

NetScaler® managed apps

Contents

NetScaler® managed apps	7
Benefits of NetScaler managed apps	7
NetScaler® managed apps support matrix	7
Supported browsers	8
Supported AWS regions	8
Supported Azure regions	9
Supported AWS deployments	10
Supported Azure deployments	11
Prepare AWS environment	11
Architecture	11
Create cloud access profiles in AWS	13
IAM role created for NetScaler® managed apps	14
IAM role created for NetScaler VPX™ instances	14
Create a cloud access profile	15
Validate cloud access profile	15
Manage an environment	15
Prerequisites	16
Connect to an origin application server in a different VPC	17
Connect to an origin application server outside the AWS account	19
Create an environment	21
Auto-upgrade an environment	21
Undeploy an environment	23
Prepare Azure environment	23
Architecture	23
Create cloud access profiles in Azure	26
Prerequisites	26

Assign permissions to NetScaler® ADM service	26
Assign permissions to NetScaler VPX™ instances	27
Create a cloud access profile	28
Run PowerShell script	28
Manage an environment in Azure	29
Prerequisites	29
Connect to an origin application server outside the Azure account	30
Create an environment	30
Undeploy an environment	31
Deliver a NetScaler® managed app	32
Prerequisites	33
Create an application	33
Specify application details	33
Manage an application	34
Manage app features and redeploy an application	34
Modify and redeploy an application	35
Check the health status of an application	36
Analyze the application configuration	36
Application status and environment name	37
Endpoint details	38
Service details	40
Service profiles	44
Application security details	44
Content policy details	45
Client SSL details	46
Create services	47
Self-heal slow application servers	48
View and change the back-end app server status	49

Monitor self-healing	50
Create service profiles	51
Configure load balancing	51
Configure back-end SSL settings	54
Configure health checks	56
Add endpoints	57
Add an SSL certificate	61
Add an SSL certificate while adding an endpoint	61
Get an A+ rating from Qualys Labs	61
Configure client authentication or mutual TLS (mTLS)	62
Configure authentication for the endpoints	62
How it works	62
Prerequisites	63
To set up authentication for the endpoints (FQDNs/IP addresses)	63
Bind a service to the IdP	63
Add content policies	64
Benefits	64
Prerequisites	65
Conditions	65
Operations	69
Create or Select a content policy	71
Edit a content policy	74
Delete a content policy	75
Add security protection	76
Benefits	77
Prerequisites	77
How security protection works	78
Get started with NetScaler Console service security checks	80

Allow and block list	84
Geo blocking	86
IP reputation	87
Rate limit	87
Logging settings	90
SQL injection	91
Buffer overflow	93
Cookie consistency	95
Cross-site Scripting	96
Command injection	97
Cross Site Request Forgery (CSRF)	99
Field format protection	100
Field consistency	102
WAF Signatures	103
HTTP security headers	104
Data leak prevention	107
Bot signatures	109
Bot trap	111
Device fingerprint	112
Bot TPS	112
App features	113
Deliver a classic application	114
Supported NetScaler® features	115
Manage a classic application	115
Modify and redeploy a classic application	116
Check the health status of an application	116
Create a classic application	116
Sample configuration for a classic application	117

Migrate the NetScaler configuration to the NetScaler Console service	118
NetScaler features supported in the NetScaler Console service	118
How to migrate a NetScaler configuration to the NetScaler Console service	118
How-to articles	120
Get an A+ security rating for your application in the Qualys SSL lab test	121
How does the NetScaler Console service help in getting an A+ security rating by Qualys SSL Labs?	121
Create an SSL policy with A+ security intent using the GUI	121
Create an SSL policy with A+ security intent using the API	121
Create an application with SSL policy for A+ security intent using the API	124
Rewrite HTTP headers for rewrite content policy	131
Problem Statement	132
How does NetScaler Console service content policies feature solve the problem?	132
Troubleshooting	159
Policy permissions associated with the cloud access profile are obsolete	160
VPC is already used in another environment	160
Subnet is not associated with any route table	161
fix	163

NetScaler® managed apps

NetScaler managed application deliver the applications based on the intent. The intent translates into policies, which translates into orchestration. This translation ensures faster deployment of applications and reduces manual configuration errors because you don't need to focus on the syntax. Using rich analytics, you can monitor the application performance and take corrective action to secure the application.

Lifecycle management of NetScalers is automated and there is no overhead for you. Also, rich app delivery features, such as application security (WAF, Bot signatures, IP reputation), SSL Offload, web app authentication, load balancing, content routing, and content processing are supported. Delivering applications with App Security, DNS, load balancing, and authentication is simple and intuitive.

NetScaler managed application ensure enhanced security and a seamless application experience. You can deliver NetScaler managed application in one of the following ways:

- Deliver applications using the public cloud
- Deliver applications using the on-premises NetScaler

If your existing network has a lot of application workloads on the on-premises NetScaler, then you can use the NetScaler managed application to offload the operational overhead to NetScaler Console.

Benefits of NetScaler managed apps

NetScaler managed application is an integrated secure application service that enables you to:

- Respond to the business intent of applications by simplifying app delivery and automating the lifecycle management of NetScalers.
- Continuously optimize and self-heal to improve user experience and automatically reconfigure when application conditions change, or performance degrades.

NetScaler® managed apps support matrix

NetScaler managed application feature of NetScaler Console deploys an auto-scaling app delivery infrastructure that scales on demand.

Supported browsers

To access NetScaler managed application, your workstation must have a supported web browser.

The following browsers are supported:

- Microsoft Edge: Citrix® ended support for IE11 in Citrix Cloud™ on March 31, 2022.
- Google Chrome: Chrome 19 and later
- Safari: Safari 5.1.1 and later
- Mozilla Firefox: Firefox 3.6.25 and later

Supported AWS regions

Application workloads in the following AWS regions are supported. The availability zones supported in each region depend on your AWS subscription.

- Asia Pacific (Hong Kong)
- Asia Pacific (Tokyo)
- Asia Pacific (Seoul)
- Asia Pacific (Singapore)
- Asia Pacific (Sydney)
- Africa (Cape Town)
- Europe (Frankfurt)
- Europe (Ireland)
- Europe (London)
- Europe (Paris)
- Europe (Stockholm)
- Europe (Milan)
- Middle East (Bahrain)
- South America (São Paulo)
- US East (N. Virginia)
- US East (Ohio)
- US West (N. California)
- US West (Oregon)
- Asia Pacific (Mumbai)

- Canada (Central)

Supported Azure regions

Application workloads in the following Azure regions are supported:

- Australia Central
- Australia Central 2
- Australia East
- Australia Southeast
- Brazil South
- Brazil Southeast
- Canada Central
- Canada East
- Central India
- Central US
- Central US EUAP
- East Asia
- East US
- East US 2
- East US 2 EUAP
- France Central
- France South
- Germany North
- Germany West Central
- Japan East
- Japan West
- Jio India Central
- Jio India West
- Korea Central
- Korea South
- North Central US

- North Europe
- Norway East
- Norway West
- South Africa North
- South Africa West
- South Central US
- Southeast Asia
- South India
- Sweden Central
- Switzerland North
- Switzerland West
- UAE Central
- UAE North
- UK South
- UK West
- West Central US
- West Europe
- West India
- West US
- West US 2
- West US 3

Supported AWS deployments

The following deployments are supported:

- Application servers and the NetScaler managed application infrastructure are deployed in the same VPC.
- Application servers and the NetScaler managed application infrastructure are connected via a Transit Gateway.
- Application servers and the NetScaler managed application infrastructure are connected via VPC peering.
- Application servers are deployed outside the current cloud account.

Supported Azure deployments

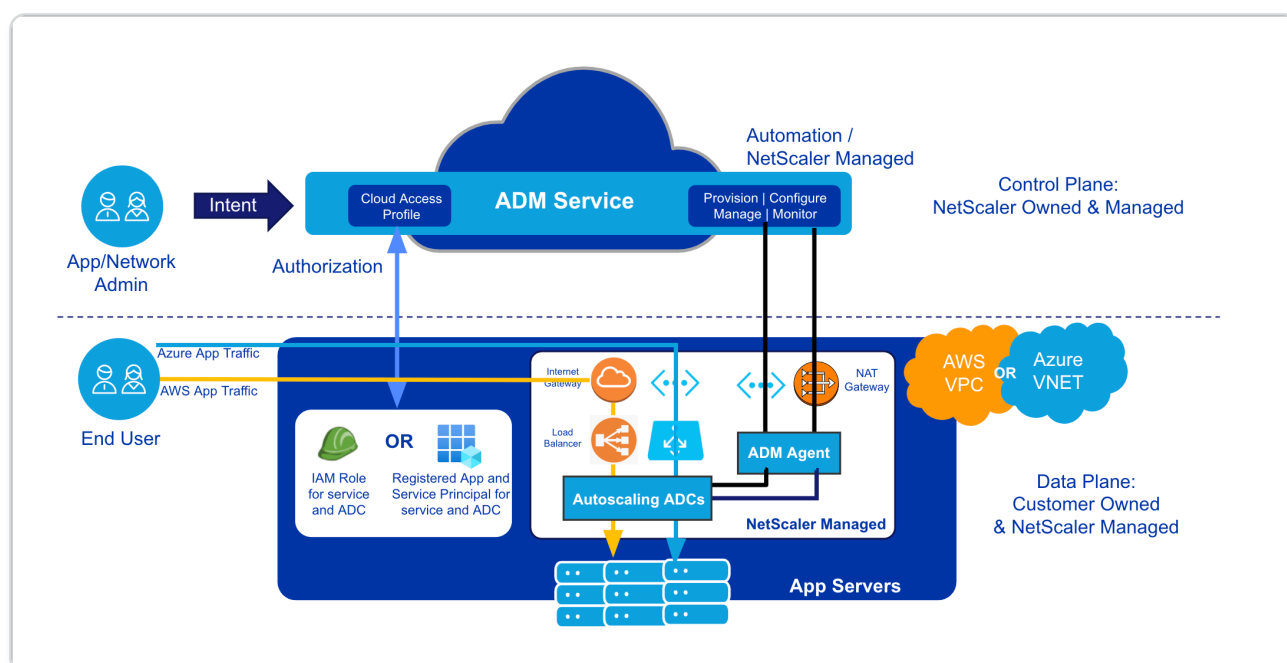
The following deployment is supported:

- Application servers and the NetScaler managed application infrastructure are deployed in the same VNet.
- Application servers are deployed outside the VNet.

Prepare AWS environment

Architecture

The following diagram illustrates the infrastructure of how NetScaler managed application are provisioned during delivery of applications on AWS.



IMPORTANT

Do not make manual changes to the infrastructure managed by the service. Doing so might interfere with the functioning of the service.

The service provisions the following elements in your AWS VPC:

- Three subnets, one each for management network, client network, and server network.
- NAT gateway that routes management (control plane) traffic from both NetScaler and the agent to the internet.

- Route table that contains entries for enabling management traffic to go to the internet through the NAT gateway.
- Agent instance that is a proxy for the service. An agent enables the service to communicate with one or multiple ADCs deployed in your VPC.
- Security groups that are associated with the NetScaler instances and the agent. Security groups control inbound and outbound traffic.

NOTE

Only port 80 and port 443 are open.

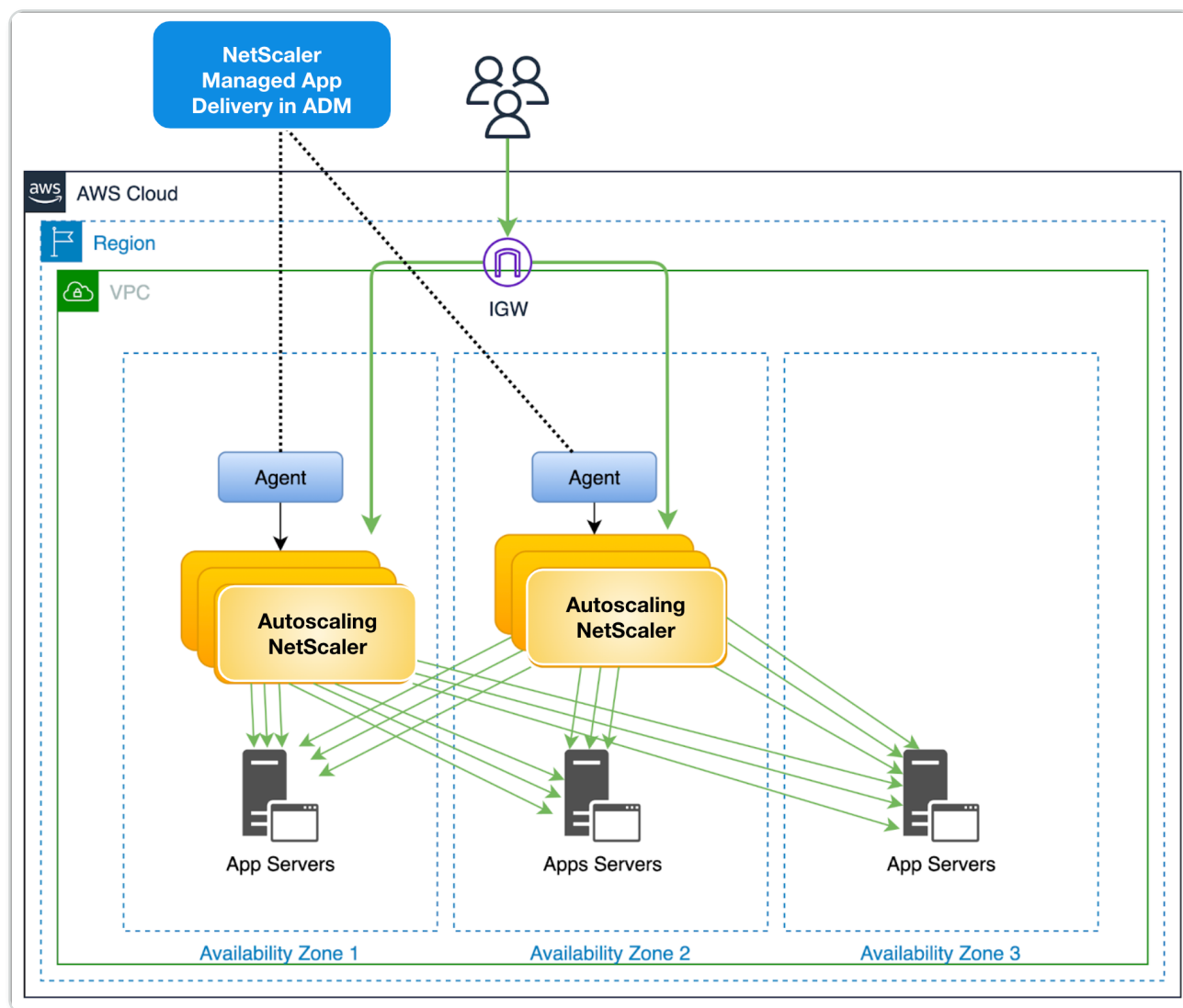
- NetScaler Autoscale cluster - A set of NetScaler instances that provide the NetScaler functionality. NetScaler instances receive traffic and distribute traffic to your application servers.

NOTES

- The service does not make any changes to the resources created by you in your VPC. It only provisions all the necessary infrastructure to deliver your applications. Provisioning includes cloud native infrastructure, such as network gateways, routing tables, subnets, security groups, NetScaler instances, and ADS service-agent instances.

Application traffic is distributed across multiple availability zones depending on the deployment of your application servers. The NetScaler cluster processes the traffic and selects an origin application server to handle the request, based on the configured load balancing settings. The origin application server might be in the same or a different availability zone.

The following diagram illustrates the traffic flow in your environment.



Create cloud access profiles in AWS

A cloud access profile is used by the service to acquire permissions on the customer's AWS account for deploying application delivery infrastructure in customer owned VPCs. To create this profile, you must be the AWS account administrator that has the necessary permissions on the VPC that you intend to use for delivering your applications. Your AWS account must allow you to use AWS CloudFormation and IAM services.

As part of creating the cloud access profile, two IAM roles on AWS are created. One IAM role gives permissions to the service to provision infrastructure on your AWS account, such as networks, NAT Gateways, security groups, and NetScaler instances. The other IAM role is used by the NetScaler VPX instances deployed in your account. The role is used to enable support for back-end auto scaling, if your application servers are deployed as an AWS autoscaling group.

NetScaler provides a ready-made CloudFormation template (CFT) to simplify the configuration of these two IAM roles. The template helps an administrator in creating these two IAM roles that are needed for the cloud access profile creation.

IAM role created for NetScaler® managed apps

The IAM role gives permissions to the service's AWS account to create and delete entities in AWS on your behalf. The following high-level permissions are granted to the service:

- Provision instances using EC2.
- Create and delete security groups and subnets.
- Create and delete NAT gateway.
- Create and delete network load balancers.
- Create and delete DNS hosted zones and DNS records inside zones using AWS Route 53.
- Attach and remove IAM roles to and from instances.

These high-level permissions are used by the service in the following scenarios:

1. When the first application is deployed in a VPC:
 - New NetScaler EC2 instances are assigned the second IAM role to enable back-end Autoscale support.
 - NAT gateway, subnets, and security groups are created.
 - Citrix agent and NetScalers are provisioned.
2. When more applications are later deployed in the same VPC:
 - IP addresses are acquired in the ADCs for the application.
 - Network load balancing is configured with these application IP addresses.
 - DNS entries are created with the domain specified for the application.
3. When autoscaling of NetScalers is done to adjust to the traffic patterns. For example, a new NetScaler instance is created if the existing set of ADCs is operating at full capacity.

The IAM role is the mechanism in AWS by which you grant these permissions to the AWS account in which you run the service.

IAM role created for NetScaler VPX™ instances

During infrastructure provisioning when the first application is deployed in a VPC:

- The NetScaler VPX instances are created.
- The IAM role with the following set of permissions is assigned to the VPX instances by the service.

The high-level permissions are used for tasks such as:

- Change the IP address on network interfaces.
- Listen to the Amazon simple queue service (SQS).
- Learn about changes to Autoscale™ groups.

The permissions are needed for the NetScalers in the following scenarios:

- During application delivery with origin application servers that are part of an Autoscale group. The NetScaler calls the AWS services to find the list of origin application servers that are part of the Autoscale group.
- If the NetScaler cluster head fails or if the cluster NetScaler head isn't reachable, then the NetScaler selects a new cluster head. The NetScaler then shifts the cluster head IP address to the newly elected cluster head.

Create a cloud access profile

1. Navigate to **NetScaler Managed Apps > App Environments**.
2. In the **Cloud Access Profiles** tab, click **Create** and select **AWS**.
3. Type a name for the profile and click **Generate Template**.
4. Follow the instructions to create a profile.
5. Click **Create**.

Validate cloud access profile

If your existing cloud access profile that connects to AWS is not in sync with the latest permissions, the service invalidates the cloud access profile. To synchronize and validate your cloud access profile with the latest IAM permissions, do the following:

1. Go to **NetScaler Managed Apps > App Environments**.
2. In the **Cloud Access Profiles** tab, click the gear icon in the **Actions** column corresponding to the required cloud access profile.
3. Follow the instructions provided on the screen.
4. Click **Validate**.

Manage an environment

An environment represents the infrastructure that is used for application delivery in a region or a data center. Typically you create one environment for an AWS VPC. The necessary infrastructure,

such as NAT gateways, agents, and ADCs, is prepared as part of the environment deployment. Once deployed, this environment can be used to deliver multiple applications.

Prerequisites

The service creates three subnets per availability zone for deploying the app delivery infrastructure. Each subnet has 255 addresses. That is, the service blocks 765 (3*255) IP addresses. Ensure that the following conditions are met for the VPC CIDR block size where the app delivery infrastructure is deployed:

- To deploy the environment in two availability zones, the VPC CIDR block size must be greater than the /21 netmask (2,048 IP addresses).
- To deploy the environment in one availability zone, the VPC CIDR block size must be greater than the /22 netmask (1,024 IP addresses). Since the infrastructure is deployed in one availability zone, we recommend not using it for a production environment. It is best suited for staging or POC.
- An instance created in the VPC must have access to the Internet through the NAT gateway and it must be able to resolve the DNS. The agent is in communication with the service at `<pop>.appdeliverysecurity.cloud.com` and it must be able to resolve the domain.

Specify a subnet

1. Navigate to **NetScaler® Managed > App Environments > Environment** tab.
2. Click **Create** and select **AWS**.
3. Select **Specify NetScaler subnet to reach the origin application servers**.
4. Specify subnets for each of the availability zones.

NOTE:

- If your origin application servers are in different VPCs, ensure that Transit Gateway and VPC peering are configured for the customized subnets.
- If your origin application servers are outside the current AWS account, ensure that the server subnets that you have added are configured to route the traffic to the origin application server subnets.

Connect to an origin application server in a different VPC

Deploying the app delivery infrastructure in the same VPC as the origin application servers saves cost and reduces management overhead. However, you might want to deploy both in different VPCs to isolate the infrastructure. For this deployment to work, the app delivery infrastructure in one VPC must be able to send traffic to the origin application servers in another VPC.

To establish communication between the VPC of the app delivery infrastructure and the VPC of the origin application servers, the following options are available:

- **VPC Peering:** Use for network connections between two VPCs.
- **Transit Gateway:** Use for network connections between many VPCs.

Prerequisites

Configure VPC peering or Transit Gateway in your AWS account so that you can select them in the GUI.

NOTES:

- The origin application server VPC cannot have overlapping CIDRs with the app delivery infrastructure VPC.
- The origin application server VPCs must have unique CIDRs.
- To avoid latency between the app delivery infrastructure VPC and the origin application servers, both must be in the same region.

For VPC peering, perform the following steps in your AWS account:

1. Create a VPC peering connection between the app delivery infrastructure VPC and origin application server VPC. A peering ID is allocated as a result.
2. Add entries to the route table for origin application servers to respond to the app delivery infrastructure VPC using the peering ID. For a sample routing table entry, see [Routing table for the VPC peering option](#).

NOTE

The NetScaler Console service configures the routes for the app delivery infrastructure VPC to send traffic to the origin application server VPC.

3. Add rules to the security group attached to the origin application server to allow traffic from the app delivery infrastructure VPC.

For Transit Gateway, perform the following steps in your AWS account:

1. Create a transit gateway.
2. Attach the VPC of the origin application servers to the transit gateway.
3. Add entries to the route table for the origin application servers to respond to the app delivery infrastructure VPC. For a sample routing table entry, see [Routing table for the Transit Gateway option](#).

NOTE

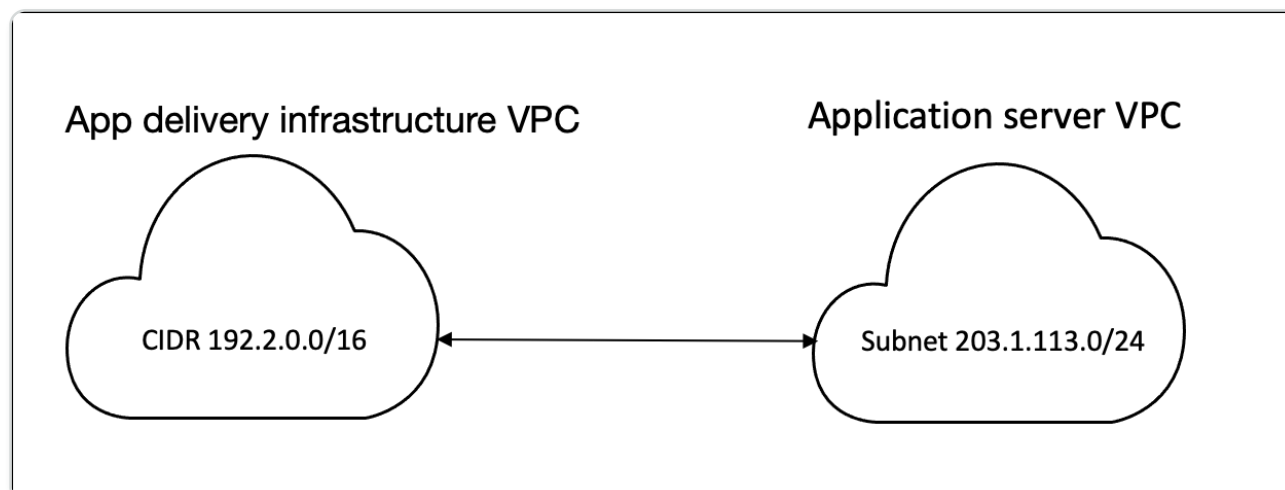
The NetScaler Console service configures the routes for the app delivery infrastructure VPC to send traffic to the origin application server VPC.

4. Add rules to the security group attached to the origin application server to allow traffic from the app delivery infrastructure VPC.

For more information about VPC peering and Transit Gateway, see the [AWS documentation](#).

Routing table for the VPC peering option

This section helps in adding entries to the route table for origin application servers to respond to the app delivery infrastructure VPC by using the peering ID.

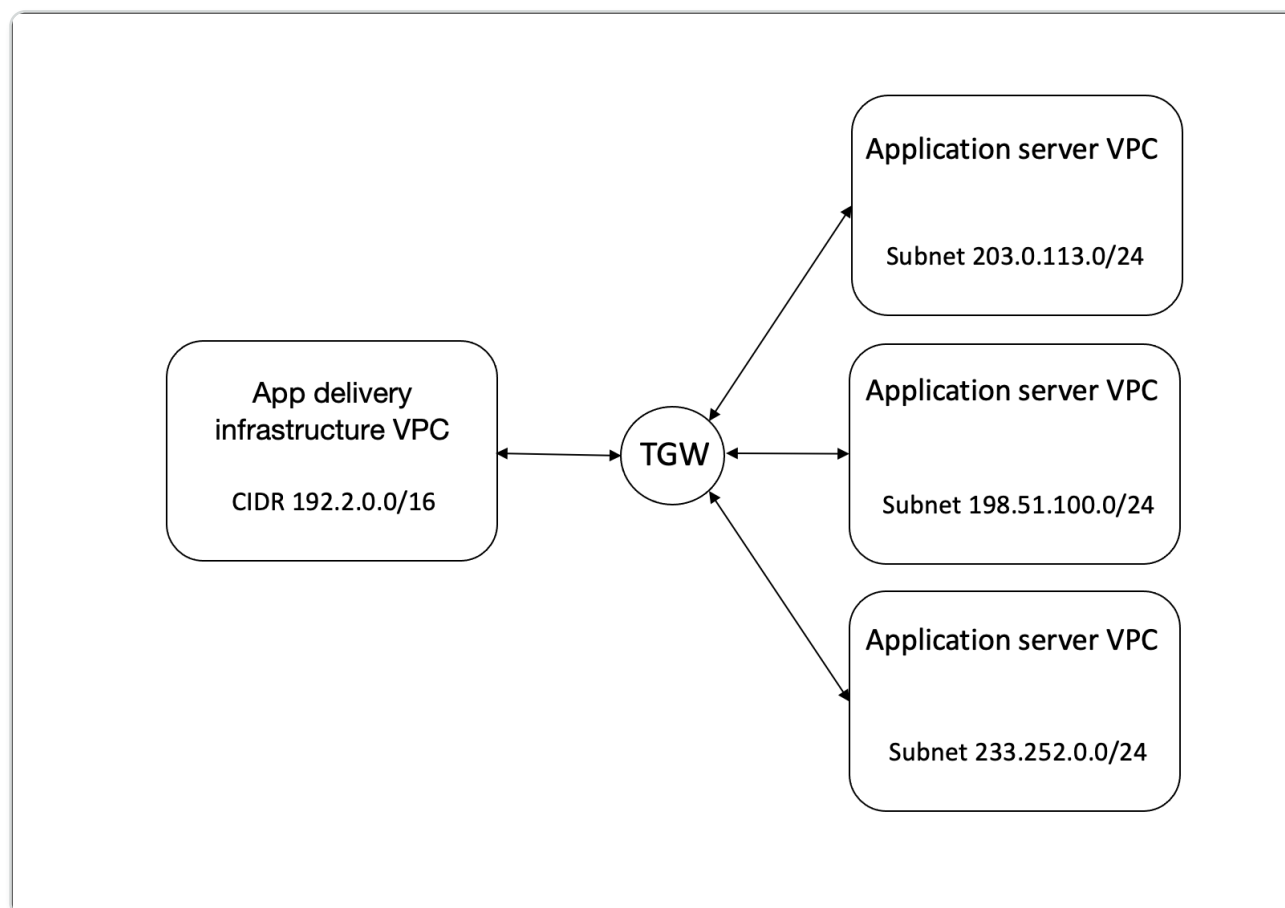


For the preceding illustration, the routing table entry in the origin application server VPC must be defined as follows:

DESTINATION	NEXT HOP
192.2.0.0/16	peering ID

Routing table for the Transit Gateway option

This section helps in adding entries to the route table for origin application servers to respond to the app delivery infrastructure VPC by using a transit gateway.



