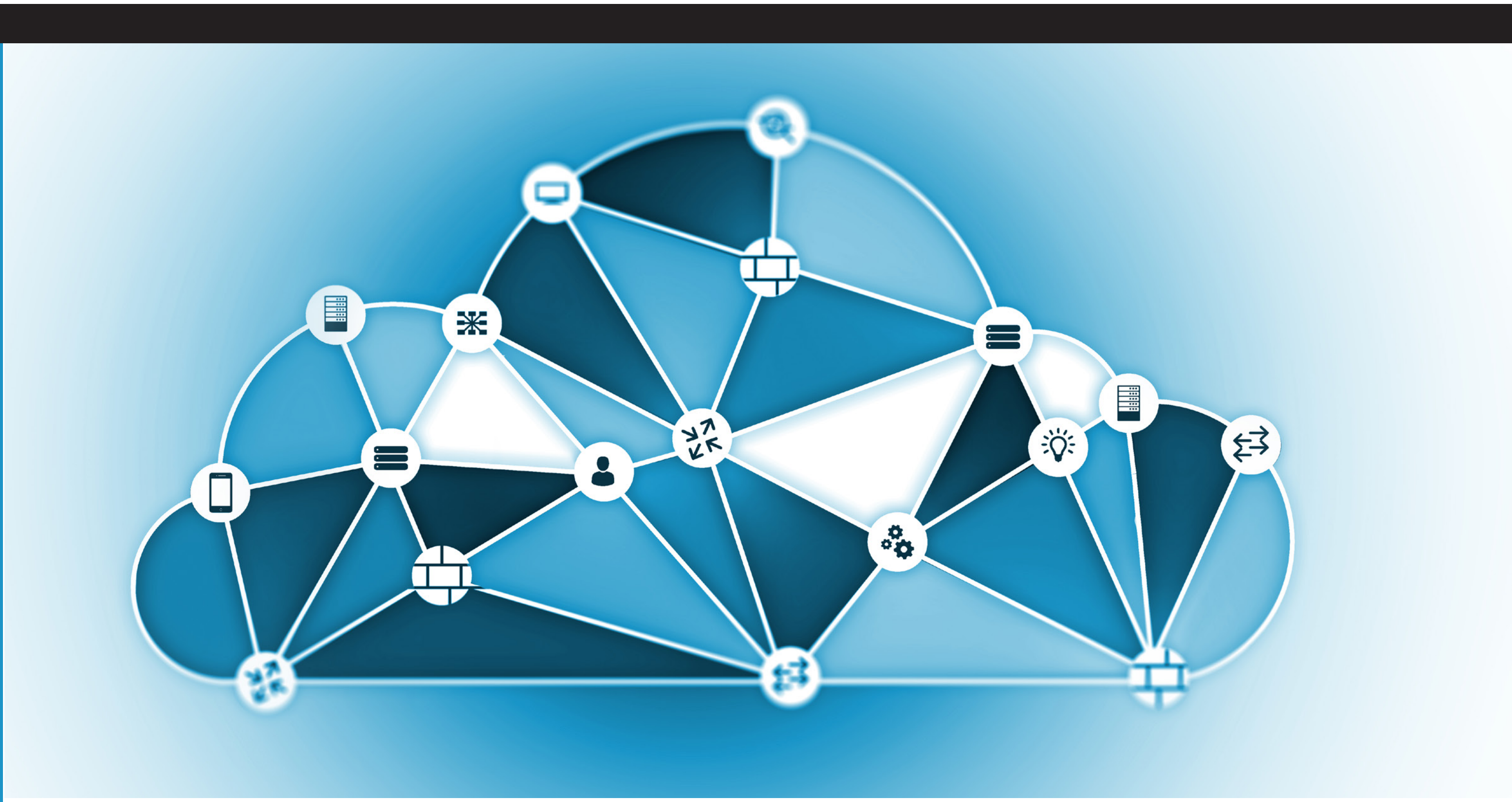


vMS Troubleshooting and Operations

Ana Perić - Carsten Koester - Christopher Clark
Doug Everett - Ivan Kovačević - Jyotsna Vemuri
Renuka Marichannappa - Riccardo Simoni
Richard Cunningham - Sameer Azad - Sunil Cherukuri

1. Introduction

1.1 Preface

This book represents a joint intense collaborative effort between Cisco’s Engineering, Technical Services and Advanced Services over a single week in the same room at Cisco Building 10 in Research Triangle Park, NC.

Authors

Ana Perić - Cisco Technical Services

Carsten Koester - Cisco Engineering

Christopher Clark - Cisco Technical Services

Doug Everett - Cisco Technical Services

Ivan Kovačević - Cisco Technical Services

Jyotsna Vemuri – Cisco Engineering

Renuka Marichannappa - Cisco Engineering

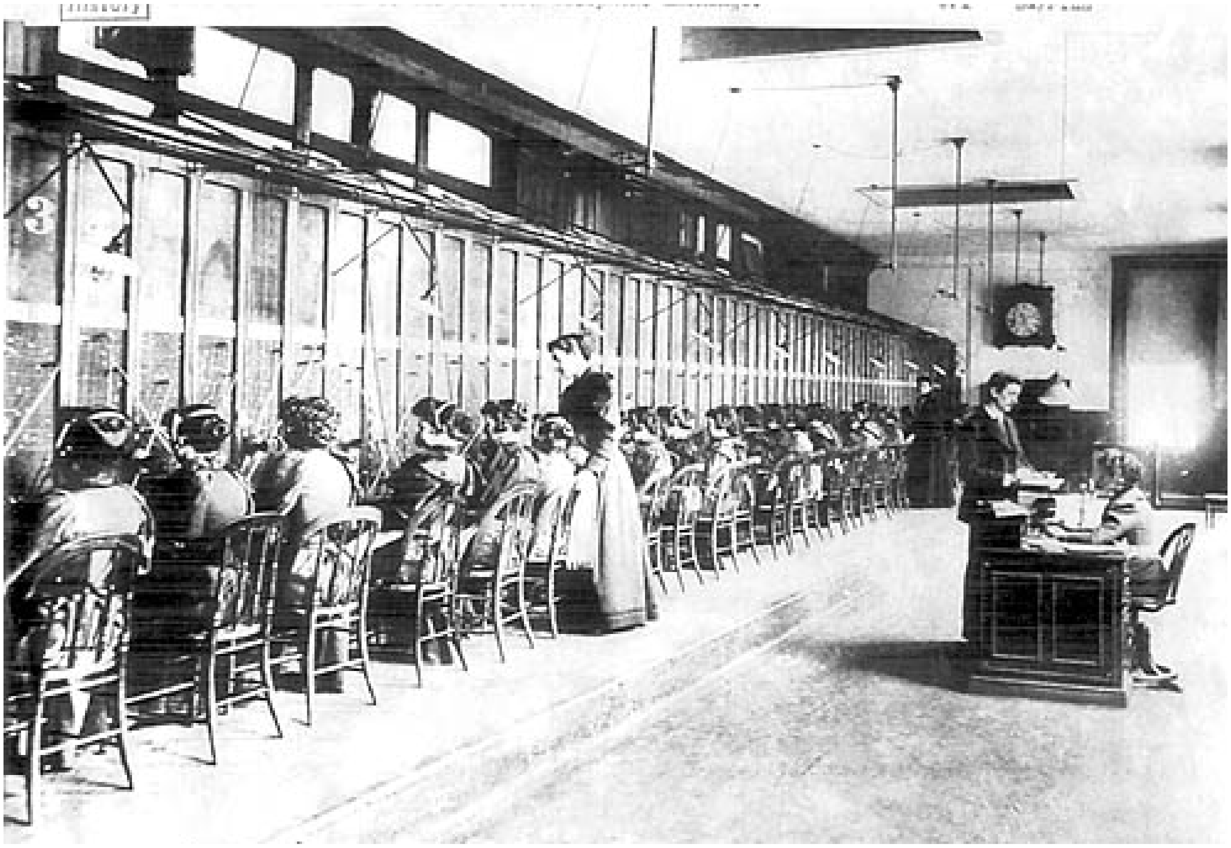
Riccardo Simoni – Cisco Technical Services

