Deploying Microsoft Share-Point 2016 with NetScaler

Deployment Guide

This guide focuses on defining the process for deploying Microsoft SharePoint 2016 with Citrix NetScaler

Table of Contents

Citrix NetScaler is a world-class product with the proven ability to load balance, accelerate, optimize, and secure enterprise applications.

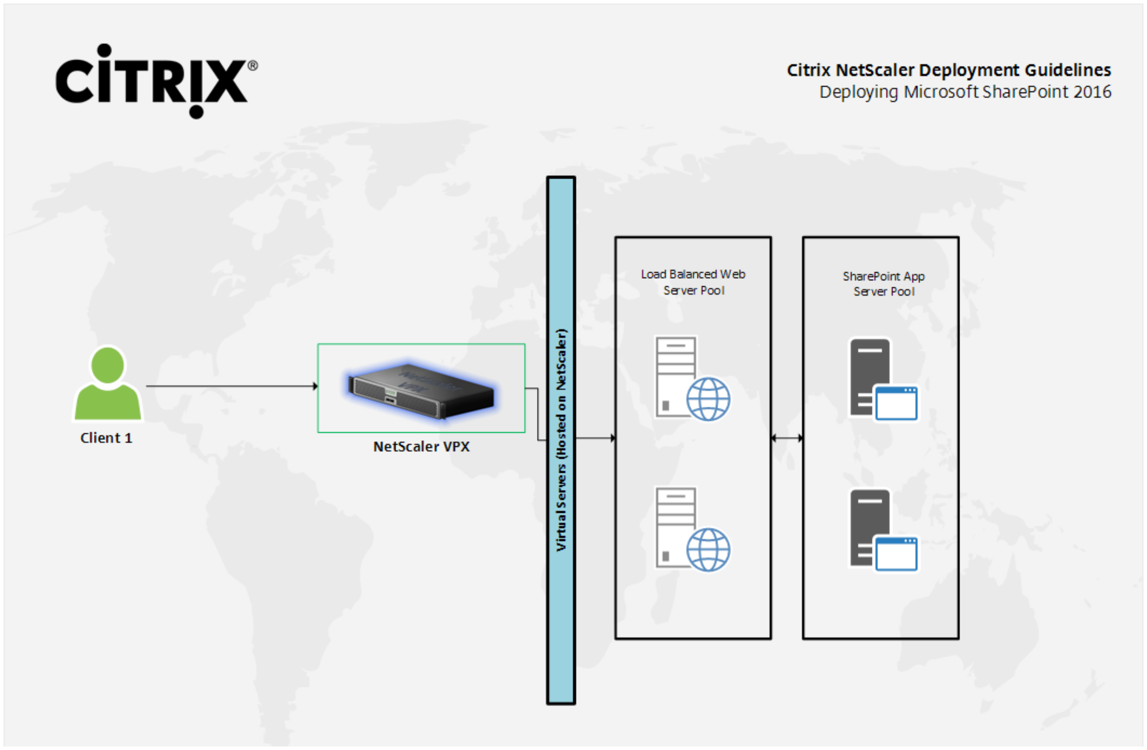
For several years, Citrix has completed certifications and provided deployment guides for key enterprise applications. NetScaler's rich application delivery capabilities significantly enhance the performance of these applications. With a comprehensive feature set, It provides availability, scalability, optimization and security for Microsoft SharePoint deployments.

Introduction

This guide defines the process for deploying Microsoft SharePoint 2016 with NetScaler.
Citrix NetScaler is a world class application delivery controller, with the proven ability to load balance, accelerate, secure and optimize enterprise applications.

Microsoft SharePoint 2016 is a web-based collaboration platform that enables users to share enterprise information and assets. Launched in 2001, SharePoint is primarily sold as a document management and storage system, but the product is highly configurable and usage varies substantially between organizations. It is supported by all major browsers.

Configuration


CITRIX
Citrix NetScaler Deployment Guidelines
Deploying Microsoft SharePoint 2016
Client 1
NetScaler VPX
Virtual Servers (Hosted on NetScaler)
Load Balanced Web Server Pool
SharePoint App Server Pool

Recommended Product Versions

Product	Version
Microsoft SharePoint	2016
NetScaler VPX	11.0 (Platinum License) – Load Balancing, Compression, Caching and FEO 11.0 (Standard License) – Only Load Balancing

NetScaler features

The following NetScaler features are discussed in this deployment guide.

- Load balancing
- Content Switching*
- HTTP Compression
- Front End Optimization (FEO)
- Integrated Caching
- Note: Content Switching is required only if a content-switched SharePoint environment is required, which can provide more granular control over the handling of different types of content at the cost of increased deployment time. While this guide will describe the basic setup configuration, the advanced configuration can be implemented using the AppExpert template for SharePoint that can be found at https://www.citrix.com/static/appexpert/appexpert-template.html. FEO will require a Platinum license for successful operation.
-

Other considerations

- Make sure you have installed, at a minimum, one license for NetScaler Enterprise/Platinum Edition.
- Set the time zone and a NTP (Network Time Protocol) server, and check the date and time on the NetScaler virtual appliance.
- Configure your DNS settings accurately.

There are two deployment architectures that can be used for this deployment. The basic deployment utilizes a load balanced virtual server for providing access to SharePoint, while the advanced configuration provides a content switched environment with custom settings for different types of requests and content types. While the performance levels of both versions are similar, the advanced version will allow for more granular control if more advanced features are implemented such as authentication and selective access to content.

Steps for load balancing configuration

Broadly, the steps to configure a load balanced SharePoint Server setup are as follows:

1. Complete initial setup for the SharePoint Server;
2. Create a service for each SharePoint Server and bind the server objects and appropriate monitors to it.
3. Now, create load balancing virtual servers (load balancing vservers) for the SharePoint Server service and bind the appropriate services and certificate to them. For this deployment, we have used a self-signed certificate; however you may use any valid server certificate.
4. When defining the load balancing vservers, provide a valid, addressable IP address.
5. Set an appropriate load balancing method (such as LEASTCONNECTION) and a persistence method such as SOURCEIP. These will ensure effective load balancing.
6. Set up compression/caching/FEO policies and bind them to the load balancing virtual servers

Solution Description

Quick Configuration Table

Configuration Item	Version
Load Balancing (Traffic Management>Load Balancing>Virtual Servers in the GUI)	Virtual Servers: SharePoint_lb_ssl, SharePoint_lb (Suggested Names) **SharePoint_lb_ssl** **Protocol:** HTTPS **Port:** 443 (or alternate as per your configuration) **Load Balancing Method:** Roundrobin/ LeastConnection **Services Bound:** SharePoint1_svc SharePoint2_svc **Compression Policy:** SharePoint_ Compression_Test **Cache Policy:** SharePoint_Cache_Test **FEO Policy:** SharePoint_Optimization_ Test **Certificate Binding:** Standard Wildcard/SAN/SNI Server certificate support (Bind the appropriate server certificate as per your configuration) **CLI Commands:** *add lb vserver SharePoint_lb_ssl SSL <IP address for vserver> 443 -persistenceType NONE -cltTimeout 180* **SharePoint_lb** **Protocol:** HTTP **Port:** 80 (or alternate as per your configuration) **Load Balancing Method:** Roundrobin/ LeastConnection **Services Bound:** SharePoint1_svc SharePoint2_svc **Compression Policy:** SharePoint_ Compression_Test **Cache Policy:** SharePoint_Cache_Test **FEO Policy:** SharePoint_Optimization_ Test **CLI Commands:** *add lb vserver SharePoint_lb HTTP <IP address for vserver> 80 -persistenceType NONE -lbMethod ROUNDROBIN -cltTimeout 180 -downStateFlush DISABLED*
Service Configuration (System>Load Balancing>Services) *Note: Both backend services are HTTP here*	**SharePoint1_svc** *Protocol: HTTP* *Port: 80 (or alternate as per your configuration)* *IP: IP address of 1st SharePoint server* **SharePoint2_svc** *Protocol: HTTP* *Port: 80 (or alternate as per your configuration)* *IP: IP address of 2nd SharePoint server* CLI Commands: *add service SharePoint1_svc <IP address for 1st CRM front end server> HTTP 80 -gslb NONE -maxClient 0 -maxReq 0 -cip ENABLED X-Forwarded-for -usip NO -useproxyport NO -sp ON -clt-Timeout 180 -svrTimeout 360 -CKA NO -TCPB NO -CMP YES* *add service SharePoint2_svc <IP address for 2nd CRM front end server> HTTP 80 -gslb NONE -maxClient 0 -maxReq 0 -cip DISABLED -usip NO -useproxyport NO -sp ON -cltTimeout 180 -svrTimeout 360 -CKA NO -TCPB NO -CMP YES*
Compression Policy Definition *(Optimization>Integrated Caching>Policies)*	Policy Name: SharePoint_Compression_Test Response Action: COMPRESS (GZIP/DEFLATE should work too) Expression: *ns_true* CLI Commands: *add cmp policy SharePoint_Compression_Test -rule ns_true -resAction GZIP* *bind lb vserver SharePoint_lb -policyName SharePoint_Compression_Test -priority 100* *bind lb vserver SharePoint_lb_ssl -policyName SharePoint_Compression_Test -priority 100*

