vShield API Programming Guide

vShield 5.0

vShield App 5.0

vShield Edge 5.0

vShield Endpoint 5.0

This document supports the version of each product listed and supports all subsequent versions until the document is replaced by a new edition. To check for more recent editions of this document, see http://www.vmware.com/support/pubs.

EN-000608-00

vmware®

You can find the most up-to-date technical documentation on the VMware Web site at:

http://www.vmware.com/support/

The VMware Web site also provides the latest product updates.

If you have comments about this documentation, submit your feedback to:

docfeedback@vmware.com

VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

Contents

About This Book

This manual, the *vShield API Programming Guide*, describes how to install, configure, monitor, and maintain the VMware® vShield™ system by using REST API requests. The information includes step-by-step configuration instructions and examples.

Intended Audience

This manual is intended for anyone who wants to use REST API to install or use vShield in a VMware vSphere environment. The information in this manual is written for experienced system administrators who are familiar with virtual machine technology, virtualized datacenter operations, and REST APIs. This manual also assumes familiarity with vShield.

VMware Technical Publications Glossary

VMware Technical Publications provides a glossary of terms that might be unfamiliar to you. For definitions of terms as they are used in VMware technical documentation go to http://www.vmware.com/support/pubs.

Document Feedback

VMware welcomes your suggestions for improving our documentation. If you have comments, send your feedback to docfeedback@vmware.com.

vShield Documentation

The following documents comprise the vShield documentation set:

- *vShield Administration Guide*
- *vShield Quick Start Guide*
- *vShield API Programming Guide*, this guide

Technical Support and Education Resources

The following sections describe the technical support resources available to you. To access the current version of this book and other books, go to http://www.vmware.com/support/pubs.

Online and Telephone Support

To use online support to submit technical support requests, view your product and contract information, and register your products, go to http://www.vmware.com/support.

Customers with appropriate support contracts should use telephone support for the fastest response on priority 1 issues. Go to http://www.vmware.com/support/phone_support.

Support Offerings

To find out how VMware support offerings can help meet your business needs, go to http://www.vmware.com/support/services.

VMware Professional Services

VMware Education Services courses offer extensive hands-on labs, case study examples, and course materials designed to be used as on-the-job reference tools. Courses are available onsite, in the classroom, and live online. For onsite pilot programs and implementation best practices, VMware Consulting Services provides offerings to help you assess, plan, build, and manage your virtual environment. To access information about education classes, certification programs, and consulting services, go to http://www.vmware.com/services.

Overview of VMware vShield 1

VMware vShield™ is a suite of network edge and application-aware firewalls built for VMware vCenter Server integration. vShield inspects client-server communications and inter-virtual-machine communications to provide detailed traffic analytics and application-aware firewall protection. It is a critical security component to protect virtualized datacenters from attacks and misuse, and helps achieve compliance-mandated goals. This chapter includes the following topics:

This guide assumes you have administrator access to the entire vShield system. If you are unable to access a screen or perform a particular task, consult your vShield administrator.

vShield Components

vShield includes components and services essential for protecting virtual machines in a virtualized datacenter. vShield can be configured with a Web-based user interface, a command line interface (CLI), or a REST API.

To run vShield, you need one vShield Manager virtual appliance and at least one vShield App or vShield Edge virtual appliance. The vShield Manager virtual appliance can run on a different ESX host than the vShield App and vShield Edge virtual appliances.

vShield Manager

vShield Manager is the centralized management component of vShield. You install it as a virtual appliance by deploying an OVA from the vSphere Client. Using vShield Manager's user interface or vSphere Client plug-in, you can install, configure, and maintain vShield appliances. The vShield Manager user interface leverages the vSphere Web Services SDK to display tabs within the vSphere Client inventory panel. For details about the user interface, see the *vShield Administration Guide*.

vShield App

A vShield App virtual appliance monitors all traffic into and out of an ESX host, and between virtual machines on the host. vShield App provides application-aware traffic analysis and stateful firewall protection, and it regulates traffic based on a set of rules, similar to an access control list (ACL).

As traffic passes through a vShield App, each session header is inspected to catalog the data. The vShield App creates a profile for each virtual machine detailing the operating system, applications, and ports used for network communication. Based on this information, the vShield App allows ephemeral port use by permitting dynamic protocols such as FTP or RPC to pass through, while maintaining lockdown on ports 1024 and higher. You cannot protect the ESX Service Console, ESXi direct console user interface (DCUI), or the VMkernel with vShield App because these components are not virtual machines.

NOTE vShield App and vApp are not the same thing. A vApp is a grouping of virtual machines in vSphere, for example a management appliance and a database appliance working together.

vShield Edge

A vShield Edge virtual appliance provides network edge security to protect the virtual machines in a vCloud tenant's network from attacks originating from the public network. The vShield Edge connects the isolated, private networks of cloud tenants to the public side of the service provider network through common edge services such as DHCP, VPN, NAT, and load balancing.

You install a vShield Edge from the vShield Manager. You can install one vShield Edge instance per tenant port group on a vNetwork Distributed Switch (vDS). You configure a vShield Edge by using REST API.

vShield Endpoint

vShield Endpoint offloads antivirus and anti-malware agent processing to a dedicated secure virtual appliance delivered by VMware partners. Since the secure virtual appliance (unlike a guest virtual machine) doesn't go offline, it can continuously update antivirus signatures thereby giving uninterrupted protection to the virtual machines on the host. Also, new virtual machines (or existing virtual machines that went offline) are immediately protected with the most current antivirus signatures when they come online.

vShield Data Security

vShield Data Security provides visibility into sensitive data stored within your organization's virtualized and cloud environments. Based on the violations reported by vShield Data Security, you can ensure that sensitive data is adequately protected and assess compliance with regulations around the world.

Compatibility Between Different REST API Versions

Each release of the vShield REST API represents a new version of the REST API code with new and changed features. If you are running a previous version of vShield component software, you might not be able to use all of the features of the latest release of the vShield REST API.

CAUTION The REST APIs described in this document can change over time. At this point, vShield does not guarantee forward compatibility.

REST API Version 2.0 in vShield 5.0

Release 5.0 of vShield introduces version 2.0 of the REST API. Many URLs changed from version 1.0 to 2.0.

You can determine the API version of a vShield component (such as Edge or App) with the following example REST calls. In the *GET* request syntax, `<vsm-ip>` represents the IP address or host name of vShield Manager.

Example 1-1. Determine the API version of the vShield Manager or vShield Endpoint

```
GET https://<vsm-ip>/api/versions
<versions>
    <version value="2.0">
        <module name="Dlp" baseUri="/api/2.0/dlp" version="2.0"/><module name="EndpointSolution"
                        baseUri="/api/2.0/endpointsecurity" version="2.0"/><module name="IPSet"
                        baseUri="/api/2.0/services/ipset" version="2.0"/><module name="UserMgmt"
                        baseUri="/api/2.0/services/usermgmt" version="2.0"/><module name="MACSet"
                        baseUri="/api/2.0/services/macset" version="2.0"/><module
                        name="SecurityGroup" baseUri="/api/2.0/services/securitygroup"
                        version="2.0"/><module name="Application"
                        baseUri="/api/2.0/services/application" version="2.0"/>
    </version>
</versions>
```

Example 1-2. Determine the API version of a vShield App

```
GET https://<vsm-ip>/api/versions/app/<datacenter-id>
<versions>
  <version version="2.0">
    <module version="2.0" baseUri="/api/2.0/app" id="datacenter-21" name="app"/>
  </version>
</versions>
```

Example 1-3. Determine the API version of a vShield Edge

```
GET https://<vsm-ip>/api/versions/edge/dvportgroup-63
<versions>
  <version version="2.0">
    <module version="2.0" baseUri="/api/2.0/networks" id="dvportgroup-63" name="edge"/>
  </version>
</versions>
```

The API version for vShield App is governed by the state of the datacenter in relation to a vShield component. If the datacenter state is in `backwardCompatible` mode, then it supports only version 1.0 REST calls. If the datacenter state is in regular mode, then it supports only 2.0 REST calls. These API versions are mutually exclusive – only one REST API version is supported at a time.

Table 1-1 lists compatibility between different versions of the REST API, vShield Manager, and the vShield virtual appliances: vShield App, vShield Endpoint, and vShield Edge.

Table 1-1. REST API Compatibility Matrix

REST API Version	vShield Manager Version	vShield Appliance Version	Supported?
1.0	1.0	1.0	Yes
1.0	2.0	1.0	Yes, however, client cannot configure any new features in vShield Manager 2.0
1.0	2.0	2.0 Backward Mode[1]	Yes, however, client cannot configure any new features in vShield Manager 2.0
2.0	2.0	1.0	No
2.0	2.0	2.0 Backward Mode	No
2.0	2.0	2.0	Yes

1. If the vShield Edge is in Backward Mode, the vShield Manager does not accept REST 2.0 calls for vShield Edge configuration. You must switch the vShield Edge to Normal Mode. After a vShield Edge has been switched to Normal Mode, you cannot change to Backward Mode.

Multitenancy

In vShield 5.0, the vShield App firewall configuration supports multitenancy. A single IP address can show up in multiple places in the network (different IP address namespaces) associated with different virtual machines. Only 2.0 REST APIs support multitenancy. In backward compatibility mode, vShield 5.0 supports the old APIs and does not enforce rules with awareness of multitenancy.

If you have written programs using 1.0 REST APIs, you should reconsider whether their design works as intended in the multitenancy scenario. If not, change your programs to use the API 2.0 calls.

An Introduction to REST API for vShield Users

REST, an acronym for Representational State Transfer, is a term that has been widely employed to describe an architectural style characteristic of programs that rely on the inherent properties of hypermedia to create and modify the state of an object that is accessible at a URL.

How REST Works

Once a URL of such an object is known to a client, the client can use an HTTP GET request to discover the properties of the object. These properties are typically communicated in a structured document with an HTTP Content-Type of XML or JSON, that provides a representation of the state of the object. In a RESTful workflow, documents (representations of object state) are passed back and forth (transferred) between a client and a service with the explicit assumption that neither party need know anything about an entity other than what is presented in a single request or response. The URLs at which these documents are available are often "sticky," in that they persist beyond the lifetime of the request or response that includes them. The other content of the documents is nominally valid until the expiration date noted in the HTTP Expires header.

IMPORTANT All vShield REST requests require authorization. The default vShield Manager login credentials are user `admin` password `default`. Unless you changed these, you can use the following basic authorization, where `YWRtaW46ZGVmYXVsdA==` is the Base 64 encoding of the default credentials `admin:default`.

```
Authorization: Basic YWRtaW46ZGVmYXVsdA==
```

Using the vShield REST API

You have several choices for programming the vShield REST API: using Firefox, Chrome, or `curl`. To make XML responses more legible, you can copy and paste them into `xmlcopyeditor` or `pspad`.

To use the REST API in Firefox

1 Locate the RESTClient Mozilla add-on, and add it to Firefox.

2 Click **Tools > REST Client** to start the add-on.

3 Click **Login** and enter the vShield login credentials, which then appear encoded in the Request Header.

4 Select a method such as GET, POST, or PUT, and type the URL of a REST API. You might be asked to accept or ignore the lack of SSL certificate. Click **Send**.

Response Header, Response Body, and Rendered HTML appear in the bottom window.

To use the REST API in Chrome

1 Search the Web to find the Simple REST Client, and add it to Chrome.

2 Click its globe-like icon to start it in a tab.

3 The Simple REST Client provides no certificate-checking interface, so use another Chrome tab to accept or ignore the lack of SSL certificate.

4 Type the URL of a REST API, and select a method such as GET, POST, or PUT.

5 In the Headers field, type the basic authorization line, as in the Important note above. Click **Send**.

Status, Headers, and Data appear in the Response window.

To use the REST API in curl

1 Install `curl` if not already installed.

2 In front of the REST URL, the -k option avoids certificate checking, and the -u option specifies credentials.

```
curl –k –u admin:default https://<vsm–ip>/api/2.0/services/usermgmt/user/admin
```

Ports Required for vShield REST API

The vShield Manager requires port 443/TCP for REST API requests.

About the REST API

REST APIs use HTTP requests (often sent by script or high-level language) as a way of making idempotent remote procedure calls that create, modify, or delete objects defined by the API. A REST API is defined by a collection of XML documents that represent the objects on which the API operates. The HTTP operations themselves are generic to all HTTP clients. To write a RESTful client, you should understand HTTP protocol and the semantics of standard HTML markup. For vShield REST API, you must know three things:

- The set of objects that the API supports, and what they represent. For example, what are vDC and Org?
- How the API represents these objects. For instance, what is the XML schema for the vShield Edge firewall rule set? What do the individual elements and attributes represent?
- How the client refers to an object on which it wants to operate. For example, what is a managed object ID?

To answer these questions, you look at vShield API resource schemas. These schemas define a number of XML types, many of which are extended by other types. The XML elements defined in these schemas, along with their attributes and composition rules (minimum and maximum number of elements or attributes, or the prescribed hierarchy with which elements can be nested) represent the data structures of vShield objects. A client can "read" an object by making an HTTP GET request to the object's resource URL. A client can "write" (create or modify) an object with an HTTP PUT or POST request that includes a new or changed XML body document for the object. Usually a client can delete an object with an HTTP DELETE request.

This document presents example requests and responses, and provides reference information on the XML schemas that define the request and response bodies.

RESTful Workflow Patterns

All RESTful workflows fall into a pattern that includes only two fundamental operations, which you repeat in this order for as long as necessary.

- Make an HTTP request (GET, PUT, POST, or DELETE). The target of this request is either a well-known URL (such as vShield Manager) or a link obtained from the response to a previous request. For example, a GET request to an Org URL returns links to vDC objects contained by the Org.
- Examine the response, which can be an XML document or an HTTP response code. If the response is an XML document, it may contain links or other information about the state of an object. If the response is an HTTP response code, it indicates whether the request succeeded or failed, and may be accompanied by a URL that points to a location from which additional information can be retrieved.

For More Information About REST

For a comprehensive discussion of REST from both client and server perspectives, see *RESTful Web Services* by Leonard Richardson and Sam Ruby, published 2007 by O'Reilly Media.

There are also many sources of information about REST on the Web, including:

- http://www.infoq.com/articles/rest-introduction
- http://www.infoq.com/articles/subbu-allamaraju-rest
- http://www.stucharlton.com/blog/archives/000141.html

vShield Manager Management 2

The vShield Manager requires communication with your vCenter Server and services such as DNS and NTP to provide details on your VMware Infrastructure inventory.

The chapter includes the following topics:

IMPORTANT All vShield REST requests require authorization. See "Using the vShield REST API" on page 12 for details about basic authorization.

Synchronizing vShield Manager with vCenter Server and DNS

You can use a single request to synchronize the vShield Manager with the vCenter Server and add DNS servers to the vShield Manager for IP address and hostname resolution. Synchronizing with vCenter Server enables the vShield Manager user interface to display your VMware Infrastructure inventory. For the `vcInfo` schema, and the `dnsInfo` schema, see "vShield Manager Global Configuration Schema" on page 77.

Example 2-1. Synchronize the vShield Manager with vCenter Server and Identify DNS Services

Request:

```
POST https://<vsm-ip>/api/2.0/global/config
```

Request Body:

```
<vsmGlobalConfig xmlns="vmware.vshield.edge.2.0">
  <vcInfo>
    <ipAddress>10.112.196.22</ipAddress>
    <userName>administrator</userName>
    <password>123</password>
  </vcInfo>
  <dnsInfo>
    <primaryDns>10.112.192.1</primaryDns>
    <secondaryDns>10.112.192.2</secondaryDns>
  </dnsInfo>
</vsmGlobalConfig>
```

Synchronization with vCenter Server requires its IP address (or URL) and administrator login credentials. Specifying DNS information is optional. You can synchronize vShield Manager with just vCenter Server.

Example 2-2. Synchronize the vShield Manager with vCenter Server

Request:

```
POST https://<vsm-ip>/api/2.0/global/config
```

Request Body:

```
<vsmGlobalConfig xmlns="vmware.vshield.edge.2.0">
  <vcInfo>
    <ipAddress>10.112.196.22</ipAddress>
    <userName>administrator</userName>
    <password>123</password>
  </vcInfo>
</vsmGlobalConfig>
```

Monitoring vShield Manager reachability

You can verify that the vShield Manager is reachable.

Example 2-3. Verify that the vShield Manager is reachable

Request:

```
GET https://<vsm-ip>/api/2.0/global/heartbeat
```

Retrieving Tech Support Logs

You can retrieve Technical Support logs from the vShield Manager and vShield Edge.

Get the vShield Manager Technical Support Log File Path

You can get the path to the diagnostic log file for the vShield Manager. You can then send the diagnostic log to technical support for assistance in troubleshooting an issue.

Example 2-4. Get the Tech Support Log File Path for a vShield Manager

Request:

```
GET https://<vsm-ip>/api/2.0/global/techSupportLogs
```

The technical support log is placed in a file at the following path, however the REST API has no provision for downloading it, and `wget` and `curl` do not have permission to download it, either. You can retrieve the log with vShield Manager by clicking **Settings & Reports > Configuration > Support > [Log Download] Initiate**.

```
/tech_support_logs/vsm/vshield_mgr_support_<date_time>GMT.log.gz
```

Get the vShield Edge Technical Support Log File Path

You can download the diagnostic log from a vShield Edge. You can then send the diagnostic log to technical support for assistance in troubleshooting an issue.

Example 2-5. Get the Tech Support Log File Path for a vShield Edge

Request:

```
GET https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/techSupportLogs
```

The technical support log is placed in a file, however the REST API has no provision for downloading it, and `wget` and `curl` do not have permission to download it, either. You can retrieve the log with vShield Manager by clicking **Settings & Reports > Configuration > Support > [Log Download] Initiate**.

User Management

The authentication and authorization APIs include methods to manage users and roles.

Get a List of Users

You can retrieve a list of vShield Manager users, both local users and vCenter users who are assigned a role.

Example 2-6. Get a list of users

Request:

```
GET https://<vsm-ip>/api/2.0/services/usermgmt/users/vsm
```

Before you add users to vShield Manager, the pre-existing defaults are local user `admin` and the vCenter user `administrator`.

Get Information About a User

You can retrieve information about a user.

Example 2-7. Get information about a user

Request:

```
GET https://<vsm-ip>/api/2.0/services/usermgmt/user/<userId>
```

User information includes user name, full name, email address, whether local or not, whether enabled, resource objects, roles, and scope.

Create a Local User on vShield Manager

You can create a local vShield Manager user.

Example 2-8. Create a local user

Request Header:

```
POST https://<vsm-ip>/api/2.0/services/usermgmt/user/local
```

Request Body:

```
<userInfo>
    <userId>somebody</userId>
    <password>123</password>
    <fullname>Person Somebody</fullname>
    <email>ps@y.com</email>
    <accessControlEntry>
        <role>security_admin</role>
    </accessControlEntry>
</userInfo>
```

Update a Local User Account

You can update a local user account including password. If a password is not provided, the existing password is retained. The <userId> variable in the request header should be same as the one specified in XML. The API returns updated information for the user.

Example 2-9. Update a local user account

Request Header:

```
PUT https://<vsm-ip>/api/2.0/services/usermgmt/user/local/<userId>
```

Request Body:

```
<userInfo>
    <userId>somebody</userId>
    <password>123</password>
    <fullname>Person Somebody</fullname>
    <email>ps@y.com</email>
    <accessControlEntry>
        <role>security_admin</role>
        <resource><resourceId>datacenter-312</resourceId></resource>
    </accessControlEntry>
</userInfo>
```

Enable or Disable a User Account

You can disable or enable a user account, either local user or vCenter user. When a user account is created, the account is enabled by default.

Example 2-10. Enable or disable a user account

Request:

```
PUT https://<vsm-ip>/api/2.0/services/usermgmt/user/<userId>/enablestate/<value>
```

The <value> can be 0 (zero) to disable the account, or 1 (one) to enable the account.

This API returns "204 No Content" if successful.

Remove a User Account

The first API removes a local user account, or removes the VSM role assignment for a vCenter user, without affecting the vCenter account. The second API removes a vCenter user's roles but is not allowed for local users.

Example 2-11. Remove a user account

Request:

```
DELETE https://<vsm-ip>/api/2.0/services/usermgmt/user/<userId>
```

Example 2-12. Removing a user role

Request:

```
DELETE https://<vsm-ip>/api/2.0/services/usermgmt/role/<userId>
```

Both APIs return "204 No Content" if successful.

Role Management

Get Role for a User

You can retrieve information about the role assigned to this user.

Example 2-13. Retrieve the role of a user

Request:

```
GET https://<vsm-ip>/api/2.0/services/usermgmt/role/<userId>
```

Possible roles are `super_user`, `vshield_admin`, `enterprise_admin`, `security_admin`, and `auditor`.

Add Role and Resources for a User

You can add role and accessible resources for the specified user. It affects only vCenter users, not local users. For local vShield Manager users, it throws error "400: User already present."

Example 2-14. Update the role of a user

Request Header:

```
POST https://<vsm-ip>/api/2.0/services/usermgmt/role/<userId>
```

Request Body:

```
<accessControlEntry>
    <role>new_role</role>
    <resource>
        <resourceId>resource-num</resourceId>
        ...
    </resource>
</accessControlEntry>
```

This API returns "204 No Content" if successful.

Change Role for a User

You can update the role assignment for a given user. The API returns an output representation specifying a new `<accessControlEntry>` for the user.

Example 2-15. Change the role of a user

Request Header:

```
PUT https://<vsm-ip>/api/2.0/services/usermgmt/role/<userId>
```

Request Body:

```
<accessControlEntry>
    <role>new_role</role>
    <resource>
        <resourceId>resource-num</resourceId>
        ...
    </resource>
</accessControlEntry>
```

Possible roles are `super_user`, `vshield_admin`, `enterprise_admin`, `security_admin`, and `auditor`.

Get a List of Possible Roles

You can retrieve the possible roles in vShield Manager.

Example 2-16. Retrieve possible roles

Request:

```
GET https://<vsm-ip>/api/2.0/services/usermgmt/roles
```

Get a List of Scoping Objects

You can retrieve a list of objects that can be used to define a user's access scope.