Richard Cunningham - Cisco Advanced Services

Sameer Azad – Cisco Engineering

Sunil Cherukuri – Cisco Engineering

1.2 Introduction to vMS Operations and Troubleshooting



Introduction

Networking today is a lot like the phone operators of the past. Back then, when a customer wanted to make a call, an operator was necessary to patch through the call connection to the other end to complete the call. This process was slow and often prone to errors by the operator. Imagine rooms full of people patching through calls all day long. This was very tedious work indeed. Why did all of those operators disappear? Long story short, the automated phone switch was invented. The phone switch automated the call connections, revolutionizing the telephone industry.

Hold that thought, and now think about how ordering and delivering networking services happen today. This is a very long, drawn-out process for both the end customer and the Service Provider. A customer submits a purchase order to the Service Provider for a new service. That purchase order is routed to the billing department for set up of a new account. The account team submits an order to the engineering team to design the new service. The engineering team sends over the design to the network operations team to implement the newly requested service during a maintenance window. The Service Provider's Network Operations Center (equivalent of an operator pool) configures the routers and switches necessary for service enablement. The Service Provider sends a field engineer to the customer location to install the customer premise equipment to complete the order. This process takes anywhere from 4- to 6-weeks today. How many people are involved in this process? And why does it take so long? It seems very inefficient, correct? The solution to this is Service Orchestration - an automated way of instantiating, deploying and configuring services.

Cisco has been working diligently with Service Providers to solve this problem for the networking industry. With Cisco's Virtual Managed Services (vMS) solution, Service Providers are able to deliver a flexible ITaaS solution that benefits both end customers (SMBs and Enterprises) and Service Providers. Cisco vMS is a Network Functions Virtualization (NFV) orchestration platform that enables fast deployment of cloud-based networking services. SMBs and Enterprises are delivered a customizable network service solution in a subscription-based, pay-as-you-go model. Service Providers benefit from a solution that allows them to reduce costs associated with service fulfillment, faster service deployment times, and quicker time to repair. Due to the efficiency gained via service automation, the service turn-up time is reduced from 4-6 weeks to a matter of a few days.

Expected Audience

This book is intended for folks needing to understand how to operate and/or troubleshoot a vMS deployment. The information captured in this guide can be used by both support engineers and implementation engineers as a reference guide for the vMS Solution. It can also be used as a vMS operational manual by Service Providers.

Organization of this Book

vMS Architecture

This chapter provides an introduction to the basic concepts of the vMS solution, its main components and the way they interact with each other to operate as an end-to-end solution.

vMS Use-Cases

The vMS Use-Cases chapter provides an overview of the different service topologies that can currently be orchestrated by the vMS orchestration stack.

Portal, Identity Management and Service Assurance

This chapter describes the end customer web user interface for the vMS solution, its main components and how customers should use it. Also provided is an overview of the vMS Service-Assurance stack. It also describes some basic issue resolution procedures.

Network Services Orchestrator (NSO)

This chapter details the architecture, configuration, operational aspects and troubleshooting of the NSO.

Elastic Services Controller (ESC)

This chapter explains what the role of the ESC is in the solution, its architecture, operations and issues resolution.

Customer Premise Equipment (CPE)

The CPE chapter discusses the details surrounding the vMS solution components located at the end customer premises.

vMS Operations & Administration

This chapter describes solution-level operational aspects and administration procedures.

OpenStack Troubleshooting

This chapter provides an overview of issue-resolution for OpenStack. The vMS solution rides on top of an OpenStack-based virtual infrastructure.

Service Chain Troubleshooting

This chapter provides insight into how to deal with data-plane issues and calls out some of the most common problem scenarios customers might encounter while deploying services using the vMS solution.

Book Writing Methodology

The Book Sprint (www.booksprints.net) methodology was used for writing this book. The Book Sprint methodology is an innovative new style of cooperative and collaborative authorship. Book Sprints are strongly facilitated and leverage team-oriented inspiration and motivation to rapidly deliver large amounts of well-authored and reviewed content, and incorporate it into a complete narrative in a short amount of time. By leveraging the input of many experts, the complete book was written in a short time period of only five days, however involved hundreds of authoring person-hours, and included thousands of experienced engineering hours, allowing for extremely high quality in a very short production time period.

