Deploying Microsoft Dynamics CRM 2015 with NetScaler

Deployment Guide

This deployment guide focuses on defining the deployment process for Microsoft Dynamics CRM with Citrix NetScaler. It includes information on setting up basic deployment, authentication and optimization.

CITRIX®

Table of Contents

NetScaler is a world-class application delivery controller (ADC) with the proven ability to load balance, accelerate, optimize and secure enterprise applications. Microsoft Dynamics CRM is a customer relationship management software package developed by Microsoft. Out of the box, the product focuses mainly on Sales, Marketing, and Service (help desk) sectors, but it can also function as an XRM platform and customized using the .NET framework. It is part of the Microsoft Dynamics family of business applications.

Introduction

This guide defines the process for deploying Microsoft Dynamics CRM with NetScaler. NetScaler is a world-class application delivery controller (ADC) with the proven ability to load balance, accelerate, optimize and secure enterprise applications. Deploying Microsoft Dynamics CRM with NetScaler brings the application acceleration and optimization capabilities of NetScaler, improving transaction speeds, making operations quicker and providing a faster user experience.

Microsoft Dynamics CRM is a customer relationship management software package. Out of the box, the product focuses mainly on sales, marketing, and service (help desk) functions. It is part of the Microsoft Dynamics family of business applications.

Dynamics CRM is a client-server application. Like Microsoft SharePoint, it is primarily an IIS-based web application that also supports extensive web services interfaces. Clients access Dynamics CRM by using a browser or a thick client plug-in to Microsoft Outlook. Besides Internet Explorer, the solution (as of Dynamics CRM 2011 update rollup 12) fully supports Chrome and Firefox browsers. It is available as a cloud or on-premises solution, or a hybrid.

For the purposes of this guide, our basic Dynamics CRM deployment consists of the following server roles:

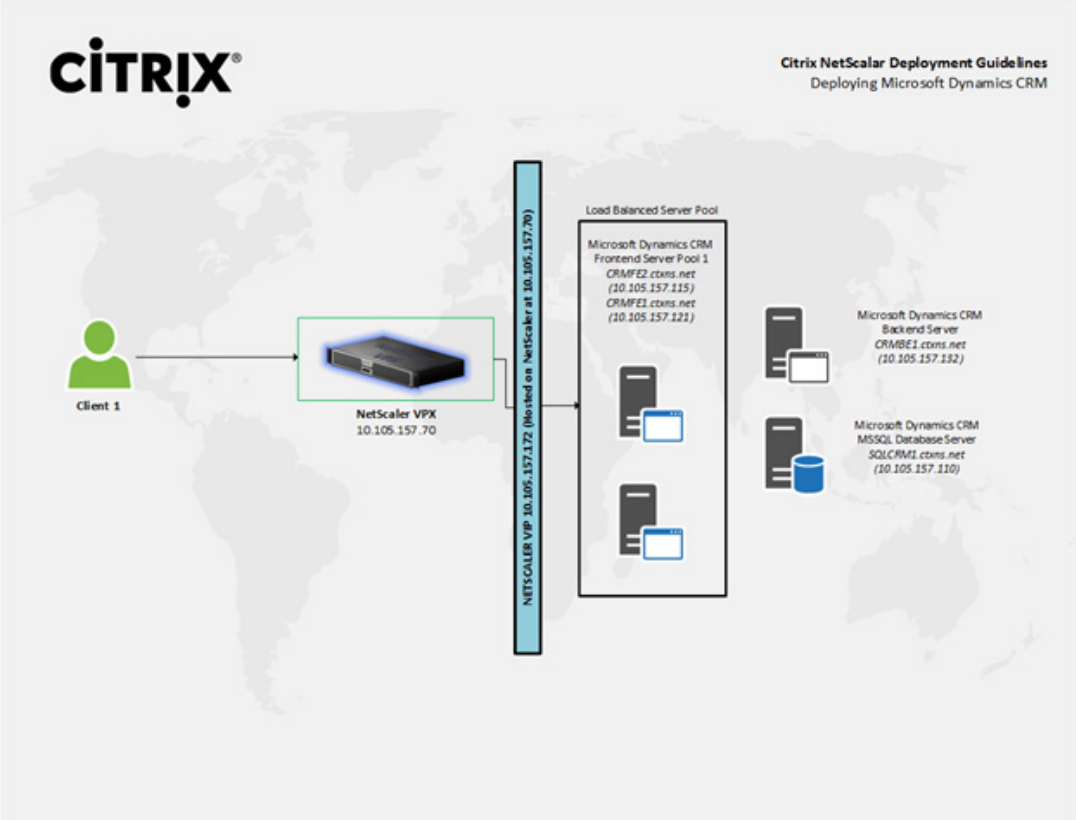
MS Dynamics CRM frontend servers – these are the servers that clients connect to for accessing information from Dynamics CRM

MS Dynamics CRM backend servers – these application servers run the backend processes for the Dynamics CRM application. Users do not generally have direct access to these servers, but administrators access them for backend configuration.

SQL Server (for hosting the Dynamics CRM database) – The SQL server hosts the main Dynamics CRM database; this is linked to the parent organization.

A detailed description of the individual functions performed by the various server roles is available at https://technet.microsoft.com/en-us/library/hh699655.aspx#BKMK_AvailableGroup

Configuration Details



Product	Version
Microsoft Dynamics CRM	2015
NetScaler VPX	10.5 (Enterprise License)

NetScaler features to be enabled

The essential NetScaler features that need to be enabled are explained below. Please ensure these features are enabled in the NetScaler system.

- Load balancing
- HTTP Compression
- Front End Optimization
- Integrated Caching
- AAA

Here is a quick explanation of how these features work.

Load Balancing

NetScaler load balancing evenly distributes requests to backend servers. Multiple algorithms (such as LEASTCONNECTION, ROUNDROBIN, etc.) are supported to provide efficient load balancing logic for every application server.

HTTP Compression

Compression of HTTP traffic using standard GZIP/DEFLATE compression methods.

Front End Optimization (FEO)

Advanced optimization feature, FEO enables NetScaler to significantly accelerate web content with various acceleration methods such as image compression etc.

Integrated Caching

Content caching allows NetScaler to serve frequently used content without requiring round trips to the source webserver.

AAA

The AAA feature set controls authentication, authorization and auditing policies for NetScaler. These policies include definition and management of various authentication schemas. NetScaler supports a wide range of authentication protocols and a strong, policy-driven application firewall capability.

Several additional features can help improve the enterprise user experience. Rewrite, responder, SSL Offloading and other features can help improve the user experience with this deployment. However, the use case described here can be deployed using the five features described above; this guide will not describe the benefits that can be achieved with these additional features. This guide assumes that you have your CRM system setup for standard Windows/LDAP authentication.

Solution description

Quick Configuration Table

Use this table if you are comfortable with using the NetScaler GUI/CLI and would like to have a working Microsoft Dynamics CRM load balanced environment in place quickly.