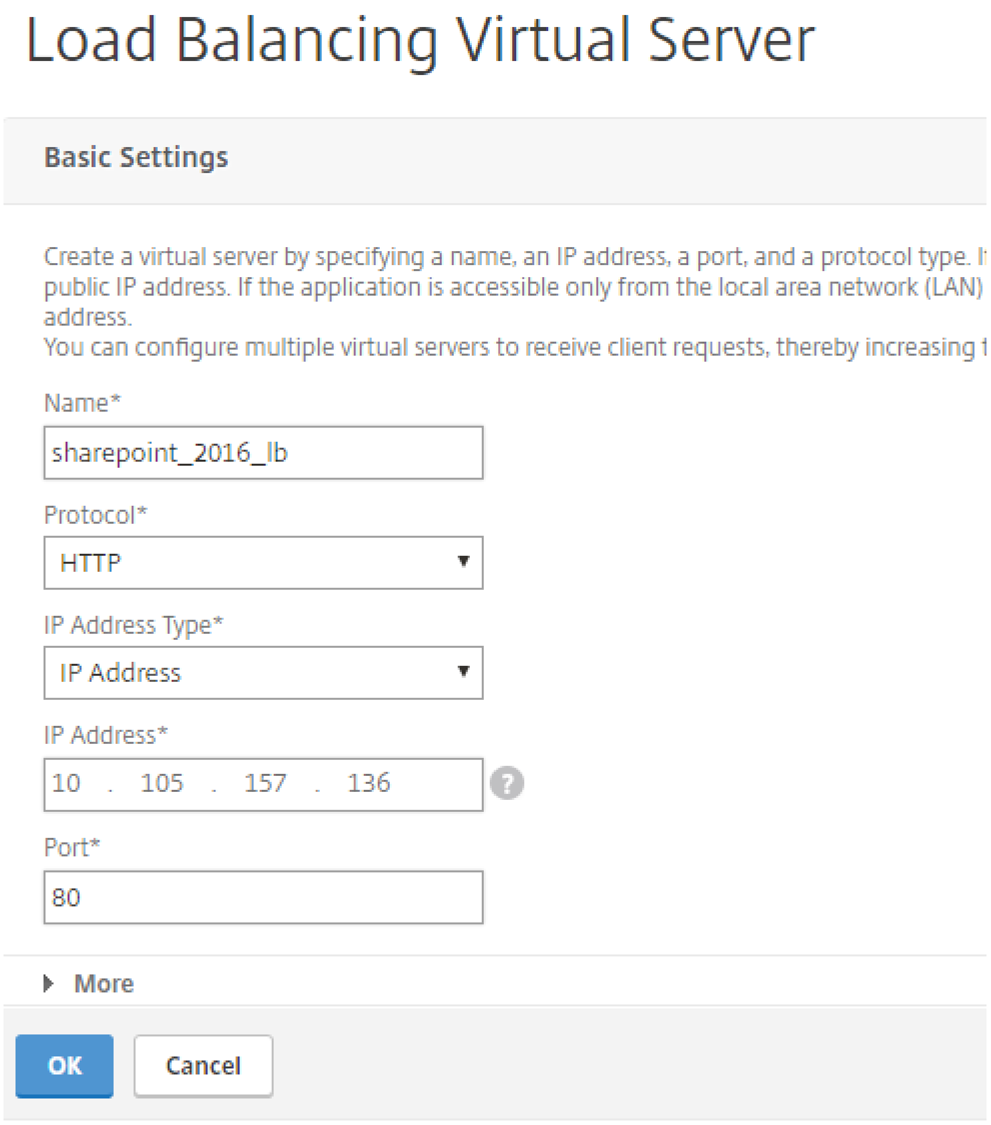
Configuration Item	Version
Cache Policy (Optimization>Integrated Caching>Policies)	Policy Name: SharePoint_Cache_Test Actions: CACHE Cache Content Group: Test Undefined-Result Action: -Global-undefined-result-action (or NOCACHE/RESET) Expression: *ns_true* Cache Content Group: Name: Test Type: HTTP Expiry Method: Heuristic (Recommended)/Custom (if specific settings are required) Default Expiry Times: As per requirement; set to 233 for test deployment. Parameterization: Leave values as is (unless Cache selectors are in use; not configured for our test setup) Memory: Define values as per your system limits Others: Use default settings. All settings have context-sensitive help available if modification is required. CLI Commands: *add cache policy SharePoint_Caching_Test -rule "SYS.EVAL_CLASSIC_EXPR(\"ns_true\")" -action CACHE -storeInGroup SharePoint_Caching_Test*
FEO (Front End Optimization) Policy (Optimization>Front end Optimization>Policies)	Optimization Policy Name: SharePoint_Optimization_Test Optimization Action: MODERATE (Preconfigured) Expression: HTTP.REQ.HEADER("Accept").CONTAINS("html") Alternate Configuration (Custom Policy): Optimization Policy Name: SharePoint_Optimization_TestCustom Optimization Action: samplefeo Expression: *HTTP.REQ.HEADER("Accept").CONTAINS("html")* SharePoint_Optimization_TestCustom Configuration: Enabled Settings: JavaScript/Make Inline, JavaScript/Move to End of Body Tag, JavaScript/Minify, Image/Optimize, Image/Lazy Load, Image/Shrink to Attributes, Image/Optimize, Image/Convert to JXR format, Image/Convert GIF to PNG, CSS/Make Inline, CSS/Move to Head Tag, CSS/Minify, CSS/Image Inline, CSS/Combine, CSS/Convert Imports to Links, HTML/Remove Comments from HTML CLI Commands: *add feo policy SharePoint_Optimization_Test "HTTP.REQ.HEADER(\"Accept\").CONTAINS(\"html\")" MODERATE* *add feo policy SharePoint_Optimization_Testcustom "HTTP.REQ.HEADER(\"Accept\").CONTAINS(\"html\")" MS_SP_custom* *bind lb vserver SharePoint_lb -policyName SharePoint_Optimization_Testcustom -priority 100 -gotoPriorityExpression END -type REQUEST* *bind lb vserver SharePoint_lb_ssl -policyName SharePoint_Optimization_Test -priority 100 -gotoPriorityExpression END -type REQUEST*

Configuring Load Balancing

A load balancing configuration consists of the definition of load balancing virtual servers (LB vServers), as well as services that are bound to the LB vservers. A service is simply a combination of a server and a protocol (e.g. HTTP, Port 80 or HTTPS, port 443).

Step 1 - Define the load balancing virtual servers (LB vservers)
Log into the NetScaler GUI. On the Configuration tab, navigate to Traffic Management>Load Balancing>Virtual Servers. For this deployment exercise, we are load balancing two Microsoft SharePoint 2016s. Here, we will create two load balancing virtual servers – SharePoint_lb (HTTP Port 80) and SharePoint_lb_ssl (HTTPS/SSL Port 443). Note that either one of the two can also be set up and will suffice for successful load balancing.

When defining a new LB vserver, you will be presented with the settings screen. Here, set the protocol to HTTP for the first vserver and SSL for the second. Set the IP address to the appropriate value.
(The steps shown here are for the HTTP vserver. Follow the same steps to configure the SSL vserver as well, only select port 443 as the port and SSL as the protocol)


Load Balancing Virtual Server

Basic Settings

Create a virtual server by specifying a name, an IP address, a port, and a protocol type. I
public IP address. If the application is accessible only from the local area network (LAN)
address.
You can configure multiple virtual servers to receive client requests, thereby increasing t