1.3 Acronyms

AAA - Authentication Authorization and Accounting
API - Application Programming Interface
ASA - Cisco Adaptive Security Firewall Virtual Appliance
BSS - Business Support Systems
CAPEX - Capital Expenditure
CIS - Cisco Intercloud Services
CLI - Command Line Interface
CPE - Customer Premise Equipment
CRUD - Create, Read, Update, Delete Operations
CSP - Cloud Service Provider
CSR - Cisco Cloud Service Router Virtual Appliance
DC - Data Center
DEA - Droplet Execution Agent
DNS - Domain Name System
ESC - Cisco Elastic Services Controller
HTTP - Hypertext Transfer Protocol
HTTPS - Hypertext Transfer Protocol Secure
IDM - Identity Management
IETF - Internet Engineering Task Force
IOS - Internetwork Operating System
IPsec - Internet Protocol Security
IT - Information Technology
LDAP - Lightweight Directory Access Protocol
MANO - ETSI NFV Management and Orchestration
MTU - Maximum Transmission Unit
NETCONF - Network Configuration Protocol
NFV - Network Functions Virtualization
NSO - Network Service Orchestrator
OPEX - Operational Expenditure
OSS - Operating Support Systems
PaaS - Platform as a Service
PDU - Protocol Data Unit
PMTUD - Path MTU Discovery
PnP - Plug and Play
QoS - Quality of Service
REST - Representational State Transfer
SA - Service Assurance
SMB - Small and Medium Business
SME - Service Metrics Engine
SNMP - Simple Network Management Protocol

SP - Service Provider
SSH - Secure Shell
SSL - Secure Socket Layer
SSO - Single Sign-On
UCS - Cisco Unified Compute System
URL - Uniform Resource Identifier
VIM - Virtualization Infrastructure Manager
VIP - Virtual IP
vMS - Virtual Managed Services
VNF - Virtual Network Function
VNFM - Virtual Network Function Manager
VPN - Virtual Private Network
WSA - Cisco Web Security Appliance
ZTD - Zero Touch Deployment

2. vMS Architecture

2.1 Introduction

This chapter describes the architecture, key components and workflows of the Cisco vMS Solution. It's divided into three sections covering the following details:

- vMS Overview - introduces the Cisco vMS Solution.
- vMS Architecture and Components - describes the solution architecture and provides an overview of the main orchestration components of the solution.
- vMS Call Flows - describes inter-component vMS call flows to enable a better understanding of vMS operations and the relations between individual solution components.

2.2 vMS Overview

Cisco Virtual Managed Services (vMS) is an open software platform that enables Service Providers to utilize Network Functions Virtualization (NFV) and offer their customers a flexible selection of VPN connectivity and security applications that are easily customized through a self-service portal. It reduces the costs for service creation, customer acquisition, service fulfillment, time to repair, and maintenance. The Cisco Virtual Managed Service (vMS) solution automates the delivery of cloud business services, shortens the time to revenue and reduce capital expenditure (CAPEX) and operational expenditure (OPEX) for the Service Provider, along with providing enhanced user experience for their customers. It leverages the market trends, rapidly reducing costs of x86 server hardware and availability of Open Source software like OpenStack. This trend moves away from dedicated purpose-built hardware to reusable compute platforms that can be re-purposed as market demands change.

With Cisco vMS, Service Providers can offer their customers cloud-based managed services, and they can use a self-service portal to obtain cloud-based VPN and Security services in minutes, instead of waiting for weeks or months. End customers can easily scale up or scale down the services as needed and avoid the capital expenditure by consuming these software capabilities as a service, reduce their IT CAPEX and OPEX, and always have access to the latest security technologies and services offered by the Service Provider in their service catalog.

The vMS platform and solution can orchestrate both physical and virtual network devices/functions, and can automate end-to-end provisioning for different use-cases and service topologies. Each release of the vMS solution will provide out-of-box capabilities to orchestrate particular use-cases, also called Service Level packages. The vMS Service Level Packages are a suite of pre-packaged software capabilities that fully automate the end-to-end service creation including ordering, service chaining, orchestration, service assurance, and all the necessary virtualized network functions (VNFs) on the Cisco vMS platform. With these fully validated service level packages, end customers can quickly turn on, control, and assure cloud-based managed services offered by the Service Provider.

vMS CloudVPN Service

The vMS 2.0 and 2.1 releases (current vMS versions as of this book writing) offer cloud-based IPsec VPN, Firewall, and Web Security services for customer premise equipment (CPEs) located in customer branch sites. This service level package is called CloudVPN, and has the following sub-packages:

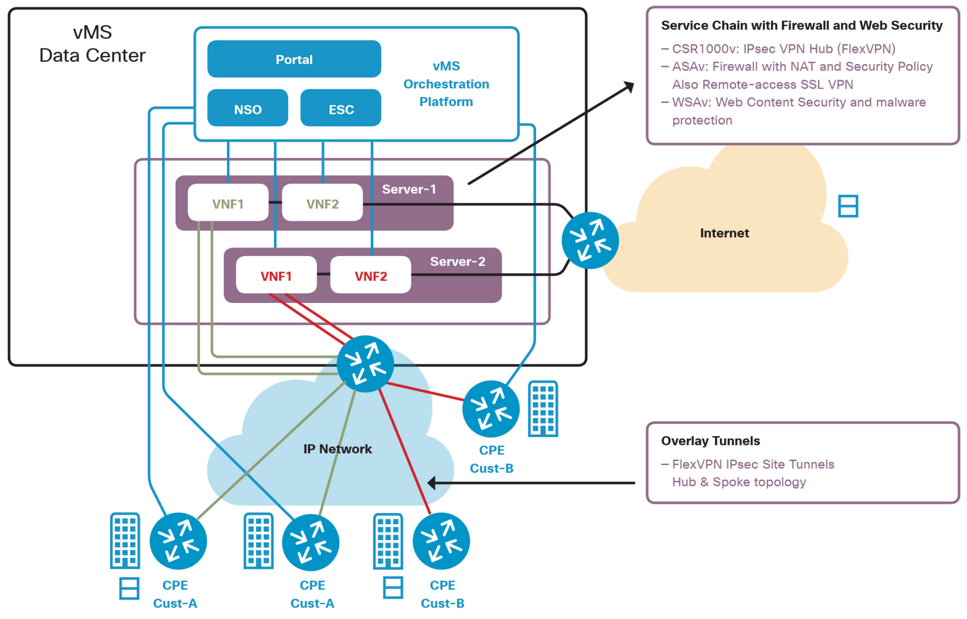
- **CloudVPN Foundation**: Designed to be an entry-level service that enables business customers to connect multiple enterprise sites securely, while managing Internet access locally on their premises. An enterprise can unify their corporate network by interconnecting sites via this hosted CloudVPN service. All customer premises equipment (CPE) based sites can securely communicate with each other over the Internet, through a hosted virtual router in Hub and Spoke topology with encrypted IPsec tunnels. The Cisco Cloud Services Router (CSR1000v) is utilized for the virtual router and IPsec hub role.

- **CloudVPN Advanced**: In addition to enabling business customers to connect multiple sites securely, this package provides a firewall service for secure access to the Internet. All traffic traversing the Hub and Spoke CloudVPN topology will be secured through encrypted IPsec tunnels. There are two optional services that may be offered via the CloudVPN Advanced Service Level Package: Remote-access Secure Socket Layer (SSL) VPN for mobile users, and web content security with advanced malware protection and real-time malware scanning capabilities. The Cisco Cloud Services Router (CSR1000v) is utilized for the virtual router and IPsec hub role. The Cisco virtual Adaptive Security Appliance (ASAv) is utilized to provide Internet firewall and SSL-VPN functionality. The Cisco virtual Web Security Appliance (WSAv) is utilized to provide web content security and malware protection functionality.