<table>
<tr><th>Configuration Item</th><th>Details</th></tr>
<tr><td>Load Balancing
(Traffic Management>Load Balancing>Virtual Servers in the GUI)</td><td>Virtual Servers: LB_MSCRM, LB_MSCRM_HTTP (Suggested Names)
<table>
<tr><th>LB_MSCRM</th><th>LB_MSCRM_HTTP</th></tr>
<tr><td>Protocol: HTTPS
Port: 443 (or alternate as per your configuration)
Load Balancing Method: Roundrobin/LeastConnection
Services Bound:
MSCRM_FE1
MSCRM_FE2
Compression Policy: MSCRM_Compression_Test
Cache Policy: MSCRM_Cache_Test
FEO Policy: MSCRM_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 LB_MSCRM SSL <IP address for vserver> 443 -persistenceType NONE -cltTimeout 180 -authn401 ON -authnVsName MSCRM_AAA -authnProfile AuthProfile_MSCRM</td><td>Protocol: HTTP
Port: 80 (or alternate as per your configuration)
Load Balancing Method: Roundrobin/LeastConnection
Services Bound:
MSCRM_FE1_HTTP
MSCRM_FE2_HTTP
Compression Policy: MSCRM_Compression_Test
Cache Policy: MSCRM_Cache_Test
FEO Policy: MSCRM_Optimization_Test
CLI Commands:
add lb vserver LB_MSCRM_HTTP HTTP <IP address for vserver> 80 -persistenceType NONE -lbMethod ROUNDROBIN -cltTimeout 180 -downStateFlush DISABLED</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Configuration Item</th><th>Details</th></tr>
<tr><td>Service configuration
(Traffic Management>Load Balancing>Services)</td><td>
<table>
<tr><th>MSCRM_FE_1</th><th>MSCRM_FE_2</th><th>MSCRM_FE1_HTTP</th><th>MSCRM_FE2_HTTP</th></tr>
<tr><td>Protocol: HTTPS
Port: 443 (or alternate as per your configuration)
IP: IP address of 1st CRM Front End server</td><td>Protocol: HTTPS
Port: 443 (or alternate as per your configuration)
IP: IP address of 2nd CRM Front End server</td><td>Protocol: HTTP
Port: 80 (or alternate as per your configuration)
IP: IP address of 1st CRM Front End server</td><td>Protocol: HTTP
Port: 80 (or alternate as per your configuration)
IP: IP address of 2nd CRM Front End server</td></tr>
</table>
CLI Commands:
add service MSCRM_FE_1 <IP address for 1st CRM front end server> SSL 443 -gslb NONE -maxClient 0 -maxReq 0 -cip DISABLED -usip NO -useproxyport NO -sp OFF -cltTimeout 180 -svrTimeout 360 -CKA NO -TCPB NO -CMP NO
set ssl service MSCRM_FE_1 -tls11 DISABLED -tls12 DISABLED

add service MSCRM_FE_2 <IP address for 2nd CRM front end server> SSL 443 -gslb NONE -maxClient 0 -maxReq 0 -cip DISABLED -usip NO -useproxyport NO -sp ON -cltTimeout 180 -svrTimeout 360 -CKA NO -TCPB NO -CMP YES

set ssl service MSCRM_FE_2 -tls11 DISABLED -tls12 DISABLED

add service MSCRM_FE1_HTTP <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 -cltTimeout 180 -svrTimeout 360 -CKA NO -TCPB NO -CMP YES

add service MSCRM_FE2_HTTP <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
</td></tr>
<tr><td>Compression Policy Definition
(Optimization>Integrated Caching>Policies)</td><td>
Policy name: MSCRM_Compression_Test
Response action: COMPRESS (GZIP/DEFLATE should work too)
Expression: ns_true

CLI Commands:
add cmp policy MSCRM_Compression_Test -rule ns_true -resAction GZIP
bind lb vserver LB_MSCRM -policyName MSCRM_Compression_Test -priority 100
bind lb vserver LB_MSCRM_HTTP -policyName MSCRM_Compression_Test -priority 100
</td></tr>
<tr><td>Cache Policy
(Optimization>Integrated Caching>Policies)</td><td>
Policy name: MSCRM_Caching_Test
Actions: CACHE
Cache content group: Test
Undefined result action: Global-undefined-result-action (or NOCACHE/RESET)
Expression: SYS.EVAL_CLASSIC_EXPR(“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.
</td></tr>
</table>

Configuration Item	Details
FEO policy (Optimization>Front end Optimization>Policies)	**Optimization:** 57.4% reduction in load time, 10.15% reduction in data transfer, 13.5% reduction in number of requests. **Optimization policy name:** MSCRM_Optimization_Test **Optimization action:** MODERATE (Preconfigured) **Expression:** HTTP.REQ.HEADER("Accept").CONTAINS("html") **Alternate configuration (custom policy)** **Optimization policy name:** MSCRM_Optimization_TestCustom **Optimization action:** MS_CRM_custom **Expression:** HTTP.REQ.HEADER("Accept").CONTAINS("html") **MSCRM_Optimization_TestCustom Configuration:** **Enabled settings:** JavaScript/Make Inline, JavaScript/Move to End of Body Tag, JavaScript/Minify, Image/Optimize, Image/Lazy Load, CSS/Make Inline, CSS/Move to Head Tag, CSS/Minify, HTML/Remove Comments from HTML

To deploy Dynamics CRM with NetScaler, the following steps must be completed:

1. Configure load balancing
2. Configure authentication (if necessary)
3. Configure optimization, caching and compression policies

Configure load balancing

Configuration of load balancing consists of defining load balancing virtual servers (vservers), as well as services that are linked to them and represent individual application servers, which are then load balanced by the vservers.

Step 1 - Define the load balancing virtual servers

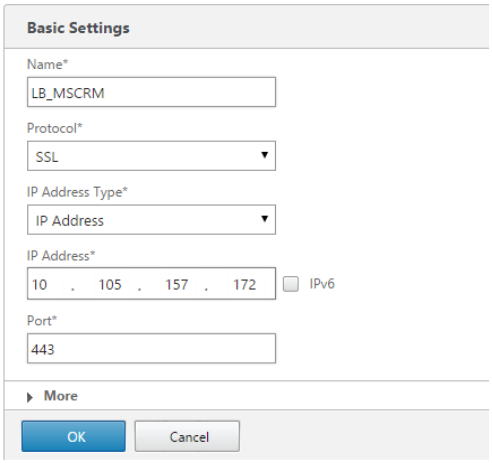
Log into the NetScaler GUI. On the Configuration tab, move to Traffic Management>Load Balancing>Virtual Servers.

For this deployment exercise, we are load balancing two frontend Dynamics CRM servers, which are connected to a single backend server. To demonstrate functionality over both HTTP and HTTPS, we have created two load balancing vservers: LB_MSCRM and LB_MSCRM_HTTP.

Step 2 - Configure the load balancing vservers

When defining a new vserver, you will be presented with the settings screen. Here, enter the settings appropriate for your setup.

Load Balancing Virtual Server



- Note that the protocol here is presented as SSL. To bring the server up, you would be required to provide a valid server certificate.
- Set the IP address type to a valid IP address. This is the address that will be used to access the CRM deployment front end; this IP should be linked to the fully qualified domain name (FQDN) if it is in use by the CRM server.
- Leave the other settings as is.

The screenshot shown earlier shows the settings for the SSL load balancing vserver. When adding the HTTP vserver, the protocol should be set to HTTP and the port to 80 (or whichever port the vserver should serve on). Leave the IP address as is so the same FQDN can be used for serving both SSL and HTTP requests.

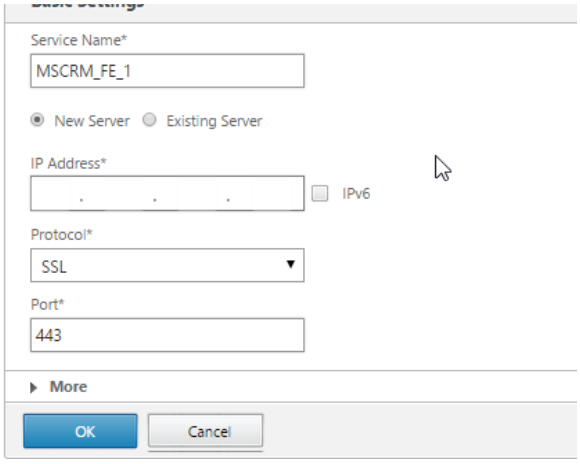
After clicking OK, you will see the Basic Settings screen for the load balancing vserver. Here, you may change settings such as the session persistence,authentication and load balancing methods, or leave them at the default setting.

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 sub-section and then, click the Add button.

Step 3 – Define load balancing vserver service binding

Every load balancing service is linked to a server; either a new server or an existing server already defined in the Servers subsection under Load Balancing.