Name*: sharepoint_2016_lb

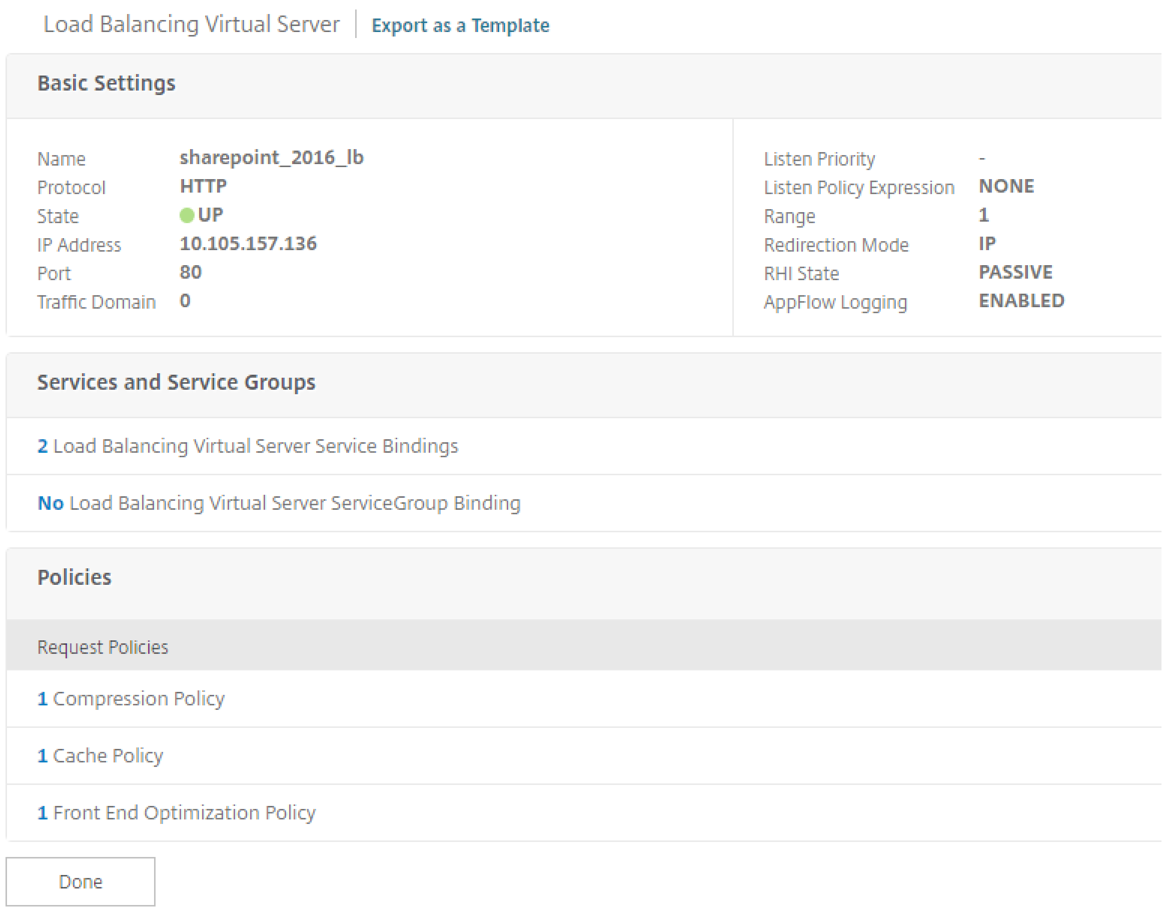
Protocol*: HTTP

IP Address Type*: IP Address

IP Address*: 10 . 105 . 157 . 136

Port*: 80

After clicking OK, you will see the Basic Settings screen for the LB vserver. Here, you may change settings such as the session persistence method, authentication and load balancing methods. Set session persistence to COOKIEINSERT, the timeout to 720 minutes (12 hours) and the load balancing method to LEASTCONNECTION for both virtual servers. For more information on these features, please refer to https://docs.citrix.com/en-us/netscaler/11.html


Load Balancing Virtual Server
Export as a Template
Basic Settings

Name	sharepoint_2016_lb	Listen Priority	-
Protocol	HTTP	Listen Policy Expression	NONE
State	● UP	Range	1
IP Address	10.105.157.136	Redirection Mode	IP
Port	80	RHI State	PASSIVE
Traffic Domain	0	AppFlow Logging	ENABLED

Services and Service Groups
2 Load Balancing Virtual Server Service Bindings
No Load Balancing Virtual Server ServiceGroup Binding
Policies
Request Policies
1 Compression Policy
1 Cache Policy
1 Front End Optimization Policy
Done

Note: In this deployment, we will be putting together an HTTP deployment. To enable an SSL-based LB vserver, you should add an SSL certificate and key pair, post which the other steps remain the same. For this, you may use either a self-signed certificate generated on the NetScaler appliance or a CA (Certificate Authority) signed one. The steps for generating a self-signed certificate on the NetScaler are as follows -

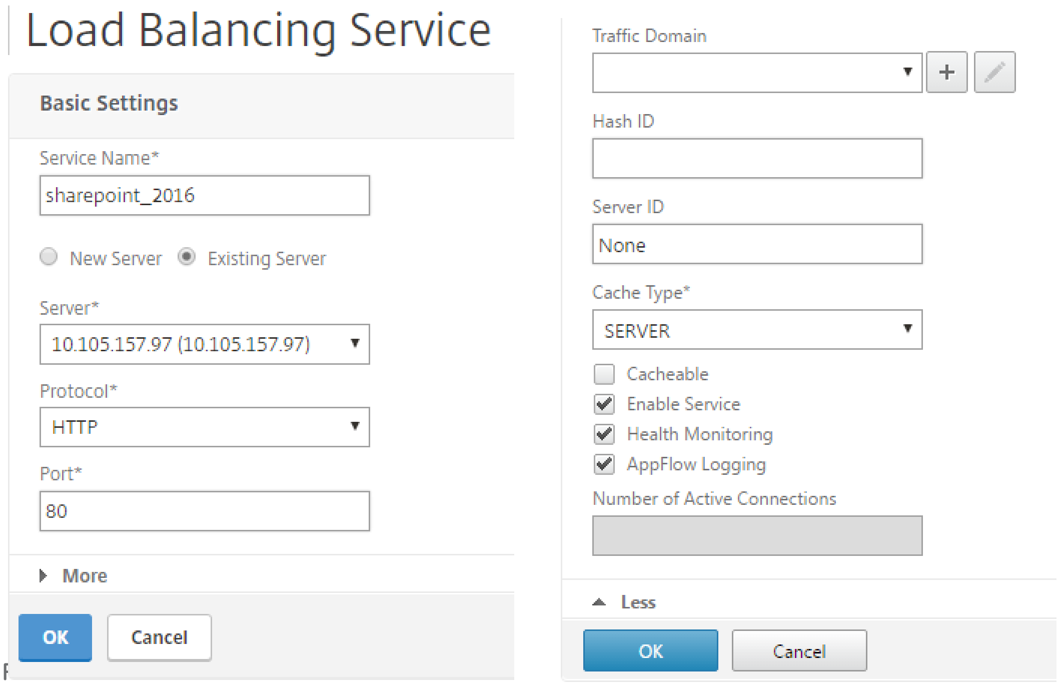
1. Login to your NetScaler appliance via the Configuration Utility.
2. Select Traffic Management > SSL
3. On the right, under Tools, select Server Certificate Wizard.
4. Here, the wizard will lead you through the series of steps for generating the self signed certificate –

- Generate the private key
- Generate the CSR (Certificate Signing Request)
- Generate the Certificate (using the ns-root.cer NetScaler root certificate)
- Save the Certificate and Key pair

Alternatively, if a certificate and key pair is already available, the same can be added by navigating to SSL>Certificates and clicking on the Add button. For more details refer to http://support.citrix.com/article/CTX109260

To improve site security and achieve an A/A+ rating on the SSLLabs.com evaluation, refer to https://www.citrix.com/blogs/2016/06/09/scoring-an-a-at-ssllabs-com-with-citrix-netscaler-2016-update/

Step 2 – Define LBVS server service group binding
Now click on the Load Balancing Virtual Server Service Binding tab in the Service and Service Groups section, or alternatively, click on Services in the Traffic Management>Load Balancing subsection and then, click on the Add button.
Every LB service is linked to a server; this can either be a new server or an existing server already defined in the Servers subsection under Load Balancing. Service groups extend this by allowing the creation of a group of services. An LB vserver can use a set of services or a service group.
Here, define the names for the services for each SharePoint server instance, the IP address (or choose from a list in the case of an existing server) for the new server and the protocol it operates on.


Load Balancing Service

Basic Settings

Service Name*: sharepoint_2016

New Server [✓ Existing Server]

Server*: 10.105.157.97 (10.105.157.97)

Protocol*: HTTP

Port*: 80

Traffic Domain:

Hash ID:

Server ID: None

Cache Type*: SERVER

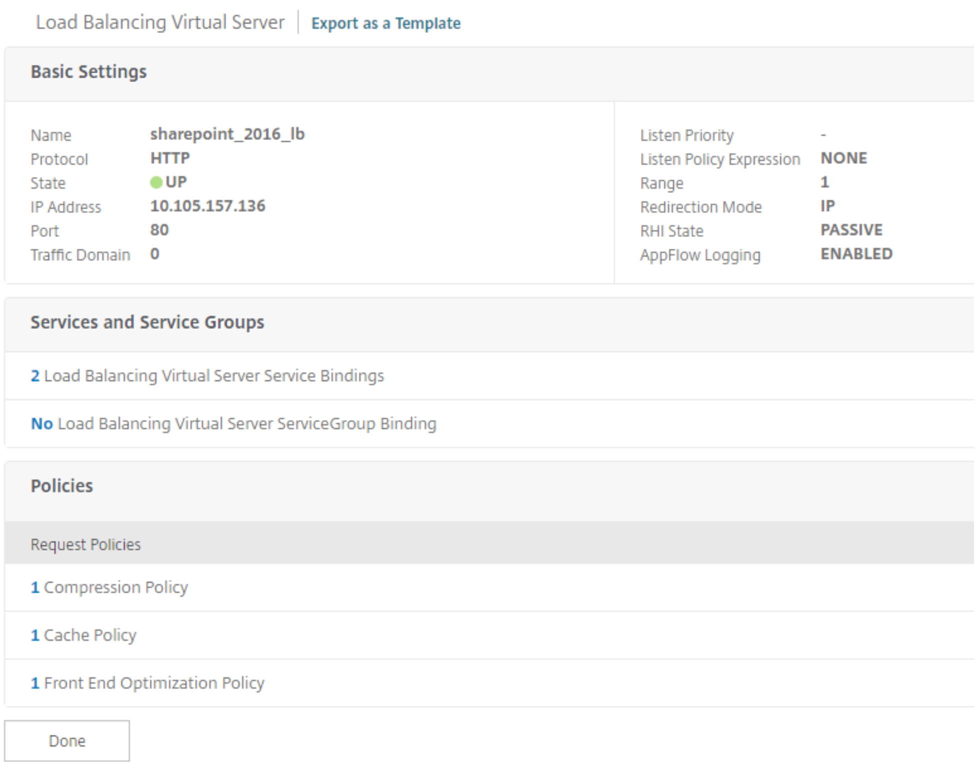
- [ ] Cacheable
- [x] Enable Service
- [x] Health Monitoring
- [x] AppFlow Logging

Number of Active Connections:

F

- Name your server instances as per their role, not with the IP address (for example, the Microsoft SharePoint 2016s can be named SharePoint1 and SharePoint2)
- As there will be multiple items linked to each application (LB vservers, services, policies among others), it is recommended that they be named appropriately for convenience. For example, the servers above can be named SharePoint1_svr, the services they bind to can be called SharePoint1_svc etc. This will make using tools such as grep with the CLI a lot easier.