> **Note**: When using the vMS portal, the above two service packages appear as three service selection options - Basic service (maps to Foundation package), Medium service (maps to Advanced package), and Full service (maps to Advanced package with Web Security add-on). The Service Provider deploying vMS may choose to customize the portal and market the service differently (for eg. Essential, Enhanced and Full). For the remainder of this book, the CloudVPN services described above will be referred to as Basic, Medium and Full, and the Portal screenshots will show these as Essential, Enhanced and Premium.

The following figure illustrates the use-case that is enabled by the vMS solution CloudVPN service package.

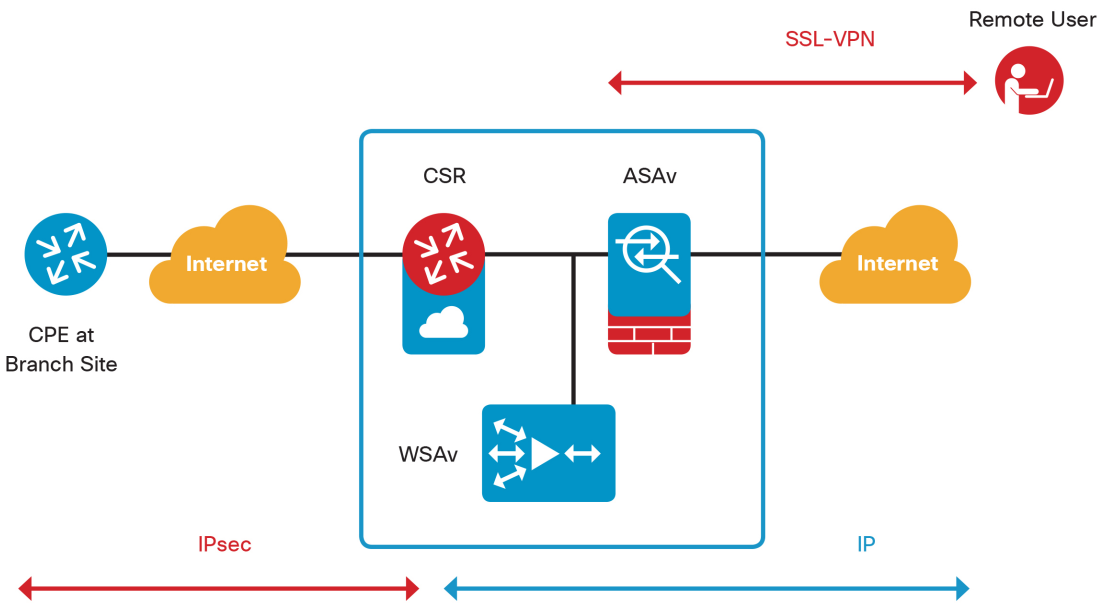
Figure 2.1 vMS CloudVPN Components


vMS
Data Center
Portal
NSO
ESC
vMS
Orchestration
Platform
VNF1
VNF2
Server-1
VNF1
VNF2
Server-2
Service Chain with Firewall and Web Security
– CSR1000v: IPsec VPN Hub (FlexVPN)
– ASAv: Firewall with NAT and Security Policy
Also Remote-access SSL VPN
– WSAv: Web Content Security and malware
protection
Internet
IP Network
CPE
Cust-B
Overlay Tunnels
– FlexVPN IPsec Site Tunnels
Hub & Spoke topology
CPE
Cust-A
CPE
Cust-A
CPE
Cust-B

vMS CloudVPN Service Chain

Before a deeper discussion of the vMS architecture and components is begun, it would be good to visualize the end-to-end logical service topology provided by the vMS CloudVPN service. The cloud-based virtual network functions that are chained together to provide managed security services for the end customers and sites, are grouped together into a "Service Chain". While more details on the service chain will be provided in subsequent chapters, the following figure illustrates the CloudVPN Advanced service chain and the end-to-end service topology that can be orchestrated by the vMS solution currently.