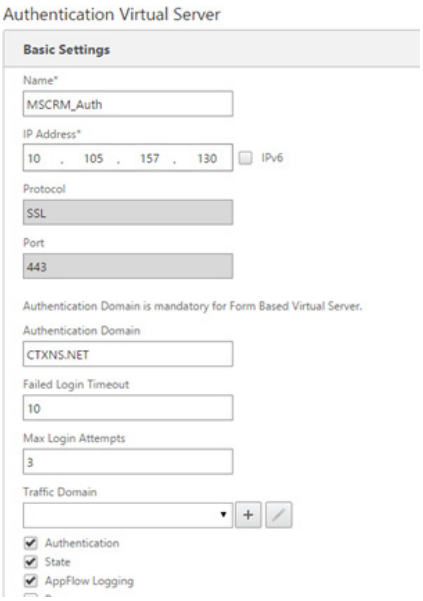
Here, define the name for the service (MSCRM_FE1, MSCRM_FE2, MSCRM_FE1_HTTP and MSCRM_FE2_HTTP for this deployment), the IP address (or choose from a list in the case of an existing server) for the new server and the protocol it operates on. For this deployment, the IPs correspond to 10.105.157.115 for the first server (FE1) and 10.105.157.121 for the second one (FE2).



You must enable the Health Monitoring checkbox if you would like to have NetScaler poll the server periodically to verify its health. If Health Monitoring is disabled, the appliance shows the server as up at all times, even if it is down. The AppFlow logging option enables monitoring of the service; it is recommended to enable this option as well.

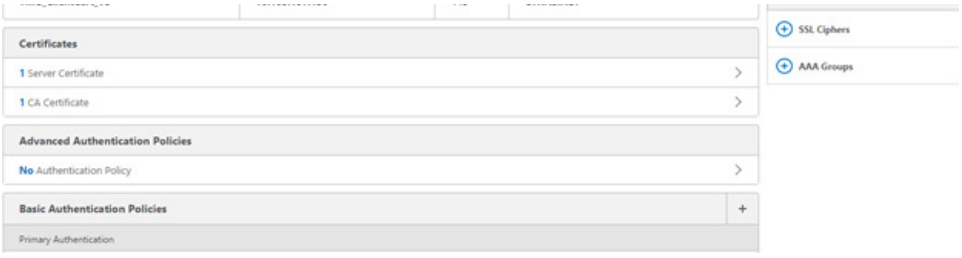
Finally, the load balancing vservers created will be displayed on the configuration screen to the right in the same screen that is obtained by accessing Traffic Management>Load Balancing>Virtual Servers.

Configuring Authentication



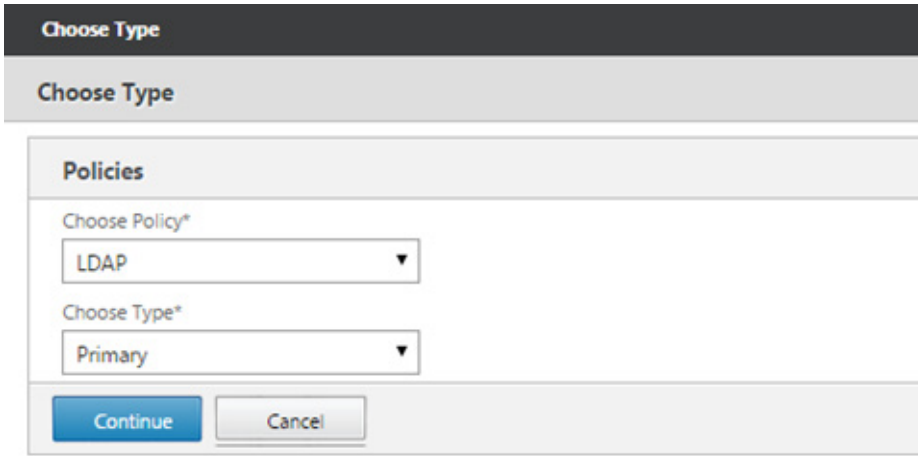
The screen above shows the configuration window for a new AAA vserver. This screen can be viewed by navigating to Security>AAA – Application Traffic>Virtual Servers and then clicking the Add button in the panel on the right-hand side of the screen. Provide a name for the vserver, an available IP, and the protocol and port, as well as the authentication domain (this is mandatory if you plan to have form-based authentication). Context-sensitive help is available for each setting. Note: The AAA vserver can only be defined over SSL; therefore, you will be required to add a server certificate to the AAA vserver. This can be a self-signed or purchased certificate, depending upon the security level required. A self-signed certificate may be easier to obtain, but will not be verifiable, and most browsers will present errors with such a certificate. A purchased certificate allows verification by a certified, recognized certificate authority.

After adding the AAA vserver, you are taken to the Basic Settings page for the vserver.



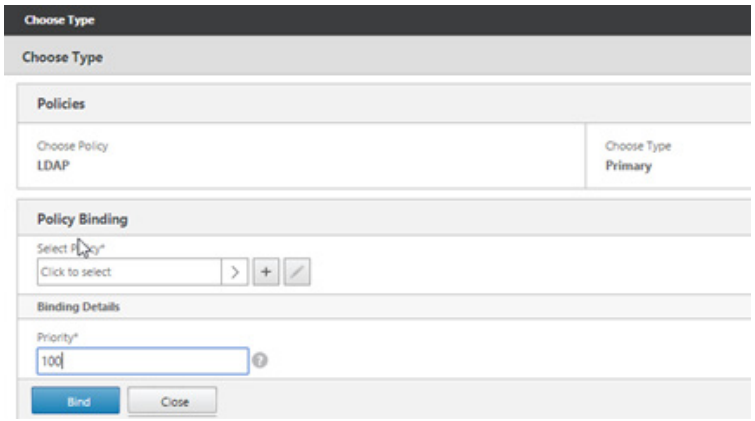
To add the authentication policy (LDAP for our configuration), you must bind it to the server by clicking the + icon next to the Basic Authentication Policies section header. If the header is not present, click the Authentication Tab in the list at the extreme right of the screen.

After you click the + icon, the following screen is shown:

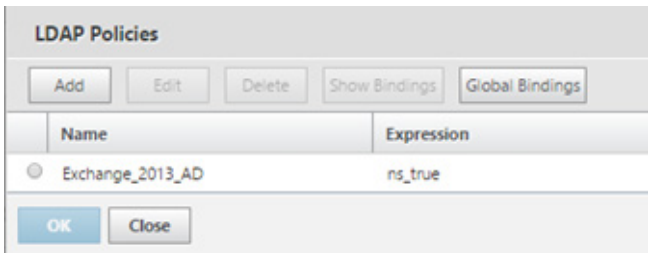


Here, you can select the type of authentication policy you would like to add. The Type parameter defines whether the authentication policy is primary or secondary, which is useful during dual-factor authentication scenarios. As we are only defining a single authentication policy, we will keep the Type parameter set as Primary.

Upon clicking Continue, you will be presented with the following screen:

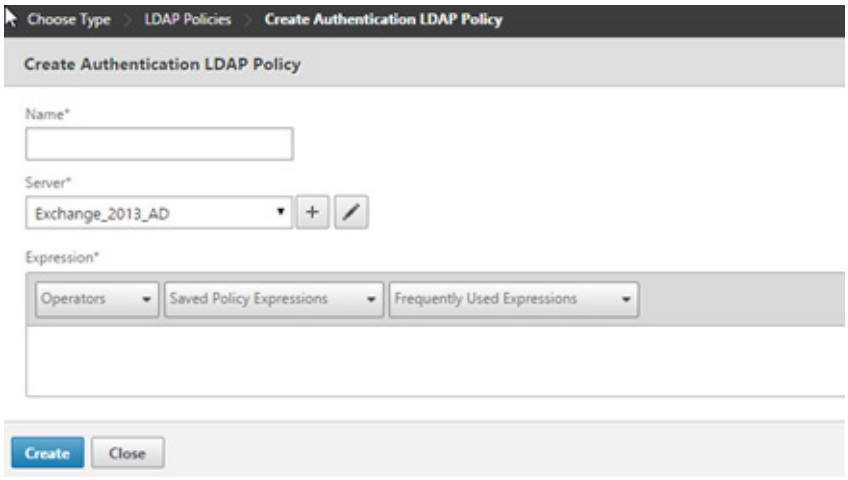


Here, you can either add a new LDAP policy or select a pre-existing policy.
When you click on Policy binding>Select Policy>Click to Select, the following screen is displayed:

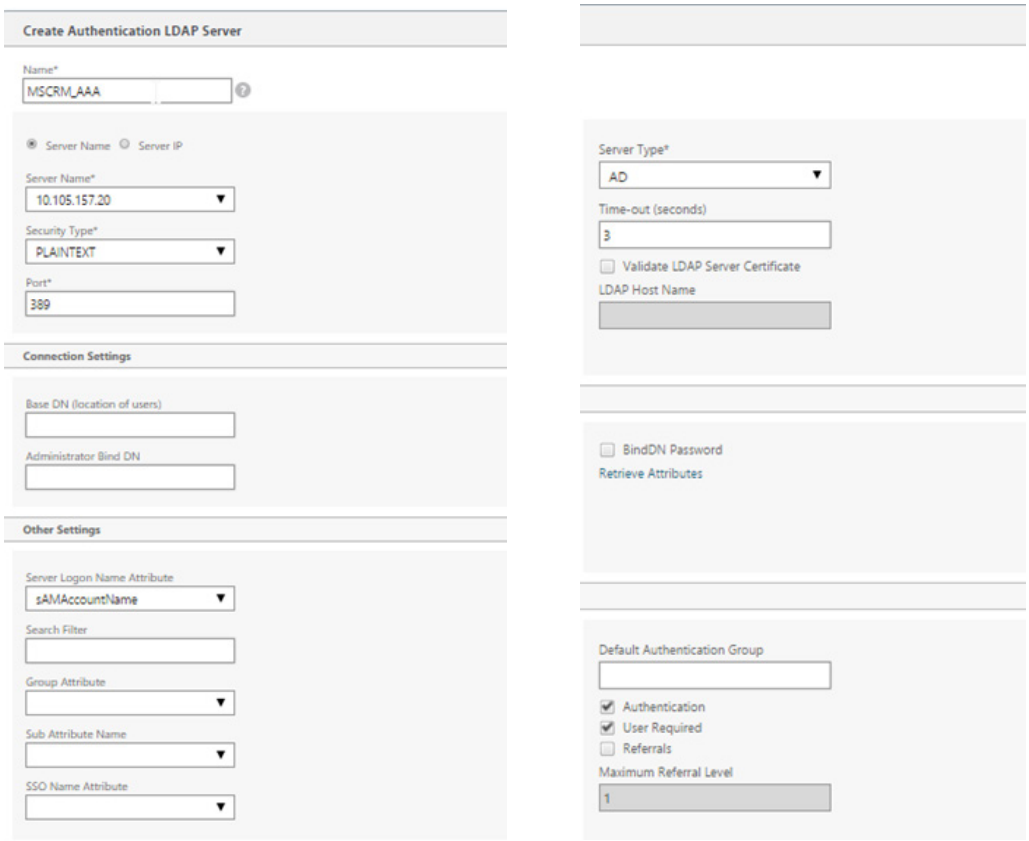


This screen presents a list of all LDAP policies defined on the NetScaler box. You can choose the appropriate policy and then click OK. Alternatively, you can add a new LDAP policy here by clicking Add.

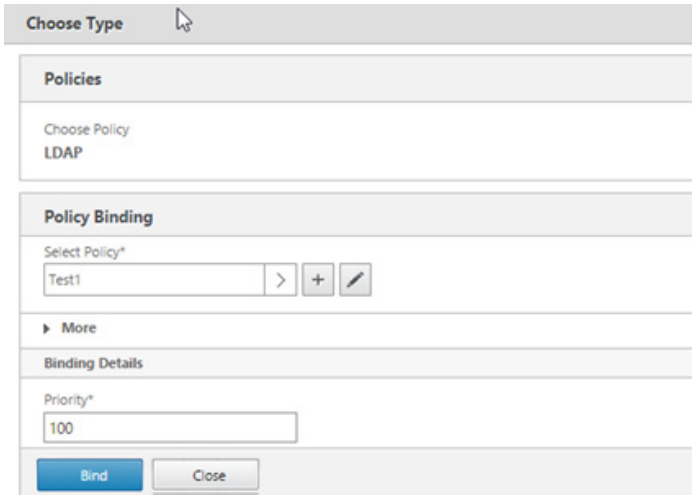
This leads to the Create Authentication LDAP Policy screen, shown below. The screen prompt is similar to other authentication mechanisms, which allow you to add settings appropriate for those schemes.



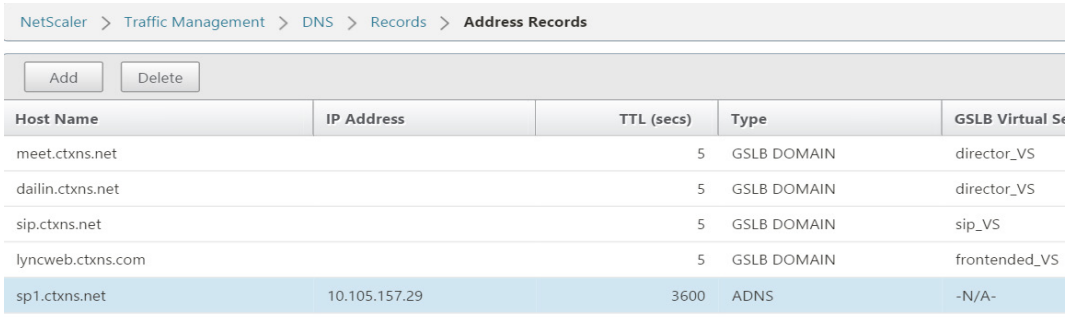
This LDAP policy needs to be bound to an LDAP server. Similar to the policy addition, a server may be chosen from a dropdown menu or a new one may be added. The new server prompt will request the settings that are necessary for the LDAP server, as shown below. The right-hand and left-hand fields must be filled in completely to facilitate LDAP authentication. Context-sensitive help (indicated by a question mark icon next to the field) is available to assist with identifying the values needed.



Upon creation of the LDAP policy, the screen below will allow you to bind the policy to the authentication vserver with the newly created policy showing in the Select Policy field and already selected.



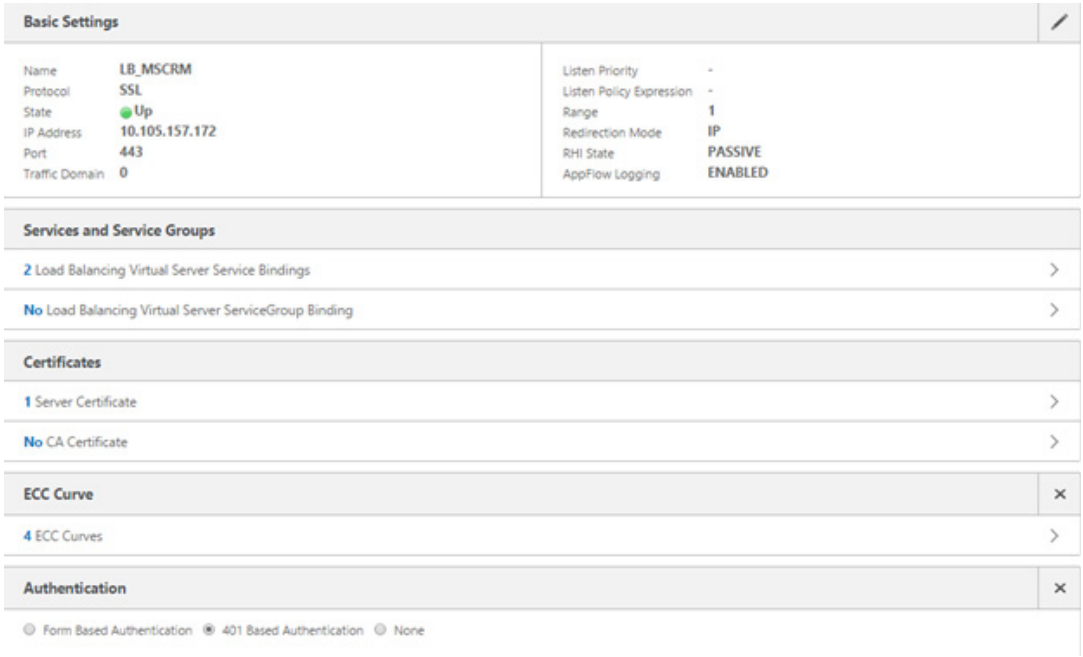
This AAA vserver should be bound to the load balancing vservers defined earlier if authentication at the NetScaler appliance is necessary.
When you look at the basic settings screen for the AAA vserver, it will show the load balancing vservers in the 401-based Virtual Servers or Form-based Virtual Servers section.
When you click on one of the options in these sections, the screen presents information about the bound 401-based load balancing vservers. The screen for the form-based vservers is similar; however, it includes information about the authentication domain.