Example 2-17. Retrieve scoping objects

Request:

```
GET https://<vsm-ip>/api/2.0/services/usermgmt/scopingobjects
```

The scoping objects are usually managed object references or vCenter Server names of datacenters and folders.

Creating IPset and MACset Containers

You can create vShield containers based on IP addresses and MAC addresses. These APIs control two types of resources: vShield Manager scope object (a datacenter or portgroup) and the IPset or MACset addresses.

List IPsets Created on a Scope

You can retrieve all the IPsets that were created on the specified scope.

Example 2-18. List IPsets on a scope

Request:

```
GET https://<vsm-ip>/api/2.0/services/ipset/scope/<scope-moref>
```

The `<scope-moref>` can be a datacenter or portgroup of the vCenter to which vShield Manager is connected.

Create an IPset on a Scope

You can create a new IPset on the specified scope.

Example 2-19. Create IPset on a scope

Request:

```
POST https://<vsm-ip>/api/2.0/services/ipset/scope/<scope-moref>
```

Request Body Example:

```
<ipset>
  <objectId />
  <type>
  <typeName />
  </type>
  <description>
  New  Description
  </description>
  <name>TestIPSet2</name>
  <revision>0</revision>
  <objectTypeName />
```

```
  <value>10.112.201.8-10.112.201.14</value>
</ipset>
```

The <scope-moref> can be a datacenter or portgroup of the vCenter to which vShield Manager is connected. In the request body example, a range of IP addresses on the 10.112 net is specified (201.8 to 201.14).

Get Details of an IPset

You can retrieve details about an IPset.

Example 2-20. Get details of an IPset

Request:

```
GET https://<vsm-ip>/api/2.0/services/ipset/<ipset-id>
```

The <ipset-id> is as returned by listing the IPset on a scope.

Modify an Existing IPset

You can modify an existing IPset and retrieve details about the modified IPset.

Example 2-21. Modify an IPset

Request:

```
PUT https://<vsm-ip>/api/2.0/services/ipset/<ipset-id>
```

Request Body Example:

```
<ipset>
  <objectId />
  <type>
  <typeName />
  </type>
  <description>
  New  Description
  </description>
  <name>TestIPSet2</name>
  <revision>0</revision>
  <objectTypeName />
  <value>10.112.201.8-10.112.201.21</value>
</ipset>
```

The <ipset-id> is as returned by listing the IPset on a scope. In the request body example, the IP address range is doubled.

Delete an IPset

You can delete an IPset. The trailing boolean flag indicates forced or unforced delete. With forced delete, the object is deleted even if used in other places such as firewall rules, causing invalid referrals. For unforced delete, the object is deleted only if it is not used by other configuration; otherwise the delete fails.

Example 2-22. Delete an IPset

Request:

```
DELETE https://<vsm-ip>/api/2.0/services/ipset/<ipset-id>
```

No input representation is needed. On success, this request returns 200 HTTP OK.

List MACsets Created on a Scope

You can retrieve all the MACsets that were created on the specified scope.

Example 2-23. List MACsets on a scope

Request:

```
GET https://<vsm-ip>/api/2.0/services/macset/scope/<scope-moref>
```

The <scope-moref> can be a datacenter or portgroup of the vCenter to which vShield Manager is connected.

Create a MACset on a Scope

You can create a MACset on the specified scope. On success, the API returns a string identifier for the new MACset.

Example 2-24. Create MACset on a scope

Request:

```
POST https://<vsm-ip>/api/2.0/services/macset/scope/<scope-moref>
```

Request Body Example:

```
<macset>
  <objectId />
  <type>
  <typeName />
  </type>
  <description>Some description</description>
  <name>TestMACSet1</name>
  <revision>0</revision>
  <objectTypeName />
  <value>22:33:44:55:66:77,00:11:22:33:44:55,aa:bb:cc:dd:ee:ff</value>
</macset>
```

The <scope-moref> can be a datacenter or portgroup of the vCenter to which vShield Manager is connected. In the request body example, a comma-separated list of MAC addresses is specified.

Get Details of a MACset

You can retrieve details about a MACset.

Example 2-25. Get details of a MACset

Request:

```
GET https://<vsm-ip>/api/2.0/services/macset/<macset-id>
```

The <MACset-id> is as returned by listing the MACset on a scope.

Modify an Existing MACset

You can modify an existing MACset and retrieve details about the modified MACset.

Example 2-26. Modify details of a MACsets

Request:

```
PUT https://<vsm-ip>/api/2.0/services/MACset/<MACset-id>
```

Request Body Example:

```
<macset>
  <objectId />
  <type>
  <typeName />
  </type>
  <description>Some description</description>
  <name>TestMACSet1</name>
  <revision>0</revision>
  <objectTypeName />
  <value>22:33:44:55:66:77,00:11:22:33:44:55</value>
</macset>
```

The <MACset-id> is as returned by listing the MACset on a scope. In the request body example, one MAC address fewer is specified.

Delete a MACset

You can delete a MACset. The trailing boolean flag indicates forced or unforced delete. With forced delete, the object is deleted even if used in other places such as firewall rules, causing invalid referrals. For unforced delete, the object is deleted only if it is not used by other configuration; otherwise the delete fails.

Example 2-27. Delete a MACset

Request:

```
DELETE https://<vsm-ip>/api/2.0/services/macset/<macset-id>
```

No input representation is needed. On success, this request returns 200 HTTP OK.

Security Group Scope and Members

APIs are available for two types of resources:

- Scope – This identifies a vShield Manager scope object, which can either be a vCenter datacenter or a PortGroup (standard or distributed virtual switch). Security groups can only be created on valid scopes.
- Members – The security group object contains members.

List Security Groups Created on a Scope

You can retrieve all the security groups that have been created on a specific scope.

Example 2-28. Get existing security groups

Request:

```
GET https://<vsm-ip>/api/2.0/services/securitygroup/scope/<scope-moref>
```

The <scope-moref> could be the managed object reference of a datacenter.

Create Security Group on a Scope

You can create a new security group on the specified scope.

Example 2-29. Create new security group

Request:

```
POST https://<vsm-ip>/api/2.0/services/securitygroup/<scope-moref>
```

Example:

```
POST https://10.24.128.128/api/2.0/services/securitygroup/datacenter-31
<?xml version="1.0" encoding="UTF-8" ?>
<securitygroup>
    <objectId />
    <type>
        <typeName />
    </type>
    <description>
        Some description 2
    </description>
    <name>
        TestSecurityGroup2
    </name>
    <revision>
        0
    </revision>
    <objectTypeName />
</securitygroup>
```

Get Members for a Scope

You can retrieve a list of applicable member elements that can be added to security groups created on a particular scope. Because security group allows only specific type of container elements to be added, this list helps you determine all possible valid elements that can be added.

Example 2-30. Get members for a security group scope

Request:

```
GET https://<vsm-ip>/api/2.0/services/securitygroup/scope/<scope-moref>/members/
```

Note that this API command requires a slash (/) at the end. The request returns a long output representation of member objects.

Get Security Group Details

You can retrieve the details about a security group.

Example 2-31. Get details of a security group

Request:

```
GET https://<vsm-ip>/api/2.0/services/securitygroup/<securitygroup-id>
<VshieldConfiguration>
    <InstallStatus>
            <InstalledServices>
                <VszInstalled>true</VszInstalled><EpsecInstalled>false</EpsecInstalled>
            </InstalledServices>
    </InstallStatus>
</VshieldConfiguration>
```

Modify a Security Group

You can modify an existing security group.

Example 2-32. Modify a security group

Request:

```
PUT https://<vsm-ip>/api/2.0/services/securitygroup/<securitygroup-id>
```

Delete a Security Group

You can delete an existing security group. The `force=` flag indicates if the delete should be forced or unforced. With forced delete, the object is deleted even if used in other places such as firewall rules, causing invalid referrals. For unforced delete, the object is deleted only if it is not used by other configuration; otherwise the delete fails.

Example 2-33. Delete a security group

Request:

```
DELETE https://<vsm-ip>/api/2.0/services/securitygroup/<securitygroup-id>?force=<true|false>
```

No input representation is needed. On success, this request returns 200 HTTP OK.

Add Member to Security Group

You can add a new member to a security group.

Example 2-34. Add a member to a security group

Request:

```
PUT https://<vsm-ip>/api/2.0/services/securitygroup/<securitygroup-id>/members/<member-moref>
```

No input representation is needed. On success, this request returns 200 HTTP OK.

Delete Member from Security Group

This API removes a member from a security group.

Example 2-35. Delete member from a security group

Request:

```
DELETE https://<vsm-ip>/api/2.0/services/securitygroup/<securitygroup-id>/members/<member-moref>
```

No input representation is needed. On success, this request returns 200 HTTP OK.

Transport Set for Applications

The vShield transport set APIs are used to manipulate applications, and control two types of resources:

- Scope – identifies the scope of a vShield Manager object, which can be either a vSphere datacenter or a port group (legacy or dvPortgroup). Applications can be created only on valid scopes.
- Application – This is the main application object itself.

List Applications on a Scope

You can retrieve a list of applications that have been created on the scope specified by managed object reference `<moref>`.

Example 2-36. List applications on a given scope

Request:

```
GET https://<vsm-ip>/api/2.0/services/application/scope/<moref>
```

A non-existent scope results in a 400 Bad Request error.

Add Application to a Scope

You can create a new application on the specified scope.

Example 2-37. Add an application to a scope

Request:

```
POST https://<vsm-ip>/api/2.0/services/application/scope/<moref>
```

Request Body:

```
<application>
  <objectId/>
  <type>
    <typeName/>
  </type>
  <description>Some description</description>
  <name>TestApplication1</name>
  <revision>0</revision>
  <objectTypeName/>
  <element>
    <applicationProtocol>UDP</applicationProtocol>
    <value>9,22-31,44</value>
  </element>
</application>
```

For `applicationProtocol`, possible values are:

- TCP
- UDP
- ORACLE_TNS
- FTP
- SUN_RPC_TCP
- SUN_RPC_UDP
- MS_RPC_TCP
- MS_RPC_UDP
- NBNS_BROADCAST
- NBDG_BROADCAST

Only TCP and UDP support comma separated port numbers and dash separated port ranges. Other protocols support a single port number only.

On success, this call returns a string identifier for the newly created application, for instance `Application-1`. The location header in the reply contains the relative path of the created Application and can be used for further GET, PUT, and DELETE calls.

Get Details of an Application

You can retrieve details about the application specified by `<application-id>` as returned by the call shown in Example 2-37.

Example 2-38. Retrieve details about an application

Request:

```
GET https://<vsm-ip>/api/2.0/services/application/<application-id>
```

A non-existent application ID results in a 404 Not Found error.

Modify Application Details

You can modify the `name`, `description`, `applicationProtocol`, or `port value` of an application.

Example 2-39. Modify application

Request:

```
POST https://<vsm-ip>/api/2.0/services/application/<application-id>
```

Request Body:

```
<application>
  <objectId>Application-1</objectId>
  <type>
    <typeName>Application</typeName>
  </type>
  <description>Some description</description>
  <name>TestApplication</name>
  <revision>2</revision>
  <objectTypeName>Application</objectTypeName>
  <element>
    <applicationProtocol>TCP</applicationProtocol>
    <value>10,29-30,45</value>
  </element>
</application>
```

The call returns XML describing the modified application.

Delete Application from Scope

You can delete an application by specifying its `<application-id>`. The `force=` flag indicates if the delete should be forced or unforced. For forced deletes, the object is deleted irrespective of its use in other places such as firewall rules, which invalidates other configurations referring to the deleted object. For unforced deletes, the object is deleted only if it is not being used by any other configuration. The default is unforced (false).

Example 2-40. Delete application

Request:

```
DELETE https://<vsm-ip>/api/2.0/services/application/<application-id>?force=<true|false>
```

ESX Host Preparation for vShield App and vShield Endpoint 3

You can extend the capabilities of vShield by adding the following services: vShield App, vShield Endpoint, and vShield Edge. You must prepare each ESX host in your environment for these services. The vShield Manager OVA file contains the drivers and files necessary to install all additional services.

This chapter includes the following topics:

IMPORTANT All vShield REST requests require authorization. See "Using the vShield REST API" on page 12 for details about basic authorization.

Installing Licenses for vShield Edge, vShield App, and vShield Endpoint

You must install licenses for vShield Edge, vShield App, and vShield Endpoint before installing these components. You can install these licenses by using the vSphere Client.

1. From a vSphere Client host that is connected to a vCenter Server system, select **Home > Licensing**.
2. For the report view, select **Asset**.
3. Right-click a vShield asset and select **Change license key**.
4. Select **Assign a new license key** and click **Enter Key**.
5. Enter the license key, enter an optional label for the key, and click **OK**.
6. Click **OK**.
7. Repeat these steps for each vShield component for which you have a license.

Installing vShield App and vShield Endpoint Services on an ESX Host

To shorten the time to deployment, you can install vShield App and vShield Endpoint services on an ESX host by using a single REST call. You can do this by including `VszInstallParams` and `EpsecInstallParams` in the POST body.

CAUTION Do not install vShield App (or vShield Zones) on the ESX host where vCenter Server is running, otherwise vShield App could interfere with vSphere management traffic.

You must specify the host ID of the target ESX host to install all services.

See "ESX Host Preparation and Uninstallation Schema" on page 80.

Example 3-1. Install a vShield App and vShield Endpoint on an ESX host

Request:

```
POST https://<vsm-ip>/api/1.0/vshield/<host-id>
```

Example:

```
POST /api/1.0/vshield/host-5450 HTTP/1.1

Content-type: application/xml; charset=UTF-8
Authorization: Basic YWRtaW46ZGVmYXVsdA==
Cache-Control: no-cache
Pragma: no-cache
Host: 10.112.196.244
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
Content-Length: 489

<VshieldConfiguration><VszInstallParams><DatastoreId>datastore-5035</DatastoreId>
               <ManagementPortSwitchId>network-4485</ManagementPortSwitchId><MgmtInterface>
               <IpAddress>10.112.196.245</IpAddress><NetworkMask>255.255.252.0</NetworkMask>
               <DefaultGw>10.112.199.253</DefaultGw></MgmtInterface></VszInstallParams>
               <EpsecInstallParams>true</EpsecInstallParams><InstallAction>install
               </InstallAction></VshieldConfiguration>
```

ESX host preparation requires the following elements:

- `DatastoreId`: VC MOID of the datastore on which the vShield App service virtual machine files will be stored.
- `ManagementPortSwitchId`: VC MOID of the port group that will host the management port of the vShield App.
- `MgmtInterface`
  - `IpAddress`: IP address to be assigned to the management port of the vShield App. This IP address must be able to communicate with the vShield Manager.
  - `NetworkMask`: Subnet mask associated with the IP address assigned to the management interface of the vShield App.
  - `DefaultGw`: IP address of the default gateway.

After installation of all components is complete, do the following:

- **vShield App**: At this point, vShield App installation is complete. Each vShield App inherits global firewall rules set in the vShield Manager. The default firewall rule set allows all traffic to pass. You must configure blocking rules to explicitly block traffic. To configure App Firewall rules, see "Configuring Firewall Rules for vCenter" on page 54.
- **vShield Endpoint**: To complete installation, see "vShield Endpoint Management (old)" on page 87.

You can install a single service by identifying only that service in the POST body. In Example 3-2, only vShield App is installed, as identified by inclusion of the `VszInstallParams` element only.

Example 3-2. Install a vShield App only

Request:

```
POST https://<vsm-ip>/api/1.0/vshield/<host-id>/vsz
```

Example:

```
POST /api/1.0/vshield/host-5126 HTTP/1.1
Content-type: application/xml; charset=UTF-8
Authorization: Basic YWRtaW46ZGVmYXVsdA==
Cache-Control: no-cache
```

```
Pragma: no-cache
Host: 10.112.196.244
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
Content-Length: 368

<VshieldConfiguration><VszInstallParams><DatastoreId>datastore-5131</DatastoreId>
               <ManagementPortSwitchId>network-5134</ManagementPortSwitchId><MgmtInterface>
               <IpAddress>10.112.196.245</IpAddress><NetworkMask>255.255.252.0</NetworkMask>
               <DefaultGw>10.112.199.253</DefaultGw></MgmtInterface></VszInstallParams>
               <InstallAction>install</InstallAction></VshieldConfiguration>
```

Getting the Installation Status of vShield Services on an ESX Host

You can retrieve the installation or uninstallation status of vShield services on an ESX host to track progress as complete or not initiated. If neither of these operations is in progress, the response includes the list of installed services on the ESX host.

Example 3-3. Get vShield service installation status on an ESX host

Request:

```
GET https://<vsm-ip>/api/1.0/vshield/<host-id>
```

Example:

Uninstalling vShield Services from an ESX Host

You can uninstall vShield App and vShield Endpoint from an ESX host by using a single request.

Before uninstalling these services, you must unregister SVMs before uninstalling vShield Endpoint from the ESX host. See "Unregister an SVM from vShield Endpoint" on page 89.

CAUTION Uninstalling any of these vShield services places the ESX host in maintenance mode. After uninstallation is complete, the ESX host reboots. If any of the virtual machines that are running on the target ESX host cannot be migrated to another ESX host, these virtual machines must be powered off or migrated manually before the uninstallation can continue. If the vShield Manager is on the same ESX host, the vShield Manager must be migrated prior to uninstalling the vShield App.

Example 3-4. Uninstall vShield services from an ESX host

Request:

```
DELETE https://<vsm-ip>/api/1.0/vshield/<host-id>
```

You can uninstall a single service by specifying the service name.

Example 3-5. Uninstall a vShield App only

Request:

```
    DELETE https://<vsm-ip>/api/1.0/vshield/<host-id>/vsz
```

vShield Edge Installation 4

After ESX host preparation is complete, you can secure internal networks by installing a vShield Edge.

This chapter includes the following topics:

IMPORTANT All vShield REST requests require authorization. See "Using the vShield REST API" on page 12 for details about basic authorization.

Installing a vShield Edge

You can install one vShield Edge per port group, vDS port group, or Cisco® Nexus 1000V. A vShield Edge requires an external port group with a physical NIC and an internal port group that contains the virtual machines to be secured. The vShield Edge sits inline between these port groups. If an internal port group does not exist, you must create this port group before installing a vShield Edge.

The vShield Edge installation API copies the vShield Edge OVF from the vShield Manager to the specified datastore and deploys a vShield Edge on the given port group. After the vShield Edge is installed, the virtual machine powers on and initializes according to the given network configuration.

Installing a vShield Edge instance adds a virtual machine to the vCenter Server inventory, which is mirrored in the vShield Manager user interface. You must specify an IP address for the management interface, and you may name the vShield Edge instance.

For the schema, see "vShield Edge Schemas" on page 88.

Example 4-1. Install a vShield Edge

Request:

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>>/edge
```

Request Body:

```
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <installParams>
    <resourcePoolId>resgroup-8</resourcePoolId>
    <hostId>host-9</hostId>
    <dataStoreId>datastore-11</dataStoreId>
    <applianceConfig>
      <hostName>vShieldEdge-network-12</hostName>
      <interface>
        <networkId>network-12</networkId>
        <ipAddress>192.168.10.1</ipAddress>
        <subnetMask>255.255.255.0</subnetMask>
```

```
        <mtu>1500</mtu>
      </interface>
      <interface>
        <isUplink>true</isUplink>
        <networkId>network-13</networkId>
        <ipAddress>10.112.2.40</ipAddress>
        <subnetMask>255.255.254.0</subnetMask>
        <defaultGw>10.112.3.253</defaultGw>
        <mtu>1500</mtu>
      </interface>
    </applianceConfig>
  </installParams>
</vshieldEdgeConfig>
```

The installation schema requires the following values:

- `resourcePoolId`: Enter the VC MOID of the resource pool.
- `hostId`: Enter the VC MOID of the ESX Host to which the vShield Edge is to be cloned. Mandatory.
- `dataStoreId`: Enter the VC MOID of the Datastore to which the vShield Edge is to be cloned.
- `applianceConfig`: The `interface` should be defined twice, once with isUplink=true for the external interface, and once with isUplink=false or absent for the internal interface.

The installation schema accepts the following optional parameters for advanced configurations:

- `hostName`: This is the fully qualified domain name set on the vShield Edge virtual machine. It will be visible in the remote syslog messages generated from Edge.
- `tenantId`: Can be used to identify the tenant to which this Edge belongs. It will be visible in the remote syslog messages generated from Edge.
- `disableInternalFirewallRules`: This defaults to False. If set to true, the administrator needs to punch the required firewall holes to let the traffic generated from vShield Edge go out.
- `macAddress`: Can be used to add a vNIC with a specified MAC address. The user is responsible for validating the uniqueness of the MAC assignments on VC entities.
- `mtu`: The user can change the interface maximum transmission unit with this field. Default is 1500.
- `vmFolderId`: Specifies a particular folder on the VC where the Edge VM should be placed.
- `vseName`: Specifies host name of the Edge VM. Default is `vshieldEdge-<internal-portgroup-on-vc>`.
- `vmxParametersList`: These can be used to add configurations for the vNICs into the VMX file.
- `customField`: These can be used to define custom fields for the Edge VM.
- `memoryAllocation`: Changes memory allocation for the Edge VM, defining limits and/or reservation. Can be an absolute value (value) or a multiplier factor to the default value (multiplier). Default is 256 MB.
- `cpuAllocation`: Changes CPU allocation for the Edge VM, defining limits and/or reservation. Can be an absolute value (value) or a multiplier factor to the default value (multiplier).

Getting the Current Configuration of a vShield Edge

You can get the full configuration present on a vShield Edge, including the list of configured services, such as DHCP, NAT, and firewall rules.

Example 4-2. Get the current configuration of a vShield Edge

Request:

```
GET https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>>/edge
```

Example Configuration:

```
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <installParams>
    <operationMode>routing</operationMode>
    <version>2.0</version>
    <resourcePoolId>resgroup-41</resourcePoolId>
    <hostId>host-27</hostId>
    <dataStoreId>datastore-28</dataStoreId>
    <vmId>vm-102</vmId>
    <applianceConfig>
      <hostName>vShieldEdge-dvportgroup-63</hostName>
      <disableInternalFirewallRules>false</disableInternalFirewallRules>
      <interface>
        <networkId>dvportgroup-63</networkId>
        <ipAddress>192.168.1.1</ipAddress>
        <subnetMask>255.255.255.0</subnetMask>
        <mtu>1500</mtu>
      </interface>
      <interface>
        <isUplink>true</isUplink>
        <networkId>network-23</networkId>
        <ipAddress>10.24.128.202</ipAddress>
        <subnetMask>255.255.252.0</subnetMask>
        <defaultGw>10.24.131.253</defaultGw>
        <mtu>1500</mtu>
      </interface>
    </applianceConfig>
    <vmFolderId>group-v3</vmFolderId>
  </installParams>
  <natConfig>
    <rule>
      <type>snat</type>
      <internalIpAddress>any</internalIpAddress>
      <externalIpAddress>10.24.130.250</externalIpAddress>
      <enableLog>false</enableLog>
    </rule>
  </natConfig>
  <firewallConfig>
    <defaultPolicy>deny</defaultPolicy>
    <enableLoggingForDefaultPolicy>false</enableLoggingForDefaultPolicy>
    <blockIcmpErrors>false</blockIcmpErrors>
    <rule>
      <networkId>dvportgroup-63</networkId>
      <protocol>tcp</protocol>
      <destinationPort>any</destinationPort>
      <destinationIpAddress>
        <ipAddress>any</ipAddress>
      </destinationIpAddress>
      <sourcePort>any</sourcePort>
      <sourceIpAddress>
        <ipAddress>192.168.0.0</ipAddress>
      </sourceIpAddress>
      <direction>in</direction>
      <action>allow</action>
      <enableLog>false</enableLog>
      <disabled>false</disabled>
    </rule>
    <rule>
      <networkId>network-23</networkId>
      <protocol>tcp</protocol>
      <destinationPort>any</destinationPort>
      <destinationIpAddress>
        <ipAddress>any</ipAddress>
      </destinationIpAddress>
      <sourcePort>any</sourcePort>
      <sourceIpAddress>
        <ipAddress>192.168.0.0</ipAddress>
      </sourceIpAddress>
      <direction>in</direction>
```

```
      <action>allow</action>
      <enableLog>false</enableLog>
      <disabled>false</disabled>
    </rule>
  </firewallConfig>
  <ipsecSiteToSiteConfig>
    <globalConfig>
      <id>10.24.131.253</id>
      <ipAddress>10.24.128.202</ipAddress>
      <enableLog>false</enableLog>
    </globalConfig>
  </ipsecSiteToSiteConfig>
</vshieldEdgeConfig>
```

Uninstalling a vShield Edge

You can uninstall a vShield Edge appliance.

Example 4-3. Uninstall a vShield Edge

Request:

```
DELETE https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
```

vShield Edge Management 5

You can manage vShield Edge services and firewall policies with the REST API. You can install Edge, post and delete configurations, and get status of various services.

This chapter includes the following topics:

IMPORTANT All vShield REST requests require authorization. See "Using the vShield REST API" on page 12 for details about basic authorization.

Configuring vShield Edge

The set of APIs in this section perform vShield Edge installation, configuration, and deletion.

List vShield Edge Installations

This call returns a list of all the vShield Edge appliances installed by vShield Manager.

Example 5-1. Get the vShield Edge installations

Request:

```
GET https://<vsm-ip>/api/2.0/networks/edge/capability
```

For each vShield Edge, it shows capability for the VPN, load balancer, NAT, firewall, DHCP, and static routing. It says on what dvPortGroup the Edge is installed, the Edge version number, and compatibility mode.

Determine API Version

This call determines the API version that the vShield Edge can process. If the Edge is working in backward compatible mode, only 1.0 version APIs are allowed. If the Edge is working in regular mode, only 2.0 version APIs are allowed. To start accepting 2.0 APIs, see "Switch to New API Version" on page 38.

Example 5-2. Determine API version

Request:

```
GET https://<vsm-ip>/api/versions/edge/<internal-portgroup-vc-moref-id>
```

Get Capabilities of a vShield Edge

This call returns capabilities of the vShield Edge installed on the specified portgroup.

The <internal-portgroup-vc-moref-id> is the managed object reference of a network or dvPortGroup.

Example 5-3. Get Capabilities of a vShield Edge

Request:

```
GET https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge/capability
```

For the specified vShield Edge, it shows capability for the VPN, load balancer, NAT, firewall, DHCP, and static routing. It also shows the dvPortGroup's ID, Edge version number, and compatibility mode.

Switch to New API Version

The vShield 4.1 REST calls (API version 1.0) work when vShield 5.0 (API 2.0) is in backward compatibility mode. In regular mode, after you enable API 2.0, the REST calls are not backward compatible. Note:

- Clients can continue to use REST 1.0 on their already installed Edges. New Edges can be also be installed and use REST 1.0, however new features are not available without using the REST 2.0 API.
- To use new features exposed in version 2.0 of the REST API, you must upgrade the vShield Edge (see "Request Sync or Upgrade" on page 50) and switch to the new API version (see Example 5-4 below).
- Once the switch is made, you cannot downgrade to (revert to) version 1.0 of the REST API.

This call shown in Example 5-4 switches the REST mode to latest, so REST 1.0 can no longer administer the vShield Edge associated with <internal-portgroup-vc-moref-id>. The compatibility mode switches from backward compatibility to regular.

Example 5-4. Enable new API on vShield Edge

Request:

```
PUT https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/enable
```

Get Full Configuration of a vShield Edge

This API is used to read the full configuration present on Edge.

The <internal-portgroup-vc-moref-id> is the managed object reference of a network or dvPortGroup.

Example 5-5. Get capabilities of a vShield Edge

Request:

```
GET https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
```

The response body may include these service configuration tags: installParams, routeConfig, natConfig, firewallConfig, dhcpConfig, dhcpService, loadBalancerConfig, loadBalancerService, certificateStoreConfig, ipsecSiteToSiteConfig, ipsecSiteToSiteService, syslogServerConfig, among others.

Change Configuration of a vShield Edge

This call changes the configuration of a vShield Edge. The <internal-portgroup-vc-moref-id> is the managed object reference of a dvPortGroup. All services, or just one, can be configured using this call.

A vShield Edge license is required. Edge Basic license includes: Install, NAT, Firewall, DHCP, static routing. Edge Premium license handles: LoadBalancer, VPN.

If a service configuration tag is present, it means replace the configuration. If a service configuration tag's block is empty, it means delete the configuration. If a service configuration tag is absent, it means do not change anything, and hence the previous configuration for that service is retained as is.

Example 5-6. Change configuration of a vShield Edge

Request:

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
```

Request Body:
see examples below.

Install vShield Edge

The post call configures a vShield Edge, as describe in "Installing a vShield Edge" on page 33.

Delete vShield Edge

The delete call uninstalls vShield Edge, as described in "Uninstalling a vShield Edge" on page 36.

Configuring Edge Services

You configure Edge services such as NAT, Firewall, DHCP, static routing. Load Balancer, and VPN with the API shown in Example 5-6. The following request bodies show various configurations made on vShield Edge.

IMPORTANT When you configure a vShield Edge service, the service is started on the appliance. If you do not want the service running, you must stop the service using an appropriate system command.

Configure DHCP

vShield Edge provides DHCP service to bind assigned IP addresses to MAC addresses, helping to prevent MAC spoofing attacks. All virtual machines protected by a vShield Edge can obtain IP addresses dynamically from the vShield Edge DHCP service.

vShield Edge supports IP address pooling and one-to-one static IP address allocation based on the vCenter managed object ID (`vmId`) and interface ID (`interfaceId`) of the requesting client. All DHCP settings configured by REST requests appear under the **vShield Edge > DHCP** tab for the appropriate vShield Edge in the vShield Manager user interface and in vSphere Client plug-in.

vShield Edge DHCP service adheres to the following rules:

- Listens on the vShield Edge internal interface (non-uplink interface) for DHCP discovery.
- As stated above, `vmID` specifies the `vc-moref-id` of the virtual machine, and `interfaceId` specifies the index of the `vNic` for the requesting client. The `hostName` is an identification of the binding being created. This `hostName` is not pushed as the specified host name of the virtual machine.
- By default, all clients use the IP address of the internal interface of the vShield Edge as the default gateway address. To override it, specify `defaultGw` under the `configParams Interface`, per binding or per pool. The client's `broadcast` and `subnetMask` values are from the internal interface for the container network.
- `configParams` and its elements are optional.
- `leaseTime` can be infinite, or a number of seconds. If not specified, the default lease time is 1 day.
- Logging is disabled by default. To enable logging, add a `<log />` element within the `<dhcpConfig>` block.

For the DHCP schema, see "vShield Edge Schemas" on page 88. Sample XML request body:

Example 5-7. Configure DHCP service

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
```

```
vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <dhcpConfig>
    <binding>
      <vmId>vm-21</vmId>
      <interfaceId>1</interfaceId>
      <hostName>DlpServer</hostName>
      <internalIpAddress>192.168.10.11</internalIpAddress>
      <configParams>
        <domainName>test.com</domainName>
        <primaryNameServer>10.112.0.1</primaryNameServer>
        <secondaryNameServer>10.112.0.2</secondaryNameServer>
      </configParams>
    </binding>
    <pool>
      <ipRange>192.168.10.2-192.168.10.10</ipRange>
      <configParams>
        <leaseTime>infinite</leaseTime>
      </configParams>
    </pool>
  </dhcpConfig>
</vshieldEdgeConfig>
```

Manage the DHCP Service

To start DHCP service, specify `up`. To stop DHCP service, specify `down`.

Example 5-8. Start DHCP service

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <dhcpService>up</dhcpService>
</vshieldEdgeConfig>
```

Example 5-9. Stop DHCP service

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <dhcpService>down</dhcpService>
</vshieldEdgeConfig>
```

Delete DHCP Configuration

Example 5-10. Delete DHCP configuration

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <dhcpConfig/>
</vshieldEdgeConfig>
```

Configure Firewall

The vShield Edge provides firewall protection for incoming and outgoing sessions. In addition to the default firewall policy, you can configure a set of rules to allow or deny traffic sessions to and from specific sources and destinations. You manage the default firewall policy and firewall rules together for each vShield Edge agent. You must specify both firewall rules and `defaultPolicy` together whenever modifying either of them, or else the one you do not specify will be deleted.

Firewall rules for a vShield Edge configured by using REST requests appear under the **vShield Edge > Firewall** tab for the appropriate vShield Edge in the vShield Manager user interface and in the vSphere Client plug-in.

For the Edge firewall schema, see "vShield Edge Schemas" on page 88. Sample XML request body:

Example 5-11. Configure firewall

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <firewallConfig>
    <defaultPolicy>deny</defaultPolicy>
    <enableLoggingForDefaultPolicy>false</enableLoggingForDefaultPolicy>
    <blockIcmpErrors>false</blockIcmpErrors>
    <rule>
      <networkId>network-12</networkId>
      <protocol>icmp</protocol>
      <icmpType>address-mask-reply</icmpType>
      <destinationIpAddress>
        <ipAddress>10.112.2.150</ipAddress>
      </destinationIpAddress>
      <sourceIpAddress>
        <ipAddress>any</ipAddress>
      </sourceIpAddress>
      <direction>out</direction>
      <action>deny</action>
      <enableLog>true</enableLog>
      <disabled>false</disabled>
    </rule>
  </firewallConfig>
</vshieldEdgeConfig>
```

After this firewall configuration, the administrator can define firewall rules on internal or external (using the dvPortgroup's managed object ID), or on the `vpnInterface` of the Edge. Rules can be defined using IPSet and Applications Grouping Objects defined on the appropriate scope. Notes:

- You can add multiple firewall rules by entering multiple `<rule></rule>` sections in the body.
- The `vpnInterface` is the external public address of the VPN.
- For `<protocol>` options `tcp` and `udp`, you must specify `sourcePort` and `destinationPort` elements. For options `icmp` and `any`, the `sourcePort` and `destinationPort` elements are not expected. Other protocol options include `igmp`, `ipencap`, `rsvp`, `gre`, `l2tp`, `sctp`, and `ipv6`. Also you have the flexibility to provide a new `protocolName` if the protocol is not listed by name in the `<protocol>` tag.
- You must add `<icmpType>` if you configure `icmp` as the protocol.
- Logging is disabled by default. To enable it, add `<enableLog>` `true` element within the `<rule>` section.
- The `sourceIpAddress` and `destinationIpAddress` can be entered in one of these formats:
  - `<ipAddress>` specified as a single IP address, a hyphen-separated IP address range (for example, `192.168.10.1-192.168.10.2555`) or a subnet in CIDR notation (`198.168.10.1/24`)
  - the keyword `any`
  - an `<ipSetIdentifier>`, the managed object ID of an IPset
- The `sourcePort` and `destinationPort` parameters can be entered in one of the following formats: the keyword `any`, the port number as an integer, or a range of port number, for example `portX-portY`.
- An `applicationIdentifier` from Grouping Objects can replace the destination port and protocol.
- The `disabled` parameter means to remember the rule on vShield Manager but not push the rule onto the Edge appliance. This is optional and defaults to `false` (push to appliance).
- The `blockIcmpErrors` parameter is advanced configuration. It is optional and defaults to `false`.

Change Firewall Rule to Allow

This deletes previously configured firewall rules and sets `allow` as the default policy.

Example 5-12. Set firewall policy to allow all

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <firewallConfig>
    <defaultPolicy>allow</defaultPolicy>
  </firewallConfig>
</vshieldEdgeConfig>
```

Revert Firewall to Default

This returns the firewall to default configuration (deny) by deleting existing rules.

Example 5-13. Reset firewall to defaults

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <firewallConfig />
</vshieldEdgeConfig>
```

Create Firewall Rule with IPset or applicationSet

To get the ID of the `IPset` or `applicationSet`, see example Example 2-18, “List IPsets on a scope,” on page 20 or Example 2-36, “List applications on a given scope,” on page 25.

If the referenced `IPset` or `applicationSet` is deleted, the rule will the disabled on the Edge appliance.

The default policy in effect is to `deny`. The example below sets it to `allow` based on two rules.

Example 5-14. IPset or applicationSet based firewall rule

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <firewallConfig>
    <defaultPolicy>allow</defaultPolicy>
    <enableLoggingForDefaultPolicy>false</enableLoggingForDefaultPolicy>
    <blockIcmpErrors>false</blockIcmpErrors>
    <rule>
      <networkId>network-12</networkId>
      <applicationIdentifier>application-20</applicationIdentifier>
      <destinationIpAddress>
        <ipAddress>10.112.2.49</ipAddress>
      </destinationIpAddress>
      <sourcePort>any</sourcePort>
      <sourceIpAddress>
        <ipsetIdentifier>ipset-2</ipsetIdentifier>
      </sourceIpAddress>
      <direction>in</direction>
      <action>allow</action>
      <enableLog>false</enableLog>
      <disabled>false</disabled>
      <comments>Used IpSet</comments>
    </rule>
    <rule>
      <networkId>network-12</networkId>
      <protocol>icmp</protocol>
      <icmpType>address-mask-reply</icmpType>
      <destinationIpAddress>
        <ipAddress>10.112.2.150</ipAddress>
      </destinationIpAddress>
      <sourceIpAddress>
        <ipAddress>any</ipAddress>
      </sourceIpAddress>
      <direction>out</direction>
      <action>deny</action>
      <enableLog>true</enableLog>
```

```
      <disabled>false</disabled>
    </rule>
  </firewallConfig>
</vshieldEdgeConfig>
```

Delete Firewall Configuration

Example 5-15. Delete firewall rules

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <firewallConfig/>
</vshieldEdgeConfig>
```

After removing a firewall configuration, the default policy that will be in effect is `Deny`.

Configure Static Routing

This uses the next-hop method for the outgoing interface. Attribute `networkId` specifies the managed object ID of the network, attribute `network` designates the IP address range, and `nextHop` the static route.

Example 5-16. Configure static route

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <routeConfig>
    <staticRoute>
      <networkId>network-12</networkId>
      <network>192.168.30.0/24</network>
      <nextHop>192.168.10.253</nextHop>
    </staticRoute>
  </routeConfig>
</vshieldEdgeConfig>
```

For the data path to work, you need to change the default firewall policy to ALLOW, or punch Firewall rules to allow data traffic on external and internal interfaces.

Delete the Static Routing

Example 5-17. Delete static route

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <routeConfig>
    <staticRoute/>
  </routeConfig>
</vshieldEdgeConfig>
```

Configure NAT

The vShield Edge provides network address translation (NAT) service to protect the IP addresses of internal (private) networks from the public network. You can configure NAT rules to provide access to services running on privately addressed virtual machines. There are two types of NAT rules that can be configured: SNAT and DNAT. When you post a NAT configuration, all the rules (both SNAT and DNAT) must be posted together. Otherwise, only the posted rules are retained, and unposted rules are deleted.

All SNAT and DNAT rules configured by using REST requests appear under the **vShield Edge > NAT** tab for the appropriate vShield Edge in the vShield Manager user interface and in the vSphere Client plug-in.

For the NAT schema, see "vShield Edge Schemas" on page 88. Sample XML request body:

Example 5-18. Configure NAT service

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge

<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <natConfig>
    <rule>
      <type>snat</type>
      <internalIpAddress>192.168.10.11</internalIpAddress>
      <externalIpAddress>10.112.2.146</externalIpAddress>
      <enableLog>false</enableLog>
    </rule>
    <rule>
      <type>dnat</type>
      <protocol>tcp</protocol>
      <internalIpAddress>192.168.10.2-192.168.10.12</internalIpAddress>
      <internalPort>any</internalPort>
      <externalIpAddress>10.112.2.146</externalIpAddress>
      <externalPort>any</externalPort>
      <enableLog>true</enableLog>
    </rule>
  </natConfig>
</vshieldEdgeConfig>
```

For the data path to work, you need to change the default firewall policy to ALLOW, or punch Firewall rules to allow data traffic on external and internal interfaces.

Rules:

- For `<protocol>` options `tcp` and `udp`, you must specify `sourcePort` and `destinationPort` elements. For options `icmp` and `any`, the `sourcePort` and `destinationPort` elements are not expected
- You must add `<icmpType>` if you configure `icmp` as the protocol.
- The `externalIpAddress` and `internalIpAddress` elements can be entered in either of these methods:
  - `<ipAddress>` specified as a single IP address, a hyphen-separated IP address range (for example, `192.168.10.1-192.168.10.2555`) or a subnet in CIDR notation (`198.168.10.1/24`).
  - the keyword `any`
- The `externalPort` and `internalPort` parameters can be entered in one of the following formats: the keyword `any`, the port number as an integer, or a range of port number, for example `portX-portY`.
- You can add multiple SNAT rules by entering multiple `<type>snat</type>` sections in the body.
- SNAT does not support `port` or `protocol` parameters.
- Logging is disabled by default. To enable logging, add an `<enableLog>` element set to `true`.

Delete NAT Configuration

To delete NAT configuration, post an empty rule set.

Example 5-19. Delete NAT rules

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <natConfig/>
</vshieldEdgeConfig>
```

Configure Load Balancer

The vShield Edge provides load balancing for HTTP traffic. Load balancing (up to Layer 7) enables Web application auto-scaling. To implement load balancing, you map an external (or public) IP address to a set of internal servers. The load balancer accepts HTTP requests on the external IP address and decides which internal server to use. Port 80 is the default listening port for load balancer service.

All Load Balancer settings configured by using REST requests appear under the **vShield Edge > Load Balancer** tab for the appropriate vShield Edge in the vShield Manager user interface and in the vSphere Client plug-in.

For the load balancer schema, see "vShield Edge Schemas" on page 88. Sample XML request body:

Example 5-20. Configure load balancer

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <loadbalancerConfig>
    <listener>
      <externalIpAddress>10.112.2.148</externalIpAddress>
      <backEndServerConfig>
        <ipAddress>192.168.10.11</ipAddress>
        <port>80</port>
      </backEndServerConfig>
      <algorithm>ip-hash</algorithm>
      <enableLog>false</enableLog>
    </listener>
  </loadbalancerConfig>
</vshieldEdgeConfig>
```

For the data path to work, you need to change the default firewall policy to ALLOW, or punch Firewall rules to allow data traffic on external and internal interfaces. Rules:

- You can map a global or public IP address to a set of internal servers for load balancing. The load balancer accepts HTTP requests on the `<ipAddress>` specified. If `<port>` is not given, 80 is the default port.
- The `<backEndServerConfig>` is a list of one or more IP addresses representing servers to use for load balancing.
- vShield Manager processes the posted XML file as a complete set of load balancing servers for the network specified. The current set of load balancing servers for a network is replaced with this new set of servers.
- You can add multiple servers as listeners by entering multiple `<listener>` sections in the body.
- You can configure the algorithm that is used to determine load balancing. The optional `<algorithm>` element can be set to `round-robin` (the default) or `ip-hash`.
- Logging is disabled by default. To enable logging, add a `<enableLog>` element set to `true`.

Manage Load Balancer Service

Example 5-21. Start load balancer

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <loadbalancerService>up</loadbalancerService>
</vshieldEdgeConfig>
```

Example 5-22. Stop load balancer

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <loadbalancerService>down</loadbalancerService>
</vshieldEdgeConfig>
```

Delete Load Balancer Configuration

Example 5-23. Delete load balancer configuration

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <loadbalancerConfig/>
</vshieldEdgeConfig>
```

Miscellaneous

You also use the Edge POST call to reconfigure IP interfaces, change credentials, and start remote logging.

Reconfigure Edge Interfaces

Example 5-24. Reconfigure IP interfaces

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <applianceConfig>
    <hostName>vShieldEdge-network-12</hostName>
    <interface>
      <networkId>network-12</networkId>
      <ipAddress>192.168.10.12</ipAddress>
      <subnetMask>255.255.255.0</subnetMask>
      <mtu>1500</mtu>
    </interface>
    <interface>
      <isUplink>true</isUplink>
      <networkId>network-13</networkId>
      <ipAddress>10.112.2.151</ipAddress>
      <subnetMask>255.255.254.0</subnetMask>
      <defaultGw>10.112.3.253</defaultGw>
      <mtu>2000</mtu>
    </interface>
  </applianceConfig>
</vshieldEdgeConfig>
```

You cannot change the `macAddress` or the `portGroup` of the interface that is currently attached.