For the preceding illustration, the routing table entry for each origin application server VPC must be defined as follows:

DESTINATION	NEXT HOP
192.2.0.0/16	TGW

Connect to an origin application server outside the AWS account

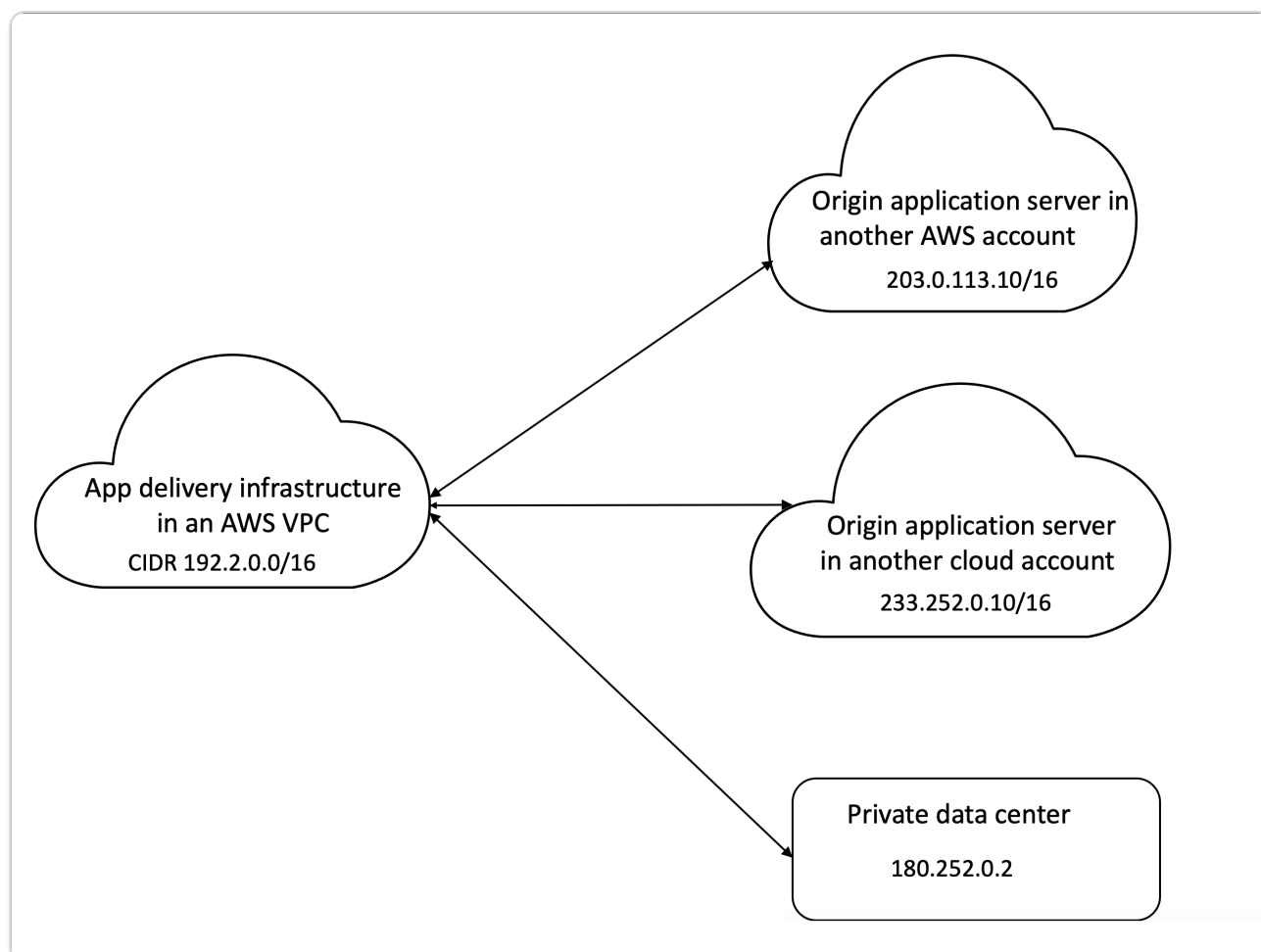
The service allows you to add the origin application servers that are deployed outside the current AWS account. For example, consider that the origin application server is in a VPC belonging to another AWS account, a private data center, or in another cloud account.

Prerequisites

- The origin application server VPC cannot have overlapping CIDRs with the app delivery infrastructure VPC.
- The origin application server VPCs must have unique CIDRs.
- To avoid latency between the app delivery infrastructure VPC and the origin application servers, both must be in the same region.

To add origin application servers outside your AWS account, you must create specific subnets that the app delivery infra must use to reach the origin application servers. While creating an AWS environment, on the **Create AWS Environment** page, select **Specify NetScaler subnet to reach the origin application servers** and add the required subnets. Only if **Specify NetScaler subnet to reach the origin application servers** is selected, the **External** tab gets enabled, where you can add the IP address or subnet of the origin application server. For more information, see [Specify a subnet](#).

For this deployment to work, you must ensure network connectivity between the origin application servers and the app delivery infrastructure. The app delivery infrastructure must be able to send traffic to the origin application servers and also receive traffic from the origin application servers.



Create an environment

1. Navigate to **NetScaler Managed Apps > App Environments**.
2. In the **Environments** tab, click **Create** and select **AWS**.
3. Select a **Cloud Access Profile**, **AWS Region**, **Availability Zone**, and **AWS VPC** where the app delivery infrastructure is to be deployed.
4. Add a **Device Access Profile** or select an existing profile. To create a profile, provide a name and a password. You can use these credentials to login to the NetScaler.
5. If you want app delivery infrastructure to be deployed in specific subnets, select **Specify subnets to be used by the app environment to reach the origin application servers** and add the subnet values.
6. Select the following tabs based on where your origin application server sites are present.
 - **Internal**: Origin application servers and app delivery infrastructure are present in the same VPC.
 - **Peering**: Origin application servers and app delivery infrastructure are in two different VPCs and are connected using VPC peering.
 - **Transit Gateway**: Origin application servers and app delivery infrastructure are spread across multiple VPCs and are connected using the transit gateway.
 - **External**: Origin application servers are provisioned in a different AWS account, a data center, or a different cloud platform. You must establish a connection between the app delivery infrastructure and the external subnets where the origin application servers are deployed.

NOTE:

The **External** tab is visible only when **Specify NetScaler subnet to reach the origin application servers** is selected.

7. Add any tags for resources if required by your AWS subscription.
8. Click **Create**.

Auto-upgrade an environment

As a network administrator, you might manage many environments running on a NetScaler managed application. An environment upgrade is a major operational activity for the administrators

who manage NetScaler instances. The NetScaler managed application automatically upgrades environments to the latest builds, thus simplifying the upgrade process in the following ways:

- The service provides three time slot options to schedule an upgrade. If you do not choose a time slot from the suggested options, the service automatically upgrades the environment in the default upgrade time slot.
- The service includes security-related updates when upgrading the environment.

Benefits

The service automatically creates maintenance jobs to upgrade your Citrix® application environments. The auto-upgrade feature provides the following benefits:

- Seamless upgrade of the environments.
- Service level agreement management. For more information, see [Service level agreement](#).

Prerequisites

You have created an environment and a cloud access profile.

View environment upgrade schedule

Navigate to **Environments> Settings** to view the following information:

- List of environments and their upgrade status.
- Environments for which an upgrade is scheduled.
- Environments for which an upgrade is in-progress. This view also shows the percentage of upgrade completed and the estimated time to complete the upgrade job in the pie chart. The pie chart representation of the in-progress upgrade is also shown.
- Environments in error state because of an upgrade failure are shown with a red indicator.
- Environments for which an upgrade is completed are shown with a green tick.

Upgrade environments

The **Citrix Environment Upgrade Scheduled** page displays the available environments. The release notes link guides you to the specific ADC release notes. You can review the new features, fixed, and known issues.

You can choose one of the following upgrade options:

- **Default schedule** – The service automatically upgrades the environment.

- **Select from system recommendation** – Select a time slot from the three available options. If a time slot is not selected, the service automatically upgrades the environment at the default upgrade time slot.

Perform the following steps to upgrade an environment:

1. In **Environments**, click **Settings**.
2. Select **Default schedule** or **Select from system recommendation schedule** and choose a time slot from the list.
3. Click **Save**.

NOTES:

- If an auto-upgrade fails, the environment rolls back to the previous version and the upgrade status displays an error state. The upgrade is then scheduled in the next time slot.
- If the rollback also fails, contact NetScaler support to resolve the issue. During this time the traffic is uninterrupted for the existing environment.

Undeploy an environment

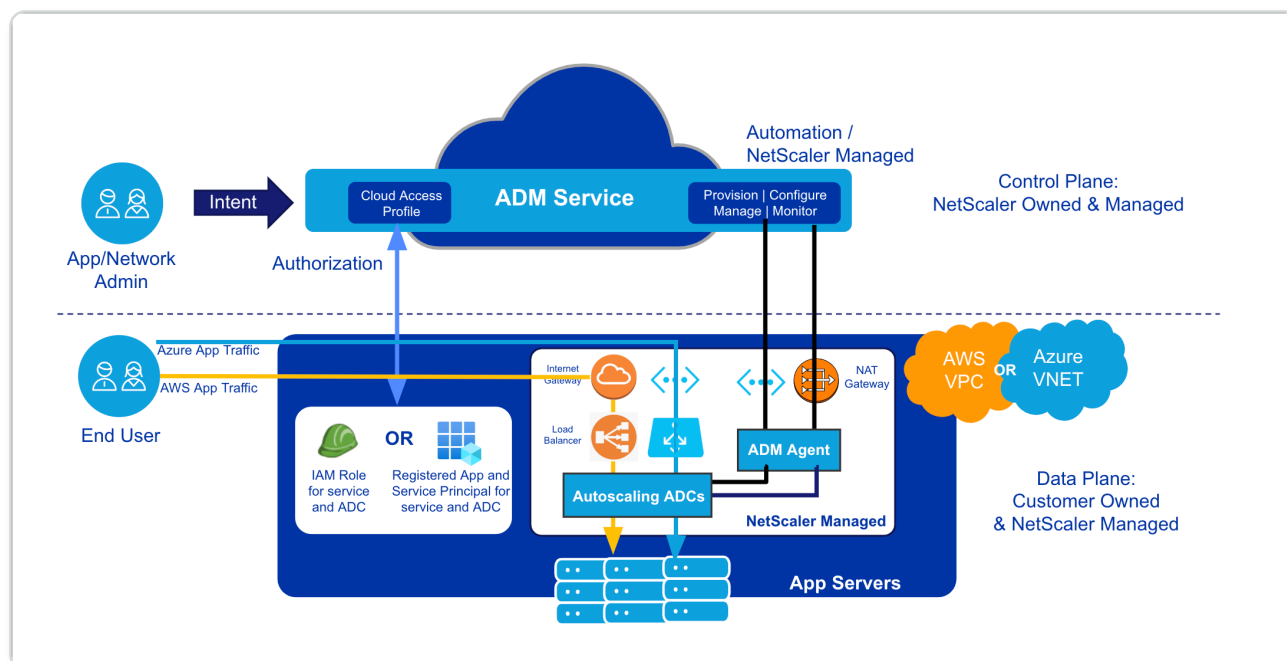
When you undeploy an environment, all the applications linked to that environment are deleted. A warning is shown if there is at least one application in the DEPLOYED state.

You can choose to cancel, undeploy the application first, and then come back to undeploy the environment.

Prepare Azure environment

Architecture

The following diagram illustrates the infrastructure of how NetScaler Console are provisioned during delivery of applications on Azure.



IMPORTANT

Do not make manual changes to the infrastructure managed by the service. Doing so might interfere with the functioning of the service.

The service provisions the following elements in your Azure Virtual Network (VNet):

- Three subnets, one each for management network, client network, and server network.
- ADM agent instance that is a proxy for the service. An agent enables the service to communicate with one or multiple NetScalers deployed in your VPC.
- Security groups that are associated with the NetScaler instances and the agent. Security groups control inbound and outbound traffic.

NOTE

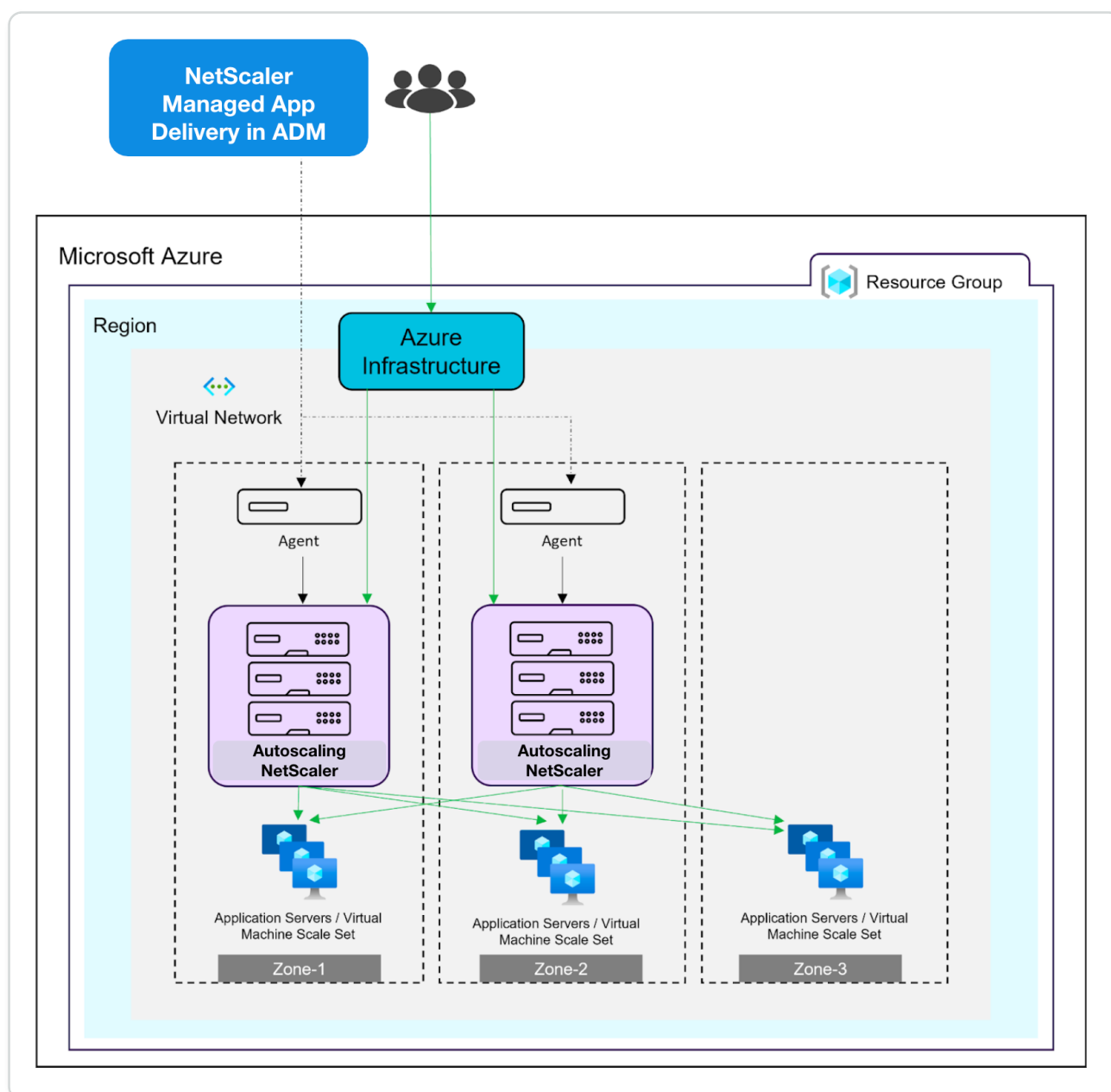
Only port 80 and port 443 are open.

- NetScaler Autoscale cluster (a set of NetScaler instances) that provide the ADC functionality. NetScaler instances receive traffic and distribute traffic to your application servers.

NOTE

The service does not make any changes to the resources created by you in your VNet. It only provisions all the necessary infrastructure to deliver your applications. Provisioning includes cloud native infrastructure, such as subnets, security groups, NetScaler instances, and NetScaler service-agent instances.

The following diagram illustrates the traffic flow in your environment. Application traffic is distributed across multiple availability zones depending on the deployment of your application servers. The ADC cluster processes the traffic and selects an origin application server to handle the request, based on the configured load balancing settings. The origin application server might be in the same or a different availability zone.



Create cloud access profiles in Azure

The service uses a cloud access profile to acquire permissions to the customer's Azure account. The service uses these permissions to deploy the application delivery infrastructure in the customer-owned Azure resource group. To create this profile, you must be an Azure account administrator that has the Owner privileges for the resource group that you use for delivering your applications.

NetScaler provides a PowerShell script that simplifies creating app registrations and secrets in the Azure CLI. The script grants required permissions to App registration to create the infrastructure and deliver apps in the customer-owned Azure resource group. When you create an app registration, an application object and service principal are automatically created in your Azure home directory or tenant. The same service principal is assigned with the permissions required for both NetScaler managed application and NetScaler VPX instances.

Prerequisites

- You have a Microsoft Azure account that supports the Azure Resource Manager deployment model.
- You have a resource group in Microsoft Azure.

For more information on how to create an account, resource group, assign privileges, and other tasks, see the [Microsoft Azure documentation](#).

Assign permissions to NetScaler® ADM service

The PowerShell script creates a custom role with permissions required for NetScaler Console. For the list of permissions assigned to the custom role, see [Permissions assigned to NetScaler Console service](#). This custom role is assigned to the service principal while creating role assignments for the resource group.

Permissions assigned to NetScaler ADM service

"Microsoft.Compute/availabilitySets/read", "Microsoft.Compute/virtualMachines/instanceView/read", "Microsoft.Compute/virtualMachines/read", "Microsoft.Compute/virtualMachines/write", "Microsoft.Compute/virtualMachines/delete", "Microsoft.Compute/virtualMachines/restart/action", "Microsoft.Compute/virtualMachineScaleSets/read", "Microsoft.Compute/virtualMachineScaleSets/networkInterfaces/read", "Microsoft.Network/virtualNetworks/read", "Microsoft.Network/virtualNetworks/subnets/read", "Microsoft.Network/virtualNetworks/subnets/write", "Microsoft.Network/virtualNetworks/subnets/delete", "Microsoft.Network/networkSecurityGroups/read", "Microsoft.Network/networkSecurityGroups/write", "Microsoft.Network/networkSecurityGroups/delete", "Microsoft.Network/publicIPAddresses/write", "Microsoft.Network/

publicIPAddresses/delete", "Microsoft.Network/publicIPAddresses/read", "Microsoft.Network/dnszones/CNAME/read", "Microsoft.Network/dnszones/CNAME/write", "Microsoft.Network/dnszones/CNAME/delete", "Microsoft.Network/loadBalancers/read", "Microsoft.Network/loadBalancers/write", "Microsoft.Network/loadBalancers/delete", "Microsoft.Network/networkInterfaces/read", "Microsoft.Network/networkInterfaces/write", "Microsoft.Network/networkInterfaces/delete", "Microsoft.Resources/subscriptions/resourceGroups/read", "Microsoft.Resources/tags/read", "Microsoft.Resources/tags/write", "Microsoft.Resources/tags/delete", "Microsoft.Resources/deployments/read", "Microsoft.Resources/deployments/write", "Microsoft.Resources/deployments/delete", "Microsoft.Resources/deployments/operationstatuses/read", "Microsoft.Resources/deployments/operations/read", "Microsoft.Network/networkSecurityGroups/join/action", "Microsoft.Compute/disks/delete", "Microsoft.Compute/disks/write", "Microsoft.Compute/disks/read", "Microsoft.Network/virtualNetworks/subnets/join/action", "Microsoft.Network/networkInterfaces/join/action", "Microsoft.Network/publicIPAddresses/join/action", "Microsoft.Network/loadBalancers/backendAddressPools/join/action", "Microsoft.Network/dnszones/read", "Microsoft.Storage/storageAccounts/write", "Microsoft.Storage/storageAccounts/read", "Microsoft.Storage/storageAccounts/delete", "Microsoft.Storage/storageAccounts/blobServices/containers/write", "Microsoft.Storage/storageAccounts/blobServices/containers/read", "Microsoft.Storage/storageAccounts/blobServices/containers/delete", "Microsoft.Compute/images/read", "Microsoft.Compute/images/write", "Microsoft.Compute/images/delete", "Microsoft.Storage/storageAccounts/blobServices/containers/blobs/read", "Microsoft.Storage/storageAccounts/blobServices/containers/blobs/write", "Microsoft.Storage/storageAccounts/blobServices/containers/blobs/delete"

Assign permissions to NetScaler VPX™ instances

The PowerShell script also creates a custom role with permissions required for NetScaler VPX instances. For the list of permissions assigned to the custom role, see [Permissions assigned to NetScaler VPX instances](#). This custom role is assigned to the same service principal while creating role assignments for the resource group.

NOTE:

The NetScaler Console service and NetScaler VPX instance use the same service principal captured in the cloud access profile.

Permissions assigned to NetScaler VPX instances

"microsoft.monitor/accounts/read", "microsoft.monitor/accounts/write", "microsoft.monitor/accounts/delete", "Microsoft.Compute/virtualMachines/read", "Microsoft.Compute/virtualMachineScaleSets/

virtualMachines/read", "Microsoft.Compute/virtualMachineScaleSets/virtualMachines/delete", "Microsoft.Compute/virtualMachineScaleSets/read", "Microsoft.Network/virtualNetworks/read", "Microsoft.Network/virtualNetworks/virtualNetworkPeerings/read", "Microsoft.Network/networkInterfaces/read", "Microsoft.Network/publicIPAddresses/read", "Microsoft.Network/loadBalancers/read", "Microsoft.ResourceGraph/resources/read", "Microsoft.Insights/AutoscaleSettings/Read", "Microsoft.Insights/AutoscaleSettings/Write", "Microsoft.Insights/AutoscaleSettings/Delete", "Microsoft.Compute/virtualMachineScaleSets/write"

Create a cloud access profile

1. Navigate to **NetScaler Managed Apps > App Environments > Cloud Access Profiles** tab, click **Create** and select **Azure**.
2. Enter a profile name.
3. Follow these instructions to create an app registration in Azure:
 - a. Download the PowerShell script provided by NetScaler. The script creates app registrations and secrets for the app registration.
 - b. Open an Azure cloud shell in a browser.
 - c. Select **PowerShell** as the type of shell.
 - d. In the Azure shell menu, select **Upload/Download**.
 - e. Upload the Citrix-provided script and run the `Connect -AzureAD` command.
 - f. Run the `./manage-app-registration.ps1` script. [Learn more](#).
4. Copy the JSON output of the script and paste it in the **Details** field of **Create Cloud Access Profile** page.
5. Click **Create** to create a cloud access profile.

Run PowerShell script

One of the important steps in creating a cloud access profile is running the PowerShell script downloaded from the NetScaler Console GUI. When you run the script, you must select one of the following options:

1. **Press '1' if you are a new user:** Creates two role definitions, one service principal, and two role assignments for each resource group. When prompted, enter the names of the resource groups where NetScaler Console service can create and configure app infrastructure. Enter the application name and the validity period in years when prompted.

2. **Press '2' to update the role:** Updates the existing role definitions with the latest set of permissions required for the NetScaler Console service.
3. **Press '3' to renew the credentials:** Deletes the expired secrets and creates new secrets with the validity period (in years) that you enter when the script prompts.
4. **Press '4' to provide access to another resource group:** Extends the scope of service principal for other resource groups. When prompted, enter the resource group name where NetScaler Console service can create and configure app infrastructure. Enter the application name when prompted.
5. **Press '5' to cleanup azure AAD application:** Removes all role assignments, both of the role definitions, and the app registration.
6. **Press 'Q' to quit:** Exits the PowerShell script.

Manage an environment in Azure

An environment represents the infrastructure that is used for application delivery. The necessary infrastructure, such as gateways, agents, and NetScalers, is prepared as part of the environment deployment. Once deployed, this environment can be used to deliver multiple applications.

Prerequisites

The service creates three subnets in Azure for deploying the app delivery infrastructure. Each subnet has 255 addresses. That is, the service blocks 765 (3*255) IP addresses.

- Ensure that the VNet CIDR block size where the app delivery infrastructure is deployed has enough space to accommodate three subnets of 255 addresses.
- To deploy the environment in one or more availability zones, the VNet CIDR block size must be greater than the /22 netmask (1,024 IP addresses). Since the infrastructure is deployed in one availability zone, we recommend not to use it for a production environment. It is best suited for staging or POC.
- A virtual machine in the VNet must have access to the Internet through the external Azure load balancer (ALB) created by the service and must be able to resolve the DNS. The agent communicates with the service at `<pop>.appdeliverysecurity.cloud.com` through the external ALB and it must be able to resolve the domain.

Specify a subnet

1. Navigate to **NetScaler® Managed > App Environments > Environment** tab.
2. Click **Create** and select **Azure**.

3. Select **Specify NetScaler subnet to reach the origin application servers**.
4. Select the subnet.

NOTE:

If your origin application servers are outside the current Azure account, ensure that the server subnet that you have added are configured to route the traffic to the origin application server subnets.

Connect to an origin application server outside the Azure account

The service allows you to add the origin application servers that are deployed outside the current Azure account. For example, consider that the origin application server is in a VNet belonging to another Azure account, a private data center, or in another cloud account.

Prerequisites

- The origin application server VNet cannot have overlapping CIDRs with the app delivery infrastructure VNet.
- The origin application server VNets must have unique CIDRs.
- To avoid latency between the app delivery infrastructure VNet and the origin application servers, both must be in the same region.

To add origin application servers outside your Azure account, you must create a specific subnet that the app delivery infrastructure must use to reach the origin application servers. While creating an Azure environment, on the **Create Azure Environment** page, select **Specify NetScaler subnet to reach the origin application servers** and add the required subnet. Only if **Specify NetScaler subnet to reach the origin application servers** is selected, the **External** tab gets enabled. Add the IP address or subnet of the origin application server. For more information, see [Specify a subnet](#).

For this deployment to work, you must ensure network connectivity between the origin application servers and the app delivery infrastructure. The app delivery infrastructure must be able to send traffic to the origin application servers and also receive traffic from the origin application servers.

Create an environment

1. Navigate to **NetScaler Managed Apps > App Environments**.
2. In the **Environments** tab, click **Create** and select **Azure**.
3. In the **Create Azure Environment** page, specify values for the following parameters:

- Name
 - Cloud Access Profile
 - Azure Region
 - Azure Resource Groups
 - Azure Virtual Network
 - Availability Zones
4. Add a **Device Access Profile** or select an existing profile. To create a profile, provide a name and a password. You can use these credentials to login to the NetScaler.
 5. If you want app delivery infrastructure to be deployed in a specific subnet, select **Specify NetScaler subnet to reach the origin application servers** and add the subnet value.
 6. Select the following tabs based on where your origin application server sites are present:
 - **Internal**: Select when the origin application servers and app delivery infrastructure are present in the same VNet. Select the **Origin application servers present in the same VNet** check box.
 - **External**: Select when the origin application servers are outside the Azure account. You must establish a connection between the app delivery infrastructure and the external subnets where the origin application servers are deployed.

NOTE:

The **External** tab becomes visible only when **Specify subnets to be used by the app environment to reach the origin application servers** is selected.

7. Add any tags for resources if required by your Azure subscription.
8. Click **Create**.

Undeploy an environment

When you undeploy an environment that has one or more applications in the DEPLOYED state linked to it, all the applications associated with that environment are forcefully deleted upon confirmation. You can choose to first delete the application, and then undeploy the environment.

Deliver a NetScaler® managed app

An application is the central component in NetScaler managed application and contains the application delivery and security information required to deliver the customer's application. Two main components of an application are its services and endpoints. Services correspond to the customer's application servers. Endpoints correspond to the FQDN and virtual IP addresses that users use to reach the application.