Host Name	IP Address	TTL (secs)	Type	GSLB Virtual Se
meet.ctxns.net		5	GSLB DOMAIN	director_VS
dailin.ctxns.net		5	GSLB DOMAIN	director_VS
sip.ctxns.net		5	GSLB DOMAIN	sip_VS
lyncweb.ctxns.com		5	GSLB DOMAIN	frontended_VS
sp1.ctxns.net	10.105.157.29	3600	ADNS	-N/A-

NOTE: Make sure the FQDNs for all the vservers and services in use are added into the DNS records located at Traffic Management>DNS>Records>Address Records.

At this point, we have configured the AAA vserver MSCRM_Auth
This AAA vserver should be bound to the load balancing vservers defined for HTTP and SSL access to the Dynamics CRM servers. To bind the vservers, go back to the Basic Settings panel for the two load balancing vservers (go to Traffic Management>Load Balancing>Virtual Servers, select the two load balancing vservers LB_MSCRM and LB_MSCRM_HTTP one at a time and then click Edit). On the Basic Settings panel, go to Authentication and click on the pencil-shaped icon.



Here, you can choose between 401 (standard) and form-based authentication. Note that for form based authentication, you will be required to provide an authentication FQDN where the authentication form will be hosted and served.

LDAP authentication

To add a new LDAP authentication policy, in the navigation menu on the left, click through to AAA Application Traffic>Policies>Authentication>Basic Policies>LDAP. Here, click Add in the pane on the right to add a new policy.
Again, make sure the server is defined first. To do so, go to the Servers tab as described earlier, then click Add.
This screen will allow you to configure your LDAP server. It also provides other capabilities, such as extracting a field other than the user's subject alternative name, or SAN (for example, the user principal name, UPN). This is defined in the Other Settings section under Server Logon Name Attribute (for server login) and SSO Name Attribute (for an alternate username).

After adding the server, you can go back to policies and add the LDAP policy, similar to the RADIUS policy. The Expression section also holds similar functionality. The screen for configuration is the same as the one seen earlier.

Configuring optimization on NetScaler

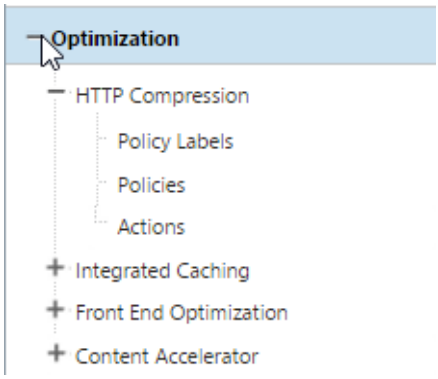
NetScaler provides a flexible, comprehensive suite of optimization capabilities that can be categorized as:

- HTTP compression
- Integrated caching
- Front-end optimization (additional optimization capabilities)

To configure these three capabilities, expand the Optimization tab in the left-hand navigation panel of the NetScaler GUI.

HTTP compression

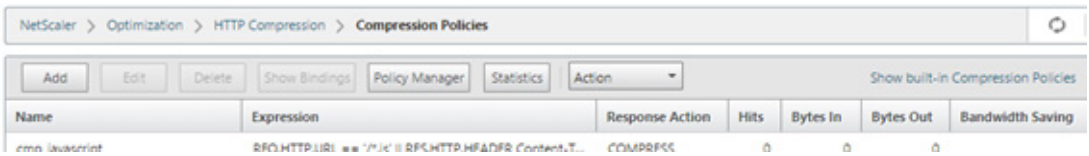
The NetScaler optimization suite is, like other NetScaler features, driven by a policy-action based architecture that allows actions driven by policies linked to specific user and system situations that are highly configurable.



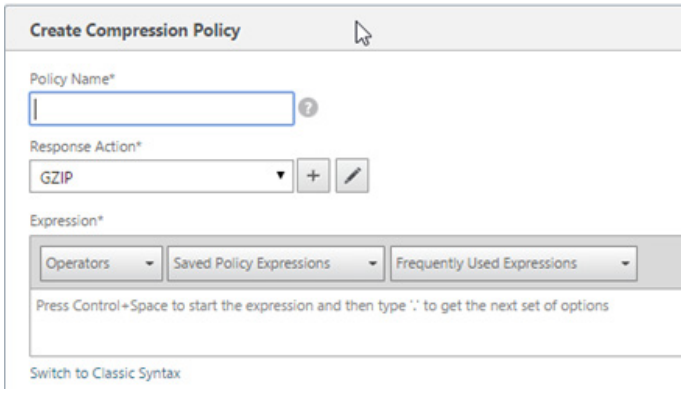
To enable HTTP compression for a particular service, you should

- Define the HTTP compression policy and action
- Bind them 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 -

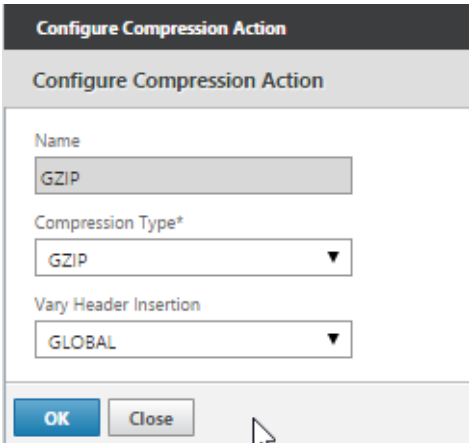


To add a new compression policy, click the Add button. This will give you the following screen:



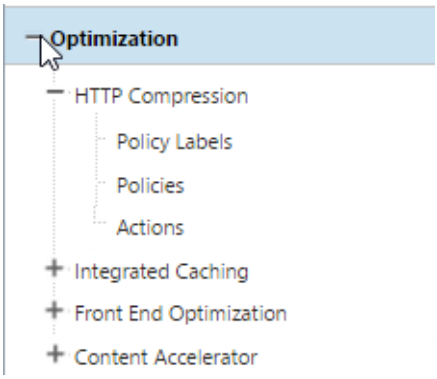
Here, you can define a name for the policy, along with an expression that defines when this policy is triggered (for example, when a particular URL is encountered. To make the policy apply to all content, use ns_true in the Expression window. For assistance here, click on the Frequently Used Expressions 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.