Set vShield Edge Credentials

Example 5-25. Set vShield Edge credentials

```
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <cliLoginCredentials>
    <username>test</username>
    <password>new-secret</password>
  </cliLoginCredentials>
</vshieldEdgeConfig>
```

Configure Remote Logging

You can configure a remote `syslog` server for vShield logging at the designated IP address. A maximum of two IP addresses can be configured.

Example 5-26. Configure remote logging

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <syslogServerConfig>
    <ipAddress>10.112.2.149</ipAddress>
  </syslogServerConfig>
```

```
</vshieldEdgeConfig>
```

Configure VPN

vShield Edge agents support site-to-site IPsec VPN between an Edge appliance and remote sites. On both ends, static one-to-one NAT is required for the VPN address. vShield Edge agents support pre-shared key mode, x/5-0 Certificate mode, IP unicast traffic, and no dynamic routing protocol between the Edge and remote VPN routers. Behind each remote VPN router, you can configure multiple subnets to connect the internal network behind a vShield Edge through IPSec tunnels. Subnets and the internal network behind vShield Edge must have non-overlapping address ranges.

You can deploy a vShield Edge agent behind a NAT device, which translates the Edge agent's VPN address into a public accessible address facing the Internet; remote VPN routers use this public address to access the vShield Edge. Remote VPN routers can be located behind a NAT device as well. You must provide both the VPN native address and the NAT public address to set up the tunnel.

All VPN settings configured by using REST requests appear under the **vShield Edge > VPN** tab for the appropriate vShield Edge in the vShield Manager user interface and in the vSphere Client plug-in.

For the VPN schema, see "vShield Edge Schemas" on page 88. Sample XML request body:

Example 5-27. Configure a VPN

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <ipsecSiteToSiteConfig>
    <globalConfig>
      <id>10.112.2.50</id>
      <ipAddress>10.112.2.50</ipAddress>
      <enableLog>false</enableLog>
    </globalConfig>
    <siteConfig>
      <peerName>site1</peerName>
      <peerId>site1</peerId>
      <peerIpAddress>10.112.2.145</peerIpAddress>
      <localSubnet>192.168.10.0/24</localSubnet>
      <peerSubnet>192.168.20.0/24</peerSubnet>
      <authenticationMode>psk</authenticationMode>
      <preSharedKey>test</preSharedKey>
      <encryptionAlgorithm>3des</encryptionAlgorithm>
      <enablePfs>true</enablePfs>
      <dhGroup>dh2</dhGroup>
    </siteConfig>
  </ipsecSiteToSiteConfig>
</vshieldEdgeConfig>
```

For the data path to work, you need to change the default firewall policy to `allow`, or punch Firewall rules to allow data traffic on VPN and internal interfaces. Rules:

- The `<id>` is a unique ID used by all peers to identify this vShield Edge VPN gateway. In the example, it is the same as `<ipAddress>`.
- Similar to the `preSharedKey` in `siteConfig`, the optional `preSharedKeyForDynamicIpSites` in `globalConfig` is a pre-shared key for use by all peers when connecting with an unknown IP address.
- The `<peerName>` a descriptive name of the peer.
- The `<peerId>` is an ID to uniquely identify the peer, used to define policies for the peer and for peer authentication. For peers using certificate authentication, this ID must be the common name in the peer's certificate. For PSK peers, this ID can be any string, but ideally the public IP address of the VPN or the FQDN for the VPN service.

- The <peerIpAddress> can be any, or an actual IP address. If any, then this side can be a responder only, waiting for the peer to initiate connection. The preSharedKeyForDynamicIpSites (see above) must be configured in order to match a peer from "any" peerIpAddress, and all peers from "any" must be configured to share the global pre-shared key. If an IP address is specified, the address should be the peer's public address that the vShield Edge can reach to make connection. This address is also required to create the site-level pre-shared key secret entry for this site.
- The <encryptionAlgorithm> can be 3des, aes, or aes256.
- If <enablePfs> is set true, Perfect Forward Secrecy (PFS) is enabled. In IPsec negotiations, PFS ensures that each new cryptographic key is unrelated to any previous key. The default is true (enabled). You must enable or disable PFS on both the tunnel peers, otherwise the IPsec tunnel cannot be established.
- The <dhGroup> can be dh2 (the default) or dh5. This is needed to support VPN across vendors. DH means Diffie-Hellman, a public-key cryptography scheme that allows two parties to establish a shared secret over an insecure communications channel.
- Logging is disabled by default. To enable logging, add an <enableLog> element set to true.
- VPN service requires encryption. Specify the <encryptionAlgorithm> element as either 3des or aes.

Manage VPN Service

Example 5-28. Start VPN service

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <ipsecSiteToSiteService>up</ipsecSiteToSiteService>
</vshieldEdgeConfig>
```

Example 5-29. Stop VPN service

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <ipsecSiteToSiteService>down</ipsecSiteToSiteService>
</vshieldEdgeConfig>
```

Delete the VPN Configuration

Example 5-30. Delete VPN configuration

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <ipsecSiteToSiteConfig/>
</vshieldEdgeConfig>
```

Generate Certificate Signing Request (CSR)

You can generate a CSR for vShield Edge. A certificate is required to configure VPN in authentication mode.

Example 5-31. Generate CSR

Request:

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge/csr
```

Sample Request Body:

```
<vshieldEdgeConfig xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="vmware.vshield.edge.2.0">
  <certificateStoreConfig>
    <csrParams>
      <commonName>up.example.com</commonName>
      <organization>Example Inc</organization>
```

```
      <department>Engg</department>
      <city>Pune</city>
      <state>MH</state>
      <country>IN</country>
      <keySize>1024</keySize>
    </csrParams>
  </certificateStoreConfig>
</vshieldEdgeConfig>
```

The call returns a CSR, which you send to the certifying authority (CA), who returns a security certificate.

Add X.509 Certificate as VPN Site

1 Generate a certificate signing request (CSR).

2 Have the CSR certified by a certificate authority (CA). Also get the certificate of this CA (caCertificate).

3 Upload the caCertificate and the vShield Edge certificate.

Example 5-32. Upload security certificates

```
POST https://<vsm-ip>/api/2.0/networks/<network-ID>/edge
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <certificateStoreConfig>
    <caCertificate>...enter_text_here...</caCertificate>
    <certificate>...enter_text_here...</certificate>
  </certificateStoreConfig>
</vshieldEdgeConfig>
```

4 When using x.509 authentication mode, the globalConfig should have certificateCn specified.

5 Add the site configuration to an existing site with this request body.

Example 5-33. Add site certificates

```
<vshieldEdgeConfig xmlns="vmware.vshield.edge.2.0">
  <ipsecSiteToSiteConfig>
    <globalConfig>
      <id>10.112.2.50</id>
      <certificateCn>up.vmware.com</certificateCn>
      <ipAddress>10.112.2.50</ipAddress>
      <enableLog>false</enableLog>
    </globalConfig>
    <siteConfig>
      <peerName>site1</peerName>
      <peerId>site1</peerId>
      <peerIpAddress>10.112.2.145</peerIpAddress>
      <localSubnet>192.168.10.0/24</localSubnet>
      <peerSubnet>192.168.20.0/24</peerSubnet>
      <authenticationMode>psk</authenticationMode>
      <preSharedKey>test</preSharedKey>
      <encryptionAlgorithm>3des</encryptionAlgorithm>
      <enablePfs>true</enablePfs>
      <dhGroup>dh2</dhGroup>
    </siteConfig>
    <siteConfig>
      <peerName>site2</peerName>
      <peerId>up.vmware.com</peerId>
      <peerIpAddress>10.112.2.148</peerIpAddress>
      <localSubnet>192.168.30.0/24</localSubnet>
      <peerSubnet>192.168.40.0/24</peerSubnet>
      <authenticationMode>x.509</authenticationMode>
      <encryptionAlgorithm>aes</encryptionAlgorithm>
      <mtu>1500</mtu>
    </siteConfig>
  </ipsecSiteToSiteConfig>
```

```
</vshieldEdgeConfig>
```

6 For the data path to work, you need to change the default firewall policy to `allow`, or punch Firewall rules to allow data traffic on VPN and internal interfaces

Operating vShield Edge

The set of APIs in this section perform vShield Edge discovery and operations.

Get Details About Edge

You can retrieve the details of a vShield Edge configuration. This shows the internal rules punched through the vShield Edge to make load balancing and VPN work.

Example 5-34. Retrieve vShield Edge details

Request:

```
GET https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge/detailed
```

This returns the current and install-time hostId, dataStoreId, vmId, and other details that give administrators insight into whether the Edge VM got vMotioned or altered.

Request Sync or Upgrade

You can synchronize or upgrade vShield Edge.

Example 5-35. Request an action from vShield Edge

Request:

```
POST https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge?action=<actiontype>
```

The `<...moref-id>` refers to some network entity. The `<actiontype>` can be one of the following:

- `forcesync` – force Edge to synchronize with the last good configuration in the vShield Manager database.

  ```
  POST https://<vsm-ip>/api/2.0/networks/dvportgroup-63/edge?action=forcesync
  ```

- `upgrade` – upgrade the Edge to the latest version if the OVF is available.

  ```
  POST https://<vsm-ip>/api/2.0/networks/dvportgroup-63/edge?action=upgrade
  ```

- `forceupgrade` – upgrade to the latest available version, creating one if an existing Edge is not found.

  ```
  POST https://<vsm-ip>/api/2.0/networks/dvportgroup-63/edge?action=forceupgrade
  ```

Get IPsec Tunnel Statistics

You can retrieve statistics about the IPsec tunnel.

Example 5-36. Get IPsec statistics

Request:

```
GET https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge/statistics/ipsec
```

Get DHCP Statistics

You can retrieve DHCP lease statistics, including details about leased IPs from the configured IP Pools.

Example 5-37. Get DHCP statistics

Request:

```
GET https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge/statistics/dhcp
```

Network Interface Statistics

You can retrieve traffic status, including external and internal interfaces, per interface, VPN to remote subnets, vShield Edge traffic processed, and dropped counters due to user-configured firewall rules.

Example 5-38. Get traffic statistics

Request:

```
GET https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge/statistics/interface
```

Get Service Status

You can retrieve the status of various services, where `<svc>` could be:

- `dhcp` – returns status of DHCP service, up or down depending on the Edge appliance's DHCP daemon.

  ```
  GET https://<vsm-ip>/api/2.0/networks/dvportgroup-63>/edge/dhcp/service
  ```

- `vpn` – returns status of VPN service, up or down depending on the Edge appliance's VPN daemon.

  ```
  GET https://<vsm-ip>/api/2.0/networks/dvportgroup-63>/edge/vpn/service
  ```

- `loadbalancer` – returns status of load balancer service, depending on the Edge appliance's daemon.

  ```
  GET https://<vsm-ip>/api/2.0/networks/dvportgroup-63>/edge/loadbalancer/service
  ```

Example 5-39. Get service status

Request:

```
GET https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge/<svc>/service
```

Debugging and Support

To help with your own debugging and to provide information for VMware technical support, APIs are available to retrieve vShield logs and get statistics about Edge services.

Retrieve Logs for Technical Support

This call provides the technical support logs from vShield Edge. These are often required for debugging purposes. The call returns the location where the compressed log files are downloaded.

Example 5-40. Get support logs

Request:

```
GET https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge/techSupportLogs
```

The technical support log is placed in a file, however the REST API has no provision for downloading it, and `wget` and `curl` do not have permission to download it, either. You can retrieve the log with vShield Manager by clicking **Settings & Reports > Configuration > Support > [Log Download] Initiate**.

Get Service Statistics

You can retrieve the vShield Edge service statistics. These are often required for debugging purposes. The call return the location where the service statistics text file is downloaded.

Example 5-41. Get service statistics

Request:

```
GET https://<vsm-ip>/api/2.0/networks/<internal-portgroup-vc-moref-id>/edge/serviceStats
```

vShield App Management 6

You can configure vShield App firewall rules and syslog service by using REST API calls.

This chapter includes the following topics:

IMPORTANT All vShield REST requests require authorization. See "Using the vShield REST API" on page 12 for details about basic authorization.

Modifying the State of a Datacenter

The state of a datacenter is determined by the version of the vShield Manager on that datacenter. For a 5.0 vShield Manager, the datacenter is in the `regular` state which means only the 5.0 API calls are supported.

When the vShield Manager on a datacenter is upgraded from a previous release, the datacenter is in the `backwardCompatible` mode which means that only the APIs from the previous release are supported. When the vShield App components on that datacenter are upgraded to 5.0, the datacenter state is automatically changed from `backwardCompatible` to `backwardCompatibleReadyForSwitch`. This means that the vShield App components are running in backward compatible mode, so only the APIs from the previous release are supported.

When the datacenter is in the `backwardCompatibleReadyForSwitch` state, you can switch the datacenter state to `migrating`. In the migrating state, data from the old vShield App is migrated to the 5.0 vShield App. Once the data migration is complete, the datacenter state switches automatically to `regular`.

Retrieve Datacenter State

You can retrieve the state of the datacenter.

Example 6-1. Retrieve the datacenter state

Example:

```
GET https://<vsm-ip>/api/2.0/app/firewall/datacenter-2/state
```

The XML response represents the DatacenterState object, containing an enumeration of datacenter status. The state could be regular, upgrading, migrating, backwardCompatible, or backwardCompatibleReadyForSwitch.

Modify Datacenter State

You can change the state of a datacenter only if it is in the `backwardCompatibleReadyForSwitch` state.

Example 6-2. Change datacenter state to migrating

Example:

```
POST https://<vsm-ip>/api/2.0/app/firewall/datacenter-2/state
```

Configuring Firewall Rules for vCenter

The primary function of a vShield App is to provide firewall protection on an ESX host by inspecting each session and returning details to the vShield Manager. Traffic details include sources, destinations, direction of sessions, applications, and ports being used. Traffic details can be used to create firewall allow or deny rules.

In the vShield Manager user interface or vSphere Client plug-in, the **App Firewall** tab contains the firewall rules enforced by vShield App instances. You can manage App Firewall rules at the datacenter, cluster, and port group levels to provide a consistent set of rules across multiple vShield App instances. As membership in these containers can change dynamically, App Firewall maintains the state of existing sessions without requiring reconfiguration of firewall rules. In this way, App Firewall effectively has a continuous footprint on each ESX host under the managed containers.

When creating App Firewall rules, you can create general rules based on incoming or outgoing traffic at the container level. For example, you can create a rule to deny any traffic from outside of a datacenter that targets a destination within the datacenter. You can create a rule to deny any incoming traffic that is not tagged with a VLAN ID.

All firewall rules configured by using REST requests appear under the **App Firewall** tab for the appropriate container in the vShield Manager user interface and vSphere Client plug-in.

For the complete firewall XML schema, see "vShield App Firewall Schema" on page 82.

Configuring the vShield App Firewall

Firewall precedence is hierarchical at each level. At the datacenter level, choices are DEFAULT, HIGH, or LOW. At the cluster and dvPortgroup level, firewall precedence is often set to NONE.

Each vShield App enforces the firewall rules in top-to-bottom ordering. A vShield App checks each traffic session against the top rule in the firewall table before moving down the subsequent rules in the table. The first rule in the table that matches the traffic parameters is enforced. See the *vShield Administration Guide* for more information about the hierarchy of vShield App firewall rules.

Query the Firewall Configuration

You can retrieve the firewall configuration associated with a datacenter, cluster, or dvPortGroup. The template for the API is as follows:

```
GET https://<vsm-ip>/api/2.0/app/firewall/<context>/config?list=<L>&precedence<P>&rulesType<R>&configId=<C>
```

`where`

- `<context>` is the context ID of a datacenter, cluster, or dvPortGroup.
- `<L>` is the listing type, one of the following:
  - `status` for brief current state
  - `config` for firewall configuration (the default)
  - `history` for configuration history

    - `consolidated` for combined configuration including all rules applicable in the context/
- <P> is the rule precedence, either `HIGH`, `LOW`, `DEFAULT`, or `NONE`.
- <R> can be `LAYER3` or `LAYER2` to filter the configuration rules for layer 3 or layer 2.
- <C> is the configuration ID used in conjunction with the history listing type.

Example 6-3. Queries for firewall configuration

Get quick status:

```
GET https://<vsm-ip>/api/2.0/app/firewall/dvportgroup-63/config?list=status
```

Get configuration of only high precedence rules:

```
GET https://<vsm-ip>/api/2.0/app/firewall/dvportgroup-63/config?list=config&precedence=HIGH
```

Get configuration of only layer 2 firewall rules:

```
GET https://<vsm-ip>/api/2.0/app/firewall/dvportgroup-63/config?list=config&rulesType=LAYER3
```

Get consolidated configurations for the context:

```
GET https://<vsm-ip>/api/2.0/app/firewall/datacenter-2/config?list=consolidated
```

Get a particular configuration history for a given context:

```
GET https://<vsm-ip>/api/2.0/app/firewall/datacenter-2/config?list=history&configID=241
```

Configuration is returned as formatted XML.

Change the Firewall Configuration

You should query the current firewall configuration for the desired context before modifying any firewall settings. The response of the query API call has an Etag header. You must specify the Etag header value in the If-Match header of the POST command within double quotes. This handles handle simultaneous configuration change requests from multiple users.

Example 6-4. Change firewall configuration

Request:

```
POST https://<vsm-ip>/api/2.0/app/firewall/dvportgroup-63/config
```

Request Body:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<VshieldAppConfiguration>
  <firewallConfiguration contextId="datacenter-2">
    <layer3FirewallRule disabled="false" precedence="default" id="1001">
      <action>allow</action>
      <logged>false</logged>
      <notes/>
      <source/>
      <destination/>
    </layer3FirewallRule>
    <layer2FirewallRule disabled="false" precedence="default" id="1002">
      <action>allow</action>
      <logged>false</logged>
      <notes/>
      <destination/>
    </layer2FirewallRule>
  </firewallConfiguration>
</VshieldAppConfiguration>
```

Revert to Default Firewall Configuration

You can revert the firewall configuration for the node to its default by deleting all rules that were created for the specified context ID, including default rules. For a datacenter or IP namespace, a fresh set of default rules are substituted.

Example 6-5. Delete firewall configuration and revert to default

Example:

```
DELETE https://<vsm-ip>/api/2.0/app/firewall/<contextID>/config
```

Working with SpoofGuard

It is possible for a guest operating system to spoof its IP address so that VMware Tools would misreport it to vCenter Server. The SpoofGuard feature allows the datacenter administrator to certify and authorize reported IP addresses, and if necessary, alter them. This is done by checking the IP address against the virtual machine's MAC address, which comes from the VMX and cannot be spoofed.

The SpoofGuard feature is orthogonal to firewall rules. SpoofGuard blocks traffic if it thinks the IP is spoofed, whether or not firewall rules say to block.

Retrieve SpoofGuard Global Settings

You can retrieve SpoofGuard settings such as the status (disabled or enabled), mode of operation, timestamp, and publishing authority.

Example 6-6. Get SpoofGuard settings

Example:

```
GET https://<vsm-ip>/api/2.0/spoofGuard/globalSettings
```

Edit SpoofGuard Global Settings

You can modify the SpoofGuard settings.

Example 6-7. Edit SpoofGuard settings

Example:

```
POST https://<vsm-ip>/api/2.0/spoofGuard/globalSettings
```

Request Body:

```
<VshieldConfiguration xmlns="vmware.vshield.global.20.spoofGuard">
  <globalSettings>
    <status>enabled</status>
    <mode>trustOnFirstUse</mode>
  </globalSettings>
</VshieldConfiguration>
```

Status can be `enabled` or `disabled`. Mode can be `trustOnFirstUse` or `manual`.

Retrieve SpoofGuard IP Settings

You can retrieve a list of SpoofGuard settings, included IP addresses suspected of being forged, thus blocked.

Example 6-8. Get SpoofGuard settings

Example:

```
GET https://<vsm-ip>/api/2.0/spoofGuard/<contextID>?list=<querytype>
```

where

- `contextID` can be the datacenterID or networkID of the portGroup which has been marked as namespace.
- `querytype` can be one of these: status, active, inActive, activeSinceLastPublished, requireReview, duplicates, or unPublished.

Save SpoofGuard IP Settings

You can save a list of SpoofGuard settings.

Example 6-9. Save SpoofGuard settings

Example:

```
POST https://<vsm-ip>/api/2.0/spoofGuard/<contextID>?action=<todo>
```

The `<todo>` action could be one of: approve, delete, publish, saveApproved.

An XML representation of VnicIdList is expected in the message body for delete and approve actions. If the action is `publish` then no message body is required. If the action is saveApproved then an XML representation of VnicInfo is expected.

Working with Namespaces

A vShield namespace is a set of vNICs that share a common IP address domain. They do not have overlapping IP addresses, so they are reachable all-at-once by simple routing or switching. There is no NAT between them. Any IP address in the namespace refers to the same vNIC regardless of where you look at it from within the IP address domain.

A datacenter (as managed by vCenter Server) stores a list of vShield namespaces. The namespace itself can specify a network name as an object ID, or it can contain a list of IP addresses.

Add Namespace in a Datacenter

You can define a new vShield namespace in the datacenter specified by `<datacenter-id>`.

Example 6-10. Add namespace in a datacenter

Request:

```
POST https://<vsm-ip>/api/2.0/namespace/datacenter/<datacenter-id>
```

Request Body:

```
<VshieldConfiguration xmlns="vmware.vshield.global.20.namespace">
  <namespace type="PORTGROUP" id="0">
    <namespacePortGroup>
      <Id>network-184</Id>
    </namespacePortGroup>
  </namespace>
</VshieldConfiguration>
```

In the request, `<namespace-id>` specifies the vShield namespace name.

In the example request body, the namespace is defined as being synonymous with object `network-184`.

Get Namespace Details

You can retrieve details about a previously added vShield namespace.

Example 6-11. Get namespace details

Request:

```
GET https://<vsm-ip>/api/2.0/namespace/datacenter/<datacenter-id>/<namespace-id>
```

Delete a Namespace

You can delete a previously added vShield namespace designated by <namespace-id>.

Example 6-12. Delete namespace

Request:

```
DELETE https://<vsm-ip>/api/2.0/namespace/datacenter/<datacenter-id>/<namespace-id>
```

Show Namespaces in a Datacenter

You can retrieve a list of all vShield namespaces in the datacenter specified by <datacenter-id>.

Example 6-13. Get datacenter namespaces

Example:

```
GET https://<vsm-ip>/api/2.0/namespace/datacenter/<datacenter-id>
```

Show Port Groups that can be Marked as Namespace

You can retrieve a list of all candidate port groups in the datacenter specified by <datacenter-id>.that can be marked as a separate namespace.

Example 6-14. Get port groups that can be marked as namespace

Example:

```
GET https://<vsm-ip>/api/2.0/namespace/datacenter/<datacenter-id>?list=candidate
```

Show Configured Namespaces in Datacenter

You can retrieve a list of all configured namespaces in the datacenter specified by <datacenter-id>.

Example 6-15. Get configured namespaces in datacenter

Example:

```
GET https://<vsm-ip>/api/2.0/namespace/datacenter/<datacenter-id>?list=configured
```

Configuring Syslog Service for a vShield App

You can configure all vShield App instances to send system events to up to two syslog servers. All vShield App instances share the same syslog server configuration.

You can retrieve a list of syslog servers configured on the first vShield App instance that responds.

Example 6-16. Get the syslog server configuration for All vShield App instances

Request:

```
GET https://<vsm-ip>/api/1.0/zones/syslogServers
```

You can configure all vShield App instances connected to the vShield Manager to send events to the specified syslog servers.

Example 6-17. Post the syslog server configuration across all vShield App instances

Request:

```
POST https://<vsm-ip>/api/1.0/zones/syslogServers
```

You can delete the syslog server configuration across all vShield App instances connected to the vShield Manager.

Example 6-18. Delete the syslog server configuration across all vShield App instances

Request:

```
DELETE https://<vsm-ip>/api/1.0/zones/syslogServers
```

You can delete a syslog server across all vShield App instances connected to the vShield Manager.

Example 6-19. Delete a single syslog server by IP address from All vShield App instances

Request:

```
DELETE https://<vsm-ip>/api/1.0/zones/syslogServers/<ip_of_syslogServer>
```

Upgrading vShield App

You can upgrade vShield App.

Example 6-20. Define namespace in a datacenter

Request:

```
POST https://<vsm-ip>/api/1.0/vshield/<host-id>/vsz
```

Request Body:

```
<VshieldConfiguration>
    <VszInstallParams>
        <DatastoreId>datastore-5131</DatastoreId>
        <ManagementPortSwitchId>network-5134</ManagementPortSwitchId>
        <MgmtInterface>
            <IpAddress>10.112.196.245</IpAddress>
            <NetworkMask>255.255.252.0</NetworkMask>
            <DefaultGw>10.112.199.253</DefaultGw>
        </MgmtInterface>
    </VszInstallParams>
    <InstallAction>upgrade</InstallAction>
</VshieldConfiguration>
```

vShield Endpoint Management 7

A vShield Endpoint appliance delivers an introspection-based antivirus solution that uses the hypervisor to scan guest virtual machines from the outside with only a thin agent on each guest virtual machine.

This chapter includes the following topics:

IMPORTANT All vShield REST requests require authorization. See "Using the vShield REST API" on page 12 for details about basic authorization.

Overview of Solution Registration

To register a third-party solution with vShield Endpoint, clients can use four REST calls to do the following:

- Register the vendor.
- Register one or more solutions.
- Set the solution IP address and port (for all hosts).
- Activate registered solutions per host.

To unregister a solution, clients essentially perform these steps in reverse:

- Deactivate solutions per host.
- Unset a solution's IP address and port.
- Unregister solutions.
- Unregister the vendor.

To update registration information for a vendor or solution, clients must first unregister that entity and then reregister. The following sections detail the specific REST calls to perform registration and unregistration.

Registering a Solution with vShield Endpoint Service

The APIs described in this section register a vendor, solutions, set network address, and activate solutions.

For a list of return status codes, see "Return Status Codes" on page 65.

Register a Vendor

You can register the vendor of an antivirus solution.

Example 7-1. Register a vendor

Request:

```
POST https://<vsm-ip>/api/2.0/endpointsecurity/registration
```

Request Body:

```
<VendorInfo>
    <id>vendor_id</id>
    <title>vendor_title</title>
    <description>vendor_description</description>
</VendorInfo>
```

In the request body, `vendor_id` is the VMware-assigned ID for the vendor, while `vendor_title` and `vendor_description` are vendor provided strings.

Register a Solution

You can register an antivirus solution.

Example 7-2. Register a solution

Request:

```
POST https://<vsm-ip>/api/2.0/endpointsecurity/registration/<vendor_id>
```

Request Body:

```
<SolutionInfo>
    <altitude>solution_altitude</altitude>
    <title>solution_title</title>
    <description>solution_description</description>
</SolutionInfo>
```

In the request, `<vendor_id>` is the previously registered ID for the vendor.

In the request body, `solution_altitude` is the VMware-assigned altitude for the solution, `solution_title` and `solution_description` are vendor provided strings. See "Altitude of a Solution" on page 62.

Altitude of a Solution

Altitude is a number that VMware assigns to solution as a filter on the security stack. The altitude describes the type of solution and the order in which the solution should receive file events. Table 7-1 shows some possible altitude assignments.

Table 7-1. Possible altitude assignments

Load-Order Groups	Group Altitude
Filter	31
Filter Top	30
Activity Monitor	29
Undelete	28
Anti-Virus	26
Replication	24
Backup	23
Content Screener	22
Quota Management	21
System Recovery	20

Table 7-1. Possible altitude assignments

Load-Order Groups	Group Altitude
Cluster File System	19
HSM	18
Imaging	17
Compression	16
Encryption	14
Virtualization	13
Physical Quota Management	12
Open File	10
Security Enhancer	8
Copy Protection	6
Filter Bottom	4
System	2

IP Address and Port for a Solution

You can set a solution's IP address and port on the vNIC host.

Example 7-3. Set IP address and port

Request:

```
POST https://<vsm-ip>/api/2.0/endpointsecurity/registration/<vendor_id>/<altitude>/location
```

Request Body:

```
<LocationInfo>
    <ip>solution_ip_address</ip>
    <port>solution_port</port>
</LocationInfo>
```

In the request, `<vendor_id>` is the previously registered ID for the vendor, and `<altitude>` for the altitude.

In the request body, `solution_ip_address` is the solution's IPv4 address for the vNIC that is connected to the VMkernel port group (for example, 192.168.8.2). This address must be within the range of VMware-assigned IP addresses for the solution. The `solution_port` is the port on which the solution accepts connections.

Activate a Solution

You can activate a solution that has been registered and located.

Example 7-4. Set IP address and port

Request:

```
POST https://<vsm-ip>/api/2.0/endpointsecurity/activation/<vendor_id>/<altitude>
```

Request Body:

```
<ActivationInfo>
    <moid>svm_moid</moid>
</ActivationInfo>
```

In the request, `<vendor_id>` is the previously registered ID for the vendor, and `<altitude>` for the altitude.

In the request body, `svm_moid` is the managed object ID of the activated solution's virtual machine.

Querying Registration Status of vShield Endpoint

You can use the same URIs shown in the previous section with the GET method to retrieve vendor registration information, solution registration information, location information, and solution activation status.

Get Vendor Registration

You can retrieve vendor registration information.

Example 7-5. Get vendor registration information

Request:

```
GET https://<vsm-ip>/api/2.0/endpointsecurity/registration/<vendor_id>
```

Get Solution Registration

You can retrieve solution registration information.

Example 7-6. Get solution registration information

Request:

```
GET https://<vsm-ip>/api/2.0/endpointsecurity/registration/<vendor_id>/<altitude>
```

Get IP Address of a Solution

This call retrieves the IP address and port associated with a solution.

Example 7-7. Get IP address and port of a solution

Request:

```
GET https://<vsm-ip>/api/2.0/endpointsecurity/registration/<vendor_id>/<altitude>/location
```

Get Activation Status of a Solution

This call retrieves solution activation status, given the managed object reference `<moid>` of its virtual machine.

Example 7-8. Get activation status of a solution

Request:

```
GET https://<vsm-ip>/api/2.0/endpointsecurity/activation/<vendor_id>/<altitude>/<moid>
```

Status can be `false` (not activated) or `true` (activated).

Unregistering a Solution with vShield Endpoint

You can use the same URIs shown in the first section with the DELETE method to unregister a vendor, unregister a solution, unset location information, or deactivate a solution.

Unregister a Vendor

This call unregisters a vendor.

Example 7-9. Unregister a vendor

Request:

```
DELETE https://<vsm-ip>/api/2.0/endpointsecurity/registration/<vendor_id>
```

Unregister a Solution

This call unregisters a solution.

Example 7-10. Unregister a vendor

Request:

```
DELETE https://<vsm-ip>/api/2.0/endpointsecurity/registration/<vendor_id>/<altitude>
```

Unset IP Address

This call unsets a solution's IP address and port.

Example 7-11. Unset IP address and port

Request:

```
DELETE https://<vsm-ip>/api/2.0/endpointsecurity/registration/<vendor_id>/<altitude>/location
```

Deactivate a Solution

This call deactivates a solution on a host.

Example 7-12. Deactivate a solution

Request:

```
DELETE https://<vsm-ip>/api/2.0/endpointsecurity/activation/<vendor_id>/<altitude>/<moid>
```

Status Codes and Error Schema

This section lists various status codes returned from the REST API, and shows the error schema.

Return Status Codes

The 200 codes indicate success, the 400 codes indicate some failure, and the 600 codes are call specific.

- 200 OK operation successful
- 201 Created: Entity successfully altered.
- 400 Bad Request: Internal error codes. Please refer to the Error Schema for more details.
- 401 Unauthorized: Incorrect user name or password.
- 600 Unrecognized vendor ID.
- 601 Vendor is already registered.
- 602 Unrecognized altitude.
- 603 Solution is already registered.
- 604 Invalid IPv4 address.
- 605 Invalid port.
- 606 Port out of range.

- 607 Unrecognized moid.
- 608 Location information is already set.
- 609 Location not set.
- 612 Solutions still registered.
- 613 Solution location information still set.
- 614 Solution still activated.
- 615 Solution not activated.
- 616 Solution is already activated.
- 617 IP:Port already in use.
- 620 Internal error.

Error Schema

Here is the XML schema for vShield Endpoint registration errors.

```
<?xml version="1.0" encoding="UTF-8"?><xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
                elementFormDefault="qualified">
<xs:element name="Error">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="code" type="xs:unsignedInt"/>
        <xs:element name="description" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
```

vShield Data Security Configuration 8

vShield Data Security provides visibility into sensitive data stored within your organization's virtualized and cloud environments. Based on the violations reported by vShield Data Security, you can ensure that sensitive data is adequately protected and assess compliance with regulations around the world.

This chapter includes the following topics:

To begin using vShield Data Security, you create a policy that defines the regulations that apply to data security in your organization and specifies the areas of your environment and files to be scanned. When you start a Data Security scan, vShield analyzes the data on the virtual machines in your vSphere inventory and reports the number of violations detected and the files that violated your policy.

After you analyze the results of the scan, you can edit your policy as required. When you edit a policy, you must enable it by publishing the changes.

Note that you cannot install vShield Data Security using a REST API. For information on installing vShield Data Security, see the *vShield Quick Start Guide*.

To deploy vShield Data Security, you must install the latest version of VMware Tools on each virtual machine that you want to scan. This installs a Thin Agent, which allows the SVM to scan the virtual machines.

vShield Data Security User Roles

A user's role determines the actions that the user can perform. A user can only have one role. You cannot add a role to a user, or remove an assigned role from a user, but you can change the assigned role for a user.

Table 8-1. vShield Data Security User Roles

Role	Actions Allowed
Enterprise administrator	All vShield operations and security.
vShield administrator	vShield operations only: for example, install virtual appliances, and configure port groups.
Security administrator	Create and publish policies, view violation reports. Cannot start or stop data security scans.
Auditor	View configured policies and violation reports. Read-only.

Defining a Data Security Policy

In order to detect sensitive data in your environment, you must create a data security policy. You must be a Security Administrator to create policies.

To define a policy, you must specify the following:

- Regulations

  A regulation is a data privacy law for protecting PCI (Payment Card Industry), PHI (Protected Health Information) and PII (Personally Identifiable Information) information. You can select the regulations that your company needs to comply to. When you run a scan, vShield Data Security identifies data that violates the regulations in your policy, and is hence sensitive for your organization.

- Excluded areas

  By default, all virtual machines in your data center are subject to sensitive data discovery. You can exclude specific areas of your environment from the data security scan if they are test environments or if you want to maintain sensitive data on them.

- File filters

  You can create filters to limit the data being scanned and exclude the file types unlikely to contain sensitive data from the scan.

In the data security APIs, `dlp` in the pathname stands for data loss prevention (DLP).

Retrieve All Regulations

You can retrieve the list of available regulations for a policy. The output includes regulation IDs and the embedded classifications for each regulation.

Example 8-1. Retrieve all SDD policy regulations

Request:

```
GET https://<vsm-ip>/api/2.0/dlp/regulation
```

Response:

```
<set>
    <Regulation>                    Regulation ID
        <id>66</id>
        <name>California AB-1298</name>
        <description>Identifies documents and transmissions that contain protected health
                     information (ePHI) and personally identifiable information (PII) as
                     regulated by California AB-1298 (Civil Code 56, 1785 and 1798)...
    <classifications>
        <Classification>                Classification ID
        <id>10</id>
        <name>Credit Card Track Data</name>
        <providerName>Credit Card Track Data</providerName>
        <description>Credit Card Track Data</description>
        <customizable>false</customizable>
        </Classification>
        ...
```

Enable a Regulation

You can enable one or more regulations by putting the regulation IDs into the policy. You can get the appropriate regulation IDs from the output of the retrieve regulations API (see Example 8-1). In the example request body, regulation 66 is California AB-1298, and regulations 67 and 68 originate elsewhere.

Example 8-2. Enable a regulation

Request:

```
PUT https://<vsm-ip>/api/2.0/dlp/policy/regulations
```

Request Body:

```
<?xml version="1.0" encoding="UTF-8"?>
<set>
    <long>66</long>
    <long>67</long>
    <long>68</long>
</set>
```

Retrieve the Classification Value

You can retrieve the classification values associated with regulations that monitor Group Insurance Numbers, Health Plan Beneficiary Numbers, Medical Record Numbers, or Patient Identification Numbers. The output includes the classification ID.

Example 8-3. Retrieve all classification values associated with customizable classifications

Request:

```
GET https://<vsm-ip>/api/2.0/dlp/classificationvalue
```

Configure a Customized Regex as a Classification Value

You can configure a `ClassificationValue` with a customized regex that must be matched during violation inspection. You must include the appropriate classification ID, which you can get from the output of the retrieve classification value API.

Example 8-4. Configure a customized regex as a classification value

Request:

```
PUT https://<vsm-ip>/api/2.0/dlp/policy/classificationvalues

Authorization: Basic YWRtaW46ZGVmYXVsdA==

<set>
    <ClassificationValue>
        <id>3</id>
        <classification>          ──► Classification ID
            <id>15</id>
            <name>Health Plan Beneficiary Numbers</name>
            <providerName>Health Plan Beneficiary Numbers</providerName>
            <description>Health Plan Beneficiary Numbers</description>
            <customizable>true</customizable>
        </classification>
        <value>PATNUM-[0-9]{10}</value>   ──► Regex
    </ClassificationValue>
</set>
```

View the List of Excludable Areas

You can retrieve the list of datacenters, clusters, and resource pools in your inventory to help you determine the areas you might want to exclude from policy inspection.

Example 8-5. View the list of excludable areas

Request:

```
GET https://<vsm-ip>/api/2.0/dlp/excludableareas
```

Response:

```
<set>
    <EnhancedInfo>
        <objectId>datacenter-2</objectId>
        <name>jdoe</name>
        <revision>32</revision>
        <objectTypeName>Datacenter</objectTypeName>
        <ownerName>VMware</ownerName>
    </EnhancedInfo>
    <EnhancedInfo>
        <objectId>datacenter-94</objectId>
        <name>jdoe</name>
        <revision>32</revision>
        <objectTypeName>Datacenter</objectTypeName>
        <ownerName>VMware</ownerName>
    </EnhancedInfo>
    <EnhancedInfo>
        <objectId>resgroup-3725</objectId>
        <name>ResourcePool1</name>
        <revision>2</revision>
        <objectTypeName>ResourcePool</objectTypeName>
        <ownerName>jdoe</ownerName>
    </EnhancedInfo>
    <EnhancedInfo>
        <objectId>domain-c2720</objectId>
        <name>Cluster1</name>
        <revision>17</revision>
        <objectTypeName>ClusterComputeResource</objectTypeName>
        <ownerName>jdoe</ownerName>
    </EnhancedInfo>
    <EnhancedInfo>
        <objectId>resgroup-3726</objectId>
        <name>ResourcePool2</name>
        <revision>1</revision>
        <objectTypeName>ResourcePool</objectTypeName>
        <ownerName>jdoe</ownerName>
    </EnhancedInfo>
</set>
```

Exclude Areas from Policy Inspection

You can exclude one or more datacenters, resource pools or clusters from policy inspection by including the object ID of each area to exclude. You can get the object ID from the output of the View the list of excludable areas API (see Example 8-5).

Example 8-6. Exclude areas from policy inspection

Request:

```
PUT https://<vsm-ip>/api/2.0/dlp/policy/excludedareas

Authorization: Basic YWRtaW46ZGVmYXVsdA==

<set>
    <string>datacenter-3720</string>
</set>
```

Configure File Filters

You can restrict the files you want to scan based on size, last modified date, or file extensions.

The following file filters are available:

- `sizeLessThanBytes` – scan only files with a byte size less than the specified number.

- lastModifiedBefore – scan only files modified before the specified date. The date must be specified in GMT format (YYYY-MM-DD HH:MM:SS).
- `lastModifiedAfter` – scan only files modified after the specified date. The date must be specified in GMT format (YYYY-MM-DD HH:MM:SS).
- `extensionsIncluded` – Boolean value as in Table 8-1.

Table 8-2. Included extensions parameter

Value of the extensionsIncluded parameter	Result
true followed by the extensions parameter containing one or more extensions	Only files with the specified extensions are scanned
false followed by the extensions parameter containing one or more extensions	All files are scanned except those with the specified extensions.

The `scanAllFiles` parameter determines if all files should be inspected during a scan operation. This parameter overrides all other parameters, so set this parameter to false if you are configuring a filter.

Example 8-7. Scan only PDF and XLXS files modified after 10/19/2011

Request:

```
PUT https://<vsm-ip>/api/2.0/dlp/policy/FileFilters
<FileFilters>
    <scanAllFiles>false</scanAllFiles>
    <lastModifiedAfter>2011-10-19 15:16:04.0 EST</lastModifiedAfter>
    <extensionsIncluded>true</extensionsIncluded>
    <extensions>pdf,xlsx</extensions>
</FileFilters>
```

Example 8-8. Scan all files except PDF and XLXS files

Request:

```
PUT https://<vsm-ip>/api/2.0/dlp/policy/FileFilters
<FileFilters>
    <scanAllFiles>false</scanAllFiles>
    <extensionsIncluded>false</extensionsIncluded>
    <extensions>pdf,xlsx</extensions>
</FileFilters>
```

Example 8-9. Scan PDF and XLXS files that are less than 100 MB in size

Request:

```
PUT https://<vsm-ip>/api/2.0/dlp/policy/FileFilters
<FileFilters>
    <scanAllFiles>false</scanAllFiles>
    <sizeLessThanBytes>100000000</sizeLessThanBytes>
    <extensionsIncluded>true</extensionsIncluded>
    <extensions>pdf,xlsx</extensions>
</FileFilters>
```

Saving and Publishing Policies

After you have defined a data security policy, you can edit it by changing the regulations selected, areas excluded from the scan, or the file filters. To apply the edited policy, you must publish it.

Retrieve the Saved SDD Policy

As a best practice, you should retrieve and review the last saved SDD policy before publishing it. Each policy contains a revision value that can be used to track version history.

Example 8-10. Retrieve the saved SDD policy

Request:

```
GET https://<vsm-ip>/api/2.0/dlp/policy/saved
Authorization: Basic YWRtaW46ZGVmYXVsdA==
```

Response: the following response contains a policy with a single regulation, Indiana HB-1101.