The configuration of an application is organized as a set of application delivery features. These features help in optimizing, securing, accelerating, and ensuring availability of customers' applications. Each application delivery feature allows a user to configure a specific feature provided by NetScaler Console. Together they constitute the overall application delivery configuration for your application.

You can create services to represent your origin application servers. You can select a default service profile that contains the most commonly used settings for a service. Alternately, you can create your own service profile and specify the load balancing settings, back-end SSL settings, and health checks for your servers. For more information about services, see [Create services](#). For more information about service profiles, see [Create service profiles](#).

Using the authentication feature, admins can add SAML authentication to the apps. Once authentication is enabled, only authenticated users are allowed to access specific services configured by the admin. For more information, see [Authentication](#).

Using content policy, you can evaluate an incoming request and apply one or more actions based on the evaluation. For example, you can drop a connection if DDoS attack is suspected or manipulate the data in HTTP requests and responses. NetScaler Console supports both rewrite and responder content policy. For more information, see [Content Policy](#).

NOTE:

You can configure the following application delivery features and reuse them across multiple applications:

- Content policies
- SSL settings
- Authentication
- Service profiles

For more information, see [Manage app features](#).

The security protection feature protects your applications from security threats. Create security protection to configure features, such as allow or block requests, add exceptions, define rules to examine the traffic, geo blocking, rate limiting, and cookie consistency. Exceptions can be added for cross-site scripting protection and buffer overflow protection. Exceptions help avoid false positives and bypass the traffic. For more information, see [Security protection](#).

The analyzer feature of the NetScaler managed application displays concise configuration information of your application in a pictorial way. This information helps you analyze the application's status easily and also helps in troubleshooting purposes. You can access the analyzer from the application dashboard. For more information, see [Visualize the application-configuration](#).

Prerequisites

- Before you can create an application, you must create at least one cloud access profile and one application environment.

A classic application in the NetScaler Console service offers granular control over the configuration as compared to a modern application and is recommended for advanced NetScaler users. For more information, see [Deliver a classic application](#).

Create an application

Follow these steps to create an application:

1. [Specify application details](#).
2. [Create services](#).
3. [Add endpoints](#).
4. [Configure authentication for the endpoints](#).
5. [Add Content policy](#).
6. [Add Security protection](#).
7. Deploy.

Specify application details

You must have at least one environment before you can proceed with application creation.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application**.
2. Type a name for the application.

3. Select one of the following options:
 - a. **Environment:** When the app delivery infrastructure is available on the cloud.
 - b. **Instance:** When the app delivery infrastructure is an on-premises data center.
4. Select the required environment or instance from the list.
5. Click **Next** and complete the configuration workflow.
6. Click **Deploy**.

Manage an application

You can manage your applications individually or through app features. When managing the applications individually, you must edit and redeploy each application. While managing through app features, you can edit an app feature setting and redeploy all the applications that use these settings in one go. For information on app features settings, see [Manage app features and redeploy an application](#).

To manage applications individually, navigate to the **NetScaler Managed Apps > App Delivery** page, click one of the following options in the **Actions** column to manage your deployed apps. Redeploy the application after making changes.

- Edit
- Redeploy
- Undeploy
- Health Status

Manage app features and redeploy an application

You can create and edit customized configurations specific to your network needs for the following app features:

- [Content policies](#)
- [SSL settings](#)
- [Authentication](#)
- [Service profiles](#)

After the configuration changes, you can deploy all the applications associated with the app features in one go.

1. Navigate to **NetScaler Managed Apps > App Features** and select the required tab.

2. Perform one of the following:

- a. To create a configuration, click **Create**.
- b. To edit an existing configuration, click the three dots in the **Actions** column and click **Edit**. After changing the configuration, click **Update and Deploy**.
- c. To deploy the applications associated with an updated configuration, click the three dots in the **Actions** column and click **Deploy**.
- d. To delete a configuration, click the three dots in the **Actions** column and click **Delete**.

NOTE:

You can delete a network function setting only if it is not associated with any application.

You can check the status of the network function setting in the **Status** column.

NOTE:

If one or more applications associated with the updated configuration is not in the **Deployed** state, the **Status** column shows **Error**. Click **Error** to view applications that failed during deployment. To know why deployment failed for a specific application, navigate to the **Application** dashboard and click **Error** in the **Status** column corresponding to that application.

Modify and redeploy an application

You can edit the application details, services, endpoints, load balancing, content policy, and security protection settings for an application.

1. Navigate to the **NetScaler Managed Apps > App Delivery** page.
2. For applications whose status is **Deployed**, click the three dots in the **Actions** column.
3. Click **Edit**.
4. Click any of the tabs to change the configured values and click **Deploy**.

You have completed the steps to modify and redeploy an application.

Check the health status of an application

The health status displays the real-time health of your deployed applications for each application service in an availability zone.

1. Navigate to the **Applications** page.
2. For applications whose status is **Deployed**, click the three dots in the **Actions** column.
3. Click **Health Status**.

Analyze the application configuration

You can view the visual representation of an application configuration and use the details for analyzing the application's status and troubleshooting.

Navigate to **NetScaler Managed Apps > Applications** and click the **Analyzer** icon in the **Actions** column.

Applications

[New Application](#)[Migrate NetScaler Configuration](#)

NAME	ENDPOINT IP	FQDN	MODEL	VISIBILITY	TARGET	STATUS	ACTIONS
demo-application	20.48.75.207	https://demoapp207.azure.org/affinity.html	MODERN	External	app207-2048-75207	DEPLOYED	 ...
...			...	...	...	...	 ...
...			...	...	...	...	 ...

Showing 1 - 3 of 3 itemsPage 1 of 110 rows

Analyzer

demo-application:

(Environment: DemoEnv)

Endpoint

SSL Profile

sample-endpoint

SSL Certs

sample-service

Service

sample-service

Default HTTPS Profile

Health Checks: 0

ipaddress_fqdns

Service Profiles

Content Policies

Security Protection

Client SSL

Default HTTPS Profile

sample-content-policy

security-protection-policy-1

BasicSSLFrontendNf-683927

BasicSSLFrontendNf-918921

BasicSSLFrontendNf-795369

Endpoint

Endpoint Name : sample-endpoint-2

Endpoint Type : External

Auto Allocated : No

FQDN : mysampledomain.tenanta.citrixconn.net

Port : 443

Protocol : HTTPS

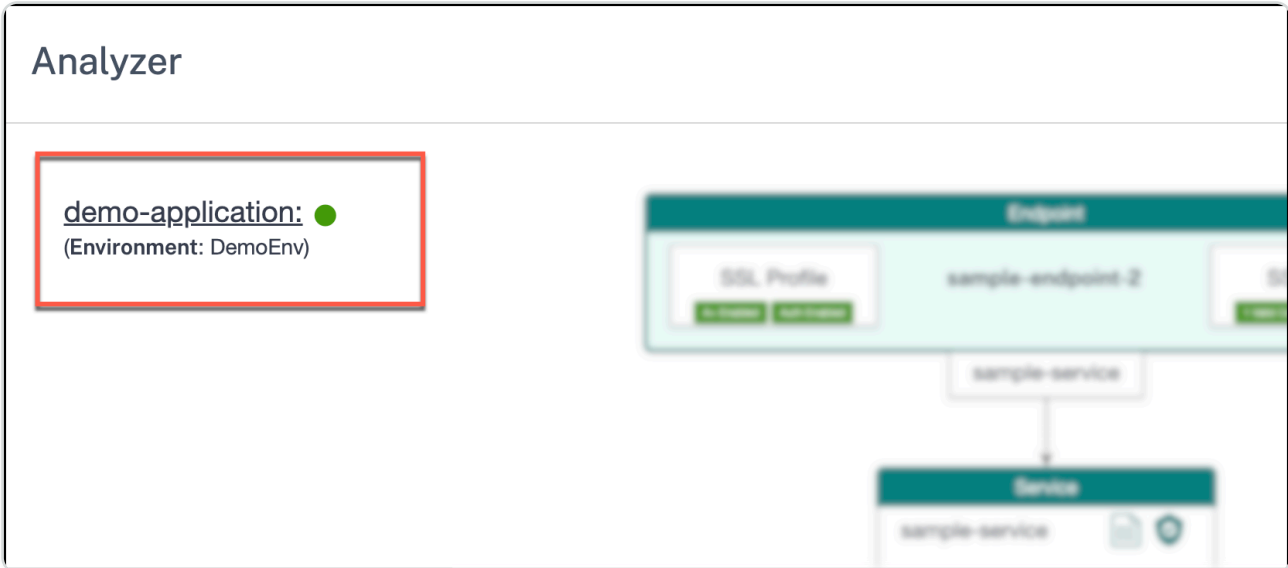
HTTPS redirect : Yes

Done

Application status and environment name

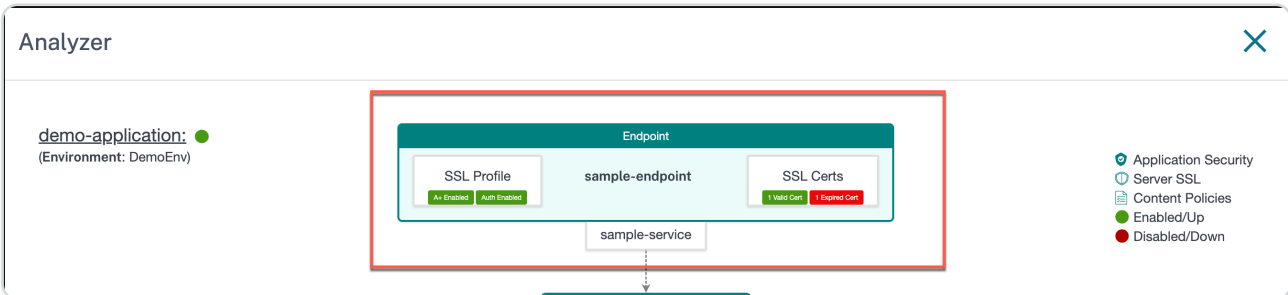
Application status is indicated next to the application name at the top-left corner. If the application is up and running, you see a green dot and if the application is down, you see a red dot.

The name of the environment where the application is deployed appears below the application name.



Endpoint details

The **Endpoints** tile displays the name of the endpoint associated with the application. The **SSL Profile** and **SSL Cert** sections is available in the **Endpoints** tile if HTTPS is configured as the protocol. The name of the service associated with the endpoint is displayed at the bottom of the **Endpoint** tile.



Click the **Endpoint** tile and view the following details in the lower pane:

- Name of the endpoint
- If the endpoint is accessible internally or externally
- If the FQDN type is auto-allocated or user-defined
- Protocol and port details
- If “Auto redirect HTTP traffic to HTTPS” is configured

Endpoint			
Endpoint Name :	sample-endpoint	Endpoint Type :	External
Auto Allocated :	No	FQDN :	mysampledomain.tenanta.citrixconn.net
Port :	443	Protocol :	HTTPS
HTTPS redirect :	Yes		

The **SSL Profile** section within the endpoints tile displays if A+ security and client authentication is enabled. Click the **SSL Profile** section to view the following details:

- Name
- Status of A Plus security and client authentication (enabled or disabled)
- Status of TLS versions from 1.0 to 1.3 (enabled or disabled)
- List of ciphers associated with the SSL profile

The screenshot shows the 'Analyzer' window with a diagram of the application architecture. The 'demo-application' (Environment: DemoEnv) is connected to an 'Endpoint' (sample-endpoint) which contains an 'SSL Profile' (highlighted with a red box), 'SSL Certs' (1 Valid Cert, 1 Expired Cert), and a 'sample-service'. The 'sample-service' is connected to a 'Service' (sample-service) which has a 'Default HTTPS' endpoint.

The 'SSL Profile' section is expanded, showing the following configuration:

Property	Value
Name :	BasicSSLFrontendNf-683927
A Plus Security :	Disabled
Client Authentication :	Disabled
TLS 1.0 :	Enabled
TLS 1.1 :	Disabled
TLS 1.2 :	Enabled
TLS 1.3 :	Disabled

Ciphers :

NAME
TLS1-AES-256-CBC-SHA
TLS1-AES-128-CBC-SHA
TLS1.2-AES-256-SHA256
TLS1.2-AES-128-SHA256
TLS1-ECDSA-RSA-AES256-SHA

Showing 1 - 5 of 5 items Page 1 of 1

The **SSL Certs** section displays the number of valid and expired certificates. Click the **SSL Certs** section to view the following details:

- Name
- Status
- Number of days for the certificate to expire
- Public key algorithm
- Subject name

Analyzer

demo-application:

(Environment: DemoEnv)

Endpoint

SSL Profile

sample-endpoint

SSL Certs

sample-service

Service

sample-service

Default HTTPS

Application Security

Server SSL

Content Policies

Enabled/Up

Disabled/Down

SSL Certificates

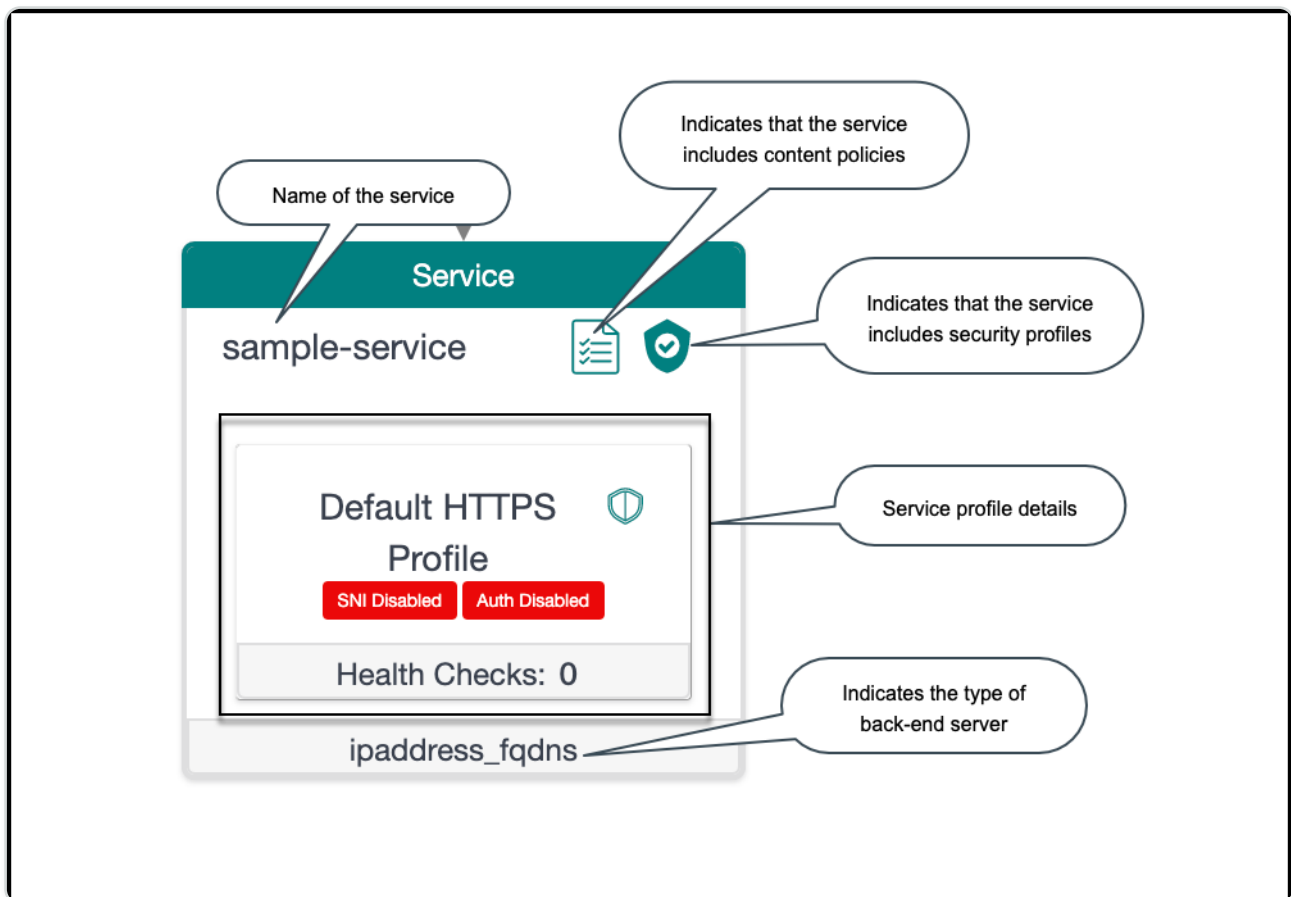
Certificates :

NAME	STATUS	DAYS TO EXPIRE	PUBLIC KEY ALGORITHM	SUBJECT NAME
SanityCert	Expired	0	rsaEncryption	CN = StarFruits, C = IN, L = FruitLand, O = Orange Farming Inc, OU = Orange Juice
cacert	Valid	6586	rsaEncryption	C = in, O = citrix, CN = ca

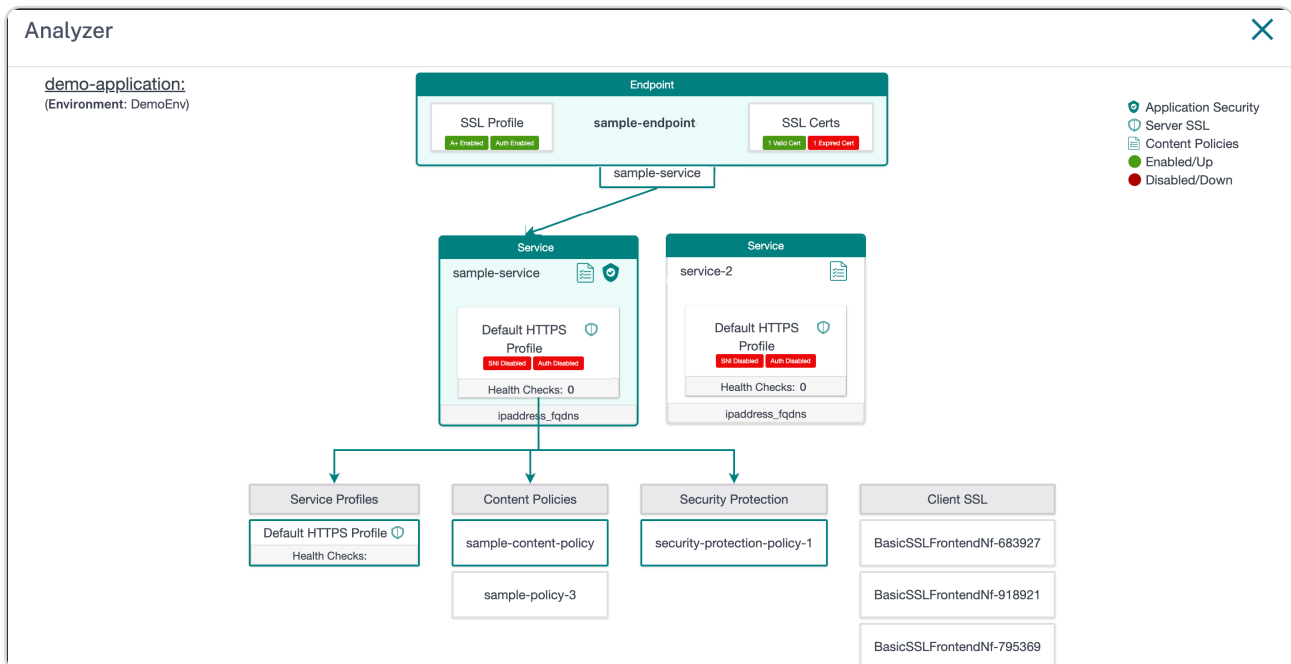
Done

Service details

The **Service** tile displays the name of the service and the service profile associated with the service. The service profile information is shown as a section within the **Service** tile. If the service contains security policies and content policies, then the corresponding legend is displayed on the **Service** tile. The type of back-end app server is indicated at the bottom of the tile.



If you have configured more than one service, then all of them appear in the analyzer. When you hover over the **Service** tile, all the configurations associated with the application get highlighted and linked.



Click the **Service** tile to view the following details:

- Name

- Back-end application server type
- Protocol and port number
- List of servers
 - Type of server
 - IP address
 - Configured status
 - Get status: Click to view details of the back-end application server status and change the status. For more information, see [View and change the back-end app server status](#).

demo-application:
(Environment: DemoEnv)

Service

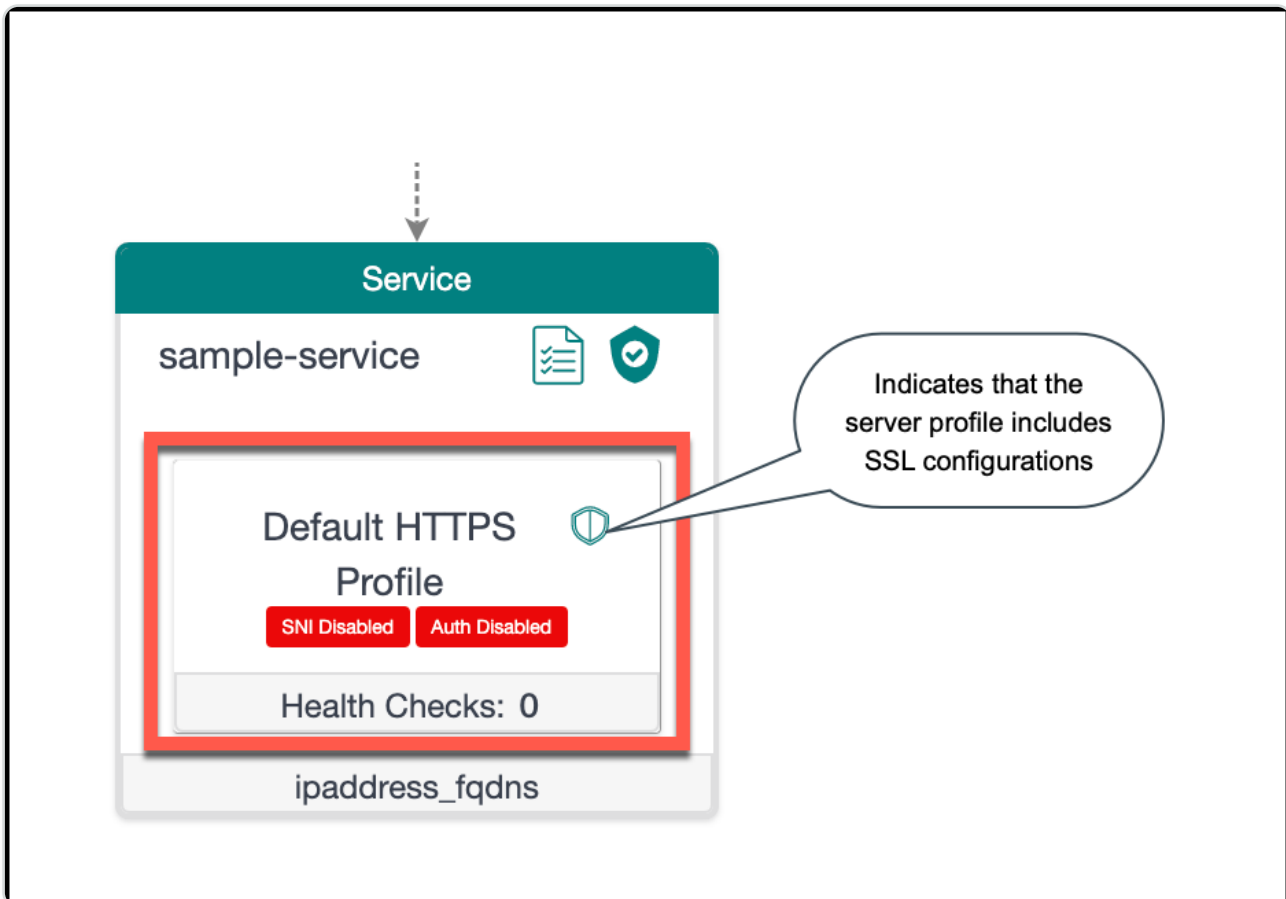
Service Name : sample-service Backend Server Type : ipaddress_fqdns
 Port : 443 Protocol : HTTPS

Server(s):

TYPE	SERVICES	CONFIGURED STATE	ACTIONS
IPAddress	1.1.1.1	●	Get Status
IPAddress	2.2.2.2	●	Get Status

Done

The service profile section displays the name of the service profile and indicates if there are any SSL configurations associated with that profile. You can see if the SNI and server authentication is enabled or disabled. The number of health checks configured is indicated at the bottom of the service profile section.



Click the service profile section to view the following information:

- Name
- If default service profile is used
- Which load balancing algorithm configured
- Maximum connections allowed
- Redirect URL
- SSL details
 - Status of SNI and server authentication (enabled or disabled)
 - Status of TLS versions from 1.0 to 1.3 (enabled or disabled)
 - List of ciphers associated with the SSL profile
 - List of health checks configured and along with protocol and port details

Service Profiles

Service Profile Name :	Default HTTPS Profile	Is Default Service Profile :	Yes
Algorithm :	LEAST_CONNECTIONS	Stickiness Type:	
Max Connections:		Redirect URL :	

Name :	ssl	Server Authentication :	Disabled
SNI Enabled :	Disabled	TLS 1.1 :	Enabled
TLS 1.0 :	Enabled	TLS 1.3 :	Disabled
TLS 1.2 :	Enabled		

Ciphers:

NAME
TLS1-AES-256-CBC-SHA
TLS1-AES-128-CBC-SHA
TLS1.2-AES-256-SHA256
TLS1.2-AES-128-SHA256
TLS1.2-AES256-GCM-SHA384
TLS1.2-AES128-GCM-SHA256

Showing 1 - 6 of 26 items Page 1 of 5

Service profiles

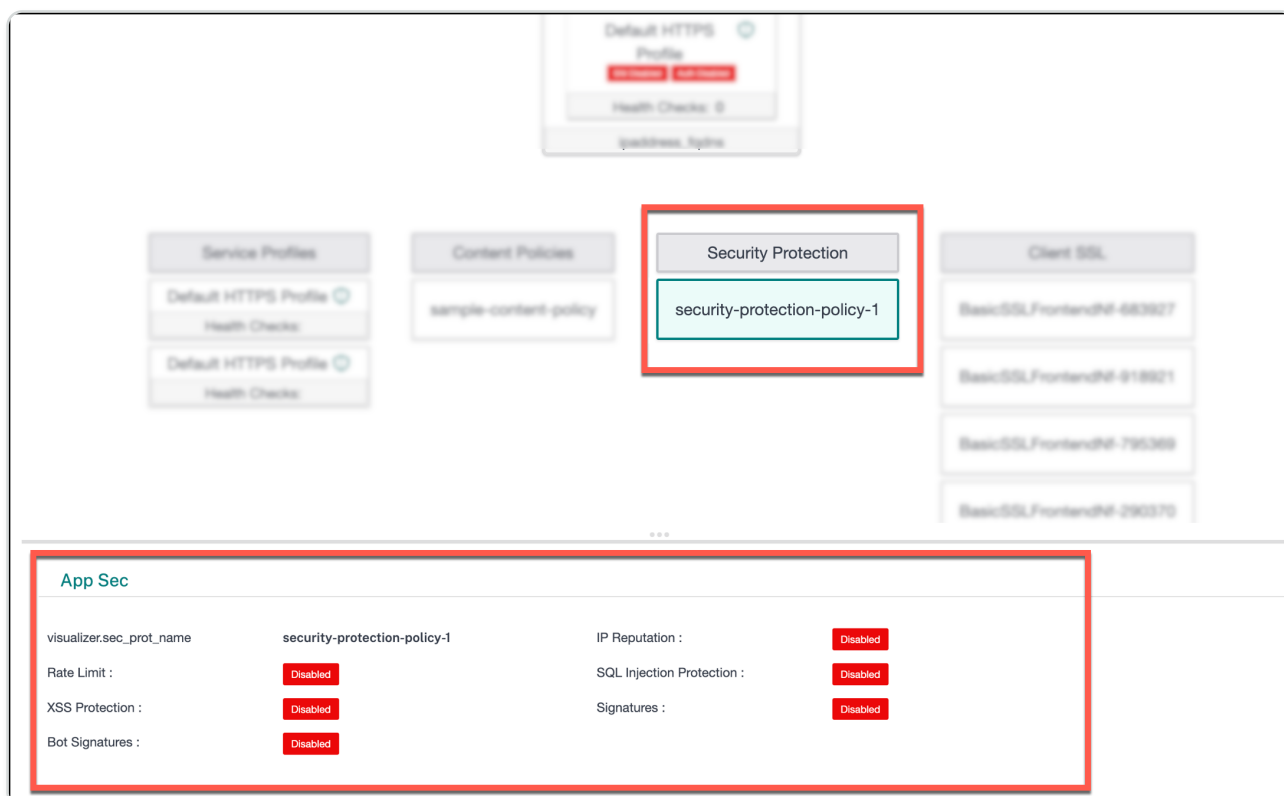
The service profiles section displays the complete list of service profiles that are part of the NetScaler managed application. Only the service profile that is associated with the application gets highlighted when you hover over the **Service** tile. You can click the required service profile and view its details in the lower pane.