You have the option to 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:



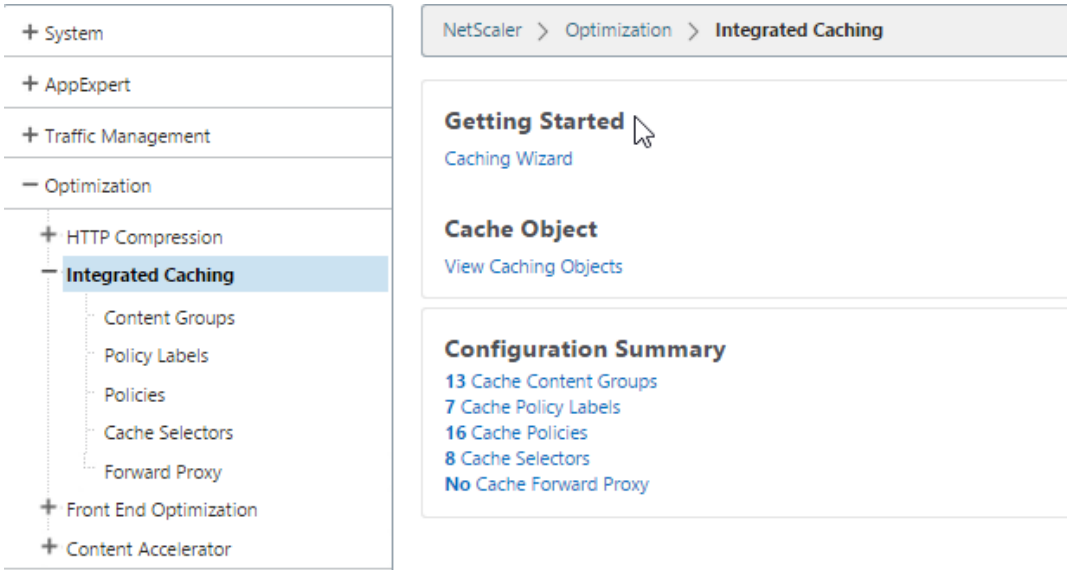
Vary header insertion is an option that is relevant for caching; the value of the vary header allows for different cache results to be returned for similar requests. For now, we will not be changing the options presented here. You can add a new action that uses a compression type of your choice.

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

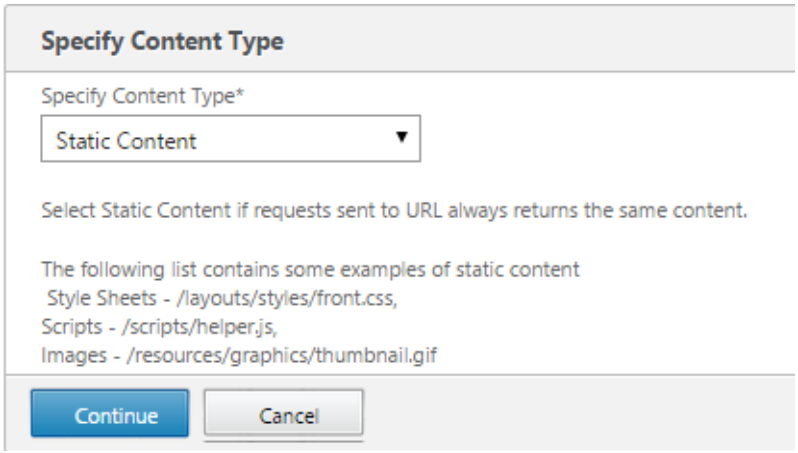


Integrated caching

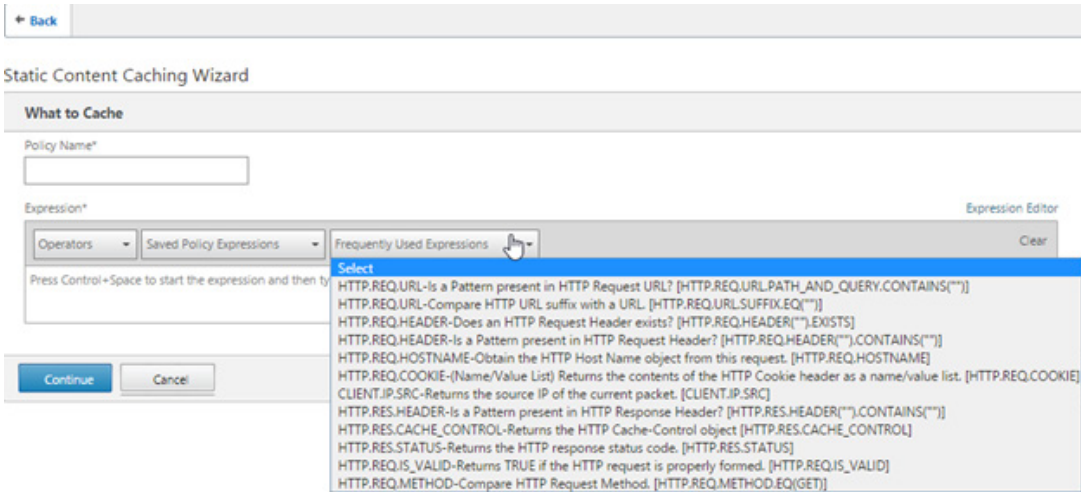
To configure caching, you can use the integrated wizard that makes configuration very straightforward. To initiate the wizard, navigate to Optimization>Integrated Caching as shown in the next screenshot.



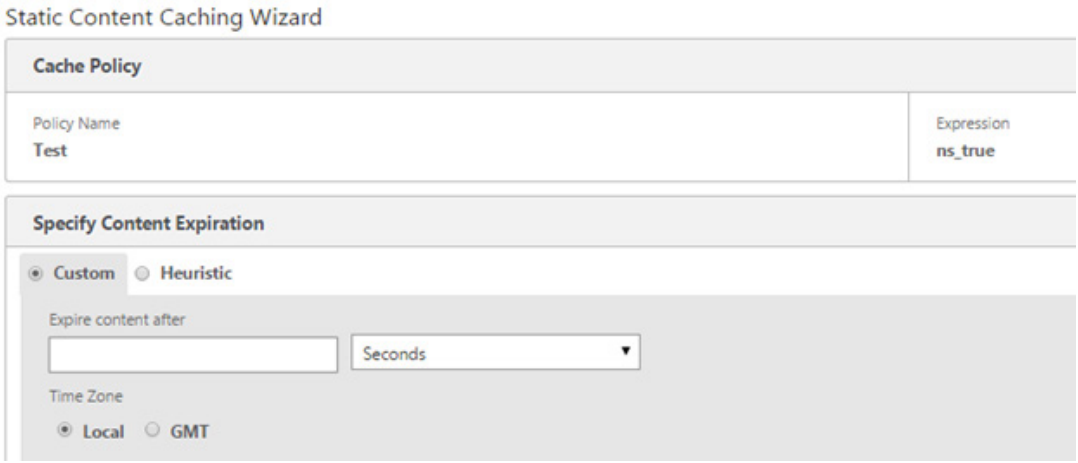
Here, you can initiate the Caching Wizard under Getting Started.



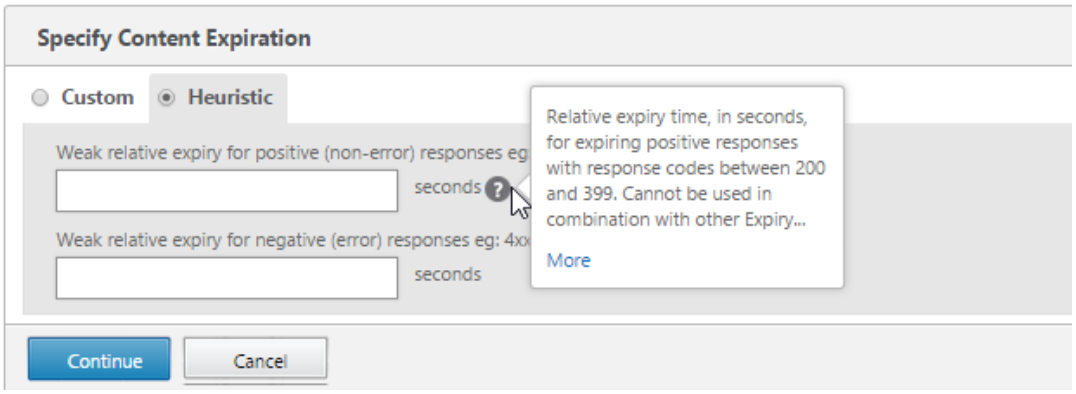
The first step requires you to specify the content type, which can be either static (examples given) or dynamic. Helpful hints are provided as shown above to help determine which type of content is relevant for you.