```
<DlpPolicy>
    <objectId>DlpPolicy-1</objectId>
    <type>
        <typeName>DlpPolicy</typeName>
    </type>
    <name>DlpPolicy-One</name>
    <revision>6</revision>
    <objectTypeName>DlpPolicy</objectTypeName>
    <regulations>
        <Regulation>
            <id>37</id>
            <name>Indiana HB-1101</name>
            <description>Indiana HB-1101</description>
            <classifications>
                <Classification>
                    <id>16</id>
                    <name>US National Provider Identifier</name>
                    <providerName>US National Provider Identifier</providerName>
                    <description>US National Provider Identifier</description>
                    <customizable>false</customizable>
                </Classification>
            <classifications>
            <regions>
                <string>North America</string>
                <string>USA</string>
            </regions>
            <categories>
                <string>PHI</string>
                <string>PCI</string>
                <string>PII</string>
            </categories>
        </Regulation>
    </regulations>
    <regulationsChanged>false</regulationsChanged>
    <excludedAreas/>
    <excludedAreasChanged>false</excludedAreasChanged>
    <fileFilters>
        <scanAllFiles>false</scanAllFiles>
        <sizeLessThanBytes>0</sizeLessThanBytes>
        <extensionsIncluded>false</extensionsIncluded>
    </fileFilters>
    <fileFiltersChanged>false</fileFiltersChanged>
    <classificationValues>
        <ClassificationValue>
            <id>1</id>
            <classification>
                <id>19</id>
                <name>Patient Identification Numbers</name>
                <providerName>Patient Identification Numbers</providerName>
                <description>Patient Identification Numbers</description>
                <customizable>true</customizable>
            </classification>
            <value>deg</value>
        </ClassificationValue>
    </classificationValues>
    <classificationValuesChanged>false</classificationValuesChanged>
</DlpPolicy>
```

Retrieve the Published SDD Policy

You can retrieve the currently published SDD policy that is active on all vShield Endpoint SVMs.

Example 8-11. Retrieve the published SDD policy

Request:

```
GET https://<vsm-ip>/api/2.0/dlp/policy/published
Authorization: Basic YWRtaW46ZGVmYXVsdA==
```

Publish the Updated Policy

After updating a policy with added regulations, excluded areas, or customized regex values publish the policy to enforce the new parameters.

Example 8-12. Publish the updated policy

Request:

```
PUT https://<vsm-ip>/api/2.0/dlp/policy/publish
```

Data Security Scanning

Running a data security scan identifies data in your virtual environment that violates your policy.

All virtual machines in your datacenter are scanned once during a scan. If the policy is edited and published while a scan is running, the scan restarts. This rescan ensures that all virtual machines comply with the edited policy. A rescan is triggered by publishing an edited policy, not by data updates on your virtual machines. After you start a scan, it continues to run until you pause or stop it.

If new virtual machines are added to your inventory while a scan is in progress, those machines will also be scanned. If a virtual machine is moved to an excluded cluster or resource pool while the data security scan is in progress, the files on that virtual machine are not scanned. In case a virtual machine is moved via vMotion to another host, the scan continues on the second host (files that were scanned while the virtual machine was on the previous host are not scanned again).

vShield Data Security scans one virtual machine on a host at a time to minimize impact on performance. VMware recommends that you pause the scan during normal business hours to avoid any performance overhead.

Retrieve the Status for a Scan Operation

You can retrieve the status of the scan operation to determine if a scan is `STARTED` (that is, in progress), `PAUSED`, or `STOPPED`. The `nextScanOps` parameter indicates the scan operations possible from your current state. In the following example, the current scan state is Stopped and the only action you can perform is Start the scan.

Example 8-13. Retrieve the status of a scan

Request:

```
GET https://<vsm-ip>/api/2.0/dlp/scanstatus
```

Response:

```
<DlpScanStatus>
    <currentScanState>STOPPED</currentScanState>
    <nextScanOps><ScanOp>START</ScanOp></nextScanOps>
    <vmsInProgress>0</vmsInProgress>
    <vmsCompleted>0</vmsCompleted>
```

```
</DlpScanStatus>
```

Start, Pause, Resume, or Stop a Scan Operation

You can start or stop a scan operation. The scan operation options are as follows:

- START: Start a new scan.
- PAUSE: Pause a started scan.
- RESUME: Resume a paused scan.
- STOP: Stop any scan.

Example 8-14. Start, pause, resume, or stop a scan operation

Request:

```
PUT https://<vsm-ip>/api/2.0/dlp/scanop

<ScanOp>STOP</ScanOp>
```

Analyzing Results

Once you start a data security scan, vShield reports the regulations that are being violated by the files in your inventory, and the violating files. If you fix a violating file (by deleting the sensitive information from the file, deleting or encrypting the file, or editing the policy), the file will continue to be displayed in the Violating files section until the current scan completes, and a new scan starts and completes.

You must be a Security Administrator or Auditor to view reports.

View the List of Violation Counts

You can view a report that displays the violated regulations with the number of violations for each regulation. The violating files report requires filtering by node ID.

Example 8-15. View the list of violated regulations

Request:

```
GET https://<vsm-ip>/api/2.0/dlp/violations/<context>
```

Where <context> is the context ID of a node (datacenter, portgroup, resource pool, or virtual machine, but not ESX host).

View the List of Violating Files

You can view a report that displays the violating files and the regulations each file violated. This API requires filtering by context node ID, and returns a formatted XML report showing violating files.

Example 8-16. View the list of violating files

Request:

```
GET https://<vsm-ip>/api/2.0/dlp/violatingfiles/<context>?pagesize=<i>&startindex=<j>
```

Where:

- <context> is the context ID of the node (datacenter, portgroup, resource pool, or virtual machine, but not ESX host).
- pagesize is the number of pages to view.

- `startindex` is the page number from which the results should be displayed.

View the List of Violating Files in CSV Format

You can view a report that displays the violating files and the regulations each file violated in a CSV format.

Example 8-17. View the list of violating files in CSV format

Request:

```
GET https://<vsm-ip>/api/2.0/dlp/violatingfilesascsv
```

View Violations in Entire Inventory

You can view a report of the violated regulations and the violating files for the entire inventory, in CSV (comma separated variable) format.

Example 8-18. View the list of violated regulations

Request:

```
GET https://<vsm-ip>/api/2.0/dlp/violatingfilescsv/<context>
```

Where `<context>` is the context ID of a node (datacenter, portgroup, resource pool, or virtual machine, but not ESX host).

Appendix

The REST API configuration of the vShield Edge and vShield App virtual machines supports schemas for installation and service management.

This appendix covers the following topics:

vShield Manager Global Configuration Schema

The following schema shows vShield Manager REST configuration.

This replaces the 1.0 API schema items for vCenter synchronization, DNS service, virtual machine information, and security groups.

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        targetNamespace="vmware.vshield.edge.2.0"
        xmlns:vse="vmware.vshield.edge.2.0"
        elementFormDefault="qualified">

   <xs:element name="vsmGlobalConfig">
      <xs:complexType>
         <xs:sequence>
            <xs:element minOccurs="0" name="vshieldEdgeReleaseInfo" type="vse:ReleaseInfoType"/>
              <!-- In response from server -->
            <xs:element minOccurs="0" name="vcInfo" type="vse:VcInfoType" />
            <xs:element minOccurs="0" name="hostInfo" type="vse:HostInfoType" />
            <xs:element minOccurs="0" name="techSupportLogsTarFilePath" type="xs:string"/>
            <xs:element minOccurs="0"  name="auditLogs" type="vse:AuditLogsType" />
            <xs:element minOccurs="0"  name="dnsInfo" type="vse:DnsInfoType" />
            <xs:element minOccurs="0"  name="versionInfo" type="xs:string" />     <!-- only in
              response -->
            <xs:element minOccurs="0"  name="vpnLicensed" type="xs:boolean" />    <!-- only in
              response -->
            <xs:element minOccurs="0"  name="ipsecVpnTunnels" type="vse:IpsecVpnTunnels" />
              <!-- only in response -->
            <xs:element minOccurs="0" maxOccurs="1" name="vsmCapability"
              type="vse:VsmCapabilityType"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:complexType name="ReleaseInfoType">                        <!-- can re-used this type -->
      <xs:sequence>
```

```
                <xs:element name="buildNumber" type="xs:NMTOKEN" />   <!-- add fields as required -->
                <xs:element minOccurs ="0" name="vseLocationOnVsm" type="xs:string" />
            </xs:sequence>
        </xs:complexType>

        <xs:complexType name="VcInfoType">
            <xs:sequence>
                  <xs:element name="ipAddress">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:minLength value="1"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="userName">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:minLength value="1"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="password">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:minLength value="1"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element minOccurs="0" name="certificateThumbprint">
                     <xs:simpleType>
                         <xs:restriction base="xs:string">
<xs:pattern value="[a-fA-F0-9]{2}:[a-fA-F0-9]{2}:[a-fA-F0-9]{2}:[a-fA-F0-9]{2}:[a-fA-F0-9]{2}:[a-fA-F0-9]
                {2}:[a-fA-F0-9]{2}:[a-fA-F0-9]{2}:[a-fA-F0-9]{2}:[a-fA-F0-9]{2}:[a-fA-F0-9]{2}:[a-fA-F0-9]
                {2}:[a-fA-F0-9]{2}:[a-fA-F0-9]{2}:[a-fA-F0-9]{2}:[a-fA-F0-9]{2}:[a-fA-F0-9]{2}:[a-fA-F0-9]
                {2}:[a-fA-F0-9]{2}:[a-fA-F0-9]{2}"></xs:pattern>
                         </xs:restriction>
                     </xs:simpleType>
                  </xs:element>
            </xs:sequence>
        </xs:complexType>

        <xs:complexType name="HostInfoType">
            <xs:sequence>
               <xs:element name="hostId" type="xs:string" />
               <xs:element name="ipAddress" type="xs:string" />
               <xs:element name="userName" type="xs:string" />
               <xs:element name="password" type="xs:string" />
            </xs:sequence>
        </xs:complexType>

        <xs:complexType name="IpList">
            <xs:sequence>
               <xs:element name="ip" type="xs:string" maxOccurs="unbounded" />
            </xs:sequence>
        </xs:complexType>

        <xs:complexType name="NodeList">
            <xs:sequence>
               <xs:element name="node" type="vse:SecurityGroupNode" maxOccurs="unbounded" />
            </xs:sequence>
        </xs:complexType>

        <xs:complexType name="SecurityGroupNode">
            <xs:sequence>
               <xs:element name="id" type="xs:string" />
               <xs:element name="name" type="xs:string" minOccurs="0" />
               <xs:element name="ipList" type="vse:IpList" minOccurs="0" />
            </xs:sequence>
```

```
</xs:complexType>

<xs:complexType name="VnicsType">
   <xs:sequence>
      <xs:element name="vnic" type="vse:VnicType" maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>

<xs:complexType name="VnicType">
   <xs:sequence>
      <xs:element name="id" type="xs:string" />
      <xs:element name="name" type="xs:string" />
      <xs:element name="ipList" type="vse:IpList" minOccurs="0" maxOccurs="1"/>
      <!--Will be good if we can also send this information
      <xs:element name="VLAN" type="xs:int" />
      <xs:element name="PortGroup" type="xs:string" />
      <xs:element name="Protected" type="xs:boolean"/> -->
   </xs:sequence>
</xs:complexType>

<xs:complexType name="AuditLogsType">
   <xs:sequence>
      <xs:element name="auditLog" type="vse:AuditLogType" maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
<xs:complexType name="DnsInfoType">
   <xs:sequence>
      <xs:element name="primaryDns" type="xs:string"/>
      <xs:element minOccurs="0" name="secondaryDns" type="xs:string"/>
      <xs:element minOccurs="0" name="tertiaryDns" type="xs:string"/>
   </xs:sequence>
</xs:complexType>

<xs:complexType name="AuditLogType">
   <xs:sequence>
      <xs:element name="id" type="xs:string" />
      <xs:element name="userName" type="xs:string" />
      <xs:element name="accessInterface" type="xs:string" />
      <xs:element name="module" type="xs:string" />
      <xs:element name="operation" type="xs:string" />
      <xs:element name="status" type="xs:string" />
      <xs:element name="operationSpan" type="xs:string" />
      <xs:element name="resource" type="xs:string" />
      <xs:element name="timestamp" type="xs:string" />
      <xs:element name="notes" type="xs:string" />
   </xs:sequence>
</xs:complexType>

<xs:complexType name="IpsecVpnTunnels">
   <xs:sequence>
      <xs:element name="lastEventId" type="xs:unsignedInt" />
      <xs:element minOccurs="0" maxOccurs="unbounded" name="ipsecVpnTunnelStatusList"
           type="vse:IpsecVpnTunnelStatus" />
   </xs:sequence>
</xs:complexType>

<xs:complexType name="IpsecVpnTunnelStatus">
   <xs:sequence>
      <xs:element name="networkId" type="xs:string" />
      <xs:element name="ipsecVpnTunnelConfig" type="vse:IpsecVpnTunnelConfigType" />
   </xs:sequence>
</xs:complexType>

<xs:complexType name="IpsecVpnTunnelConfigType">  <!--only in response -->
   <xs:sequence>
      <xs:element name="peerName">
         <xs:simpleType>
            <xs:restriction base="xs:string">
```

```
                    <xs:minLength value="1"/>
                    <xs:maxLength value="256"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:element>
        <xs:element name="peerId" type="xs:string" />
        <xs:element name="peerIpAddress" type="xs:string" />
        <xs:element maxOccurs="64" name="localSubnet" type="xs:string" />  <!-- localSubnet *
             peerSubnet * noOfSites should not be more than 64 -->
        <xs:element maxOccurs="64" name="peerSubnet" type="xs:string" /> <!-- localSubnet *
             peerSubnet * noOfSites should not be more than 64 -->
        <xs:element name="authenticationMode" >
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:pattern value="((psk)|(x.509))"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:element>
        <xs:element minOccurs="0" name="preSharedKey" type="xs:string" />
        <xs:element minOccurs="0" name="encryptionAlgorithm" type="xs:string" />
        <xs:element minOccurs="0" name="mtu" type="xs:unsignedInt" />
        <xs:element minOccurs="0" name="status" type="xs:string" />
        <xs:element minOccurs="0" name="stateChangeReason" type="xs:string" />
    </xs:sequence>
</xs:complexType>

    <xs:complexType name="VsmCapabilityType">
            <xs:sequence>
                    <xs:element name="ipsecVpnCapability" type="xs:boolean"/>
                    <xs:element name="webLoadBalancerCapability" type="xs:boolean"/>
                    <xs:element name="natCapability" type="xs:boolean"/>
                    <xs:element name="firewallCapability" type="xs:boolean"/>
                    <xs:element name="dhcpCapability" type="xs:boolean"/>
                    <xs:element name="staticRoutingCapability" type="xs:boolean"/>
                    <xs:element name="vsmVersion" type="xs:string"/>
            </xs:sequence>
    </xs:complexType>
</xs:schema>
```

ESX Host Preparation and Uninstallation Schema

This schema can be used to install or uninstall vShield App and vShield Endpoint services on an ESX host.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">

    <xs:element name="VshieldConfiguration">
        <xs:complexType>
            <xs:all>
                <xs:element minOccurs="0" name="VszInstallParams" type="VszInstallParams"/>
                <xs:element minOccurs="0" name="EpsecInstallParams" type="xs:boolean"/>
                <xs:element name="InstallAction" type="InstallAction"/>    <!-- InstallAction to
                              be taken on appliance - install/upgrade  -->
                <xs:element name="InstallStatus" type="InstallStatus"/>    <!-- only in response
                              -->
            </xs:all>
        </xs:complexType>
    </xs:element>

    <xs:complexType name="InstallStatus">
        <xs:sequence>
            <xs:element minOccurs="0" name="ProgressState" type="xs:string"/>
            <xs:element minOccurs="0" name="ProgressSubState" type="xs:string"/>
            <xs:element minOccurs="0" name="InstalledServices" type="InstalledServices"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="InstalledServices">
```

```
        <xs:sequence>
            <xs:element name="VszInstalled" type="xs:boolean"/>
            <xs:element name="EpsecInstalled" type="xs:boolean"/>
        </xs:sequence>
    </xs:complexType>

    <!-- Install parameters -->
    <xs:complexType name="VszInstallParams">
        <xs:sequence>
            <xs:element name="DatastoreId" type="Moid"/>
            <xs:element name="ManagementPortSwitchId" type="xs:string"/>    <!-- contains the
                        networkId of the mgmt portgroup -->
            <xs:element name="MgmtInterface" type="MgmtInterfaceType"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="MgmtInterfaceType">
        <xs:sequence>
            <xs:element name="IpAddress" type="IP"/>
            <xs:element name="NetworkMask" type="IP"/>
            <xs:element name="DefaultGw" type="IP"/>
        </xs:sequence>
    </xs:complexType>


    <xs:simpleType name="InstallAction">
        <xs:restriction base="xs:string">
            <xs:enumeration value="install"/>
            <xs:enumeration value="upgrade"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="IP">
        <xs:restriction base="xs:string">
            <xs:pattern value=
                        "((25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9]?[0-9])\.){3}(25[0-5]|2[0-4][
                        0-9]|1[0-9][0-9]|[1-9]?[0-9])"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="Moid">
        <xs:restriction base="xs:string">
            <xs:pattern value="[a-zA-Z0-9\-]+"/>
        </xs:restriction>
    </xs:simpleType>

</xs:schema>
```

vShield App Schemas

The following schemas detail vShield App configuration via REST API.

vShield App Configuration Schema

This schema configures a vShield App after installation.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">

    <xs:element name="ZonesConfiguration">
        <xs:complexType>
            <xs:all>
                <xs:element name="VszInstallParams" type="VszInstallParams" minOccurs="0"/>
            </xs:all>
        </xs:complexType>
    </xs:element>
```

```
    <!-- Install parameters -->
    <xs:complexType name="VszInstallParamsType">
        <xs:sequence>
            <xs:element name="NodeId" type="xs:string"/>
            <xs:element name="DatacenterId" type="xs:string"/>
            <xs:element name="DatastoreId" type="xs:string"/>
            <xs:element name="NameForZones" type="xs:string"/>
            <xs:element name="VswitchForMgmt" type="xs:string"/>
            <xs:element name="MgmtInterface" type="InterfaceType"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="InterfaceType">
        <xs:sequence>
            <xs:element name="IpAddress" type="xs:NMTOKEN"/>
            <xs:element name="NetworkMask" type="xs:NMTOKEN"/>
            <xs:element name="DefaultGw" type="xs:NMTOKEN"/>
            <xs:element minOccurs="0" name="VlanTag" type="xs:string"/>
        </xs:sequence>
    </xs:complexType>

</xs:schema>
```

vShield App Firewall Schema

This schema configures the firewall rules enforced by a vShield App.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" >

    <xs:element name="VshieldAppConfiguration">
        <xs:complexType>
            <xs:choice>
                <xs:element name="firewallConfiguration" type="FirewallConfigurationDto" />
                <xs:element name="firewallConfigurationHistoryList"
                               type="FirewallConfigHistoryInfoListDto" />
                <xs:element name="consolidatedConfiguration" type="FirewallConfigurationDto"
                               maxOccurs="unbounded" />
                <xs:element name="status" type="StatusDto" />
                <xs:element name="datacenterState" type="DatacenterStateDto" />
                <xs:element name="protocolsList" type="ProtocolListDto" />
                <xs:element name="protocolTypes" type="ProtocolsTypeEnum" maxOccurs="4" />
            </xs:choice>
        </xs:complexType>
    </xs:element>

    <xs:complexType name="ProtocolListDto">
        <xs:sequence>
            <xs:element name="protocol" type="ProtocolDto" maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="protocolsType" type="ProtocolsTypeEnum" />
        <xs:attribute name="applicableOnFirewallLayer" type="xs:string" use="optional" />
        <xs:attribute name="subProtocolOfTypeName" type="ProtocolsTypeEnum" use="optional" />
        <xs:attribute name="subProtocolOfTypeValue" type="xs:int" use="optional" />
    </xs:complexType>

    <xs:complexType name="ProtocolDto">
        <xs:sequence>
            <xs:element name="name" type="xs:string" />
            <xs:element name="value" type="xs:int" />
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="FirewallConfigHistoryInfoListDto">
        <xs:sequence>
            <xs:element name="contextId" type="xs:string" />
            <xs:element name="firewallConfigHistoryInfo" type="FirewallConfigHistoryInfoDto"
                           maxOccurs="unbounded" />
```

```
    </xs:sequence>
</xs:complexType>

<xs:complexType name="FirewallConfigHistoryInfoDto">
    <xs:sequence>
        <xs:element name="configId" type="xs:long" />
        <xs:element name="userId" type="xs:string" />
        <xs:element name="timestamp" type="xs:long" />
        <xs:element name="status" type="xs:string" minOccurs="0" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="DatacenterStateDto">
    <xs:sequence>
        <xs:element name="datacenterId" type="xs:string" />
        <xs:element name="userId" type="xs:string" minOccurs="0" />
        <xs:element name="timestamp" type="xs:long" minOccurs="0" />
        <xs:element name="status" type="DatacenterStatusEnum" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="StatusDto">
    <xs:sequence>
        <xs:element name="currentState" type="ConfigStateEnum" />
        <xs:element name="failedPublishInfo" type="FailedPublishInfoDto"
                    maxOccurs="unbounded" minOccurs="0" />
    </xs:sequence>
    <xs:attribute name="contextId" type="xs:string" use="required" />
    <xs:attribute name="generationNumber" type="xs:long" />
</xs:complexType>

<xs:complexType name="FailedPublishInfoDto">
    <xs:sequence>
        <xs:element name="applianceIp" type="xs:string" />
        <xs:element name="timestamp" type="xs:long" />
        <xs:element name="errorDescription" type="xs:string" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="FirewallConfigurationDto">
    <xs:sequence>
        <xs:element name="layer3FirewallRule" type="Layer3FirewallRuleDto"
                    maxOccurs="unbounded" minOccurs="0" />
        <xs:element name="layer2FirewallRule" type="Layer2FirewallRuleDto"
                    maxOccurs="unbounded" minOccurs="0" />
    </xs:sequence>
    <xs:attribute name="provisioned" type="xs:boolean" use="optional" />
    <xs:attribute name="contextId" type="xs:string" use="required" />
    <xs:attribute name="timestamp" type="xs:long" use="optional" />
    <xs:attribute name="generationNumber" type="xs:long" use="optional" />
</xs:complexType>

<xs:complexType name="ApplicationDto">
    <xs:choice>
        <xs:element name="applicationSetId" type="xs:string" />
        <xs:sequence>
            <xs:element name="portInfo" type="xs:string" minOccurs="0" />
            <xs:element name="protocol" type="xs:int" />
            <!-- Only in response, not considered in request -->
            <xs:element name="protocolName" type="xs:string" minOccurs="0" />
            <!-- only in case of ICMP -->
            <xs:element name="subType" type="xs:int" minOccurs="0" />
            <!-- Only in response, not considered in request -->
            <xs:element name="subTypeName" type="xs:string" minOccurs="0" />
        </xs:sequence>
    </xs:choice>
</xs:complexType>
```

```
<xs:complexType name="DestinationDto" abstract="true">
    <xs:sequence>
        <xs:element name="protocol" type="xs:int" minOccurs="0" />
        <xs:element name="address" type="AddressDto" minOccurs="0" />
        <!-- Only in response, not considered in request -->
        <xs:element name="protocolName" type="xs:string" minOccurs="0" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="Layer2DestinationDto">
    <xs:complexContent>
        <xs:extension base="DestinationDto">
        </xs:extension>
    </xs:complexContent>
</xs:complexType>

<xs:complexType name="Layer3DestinationDto">
    <xs:sequence>
        <xs:element name="address" type="AddressDto" minOccurs="0" />
        <xs:element name="application" type="ApplicationDto" minOccurs="0" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="Layer3SourceAddressDto">
    <xs:sequence>
        <xs:element name="address" type="AddressDto" minOccurs="0" />
        <xs:element name="portInfo" type="xs:string" minOccurs="0" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="FirewallRuleDto" abstract="true">
    <xs:sequence>
        <xs:element name="action" type="ActionEnum" />
        <xs:element name="logged" type="xs:boolean" />
        <xs:element name="notes" type="xs:string" minOccurs="0" />
    </xs:sequence>
    <xs:attribute name="id" type="xs:long" use="required" />
    <xs:attribute name="precedence" type="PrecedenceEnum" use="optional" />
    <xs:attribute name="disabled" type="xs:boolean" use="optional" />
</xs:complexType>