You should enable Health Monitoring if you would like to have NetScaler poll the server periodically to verify its health – it is recommended that this setting should not be disabled except for diagnostic purposes. This and additional settings can be accessed by clicking on the More dropdown (as shown above). If Health Monitoring is disabled, the appliance shows the server UP at all times. Bind these service groups to the appropriate LB vservers and confirm that they have been bound correctly by checking the same in the LB vserver Basic Settings screen. Add all the SharePoint Server servers to be load balanced and bind them to the load balancing virtual server.



Basic Settings

Name	sharepoint_2016_lb	Listen Priority	-
Protocol	HTTP	Listen Policy Expression	NONE
State	● UP	Range	1
IP Address	10.105.157.136	Redirection Mode	IP
Port	80	RHI State	PASSIVE
Traffic Domain	0	AppFlow Logging	ENABLED

Services and Service Groups

2 Load Balancing Virtual Server Service Bindings

No Load Balancing Virtual Server ServiceGroup Binding

Policies

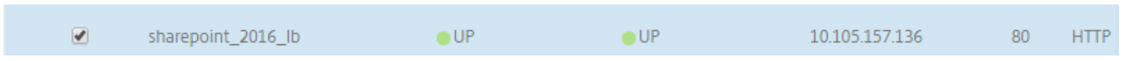
Request Policies

1 Compression Policy

1 Cache Policy

1 Front End Optimization Policy

Finally, the LB vservers created will be displayed on the configuration screen to the right in the same screen that is obtained by accessing Traffic Management>Load Balancing>Virtual Servers.

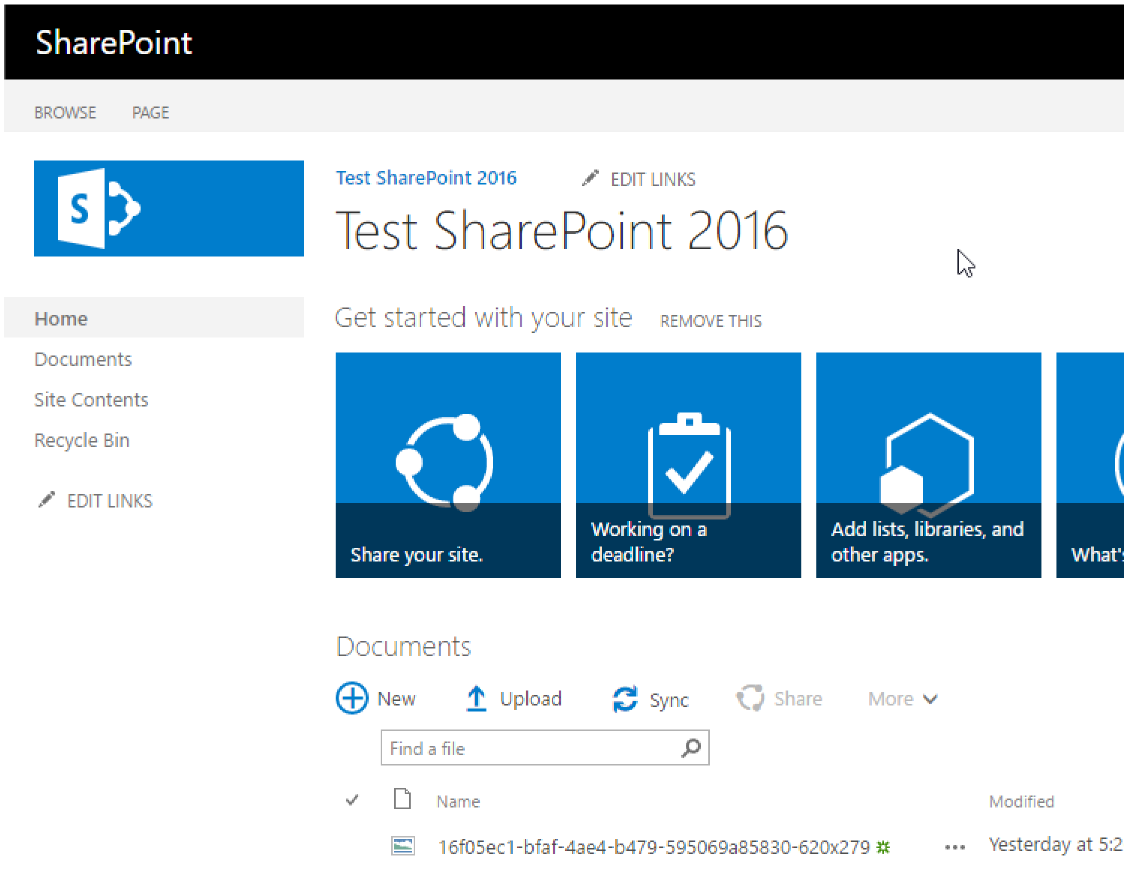


✓	sharepoint_2016_lb	● UP	● UP	10.105.157.136	80	HTTP

This completes essential load balancing configuration for SharePoint Server.

Verification

The functioning solution can be verified with a default SharePoint installation by navigating to https://<FQDN of LB vserver>/SitePages/home.aspx
This will first prompt for authentication, and if successful will show the default home screen for SharePoint.



Configuring Optimization on NetScaler

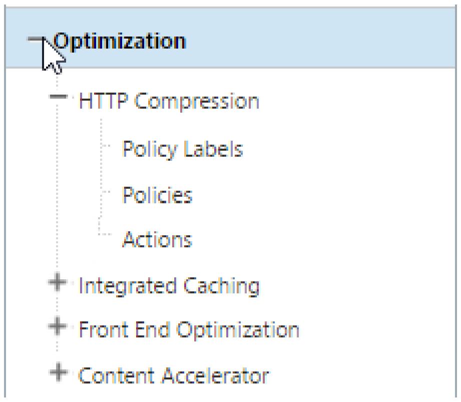
NetScaler provides a flexible, comprehensive suite of optimization capabilities that can be categorized as follows :

- HTTP Compression
- Integrated Caching
- Front End Optimization (additional optimization capabilities)

To configure HTTP Compression, Integrated Caching and Front End Optimization, expand the Optimization tab in the NetScaler GUI's left hand side navigation panel.

HTTP Compression

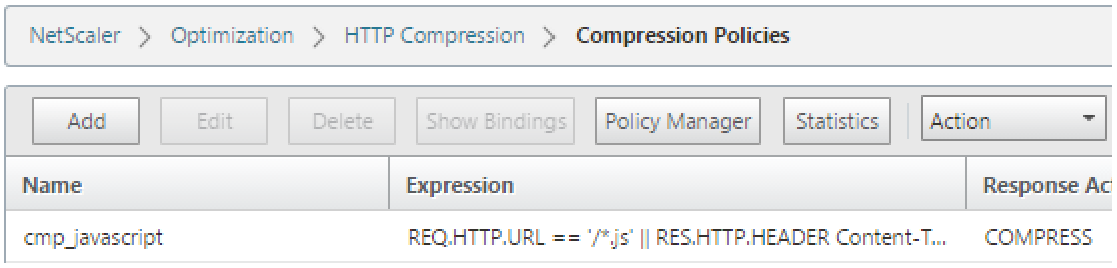
NetScaler's optimization suite is, like other NetScaler features, driven by a policy-action architecture.



To enable HTTP Compression for a particular service, you should

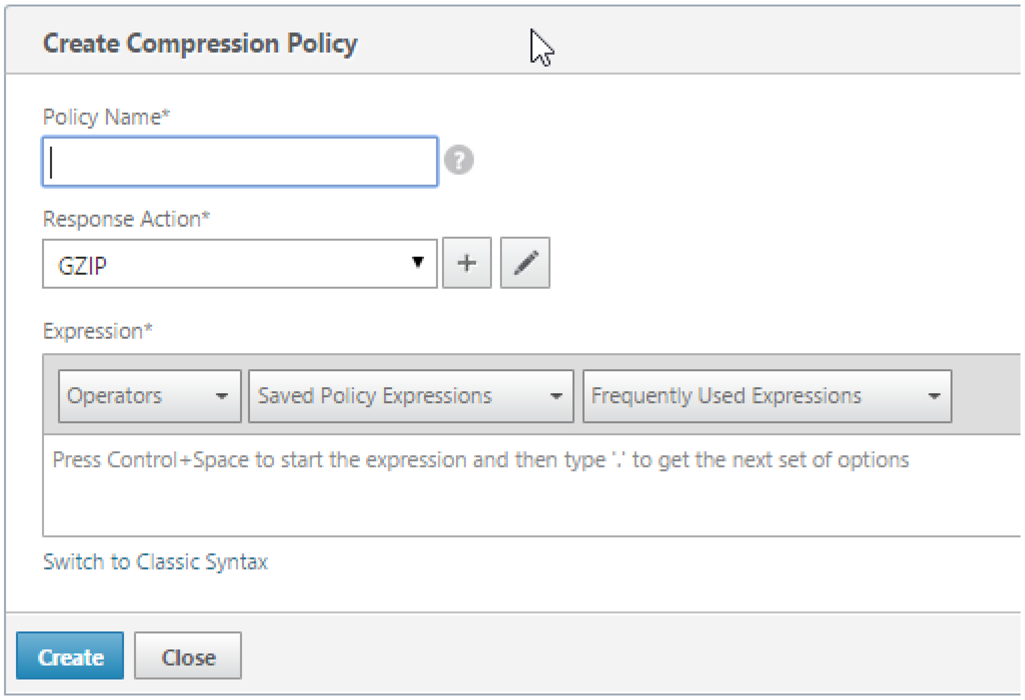
- Define the HTTP Compression Policy and Action
- Bind the same to the relevant virtual server