Figure 2.2 vMS CloudVPN Service Chain

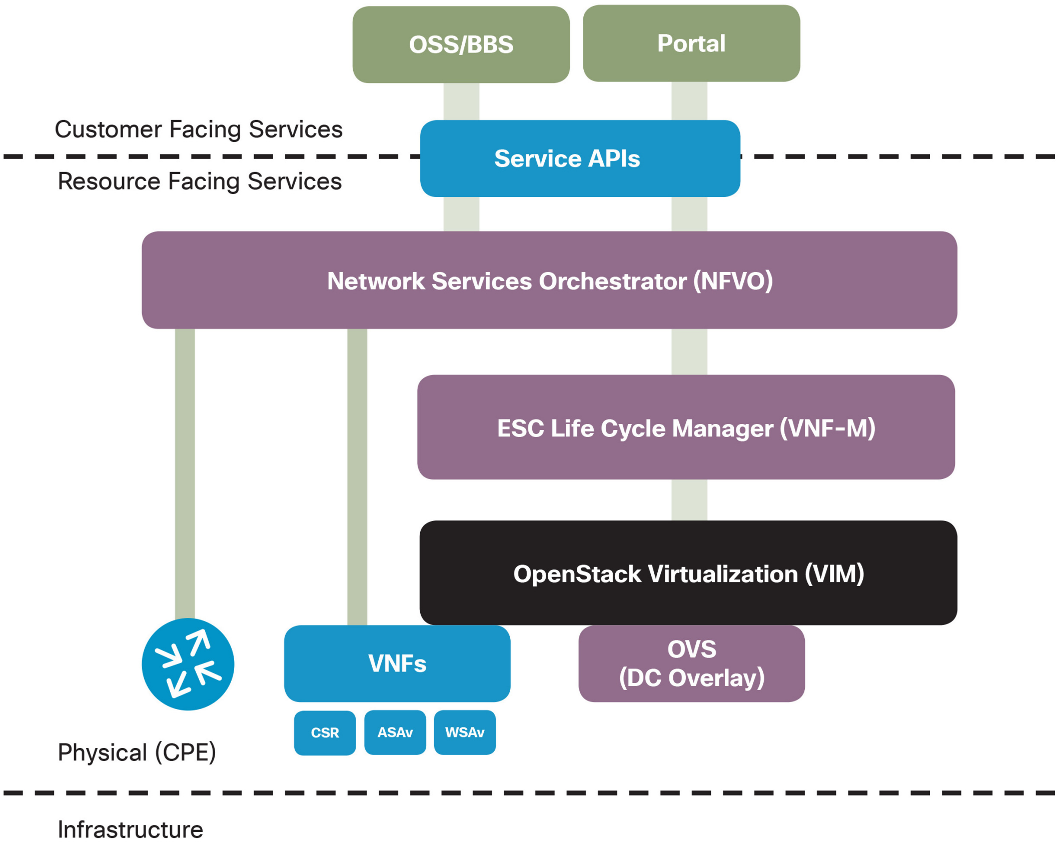


2.3 vMS Architecture and Components

vMS Architecture

The vMS solution for orchestrating cloud-based Network Functions Virtualization (NFV) use-cases is comprised of the Orchestration platform, self-service Portal, Virtual Network Function (VNF) appliances, customer premise equipment (CPE), and Virtualization Infrastructure to host the VNFs. The Orchestration is comprised of service orchestration (for both physical and virtual devices) and VNF life cycle management. The vMS architecture is aligned to the ETSI NFV Orchestration and Management (MANO) reference architecture. The following figure illustrates the vMS architecture. In this figure, the key vMS management components are mapped to the ETSI-MANO terminology such as NFV Orchestrator (NFVO), VNF Manager (VNF-M) and Virtualization Infrastructure Manager (VIM)

Figure 2.3 vMS Architecture



vMS Components & Roles

The vMS solution architecture depicted in Figure 2.3 is comprised of the following key components:

vMS Portal

The vMS Portal comprises an end customer-facing self-service portal to create, order, customize, and manage a cloud service offering; and an operator-facing portal with dashboard application for vMS operations. The vMS Portal provides different options when ordering a cloud service, and subsequent management of the deployed cloud service. The Portal also provides APIs for integration with the operator's OSS/BSS systems. The vMS Portal sub-system is comprised of several containerized services built using a micro-services framework.

Network Services Orchestrator (NSO)

The vMS service orchestration is provided by the Cisco Network Services Orchestrator (NSO) which automates the dynamic provisioning of physical, virtual and software assets required to create the end-to-end service. Cisco NSO is a model-driven orchestrator and uses YANG for modelling the services, and can use NETCONF, SSH, REST etc to provision the devices. In the vMS solution, NSO also functions as the Plug-N-Play (PnP) server, which is utilized for zero-touch deployment (ZTD) of the CPE devices. NSO runs as a Virtual Machine (VM) in the vMS solution (multiple NSO VMs for HA or Clustering).

Elastic Services Controller (ESC)

The Virtual Network Function Manager (VNFM) role in the vMS solution is performed by the Cisco Elastic Services Controller (ESC). The Cisco ESC automates the VNF lifecycle management, including VNF instantiation, initial Day 0 configuration, and VNF monitoring. ESC can currently instantiate VNFs on an OpenStack/KVM infrastructure. Future versions of ESC will also support other virtual infrastructures such as VMware's vSphere. ESC can instantiate Cisco VNFs and also VNFs from other vendors. ESC runs as a Virtual Machine (VM) in the vMS solution (two ESC VMs for HA).

Virtualization Infrastructure (OpenStack)

In the vMS solution, the current virtualization infrastructure is the KVM hypervisor, and the virtualization infrastructure manager (VIM) is OpenStack. While the vMS solution can work with different OpenStack versions and distributions, the vMS 2.0 solution has been validated with Canonical OpenStack Icehouse release, running on Cisco Unified Compute System (UCS) servers.

Virtual Network Functions (VNF)

The vMS 2.0/2.1 solution with the Cloud VPN service package utilizes the following VNFs:

- Cisco Cloud Services Router (CSR1000v) - the CSR1000v is utilized as a virtual Router and IPsec-VPN Hub (FlexVPN Hub) in the cloud.
- Cisco virtual Adaptive Security Appliance (ASAv) - the ASAv is utilized as a virtual Internet Firewall

and SSL-VPN Hub (AnyConnect) in the cloud.

- Cisco virtual Web Security Appliance (WSAv) - the WSAv is utilized as a virtual Web Content (URL) filter and anti-Malware protector in the cloud.

Service Assurance

The vMS 2.0/2.1 solution has some base Service Assurance functionality, where the status and statistics of the service chain VNFs and CPEs are monitored and displayed in the portal. The Cisco Service Metrics Engine (SME) is utilized as the statistics collector, and SME gets the status (status of device, interface, IPsec tunnel etc) and statistics (interface packet counts, tunnel packet counts, throughput etc) via NSO - which sends SNMP queries to the CPE and VNFs to obtain the SNMP statistics. Future releases of the vMS solution will include log collectors and an analytics engine to provide optional and enhanced service assurance capabilities.

Other Components

There are other components within the vMS solution which are not shown in the architecture diagram in Figure 2.3 "vMS Architecture" above. These include:

Identity Management (IDM)

The Portal interacts with an Identity Management component for user management and authentication. The IDM sub-system comprises of several VM's.

Management Hub

A CSR 1000v is utilized as the IPsec Hub for terminating Management FlexVPN Tunnels from the CPEs. The CSR 1000v hub is common for all tenants (end customers), and the CPEs belonging to all tenants (end customers) bring up the Management IPsec FlexVPN tunnel to this hub, after completing the Plug-n-Play (PnP) procedure and obtaining the Day 0 configuration. The Management tunnels are utilized for NSO to securely communicate with the CPE devices for Day 1 and Day 2 provisioning, and also for SNMP monitoring.

License Proxy

A majority of the VNFs today require software licenses to fully function. Many of the Cisco VNFs (and even physical routers) are transitioning to utilize centralized licensing functionality, whereby the devices call home to Cisco Smart Licensing to authenticate and obtain the required licenses. Out of the three VNFs utilized in the vMS Cloud VPN use-case, the CSR1000v and ASAv support this functionality today. Both the CSR1000v and ASAv call-out from their management interfaces, via the Internet, to obtain licenses from the Cisco Cloud Licensing service. However, in the Cloud VPN use-case, the CSR1000v and ASAv management interfaces are provisioned with private RFC1918 addresses, so they cannot connect to the Internet directly. To enable the licensing, the vMS system installs a HTTP/HTTPS proxy in the management stack, which allows the VNFs to reach the Smart Licensing servers. While any off-the-shelf proxy server can be utilized with this, in the vMS installation and validation, Tiny Proxy has been utilized. Tiny Proxy is a Linux HTTP/HTTPS proxy server daemon, and this is enabled in the Bootstrap server in the vMS solution.

