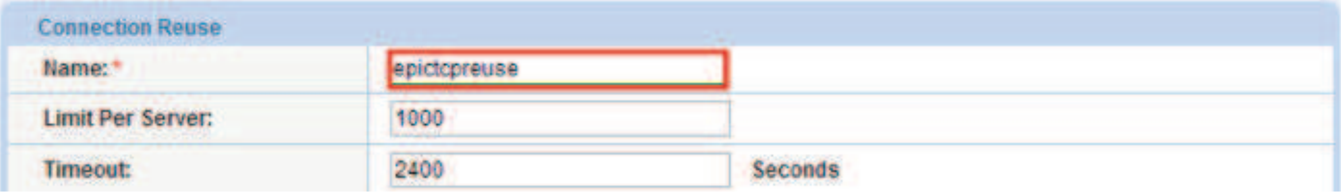
TCP CONNECTION REUSE

The Thunder ADC Connection Reuse feature reduces the overhead associated with TCP connection setup by establishing TCP connections with Epic web servers and then reusing those connections for multiple client requests. This reduces the total number of TCP connections to each Epic server.

The advantage of reusing connections is to off-load the server TCP stack in order to provide faster response times and to increase server scalability. If Connection Reuse is enabled, Source NAT must be enabled. Refer to Source NAT for configuration information.

CREATE CONNECTION REUSE TEMPLATE

1. Navigate to **Config Mode> SLB > Template > Connection Reuse**.
2. Click "**Add**".
3. Enter Name: "**epictcpreuse**".
4. Click "**OK**" and then click "**Save**" to store your configuration changes.



Connection Reuse

Name:*	epictcpreuse
Limit Per Server:	1000
Timeout:	2400 Seconds

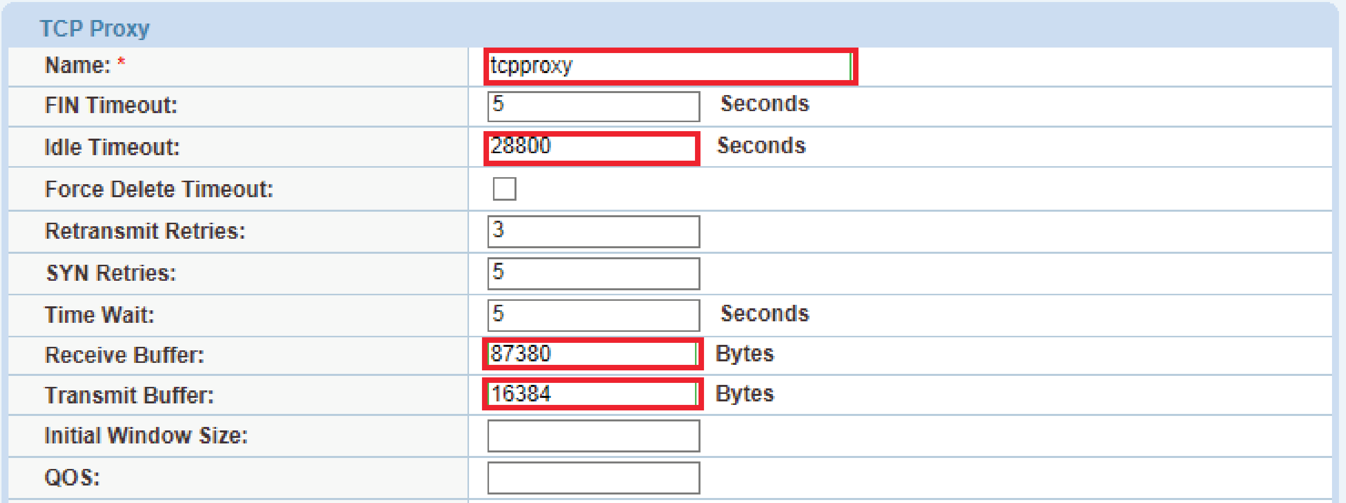
Figure 18: Connection reuse overview

TCP PROXY

TCP Proxy controls TCP stack settings, such as the TCP idle connection timeout. The TCP idle connection timeout determines how long users can be idle before the Thunder terminates the connection.