To define the Compression Policy and Action, click on the Policies option under HTTP Compression, shown above. This gives you the following screen -


NetScaler > Optimization > HTTP Compression > Compression Policies
Add
Edit
Delete
Show Bindings
Policy Manager
Statistics
Action

Name	Expression	Response Ac
cmp_javascript	REQ.HTTP.URL == '/*.js' || RES.HTTP.HEADER Content-T...	COMPRESS

To add a new compression policy, click on the Add button. This will give you the following screen –


Create Compression Policy

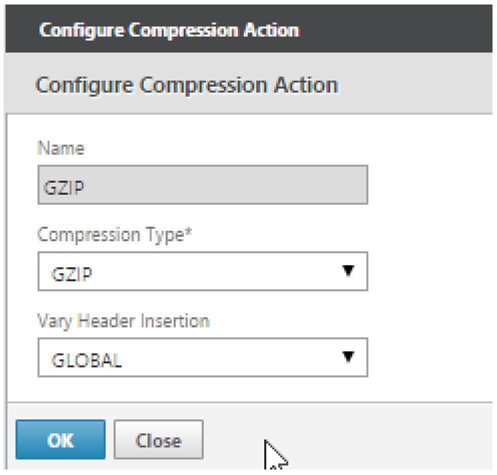
Policy Name*

Response Action*
GZIP

Expression*
Operators
Saved Policy Expressions
Frequently Used Expressions
Press Control+Space to start the expression and then type '.' to get the next set of options
Switch to Classic Syntax
Create
Close

Here, you can define a name for the policy, an Expression that defines when this policy is triggered (for example, when a particular URL is encountered. To make the policy apply to all content, use ns_true in the Expression window. For assistance here, click on the Frequently Used Expressions drop down) and the Response Action that should be taken. Here, the Actions available are COMPRESS (GZIP or DEFLATE compression, with GZIP given priority), GZIP (GZIP standard compression), DEFLATE (DEFLATE compression) and NOCOMPRESS.

Here, you have the option to either add a new Action or reconfigure the existing ones. You can Add using the + button, or edit/configure using the pencil-shaped button. Either option gives you a screen similar to the one shown below


Configure Compression Action

Configure Compression Action

Name
GZIP

Compression Type*
GZIP

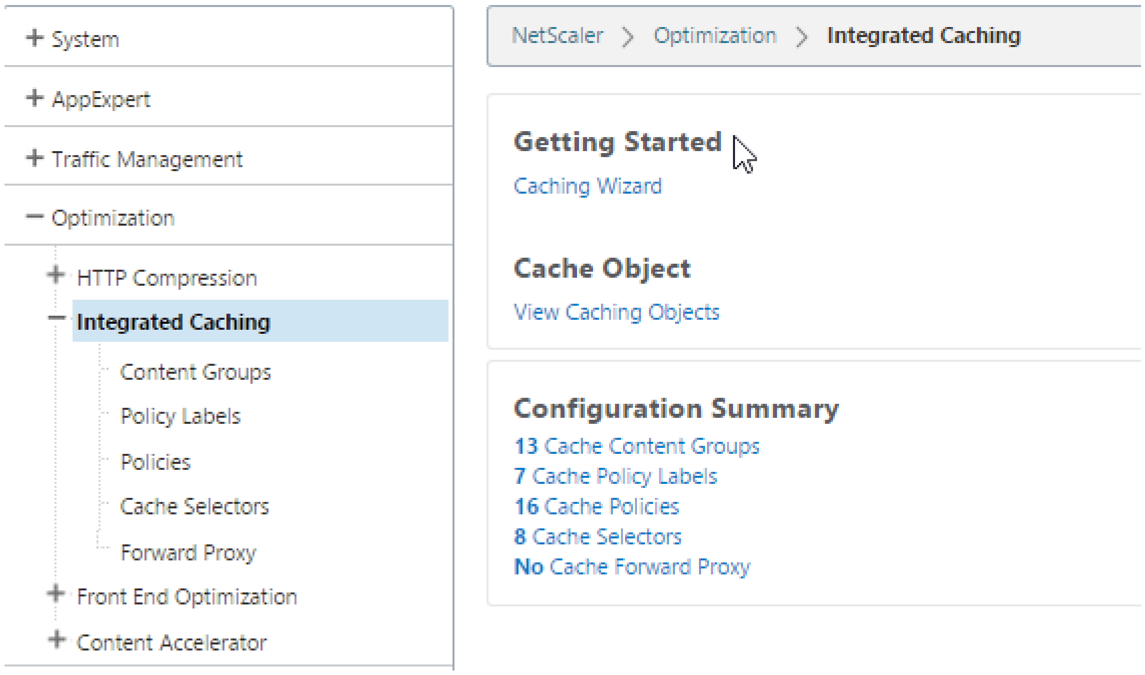
Vary Header Insertion
GLOBAL

Vary Header Insertion is an option that is relevant for caching; the value of the Vary header allows for different cache results to be returned for similar requests. For now, we will not be changing the options presented here. You can add a new action that uses a compression type of your choice.

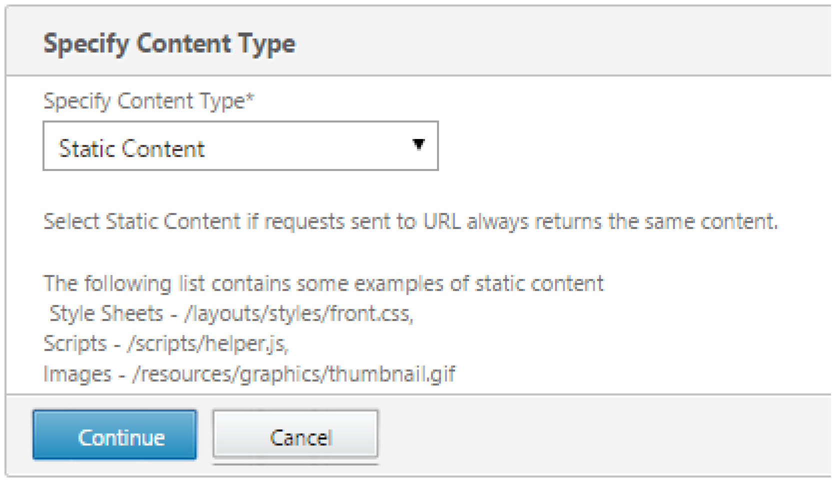
For the SharePoint deployment, the following settings have been used for HTTP compression –
Policy Name: SharePoint_Compression_Test
Response Action: GZIP (Compress/DEFLATE should work too)
Expression: ns_true

Integrated Caching

To configure caching, you can use the integrated wizard that makes configuration very straightforward. To initiate the wizard, navigate to Optimization>Integrated Caching as shown below:



Here, you can initiate the Caching Wizard under Getting Started.


Specify Content Type

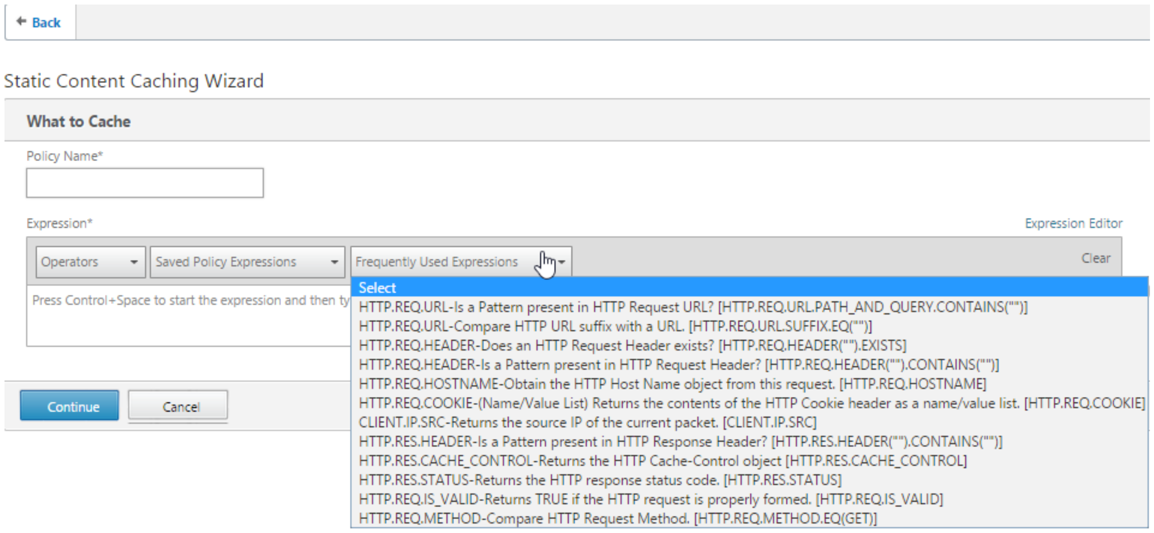
Specify Content Type*

Static Content

Select Static Content if requests sent to URL always returns the same content.