The screenshot shows the NetScaler managed application interface. At the top, a 'Service' tile is highlighted, displaying details for 'sample-service'. Below it, a 'Default HTTPS Profile' is shown with 'SNI Disabled' and 'Auth Disabled' status, and 'Health Checks: 0'. The 'ipaddress_fqdns' option is selected. In the lower pane, the 'Service Profiles' tile is highlighted with a red box, showing details for 'Default HTTPS Profile' and 'Health Checks: 0'. Other tiles visible include 'Content Policies', 'Security Protection', and 'Client SSL'.

Application security details

Displays the complete list of security profiles that are part of the NetScaler managed application. Click the required security profile tile to view the following details:

- Name
- Status of the following security features (enabled or disabled)
 - IP reputation
 - Rate limit
 - SQL injection protection
 - Cross-site scripting protection
 - Signatures
 - Bot signatures



Content policy details

Displays the complete list of content policies that are part of the NetScaler managed application. Click the required content policy tile to view the following details:

- Name
- The list of rules including the name, filter, expression, and action.

Service
sample-service

Default HTTPS Profile
SNI Disabled Auth Disabled
Health Checks: 0
ipaddress_fqdns

Content Policies
sample-content-policy
sample-policy-3

Security Protection
security-protection-policy-1

Client SSL
BasicSSLFrontend#683927
BasicSSLFrontend#918921
BasicSSLFrontend#795369

Content Rule

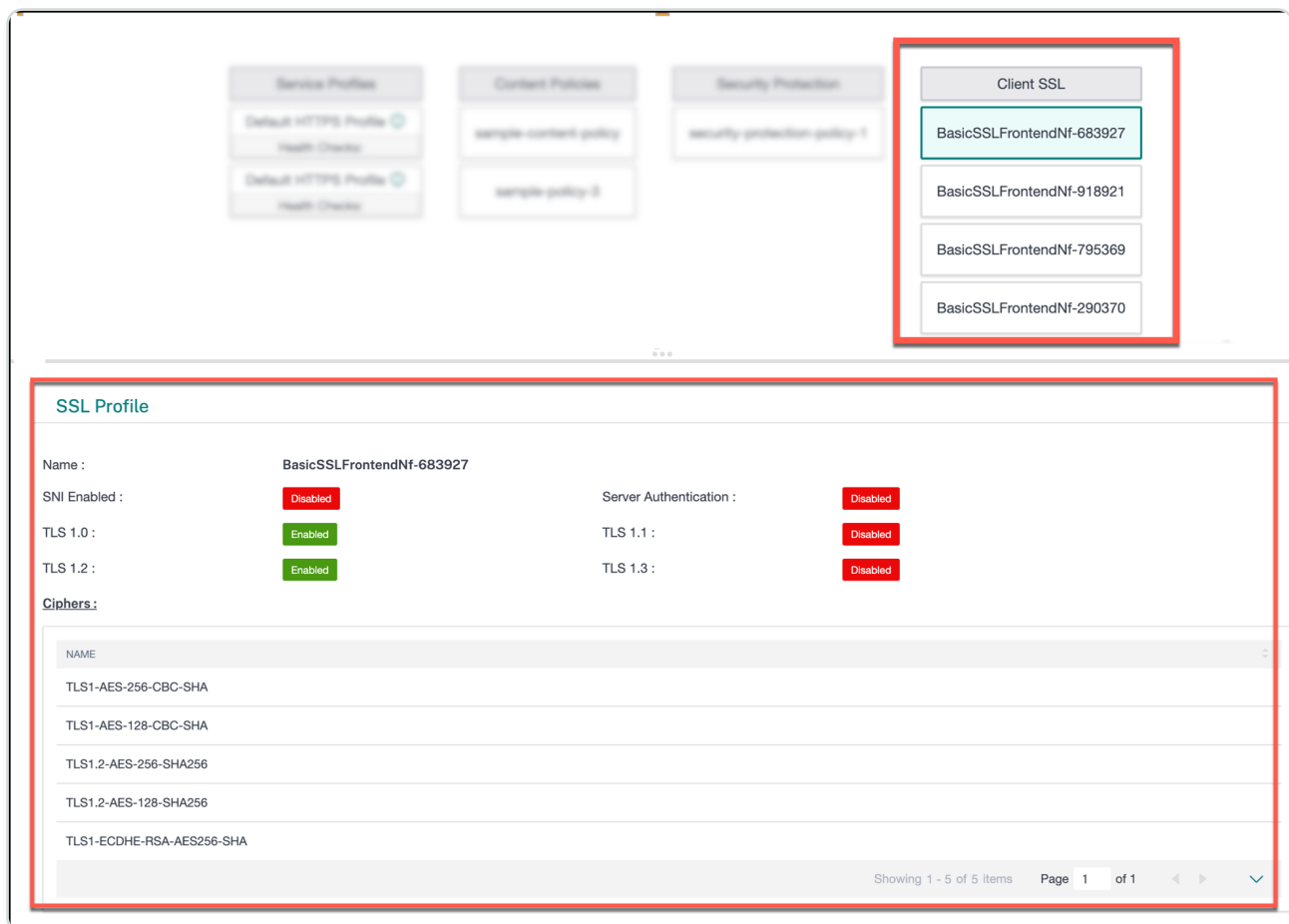
Content Rule Name : sample-content-policy

Rule List :

RULE NAME	FILTER NAME	EXPRESSION	ACTION
sample-rule-3	CR_4184565661990307	{!not equals:{!req_ursuffix:{!suffix:{!sample}}}}	
sample-rule-2	CR_8353614401158602	{!equals:{!req_url:{!path_list:{!testpath/rule}}}}	
sample-rule-1	CR_39261652601863115	{!isvalid:{!http_req:true}}	

Client SSL details

Displays the complete list of SSL profiles that are part of the NetScaler managed application. Click the required SSL profile tile and view its details in the lower pane.



Create services

The services tab comprises services and service profiles. Services represent the back-end servers in your VPC. These servers are the origin servers of your application. Service profiles contain the load balancing and SSL settings for these services. Two service profiles are available by default: **Default HTTP Profile** and **Default HTTPS Profile**. While creating a service, if you select one of these default profiles then creating a service profile is not mandatory. However, if you create a service profile, you must create a load balancer.

1. Click **Create Service**.
2. Specify values for the following parameters:
 - Name
 - Protocol: Select HTTPS to create an SSL service.
 - Port
 - Service Profile: Select an existing profile or create a service profile. For information about creating a service profile, see [Create service profiles](#).

- **App Server Type:** For an AWS environment, select from **Auto Scaling Group**, **EC2 Instances**, or **Servers**. Select an auto-scaling group, EC2 instance, or server type respectively. Only app servers from the selected environment are displayed. For an Azure environment, select from **Azure VM Scale Set**, **Azure VMs**, or **IPAddress/FQDN**. Select a scale set, aVM, or IP address/FQDN respectively.
 - **Server Type/Address Type** includes two options: **IP address** and **FQDN**. Select an option.
 - Specify an IP address or FQDN, and click **Add Server**. The details appear in the **List of Servers** table. You can view the status of the server and also enable or disable the server from the application dashboard. For more information, see [View and change the back-end app server status](#).

3. Click **Create**.

4. Click **Next**.

Self-heal slow application servers

Note: This feature is not currently supported in an Azure environment.

A poorly performing server can affect the performance of your applications. If an Autoscale™ group, or EC2 instances with IP address in AWS is configured, the self-healing capability of the service can detect a slow server and take remedial action accordingly.

For example, if the response time of one of the servers is consistently poor, the service identifies this server, and gracefully replaces it with a healthy server.

The service considers the following conditions while replacing a faulty server:

- All persistent connections are honored.
- Existing connections are completed.
- New connections aren't accepted for the faulty server.

Related actions taken by the NetScaler Console service are logged and admins can view them in the Action History. For more information, see [Monitor self-healing](#).

Configure detecting and auto-replacing a slow server

1. Navigate to **NetScaler® Managed Apps > App Delivery > New Application**.
2. Type a name for the application, and select an environment.
3. Click **Next**.

4. Click **Create Service**.

5. In the **Create Service** page, type values for the following parameters:

- Name
- Protocol
- Port
- Service Profile
- App Server Type - Select either **Auto Scaling Group**, or **Servers**

You can auto-replace a slow server, if the server is added using an IP address and hosted as an EC2 instance in AWS.

If you select **Servers**, select the **IPAddress** in the **Address Type** list, and enter the required IP address in the **IP Address** field. **FQDN** is not supported.

- Select a group from the list.

6. Slide the following toggles to enable the functionality:

- Detect slow server
- Auto-replace slow server

View and change the back-end app server status

The status of the back-end app servers that host the application is determined using health checks. The health checks send probes to the servers and based on the responses received, the status of the server is determined.

The NetScaler Console service allows you to view the status of the back-end app servers and their associated health checks. The back-end app server status information is helpful in troubleshooting. For example, consider a scenario where you are unable to access an application. One of the reasons for this scenario can be that the back-end app server is down or disabled. As part of troubleshooting, you can view the status of the back-end app server.

To view the status of the back-end app server, do the following:

1. Navigate to the **NetScaler Managed Apps > App Delivery** and click the **Analyzer** icon in the **Actions** column.
2. Click the required service tile. The list of back-end app servers associated with the service is listed in the lower pane.

3. Click **Get Status** in the **Actions** column corresponding to the required server.

The status of the back-end app server might change over time. To view the latest status, click **Refresh**.

You can enable or disable the configured state of the back-end server without removing it from the application. For example, consider a planned network maintenance activity that needs to disable a back-end app server. In such a scenario, you can disable the required back-end app server without redeploying the application. You can enable back the server after the maintenance activity is completed.

To disable the back-end app server, click **Disable**. Optionally, select the **Graceful** checkbox and enter the **Delay** in seconds to disable the server gracefully.

NOTE:

After disabling a back-end server gracefully, the status shows **Graceful: Yes** only until the server is shutting down gracefully. Once the server is shut down, the status shows **Graceful: No**.

To enable the server, click **Enable**.

Monitor self-healing

The NetScaler managed application offer self-healing capability for the application server. The self-healing capability provides deep application analytics with improved application experience and security.

With the self-healing capability:

- You can automatically detect, remediate, and replace the defective server with a healthy server whenever the performance of an application server degrades or starts to malfunction.
- You can automatically detect the security attacks and take remediation actions.

The following use cases are supported as part of the self-healing capability:

- **Automatically replace Slow Server:** In a stack of applications running as a back-end server in an Autoscale group, if one of the instances goes faulty and if that instance starts responding slowly based on the response time, then the self-healing feature enables detecting the faulty server and replacing the faulty server with the healthy server automatically.

- **Block past offenders:** Enables you to check and send alerts if the client IPs are causing excessive AppSec violations in the last 20 minutes. For more information, see [SQL injection](#) and [Cross-site scripting](#).

Create service profiles

A service profile is a predefined set of the service-related configurations that includes load balancing, back-end SSL, and health check settings. Every service must have a service profile associated with it.

The NetScaler Console service provides you two default service profiles, **Default HTTP Profile** and **Default HTTPS Profile**. While creating a service, you can associate one of these default profiles or create a profile according to your preference. If you select one of these default profiles then creating a service profile is not mandatory. If you create a service profile, you must create a load balancer. You can define other service-related settings as per your preference.

Configure load balancing

In a real-world scenario with a limited number of servers providing service to many clients, a server can become overloaded and degrade the performance of the server farm. Using load balancing algorithms, you can evenly distribute network traffic to avoid overloading any resource. The result is a better experience for end users in terms of improved performance and availability of your apps.

The load balancing module of the NetScaler Console distributes client requests across multiple app servers. For example, in an AWS environment, the client requests are distributed across EC2 instances, FQDN/IP address-based servers, and Autoscale groups. In an Azure environment, client requests are distributed across Virtual Machines, FQDN/IP address-based servers, and Virtual Machine Scale Sets.

When the NetScaler Console receives a client request, it selects a target server based on the specified rules or routes all traffic to the default server. It uses load balancing criteria to prevent bottlenecks by forwarding each client request to the server best suited to handle the request when it arrives.

Specifying the maximum number of requests per server ensures that a server is not overloaded. You can also specify a URL to redirect traffic, to avoid dropping requests if a server is not reachable.

Prerequisites

1. You have created an environment and a cloud access profile.

2. You have specified the basic details, such as the name of the application and environment by navigating to **NetScaler Managed Apps > App Delivery > New Application**. For more information, see [Deliver an application](#).

Before you begin

Decide which algorithm and stickiness you plan to use for your load balancer. The following algorithms and stickiness are supported.

ALGORITHM

Algorithm is the method used to direct the client request to the server. In the HASH methods, the NetScaler managed application extracts a predetermined portion of the request, creates a hash of the portion, and uses that value to select the server. The following methods are supported:

- **Round Robin:** Requests are distributed across the servers sequentially regardless of the load.
- **Least Connection:** Request is assigned to the server with the least number of active connections depending on the relative compute capacity of the server. This method is the most used method and is also the default method.
- **Least Response Time:** Request is assigned to the server with the lowest average response time.
- **SOURCEIPHASH** - Create a hash of the source IP address.
- **DOMAINHASH** - Create a hash of the domain name, or part of the domain name if it exceeds 80 characters, in the request. The domain name is taken from either the URL or the host header. If the domain name appears in both, the URL is preferred. If the request does not contain a domain name, the algorithm defaults to LEASTCONNECTION.
- **URLHASH:** Create a hash of the request URL, or part of the URL if it exceeds 80 characters.

STICKINESS

The NetScaler managed application use the configured load balancing method for the initial selection of a server. It forwards all subsequent requests from the same client to that same server during a session. This setting is especially helpful in shopping cart and banking transactions. However, a server can become overloaded if it has too many open sessions or a specific session requires more resources.

The following stickiness types are supported:

- **NONE** - No stickiness is configured. Successive requests from the same client are directed to any server. This option is the preferred one for stateless applications. This value is the default value.

- **SESSION COOKIE** - The NetScaler managed application insert an HTTP cookie that is used to direct all requests to the same server during a session. This cookie uniquely identifies the session when a client accesses the application for the first time. It then refers to the cookie for subsequent requests from the same client. The HTTP cookie inserted by the service is a temporary cookie and is erased when the browser is closed. This type of stickiness is typically selected when the stickiness must be valid only until the browser is open.
- **STICKY COOKIE** - The NetScaler managed application insert an HTTP cookie that is valid for the configured duration. Default duration = 2 mins.
- **SOURCEIP** - Connections from the same IP address belong to the same stickiness session. That is, all requests from the same source IP address are forwarded to the same server for the configured duration. Default duration = 2 mins.
- **SSLSESSION** - Connections with the same SSL session ID belong to the same stickiness session. The back-end app server assigns an SSL session ID to each session. All requests with the same SSL session ID are forwarded to the same server for the duration configured. For example, use SSL session ID if you cannot use source IP stickiness because the device is behind a NAT gateway. As a result, the requests might not have the same source IP address. Default timeout= 2 mins.

Create a load balancer

A default load balancer is bound to all services through the service profile. If you create a service profile, you must create a load balancer.

1. Navigate to **NetScaler® Managed Apps > App Delivery > New Application**.
2. Type a name for the application, and select an environment.
3. Click **Next**.
4. Click the **Service Profiles** tab.
5. Click **Create**.
6. In the **Create Service Profile** page, type a name for the profile and click **Create**.
7. In the **Create Load Balancer** page, specify values for the following parameters.
 - Name* - Name for the load balancer.
 - Algorithm*
 - Stickiness*

- **Stickiness Timeout** - Time period, in minutes, for which stickiness is in effect. After the timeout expires, the request is treated as a new request and can be directed to any server depending on the configured load balancing algorithm.
- **Max Connections to Server** - Maximum number of permissible simultaneous open connections per server. This number ensures that the server is not overloaded.
- **Redirect URL** - If the server is DOWN or not reachable due to any reason, the requests are dropped. To avoid such a situation, you can specify a URL to redirect traffic to, if none of the servers are reachable. For example, the redirect might be to a page that simply states that "The servers are not reachable. Try after some time." * Indicates a required parameter

8. Click **Create**.

9. Click **Create**.

10. Click **Next**.

Configure back-end SSL settings

SSL settings on the NetScaler managed application service contain settings related to SNI, server authentication, cipher suites, and protocol versions configured on a back-end SSL application server.

Support for SNI on the back-end application servers

The NetScaler managed application service supports dynamic SNI on the back-end TLS connections. SNI helps to enable SSL encryption on multiple domains if the domains are controlled by the same organization and share the same second-level domain name. For example, *.sports.net can be used to secure domains such as login.sports.net and help.sports.net.

If the back-end server is configured for multiple domains, the server can respond with the correct certificate based on the SNI received in the Client Hello message. The service learns the SNI in the client connection and uses it in the server-side connection. In other words, the common name received in the SNI extension of the Client Hello message is forwarded to the back-end SSL connection.

When server authentication is enabled, the server certificate is verified by the CA certificate and the common name/SAN entries in the server certificate are matched with the SNI. Therefore, the CA certificate must be bound to the service.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application**.
2. Type a name for the application, and select an environment.
3. Click **Next**.

4. Click the **Service Profiles** tab.
5. Click **Create**.
6. In the **Create Service Profile** page, type a name for the profile and click **Backend SSL**.
7. Click **Configure SSL Settings**.
8. In the **Create SSL Policy** page, type a name for the policy, select **SNI**, and click **Create**. The policy is listed on the **Backend SSL** tab in the **Create Service Profile** page.
9. Click **Create**.
10. Click **Next**.

Configure server authentication

Since the NetScaler managed application service performs SSL offload and acceleration on behalf of an application server, the service does not usually authenticate the origin application server's certificate. However, you can authenticate the server in deployments that require end-to-end SSL authentication.

In such a situation, the service becomes the SSL client and carries out a secure transaction with the back-end application server. It verifies that a CA whose certificate is bound to the service has signed the server certificate, and checks the validity of the server certificate.

To authenticate the server, enable server authentication and select the certificate of the CA that signed the server's certificate to the SSL service.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application**.
2. Type a name for the application, and select an environment.
3. Click **Next**.
4. Click the **Service Profiles** tab.
5. Click **Create**.
6. In the **Create Service Profile** page, type a name for the profile and click **Backend SSL**.
7. Click **Configure SSL Settings**.
8. In the **Create SSL Policy** page, type a name for the policy, select **Server Authentication**, and click **Create**.

The policy is listed on the **Backend SSL** tab in the **Create Service Profile** page.

9. Click **Bind CA Certificates**.
10. Click **Create SSL Certificate**.

11. Specify a name for the certificate and choose a certificate file. CA certificates don't need a key.
12. Click **Create**. The certificate is listed on the SSL certificates page.
13. Click **Create**.
14. Click **Next**.

Configure SSL cipher suites and protocol

SSL cipher suites help to establish a secure connection between your app servers and the client. Select the cipher suites to use in your setup.

Protocol version is used during an SSL handshake. Select one or more protocol versions that your app servers support. By default, the highest protocol version that both peers support is selected.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application**.
2. Type a name for the application, and select an environment.
3. Click **Next**.
4. Click the **Service Profiles** tab.
5. Click **Create**.
6. In the **Create Service Profile** page, type a name for the profile and click **Backend SSL**.
7. Click **Configure SSL Settings**.
8. In the **Create SSL Policy** page, type a name for the policy.
9. Click **Select All Cipher** or select individual cipher suites from the list. You can expand each group to view the ciphers that are part of the group and, select or unselect individual ciphers.
10. Select the **Protocol Version**.
11. Click **Create**. The policy is listed in the **Backend SSL** tab on the **Create Service Profile** page.
12. Click **Create**.
13. Click **Next**.

Configure health checks

Health checks monitor the health of servers by sending probes to the back-end app servers. If the server responds within the specified time interval, the state of the server is set to UP. That is, the server can accept traffic. If the server does not respond to the designated number of probes within the specified time period, the server is set to DOWN. No requests are forwarded to this server until its state changes to UP. Traffic is distributed only among healthy servers.

1. In the **Create Service Profile** page, click **Health Checks**.

2. Click **Add Health Check**.

3. Specify values for the following parameters:

- Name* - Name for the health check.
- Protocol - Protocol of the health check and port number used by the server in your EC2 instance or Virtual Machine.
- Port - Destination port where the health checks are sent. If this parameter is not configured, the health checks are sent to the same port as regular traffic.
- Interval - Time interval, in seconds, between two successive health checks to the server. Must be greater than the "Response Timeout" value.
- Response Timeout - The time, in seconds, within which a response must be received from the server. Must be less than the "Interval" value.
- Unhealthy Time Interval - Time interval, in seconds, between two successive health checks to an unhealthy server.
- Failed Check Count- Number of failed health checks after which a server is marked unhealthy.
- Request URL Path* - HTTP request URL path to which health checks are sent. Applicable only to HTTP/HTTPS health checks. A "/" specifies that the monitor checks the server. To check a specific endpoint, specify the URL for the endpoint. For example, if the server is "example.com" then a specific endpoint might be "/example.com/comments.html".
- Request Header - HTTP header to be sent in the health checks. Applicable only to HTTP/HTTPS health checks.
- Response Type* - Expected HTTP response code or text to be present in the HTTP response body to mark the server as healthy. Applicable only to HTTP/HTTPS health checks. * Indicates a required parameter

4. Click **Add**. The configured health checks are displayed in a tabular format.

5. Click **Create**.

6. Click **Next**.

Add endpoints

Endpoints represent the FQDN or the IP address that clients connect to. It can be internal, that is within the VPC. Clients can access the application only within the internal network. If an external endpoint is selected, any client can access the application over the internet.

The FQDN can be user-defined or auto allocated. **User-defined** lets you use AWS Route 53/Azure DNS as a DNS provider to host your application's FQDN. You must have bought and registered a domain with Route 53/Azure DNS already. That is, the zone must be properly acquired from AWS/Azure DNS or delegated to AWS/Azure DNS. **Auto-allocated** lets you use a DNS provider other than AWS Route 53/Azure DNS. The FQDN is displayed after you deploy the application. You must add this FQDN as a CNAME in your authoritative DNS zone configuration. For example, if the registered zone selected is `example.net` and the domain specified is `app1`, then `app1.example.net` is the FQDN that clients connect to access your app.

For more information about AWS, see `<https://docs.aws.amazon.com/acm/latest/userguide/setup-domain.html>`.

For more information about Azure, see `<https://learn.microsoft.com/en-us/azure/app-service/manage-custom-dns-buy-domain>`.

An endpoint must have a default content route associated with it. The route includes the conditions and a target service. If the traffic matches the condition, the request is directed to the specified service.

Choose the clients who can access your application. Select **Internal** to deliver your application privately within the VPC. Only clients within the data center or VPC can access the application. Select **External** to deliver your application publicly. Any client on the internet can access the application.

Also specify the FQDN generation mode. Select **Auto-allocated** if you want to use a DNS provider other than Route 53 or Azure DNS to host your application's FQDN. The auto-allocated application FQDN is displayed in the Application dashboard after the application is deployed. Configure your application FQDN in your DNS provider as a CNAME record by using the auto-allocated FQDN. Select **User-defined** if you want to use Route 53 or Azure DNS as a DNS provider to host your application's FQDN. The application FQDN's DNS zone must already be hosted in Route 53 or Azure DNS. Define an FQDN for the application. It is auto-configured in Route 53 or Azure DNS during application deployment.

You can create or select an endpoint.

Follow these steps to select an endpoint.

1. Navigate to **NetScaler® Managed Apps > App Delivery > New Application**.
2. Specify basic details, such as name of the application, environment, and services. For more information, see [Deliver an application](#).

3. Click **Select**.
4. In the **Select Endpoint** page, select an endpoint from the list and click **Add**.

Follow these steps to create an endpoint.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application**.
2. Specify basic details, such as name of the application, environment, and services. For more information, see [Deliver an application](#).
3. Click **Create**.
4. Specify values for the following parameters:
 - Name
 - Access: Specify **Internal** or **External**.
 - FQDN: Specify **Auto allocated** or **User defined**.
 - Protocol: Specify **HTTP** or **HTTPS**. If you select **HTTPS**, you must add a certificate and optionally, add an SSL policy to get an A+ rating for your applications. For more information, see [Add an SSL certificate](#). You can select one or more certificates, if present, from the list. Select **Auto Redirect HTTP traffic to HTTPS** to ensure that the clients communicate over secure SSL.
 - Port
5. Click **Create**.
6. (Optional) To add a route based on some conditions click **Add**. Specify a name, condition, and a target service. Click **Add**.

Conditions and their corresponding operators and values are listed in the following table:

CONDITION	DESCRIPTION	OPERATORS	VALUES
HTTP Request URL	Identifies an element in the URL portion of an HTTP request.	Contains, Equals, Startswith, Endswith, Not contains, Not equals, Not startswith, Not endswith	URL Path
HTTP Request URL Suffix	Identifies an element in the URL	Contains, Equals, Startswith,	URL Suffix Value

CONDITION	DESCRIPTION	OPERATORS	VALUES
	suffix of an HTTP request.	Endswith, Not contains, Not equals, Not startswith, Not endswith	
HTTP Request Method	Identifies an element in the HTTP request or response by using a method in the HTTP request to evaluate HTTP request data.	Equals, Not equals	GET, PUT, POST, DELETE
Client IP Address	Identifies and returns the client IP address in a TCP/IP packet.	Between, Equals, Insubnet, Not between, Not equals, Not insubnet	IP Addresses, Subnet IP Addresses, Range Start, Range End
Client TCP Address Source Port	Identifies and returns the source port in a TCP/IP packet.	Equals	TCP Source Ports
Client TCP Address MSS	Identifies and returns the maximum segment size (MSS) in a TCP/IP packet.	Greater than	TCP MSS
HTTP Request Header	Identifies the HTTP request data to determine if the data contains a specific header.	Contains, Exists, Not contains, Not exists	Header Names, Header Value
HTTP Request Hostname	Identifies the HTTP request data to determine if the	Contains, Equals, Startswith, Endswith, Not contains, Not equals, Not	Hostname Values

CONDITION	DESCRIPTION	OPERATORS	VALUES
	data contains a specific host name.	startswith, Not endswith	
HTTP API Resource Path	Identifies the URL portion in an HTTP-based API request.	Equals, Not equals, Startwith	URL Path

7. To add a default route, select a service from the **Default Content route** list. Click **Add Default Content route**. Requests that do match any condition are forwarded to this service.

You have completed the steps to create an endpoint. Select from one of the following options:

- Click **Next** to configure a load balancer, content rules, and security protection.
- Click **Deploy** to start application delivery.

Add an SSL certificate

Add an SSL certificate if you select the HTTPS protocol. You can add an SSL certificate in the endpoint workflow or using the **Infrastructure > SSL Dashboard > Certificate Store**.

Add an SSL certificate while adding an endpoint

1. Click **Add SSL Certificate**.
2. In the **Select SSL Certificates** page, click **Create SSL Certificate**.
3. In the **Create SSL Certificate** page, type a certificate name.
4. Browse to the location of the certificate and key file on your computer.
5. If the key is encrypted, add a password.
6. To add the certificate in the certificate chain, select **Add certificate in certificate chain**.
7. Click **Create**.

Get an A+ rating from Qualys Labs

Perform the following actions to get an A+ rating from Qualys Labs for your applications.

1. Click **Add SSL Policy**.
2. In the **Create SSL Policy** page, type a name for the policy.
3. Select **A+ Security**.
4. Click **Create**.

Configure client authentication or mutual TLS (mTLS)

In a typical SSL transaction, the client that is connecting to a server over a secure connection checks the validity of the server. To do so, it checks the server's certificate before initiating the SSL transaction. Sometimes, however, you might want to configure the server to authenticate the client that is connecting to it.