CREATE TCP PROXY TEMPLATE

1. Navigate to **Config Mode > Template > TCP Proxy**.
2. Click "**Add**".
3. Enter TCP Proxy Name: "**tcpproxy**".
4. Idle Timeout-number: "**28800**" (This is the number of seconds that a connection can be idle before the Thunder Series terminates the connection.
5. Receive Buffer: "**87380**" Bytes (Max number of bytes addressed to the port that the Thunder ADC will buffer.
6. Transmit Buffer: "**16384**" Bytes (Number of bytes sent by the port that the Thunder ADC will buffer.
7. Click "**OK**" and then click "**Save**" to store your configuration changes.



TCP Proxy

Name: *	tcpproxy	
FIN Timeout:	5	Seconds
Idle Timeout:	28800	Seconds
Force Delete Timeout:	□	
Retransmit Retries:	3	
SYN Retries:	5	
Time Wait:	5	Seconds
Receive Buffer:	87380	Bytes
Transmit Buffer:	16384	Bytes
Initial Window Size:		
QOS:		

Figure 19: TCP proxy configuration

RAM CACHING

Cacheable data is cached within the Thunder ADC, thus reducing overhead on each Epic server and increasing their capacity. RAM caching reduces the number of connections and server requests that need to be processed on the backend servers. This feature is essential within the Epic server as documents and pictures are commonly exchange across the Epic network.

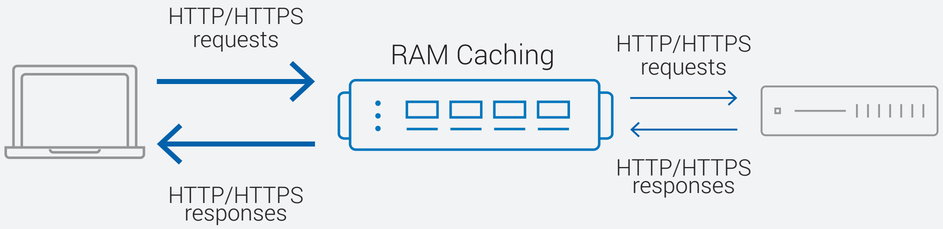
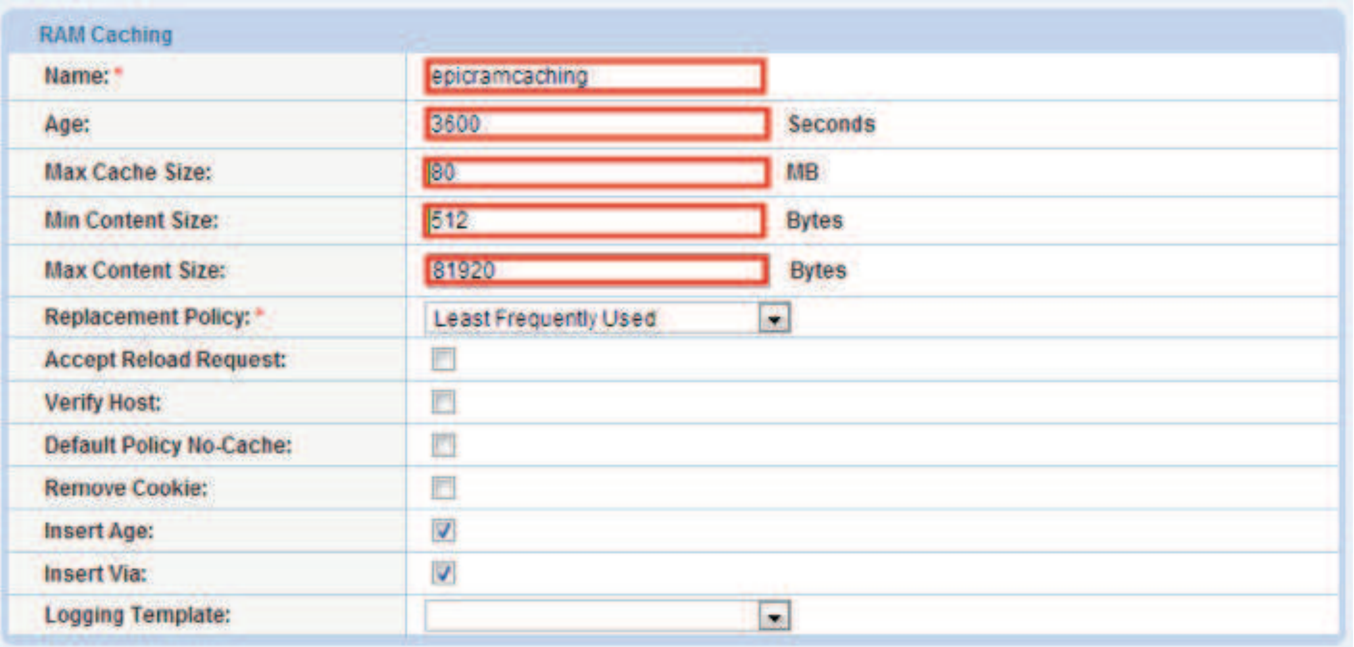