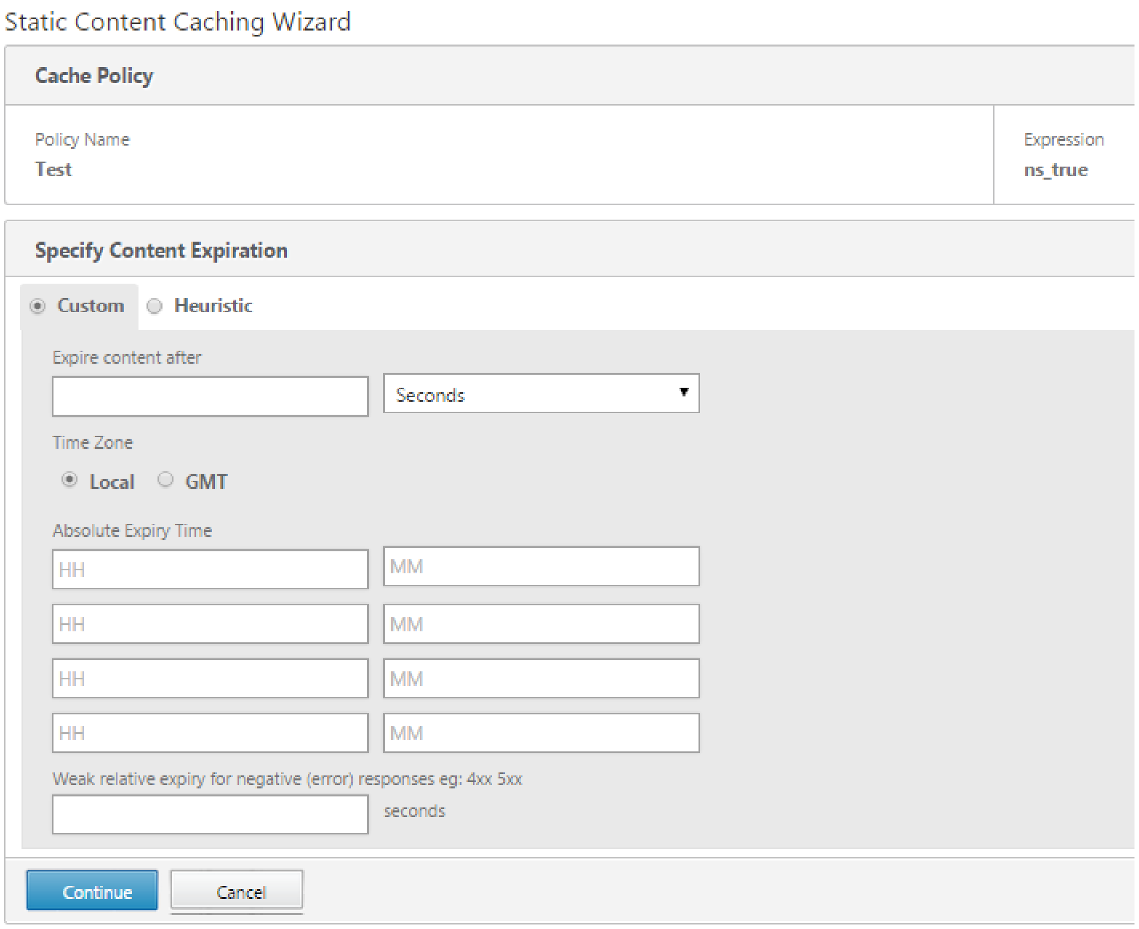
The following list contains some examples of static content
Style Sheets - /layouts/styles/front.css,
Scripts - /scripts/helper.js,
Images - /resources/graphics/thumbnail.gif

The first step requires you to specify the content type. This can be either static (examples given) or dynamic. Helpful hints are provided as shown above to help determine which type of content is relevant for you.

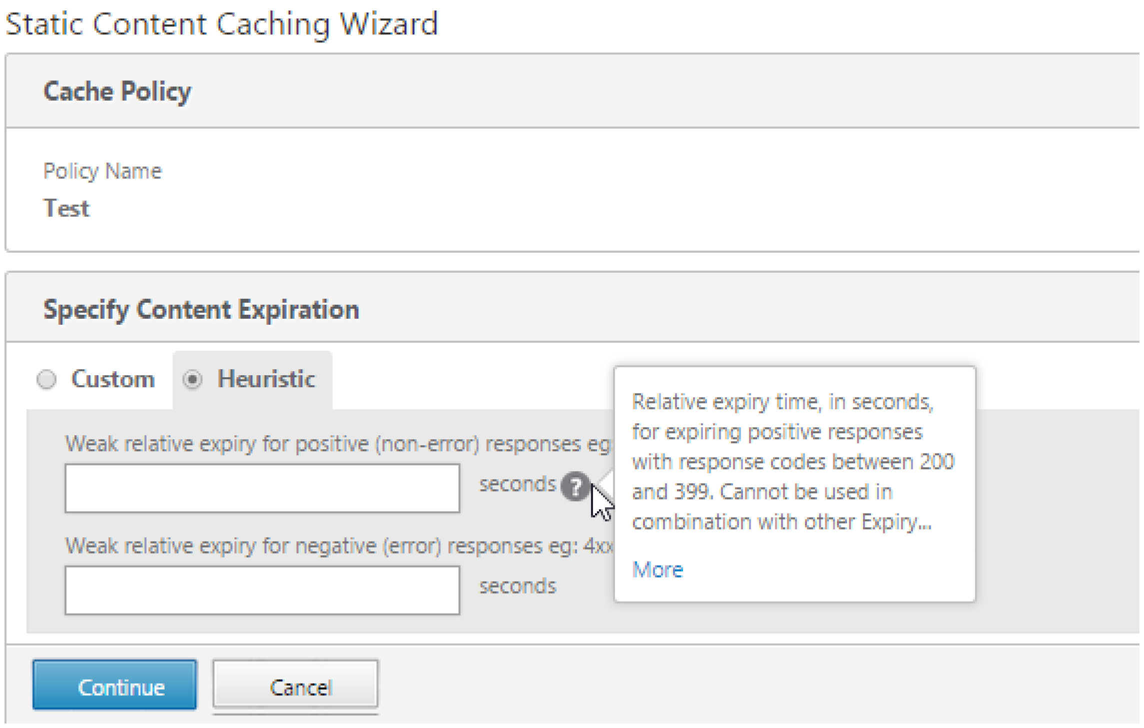

Back
Static Content Caching Wizard
What to Cache
Policy Name*
Expression*
Expression Editor
Operators
Saved Policy Expressions
Frequently Used Expressions
Clear
Press Control+Space to start the expression and then ty
Continue
Cancel

Select
HTTP.REQ.URL-Is a Pattern present in HTTP Request URL? [HTTP.REQ.URL.PATH_AND_QUERY.CONTAINS("")]
HTTP.REQ.URL-Compare HTTP URL suffix with a URL. [HTTP.REQ.URL.SUFFIX.EQ("")]
HTTP.REQ.HEADER-Does an HTTP Request Header exists? [HTTP.REQ.HEADER("").EXISTS]
HTTP.REQ.HEADER-Is a Pattern present in HTTP Request Header? [HTTP.REQ.HEADER("").CONTAINS("")]
HTTP.REQ.HOSTNAME-Obtain the HTTP Host Name object from this request. [HTTP.REQ.HOSTNAME]
HTTP.REQ.COOKIE-(Name/Value List) Returns the contents of the HTTP Cookie header as a name/value list. [HTTP.REQ.COOKIE]
CLIENT.IP.SRC-Returns the source IP of the current packet. [CLIENT.IP.SRC]
HTTP.RES.HEADER-Is a Pattern present in HTTP Response Header? [HTTP.RES.HEADER("").CONTAINS("")]
HTTP.RES.CACHE_CONTROL-Returns the HTTP Cache-Control object [HTTP.RES.CACHE_CONTROL]
HTTP.RES.STATUS-Returns the HTTP response status code. [HTTP.RES.STATUS]
HTTP.REQ.IS_VALID-Returns TRUE if the HTTP request is properly formed. [HTTP.REQ.IS_VALID]
HTTP.REQ.METHOD-Compare HTTP Request Method. [HTTP.REQ.METHOD.EQ(GET)]

The next step involves defining which content should be cached. The Frequently Used Expressions dropdown helps you define the correct expression; however, if you want the caching policy to run for all content, you can use *ns_true* as the expression as shown below:


Static Content Caching Wizard
Cache Policy
Specify Content Expiration
Expire content after
Time Zone
Absolute Expiry Time
HH
MM
Weak relative expiry for negative (error) responses eg: 4xx 5xx
seconds
Continue
Cancel

Policy Name: Test
Expression: ns_true
[✓ Custom] Heuristic
Seconds
[✓ Local] GMT


Static Content Caching Wizard

Cache Policy

Policy Name
Test

Specify Content Expiration

Custom [✓ Heuristic]