DNS Server

A DNS server is required within the vMS management stack to enable Fully Qualified Domain Name (FQDN) based communication between the various vMS components. In addition, when a new service chain and the constituent VNFs are created, their hostname-to-IP mapping is updated in this DNS server. NSO updates the DNS server once the VNFs have been created. While any DNS server can be utilized for this purpose, the vMS system has been installed and validated with Bind9 which is a Linux DNS daemon. The Bind9 daemon is enabled on the Bootstrap server.

Device Configurations

The orchestration call flows that create the vMS Service Chains use the concept of Day -1 (minus 1), Day 0, Day 1 and Day 2 configurations when referring to the stages of device configurations (E.g. VNFs and CPEs) and the various stages of service instantiation. A brief description of each configuration is given below.

- **Day -1** - Contains the minimal configuration necessary to enable a new CPE to contact the NSO PnP server including SSL certificates.
- **Day 0** - Contains the minimal configuration necessary to boot a new VNF and enable SSH access via the management network interface. The CPE Day 0 configuration is the minimal configuration required to establish an IPsec management tunnel (Tunnel0) back the Management Hub so that NSO can access the device via SSH.
- **Day 1** - Refers to the configuration of all devices in an initial unmodified Service Chain after at least one CPE is on-boarded.
- **Day 2** - Refers to configuration modifications made to a fully deployed Service Chain in order to customize parameters such as QoS settings and Web filtering levels which are configured via the portal.

vMS Deployment Models

The vMS solution has two deployment models currently.

On-Premise Deployment

In this deployment model, the vMS solution is deployed in the Service Provider's Data Center (DC). The Service Provider (SP) installs and manages OpenStack and the vMS software components on top of it. The SP then uses their vMS deployment to provide cloud-based managed services (like CloudVPN) to their end customers (SMBs and Enterprises).

Typically this means the Service Provider has full access to both the hardware and software components (servers, OpenStack, vMS management components etc) and can create and manage the OpenStack quotas and can add or delete servers. Typically this also means that the vMS management stack components (ESC) has administrative privileges into OpenStack, and ESC can create OpenStack Nova flavors and upload VNF images into OpenStack Glance. From the perspective of ESC's relationship with OpenStack, this can be referred as In-band Image and Flavor management.

Hosted Deployment

In this deployment model, the vMS solution is deployed by a Cloud Service Provider (CSP) in their Data Center, and offered as-a-Service to other Service Providers (SP). The CSP installs and manages OpenStack and the vMS software components on top of it. The Service Provider (typically telcos) utilizes this vMS-as-a-Service provided by the CSP to provide vMS services (such as CloudVPN) to their end customers. An example of a CSP providing vMS-as-a-Service currently is Cisco Intercloud Services (CIS).

Typically this means the CSP has full access to both the hardware and software components (servers and OpenStack), and can create and manage the OpenStack quotas. The SP only has tenant privileges in OpenStack, but has full access into the vMS components. Typically this also means that the vMS management stack components (ESC) only has tenant privileges into OpenStack, and ESC cannot create OpenStack Nova flavors. The OpenStack quota management and Nova flavor management are done directly by the CSP, however the Service Provider can manage images in Glance. From the perspective of ESC's relationship with OpenStack, this can be referred as Out-band Image and Flavor management.

Note: The descriptions of On-Premise and Hosted deployment models, while being typical scenarios, are not restrictive. There could be other scenarios and combinations of privilege/access models possible, especially in the context of the ESC In-band or Out-Band Image and Flavor management relationship with OpenStack.

2.4 vMS Call Flows

As the previous section explained, the vMS solution consists of many orchestration components that communicate with each other using different protocols. Detailed communication flows between components will be explained to show how a successful vMS solution operates.

The subsections below explain:

- vMS Initialization Call Flows
- vMS Service Chain-related Call Flows: call flows that describe communication between the vMS solution components related to operations over Service Chains (such as adding/deleting, upgrading and downgrading of a Service Chain, and modification of existing Service Chain settings)
- vMS CPE Call Flows: describes call flows of the process bringing (on-boarding) a new CPE to the existing Service Chain, deleting / removing (off-boarding) a CPE from the existing Service Chain, and replacing CPE devices.

Call Flows - Initialization Flows

ESC Registration Flow

When the vMS solution is deployed for the first time, the ESC orchestration component needs to be registered and appropriate images have to be registered in OpenStack.

Please note that the ESC registration process happens only once, and this section describes it in order to have a complete overview of the Call Flows.

A detailed call flow of ESC registration is shown in Figure 2.4.

1. NSO starts registration of Tenants for each Provider configured:

- NSO sends Create Tenant Request to ESC
- ESC forwards Create Tenant Request to OpenStack
- OpenStack responds with OK upon Tenant creation
- ESC sends Notification: Create Tenant Success message back to NSO.

At this point, Tenants are created successfully.