With client authentication enabled, the NetScaler Console service asks for the client certificate during the SSL handshake. The service checks the certificate presented by the client for normal constraints, such as the issuer signature and expiration date.

If the certificate is valid, the service allows the client to access all secure resources. But if the certificate is invalid, the service drops the client request during the SSL handshake.

You can add an SSL certificate at the time of enabling client authentication, or select from an existing certificate in the certificate store.

1. Click **Configure SSL Settings**.
2. In the **Select SSL Settings** page, click **Create**.
3. In the **Create SSL Policy** page, select **Client Authentication** and click **Create**.
4. Select the SSL policy and click **Add**.
5. Add a CA certificate, and then click **Create**.

NOTE:

An SSL certificate is required if the HTTPS protocol is selected.

Configure authentication for the endpoints

Authentication provides security for a distributed internet environment by allowing any client with the proper credentials to connect securely to protected application servers from anywhere on the Internet.

The NetScaler Console service supports SAML based authentication. Also, authentication is only supported for user defined FQDNs and HTTPS applications.

How it works

The SAML IdP (Identity Provider) is a SAML entity that is deployed on the public internet. The IdP receives requests from the SAML SP and redirects users to a logon page, where they must enter their credentials. The IdP authenticates these credentials with the active directory (external

authentication server, such as LDAP) and then generates a SAML assertion that is sent to the SP. The SP validates the token, and the user is then granted access to the requested protected application.

Prerequisites

1. You have created an environment and a cloud access profile.
2. You have specified the basic details, such as the name of the application, environment, services, and endpoints by navigating to **NetScaler® Managed Apps > App Delivery > New Application**.
3. You have configured the required services and end points.
4. You have set up the FQDN type to User defined and Protocol to HTTPS in the endpoint configuration.

To set up authentication for the endpoints (FQDNs/IP addresses)

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application**.
2. Navigate to **Authentication** and click **Create**.
3. In the **Create SAML Identity Provider** page, specify values for the following parameters:
 - **Name** – Name for the SAML IdP.
 - **Metadata URL** – URL of the XML document that contains information about the SAML server.
 - **Issuer Name** – The name to be used in requests sent from NetScaler managed application to IdP to uniquely identify NetScaler managed application.
 - **User field** – User name. SAML user ID, as provided in the SAML assertion.
4. Click **Create**. The IdP you created appears in the Authentication page.

NOTE:

You can add only one IdP per application.

Bind a service to the IdP

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application**.
2. Navigate to **Authentication** and select the preferred services in the **Services** list to bind to the IdP. You can bind the IdP to one or more services as per the requirement. If you select **Select**

All, all the services that are available in the application are authenticated. This is the default option.

3. Select from one of the following options:

- Click **Next** to configure content rules and security protection.
- Click **Deploy** to start application delivery.

After you have enabled SAML authentication, only authenticated users are allowed to access specific services configured by the admin.

Add content policies

Do you want to take a specific action on HTTP requests based on the criteria, such as user location, header, and IP address? NetScaler managed application enables you to respond to HTTP requests depending on the following criteria:

- Render a different page based on:
 - User's geographical location.
 - Browser specification.
 - Languages the browser accepts.
 - Order of preference.

Content policies also help you to do the following:

- Drop the connection if the request is from an IP or IP range that is a suspected malicious user attempting penetration or DDoS attack.
- Redirect traffic, respond with custom messages, or manipulate data on HTTP requests or responses.
- Remove unnecessary HTTP headers, mask internal URLs, redirect webpages, queries, or keywords.

Use content policies to examine the request from the client or response from the server and forward the request depending on the applicable rule. The content policies feature evaluates the data and applies one or more actions based on the evaluation.

Benefits

With content policies, you can do the following:

- Evaluate any part of the header or body of an HTTP or HTTPS request or response and take the configured action based on evaluation. The content policies enable you to configure more actions. For example, transforming the value of an HTTP header in the request or response.
- Redirect the request to a secure webpage. For example, while handling sensitive financial data, ensure that the client uses a secure connection to browse a site by redirecting the request to a secure connection. That is, use `https://` instead of `http://`.
- Show a custom error page instead of the default 404 error page. For example, if you show the home page or site map of the website instead of an error page, the visitor remains on the site instead of moving away from the website.
- Append the default page name to the URL of a website. For example, if the default page of a company's website is `http://www.abc.com/index.php`, when the user types `abc.com` in the address bar of the browser, you can rewrite the URL to `abc.com/index.php`.
- Evaluate basic characteristics of traffic and data. For example, content policies can identify whether an HTTP request or response contains a particular type of header or URL. You can drop a connection or reset a connection at the request level based on the configured rule.

Prerequisites

1. You have created an environment.
2. You have specified the basic details, such as the name of the application, environment, services, and endpoints by navigating to **NetScaler® Managed Apps > App Delivery > New Application**.

Conditions

Conditions evaluate the traffic based on the rules, settings, or string match operation that is defined in the rewrite and responder.

The following are the conditions and their corresponding operators and values:

- **HTTP headers:** - A common method of evaluating the HTTP traffic is to examine the headers in a request or a response.

A header performs the following functions:

- Provide cookies that contain data about the sender.
- Identify the type of data that is being transmitted.
- Identify the route that the data has travelled (through header).

- **Valid HTTP Request** - Identifies the validity of the incoming HTTP request.
- **Valid HTTP Response** - Identifies the validity of the generated HTTP response.
- **HTTP Request Method** - Identifies an element in the HTTP request or response by using a method in the HTTP request to evaluate HTTP request data.
 - Valid operators: **Equals, Not equals.**
 - Value: **GET, PUT, POST, DELETE.**
- **HTTP Request Header** - Identifies the HTTP request data to determine if the data contains a specific header.
 - Valid operators: **Contains, Exists, Not contains, Not exists.**
 - Value: **Header Names, Header Value.**
- **HTTP Response Header** - Identifies the HTTP response data to determine if the data contains a specific header.
 - Valid operators: **Contains, Exists, Not contains, Not exists.**
 - Value: **Header Names, Header Value.**
- **HTTP Request Hostname** - Identifies the HTTP request data to determine if the data contains a specific host name.
 - Valid operators: **Contains, Equals, Startswith, Endswith, Not contains, Not equals, Not startswith, Not endswith.**
 - Value: **Hostname Values.**
- **HTTP Request URL** - Identifies an element in the URL portion of an HTTP request.
 - Valid operators: **Contains, Equals, Startswith, Endswith, Not contains, Not equals, Not startswith, Not endswith.**
 - Value: **URL Path.**
- **HTTP Request URL Query** - Identifies an element in the URL query of an HTTP request.
 - Valid operators: **Contains, Equals, Startswith, Endswith, Not contains, Not equals, Not startswith, Not endswith.**
 - Value: **URL Path.**
- **HTTP Request URL Suffix** - Identifies an element in the URL suffix of an HTTP request.

- Valid operators: **Contains, Equals, Startswith, Endswith, Not contains, Not equals, Not startswith, Not endswith.**
- Value: **URL Suffix Value.**
- **Client IP Address** - Identifies the client IP address. Identify and return a client IP address in a TCP/IP packet.
 - Valid operators: **Between, Equals, Insubnet, Not between, Not equals, Not insubnet.**
 - Value: **IP Addresses, Subnet IP Addresses, Range Start, Range End.**
- **HTTP Request Cookie** - Identifies the HTTP request based on its cookie. Identifies almost any type of information in a cookie header Returns the contents of the HTTP Cookie header.
 - Valid operators: **Contains, Not contains.**
 - Value: **Cookie Name, Cookie Value.**
- **HTTP Response Status Code** - Identifies the response status code from the server. Returns the response status code.
 - Valid operators: **Equals, Not equals, Between, Not between.**
 - Value: **Response Status Codes, Range Start, Range End.**
- **Operators for HTTP Headers** – Operators represent actions that match text and HTTP header values with a collection of static strings in the value.
 - **Equals**- An operation that evaluates whether the condition (target) matches the value (static strings).
 - **Contains** - An operation that identifies if the string argument appears in any instance of the header value. Evaluates whether the target contains any of the strings that are bound to the value.
 - **Startswith** - An operation that evaluates whether the target starts with any of the strings that are bound to the value.
 - **Endswith** - An operation that evaluates whether the target ends with any of the strings that are bound to the value.
 - **Exists** – An operation that evaluates if a request or response contains a particular object.
 - **Between** – An operation that matches the individual strings and sets of strings with any portion of a header value. The header value can be an entire string, the start of a string, or any portion of text in between the start and the end of the string.

- **Not between** - An operation that evaluates the header value that can be an entire string, the start of a string, or any portion of text that is not between the start and the end of the string.
- **Is valid** - An operation that evaluates if a request or response is valid.
- **Insubnet** - An operation that evaluates if a request or response is from a particular subnet.
- **Not insubnet** - An operation that evaluates if a request or response is not from a particular subnet.
- **Not startswith** - An operation that evaluates whether the request or response does not start with any of the strings that are bound to the value.
- **Not endswith** - An operation that evaluates whether the request or response does not end with any of the strings that are bound to the value.
- **Not contains** - An operation that evaluates if the string argument appears in any instance of the header value. Evaluates whether the request or response contains any of the strings that are not bound to the value.
- **Not equals** - An operation that evaluates whether the request or response does not match the value.
- **Not exists** - An operation that evaluates if a request or response does not contain a particular object.
- **Value** - Value is a parameter specific to an HTTP request or response. Evaluate specific data that the HTTP response or request contains.
 - **GET** - Retrieves information from the server.
 - **PUT** - Sends the data to a server to create/update a resource.
 - **POST** - Submits an entity to the specified resource that causes a change in state or effects on the server.
 - **DELETE** – Deletes a resource from the server.
 - **Query Name** - Identifies a specific name that the HTTP request or response contains.
 - **URL Path** - Identifies a specific segment of a URL that the HTTP request contains.
 - **Query Value** - Identifies a specific value that the HTTP request or response contains.
 - **URL Suffix Value** - Identifies a specific suffix value of a URL that the HTTP request contains.
 - **Range End** - Option to set the highest value of its range.
 - **Range Start** - Option to set the lowest value of its range.

- **Subnet IP Address** - Identifies the specific subnet address.
- **IP Address** - Identifies the specific IP address.
- **Header Names** - Identifies a specific header name.
- **Header Value** - Identifies a specific header value.
- **Hostname Values** - Identifies a specific host name value.
- **Response Status Code** - Identifies the status code generated by servers.
- **Cookie Values** - Identifies a specific cookie header that the HTTP request or response contains.
- **Cookie Name** - Identifies a specific cookie name that the HTTP request or response contains.

Operations

If the conditions are met, then the configured operations should be done. Select the required operation and the corresponding fields for the required operation.

The following are the operations and their corresponding operators and values:

- **Forward Client IP with header name X-Forwarded-For** - Forwards the HTTP header **X-Forwarded-For** in an HTTP request with the client's IP address as its value.
- **Forward Port with header name X-Forwarded-Port** - Forwards the HTTP header **X-Forwarded-Port** in an HTTP request with the client's port as its value.
- **Forward Protocol with header name X-Forwarded-Proto** - Forwards the HTTP header **X-Forwarded-Proto** in an HTTP request with the client's protocol as its value.
- **Insert Custom HTTP Header in Request** - A rewrite action to insert a user defined custom HTTP header in a request.
 - Valid Fields: **Custom Header Name, Header Value**
- **Insert Custom HTTP Header in Response** - A rewrite action to insert a user defined custom HTTP Header in a response.
 - Valid Fields: **Custom Header Name, Header Value**
- **Replace an HTTP Header in Request** - A rewrite action to replace an HTTP header in a request.
 - Valid fields: **Header Name, Header Value**

- **Relace an HTTP Header in Response** - A rewrite action to replace an HTTP header in a response.
 - Valid fields: **Header Name, Header Value**
- **Remove an HTTP Header in Request** - A rewrite action to remove an HTTP header in a request.
 - Valid fields: **Header Name**
- **Remove an HTTP Header in Response** - A rewrite action to remove an HTTP header in a response.
 - Valid fields: **Header Name**
- **Custom Header Name** - A string, customized name for the HTTP request or response header.
 - **Header Value** - Custom header string for HTTP request or response header. For example, for a header name "Strict-Transport-Security", the value is "max-age=31536000".
 - **Header Name** - A string, name for the HTTP request or response header.
- **Replace request URL** - A rewrite action to replace the request URL with the specified URL. The NetScaler Console service enables you to convert the URL available in the client request to another URL that the back end server can understand. You can achieve the following benefits by using this rewrite feature:
 - Enhances the security by hiding the actual URL to the resource, which is requested by the client.
 - Prevents the unauthorized user access from gaining access to the network resources.
 - Valid fields: **Enter URL you like to replace.**
 - **Enter URL you like to replace** - The URL that has to be replaced with the request URL. For example, you can change the URLs in the client request temporarily when the website is under maintenance. If the request URL is `http://www.example.com` in the web browser. You can configure the NetScaler Console service to replace the URL in the client request by giving the replacing URL input as `http://www.example.com/resource/inventory/s?t=112`.
- **Redirect to URL** - Modify a URL to redirect traffic from URL A to URL B.
 - Valid fields - **URL value.**

The **URL value** can be an absolute value or a relative value. For example, `https://www.example.com`, modifies the server to which the request is redirected. `/index.html`, redirects the request to the resource `index.html` on the existing server that is configured.

- **Drop the request** - Drop a connection based on the HTTP request-based parameter.
- **Reset the request** - Reset a connection based on the HTTP request or response parameter.
- **Respond with Error** - Sends the chosen error response.
 - Valid Fields: **Choose the Error**. You can choose **DEFAULT** or **CUSTOM**.
 - If you choose **DEFAULT** error, valid fields: **Custom Error Message**.
 - If you choose **CUSTOM** error, valid fields: **Error Code**, **Custom Error Message**.
 - **Choose an Error** - Select the error codes provided in the drop-down list to respond with that error code. You can select from the list of **DEFAULT** errors or select **CUSTOM** to provide a custom error.
 - **Error Code** - Provide an appropriate error code for your custom error.
 - **Custom Error Message** - Send the custom error message by providing an appropriate error message.
- **Respond with an HTML page** - Choose an HTML file to respond with a custom message. You can upload the required HTML page. Use this type of action to send the desired HTML page as the response.
 - **HTML File** - You can upload an HTML page with a custom message during responder run time. Click **Upload File** to choose the required HTML file.

Create or Select a content policy

The content policies feature of the NetScaler Console service examines the request from the client or response from the server, performs action according to the specified rules, and forwards the traffic to the client or the server.

You can create a content policy or select an existing content policy for your service.

Follow these steps to create a content policy.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application**.
2. Navigate to **Content Policies** and click **Create**.
3. In the **Create Content Policy** page, type a unique name for the content policy in **Name**, create a rule and click **Create**.

Follow these steps to select an existing content policy.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application**.
2. Navigate to **Content Policies** and click **Select**.
3. In the **Select Content policies** page, select the required content policy in the list by selecting the check box and click **Add**.

In a content policy you can create:

- A rewrite rule.
- A responder rule.
- Combination of rewrite and responder rules.

NOTE:

You can add only one content policy for an application.

Add a rule

By adding a rule, you can do the following:

- Rewrite the required information in the requests or responses handled by the NetScaler managed application.
- Content filtering functions. NetScaler managed application examines the request from the client, takes action according to the applicable rule, sends the response to the client, and closes the connection with the client.

Follow these steps to create a content policy by adding the rules:

1. In **Create Content Policy** page, click **Add Rule**.
2. In the **Add Rule** page, enter the following parameters:
 - **Rule Name*** – Type a unique name for the rule. Must begin with a letter, number, or the underscore character (_). Must contain only letters, numbers, and the hyphen (-), period (.), hash (#), space (), at (@), equals (=), colon (:), and underscore characters.
 - **If the following condition is met** – Evaluates the traffic based on condition, settings, or string match operation. This tab corresponds to the rule. Only if the conditions are met, the actions are performed. Select the required condition and select or type the corresponding values and operators from the drop-down list. For more information on conditions, operators, and values, see [Conditions](#) section.

Click **AND Condition** to add more than one condition to the same rule, and configure the required parameters according to the required specification. To remove the added conditions, click the **Minimize** icon.

- **Then do the following** - If the conditions are met, then the configured operations are done. For more information on operations, see [Operations](#) section. Select the required operation and select or type the corresponding value fields such as header value, header name.

When you select the required condition, the corresponding applicable operations are highlighted in the drop-down list, and the not applicable actions are grayed out.

1. In the **Add Rule** page, after entering the required parameters, click **Add**.

The created rule is listed in the **Create Content Policy** page and displays the following information:

- **Name** - The content policy name is a unique name for the content policy. Must begin with a letter, number, or the underscore character (_). Must contain only letters, numbers, and the hyphen (-), period (.) hash (#), space (), at (@), equals (=), colon (:), and underscore characters.
- **Order** - The rules are run in the order in which they are listed. The rules with the responder operation category are always performed first. You can change the order of the rules with the same combination of traffic flow and operation category.
- **Traffic Flow** - There are two types of Traffic Flow.
 - **Request** - The direction of the traffic flow which is the request from client forwarded to the server or the NetScaler Console service.
 - **Response** - The direction of the traffic flow which is the response from the server or the NetScaler Console service to the client.
- **Operation Category** - There are two types of Operation category.
 - **Rewrite** - Rewrite refers to the rewriting of required information in the requests or responses handled by the NetScaler Console service. Rewriting can help in providing access to the requested content without exposing unwanted details about the website's actual configuration. Rewrite is used to identify the HTTP data that you want to modify before serving. For example, you can modify the data to mask server information in a response for security purposes.
 - **Responder** - The responder helps in content filtering functions. The responder helps to select the requests to which the NetScaler Console service responds. The responder is

associated with an action, which is performed if a request matches the rule. Responder supports protocols such as TCP, DNS (UDP), and HTTP. With responder enabled in your application, the NetScaler Console service responds on behalf of the server. server responses can be based on who sends the request, where it is sent from, and other criteria with security and system management implications. For example, when users request a webpage, you can provide a different page based on the situation. You can drop the connection if the request is coming from an IP range that has been generating DDoS attacks or initiating hacking attempts.

- **RULE NAME** - Unique name of the rule.
- **OPERATION** - The list of actions that are performed when the conditions are met.
- **ENABLED** - Enable or disable the rule by toggling the **ENABLED** field.
- **ACTIONS** - Denotes the edit and delete action that is configured for that specific content policy.

2. Click **Create**.

The rule is listed in the **Deliver an Application** page. You have completed the steps to create a content policy.

Bind a service

NetScaler Console service enables you to bind the required services to the content policy.

- In the **Deliver an Application** page, select the preferred services in the **SERVICES** drop-down list by selecting the checkbox.

You can view the selected services in the drop-down list under the **Services** tab.

Click **Next** to configure the **Security Protection** feature. Otherwise, click **Deploy**.

Edit a content policy

You can edit the characteristics of an HTTP request or response of a content policy such as host name, URL. You can modify the conditions, actions, and corresponding fields.

The **Deliver an Application** page lists the created content policy with the following details:

- **NAME** - Unique name of the content policy.
- **RULES** - The rule enables the NetScaler Console service to evaluate a piece of traffic or data. For example, a rule can enable the NetScaler Console service to determine whether an HTTP request originated from a particular IP address.

- **SERVICES** - Denotes the services that are associated with the configured content policy.
- **ACTIONS**- Denotes the edit and delete action that is configured for that specific content policy.

Follow these steps to edit a content policy:

1. In the application creation workflow, navigate to **Content Policies** and click **Select**.

The **Select Content policies** page lists the following details:

- **NAME** - Unique name of the content policy.
- **RULES** - The rule enables the NetScaler Console service to evaluate a piece of traffic or data. For example, a rule can enable the NetScaler Console service to determine whether an HTTP request originated from a particular IP address.
- **ACTIONS**- Denotes the **Edit** and **Delete** actions.

2. Click the **Edit** icon in the **Actions** column.
3. In the **Update Content policy- <rule name>** page, click the **Edit** icon.
4. Click **Update** to save the updates.

Edit a rule

1. In the **Update Content Policy- <rule name>** page, select the required rule. Click the **Edit** icon to edit the condition and action.
2. In the **Add Rule** page, modify the required condition, action, and corresponding fields.
3. Click **Update** to save the updates.
4. Click **Update** in the **Update Content Policy- <rule name>** page.

Delete a content policy

Follow these steps to delete a content policy:

1. Select the required content policy in the **Deliver an Application** page, and then click the **Delete** icon in the **Actions** column.
2. Click **Remove**.

The selected content policy is removed from the application.

In the **Select Content policies** page, click the **Delete** icon under the **ACTIONS** column to delete the content policy.

1. Click **Delete**.

The selected content policy is removed from the application.

Delete a rule

1. In the **Update Content Policy** - **<rule name>** page, select the required rule and click the **Delete** icon in the **Actions** column.
2. Click **Delete**. Verify that the rule is removed from the list.

Add security protection

NetScaler Console service provides an intent-based security protection for your web applications and APIs from OWASP Top-10 attacks and exploits. With the security protection, you can define your desired business intent without having to understand the complexities of OWASP Top-10, network, policies, and other application security configuration that the traffic passes through. In addition to the web application firewall and bot management, the service also provides visibility into security insights that provide real-time monitoring of traffic violations and vulnerabilities.

NetScaler Console service supports both 2017 and 2021 versions of OWASP Top 10 security checks. You can enable one or both the checks. OWASP Top 10 2017 is a subset of OWASP Top 10 2021. If you enable OWASP Top 2021, then the security checks belonging to 2017 are enabled by default. If you do not select any version, then the default security checks are enabled.

The following table lists the security checks available under 2017 and 2021 versions.

OWASP TOP 10 - 2017	OWASP TOP 10 - 2021
Allow and block list	Allow and block list
Geo blocking	Geo blocking
IP reputation	IP reputation
SQL injection	SQL injection
Buffer overflow	Buffer overflow
Cookie consistency	Cookie consistency
Cross-site scripting	Cross-site scripting
Command injection	Command injection
Cross-Site Request Forgery (CSRF)	Cross-Site Request Forgery (CSRF)

OWASP TOP 10 - 2017	OWASP TOP 10 - 2021
Field format	Field format
Field consistency	Field consistency
Signatures	Signatures
Data leak prevention	Data leak prevention
-	Rate limit
-	HTTP Security Headers
-	Bot signatures
-	Bot trap
-	Device fingerprint

Enabling the 2017 or 2021 version automatically enables the security checks in the version. Use the **Security Protections** tab to add any exceptions, or to change the configuration.

Benefits

1. Automatically assigns the business intent to security policies.
2. Translates high-level security requests into relevant configuration changes.
3. Simplify deployment and management of security for applications.
4. Get valuable insights and visibility into security events and automate remediation actions.
5. Get continuous protection with auto-update of WAF or bot signature and IP reputation feed.

Prerequisites

Complete the following prerequisites before you add security protection to your application. By default, security protection is disabled.

1. Sign up for Citrix Cloud.
1. Set up application environment.

How security protection works

The security protection has a collection of security checks that protect your application from malicious attacks, software vulnerabilities, SQL database vulnerabilities, errors in the code design, and failures to secure websites that host or can access sensitive information. You can enable and configure security checks to protect your application from various security attacks.

Security check

A security check is a check that inspects malicious or unknown attacks on your web applications. The security check use heuristics, positive security, and other detection techniques to identify attacks. You can configure a security check by first enabling it and then configuring it to block requests, log request details, add exceptions, or define rules to examine traffic.

The security checks are classified into the following categories:

- **Basic security check.** Web security protection applicable to any content type.
- **HTML security check.** Web security protection applicable only to HTML-based websites and to HTML portions of Web 2.0 sites that contain both HTML and XML content.
- **JSON security check.** Web security protection applicable only to JSON-based websites and to JSON content type in Web 2.0 sites.

Signatures

Signatures are configurable rules to simplify the task of protecting your websites from attacks. A signature represents a pattern of a component attack on an operating system, web server, website, or other resources.

The Security Protection has a built-in default signature object consisting of more than 1,300 signature rules. The Security Protection and Bot signatures can be customized by adding new rules. A signature rule can have multiple patterns and can flag a violation only when all the patterns are matched to avoid false positives.

Actions

All security checks have a set of configurable actions which control how Security Protection can handle a request or response.

BLOCK ACTION

The block action prevents vulnerabilities from attacking your application. By default, security checks have the block action disabled. You must enable if you want to block a request that contains a malicious attack. If block action is not enabled, the NetScaler Console service only logs the request data.

LOG ACTION

The log action enables you to collect request details that have a malicious attack. The log details can be further investigated for monitoring and analytics purposes. When you enable a security check that the log-only action is auto-enabled for the feature.

Exception (rule)

For security check, an exception is a rule written to allow access to specific data and block the rest. Exceptions are applicable only for a few security checks such as SQL injection, cross-site scripting, Cross-Site Request Forgery (CSRF), Field Consistency, and Cookie Consistency. These rules are written to bypass security validation for false-positive scenarios.

You can configure an exception to bypass security checks for legitimate requests. A security check is performed on the incoming traffic payload and malicious requests, patterns are identified even if they are spread across multiple lines. When examining the traffic, you can apply exceptions to bypass the security check for the following criteria:

- Incoming traffic with SQL injection attacks that are malicious.
- Incoming traffic with cross-site scripting attacks that are vulnerable to your web applications.
- Incoming traffic with cookie consistency attacks to gain access to a legitimate session of the target user.
- Incoming traffic with field consistency attacks that send unauthorized web from data.
- Incoming traffic with CSRF attacks induces a user to perform actions that are not intended to perform.

The security protection not only enables you to create an exception but also manages exceptions for the security check through the GUI or API interface.

Example use case 1: Exception for Cross-Site Scripting protection:

Consider a form field `dpasswd` in the URL, `https://adcsvc.example.com/login.php` not prone to cross site scripting. The corresponding field can be exempted from the cross-site scripting protection. To configure this exception, you can add a rule in which the URL pattern must be defined as `https://adcsvc.example.com/login.php(regex)`, field type must be `Form Field`, and form field name must be `"\^dpasswd$(regex)"`.

SAMPLE PAYLOAD	URL	FIELD TYPE	FORM FIELD
<pre>https:// adcsvc.example.co m/login.php? uname=test&dpassw d= foo<script>alert(document.cookie)< /script></pre>	<pre>https://adcsvc\ example\.com/ login\.php</pre>	Form Field	\^dpasswd \$

Example use case 2: Exception for cookie consistency validation:

False positives occur when a cookie mismatch is incorrectly flagged as a vulnerability by a scanning tool. To handle such scenarios, you can add an exception for the header cookie that can be exempted from cookie validation. Consider a client sending a login request to the server. On successful login, the server response includes the `set-header` cookie that contains the `JSESSIONID` and its value.

```
Set-Cookie: JSESSIONID=abc123; Path=/; HttpOnly
```

The client sends the cookie ID in its subsequent request to the server. To handle a false positive or legitimate request, you can add an exception to bypass cookie validation for `JSESSIONID` for any request coming from the client.

Conditions

Conditions are characteristics that you want to examine in an incoming request. Following are some conditions for which the NetScaler Console service evaluates the traffic.

1. Malicious scripts embedded in a request.
2. IP addresses or address ranges that requests originate from.
3. Country or geographical location that requests originate from.
4. Length of a specified portion of the request, such as the query string.

Get started with NetScaler Console service security checks

The NetScaler Console service security protection features are classified into three categories, General, WAF, and Bot.

General

- **Allow and block list**—Inspects the incoming traffic with rules defined to Allow or Block a client's access to web application resources, based on parameters such as IP address, subnet, or HTTP request headers.
- **Geo blocking**—Restricts unauthorized access based on user's geographical location.
- **IP reputation**—Blocks traffic coming from an IP address with a bad reputation score
- **Rate Limit**—Limits the amount of network traffic. For example, if the feature limits the number of incoming requests to 100, then more requests to the web application are blocked. Rate limiting controls traffic flow based on client IP address, client subnet, URL, or URL pattern, and HTTP headers.