Weak relative expiry for positive (non-error) responses eg

seconds

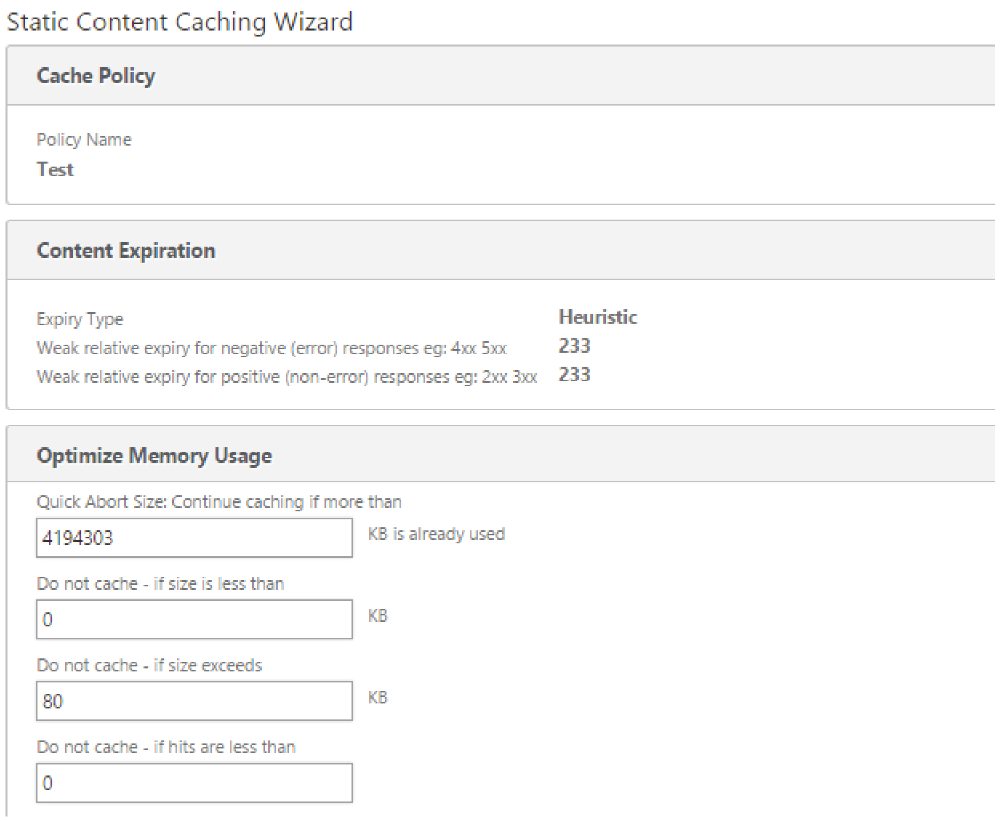
Weak relative expiry for negative (error) responses eg: 4xx

seconds

Relative expiry time, in seconds, for expiring positive responses with response codes between 200 and 399. Cannot be used in combination with other Expiry...

More

The next step involves definition of the caching space to be used on the NetScaler and the minimum size of objects to be cached.


Static Content Caching Wizard

Cache Policy

Policy Name
Test

Content Expiration

Expiry Type	Heuristic
Weak relative expiry for negative (error) responses eg: 4xx 5xx	233
Weak relative expiry for positive (non-error) responses eg: 2xx 3xx	233

Optimize Memory Usage

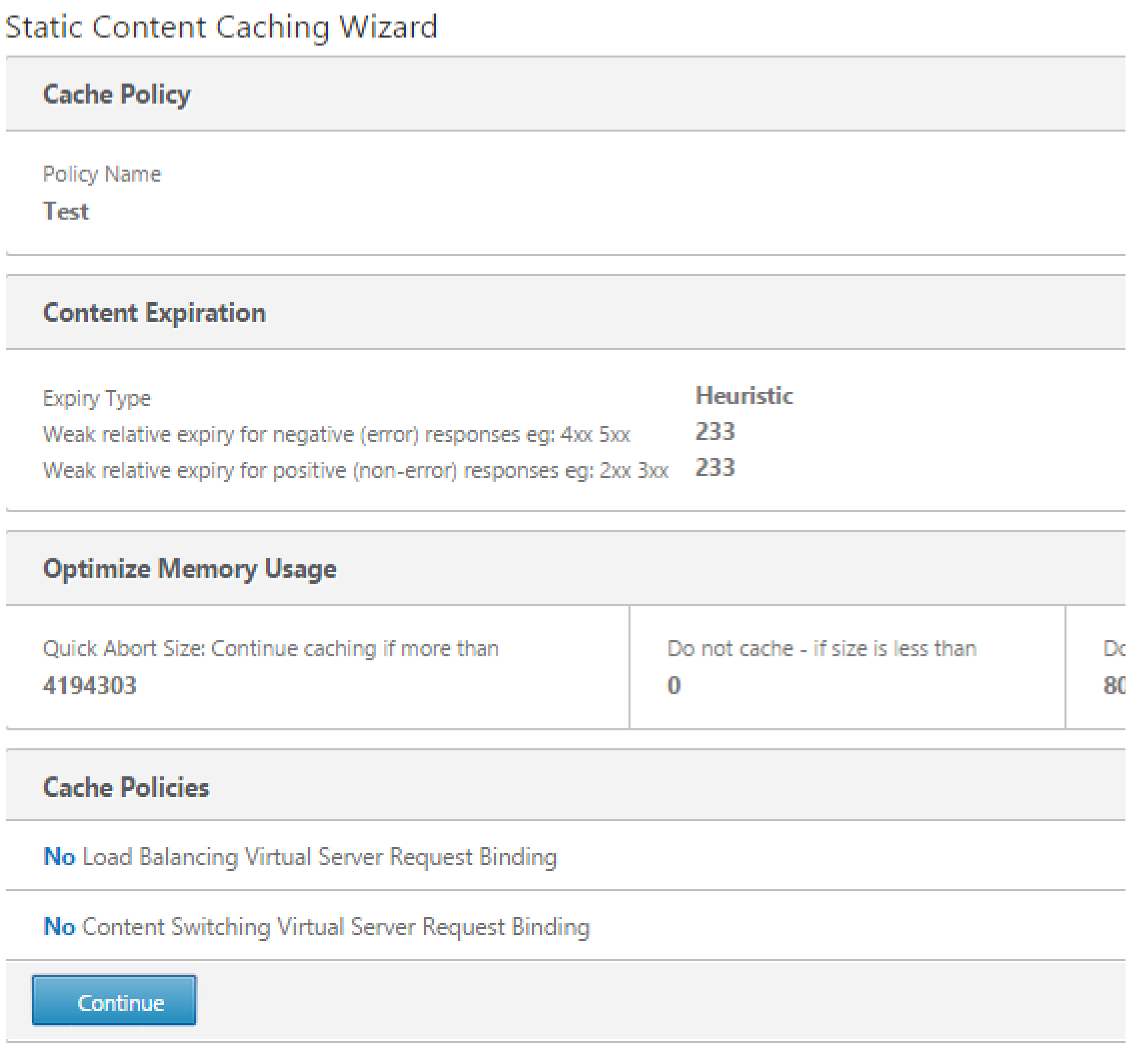
Quick Abort Size: Continue caching if more than
4194303 KB is already used

Do not cache - if size is less than
0 KB

Do not cache - if size exceeds
80 KB

Do not cache - if hits are less than
0

Finally, the cache policy should be bound to the relevant vserver.



Static Content Caching Wizard

Cache Policy

Policy Name
Test

Content Expiration

Expiry Type	Heuristic
Weak relative expiry for negative (error) responses eg: 4xx 5xx	233
Weak relative expiry for positive (non-error) responses eg: 2xx 3xx	233

Optimize Memory Usage

Quick Abort Size: Continue caching if more than	Do not cache - if size is less than	Dc
4194303	0	80

Cache Policies

No Load Balancing Virtual Server Request Binding

No Content Switching Virtual Server Request Binding

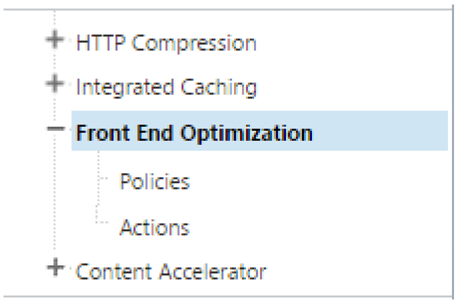
These definitions can be made under Cache Policies as shown in the screenshot above.

For the SharePoint deployment, the following settings have been used for caching –
Policy Name: SharePoint_Cache_Test
Actions: CACHE
Cache Content Group: Test
Undefined-Result Action: -Global-undefined-result-action (or NOCACHE/RESET)
Expression: *ns_true*

Cache Content Group:
Name: Test
Type: HTTP
Expiry Method: Heuristic (Recommended)/Custom (if specific settings are required)
Default Expiry Times: As per requirement; set to 233 for test deployment.
Parameterization: Leave values as is (unless Cache selectors are in use; not configured for our test setup)
Memory: Define values as per your system limits
Others: Use default settings. All settings have context-sensitive help available if modification is required.

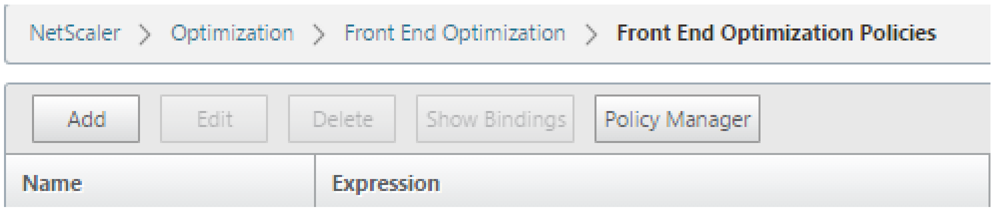
Front End Optimization

The front end optimization feature set makes NetScaler an extremely capable optimization device by implementing enhanced optimization routines for specific front end entities such as images, JavaScript etc. These features provide improved optimization performance than that achieved by compression and caching.