<xs:complexType name="Layer2FirewallRuleDto">
    <xs:complexContent>
        <xs:extension base="FirewallRuleDto">
            <xs:sequence>
                <xs:element name="source" type="AddressDto" minOccurs="0" />
                <xs:element name="destination" type="Layer2DestinationDto" />
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>


<xs:complexType name="Layer3FirewallRuleDto">
    <xs:complexContent>
        <xs:extension base="FirewallRuleDto">
            <xs:sequence>
                <xs:element name="source" type="Layer3SourceAddressDto" minOccurs="0" />
                <xs:element name="destination" type="Layer3DestinationDto" minOccurs="0" />
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>

<xs:complexType name="AddressDto">
    <xs:choice>
        <xs:element name="ipAddress" type="xs:string" />
        <xs:element name="macAddress" type="xs:string" />
```

```
            <xs:element name="containerId" type="xs:string">
            </xs:element>
        </xs:choice>
        <xs:attribute name="exclude" type="xs:boolean" use="optional" default="false" />
    </xs:complexType>


    <xs:simpleType name="ActionEnum">
        <xs:restriction base="xs:NCName">
            <xs:enumeration value="allow" />
            <xs:enumeration value="deny" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="PrecedenceEnum">
        <xs:restriction base="xs:NCName">
            <xs:enumeration value="high" />
            <xs:enumeration value="low" />
            <xs:enumeration value="default" />
            <xs:enumeration value="none" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="ConfigStateEnum">
        <xs:restriction base="xs:NCName">
            <!-- <xs:enumeration value="saved" /> -->
            <xs:enumeration value="published" />
            <xs:enumeration value="inprogress" />
            <xs:enumeration value="publishFailed" />
            <xs:enumeration value="Deleted" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="DatacenterStatusEnum">
        <xs:restriction base="xs:NCName">
            <xs:enumeration value="upgrading" />
            <xs:enumeration value="backwardCompatible" />
            <xs:enumeration value="backwardCompatibleReadyForSwitch" />
            <xs:enumeration value="migrating" />
            <xs:enumeration value="regular" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="ProtocolsTypeEnum">
        <xs:restriction base="xs:NCName">
            <xs:enumeration value="application" />
            <xs:enumeration value="ipv4" />
            <xs:enumeration value="icmp" />
            <xs:enumeration value="ethernet" />
        </xs:restriction>
    </xs:simpleType>

</xs:schema>
```

vShield App SpoofGuard Schema

The following schema details SpoofGuard configuration.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"elementFormDefault="qualified">

    <xs:element name="VshieldConfiguration">
        <xs:complexType>
            <xs:choice>
                <xs:element name="globalSettings" type="GlobalSettingsDto" />
                <xs:element name="ipAssignmentStatistic" type="IpAssignmentStatisticDto" />
                <xs:element name="vnicIdList" type="VnicIdListDto" />
                <xs:element name="ipAssignmentDetailsList" type="IpAssignmentDetailsListDto" />
```

```
            <xs:element name="pagedIpAssignmentDetailsList"
                        type="PagedIpAssignmentDetailsListDto" />
            <xs:element name="approveIpInfo" type="VnicInfoDto" />
        </xs:choice>
    </xs:complexType>
</xs:element>

<xs:complexType name="PagedIpAssignmentDetailsListDto">
    <xs:sequence>
        <xs:element name="ipAssignmentDetails" type="IpAssignmentDetailsDto"
                    maxOccurs="unbounded" />
        <xs:element name="pagingDetails" type="PagingInfoDto" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="PagingInfoDto">
    <xs:sequence>
        <xs:element name="pageSize" type="xs:int" />
        <xs:element name="startIndex" type="xs:int" />
        <xs:element name="totalCount" type="xs:int" />
        <xs:element name="sortOrderAscending" type="xs:boolean" />
        <xs:element name="sortBy" type="PagingSortByEnum" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="IpAssignmentDetailsListDto">
    <xs:sequence>
        <xs:element name="ipAssignmentDetails" type="IpAssignmentDetailsDto"
                    maxOccurs="unbounded" />
    </xs:sequence>
</xs:complexType>

    <xs:complexType name="IpAssignmentDetailsDto">
    <xs:sequence>
        <xs:element name="vnicId" type="xs:string" />
        <xs:element name="macAddress" type="xs:string" />
        <xs:element name="ipAddress" type="xs:string" />
        <xs:element name="vnicName" type="xs:string" />
        <xs:element name="networkId" type="xs:string" />
        <xs:element name="vmId" type="xs:string" />
        <xs:element name="vmName" type="xs:string" />
        <xs:element name="approvedIpAddress" type="xs:string" />
        <xs:element name="approvedBy" type="xs:string" />
        <xs:element name="approvedOn" type="xs:long" />
        <xs:element name="publishedIpAddress" type="xs:string" />
        <xs:element name="publishedBy" type="xs:string" />
        <xs:element name="publishedOn" type="xs:long" />
        <xs:element name="reviewRequired" type="xs:boolean" />
        <xs:element name="duplicateCount" type="xs:int" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="IpAssignmentStatisticDto">
    <xs:sequence>
        <xs:element name="contextId" type="xs:string" />
        <xs:element name="inSync" type="xs:boolean" />
        <xs:element name="activeCount" type="xs:long" />
        <xs:element name="inactiveCount" type="xs:long" />
        <xs:element name="activeSinceLastPublishedCount" type="xs:long" />
        <xs:element name="requireReviewCount" type="xs:long" />
        <xs:element name="duplicateCount" type="xs:long" />
        <xs:element name="unpublishedCount" type="xs:long" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="VnicIdListDto">
    <xs:sequence>
        <xs:element name="vnicId" type="xs:string" maxOccurs="unbounded" />
```

```
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="VnicInfoDto">
        <xs:sequence>
            <xs:element name="vnicId" type="xs:string" />
            <xs:element name="ipAddress" type="xs:string" />
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="GlobalSettingsDto">
        <xs:sequence>
            <xs:element name="status" type="OperationStatusEnum" />
            <xs:element name="mode" type="OperationModeEnum" />
            <!-- optional parameters will be part of response only -->
            <xs:element name="timestamp" type="xs:long" minOccurs="0" />
            <xs:element name="publishedBy" type="xs:string" minOccurs="0" />
        </xs:sequence>
    </xs:complexType>

    <xs:simpleType name="OperationStatusEnum">
        <xs:restriction base="xs:NCName">
            <xs:enumeration value="enabled" />
            <xs:enumeration value="disabled" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="OperationModeEnum">
        <xs:restriction base="xs:NCName">
            <xs:enumeration value="trustOnFirstUse" />
            <xs:enumeration value="manual" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="PagingSortByEnum">
        <xs:restriction base="xs:NCName">
            <xs:enumeration value="VM_NAME" />
            <xs:enumeration value="MAC" />
            <xs:enumeration value="APPROVED_IP" />
            <xs:enumeration value="CURRENT_IP" />
        </xs:restriction>
    </xs:simpleType>

</xs:schema>
```

vShield App Namespace Schema

The following schema details namespace configuration.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
               targetNamespace="vmware.vshield.global.20.namespace"
               xmlns:vsns="vmware.vshield.global.20.namespace" elementFormDefault="qualified">

    <xs:element name="VshieldConfiguration">
        <xs:complexType>
            <xs:choice>
                <xs:element maxOccurs="unbounded" name="namespace" type="vsns:NamespaceDto" />
                <xs:element maxOccurs="3" name="namespacesType" type="vsns:NamespacesTypeEnum" />
            </xs:choice>
        </xs:complexType>
    </xs:element>

    <xs:complexType name="NamespaceDto">
        <xs:sequence>
<xs:element minOccurs="0" maxOccurs="unbounded" name="namespacePortGroup"
               type="vsns:PortGroupDto" />
</xs:sequence>
```

```
<xs:attribute name="type" use="required" type="vsns:NamespacesTypeEnum" />
<xs:attribute name="id" use="optional" type="xs:long" />
</xs:complexType>

<xs:complexType name="PortGroupDto">
<xs:sequence>
<xs:element maxOccurs="1" name="Id" type="xs:string" />
</xs:sequence>
</xs:complexType>

<xs:simpleType name="NamespacesTypeEnum">
<xs:restriction base="xs:NCName">
<xs:enumeration value="DEFAULT" />
<xs:enumeration value="PORTGROUP" />
<xs:enumeration value="NONE" />
</xs:restriction>
</xs:simpleType>