Figure 20: RAM caching template

CREATE RAM CACHING TEMPLATE

1. Navigate to **Config Mode > SLB > Template > Application > RAM Caching**.
2. Click "**Add**".
3. Enter name: "**epicramcaching**".
4. Enter Age: **3600** seconds.
5. Max Cache Size: **80 MB.**
6. Min Content Size: **512 Bytes.**
7. Max Content Size: **81920 Bytes.**
8. Enter the desired Replacement Policy from the drop down menu: "**Least Frequently Used**" (Default).
9. Click "**OK**" and then click "**Save**" to store your configuration changes.



RAM Caching

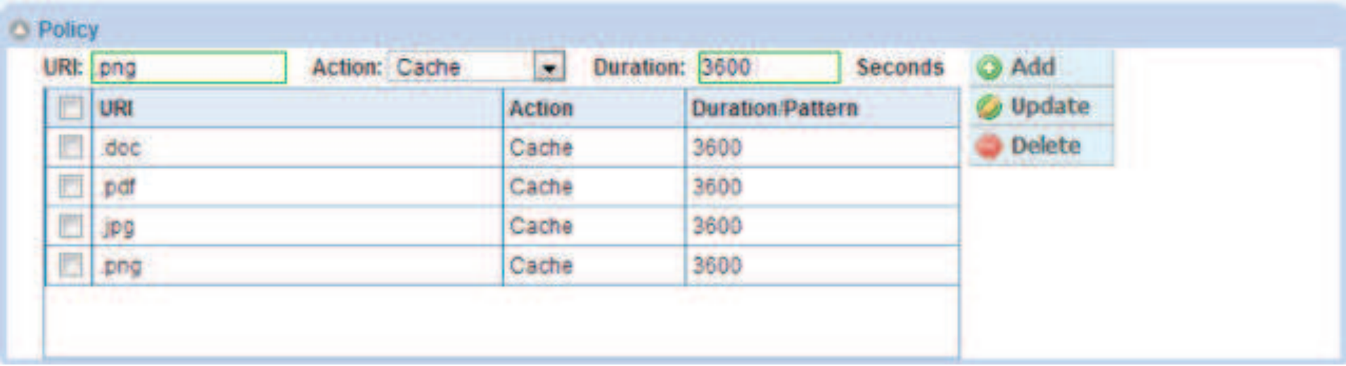
Name: *	epicramcaching	
Age:	3600	Seconds
Max Cache Size:	80	MB
Min Content Size:	512	Bytes
Max Content Size:	81920	Bytes
Replacement Policy: *	Least Frequently Used	
Accept Reload Request:	□	
Verify Host:	□	
Default Policy No-Cache:	□	
Remove Cookie:	□	
Insert Age:	☑	
Insert Via:	☑	
Logging Template:		

Figure 21: RAM caching overview

Additionally, you can configure policies for dynamic RAM caching. Dynamic RAM caching policies override and augment standard HTTP behavior.