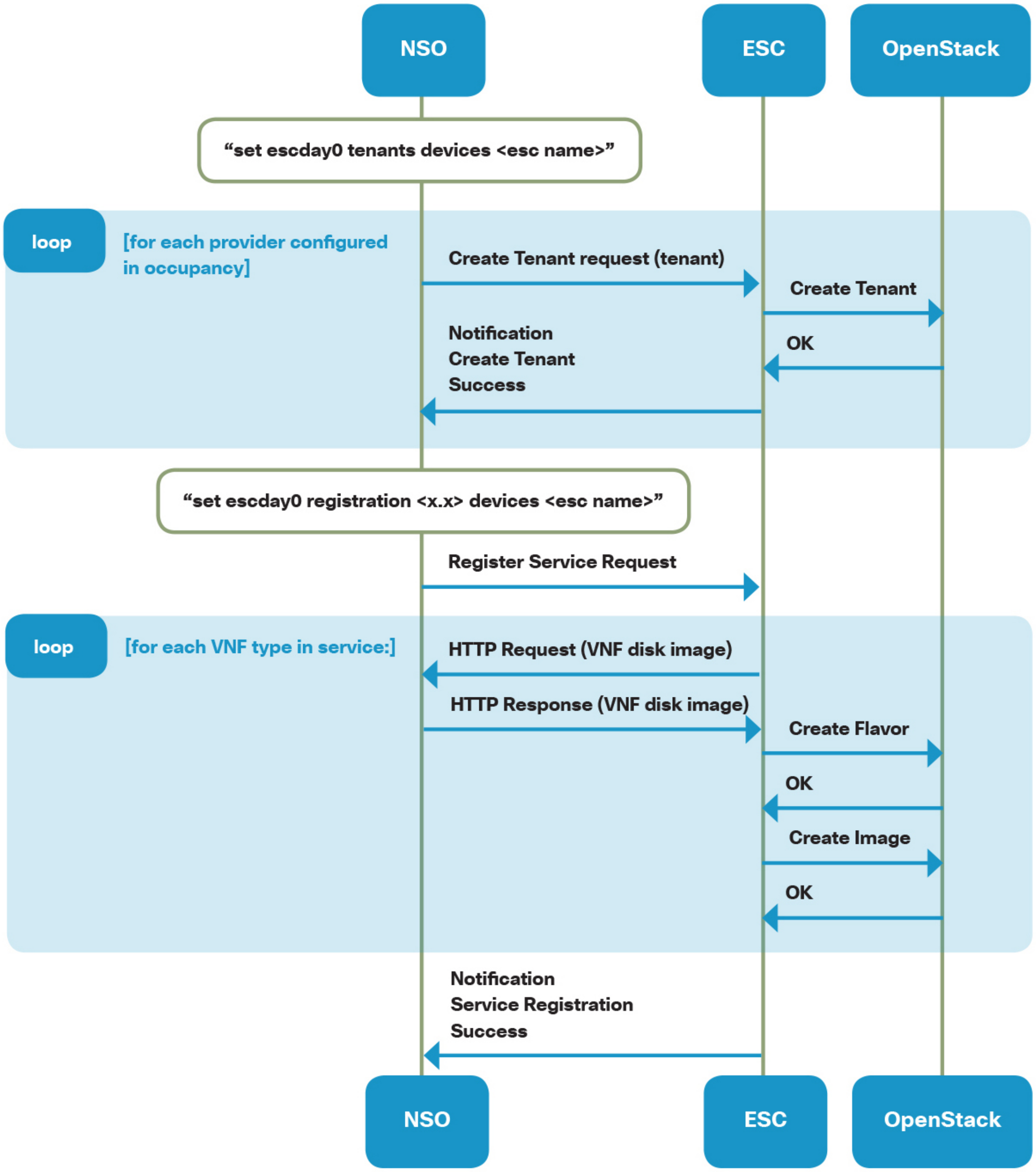
2. Service Registration is initiated for each VNF type supported by the solution:

- ESC downloads VNF image from NSO via HTTPS

- ESC creates a Nova flavor for each VNF type
- ESC uploads the image to Glance

At this point, all the needed VNF flavors and images are registered in ESC and OpenStack Glance.

Figure 2.4 ESC registration call flow


NSO
ESC
OpenStack
“set escday0 tenants devices <esc name>”
loop
[for each provider configured in occupancy]
Create Tenant request (tenant)
Create Tenant
OK
Notification
Create Tenant
Success
“set escday0 registration <x.x> devices <esc name>”
Register Service Request
loop
[for each VNF type in service:]
HTTP Request (VNF disk image)
HTTP Response (VNF disk image)
Create Flavor
OK
Create Image
OK
Notification
Service Registration
Success
NSO
ESC
OpenStack

Call Flows - Service Chain Flows

Service Chain Creation Flow

Each time an end customer deploys a new service as a part of the vMS solution, a new Service Chain will be created in the underlying infrastructure. This section describes a call flow for a Service Chain Creation. Due to the higher complexity of this flow diagram, a simple version of Service Chain creation will be explained, and a figure with slightly more details will be shown also.

Simplified Service Chain creation flow is described in Figure 2.5.

Service Chain Creation Flow explained

1. The Service Chain creation process begins with an end customer's interaction with the vMS portal, and selecting of desired service (Basic, Medium, Full).

2. Once all the details are selected, and the end customer submits the order through the vMS Portal, the Portal sends a NETCONF "**edit-config**" message with all the relevant configuration parameters (i.e. CloudVPN data, Provider etc.) to NSO.

3. NSO receives the order using NETCONF, and based on the Provider field sent by the Portal, sends a "**Deploy Service Request**" message to the ESC.

4. ESC retrieves Day 0 device configuration and license files from NSO via an HTTPS download.

5. ESC creates an "Inside" network and subnet in OpenStack. The inside network is used to connect the CSR1000v to the ASAv and WSAv internally.

Once OpenStack creates network and subnet successfully, ESC will send a **Notification - Create Network/Subnet - Success** message back to the NSO.

6. At this stage, ESC will boot each VNF individually, with Day 0 configurations as config-drive.

7. ESC will send a **VM_DEPLOYED** message to NSO for each VNF that was successfully booted.

> **Note:** For performance reasons, all of the VNFs in a Service Chain are deployed on the same OpenStack Compute Node. In Medium and Full Service Chains, the ASAv is scheduled to be deployed first. ESC then checks which compute host it was deployed to and requests other VNF's are deployed to the same compute host.

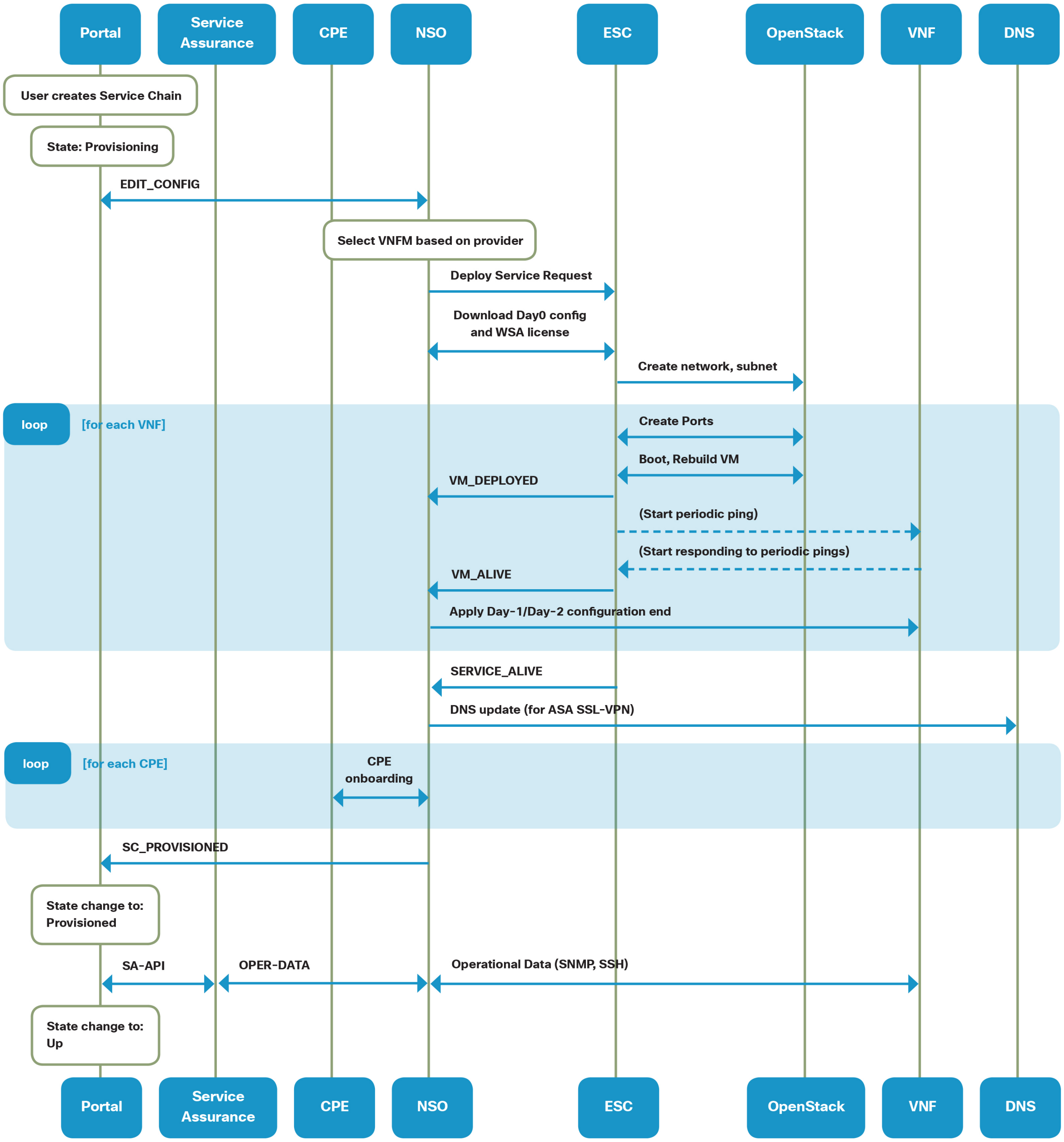
7. ESC starts to poll VM status in Nova. Once the VM status is ALIVE, then ESC starts to monitor them via ICMP. When the VM responds, ESC sends **VM_ALIVE** to NSO for each VNF.