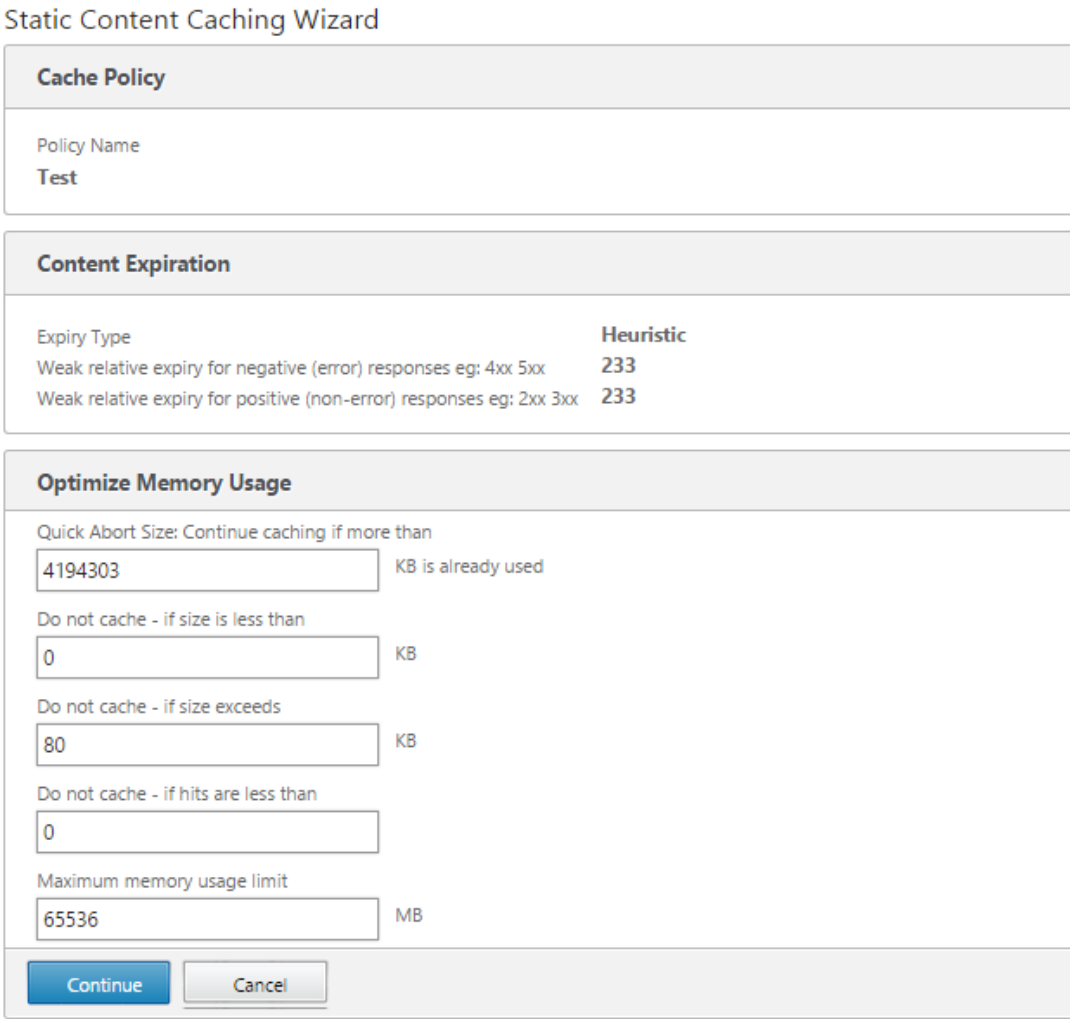
The next step involves defining which content should be cached. The Frequently Used Expressions dropdown helps you define the correct expression; however, if you want the caching policy to apply to all content, you can use ns_true as the expression. (shown in the screenshot below)



The next screen allows you to define when content expires. This can be custom (a defined interval) or heuristic (NetScaler makes the determination based on a percentage of the time since the object was last modified, with a deadline to be set that in used in case the heuristic measurement cannot be made).

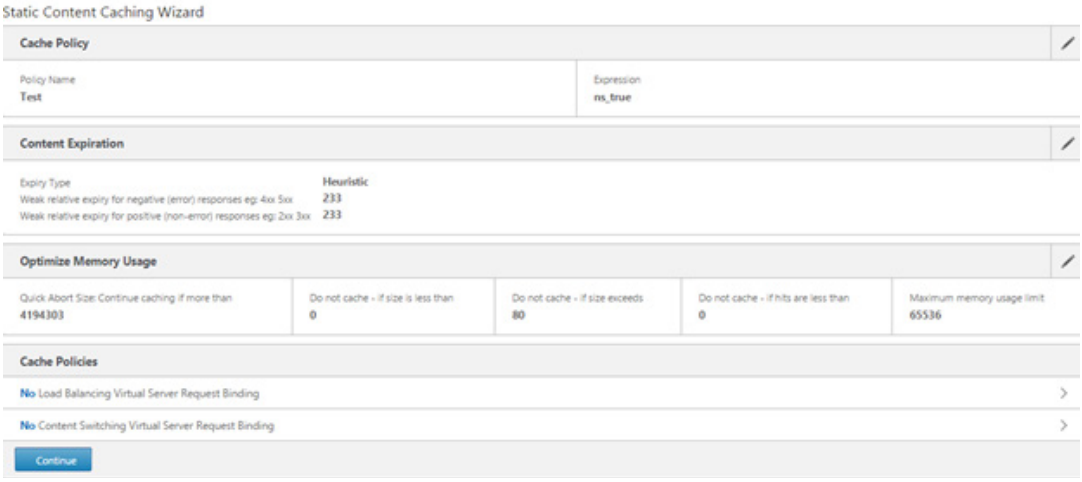


The next step involves definition of the caching space to be used on the NetScaler appliance and the minimum size of objects to be cached.



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

These definitions can be made under Cache Policies as shown in the screen shot below.

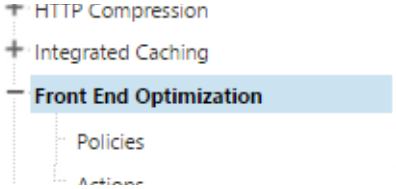


For the MSCRM deployment, the following settings have been used for caching –
Policy Name: MSCRM_Cache_Test
Actions: CACHE
Cache Content Group: Test
Undefined-Result Action: -Global-undefined-result-action (or NOCACHE/RESET)
Expression: SYS.EVAL_CLASSIC_EXPR("ns_true")

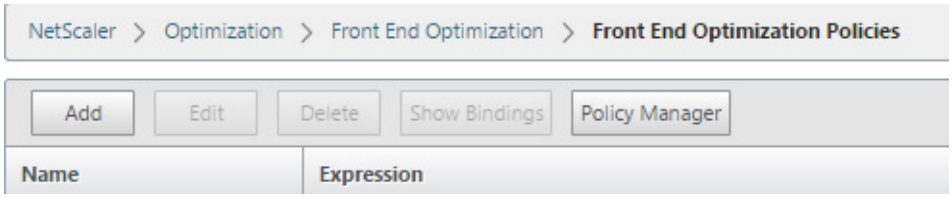
Cache Content Group:
Name: Test
Type: HTTP
Expiry Method: Heuristic (Recommended)/Custom (if specific settings are required)
Default Expiry Times: As per requirement; set to 233 for test deployment.
Parameterization: Leave values as is (unless Cache selectors are in use; not configured for our test setup)
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