To configure a cache policy:

1. In the **URI** field of **Policy** section, enter the portion of the URI string to match on.
2. Select "Cache" from the **Action** drop-down list. The **Duration** field appears.
3. By default, the content is cached for the number of seconds specified in the **Age** field of the RAM Caching section. To override the aging period, specify the number of seconds in the Duration field.
4. Click **Add**.
5. Click "**OK**" and then click "**Save**" to store your configuration changes.


Policy

URI: .png Action: Cache Duration: 3600 Seconds

	URI	Action	Duration/Pattern
□	.doc	Cache	3600
□	.pdf	Cache	3600
□	.jpg	Cache	3600
□	.png	Cache	3600

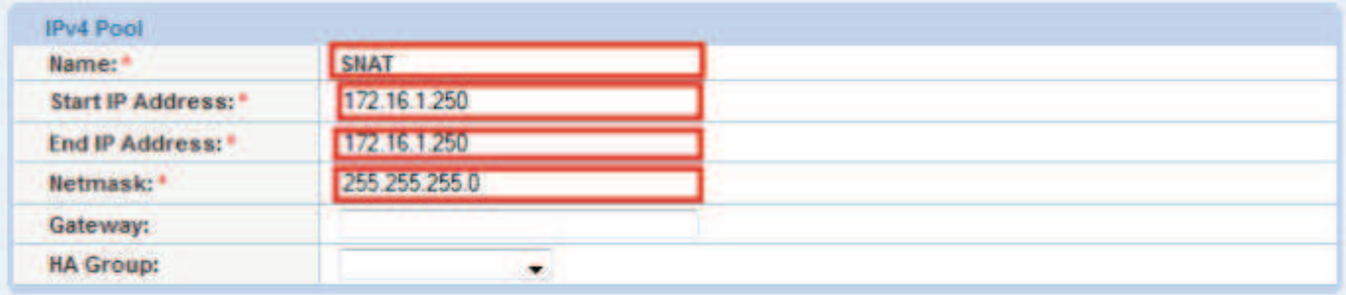
Figure 22: Dynamic RAM caching policy

IP SOURCE NAT

This section configures the IP Address pool to be used for IP Source Network Address Translation (SNAT). When incoming traffic from a client accesses the VIP address (For example: 172.16.1.200), the client requests are "source NAT-ed", which means that the Thunder ADC replaces the client's source IP address based on the configured address pool of the source NAT. SNAT is required when your network topology is based on "one-arm" deployment and if you have internal clients that reside on the same subnet as the VIP. The Source NAT template must be applied in the virtual server port for the NAT to take effect.

CREATE IP SOURCE NAT TEMPLATE

1. Navigate to Config Mode > IP Source NAT > IPv4 Pool.
2. Click "Add".
3. Enter IP Source NAT Name: "SNAT".
4. Enter Start IP Address:*172.16.1.250* (Example).
5. Enter End IP Address: *172.16.1.250* (Example).
6. Enter Netmask: *255.255.255.0*


IPv4 Pool

Name: *	SNAT
Start IP Address: *	172.16.1.250
End IP Address: *	172.16.1.250
Netmask: *	255.255.255.0
Gateway:	
HA Group:	