WAF

- **SQL injection**—Examines HTTP or HTTPS requests and blocks HTML-based, and JSON-based SQL attacks that can affect your web applications.
- **Buffer overflow**—Examines if an incoming traffic can cause a buffer overflow on the web server. If the service detects a URL, a cookie, or a header longer than the specified maximum length in a request, it blocks a JSON-based request or HTML based requests to prevent a buffer overflow attack.
- **Cookie Consistency**—Examines the cookie sent by the browser and blocks the request if the cookie does not match the cookie in the web application. To resolve a false positive and bypass the security check, you can add an exception.
- **Cross-site scripting**—Examines if an attacker inserts malicious code into an incoming request that can have huge implications, such as compromising website security or user authentication. To resolve a false positive and bypass the security check, you can add an exception.
- **Command Injection**—Examines if an incoming request has any vulnerable commands. To resolve a false positive and bypass the security check, you can add an exception.
- **CSRF**—Inspect HTTP or HTTPS requests and blocks HTML-based, or JSON-based CSRF attacks that can affect your web applications.
- **Field Format**—Examine both the length and type of web form data to ensure that it is appropriate for the field. If inappropriate web form data is found in a user request, you can configure the NetScaler Console service to block the request.
- **Field Consistency**—Examines web forms returned by users for your application and verifies that web forms were not modified inappropriately by the client. The security check is only for HTML requests that contain a web form, with or without data.

- **WAF Signatures**—A list of signature rules is available under different categories to protect your web application against security attacks. The NetScaler security protection examines the traffic and blocks the request if it matches a signature pattern. By default, all signatures are enabled.
- **HTTP Security Headers**—HTTP response headers can be used to prevent security attacks on the web application by controlling the browser behavior during application access.
- **Data Leak Prevention**—Identifies if the responses coming from the webserver contain any user-specific sensitive information. When a match is found, the NetScaler Console service takes the configured action to avoid leakage of such information.

Bot

- **Bot Signatures**—A list of bot signature rules is available under different categories to protect your web application against security attacks. The NetScaler security protection examines the traffic and blocks the request if it matches a signature pattern. By default, all bot signatures are enabled.
- **Bot Trap**—Detects and blocks automated bots by inserting a random trap URL in the server response.
- **Device Fingerprint**—Detects the incoming traffic as a bot by inserting a JavaScript script in the HTML response to the client.
- **Bot TPS**—Detects the incoming traffic as a bot, based on the number of transactions.

Before you enable a security check, you must create a security protection profile or use an existing security profile.

Add a security protection profile

If you already have a security protection profile, you can select and add the profile for your application, otherwise you can create a profile.

Complete the following steps to add a security protection by using the GUI:

1. Navigate to **NetScaler® Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Select**.
3. In the **Add Security Protection** slide, select a security protection profile and click **Add**.
4. The profile is added to the security protection summary table.
5. Select and bind the profile to a service.

NOTE:

You can add only one security profile to an application service.

The security protection profile is now added to the application before it is ready to be deployed in the cloud. If you do not see an existing profile in the list, you can create a one.

Create a security protection profile

As a first-time user, you can create a profile for your application. Complete the following steps by using the GUI.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Select 2017 or 2021 OWASP Top 10 security protection. Based on the version selected, a predefined set of security checks are enabled.
4. Click **Create**.

You can view the newly created security protection profile.

5. Select and bind the profile to a service.

NOTE:

You can add only one security profile to an application.

The profile is added to your application.

Edit a security protection profile

If you already have a security protection profile and you prefer to update its details, you can use the edit functionality.

Complete the following steps to update a security protection profile by using the GUI:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Select**.
3. In the **Add Security Protection** slide, select a security protection profile and click the pencil icon to update.

4. In the **Update Security Protection** page, modify details and click **Update**. The **Add Security Protection** page displays the updated profile.
5. Select the profile and click **Add**. The profile is added to the security protection summary table.
6. Select and bind the profile to a service.

Allow and block list

The Allow and block list functionality enables you to create a security rule to allow or block requests based on user parameters such as IP address, subnet address, or HTTP header.

Allow list. A rule that allows user requests to access internal resources if all conditions configured for the rule match the request.

Block list. A rule that blocks user request to internal resources if all conditions configured for the rule match the request.

Enable allow and block list protection

Before you create a security rule, as a first step, you must enable the security check toggle. If you disable the toggle, the NetScaler Console service does not inspect the traffic.

Create a rule

As a first-time user, if you do not see a security rule for allow or block list, you can create one. Complete the following steps to create a rule by using the GUI.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **Allow and Block list**.
4. Select **Enable Allow and Block list protection** and click **Add Rule**.
5. In the **Allow and Block list** section, set the following parameters:
 - a. Rule name. Name of the rule for an allow list or a block list.
 - b. If the following conditions are met. The conditions to define based on user parameters such as IP address, subnet address, or HTTP header. You can add multiple conditions for a rule using either **AND Condition** or **OR Condition**. A combination of **AND Condition** and **OR Condition** is not supported. The NetScaler Console service security evaluates traffic based on the conditions that you define.
 - i. HTTP Request Method. Request method for create, read, update, or delete operations.

- i. Operator. Logical condition to evaluate the HTTP request method.
 - ii. Value. HTTP request method operation.
- ii. HTTP Request Header. Header used in an HTTP request to provide information about the request.
 - i. Operator. Logical condition to evaluate the value of specific headers.
 - ii. Name. Name of the HTTP request header field.
 - iii. Value. Request header field value.
- iii. HTTP Request host name. HTTP host header that has the host name of the client request.
 - i. Operator. Logical condition to evaluate the request host name.
 - ii. Value. Host name value.
- iv. HTTP Request URL. HTTP request made by a client to a named host located on the back-end server.
 - i. Operator. Logical condition to evaluate the request URL.
 - ii. Path. Request URL path.
- v. IP Allow and Block list.
 - i. Operator. Logical condition to evaluate the IP address of the client request.
 - ii. Value. IP address or subnet address.
- c. Then do the following Action. Action to apply based on evaluation. Action types are as follows:
 - i. Allow. Allow user request to access internal resources.
 - ii. Drop. Drop user request without sending a response.
 - iii. Reset. Reset client connection by closing it.
- d. Click **Add Rule**.

You can view the newly created rule on the **Allow and Block list** page.

Click the pencil icon to modify rule details.

Edit an allow or block rule

Complete the following steps to update a rule by using the GUI:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click the pencil icon under the **Actions** column corresponding to the security protection that you want to edit.
3. Navigate to the **Security Protections** tab and click **Allow and Block list**.
4. In the **Allow and Block list** page, click the pencil icon in the **Actions** column corresponding to the required rule.
5. Edit the rule as required and click **Update Rule**.

Geo blocking

The geo blocking security check enables you to allow or block requests based on a geographical location from where the request originates from. The filter works based on a geo match condition that has a list of countries in an allow list and a block list. If a request matches a geo match condition, then the request is either allowed or blocked based on the location the request originates from.

Enable geo blocking protection

Before you configure geo-blocking, you must enable the security check. When disabled, the NetScaler Console service does not allow, or block requests based on the client's geo location.

Classify geo locations

Complete the following steps to classify geo locations under an allowed or blocked country list:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **Geo Blocking**.
4. Select **Enable Geo Blocking**.
5. In the **Geo Blocking** section, classify countries under **Allowed Countries** or **Blocked Countries** list.

IP reputation

IP reputation security check blocks unwanted requests coming from an IP address that has a bad reputation score. The security check enables you to classify IP address threat categories as an allowed list and blocked list. If a request matches a condition, the request is either allowed or blocked based on its IP address and its threat category. Some of the key threat categories are spam sources, Windows exploits, botnets, scanners, and so forth.

Enable IP reputation protection

Enable the IP reputation security check to examine traffic for the IP reputation. When disabled, the NetScaler Console service does not inspect the traffic for its IP address and threat category.

Classify IP address threat category

Complete the following steps to classify the threat categories as an allowed list or a block list:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **IP Reputation**.
4. Select **Enable IP Reputation**.
5. In the **IP Reputation** section, classify the threat categories under allowed list and blocked list.

Rate limit

The rate limit security check limits network traffic and protects your application from security vulnerabilities and bot attacks. The filter limits traffic by restricting the number of user requests that an application can receive within a time frame.

The filter uses rate limit conditions that measure the number of requests received within a time frame. If there are too many requests within a time frame, the check blocks requests that exceed the threshold limit. By doing this, the NetScaler Console service protects your application from excess traffic.

You can either select a single condition using the **Limit Requests** option or select multiple conditions using the **Add Conditions** option.

Example use case: Rate limiting requests per URL for a ticket booking website Consider a ticket booking website `www.adcsvc.example.com` that receives huge traffic every 2 minutes. To limit traffic to the website, you can limit the number of requests per URL to 200 for 120 seconds. If

there are more than 200 requests received per URL within 120 seconds, then the rate limit security check validation fails, and the check blocks the excess requests per URL.

Enable rate limit protection

Before you configure the rate limit policy you must enable the security check. When disabled, the NetScaler Console service does not rate limit traffic to your application.

Configure rate limit policy

Complete the following steps to configure the rate limit policy for enhanced application security.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **Rate Limit**.
4. Select **Enable Rate Limit**.
5. In the **Rate Limit** section, click **New Policy**.
6. In the **Configure Rate Limit Policy** page, set the following parameters:
 - a. Policy Name. Name of the rate limit policy.
 - b. Limit Requests. Logical condition to rate limit traffic.

Server

- For overall app. Logical condition to rate limit traffic on the server-side.
- Per URL. Logical condition to rate-limit traffic if it matches the pre-defined URL on the server-side.
- Per URL matching the pattern. Logical condition to rate-limit traffic only if the incoming request URL exactly matches the pre-defined URL on the server-side.
- Matching the URL pattern. Logical condition to rate-limit traffic only if the incoming request URL matches the URL syntax pattern on the server-side.

Client

- Per client IP. Logical condition to rate limit traffic based on client IP address.
- Per client matching the following client pattern. Logical condition to rate-limit traffic if the response URL matches the predefined URL on the client side.
- Matching the client pattern. Logical condition to rate-limit traffic if the response URL matches the predefined client or subnet address.

- Per client identified by a specified HTTP header. Logical condition to rate-limit traffic per client identified by a specific HTTP header.
- Per client identified by values in an HTTP header. Logical condition to rate-limit traffic per client identified by a specific HTTP header value.
- Per client identified by a cookie. Logical condition to rate-limit traffic per client identified by a cookie.
- Per client identified by a user agent. Logical condition to rate-limit traffic per client identified by a user agent.

Server and Client

- Per client matching the specific URL. Logical condition to rate-limit traffic per client matching exactly the URL.

c. Add Conditions. (Optional) A list of logical conditions to rate limit traffic.

Condition	Description	Operators	Values
HTTP Request URL	Identifies an element in the URL portion of an HTTP request.	Contains, Equals, Startswith, Endswith, Not contains, Not equals, Not startswith, Not endswith	URL Path HTTP Request URL Query
HTTP Request URL Query	Identifies an element in the URL query of an HTTP request.	Contains, Equals, Startswith, Endswith, Not contains, Not equals, Not startswith, Not endswith	Query Name, Query Value
HTTP Request URL Suffix	Identifies an element in the URL suffix of an HTTP request.	Contains, Equals, Startswith, Endswith, Not contains, Not equals, Not startswith, Not endswith	URL Suffix Value
HTTP Request Method	Identifies an element in the HTTP request or response by using a method in the HTTP request to evaluate HTTP request data.	Equals, Not equals	GET, PUT, POST, DELETE
Client IP Address	Identifies the client IP address. Identify and return a client IP address in a TCP/IP packet.	Between, Equals, Insubnet, Not between, Not equals, Not insubnet	IP Addresses, Subnet IP Addresses, Range Start, Range End
HTTP Request Header	Identifies the HTTP request data to determine if the data contains a specific header.	Contains, Exists, Not contains, Not exists	Header Names, Header Value
HTTP Request Hostname	Identifies the HTTP request data to determine if the data contains a specific host name.	Contains, Equals, Startswith, Endswith, Not contains, Not equals, Not startswith, Not endswith	Hostname Values
HTTP Request Cookie	Identifies the HTTP request based on its cookie. Identifies almost any type of information in a cookie header	Returns the contents of the HTTP Cookie header.	Contains, Not contains
Cookie Name, Cookie Value			

NOTE:

When the **Add Conditions** option is selected, the preceding list of conditions is displayed and the **Limit Requests** drop-down menu list only the three basic options, **For overall app**, **Per URL**, and **Per client IP**.

- d. Set maximum requests to be allowed within a given time frame.
- e. Set the time frame in seconds.
- f. Set the **Limit type**:

Bursty: Use this limit type if your application traffic is sporadic. It is helpful if the load peaks anytime within the set time frame.

Smooth: Use this limit type if your application traffic is consistent. It evenly spreads the load across each time slice of the set time frame.

For example, the set maximum requests are 100 and the time frame is 10 seconds. If your application receives 80 requests in the first second, these limit types behave differently.

The bursty limit type allows the requests to pass through because the load is below the set threshold. However, the smooth limit type evenly spreads the load across the set time frame. In this scenario, it can only allow 10 requests per second. So, it applies the configured action for the excess load.

- g. Configure any of the following actions for subsequent requests that exceed the limit.
 - **Drop** - Connections are silently dropped.
 - **Reset** - The connection with the client is reset.
 - **Respond with 429** - Display the status code 429 to a user. This code suggests that there are too many requests for the application.
- h. Status. Enable or disable rate limit policy configuration.

7. Click **Add Policy**.

You can view the policy in the **Rate Limit** section.

Logging settings

Logging settings collects more information about the payload than what is collected in a regular log. It can collect verbose logs such as the log pattern, pattern payload, and HTTP header details.

The payload information gives you more context while troubleshooting issues. For example, if a violation is detected, you can look at the request that triggered the violation.

To set logging settings, complete the following steps:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **Logging Settings**.
4. Select the verbose log type.
5. Click **Create**.

The following table lists the verbose log types supported for WAF and Bot traffic:

VERBOSE LOG TYPE	WAF TRAFFIC	BOT TRAFFIC
Pattern	Logs only the violation pattern	-
Pattern Payload	Logs violation pattern and 150 bytes of extra payload	-
Pattern Payload and Header	Logs violation pattern, 150 bytes of extra payload, and HTTP header information.	Logs full HTTP header information such as domain address, URL, user-agent header, cookie header.

View verbose logs

Complete the following steps to see verbose logs:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Select **WAF** or **Bot** tab based on which traffic type that you want to see verbose logs.
3. Click the required application and click the link under the **Request URL** column.

SQL injection

The SQL injection filter examines the incoming request for SQL Injection attacks. If an SQL attack is detected in the payloads, the security check blocks the request. Use the SQL protection check to secure your web application and prevent SQL injection attacks.

Enable SQL injection protection

Before you configure the SQL injection exception, you must first enable security protection for HTML or JSON content type requests. When disabled, the NetScaler Console service does not inspect the traffic for SQL injection attacks. Also, you must toggle the Block action to block requests that contain malicious SQL attacks. When the block action is not selected, NetScaler Console service only logs the data.

INTENT BASED AUTO BLOCKING

The **Block clients with 20 violations within 30 minutes** option allows you to define the intent of auto blocking malicious clients that are attempting SQL injection attacks.

If the number of requests violating the security check received from a client increases to 20 or more in a 30-minute duration, the client is considered malicious. Select **Block clients with 20 violations within 30 minutes** to block such clients for a 30-minute duration.

Create an SQL injection exception

An SQL injection attack can occur in several parts of a request. The variants of an SQL injection attack are:

- **Form fields.** A basic SQL injection attack can be through a login page where the user provides input. The web application accepts inputs through the page, which pass the user input to the database for processing. If the web application accepts inputs without sanitizing them, an attacker can inject SQL statements through the form fields that can delete, copy, or modify contents in the database.
- **Header.** Server variables such HTTP headers can also be vulnerable for SQL injection. If a web application accepts inputs from HTTP headers, fake headers containing arbitrary SQL can inject code into the database.
- **Cookie.** Cookies are another vulnerability for SQL injection. This is done by modifying cookies to affect the database queries. Web applications often load cookies and use the data for database operations. A malicious user, or malware deployed on a user's device, can modify cookies, to inject SQL in an unexpected way into the back-end database.

To avoid false positives and bypass a security check, you can add an SQL injection exception for form fields, header, or cookie values

Complete the following steps to add an exception.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.

2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **SQL Injection**.
4. Select the required content type.
5. Click **New exception** and set the following parameters to add an exception for SQL injection protection:
 - a. URL Pattern. The URL for which the exception is required. It can be a regular expression. For any URL enter '.*'.
 - b. Field type. The field type for which the exception is required.
 - c. Form field names. The form field in which the exception is required. It can be a regular expression.
 - d. Status. Enable or disable the SQL injection exception.
6. Click **Add Exception**.

You can view the exception in the **SQL Injection** section.

Edit an SQL injection exception

Complete the following steps to update an SQL injection exception:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click the pencil icon under the **Actions** column corresponding to the security protection that you want to edit.
3. Navigate to the **Security Protections** tab and click **SQL Injection**.
4. In the **SQL Injection** page, click the pencil icon in the **Actions** column corresponding to the required exception.
5. Edit details and click **Update Exception**.
6. You can view the updated exception in the SQL Injection summary section.

Buffer overflow

The buffer overflow security check detects if your application receives more input than expected that can cause a buffer overflow on the application server. For example, let us consider the origin server is set to handle a URL length of maximum of 500 characters but if an incoming URL length is 700 characters, the buffer overflow security check validation fails, and it blocks the request. To prevent buffer overflows, the NetScaler Console service security check enables you to set the

maximum length for request parameters such as URL, cookie, or header value. If the input value for these parameters is longer than the profiled length, then the check blocks the request because it can cause an overflow.

Enable buffer overflow protection

Before you configure the buffer overflow limit, you must enable the security check. When disabled, the NetScaler Console service does not inspect the traffic for buffer overflows. Also, you must enable the Block toggle to block requests that can cause a buffer overflow. When the block toggle is not selected, the NetScaler Console service only logs the data.

Configure buffer overflow limit

You can prevent the buffer overflow vulnerability by configuring the maximum length for parameters such as URL, cookie, or header length.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **Buffer Overflow**.
4. Select the required content types.
5. In the **Buffer Overflow** section, set the following parameters:
 - a. **Maximum Container Depth**. Enter the maximum allowed nested depth for a JSON element. Requests with container depth greater than the configured value are blocked. Default: 5, minLength: 0, maxLength: 127.
 - b. **Maximum Document Length**. Enter the maximum allowed document length in a JSON request. Requests with document size greater than the configured value are blocked. Default: 20000000, minLength: 0, maxLength: 2147483647.
 - c. **Maximum Object Key Count**. Enter the maximum allowed object key count in a JSON request. Requests with object key count greater than the configured value are blocked. Default: 10000, minLength: 0, maxLength: 2147483647.
 - d. **Maximum Object Key Length**. Enter the maximum allowed length for an object key in a JSON request. Requests with object keys of length greater than the configured value are blocked. Default: 128, minLength: 0, maxLength: 2147483647.
 - e. **Maximum Array Length**. Enter the maximum length for an array in a JSON request. Requests with array length greater than the configured value are blocked. Default: 10000, minLength: 0, maxLength: 2147483647.

- f. **Maximum String Length.** Enter the maximum allowed length for a string in a JSON request. Requests with json string length greater than the configured value are blocked. Default: 1000000, minLength: 0, maxLength: 2147483647.

Cookie consistency

The cookie consistency filter examines if the cookie returned by the user matches the cookies set by the website. If a modified value is found, the cookie is stripped from the request before forwarding it to the application server.

Enable cookie consistency

Before configuring the cookie consistency security check, you must first enable the feature. When disabled, NetScaler Console service does not inspect the traffic for cookie consistency attacks. Also, you must toggle the Block action to block requests that contain malicious SQL attacks. If the block action is not selected, NetScaler Console service only logs the data.

Create a cookie consistency exception

You can create an exception to prevent blocking legitimate requests or resolve false positives and bypass the security check. Complete the following steps to add an exception.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **Cookie Consistency**.
4. Select **Enable Cookie Consistency Protection**.
5. Click **New Exception** and set the following parameters to add an exception for cookie consistency:
 - a. **Cookie Name.** Name of the cookie set by the server.
 - b. **Status.** Enable or disable cookie consistency exception.
6. Click **Add Exception**.

You can view the exception in the cookie consistency summary section.

Edit a cookie consistency exception

Complete the following steps to update a cookie consistency exception:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.

2. Click the pencil icon under the **Actions** column corresponding to the security protection that you want to edit.
3. Navigate to the **Security Protections** tab and click **Cookie Consistency**.
4. In the **Cookie Consistency** page, click the pencil icon in the **Actions** column corresponding to the required exception.
5. Edit details and click **Update Exception**.
6. You can view the updated exception in the **Cookie Consistency** section.

Cross-site Scripting

In a Cross-Site Scripting (cross-site scripting) attack, an attacker inserts malicious code into an incoming request that can have huge implications, such as compromising website security or user authentication. Use the cross-site scripting feature to secure your web application and prevent cross-site scripting attacks.

Enable Cross-site Scripting protection

Before you configure the Cross-site Scripting exception, you must first enable the security check for HTML or JSON content type requests. When disabled, the NetScaler Console service does not inspect the traffic. Also, you must enable the Block toggle to block requests that contain malicious Cross-site scripting attacks. When the block is not selected, NetScaler Console service only logs the data.

INTENT BASED AUTO BLOCKING

The **Block clients with 20 violations within 30 minutes** option allows you to define the intent of auto blocking malicious clients that are attempting cross-site scripting attacks.

If the number of requests violating the security check received from a client increases to 20 or more in a 30-minute duration, the client is considered malicious. Select **Block clients with 20 violations within 30 minutes** to block such clients for a 30-minute duration.

Create a cross-site scripting exception

You can add a Cross-site Scripting exception for form field, header, and cookie values to prevent blocking legitimate requests or resolve false positives, and bypass the security check. Complete the following steps to add an exception.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.

3. Navigate to the **Security Protections** tab and click **Cross-site Scripting**.
4. Select the required content types.
5. Click **New Exception** and set the following parameters to add the cross-site scripting exception:
 - a. URL Pattern. The URL for which the exception is required. It can be a regular expression. For any URL enter '. *'.
 - b. Field type. The field type for which the exception is required.
 - c. Form field names. The form field in which the exception is required. It can be a regular expression.
 - d. Status. Enable or disable the cross-site scripting exception.
6. Click **Add Exception**.

You can view the exception on the cross-site scripting summary section.

Edit a cross-site scripting exception

Complete the following steps to update a Cross-site scripting exception:

1. Navigate to **NetScaler Managed Apps > App Delivery > New application > Security**.
2. Click the pencil icon under the **Actions** column corresponding to the security protection that you want to edit.
3. Navigate to the **Security Protections** tab and click **Cross-site Scripting**.
4. In the **Cross-site Scripting** page, click the pencil icon in the **Actions** column corresponding to the required exception.
5. In the **Configure Cross-site Scripting Exception** page, edit details and click **Update Exception**.
6. You can view the updated exception in the **Cross-site Scripting** page.

Command injection

The command injection security check examines if an incoming request has unauthorized commands that can break the system security or modify the system. If the request has malicious commands, the NetScaler Console service blocks the request.

Enable command injection

Before configuring the command injection security check, enable the feature for HTML and JSON content type requests. When disabled, the NetScaler Console service does not inspect the

traffic for command injection attacks. Also, you must toggle the Block action to block requests that contain unauthorized command attacks. If the block action is not selected, the NetScaler Console service only logs the data.

Intent based auto blocking

The **Block clients with 20 violations within 30 minutes** option allows you to define the intent of auto blocking malicious clients that are attempting command injection attacks.

If the number of requests violating the security check received from a client increases to 20 or more in a 30-minute duration, the client is considered malicious. Select **Block clients with 20 violations within 30 minutes** to block such clients for a 30-minute duration.

Create a command injection exception

You can add an exception for legitimate requests or resolve false positives, and bypass command injection check. Complete the following steps to add an exception.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **Command Injection**.
4. Select the required content types.
5. Click **New Exception** and set the following parameters to add a command injection exception:
 - a. Exception type. Select a content type to add an exception for command injection check.
 - b. URL Pattern. The URL for which the exception is required. Only for JSON content type, the URL is applicable. It can be a regular expression. For any URL enter `'.*'`.
 - c. Field Type. Vulnerable field that can be excluded from security check input validation.
 - i. Form Field. If the vulnerable field type is selected as form field, then enter the field name.
 - ii. Header. If the vulnerable field type is selected as header, then enter the header name.
 - iii. Cookie. If the vulnerable field type is selected as cookie, then enter the cookie name.
 - d. Status. Toggle command injection exception status.
6. Click **Add Exception**.

You can view the exception in the command injection summary section.

Edit a command injection exception

Complete the following steps to update a command injection exception:

1. Navigate to **NetScaler Managed Apps > App Delivery**.
2. Click the pencil icon under the **Actions** column corresponding to the security protection that you want to edit.
3. Navigate to the **Security Protections** tab and click **Command Injection**.
4. In the **Command Injection** page, click the pencil icon in the **Actions** column corresponding to the required exception.
5. Edit details, and click **Update Exception**.
6. You can view the updated exception on the **Command Injection** page.

Cross Site Request Forgery (CSRF)

The Cross Site Request Forgery (CSRF) security check tags each web form sent by a protected website to users with a unique and unpredictable Form ID. When there is an incoming request from the user, the security filter examines the web form to ensure that the supplied Form ID is correct. The security check is applicable only to HTML requests that contain a web form, with or without data.

Enable CSRF

Before configuring the CSRF security check, you must first enable the feature. When disabled, NetScaler Console service does not inspect the traffic for CSRF attacks. Also, you must toggle the Block action to block requests that contain malicious CSRF attacks. If the block action is not selected, NetScaler Console service only logs the data.

Create a CSRF exception

You can create a CSRF exception to prevent blocking legitimate requests or resolve false positives and bypass the security check. Complete the following steps to add an exception.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **CSRF**.
4. Select the required content types.
5. Click **New Exception** and set the following parameters to add an exception for CSRF:

- a. URL Pattern. The URL for which the exception is required. It can be a regular expression. For any URL enter '. *'.
- b. Status. Toggle command injection exception status.

6. Click **Add Exception**.

You can view the exception in the CSRF summary section.

Edit a CSRF exception

Complete the following steps to update a CSRF exception:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click the pencil icon under the **Actions** column corresponding to the security protection that you want to edit.
3. Navigate to the **Security Protections** tab and click **CSRF**.
4. In the **CSRF** page, click the pencil icon in the **Actions** column corresponding to the required exception.
5. Edit details and click **Update Exception**.
6. You can view the updated exception on the CSRF section.

Field format protection

With the field format protection feature of the NetScaler Console service, you can examine both the length and type of web form data to ensure that it is appropriate for the field. If inappropriate web form data is found in a user request, you can configure the NetScaler Console service to block the request.

The NetScaler Console service supports examining the different field formats such as numeric, alphabetic, and alphanumeric. For example, if a particular field expects the user to enter a phone number, the field format protection feature examines the user-submitted input to ensure that the data matches the format of a phone number. If a field expects the first name, this feature ensures that the data in that field is of a type and length appropriate for a first name. It does the same thing for each form field that you configure it to protect.

Enable field format protection

Before configuring the field format protection feature, you must first enable the feature. When disabled, the NetScaler Console service does not inspect the field formats of web form data. You

must enable the Block Requests toggle to block the requests that violate the field formats. If the block action is disabled, the NetScaler Console service only logs the malicious data.

In addition to enabling the feature, you can add one or more enforcement rules to configure field format protection for an individual field of a specific form.

Add a field format enforcement

Complete the following steps to add field format protection for a specific field:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **Field Format**.
4. Select **Enable Field Format Protection**.
5. Click **New Enforcement** and set the following parameters:
 - a. URL Pattern: The client URL for which field format protection is applied.
 - b. Field name: The name of the field that is validated for the data entered.
 - c. Field Format: The type of data that is allowed for the field. The following types of data are supported:
 - i. Integer
 - ii. Alphabets
 - iii. Alphanumeric
 - iv. No HTML
 - v. Any
 - vi. SSN
 - vii. Credit card
 - viii. Custom Format
 - d. Minimum Length: The minimum length of data allowed in the field.
 - e. Maximum Length: The maximum length of data allowed in the field.
 - f. Status: Select the check box to enable the enforcement.
6. Click Add Enforcement.

Edit a field format enforcement

Complete the following steps to modify a field format enforcement:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click the pencil icon under the **Actions** column corresponding to the security protection that you want to edit.
3. Navigate to the **Security Protections** tab and click **Field Format**.
4. In the **Field Format** page, click the pencil icon in the **Actions** column corresponding to the required enforcement.
5. Edit the values (or enforcement) and click **Update Enforcement**.