</xs:schema>Retrieved from "https://wiki.eng.vmware.com/NS_DEV/vShieldManager/VSM30/App/ipad/xsd"
```

vShield Edge Schemas

The following schemas detail vShield Edge installation and configuration.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
   targetNamespace="vmware.vshield.edge.2.0"
   xmlns="vmware.vshield.edge.2.0"
   elementFormDefault="qualified">
   <xs:element name="vshieldEdgeConfig" >
      <xs:complexType>
         <xs:all>
            <xs:element minOccurs="0" name="installParams" type="InstallParams" />
            <xs:element minOccurs="0" name="applianceConfig" type="ApplianceConfig" />
            <xs:element minOccurs="0" name="routeConfig" type="RouteConfig" />
            <xs:element minOccurs="0" name="natConfig" type="NatConfig"/>
            <xs:element minOccurs="0" name="firewallConfig" type="FirewallConfig"/>
            <xs:element minOccurs="0" name="dhcpConfig" type="DhcpConfig"/>
            <xs:element minOccurs="0" name="dhcpService" type="ServiceStatus" />
            <xs:element minOccurs="0" name="loadbalancerConfig" type="LoadBalancerConfig"/>
            <xs:element minOccurs="0" name="loadbalancerService" type="ServiceStatus" />
            <xs:element minOccurs="0" name="ipsecSiteToSiteConfig"
              type="IpsecSiteToSiteConfig"/>
            <xs:element minOccurs="0" name="ipsecSiteToSiteService" type="ServiceStatus" />
            <xs:element minOccurs="0" name="syslogServerConfig" type="SyslogServerConfig"/>
            <xs:element minOccurs="0" name="certificateStoreConfig"
              type="CertificateStoreConfig"/>
            <xs:element minOccurs="0" name="cliLoginCredentials" type="CliLoginCredentials"/>
            <xs:element minOccurs="0" name="techSupportLocation" type="xs:string"/> <!-- Only
              for response -->
            <xs:element minOccurs="0" name="serviceStatsLocation" type="xs:string"/> <!-- Only
              for response -->
            <xs:element minOccurs="0" name="vseCapability" type="VseCapabilityList"/>    <!--
              Only for response -->
            <xs:element minOccurs="0" name="statistics" type="Statistics"/> <!-- Only for
              response -->
         </xs:all>
      </xs:complexType>
   </xs:element>

   <!-- In POST call to Install Edge  -->
   <!-- In GET call to show vshieldEdge configuration -->
   <xs:complexType name="InstallParams">
      <xs:sequence>
```

```
      <xs:element minOccurs="0" name="version"> <!-- Only in Response . Displays the vse
           appliance version -->
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:pattern value="((1.0)|(2.0))"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:element>
      <xs:element name="resourcePoolId" type="Moid" />
      <xs:element minOccurs="0" name="resourcePoolIdAtInstall" type="Moid" /> <!-- Only in
           Response. -->
      <xs:element name="hostId" type="Moid" />
      <xs:element minOccurs="0" name="hostIdAtInstall" type="Moid" /> <!-- Only in Response.
           -->
      <xs:element name="dataStoreId" type="Moid" />
      <xs:element minOccurs="0" name="dataStoreIdAtInstall" type="Moid" /> <!-- Only in
           Response. -->
      <xs:element minOccurs="0" name="vmId" type="Moid" /> <!-- Only in Response -->
      <xs:element minOccurs="0" name="vcUuid" type="xs:string" /> <!-- Only in Response -->
      <xs:element name="applianceConfig" type="ApplianceConfig"/>
      <xs:element minOccurs="0" name="currentStatus" type="xs:string" /> <!-- Only in
           Response -->
      <xs:element minOccurs="0" name="vmFolderId" type="Moid" />
      <xs:element minOccurs="0" name="vseName" type="xs:string" />
      <xs:element minOccurs="0" maxOccurs="unbounded" name="vmxParametersList"
           type="ConfigParameters" />
      <xs:element minOccurs="0" maxOccurs="unbounded" name="customField"
           type="ConfigParameters" />
      <xs:element minOccurs="0" name="memoryAllocation" type="ResourceAllocation" /> <!--
           Optional. When not specified, defaults are used -->
      <xs:element minOccurs="0" name="cpuAllocation" type="ResourceAllocation" /> <!--
           Optional. When not specified, defaults are used -->
   </xs:sequence>
</xs:complexType>

<xs:complexType name="ResourceAllocation">
   <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="limit" type="AllocationUnitTypes" /> <!--
           Optional. When not specified, defaults are used -->
      <xs:element minOccurs="0" maxOccurs="1" name="reservation" type="AllocationUnitTypes"
           /> <!-- Optional. When not specified, defaults are used -->
   </xs:sequence>
</xs:complexType>

<xs:complexType name="AllocationUnitTypes">
   <xs:choice>
      <xs:element minOccurs="0" name="value" type="xs:long" />
      <xs:element minOccurs="0" name="multiplier" type="xs:float" />
   </xs:choice>
</xs:complexType>

<xs:complexType name="ConfigParameters">
   <xs:sequence>
      <xs:element name="key" type="xs:string" />
      <xs:element name="value" type="xs:string" />
   </xs:sequence>
</xs:complexType>

<!-- To reconfigure Edge Appliance configurations like IP, subnet, defaultGw of interfaces -->
<xs:complexType name="ApplianceConfig">
   <xs:sequence>
      <xs:element minOccurs="0" name="hostName" type="Fqdn" />
      <xs:element minOccurs="0" name="tenantId">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:minLength value="1"/>
               <xs:maxLength value="250" />
            </xs:restriction>
```

```
            </xs:simpleType>
         </xs:element>
         <xs:element minOccurs="0" name="disableInternalFirewallRules" type="xs:boolean" /> <!--
              When not specified, default is false -->
         <xs:element maxOccurs="2" name="interface" type="Interface" />
      </xs:sequence>
   </xs:complexType>

   <xs:complexType name="Interface">
      <xs:sequence>
         <xs:element minOccurs="0" name="isUplink" type="xs:boolean" /> <!-- default is internal
              -->
         <xs:element name="networkId" type="Moid" />
         <xs:element name="ipAddress" type="Ip" />
         <xs:element name="subnetMask" type="Ip" />
         <xs:element minOccurs="0" name="macAddress" type="xs:string" />
         <xs:element minOccurs="0" name="defaultGw" type="Ip" /> <!-- Only for uplink interface
              -->
         <xs:element minOccurs="0" name="mtu" type="Mtu" />  <!-- default is 1500 -->
      </xs:sequence>
   </xs:complexType>

   <xs:complexType name="RouteConfig">
      <xs:sequence>
         <xs:element minOccurs="0" maxOccurs="2048" name="staticRoute"
              type="StaticRouteConfig"/>
      </xs:sequence>
   </xs:complexType>

   <xs:complexType name="StaticRouteConfig">
      <xs:sequence>
         <xs:element name="networkId" type="Moid" />
         <xs:element name="network" type="Cidr" />
         <xs:element name="nextHop" type="Ip" />
         <xs:element minOccurs="0" name="mtu" type="Mtu" /> <!-- default is that of the
              interface -->
      </xs:sequence>
   </xs:complexType>

   <xs:complexType name="NatConfig">        <!--  NATConfiguration   -->
      <xs:sequence>
         <xs:element minOccurs="0" maxOccurs="unbounded" name="rule" type="NatRule" />
      </xs:sequence>
   </xs:complexType>

   <xs:complexType name="NatRule">
      <xs:sequence>
         <xs:element minOccurs="0" name="state" type="StateOnVsm" /> <!-- Only in Response -->
         <xs:element minOccurs="0" name="ruleType" type="xs:string" />  <!-- Only in response.
              It will be used to tag the internal readOnly rules -->
         <xs:element name="type" >
            <xs:simpleType>
               <xs:restriction base="xs:string">
                  <xs:pattern value="snat|dnat"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:element>
         <xs:element minOccurs="0" name="protocol">
            <xs:simpleType>
               <xs:restriction base="xs:string">
                  <xs:pattern value="tcp|udp|icmp|any"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:element>
         <xs:element minOccurs="0" name="icmpType" type="IcmpType" />    <!-- Mandatory only
              when protocol=icmp -->
         <xs:element name="internalIpAddress" type="IpInfo" />
```

```
      <xs:element minOccurs="0" name="internalPort" type="PortInfo" /> <!-- port is valid
           only for protocol tcp|udp -->
      <xs:element name="externalIpAddress" type="IpInfo" />
      <xs:element minOccurs="0" name="externalPort" type="PortInfo" />  <!-- port is valid
           only for protocol tcp|udp -->
      <xs:element minOccurs="0" name="enableLog" type="xs:boolean" /> <!-- Not when present,
           default behavior is false -->
      <xs:element minOccurs="0" name="comments" type="xs:string" /> <!-- When present in
           response for an internalReadOnlyRule, it marks the service for which this rule is
           added -->
   </xs:sequence>
</xs:complexType>

<xs:complexType name="FirewallConfig">        <!--  FirewallConfiguration   -->
   <xs:sequence>
      <xs:element minOccurs="0" name="defaultPolicy">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:pattern value="allow|deny"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:element>
      <xs:element minOccurs="0" name="enableLoggingForDefaultPolicy" type="xs:boolean" />
           <!-- When not present, default behavior is false -->
      <xs:element minOccurs="0" name="blockIcmpErrors" type="xs:boolean" /> <!-- When not
           present, default behavior is false -->
      <xs:element minOccurs="0" maxOccurs="unbounded" name="rule" type="FirewallRule" />
   </xs:sequence>
</xs:complexType>

<xs:complexType name="FirewallRule">
   <xs:sequence>
      <xs:element minOccurs="0" name="state" type="StateOnVsm" /> <!-- Only in Response -->
      <xs:element minOccurs="0" name="ruleType" type="xs:string" />  <!-- Only in response.
           It will be used to tag the internal readOnly rules -->
      <xs:choice>
         <xs:element name="networkId" type="Moid" />
         <xs:element name="vpnInterface"/>
      </xs:choice>
      <xs:choice>
         <xs:sequence>
            <xs:element name="protocol" type="Protocol" />
            <xs:element minOccurs="0" name="icmpType" type="IcmpType" />    <!-- Mandatory
           only when protocol=icmp -->
            <xs:element minOccurs="0" name="destinationPort" type="PortInfo" />
         </xs:sequence>
         <xs:element name="applicationIdentifier" type="xs:string"/>
      </xs:choice>
      <xs:element name="destinationIpAddress" type="FwIpInfo" />
      <xs:element minOccurs="0" name="sourcePort" type="PortInfo" />
      <xs:element name="sourceIpAddress" type="FwIpInfo" />
      <xs:element name="direction">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:pattern value="in|out"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:element>
      <xs:element name="action">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:pattern value="allow|deny"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:element>
      <xs:element minOccurs="0" name="enableLog" type="xs:boolean" /> <!-- When not present,
           default behavior is false -->
```

```
      <xs:element minOccurs="0" name="disabled" type="xs:boolean" />  <!-- When not present,
           default behaviour is enabled -->
      <xs:element minOccurs="0" name="comments" type="xs:string" /> <!-- Only in Response.
           Marks the service for the InternalReadOnlyRule -->
   </xs:sequence>
</xs:complexType>

<xs:complexType name="FwIpInfo">
   <xs:choice>
      <xs:element name="ipsetIdentifier" type="xs:string"/>
      <xs:element name="ipAddress" type="IpInfo" />
   </xs:choice>
</xs:complexType>

<xs:complexType name="DhcpConfig">     <!--  DHCP Configuration -->
   <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="2048" name="binding" type="DhcpBinding" />
      <xs:element minOccurs="0" maxOccurs="2048" name="pool" type="DhcpPool" />
      <xs:element minOccurs="0" name="enableLog" type="xs:boolean" /> <!-- Not when present,
           default behavior is false -->
   </xs:sequence>
</xs:complexType>

<xs:complexType name="DhcpBinding">
   <xs:sequence>
      <xs:element name="vmId" type="Moid" />
      <xs:element name="interfaceId">
         <xs:simpleType>
            <xs:restriction base="xs:unsignedInt">
               <xs:minInclusive value="1"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:element>
      <xs:element name="hostName" type="Fqdn" />
      <xs:element name="internalIpAddress" type="Ip" />
      <xs:element minOccurs="0" name="configParams" type="DhcpConfigParams" />
   </xs:sequence>
</xs:complexType>

<xs:complexType name="DhcpPool">
   <xs:sequence>
      <xs:element name="ipRange" type="IpRange" />
      <xs:element minOccurs="0" name="configParams" type="DhcpConfigParams" />
   </xs:sequence>
</xs:complexType>

<xs:complexType name="DhcpConfigParams">
   <xs:sequence>
      <xs:element minOccurs="0" name="defaultGw" type="Ip" />  <!-- Default is the internal
           interface IP of the VSE -->
      <xs:element minOccurs="0" name="domainName" type="Fqdn" />
      <xs:element minOccurs="0" name="primaryNameServer" type="Ip" />
      <xs:element minOccurs="0" name="secondaryNameServer" type="Ip" />
      <xs:element minOccurs="0" name="leaseTime"> <!-- Default is 1 day -->
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:pattern value="(infinite|\d{2,}|[1-9])"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:element>
   </xs:sequence>
</xs:complexType>

<xs:complexType name="LoadBalancerConfig">  <!--  LoadBalancerConfig -->
   <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="8" name="listener" type="Listener" />
   </xs:sequence>
</xs:complexType>
```

```
<xs:complexType name="Listener">
   <xs:sequence>
      <xs:element name="externalIpAddress" type="Ip" />
      <xs:element maxOccurs="16" name="backEndServerConfig" type="LbIpInfo" />
      <xs:element minOccurs="0" name="algorithm">      <!--default is round-robin -->
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:pattern value="((round-robin)|(ip-hash))"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:element>
      <xs:element minOccurs="0" name="enableLog" type="xs:boolean" /> <!-- Not when present,
           default behavior is false -->
   </xs:sequence>
</xs:complexType>

<xs:complexType name="LbIpInfo" >
   <xs:sequence>
      <xs:element name="ipAddress" type="Ip" />
      <xs:element minOccurs="0" maxOccurs="1" name="port" type="Port" />
   </xs:sequence>
</xs:complexType>

<xs:complexType name="IpsecSiteToSiteConfig">
   <xs:sequence>
      <xs:element minOccurs="0" name="globalConfig" type="IpsecVpnGlobalConfig"/>
      <xs:element minOccurs="0" maxOccurs="64" name="siteConfig" type="IpsecVpnSiteConfig"/>
   </xs:sequence>
</xs:complexType>

<xs:complexType name="IpsecVpnGlobalConfig">
   <xs:sequence>
      <xs:element name="id" type="xs:string" />
      <xs:element minOccurs="0" name="certificateCn" type="Fqdn" /> <!--Optional, required
           for certficate mode authentication-->
      <xs:element minOccurs="0" name="ipAddress" type="Ip" />
      <xs:element minOccurs="0" name="preSharedKeyForDynamicIpSites" type="VpnPreSharedKey"
           /> <!--For all peers connecting from unknown IP (peerIPAddress == 'any')  -->
      <xs:element minOccurs="0" name="enableLog" type="xs:boolean" />   <!-- Not when
           present, default behavior is false -->
   </xs:sequence>
</xs:complexType>

<xs:simpleType name="VpnPreSharedKey">
   <xs:restriction base="xs:string">
      <xs:minLength value="1"/>
      <xs:maxLength value="128"/>
   </xs:restriction>
</xs:simpleType>

<xs:complexType name="IpsecVpnSiteConfig">
   <xs:sequence>
      <xs:element name="peerName">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:minLength value="1"/>
               <xs:maxLength value="256"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:element>
      <xs:element name="peerId" type="xs:string" />
      <xs:element name="peerIpAddress" type="IpOrAny" />
      <xs:element maxOccurs="64" name="localSubnet" type="Cidr" />  <!-- localSubnet *
           peerSubnet * noOfSites should not be more than 64 -->
      <xs:element maxOccurs="64" name="peerSubnet" type="Cidr" /> <!-- localSubnet *
           peerSubnet * noOfSites should not be more than 64 -->
      <xs:element minOccurs="0" name="authenticationMode" > <!-- Default is psk -->
```

```
            <xs:simpleType>
               <xs:restriction base="xs:string">
                  <xs:pattern value="((psk)|(x.509))"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:element>
         <xs:element minOccurs="0" name="preSharedKey" type="VpnPreSharedKey" /> <!-- Requried
              only when authenticationMode='psk' And peerIPAddress!='any' -->
         <xs:element name="encryptionAlgorithm" type="VpnEncryptionAlgo" />
         <xs:element minOccurs="0" name="enablePfs" type="xs:boolean" /> <!-- Default will be
              true -->
         <xs:element minOccurs="0" name="dhGroup" type="DhGroup" /> <!-- Default will be DH2 -->
         <xs:element minOccurs="0" name="mtu" type="Mtu" />  <!--Default is that of the uplink
              interface of the Appliance -->
         <xs:element minOccurs="0" name="stats" type="IpsecVpnSiteStats" /> <!-- Only in
              Response -->
      </xs:sequence>
   </xs:complexType>

   <xs:simpleType name="VpnEncryptionAlgo">
      <xs:restriction base="xs:string">
         <xs:pattern value="aes|aes256|3des"/>
      </xs:restriction>
   </xs:simpleType>

   <xs:simpleType name="DhGroup">
      <xs:restriction base="xs:string">
         <xs:pattern value="dh2|dh5"/>
      </xs:restriction>
   </xs:simpleType>

   <xs:complexType name="IpsecVpnSiteStats"> <!-- Only in Response -->
      <xs:sequence>
         <xs:element name="siteStatus" >
            <xs:simpleType>
               <xs:restriction base="xs:string">
                  <xs:pattern value="green|yellow|red"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:element>
         <xs:element name="txBytesFromSite" type="xs:unsignedLong"/>
         <xs:element name="rxBytesOnSite" type="xs:unsignedLong"/>
         <xs:element name="ikeStatus" type="IpsecVpnSiteIkeStatus"/>
         <xs:element name="tunnelStats" type="IpsecVpnTunnelStats" minOccurs="0"
              maxOccurs="unbounded"  />
      </xs:sequence>
   </xs:complexType>

   <xs:complexType name="IpsecVpnSiteIkeStatus">  <!-- Only in Response -->
      <xs:sequence>
         <xs:element name="channelStatus" type="ServiceStatus"/>
         <xs:element name="channelState" type="xs:string"/>
         <xs:element name="lastInformationalMessage" type="xs:string"/>
         <xs:element name="localIpAddress" type="xs:string"/>
         <xs:element name="peerId" type="xs:string"/>
         <xs:element name="remoteIpAddress" type="xs:string"/>
      </xs:sequence>
   </xs:complexType>

   <xs:complexType name="IpsecVpnTunnelStats">  <!-- Only in Response -->
      <xs:sequence>
         <xs:element name="tunnelStatus" type="ServiceStatus"/>
         <xs:element name="tunnelState" type="xs:string"/>
         <xs:element name="lastInformationalMessage" type="xs:string"/>
         <xs:element name="localSubnet" type="Cidr" />
         <xs:element name="peerSubnet" type="Cidr" />
         <xs:element name="encryptionAlgorithm" type="xs:string"/>
         <xs:element name="authenticationAlgorithm" type="xs:string" />
```

```
      <xs:element name="localSpi" type="xs:string" minOccurs="0" />
      <xs:element name="remoteSpi" type="xs:string" minOccurs="0" />
      <xs:element name="establishedDate" type="xs:string" />
      <xs:element name="txBytesFromLocalSubnet" type="xs:unsignedLong" />
      <xs:element name="rxBytesOnLocalSubnet" type="xs:unsignedLong" />
   </xs:sequence>
</xs:complexType>


<xs:complexType name="SyslogServerConfig">
   <xs:choice>
      <xs:element minOccurs="0" maxOccurs="2" name="ipAddress" type="Ip" />
   </xs:choice>
</xs:complexType>

<xs:complexType name="CliLoginCredentials">
   <xs:sequence>
      <xs:element name="username">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:minLength value="1"/>
               <xs:maxLength value="33" />
               <xs:pattern value="[a-z][a-z0-9_]*"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:element>
      <xs:element name="password">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:minLength value="1"/>
               <xs:pattern value="[^\s]+"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:element>
   </xs:sequence>
</xs:complexType>

<xs:complexType name="CertificateStoreConfig">
   <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="caCertificate" type="xs:string"/>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="certificate" type="xs:string"/>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="crl" type="xs:string"/>
      <!--Params for CSR generation-->
      <xs:element minOccurs="0" name="csrParams" type="CsrParams"/>
      <xs:element minOccurs="0" name="csr" type="xs:string" /> <!-- only response -->
   </xs:sequence>
</xs:complexType>

<xs:complexType name="CsrParams">
   <xs:sequence>
      <xs:element name="commonName" type="Fqdn" />
      <xs:element minOccurs="0" name="organization" type="xs:string" />
      <xs:element minOccurs="0" name="department" type="xs:string" />
      <xs:element minOccurs="0" name="city" type="xs:string"/>
      <xs:element minOccurs="0" name="state" type="xs:string"/>
      <xs:element minOccurs="0" name="country" type="xs:string"/>
      <xs:element name="keySize" type="KeySize"/>
   </xs:sequence>
</xs:complexType>

<xs:complexType name="Statistics">
   <xs:sequence>
      <xs:element minOccurs="0" name="interfaceStats" type="InterfaceStats"/>
      <xs:element minOccurs="0" name="trafficStats" type="TrafficStatistics"/>
      <xs:element minOccurs="0" maxOccurs="10" name="ipsecStats" type="IpsecVpnSiteStats"/>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="dhcpLeaseInfo"
           type="DhcpLeaseInfo"/>
   </xs:sequence>
```

```
</xs:complexType>

<xs:complexType name="InterfaceStats">
   <xs:sequence>
      <xs:element minOccurs="0" name="internalInterface" type="InterfaceStatsOnInterface"/>
      <xs:element minOccurs="0" name="externalInterface" type="InterfaceStatsOnInterface"/>
   </xs:sequence>
</xs:complexType>

<xs:complexType name="InterfaceStatsOnInterface">
   <xs:sequence>
      <xs:element minOccurs="0" name="rx" type="InterfaceStatsData"/>
      <xs:element minOccurs="0" name="tx" type="InterfaceStatsData"/>
   </xs:sequence>
</xs:complexType>

<xs:complexType name="InterfaceStatsData">
   <xs:sequence>
      <xs:element minOccurs="0" name="bytes" type="xs:unsignedLong"/>
      <xs:element minOccurs="0" name="packets" type="xs:unsignedLong"/>
      <xs:element minOccurs="0" name="carrier" type="xs:unsignedLong"/>
      <xs:element minOccurs="0" name="collisons" type="xs:unsignedLong"/>
      <xs:element minOccurs="0" name="dropped" type="xs:unsignedLong"/>
      <xs:element minOccurs="0" name="errors" type="xs:unsignedLong"/>
      <xs:element minOccurs="0" name="mcast" type="xs:unsignedLong"/>
      <xs:element minOccurs="0" name="overrun" type="xs:unsignedLong"/>
   </xs:sequence>
 </xs:complexType>

<xs:complexType name="TrafficStatistics">
   <xs:sequence>
      <xs:element minOccurs="0" name="internalInterface" type="TrafficStatsOnInterface"/>
           <!-- traffic from/to intif -->
      <xs:element minOccurs="0" name="externalInterface" type="TrafficStatsOnInterface"/>
           <!-- traffic from/to extif -->
      <xs:element minOccurs="0" name="vpnInterface" type="TrafficStatsOnInterface"/> <!--
           traffic from/to vpn remoteSubnets -->
      <xs:element minOccurs="0" name="vseTraffic" type="TrafficStatsOnInterface"/> <!--
           traffic processed at/generated from by VSE -->
      <xs:element minOccurs="0" name="droppedSessions" type="DroppedSessions"/> <!-- Drop
           counters due to user configured firewall -->
   </xs:sequence>
</xs:complexType>

<xs:complexType name="TrafficStatsOnInterface">
   <xs:sequence>
      <xs:element minOccurs="0" name="ingress" type="DirectionOnInterface"/> <!-- Incoming
           -->
      <xs:element minOccurs="0" name="egress" type="DirectionOnInterface"/> <!-- Outgoing -->
    </xs:sequence>
</xs:complexType>

<xs:complexType name="DirectionOnInterface">
   <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="addressCounting"
           type="TrafficStatsData"/> <!-- per address -->
      <xs:element minOccurs="0" maxOccurs="unbounded" name="networkCounting"
           type="TrafficStatsData"/> <!-- grouped by network -->
   </xs:sequence>
</xs:complexType>

<xs:complexType name="DroppedSessions">
   <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="ingressFirewallConfig"
           type="TrafficStatsData"/> <!-- Due to user configured ingress firewall rules -->
      <xs:element minOccurs="0" maxOccurs="unbounded" name="egressFirewallConfig"
           type="TrafficStatsData"/> <!-- Due to user configured egress firewall rules -->
```

```
      <xs:element minOccurs="0" maxOccurs="unbounded" name="ingressVpnFirewallConfig"
            type="TrafficStatsData"/> <!-- Due to user configured ingress firewall rules on
            vpn interface-->
      <xs:element minOccurs="0" maxOccurs="unbounded" name="egressVpnFirewallConfig"
            type="TrafficStatsData"/> <!-- Due to user configured egress firewall rules on vpn
            interface-->
      <xs:element minOccurs="0" maxOccurs="unbounded" name="ingressVseTraffic"
            type="TrafficStatsData"/> <!-- For packets targeting VSE -->
   </xs:sequence>
</xs:complexType>

<xs:complexType name="TrafficStatsData">
   <xs:sequence>
      <xs:element minOccurs="0" name="protocol" type="xs:string" />
      <xs:element minOccurs="0" name="sourceIp" type="xs:string" />
      <xs:element minOccurs="0" name="destinationIp" type="xs:string" />
      <xs:element minOccurs="0" name="packets" type="xs:unsignedLong" />
      <xs:element minOccurs="0" name="bytes" type="xs:unsignedLong" />
   </xs:sequence>
</xs:complexType>

<xs:complexType name="DhcpLeaseInfo">
   <xs:sequence>
      <xs:element minOccurs="0" name="uid" type="xs:string" />
      <xs:element minOccurs="0" name="macAddress" type="xs:string" />
      <xs:element minOccurs="0" name="ipAddress" type="xs:string" />
      <xs:element minOccurs="0" name="nextBindingState" type="xs:string" />
      <xs:element minOccurs="0" name="tsfp" type="xs:string" />
      <xs:element minOccurs="0" name="ends" type="xs:string" />
      <xs:element minOccurs="0" name="clientHostname" type="xs:string" />
      <xs:element minOccurs="0" name="tstp" type="xs:string" />
      <xs:element minOccurs="0" name="bindingState" type="xs:string" />
      <xs:element minOccurs="0" name="starts" type="xs:string" />
      <xs:element minOccurs="0" name="cltt" type="xs:string" />
      <xs:element minOccurs="0" name="abandoned" type="xs:string" />
      <xs:element minOccurs="0" name="hardwareType" type="xs:string" />
   </xs:sequence>
</xs:complexType>

<xs:simpleType name="KeySize">
   <xs:restriction base="xs:int">
      <xs:enumeration value="512"/>  <!-- Very less secure-->
      <xs:enumeration value="1024"/> <!-- Less secure-->
      <xs:enumeration value="2048"/> <!-- Very secure-->
   </xs:restriction>
</xs:simpleType>

<xs:complexType name="VseCapabilityList">
             <xs:sequence>
                    <xs:element minOccurs="0" maxOccurs="unbounded" name="vseCapabilityList"
            type="VseCapability"/>
              </xs:sequence>
     </xs:complexType>

     <xs:complexType name="VseCapability">
             <xs:sequence>
                    <xs:element name="ipsecVpnCapability" type="xs:boolean"/>
                    <xs:element name="webLoadBalancerCapability" type="xs:boolean"/>
                    <xs:element name="natCapability" type="xs:boolean"/>
                    <xs:element name="firewallCapability" type="xs:boolean"/>
                <xs:element name="dhcpCapability" type="xs:boolean"/>
                    <xs:element name="staticRoutingCapability" type="xs:boolean"/>
                    <xs:element name="networkId" type="xs:string"/>
                    <xs:element name="vseVersion" type="xs:string"/>
                    <xs:element name="compatibilityMode">
                           <xs:simpleType>
                                  <xs:restriction base="xs:string">
```

```
                                                <xs:pattern
            value="backwordCompatibilityMode|regularMode"/>
                                          </xs:restriction>
                                    </xs:simpleType>
                             </xs:element>
              </xs:sequence>
     </xs:complexType>

<xs:simpleType name="ServiceStatus">
   <xs:restriction base="xs:string">
      <xs:pattern value="up|down"/>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="Ip">
   <xs:restriction base="xs:string">
      <xs:pattern
            value="((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)\.){3}(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)"/
            >
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="IpOrAny">
   <xs:restriction base="xs:string">
      <xs:pattern
            value="(((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)\.){3}(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d))
            |(any)"/>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="Cidr">
   <xs:restriction base="xs:string">
      <xs:pattern
            value="(((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)\.){3}(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d))
            (\/)(3[0-2]|[1-2]\d|[1-9])"/>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="IpRange">
   <xs:restriction base="xs:string">
      <xs:pattern
            value="(((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)\.){3}(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d))
            (-)(((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)\.){3}(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d))"/>
            <!-- IP Range (IP1-IPn) -->
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="IpInfo">
   <xs:restriction base="xs:string">
      <xs:pattern value="any"/> <!-- any -->
      <xs:pattern
            value="(((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)\.){3}(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d))
            "/> <!-- IP -->
      <xs:pattern
            value="(((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)\.){3}(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d))
            (\/)(3[0-2]|[1-2]\d|[1-9])"/>  <!--CIDR -->
      <xs:pattern
            value="(((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)\.){3}(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d))
            (-)(((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)\.){3}(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d))"/>
            <!-- IP Range (IP1-IPn) -->
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="Port">
   <xs:restriction base="xs:string">
      <xs:pattern
            value="(6553[0-5]|655[0-2]\d|65[0-4]\d{2}|6[0-4]\d{3}|[1-5]\d{4}|[1-9]\d{1,3}|\d)"
            />
```

```
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="PortOrAny">
   <xs:restriction base="xs:string">
      <xs:pattern
            value="(6553[0-5]|655[0-2]\d|65[0-4]\d{2}|6[0-4]\d{3}|[1-5]\d{4}|[1-9]\d{1,3}|\d|(
            any))"/>
   </xs:restriction>
</xs:simpleType>

<xs:complexType name="PortRange">
   <xs:sequence>
      <xs:element name="rangeStart" type="Port" />
      <xs:element name="rangeEnd" type="Port" />
   </xs:sequence>
</xs:complexType>

<xs:simpleType name="PortInfo">
   <xs:restriction base="xs:string">
      <xs:pattern value="any"/> <!-- any -->
      <xs:pattern
            value="(6553[0-5]|655[0-2]\d|65[0-4]\d{2}|6[0-4]\d{3}|[1-5]\d{4}|[1-9]\d{1,3}|\d)"
            /> <!-- port -->
      <xs:pattern
            value="(6553[0-5]|655[0-2]\d|65[0-4]\d{2}|6[0-4]\d{3}|[1-5]\d{4}|[1-9]\d{1,3}|\d)(
            -)(6553[0-5]|655[0-2]\d|65[0-4]\d{2}|6[0-4]\d{3}|[1-5]\d{4}|[1-9]\d{1,3}|\d)"/>
            <!-- Port Range (Port1:Portn) -->
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="Protocol">
   <xs:restriction base="xs:string">
      <xs:pattern
            value="icmp|igmp|ipencap|tcp|udp|ipv6|ipv6-route|ipv6-frag|rsvp|gre|esp|ah|ipv6-ic
            mp|ipv6-nonxt|ipv6-opts|l2tp|sctp|ipcomp|any"/>
      <xs:pattern value="(2[0-5][0-5]|[0-1]\d{1,2}|\d{1,2}|\d)"/>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="IcmpType">
   <xs:restriction base="xs:string">
      <xs:enumeration value="echo-reply"/>
      <xs:enumeration value="destination-unreachable"/>
      <xs:enumeration value="source-quench"/>
      <xs:enumeration value="redirect"/>
      <xs:enumeration value="echo-request"/>
      <xs:enumeration value="router-advertisement"/>
      <xs:enumeration value="router-solicitation"/>
      <xs:enumeration value="time-exceeded"/>
      <xs:enumeration value="parameter-problem"/>
      <xs:enumeration value="timestamp-request"/>
      <xs:enumeration value="timestamp-reply"/>
      <xs:enumeration value="address-mask-request"/>
      <xs:enumeration value="address-mask-reply"/>
      <xs:enumeration value="any"/>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="Moid">
   <xs:restriction base="xs:string">
      <xs:pattern value="[a-zA-Z0-9_]+"/>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="Mtu">
   <xs:restriction base="xs:unsignedInt">
      <xs:minInclusive value="68"/>
```

```
         <xs:maxInclusive value="9000"/>
      </xs:restriction>
   </xs:simpleType>

   <xs:simpleType name="Fqdn">
      <xs:restriction base="xs:string">
         <xs:minLength value="1"/>
         <xs:maxLength value="256" />
         <xs:pattern
               value="[A-Za-z0-9][A-Za-z0-9_-]*(([.][A-Za-z0-9]+)|(([.][A-Za-z0-9_-]+)+[.][A-Za-z
               0-9]+))?"/>
      </xs:restriction>
   </xs:simpleType>

   <xs:simpleType name="StateOnVsm">
      <xs:restriction base="xs:string">
         <xs:enumeration value="out-of-sync"/>
      </xs:restriction>
   </xs:simpleType>
</xs:schema>
```

Error Message Schema

This schema details error messages.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">

   <xs:element name="Errors">
      <xs:complexType>
         <xs:sequence>
            <xs:element maxOccurs="unbounded" name="Error" type="ErrorType"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:complexType name="ErrorType">
      <xs:sequence>
         <xs:element name="code" type="xs:unsignedInt"/>
         <xs:element name="description" type="xs:string"/>
         <xs:element minOccurs="0" name="detailedDescription" type="xs:string"/>
         <xs:element minOccurs="0" name="index" type="xs:int"/>
         <xs:element minOccurs="0" name="resource" type="xs:NMTOKEN"/>
         <xs:element minOccurs="0" name="requestId" type="xs:NMTOKEN"/>
         <xs:element minOccurs="0" name="module" type="xs:NMTOKEN"/>
    </xs:sequence>
   </xs:complexType>

</xs:schema>
```

If a REST API call results in an error, the HTTP reply contains the following information.

- An XML error document as the response body
- Content-Type: application/xml
- An appropriate 2xx, 4xx, or 5xx HTTP status code

Table 9-1. Error Message Status Codes

Code	Description
`200 OK`	The request was valid and has been completed. Generally, this response is accompanied by a body document (XML).
`201 Created`	The request was completed and new resource was created. The `Location` header of the response contains the URI of newly created resource.
`204 No Content`	Same as `200 OK`, but the response body is empty (No XML).

Table 9-1. Error Message Status Codes

Code	Description
`400 Bad Request`	The request body contains an invalid representation or the representation of the entity is missing information. The response is accompanied by Error Object (XML).
`401 Unauthorized`	An authorization header was expected. Request with invalid or no vShield Manager Token.
`403 Forbidden`	The user does not have enough privileges to access the resource.
`404 Not Found`	The resource was not found. The response is accompanied by Error Object (XML).
`500 Internal Server Error`	Unexpected error with the server. The response is accompanied by Error Object (XML).
`503 Service Unavailable`	Cannot proceed with the request, because some of the services are unavailable. Example: vShield Edge is Unreachable. The response is accompanied by Error Object (XML).

Index