Figure 23: IP Source NAT configuration

7. Click "OK" and "Save" configuration.
   To assign the template to the VIP, navigate to

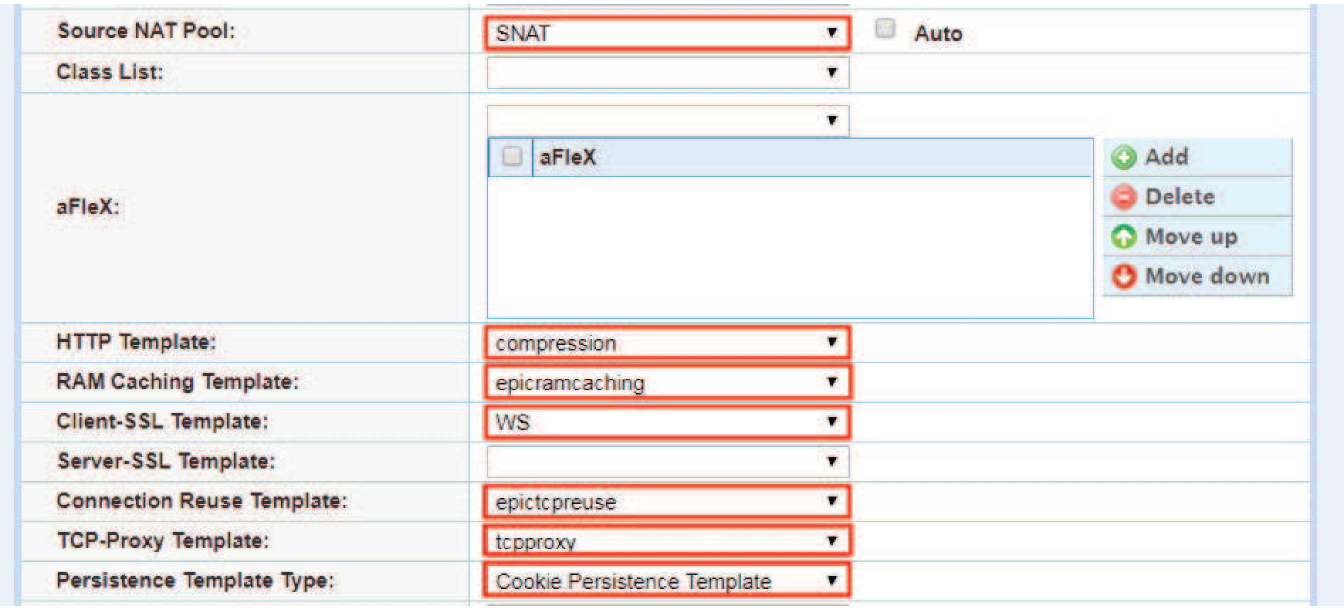
Note: If the Epic environment will consist of many concurrent users, it is advisable to configure multiple SNAT IP addresses. One IP address can be used for up to 64,000 flows.

APPLY FEATURE TEMPLATES

After configuring the feature templates apply them to the Virtual Server.

Navigate to **Config Mode** > **SLB** > **Service** > **Virtual Server**

1. Select the on Virtual Server name
2. Select “443” and click “**Edit**”
3. Apply the templates as shown
4. Click “**OK**” and “**Save**” configuration



Source NAT Pool:	SNAT ▾ □ Auto
Class List:	▾
aFleX:	▾ □ aFleX
HTTP Template:	compression ▾
RAM Caching Template:	epicramcaching ▾
Client-SSL Template:	WS ▾
Server-SSL Template:	▾
Connection Reuse Template:	epictcpreuse ▾
TCP-Proxy Template:	tcpproxy ▾
Persistence Template Type:	Cookie Persistence Template ▾

Figure 24: Apply feature templates

SUMMARY AND CONCLUSION

In summary, the configuration steps described above show how to set up the Thunder ADC for Epic Systems Electronic Medical Records. By using the Thunder ADC to load balance Epic Systems web servers, the following key advantages are achieved:

- Obtain higher availability when if an Epic Web Server fails, meaning there is no direct impact on how users can access the applications.
- Lower CPU utilization rates as Thunder ADC transparently load balances requests across multiple Epic Systems applications and web servers.
- Higher connection throughput and faster end user responsiveness by off-loading security processing to the Thunder ADC.