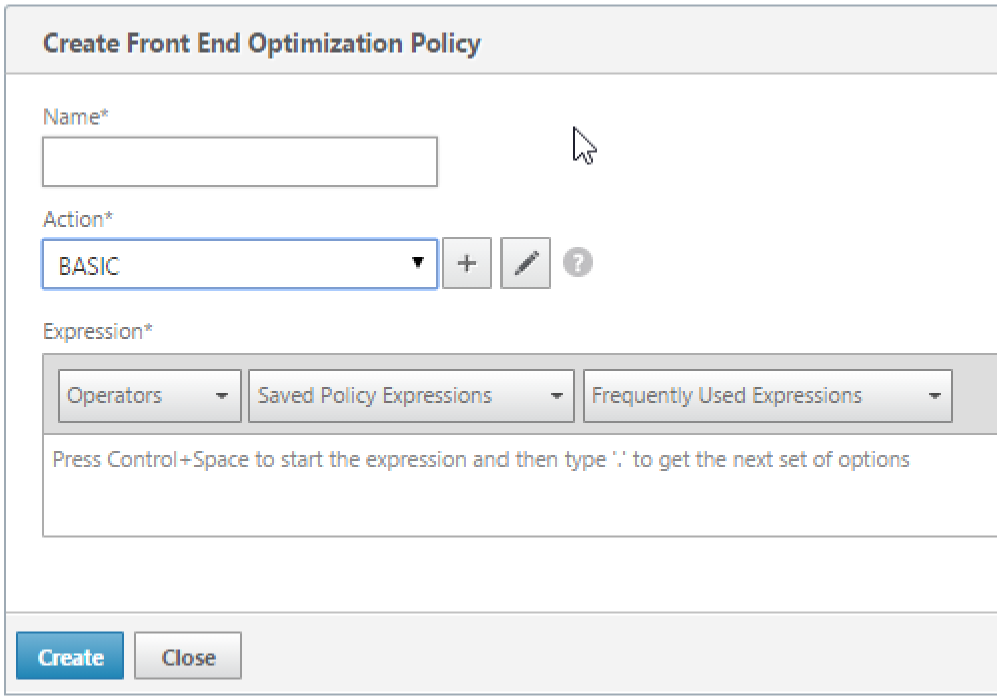


Front End Optimization capabilities can be activated by navigating to Optimization>Front End Optimization. As with all NetScaler features, these are implemented using a policy-action mechanism.

To add a new policy, navigate to Optimization>Front End Optimization and then, click on Policies. To add a new policy, click on Add in the section displayed to the right of the navigation menu.

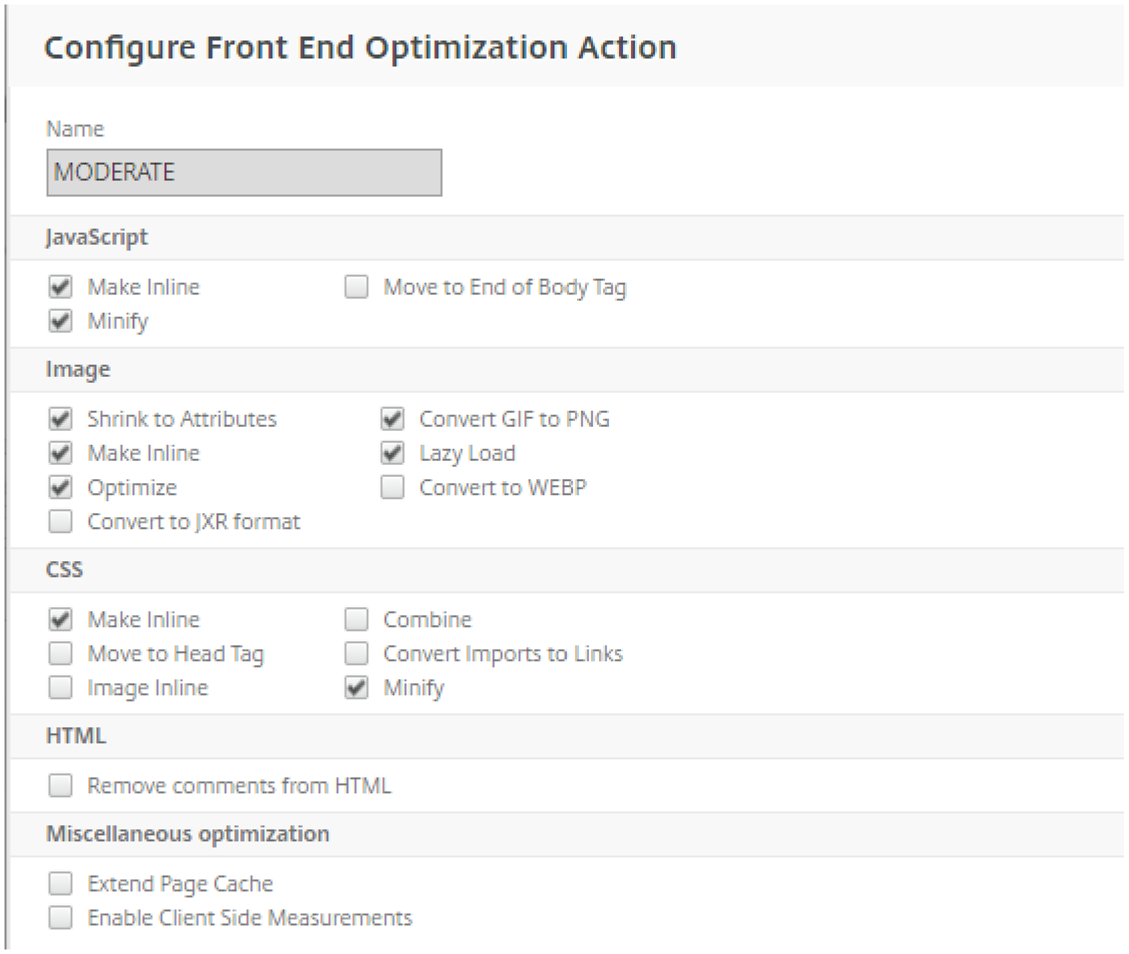

NetScaler > Optimization > Front End Optimization > Front End Optimization Policies
Add
Edit
Delete
Show Bindings
Policy Manager
Name
Expression

This will give you the following screen for definition of a new FEO policy.


Create Front End Optimization Policy
Name*
Action*
BASIC
Expression*
Operators
Saved Policy Expressions
Frequently Used Expressions
Press Control+Space to start the expression and then type '.' to get the next set of options
Create
Close

The Expression here works much on the same lines as expressed for the earlier features; the Frequently Used Expressions drop down can be used for assistance. There are certain predefined actions that can be assigned here, all of which have different configurations for the same settings; you can also either edit or create a custom action, which can be done using the plus or pencil buttons next to the Action name. The Plus icon enables the setup of a custom profile.

Upon clicking either of these buttons, the following screen (or a similar one) is observed:


Configure Front End Optimization Action

Name: MODERATE

JavaScript
- [✓] Make Inline
- [ ] Move to End of Body Tag
- [✓] Minify

Image
- [✓] Shrink to Attributes
- [✓] Convert GIF to PNG
- [✓] Make Inline
- [✓] Lazy Load
- [✓] Optimize
- [ ] Convert to WEBP
- [ ] Convert to JXR format

CSS
- [✓] Make Inline
- [ ] Combine
- [ ] Move to Head Tag
- [ ] Convert Imports to Links
- [ ] Image Inline
- [✓] Minify

HTML
- [ ] Remove comments from HTML

Miscellaneous optimization
- [ ] Extend Page Cache
- [ ] Enable Client Side Measurements

This screen presents all the various front end optimization options available; NetScaler can help to optimize web traffic with JavaScript, Image, CSS (Cascading Style Sheets), HTML and Miscellaneous Optimization. This last section also allows for domain sharding, which splits resources across subdomains to improve optimization and page load times.

For this deployment, the recommended FEO policy setting is Moderate; this default setting provides a good level of optimization while not affecting the performance of the SharePoint setup. The lab tests show an improvement in page load times of approximately 15%, along with a compression level of 3.25x, resulting in over 70% data transfer reduction..

Optimization settings for the Microsoft SharePoint deployment:

```
Optimization Policy Name: SharePoint_Optimization_Test
Optimization Action: MODERATE (Preconfigured)
Expression: HTTP.REQ.HEADER("Accept").CONTAINS("html")

Alternate Configuration (Custom Policy)
Optimization Policy Name: SharePoint_Optimization_TestCustom
Optimization Action: samplefeo
Expression: HTTP.REQ.HEADER("Accept").CONTAINS("html")

SharePoint_Optimization_TestCustom Configuration:
Enabled Settings: JavaScript/Make Inline, JavaScript/Move to End of Body Tag, JavaScript/Minify, Image/Op-
timize, Image/Lazy Load, Image/Shrink to Attributes, Image/Optimize, Image/Convert to JXR format, Image/
Convert GIF to PNG, CSS/Make Inline, CSS/Move to Head Tag, CSS/Minify, CSS/Image Inline, CSS/Combine,
CSS/Convert Imports to Links, HTML/Remove Comments from HTML
```

Conclusion

NetScaler enables highly available Microsoft SharePoint 2016 deployments with its load balancing capabilities. With NetScaler, enterprises can enable a host of additional capabilities including but not limited to authentication offload, end point analysis checks, selective server access, URL rewrites, compression, caching, front end optimizations and much more.

With NetScaler, enterprises can not only enable high availability for their SharePoint environments, but also extend capabilities for security and optimized access. The policy engine used by NetScaler enables enterprises to deploy any specific use cases that they may require, making the NetScaler solution a flexible and robust one that can meet all enterprise requirements.