Field consistency

The field consistency security check examines web forms returned by a user to a website and verifies if the web forms were modified inappropriately by the client. The security check applies only to HTML requests that contain web forms, with or without data.

Enable field consistency

Before configuring field consistency security check, you must first enable the feature. When disabled, NetScaler Console service does not inspect the traffic for form field attacks. Also, you must toggle the Block action to block requests that contain malicious attacks. If the block action is not selected, NetScaler Console service only logs the data.

Create field consistency exception

You can create a field consistency exception to prevent blocking legitimate requests or resolve false positives and bypass the security check. Complete the following steps to add an exception.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **Field Consistency**.
4. Select the required content types.
5. Click **New Exception** and set the following parameters to add an exception for CSRF:
 - a. URL Pattern. The URL for which the exception is required. It can be a regular expression. For any URL enter `' . *'`.
 - b. Status. Toggle command injection exception status.

6. Click **Add Exception**.

You can view the exception in the **Field Consistency** section.

Edit field consistency exception

Complete the following steps to update a Cross-site scripting exception:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click the pencil icon under the **Actions** column corresponding to the security protection that you want to edit.
3. Navigate to the **Security Protections** tab and click **Field Consistency**.
4. In the **Field Consistency** page, click the pencil icon in the **Actions** column corresponding to the required exception.
5. In the **Configure Field Consistency Exception** page, edit details, and click **Update Exception**.

WAF Signatures

A signature represents a pattern that is a component of a known attack on an operating system, web server, website, XML-based web service, or any other resource. A set of configurable rules offers an easy-to-use security service, applying the power of pattern matching to detect attacks and protect your application against common vulnerabilities and exposures.

Enable WAF signature protection

Before you customize a signature rule, you must enable the security check. When disabled, the NetScaler Console service does not inspect the traffic for common vulnerability attacks. Also, enable the Block toggle to block requests with a common vulnerability attack. When the block toggle is not selected, the NetScaler Console service only logs data for the selected signature rule.

Manage WAF signatures

The **Signature** section displays a preconfigured list of signatures with new rules added periodically or updates added to old rules. As a user, you might want to customize a signature or list of signatures under a category. To filter signatures, select a category and use the search functionality to narrow-down your search.

Categories. Signatures are classified under various categories. You can select a category to view the list of signatures classified under it. For example, selecting `web-cgi` reduces the table to display signatures that reference `web-cgi` attack type.

Search. The search functionality enables you to locate a signature based on the category that you have selected. For example, selecting `web-cgi` as the signature category, you can use the signature attributes to filter signatures that reference this category. Following are the signature attributes and its search operators to filter your search:

- **ID.** Unique identifier for a signature. Use the search operator to sort rules based on signature ID.
- **Log string.** Log message for a signature. Use the search operator to sort rules based on log string value.
- **Year.** The year when the rule was newly added. Use a search operator to sort rules based on a year value.
- **Reference.** External references to a signature. Use a search operator to sort rules based on reference.
- **Block.** Block toggle to block traffic. Use a search operator to sort rules based on block toggle status.
- **Log.** Log toggle to only log data. Use a search operator to sort rules based on log toggle status.

Customize a WAF signature

Complete the following steps to customize a signature:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **Signatures**.
4. Select **Enable Signatures**.
5. In the **Signature** section, select signatures and set the following parameters:
 - a. **Log.** Enable log toggle to only log signature violation.
 - b. **Block.** Enable block toggle to block traffic.

HTTP security headers

HTTP response headers are used to prevent security attacks on your web application by controlling the browser behavior during application access. In addition to the default headers, the security check enables you to add a security header or delete the Server: Apache/2.4.1 (UNIX) header in the HTTP responder headers for enhanced security protection.

Enable HTTP security header protection

Before you configure the HTTP headers, you must enable the security check. When disabled, the NetScaler Console service does not inspect the traffic for HTTP security headers.

Configure default HTTP security header

Complete the following steps to insert a default HTTP header:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **HTTP Security Headers**.
4. Select **Enable HTTP Security Headers**.
5. In the **HTTP Security Headers** section, set the following recommended default headers:
 - a. **X-Frame-Options**. Specifies the browser how to behave when rendering a page in `<frame>`, `<iframe>`, `<embed>` or, `<object>` format. The security header protects the website from clickjacking attacks. There are two directives for **X-Frame** options – **DENY** and **SAMEORIGIN**. If you specify **DENY** attempts to load frames from both the same sites and other sites will FAIL. If you specify **SAMEORIGIN**, page loading works from the same website and fails for other websites.
 - i. **Value**. Select directive type as **DENY** and **SAMEORIGIN**.
 - ii. **Status**. Select the toggle to insert the header when examining the traffic.
 - iii. **Actions**. Click the pencil icon to edit a directive.
 - b. **Content-Security-Policy**. Enables administrators to mitigate Cross site script and clickjacking attacks by controlling how the browser loads JavaScript's images, CSS from different sources. For example, **Content-Security-policy**: `script-src 'self' js.example.com`. Indicates how JavaScript can be loaded from js.example.com and not from anything else.
 - i. **Value**. Set the source directive value.
 - ii. **Status**. Select the toggle to insert the header when examining the traffic
 - iii. **Actions**. Click the pencil icon to edit.
 - c. **X-Content-Type-Options**. Specifies the browser that only the MIME types advertised by the original web server in the Content-Type headers must be used. This header protects the website against MIME sniffing vulnerabilities.

- d. Referrer-Policy. Specifies the browser how much referrer information must be included with requests. There are two directives for referrer-policy security header – no-referrer and no-referrer-when-downgrade (default).
 - i. Referrer-policy security header – no-referrer. If you specify the no-referrer directive, the Referrer header is omitted, and no referrer information is sent along with the requests.
 - ii. No-referrer-when-downgrade (default). If you specify the no-referrer-when-downgrade (default) directive, the origin, path, and query string of the URL are sent as a referrer when the protocol security level stays the same (HTTP → TTP, HTTPS → HTTPS) or improves (HTTP → HTTPS), but not sent to less secure destinations (HTTPS → HTTP). For "origin" derivative, only the origin of the document is sent as referrer.
 - i. Value. Directives type for referrer-policy security header.
 - ii. Status. Select the toggle to insert the header when examining the traffic
 - iii. Actions. Click the pencil icon to select a directive.

Add an HTTP security header

The NetScaler Console service security protection enables you to add an extra HTTP security header for enhanced security protection. Complete the following steps to add a new HTTP header:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **HTTP Security Headers**.
4. In the **HTTP Security Headers** section, click **Insert Header** and set the following parameters:
 - a. Header Name. Name of the HTTP security header.
 - b. Value. Directive for the security header.
 - c. Status. Select the toggle to insert the header when examining the traffic.
5. Click **Add Header**.

You can view the header in the **Inserted Headers** section.

Delete an HTTP server header

The NetScaler Console service security protection enables you to delete a server header from the HTTP server-side response. Complete the following steps to delete a header:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection > Create > HTTP Security Headers**.
2. In the **HTTP Security Headers** section, click **Delete Headers**.
3. In the **Delete Headers** page, set the following parameters.
 - a. **Header Name**. Name of the server header to be removed in the HTTP response.
4. Click **Delete Header**.

You can view the header in the **Delete Headers** section.

Data leak prevention

Your websites might have access to database servers that store user-specific sensitive information such as credit card numbers and Social Security Numbers. The leakage of such information might pose a security risk. The Data Leak Prevention feature of the NetScaler Console service helps to avoid the leak of such sensitive information.

With this feature, the NetScaler Console service identifies if the responses coming from the webserver contain any user-specific sensitive information. When a match is found, the NetScaler Console service takes configured action to avoid leakage of such information.

The data leak prevention feature allows you to create safe objects that define the rules to prevent attacks. These safe objects contain regular expressions to match the sensitive information in the responses and the action to perform when there is a match.

If the responses from the webserver match the regular expression defined in the safe object, you can configure the NetScaler Console service to perform one of the following actions:

- **Block**: Block the response.
- **Mask**: Mask the sensitive information with an X in the responses before processing it further.
- **None**: Take no action. The NetScaler Console service processes the responses as is without any changes.

The safe objects for credit card and Social Security Number (SSN) are created by default. You can enable or disable these default safe objects and change the action associated with them. You cannot change the name or regular expression of these default safe objects.

You can configure actions at the global level. Configuring the global level action changes the action associated with all the individual safe objects. For example, if you configure the global action as

block, the action associated with all the safe objects gets changed to block. You can also change the action specific to individual safe objects.

Enable data leak prevention

Before you configure the data leak prevention, you must first **Enable Data Leak Prevention**. By default, the credit card and SSN safe objects are created, disabled, and the action is set to block. If necessary, you can edit the status or safe object action for the default safe objects. You can also create new safe objects by clicking **Add Safe Object Rule**.

NOTE:

If the data leak prevention feature is disabled, the default safe objects for credit card and Social Security Number (SSN) are not effective even if their status is enabled.

Add a safe object

Complete the following steps to create a safe object:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **Data Leak Prevention**.
4. Select **Enable Data Leak Prevention** and click **Add Safe Object Rule**.
5. On the **Configure Safe Object Rule** page, configure the following parameters:
 - Safe Object Name: Name for the safe object.
 - Safe Objection Action: The action to perform when the server response matches the configured regular expression. You can choose Block, Mask, or None.
 - Maximum Match Length: The numerical value representing the maximum length of the string that you want to match.
 - Regular expression: A Perl compatible regular expression that is used for matching the response from the server.
 - Comments: Any remarks about the safe object.
6. Select **Status** to enable the safe object rule.
7. Click **Add**.

You can view the newly added safe object on the **Data Leak Prevention** page.

Edit the safe object

Complete the following steps to edit a safe object:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click the pencil icon under the **Actions** column corresponding to the security protection that you want to edit.
3. Navigate to the **Security Protections** tab and click **Data Leak Prevention**.
4. Click the edit icon under the **Actions** column corresponding to the safe object that you want to edit.
5. Perform the required changes and click **Update**.

You can also delete the safe object by clicking the trash icon under the **Actions** column corresponding to the desired safe object.

Configure global actions for safe objects

Using the Global Actions option, you can change the safe object action for all the existing safe objects at once.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click the pencil icon under the **Actions** column corresponding to the security protection that you want to edit.
3. Navigate to the **Security Protections** tab and click **Data Leak Prevention**.
4. Select **Enable Data Leak Prevention**.
5. Choose an action from the **Global Actions** drop-down list.
6. Click **Yes, Proceed**.

Bot signatures

The bot signature security check protects your web application against bot attacks. Bot signatures help in identifying good and bad bots based on request parameters such as user-agent in the incoming request.

Enable bot signature protection

Before you customize a bot signature rule, you must enable the security check. When disabled, the NetScaler Console service does not inspect the traffic for bot attacks. Also, enable the Block toggle

to block requests with a bot attack. When the block toggle is not selected, the NetScaler Console service only logs data for the selected bot rule.

Manage bot signatures

The list of bot signatures is huge and new rules get added and stale ones are removed periodically. As a user, you might want to search for a specific bot signature or list of signatures under a category. To filter signatures easily, the bot signature page provides an enhanced search capability. The search function enables you to find bot rules and customize its property based on one or more bot signature attributes.

Categories. Bot signatures are classified under different categories. You can select a category to view the list of bot signatures classified under it. For example, selecting "Crawler" reduces the table to bot rules that reference Crawler, and it is an easy way to locate Crawler type bot attacks.

Search. The search functionality enables you to locate a bot signature based on the category that is selected. For example, selecting "Crawler" as the bot signature category, you can use the bot signature attributes and its search operators to filter a signature that reference the selected bot category. Following are the bot signature attributes and its search operators to filter your search.

- **Id.** Sort bot signatures based on bot rule ID.
- **Name.** Sort bot signatures based on rule name.
- **Developer.** Sort bot signatures based on the host company publisher.
- **Type.** Sort bot signatures based on signature type.
- **Action.** Sort bot signatures based on bot action.
- **Log.** Sort bot signatures that have logging enabled.
- **Block.** Sort bot signatures that have blocking enabled.

Customize a bot signature

Complete the following steps to customize a bot signature:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **Bot Signatures**.
4. In the **Signature** section, select signatures and set the following parameters:
 - a. **Log.** Enable or disable log to only log data.

- b. Block. Enable or disable block toggle to block traffic.
- c. Action. Action to apply on bot evaluation. Action types are as follows:
 - i. Drop. Drop user request without sending a response.
 - ii. Reset. Reset client connection by closing it.

5. After customizing the bot signature rule, click **Create**.

In the **Deliver an Application** page, bind the security profile to a service, and click **Deploy**.

Bot trap

The bot trap is an effective technique to block attacks from bots. This technique detects and blocks automated bots by inserting a random trap URL in the server response. The trap URL is an alpha-numeric URL auto-generated by the NetScaler Console service and is invisible to human users. However, if the client is an automated bot and if the trap URL is accessed, the NetScaler Console service can block the request from the client.

Enable bot trap

Before you configure the bot trap, you must first enable the feature. When disabled the NetScaler Console service does not insert the trap URL in the server response. You must toggle the **Block Requests** to block requests from an automated bot. If the block action is not selected, NetScaler Console service only logs the data.

Configure bot trap enforcement

To prevent bot attacks with specific webpages, add the application URLs for which the bot trap URL must be inserted.

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click **Create**. On the **General** tab, type a **Name**.
3. Navigate to the **Security Protections** tab and click **Bot Trap**.
4. Select **Enabled** and **Block Requests**.
5. Click **Add Enforcement** and enter the URL for which the bot trap URL is inserted in the server response. The entity can be a regular expression.
6. Select the **Status** check box to enable the enforcement.
7. Click **Add Enforcement**.

Edit bot trap enforcement

Complete the following steps to update a bot trap enforcement:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click the pencil icon under the **Actions** column corresponding to the security protection that you want to edit.
3. Navigate to the **Security Protections** tab and click **Bot Trap**.
4. In the **Bot Trap** page, click the pencil icon in the **Actions** column corresponding to the required rule.
5. Click the pencil icon under the **Actions** column corresponding to the security protection that you want to edit.
6. Select **Bot Trap** on the left navigation menu.
7. Click the pencil icon under the **Actions** column corresponding to the enforcement that you want to edit.
8. Edit the values (or enforcement) and click **Update Enforcement**.

Device fingerprint

Device fingerprint technique detects the incoming traffic as a bot by inserting a JavaScript script in the HTML response to the client. The JavaScript script when invoked by the browser, collects browser and client attributes and sends a request to the NetScaler Console service. The attributes are examined to determine whether the traffic is a Bot or a human.

Select **Enable Device Fingerprint** to enable the device fingerprint technique for incoming traffic. Select **Block Requests** to block the requests that are detected as requests coming from a bot. If the block action is disabled, the NetScaler Console service only logs the requests that are coming from a bot.

Bot TPS

Enable bot TPS (Transactions Per Second) for your application. It helps you to detect the incoming traffic as bot based on one of the following parameters:

- Number of transactions per second
- Surge in transactions (%) for the last 30 minutes

You can drop or reset the connection if the incoming requests are from a bot.

Add a policy for bot TPS

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application > Security Protection**.
2. Click the pencil icon under the **Actions** column corresponding to the security protection that you want to edit.
3. Navigate to the **Security Protections** tab and click **Bot TPS**.
4. Select **Enable Bot TPS**.
5. Click **Add Policy** and specify the following fields:
 - a. **Input Type** - Defines the source type to calculate the transactions per second.
 - b. **Maximum Transactions (per second)** - The maximum transactions you want allow within one second for the selected input type.
 - c. **Maximum TPS increase percentage** - Surge in transactions (%) for the last 30 minutes.
 - d. **Action** - Set an action for the incoming bot traffic.
6. Click **Add Policy**.

In this example policy, the incoming traffic is detected as bot if one of the following conditions is met at any time:

- Exceeds 100 transactions per second from the same client's IP address.
- 15 percent surge in transactions for the last 30 minutes.

App features

You can create and edit customized configurations specific to your network needs for the following app features:

- [Content policies](#)
- [SSL settings](#)
- [Authentication](#)
- [Service profiles](#)

After the configuration changes, you can deploy all the applications associated with the app features in one go.

1. Navigate to **App Features** and select the required tab.

2. Perform one of the following:

- a. To create a configuration, click **Create**.
- b. To edit an existing configuration, click the three dots in the **Actions** column and click **Edit**.
After changing the configuration, click **Update and Deploy**.
- c. To deploy the applications associated with an updated configuration, click the three dots in the **Actions** column and click **Deploy**.
- d. To delete a configuration, click the three dots in the **Actions** column and click **Delete**.

NOTE:

You can delete an app feature only if it is not associated with any application.

You can check the status of the app feature setting in the **Status** column.

NOTE:

If one or more applications associated with the updated configuration is not in the **Deployed** state, the **Status** column shows **Error**. Click **Error** to view applications that failed during deployment. To know why deployment failed for a specific application, navigate to the **Application** dashboard and click **Error** in the **Status** column corresponding to that application.

Deliver a classic application

A classic application in the NetScaler Console service offers granular control over the configuration as compared to a modern application and is recommended for advanced NetScaler users.

Use the classic application workflow in the NetScaler Console service if:

- You are an existing NetScaler on-premises customer with a substantial investment in existing NetScaler configurations.
- Interested to move to the Cloud but prefer to have the familiar configuration experience as the NetScaler on-premises product.

The NetScaler Console service offers two options:

- Create an application from scratch using the NetScaler configuration model. Recommended if you are familiar with NetScaler configurations and want to create similar configurations in the Cloud.
- Migrate an existing NetScaler configuration to the Cloud. Recommended if you have a stable configuration that has been working well for a long time and you want to move the configuration as is to the NetScaler Console service.

To create an application from scratch, see [Create a classic application](#).

To migrate an NetScaler configuration, see [Migrate the NetScaler configuration to the NetScaler Console service](#).

For a simplified, streamlined experience and when you don't have an NetScaler configuration, use the NetScaler managed application. For more information, see [Deliver an application](#).

Supported NetScaler® features

The following NetScaler features are supported in the classic application workflow in the NetScaler Console service:

- Audit Logging
- Content Switching
- DNS
- Load Balancing
- Monitoring
- Rate limiting
- Rewrite
- Responder
- SSL Offloading
- URL Transform
- Web application firewall (WAF)

Manage a classic application

In the **NetScaler Managed Apps > App Delivery** page, click one of the following options in the **Actions** column to manage your deployed classic apps. Redeploy the application after making changes.

- Edit
- Redeploy
- Undeploy
- Health Status

Modify and redeploy a classic application

You can edit the application details, services, endpoints, load balancing, content rules, and security protection settings for an application.

1. Navigate to the **NetScaler Managed Apps > App Delivery** page.
2. For applications whose status is **Deployed**, click the three dots in the **Actions** column.
3. Click **Edit**.
4. Update the configured values and click **Deploy**.

You have completed the steps to modify and redeploy an application.

Check the health status of an application

The health status displays the real-time health of your deployed applications for each application service in an availability zone.

1. Navigate to the **NetScaler Managed Apps > App Delivery** page.
2. For applications whose status is **Deployed**, click the three dots in the **Actions** column.
3. Click **Health Status**.

Create a classic application

The classic application workflow mirrors the NetScaler configuration model. It uses the same structure and the same names, such as load balancing virtual server, service and service group, and bindings.

NOTE:

Classic apps are disabled by default. To use classic apps, contact NetScaler® support to enable it for your account.

Sample configuration for a classic application

You are a NetScaler customer interested in using load balancing to evenly distribute network traffic and avoid overloading any back-end server. You also want to ensure that requests from a client are sent to the same back-end server.

Create the following entities:

- Load balancing virtual server. Configure persistency settings on the virtual server.
- HTTP service to bind to the virtual server.

Perform the following steps:

1. Navigate to **NetScaler Managed Apps > App Delivery > New Application**.
2. Click **Switch to Classic Application**.
3. Type a name for the application and select an environment. Click **Next**.
4. Select **Load Balancing**.
5. Add a server. Click **Add** and specify a server name and the IP address of your back-end application server. Click **Create**.

The server appears in the list of servers for the load balancing virtual server.

6. Click **Add** in the **Services** section.
7. Specify a service name and service type. Click **Server Settings**. Specify the server name that you created earlier and port as 80. Click **Create**.

The service appears in the list of services for the load balancing virtual server.

8. Add a load balancing virtual server. In the **LB VServers** section, click **Add**.
9. Select **Endpoint VServer** and specify values for the following parameters:
 - a. Name for the virtual server.
 - b. Set **Vserver Port** to **80**.
 - c. Set **Service Type** to **HTTP**.
 - d. Set **Persistence Type** to **SOURCEIP**.
10. In **Service Bindings**, click **Add**.
11. Type the name for the service that you created earlier and click **Create**.

The service appears in the list of services bound to the load balancing virtual server.

- Click **Create** and then click **Next**. Provide the endpoint details. You have successfully created a virtual server. Click **Submit**.

The application is listed on the Applications page.

- In the **Actions** column, click the three dots and select **Deploy**.

You have successfully deployed a classic application.

Migrate the NetScaler configuration to the NetScaler Console service

In the NetScaler Console service, you can either create an application or migrate an application from an existing NetScaler appliance using the **Migrate NetScaler Application** feature. With this feature, you can recreate an application configuration in NetScaler Console service that exists on a NetScaler appliance.

To migrate an application, you require a set of CLI commands as a starting point. For example, you can use the NetScaler configuration file (`ns.conf`). The NetScaler Console service analyzes the CLI commands and identifies the list of distinct applications embedded in the configuration. You can then select one application configuration that you want to migrate. The NetScaler Console service extracts the corresponding CLI commands, which you can review before the migration process starts. You can also view and change any of the following values:

- **IP Address/ports** - The IP address and port of the services and service group members from the original application configuration.
- **Configuration Files/Secrets** - Resource files or secrets specified in the application configuration. For example, password and certificate files.

You're ready to start migrating the application configuration. Migration creates an application. Deploy this application on the target environment.

NetScaler features supported in the NetScaler Console service

For a list of supported NetScaler features in the NetScaler Console service, see [Supported NetScaler features](#).

How to migrate a NetScaler configuration to the NetScaler Console service

Follow these steps to migrate a NetScaler configuration as applications to the NetScaler Console service.

- Click **NetScaler® Managed Apps > App Delivery** and select **Migrate NetScaler Configuration**.

2. Select one of the following options:

- **Import using CLI commands:** Add the CLI commands manually, or paste the commands copied from the configuration file (`ns.conf`) of a source NetScaler appliance.
- **Upload configuration file:** Click **Choose file** and navigate to the file on your computer. The commands appear in the box.

3. Select the target application environment where you want to deploy the application. Click **OK** and then click **Next**.

The service starts discovering the applications from the configuration. Click **OK**.

4. Specify the following parameters and then click **Next**.

- Application Name
- Select the virtual servers that form the application endpoint. You can filter the list of virtual servers based on **Virtual-Server-Name**, **Protocol**, **IP-Address**, and **LB-Group**. For each virtual server, specify the following parameters:
 - Name: Name for the endpoint associated with the selected virtual server.
 - Access Type: Select **Internal** or **External**.
 - FQDN Type: Select **Auto allocated** or **User defined**.
 - **Auto Redirect HTTP traffic to HTTPS** Select to ensure that the clients communicate over secure SSL. (Applicable only if the virtual server is of type SSL.)

The service starts parsing the commands and extracting the application configuration. Click **OK**.

5. Click **Next**. The service validates the CLI commands. Click **OK**.

The commands are displayed in three sections, as applicable. The first section lists the commands that are not applicable to the NetScaler Console service.

The second section lists the commands for features that are currently not supported in the NetScaler Console service.

The third section lists the commands that contain references to entities that do not exist in your configuration.

6. Click **Next**. Any issues found in the configuration are highlighted in red color. The suggested configuration, after fixing the issues, is also listed.

7. Do one of the following:

- Click **Next** to accept the fix.
 - Click **Application Configuration** to edit the configuration. You are taken to the **Application Configuration** page where you can edit the commands.
8. (Optional) Change the IP address and port values for the application servers by clicking the pencil icon. Click **Next**.
 9. (Optional) Click the upload icon to upload any SSL certificates required by the application and specify their passwords. You are taken to the SSL certificate store. Select a certificate from the list and click **Select**. Then, click **Next**.
 10. Review the configuration summary and click **Migrate**. After successful migration, an application is created with the specified configuration. The following page appears. Click **Close**.

You are taken to the applications landing page.

11. Navigate to your application using the pagination at the bottom of the page. Click the three dots in the **Actions** column and select **Deploy**.

The application summary page appears.

After successful deployment, the application status is updated on the **Applications** page.

If you face any issues, contact NetScaler support at [<https://www.citrix.com/support/>](https://www.citrix.com/support/).

How-to articles

The NetScaler Console service provides an outcome-focused model that allows you to specify the generic, functional, and business intent of your application deployment.

The NetScaler Console service “How-to articles” are simple, relevant, and easy to implement articles on the features available with the service. The following articles are a step-by-step guide for using NetScaler Console service features to secure your application and to keep your data confidential.

[Get an A+ security rating for your application in the Qualys SSL Lab test](#)

[Rewrite HTTP headers for rewrite content policy](#)

Get an A+ security rating for your application in the Qualys SSL lab test

Using insecure ciphers and protocols for transactions can impact the privacy, data integrity, and security of the users accessing your applications. Your application might be compromised if the correct settings are not in place.

How does the NetScaler Console service help in getting an A+ security rating by Qualys SSL Labs?

SSL keeps your connection secure and prevents anyone from reading or modifying information while it is transferred between the two systems. Qualys SSL Labs rates the applications based on the ciphers, protocols, and other SSL settings. If your application servers meet the required settings, they are given an A+ rating. For more information, see [SSL Server Test](#).

As an application owner, you can choose to secure your application by selecting **A+ Security**. When you select this option, the service selects the required ciphers, certificates, key exchange algorithms, and protocols required to get an A+ rating on your back-end application servers. The service adds this setting using SSL policies.

Create an SSL policy with A+ security intent using the GUI

Start the **Create Application** workflow. After specifying the application details and creating services, add endpoints.

For more information about specifying application details, see [Create an application](#). For more information about creating services, see [Create services](#).

For more information about creating an endpoint, see [Add endpoint](#).

1. Navigate to **Endpoints**.
2. Click **Add Endpoint**.
3. Modify an existing endpoint or click **Create Endpoint**.
4. Click **Add SSL Policy**.
5. Type a name for the policy and select **A+ Security**.
6. Click **Create**.

An SSL policy is added with the A+ security intent to secure your application.

Create an SSL policy with A+ security intent using the API

Send the API to the following endpoint.

METHOD	URL
POST	/adcaas/nitro/v1/config/network_functions

HEADER	VALUE
Content-Type	application/json
Cookie	SESSID={{Session ID}}
ADS_SERVICE_TYPE	INTENT

The following snippet is an example of the API payload.

```
{
  "network_functions": [
    {
      "name": "DemoSSLPolicy",
      "type": "ssl_frontend",
      "ssl_frontend_params": {
        "aplussecurity": true,
        "ciphers": [],
        "tls1": false,
        "tls11": false,
        "tls12": false,
        "tls13": false
      }
    }
  ]
}
```

```
]

}
```

Example response

```
{
  "network_functions": [
    {
      "created_at": "2021-12-10T10:35:50Z",
      "id": "680aaf2b8537abbc9b2ac0f11aa98382",
      "name": "DemoSSLPolicy",
      "ssl_frontend_params": {
        "aplussecurity": true,
        "ciphers": [],
        "tls1": false,
        "tls11": false,
        "tls12": false,
        "tls13": false
      },
      "type": "ssl_frontend",
      "updated_at": "2021-12-10T10:35:50Z"
    }
  ]
}
```

```
}
```

Use this policy while creating an application as required.

Create an application with SSL policy for A+ security intent using the API

The following API creates an application with the specified:

- Environment
- Services
- Endpoints including a valid certificate
- Load balancing configuration

Send the API to the following endpoint.