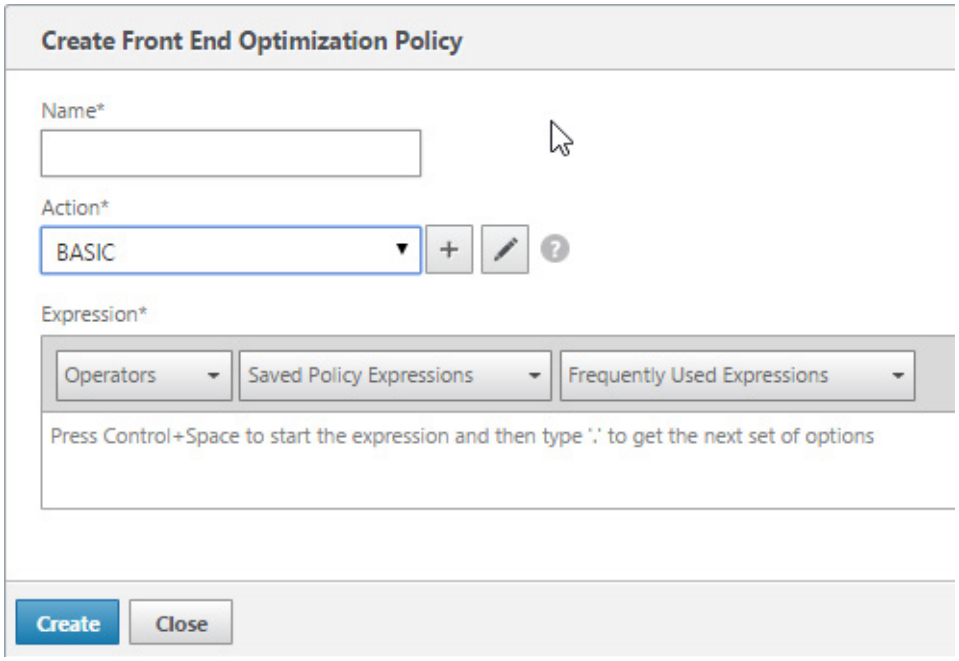
The FEO feature set makes NetScaler an extremely capable optimization device by implementing enhanced optimization routines for specific frontend entities such as images, JavaScript, etc. These features provide better optimization performance than can be achieved by compression and caching alone.
FEO 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 click Policies. Then, click Add in the section displayed to the right of the navigation menu.

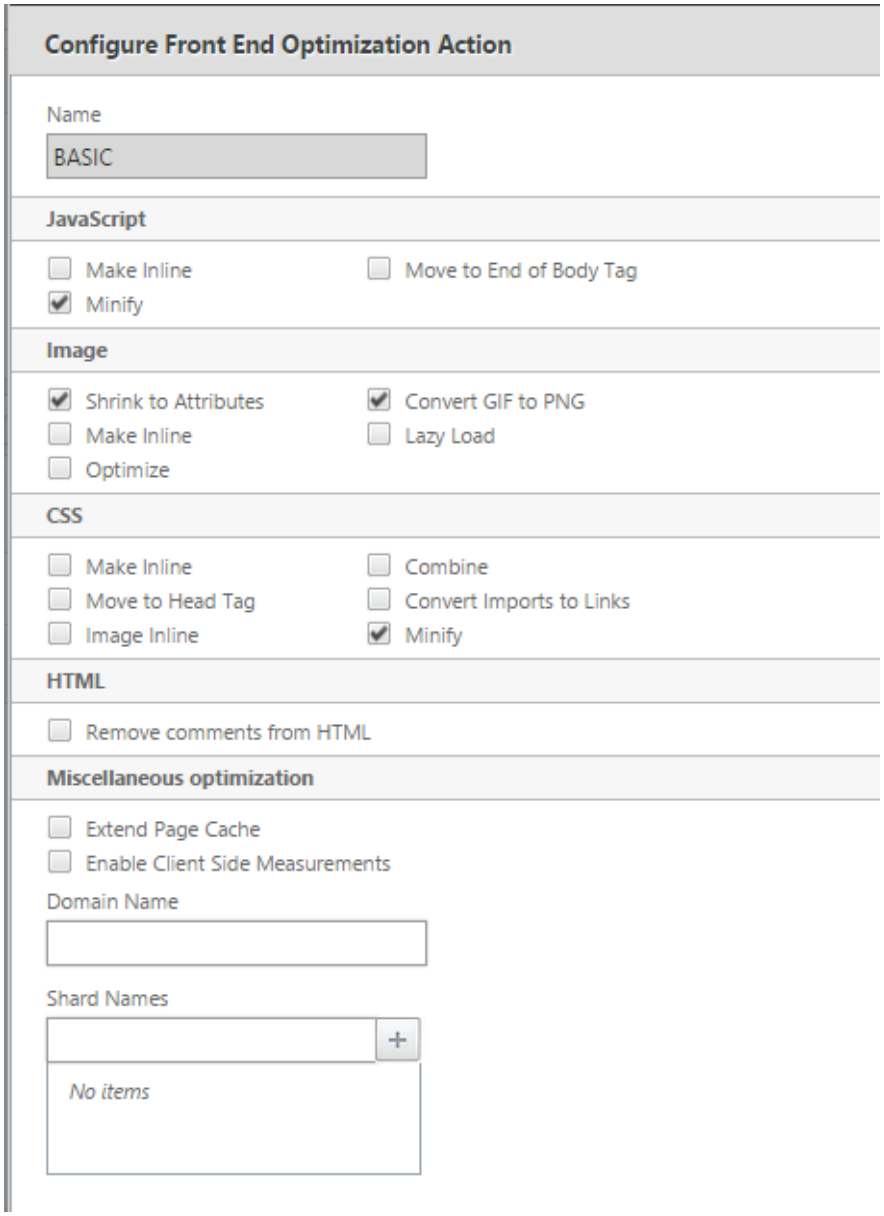


This will give you the following screen for definition of a new FEO policy.



The expression here works much the same as for the earlier features; the Frequently Used Expressions drop down can be used for assistance. Certain predefined actions can be assigned here, all of which have different configurations for the same settings. You can also edit or create a custom action using the plus or pencil buttons next to the Action name.

Upon clicking either of these buttons, you will see the following screen (or a similar one):



This screen presents all the various FEO options available. NetScaler can help to optimize web traffic with JavaScript, Image, Cascading Style Sheets (CSS), 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 setup.
Alternatively, you can set up a custom profile. The recommended settings you should enable are: JavaScript/Make Inline, JavaScript/Move to End of Body Tag, JavaScript/Minify, Image/Optimize, Image/Lazy Load, CSS/Make Inline, CSS/Move to Head Tag, CSS/Minify, HTML/Remove Comments from HTML.
Both of these profiles (Moderate as well as the custom set) give similar, good optimization numbers. These settings provided an approximate 60 percent reduction in load times, 10 percent reduction in the amount of data transferred and 13.5 percent reduction in number of requests on our test setup for generic CRM operations. Results may differ for your setup.

Optimization settings for the Dynamics CRM deployment:
Optimization policy name: MSCRM_Optimization_Test
Optimization action: MODERATE (Preconfigured)
Expression: HTTP.REQ.HEADER("Accept").CONTAINS("html")

Alternate configuration (custom policy):
Optimization policy name: MSCRM_Optimization_TestCustom
Optimization action: MS_CRM_custom
Expression: HTTP.REQ.HEADER("Accept").CONTAINS("html")

MSCRM_Optimization_TestCustom Configuration:
Enabled settings: *JavaScript/Make Inline, JavaScript/Move to End of Body Tag, JavaScript/Minify, Image/Optimize, Image/Lazy Load, CSS/Make Inline, CSS/Move to Head Tag, CSS/Minify, HTML/Remove Comments from HTML*

Conclusion

NetScaler enables an optimized and responsiveexperience with Microsoft Dynamics CRM through a superior set of optimization and application delivery capabilities. NetScaler can not only help deliver a load balanced and responsive CRM system, but can also provide for several additional capabilities such as authentication and optimization of dynamic CRM content.

Corporate Headquarters
Fort Lauderdale, FL, USA

Silicon Valley Headquarters
Santa Clara, CA, USA

EMEA Headquarters
Schaffhausen, Switzerland

India Development Center
Bangalore, India

Online Division Headquarters
Santa Barbara, CA, USA

Pacific Headquarters
Hong Kong, China

Latin America Headquarters
Coral Gables, FL, USA

UK Development Center
Chalfont, United Kingdom

CITRIX®

About Citrix
Citrix (NASDAQ:CTXS) is leading the transition to software-defining the workplace, uniting virtualization, mobility management, networking and SaaS solutions to enable new ways for businesses and people to work better. Citrix solutions power business mobility through secure, mobile workspaces that provide people with instant access to apps, desktops, data and communications on any device, over any network and cloud. With annual revenue in 2014 of $3.14 billion, Citrix solutions are in use at more than 330,000 organizations and by over 100 million users globally. Learn more at www.citrix.com.