By using the Thunder ADC, significant benefits are achieved for all Epic Systems users. For more information about A10 Thunder ADC products, please refer to the following URLs:

www.a10networks.com/products/load-balancer-application-delivery

www.a10networks.com/solutions/healthcare

APPENDIX

Thunder ADC CLI sample configurations:

```
ip nat pool SNAT 172.16.1.250 172.16.1.250
netmask /24
health monitor epichc
 method http
slb server WS1 172.16.1.5
   health-check epichc
   port 80  tcp
slb server WS2 172.16.1.6
   health-check epichc
   port 80  tcp
slb service-group epicservers tcp
    method least-connection
    health-check epichc
    member WS1:80
    member WS2:80
slb template connection-reuse epictcpreuse
slb template tcp-proxy tcpproxy
   idle-timeout 28800
   receive-buffer 87380
transmit-buffer 16384
slb template cache epicramcaching
   policy uri .doc cache
   policy uri .pdf cache
   policy uri .jpg cache
   policy uri .png cache
slb template http compression
   compression auto-disable-on-high-cpu 75
slb template client-ssl WS
   cert “WS“
   chain-cert “WS“
   key “WS“
   cipher TLS1_DHE_RSA_AES_128_GCM_SHA256
   cipher TLS1_DHE_RSA_AES_128_SHA
   cipher TLS1_DHE_RSA_AES_128_SHA256
   cipher TLS1_DHE_RSA_AES_256_GCM_SHA384
   cipher TLS1_DHE_RSA_AES_256_SHA
   cipher TLS1_DHE_RSA_AES_256_SHA256
   cipher TLS1_ECDHE_ECDSA_AES_128_GCM_SHA256
   cipher TLS1_ECDHE_ECDSA_AES_128_SHA
   cipher TLS1_ECDHE_ECDSA_AES_128_SHA256
   cipher TLS1_ECDHE_ECDSA_AES_256_GCM_SHA384
   cipher TLS1_ECDHE_ECDSA_AES_256_SHA
   cipher TLS1_ECDHE_RSA_AES_128_GCM_SHA256
   cipher TLS1_ECDHE_RSA_AES_128_SHA
   cipher TLS1_ECDHE_RSA_AES_128_SHA256
   cipher TLS1_ECDHE_RSA_AES_256_GCM_SHA384
   cipher TLS1_ECDHE_RSA_AES_256_SHA
   cipher TLS1_RSA_AES_128_GCM_SHA256
   cipher TLS1_RSA_AES_128_SHA
   cipher TLS1_RSA_AES_128_SHA256
   cipher TLS1_RSA_AES_256_GCM_SHA384
   cipher TLS1_RSA_AES_256_SHA
   cipher TLS1_RSA_AES_256_SHA256
   disable-sslv3
slb template persist cookie epiccookie
   name epiccookie
   expire 86400
slb virtual-server EPICVIP 172.16.1.200
   port 443  https
      name _172.16.1.200_HTTPS_443
      source-nat pool SNAT
      service-group epicservers
      template tcp-proxy tcpproxy
      template http compression
      template cache epicramcaching
      template client-ssl WS
      template connection-reuse epictcpreuse
      template persist cookie epiccookie
end
```

ABOUT A10 NETWORKS

A10 Networks (NYSE: ATEN) provides Reliable Security Always™ through a range of high-performance solutions that enable intelligent automation with deep machine learning to ensure business critical applications are protected, reliable and always available. Founded in 2004, A10 Networks is based in San Jose, Calif., and serves customers globally with offices worldwide.

For more information, visit: a10networks.com or tweet @a10Networks

LEARN MORE

ABOUT A10 NETWORKS

CONTACT US

a10networks.com/contact

Part Number: A10-DG-16131-EN-03 SEP 2018