METHOD	URL
POST	/adcaas/nitro/v1/config/applications

HEADER	VALUE
Content-Type	application/json
Cookie	SESSID={{Session ID}}
ADS_SERVICE_TYPE	INTENT

The following snippet is an example of the API payload.

```
{
  "applications":
  {
    "edition": "premium",
    "name": "DemoAppHTTPS",
    "type": "aws",
```

```
"environment": {  
  
    "id": "3b6ed423c509404915c57a06ee675e6b"  
  
},  
  
"endpoints": [  
  
    {  
  
        "name": "DemoHTTPSEndpoint",  
  
        "endpoint_type": "External",  
  
        "fqdns": [],  
  
        "fqdn_auto_allocated": true,  
  
        "listeners": [  
  
            {  
  
                "port": 443,  
  
                "protocol": "HTTPS",  
  
                "certificates": [  
  
                    "3bf39d58-7fcc-49b8-b050-a230681c97ac"  
  
                ],  
  
                "ssl_network_function": {  
  
                    "name": "DemoSSLPolicy"  
  
                },  
  
                "is_https_redirect": true,  
  
                "redirect_port": 80  
  
            }  
  
        ]  
  
    }  
  
]
```

```
    ]  
  
  }  
  
],  
  
"network_functions": [  
  
  {  
  
    "name": "DemoLB",  
  
    "type": "lb",  
  
    "lb_params": {  
  
      "algorithm": "ROUND_ROBIN",  
  
      "stickinessType": "SOURCE_IP"  
  
    }  
  
  },  
  
  {  
  
    "name": "DemoSSLPolicy",  
  
    "type": "ssl_frontend",  
  
    "ssl_frontend_params": {  
  
      "aplussecurity": true,  
  
      "ciphers": [],  
  
      "tls1": false,  
  
      "tls11": false,  
  
      "tls12": false,  
  
      "tls13": false
```

```
    }

  }

],

"application_services": [

  {

    "name": "DemoAppSvc",

    "protocol": "HTTPS",

    "port": 443,

    "app_server_type": "servers",

    "aws_autoscalegroup": "",

    "aws_instances": [],

    "servers": [

      {

        "type": "IPAddress",

        "value": "2.2.2.2"

      }

    ],

    "network_functions": [

      {

        "name": "DemoLB",

        "type": "lb"

      }

    ]

  }

]
```

```
        ]
      }
    ],
    "application_endpoints": [
      {
        "endpoint_name": "DemoHTTPSEndpoint",
        "default_route": "DemoAppSvc",
        "routes": []
      }
    ]
  }
]
```

Example response

```
{
  "applications": [
    {
      "application_endpoints": [
        {
          "default_route": "DemoAppSvc",
          "endpoint_id": "fac30d4b980deccb291f97cdfd7d63f2",
```

```
        "endpoint_name": "DemoHTTPEndpoint",

        "id": "968d6af0fc0d50a5b30cf99fe19ddf20",

        "routes": []

    }

],

"application_services": [

    {

        "app_server_type": "servers",

        "id": "0ad8677bf3df9bce95b20bd48818ed",

        "name": "DemoAppSvc",

        "network_functions": [

            {

                "name": "DemoLB",

                "type": "lb"

            }

        ],

        "port": 443,

        "protocol": "HTTPS",

        "servers": [

            {

                "type": "IPAddress",

                "value": "192.0.2.2"
```

```
        }

    ],

    "slow_server_settings": {

        "action": "detect"

    }

}

],

"created_at": "2021-12-10T11:09:42Z",

"deployed_at": null,

"edition": "premium",

"environment": {

    "cloud": "aws",

    "id": "3b6ed423c509404915c57a06ee675e6b",

    "name": "DemoEnv"

},

"id": "501d0fa420bbcd6d079cbbda7faf6e12",

"job_id": null,

"name": "DemoAppHTTPS",

"status": "INDRAFT",

"type": "aws",

"updated_at": "2021-12-10T11:09:42Z"

}
```

]

}

Rewrite HTTP headers for rewrite content policy

The content policies feature enables rewrite of information in requests or responses from the client or the server. Rewriting provides access to the requested content without showing unnecessary details about the website's actual configuration. Following are a few scenarios in which the rewrite is useful:

- Improve Security. You can rewrite all the `http://` to `https://` in the response body.
- Retain visitor on the website. You can display a custom 404 error page instead of the default 404 error page. You can display the website home page or a site map instead of the default 404 error page. This helps retain visitors on the website even when the 404 error exists.
- Redirect visitors. You can use rewrite to redirect the website instance to a different location while using the old URL.
- Publish SEO and user-friendly URLs. When a page on a site has a complicated or a lengthy URL, you can rewrite it to a simple, easy-to-remember URL, also referred to as the 'cool URL'.
- Set default URLs. You can append the default page name to the URL of a website. For example, if the default page of a company's website is `http://www.example.com/index.php`, and the user enters `example.com` in the browser address bar, you can rewrite the URL to always point to `example.com/index.php`.

When you enable the rewrite feature, the NetScaler Console service can modify the headers and body of the HTTP requests and responses.

In the HTTP traffic, you can take the following actions:

- Modify the URL of a request
- Add, modify, or delete headers
- Add, replace, or delete any specific string within the body or headers.

Here is a use case that explains configuring the content policies in the NetScaler Console service by defining the rewrite rule and action to remove the HTTP header and client IP from the request to keep the customer data confidential.

Problem Statement

A mid-sized manufacturing company uses its website to handle a large portion of its sales, deliveries, and customer support. The manufacturing company has two domains: `example.com`, for its website and email to customers, and `example.net`, for its intranet. Customers use the website to place orders, request quotes, research products, and customer service. The company can take specific action on HTTP requests based on the criteria such as user location, header, IP address, and while doing so, the website must have no performance degradation and keep customer data confidential.

How does NetScaler Console service content policies feature solve the problem?

With the NetScaler Console service content policies feature, you can deliver an application by configuring a content policy to do specific tasks. The service balances the website load and manages traffic to-and-from the company's web servers. Content policies enable you to respond to HTTP requests depending on the requirement. It examines the request from the client or response from the server and forwards the request depending on the applicable rule. Also, it evaluates the data and applies one or more actions based on the set rule.

Example

The following examples are seen in the server room of the Example Inc., a mid-sized manufacturing company that uses its website to handle a substantial portion of its sales, deliveries, and customer support. The company therefore has several web servers.

The company's system administrators use the rewrite features to do the tasks.

Scenario 1: Remove an HTTP Header in request

Example Inc. wants to remove the old `x-forwarded-for` and `client-IP` HTTP headers from incoming requests, so that only the `x-forwarded-for` headers are the ones added by the local server. This configuration can be done using the content policies.

The example below explains how to set up configuration in the NetScaler Console service. The procedures are shortened assuming that you are familiar with the basics of creating rewrite rules.

For more detailed information about creating rewrite rules, see [Add content policies](#).

In the **Create Rule** dialog box, create two rules one for x-forwarded-for and another for client-IP

- Create the rewrite rule for x-forwarded-for

Rule Name: act_del_xfor

- Check if the following condition exists:

HTTP HEADER: HTTP Request Header

Operator: Exists

Value: x-forwarded-for

- Create a rewrite action with the following description:

Type: Remove a HTTP Header in Request

Value: x-forwarded-for

- Create the rewrite rule for client-IP

Rule Name: act_del_client

- Check if the following condition exists:

HTTP HEADER: HTTP Request Header

Operator: Exists

Value: client-IP

- Create a rewrite action with the following description:

Type: Remove a HTTP Header in Request

Value: client-IP

All old x-forwarded-for and client-IP HTTP headers are now deleted from incoming requests.

Assumptions

The following are the assumptions made in this example:

- You have created an environment and a cloud access profile.
- You have specified the basic details, such as the name of the application, environment, services, and endpoints by navigating to **Applications > New Application**.

Follow these steps to configure the content policy:

- Step 1: Navigate to **Application > Content Policies**.
- Step 2: **Create** or **Select**. Here, you can create or add an existing Rewrite rule or a Responder rule.

- Step 3: Configure a content policy for your requirements. Here you can add the condition and operations suitable for your requirements and then bind the content policy to the services.
- Step 4: Deploy the application.

Remove an HTTP header in the request using the GUI

- **Step 1:** Navigate to **Applications > Content Policies**
 - a. In the **Deliver an Application** page, click **Create**
 - b. In the **Create Content Policy** page, click **Create**
 - c. In the **Create Content Policy** page, type a unique name for the content policy in **Name**.
- **Step 2:** Configure content policy, rules and operations. Create two rewrite rules one for `x-forwarded-for` and another for `client-IP`

In the **Add Rule** page, enter the following parameters, then click **Add**. **Rule name:** Example

- a. **If the following condition is met:**

Select **HTTP Request Header** in the drop-down list

Select **Exists** operator

Select **Value** and type `xforwarded-for`

- b. **Then do the following:**

Select **Remove the HTTP Header in Request**

In the **Header Name** type `xforwarded-for`

Rule name: Example1

- c. **If the following condition is met:**

Select **HTTP Request Header** in the drop-down list

Select **Exists** operator

Select **Value** and type `client-IP`

- d. **Then do the following:**

Select **Remove the HTTP Header in Request**

In the **Header Name** type `client-IP`

- e. In the **Create Content Policy** page, click **Create**.
- f. In the **Select Content Policies** page, click **Add**, to add the content policy.
- g. In the **Deliver and Application** page, Select the preferred service in the **Services** drop-down list by selecting the checkbox.

All old `x-forwarded-for` and `client-IP` HTTP headers are now deleted from incoming requests.

Remove an HTTP Header in Request using the API

PREREQUISITES

- A managed application is created under an environment using an auto generated FQDN.
- For the purpose of this document, it is not relevant whether the application has been created through the UI or the API.
- You have the environment details.

Set the following values in your API request:

```
Header :  
  
Content-Type:application/json  
  
INTERNAL_CLOUD_NITRO:true  
  
isCloud:true  
  
Accept:application/json  
  
Cookie:SESSID={{Session ID}};
```

Overview

The following API configuration guidelines apply to configuring content policies, primary data center managed by the NetScaler Console service. To create a content policy in this deployment mode, you need the environment id.

While the remainder of this document assumes that the managed application pre-exists, it provides instructions on retrieving the environment details using NetScaler Console service APIs.

Retrieve environment details:

Request URL:

```
GET https://{{serviceprofile}}/adcaas/nitro/v1/config/environments?
owned_by=self
```

The response is an array with details of all the deployed environments.

Note: The following response payload is adjusted for readability and only includes a single environment when the managed application is deployed.

```
{

  "environments" : [

    {

      "cloud" : "aws",

      "id" : "f3826bb4fafe7a31367a335171dc45b4",

      "aws_params" : {

        <snip>

        "region" : "us-east-2",
```

```
    "availability_zones" : [  
  
        "us-east-2a"  
  
    ],  
  
    "vpc_name" : "<snip>",  
  
    "region_name" : "US East (Ohio)",  
  
    <snip>  
  
    },  
  
    <snip>  
  
    }  
  
    ]  
  
}
```

Here, the retrieved environment id is `f3826bb4fafe7a31367a335171dc45b4`. This environment id is used while binding the network function to the application service.

Add a content policy to check the **HTTP response** for `client-IP` and `x-forwarded-for`.

Add Filters:

Filters must be created before creating content policy network functions. These filters are used while creating content policy network functions.

Request URL:

```
POST https://{{serviceprofile}}/adcaas/nitro/v1/config/filters
```

The filters are configured as follows:

- Create two filters with names, `check-client-ip` and `check-x-forwarded-for`.
- The condition detects if the request header contains the data `client-ip` and `x-forwarded-for`.

Header:

Content-Type:application/json

INTERNAL_CLOUD_NITRO:true

isCloud:true

Accept:application/json

Cookie:SESSID={{Session ID}};

ADS_SERVICE_TYPE= INTENT

Request Payload to create a filter:

```
{
```

```
"filters": [{  
  
  "name": "check-client-ip",  
  
  "expression": {  
  
    "exists": {  
  
      "req_header": {  
  
        "name": "client-ip"  
  
      }  
  
    }  
  
  }  
  
}, {  
  
  "name": "check-x-forwarded-for",
```

```

"expression": {

  "exists": {

    "req_header": {

      "name": "x-forwarded-for"

    }

  }

}

}]

}

```

Example Response: If the unique ID is received in the response the filters are successfully created.

{

```
"filters": [  
  
  {  
  
    "expression": {  
  
      "exists": {  
  
        "req_header": {  
  
          "name": "client-ip"  
  
        }  
  
      }  
  
    }  
  
  },  
  
  {  
  
    "id": "ac1105409aea48f4f04d6fe8d1b2b9de",  
  
    "name": "check-client-ip"
```

```
    },  
  
    {  
  
        "expression": {  
  
            "exists": {  
  
                "req_header": {  
  
                    "name": "x-forwarded-for"  
  
                }  
  
            }  
  
        },  
  
        "id": "044885e7f8a1d0a762fe31e70ba55774",  
  
        "name": "check-x-forwarded-for"
```

```

    }

  ]

}

```

Create the network function:

Filters must be created before creating a content policy network function.

Request URL:

POST `https://{{serviceprofile}}/adcaas/nitro/v1/config/network_functions`

A content policy is configured as follows:

- `ContentPolicyNF` is the name of the content policy.
- Content policies feature is enabled.
- Rule names are set to `remove-client-IP` and `remove-x-forwarded-for`
- Rewrite action is to remove the header in the request, when the header name is either `client-IP` or `x-forwarded-for` is detected.

Request Payload to create a Content Rule Network function to remove the identified HTTP response headers `client-IP` and `x-forwarded-for`

```

{

  "network_functions": [{

    "name": "ContentPolicyNF",

```

```
"type": "content_policy",

"content_policy_params": {

    "enable_content_policy": true,

    "rule_list": [{

        "rule_name": "remove-client-ip",

        "filter_name": "check-client-ip",

        "rewrite_action": "remove_header_in_request",

        "remove_header_in_request_params": {

            "header_name": "client-ip"

        },

        "state": true
```

```
    }, {  
  
      "rule_name": "remove-x-forwarded-for",  
  
      "filter_name": "check-x-forwarded-for",  
  
      "rewrite_action": "remove_header_in_request",  
  
      "remove_header_in_request_params": {  
  
        "header_name": "x-forwarded-for"  
  
      },  
  
      "state": true  
  
    }  
  
  ]  
  
}  
  
}]
```

```
}
```

Example response: If the unique ID is received in the response, the network function of the type of content policy is successfully created.

```
{

"network_functions": [

{

"content_policy_params": {

"enable_content_policy": true,

"rule_list": [{

"rule_name": "remove-client-ip",

"filter_name": "check-client-ip",

"rewrite_action": "remove_header_in_request",
```

```
    "remove_header_in_request_params": {  
  
        "header_name": "client-ip"  
  
    },  
  
    "state": true  
  
}, {  
  
    "rule_name": "remove-x-forwarded-for",  
  
    "filter_name": "check-x-forwarded-for",  
  
    "rewrite_action": "remove_header_in_request",  
  
    "remove_header_in_request_params": {  
  
        "header_name": "x-forwarded-for"  
  
    },  
  
}
```

```
        "state": true

      }

    ],

    "created_at": "2021-11-26T08:35:38Z",

    "id": "1df2628479d0c1b5b1d8040bdd04ccb4",

    "name": "ContentPolicyNF",

    "type": "content_policy",

    "updated_at": "2021-11-26T08:35:38Z"

  }

]

}
```

Bind the network function to the application service:

The application services can be associated with the existing network functions just by specifying the name/id and type of the network function in the payload.

```
POST https://{{serviceprofile}}/adcaas/nitro/v1/config/applications
```

Note: Specify the retrieved environment id `{{id}}` in the application payload. The environment id retrieved is `f3826bb4fafe7a31367a335171dc45b4`.

Headers:

Content-Type:application/json

INTERNAL_CLOUD_NITRO:true

isCloud:true

Accept:application/json

Cookie:SESSID={{Session ID}};

The content policy network function created can be used in the application payload as follows:

```
{

  "applications": [{

    "edition": "premium",

    "name": "DemoApp",
```

```
"type": "aws",

"environment": {

"id": " f3826bb4fafe7a31367a335171dc45b4"

},

"endpoints": [],

"network_functions": [],

"filters": [],

"application_services": [

{

"name": "DemoAppSvc",

"protocol": "HTTP",
```

```
"port": 80,

"app_server_type": "servers",

"aws_autoscalegroup": "",

"aws_instances": [],

"servers": [

{

"type": "IPAddress",

"value": "2.2.2.2"

}

],

"certificates": [],
```

```
"network_functions": [  
  
  {  
  
    "name": "LB_default",  
  
    "type": "lb"  
  
  },  
  
  {  
  
    "name": "ContentPolicyNF",  
  
    "type": "content_policy"  
  
  }  
  
]  
  
}
```

```
],  
  
"application_endpoints": [  
  
  {  
  
    "endpoint_name": "DemoApp",  
  
    "default_route": "DemoAppSvc",  
  
    "routes": []  
  
  }  
  
]  
  
}  
  
]
```

Application response: The unique ID is received in the response. The application is successfully created.

```
{

  "applications": [

    {

      "application_endpoints": [

        {

          "default_route": "DemoAppSvc",

          "endpoint_id": "d2982a29642ad23f14168446523e546b",

          "endpoint_name": "DemoEndpoint",

          "id": "bc7c9038e1fced6ae176a5525f5d3860",

          "routes": []

        }

      ]

    }

  ]

}
```

```
}

],

"application_services": [

{

"app_server_type": "servers",

"certificates": [],

"id": "c34fb994ae7f9a73fbe6956fca5f511d",

"name": "DemoAppSvc",

"network_functions": [

{

"name": "LB_default",
```

```
"type": "lb"
```

```
},
```

```
{
```

```
"name": "ContentPolicyNF",
```

```
"type": "content_policy"
```

```
}
```

```
],
```

```
"port": 80,
```

```
"protocol": "HTTP",
```

```
"servers": [
```

```
{
```

```
"type": "IPAddress",

"value": "2.2.2.2"

}

],

"slow_server_settings": {

"action": "detect"

}

}

],

"created_at": "2021-11-26T08:47:59Z",

"deployed_at": null,
```

```
"edition": "premium",

"environment": {

"cloud": "aws",

"id": "3b6ed423c509404915c57a06ee675e6b",

"name": "DemoEnv"

},

"id": "2839de5e613060fb85b0ec0aa1eb378c",

"job_id": null,

"name": "DemoApp",

"status": "INDRAFT",

"type": "aws",
```

```
"updated_at": "2021-11-26T08:47:59Z"

}

]

}
```

Here, the retrieved application id is `2839de5e613060fb85b0ec0aa1eb378c`. This application id is used while deploying the application.

Deploy the application:

Request URL

```
POST https://{{serviceprofile}}/adcaas/nitro/v1/config/applications/
{{id}}/actions/deploy
```

Provide the `new{{id}}` retrieved from the application response. This API call creates a background job to deploy the application. Here the retrieved application id is `2839de5e613060fb85b0ec0aa1eb378c`.

After deploying the application, when the incoming traffic contains any confidential information, the NetScaler Console service initiates the action as defined in the rewrite rule and action to remove the specified HTTP header and the client IP.

Troubleshooting

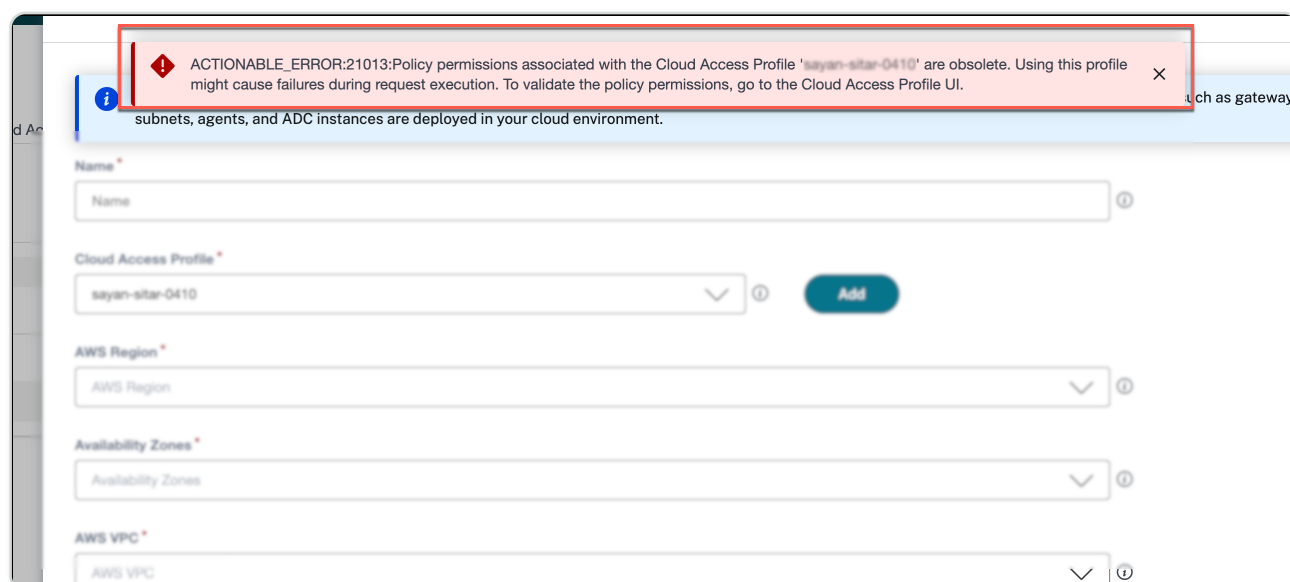
While configuring, deploying, and managing the NetScaler managed application, you might come across some actionable errors. This topic covers a few such errors and their solution.

Policy permissions associated with the cloud access profile are obsolete

Error: During AWS environment creation, the following error message is displayed: `Policy permissions associated with the Cloud Access Profile are obsolete.`

Cause: The existing cloud access profile that connects to AWS does not have all the latest role permissions.

Solution: Synchronize your cloud access profile with the latest role permissions. Navigate to **NetScaler® Managed Apps > App Environments > Cloud Access Profile**. Click the gear icon for the required cloud access profile. Follow the instructions provided on the GUI and click **Validate**.

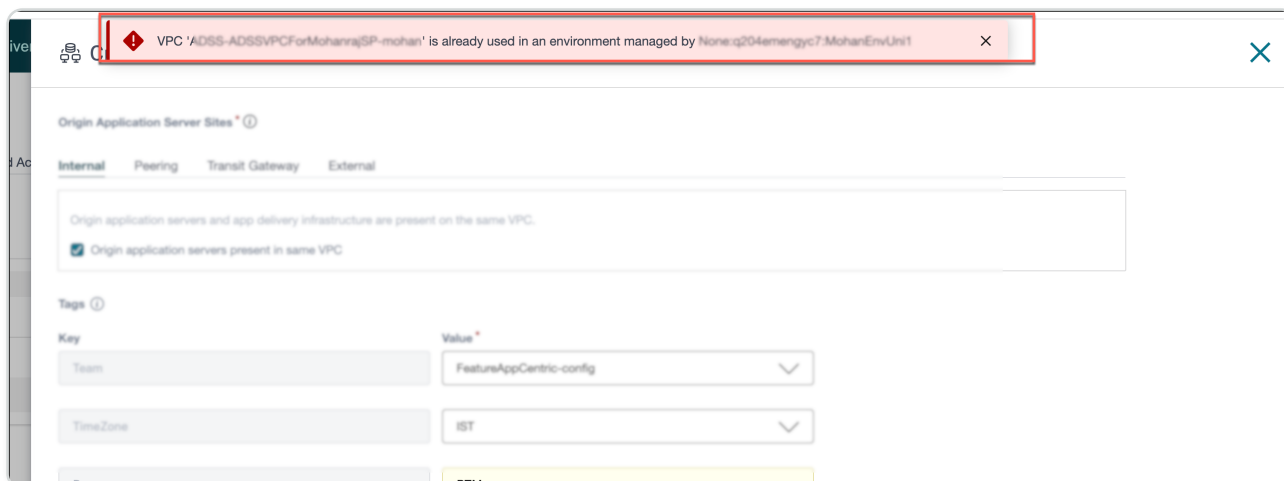


VPC is already used in another environment

Error: During Azure environment creation, the following error message is displayed: `VPC xxx already used in an environment managed by xxx.`

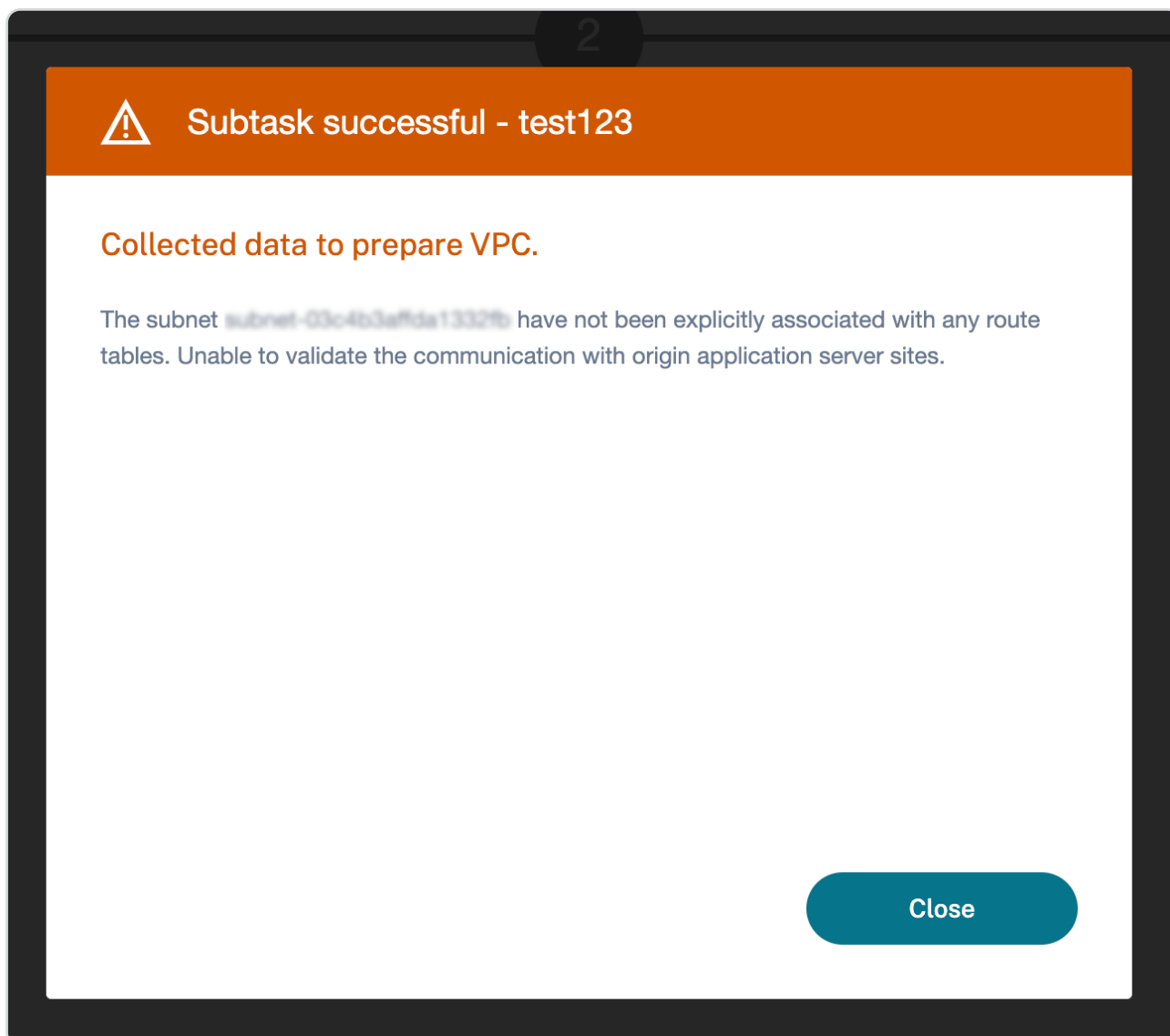
Cause: One VPC can be associated with only one environment. The VPC that is used to create an environment is already used by another environment.

Solution: Make sure that no other existing environment uses the same VPC. Undeploy the environment that is using the VPC and also delete the environment. For detailed information on how to undeploy an environment, see [Undeploy an environment](#).



Subnet is not associated with any route table

Error: During AWS environment deployment, the following error message is displayed: The subnet XXXX is not explicitly associated with any route table. Unable to validate the communication with origin application server sites.



Cause: The virtual subnet of the origin application server is not associated with any route table.

Solution: Associate the virtual subnet of the origin application server with a route table.

fix

This is the content for the sample heading topic. It is hidden from the main table of contents but remains accessible via links.