8. When all VNFs are in **VM_ALIVE** status, the entire Service Chain will be declared as **SERVICE_ALIVE** status, and a message sent to NSO.

9. A **SERVICE_ALIVE** message indicates that the NSO can then deploy: Day 1 and Day 2 configurations to each of the VNFs in the service chain.

10. If ASAv is part of a Service Chain, after Service status is changed to alive, NSO will update DNS records with FQDN of newly created ASAv (for SSL VPN purposes).

Figure 2.5 Service Chain Creation Simplified Call Flow



The following flow diagram describes a detailed end-to-end Service Chain creation flow, with more detailed messages sent between Orchestration components and VNFs. A detailed call flow is described below as well as in Figure 2.6.

Figure 2.6 Service Chain Creation Detailed Call Flow

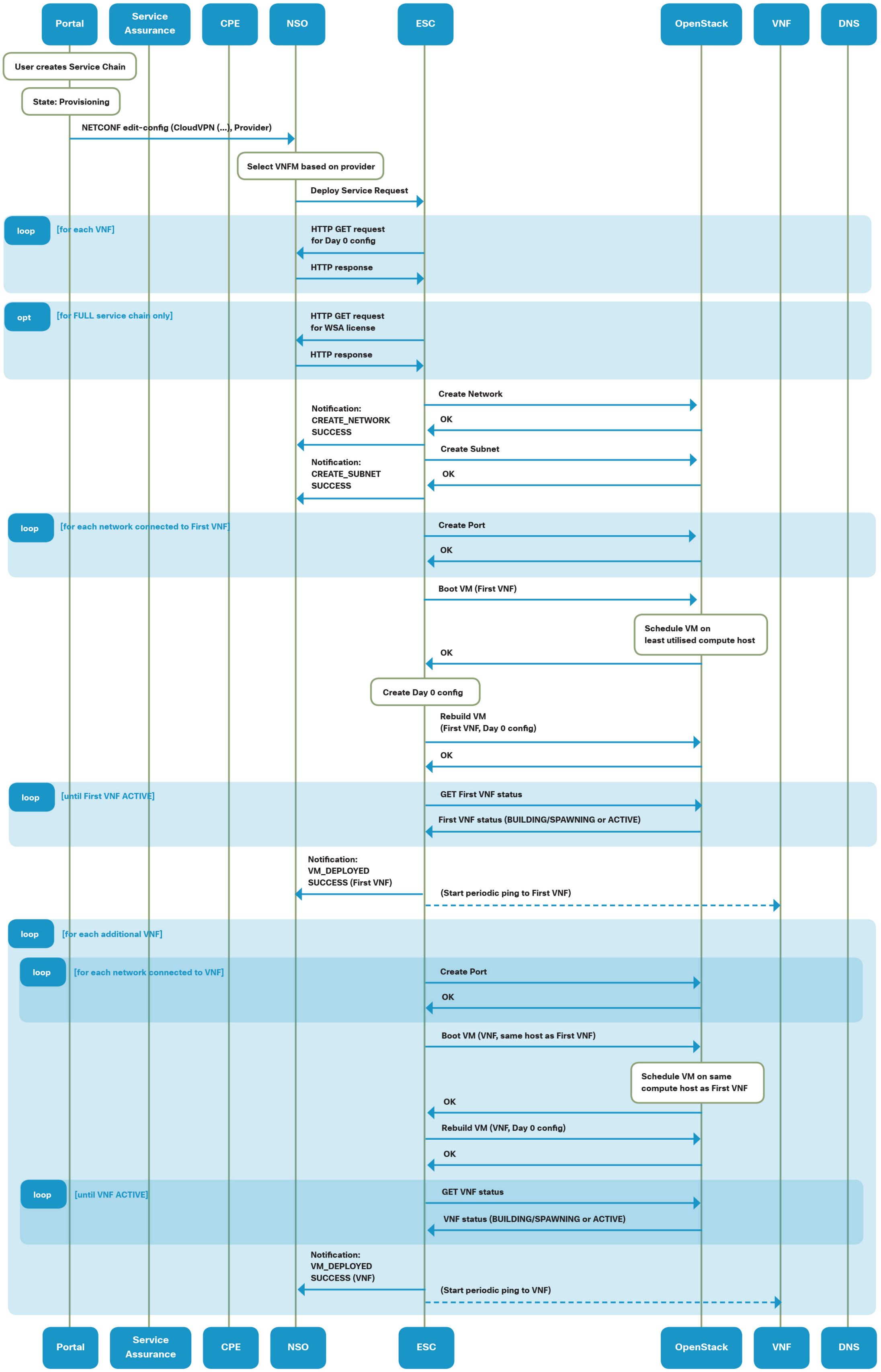
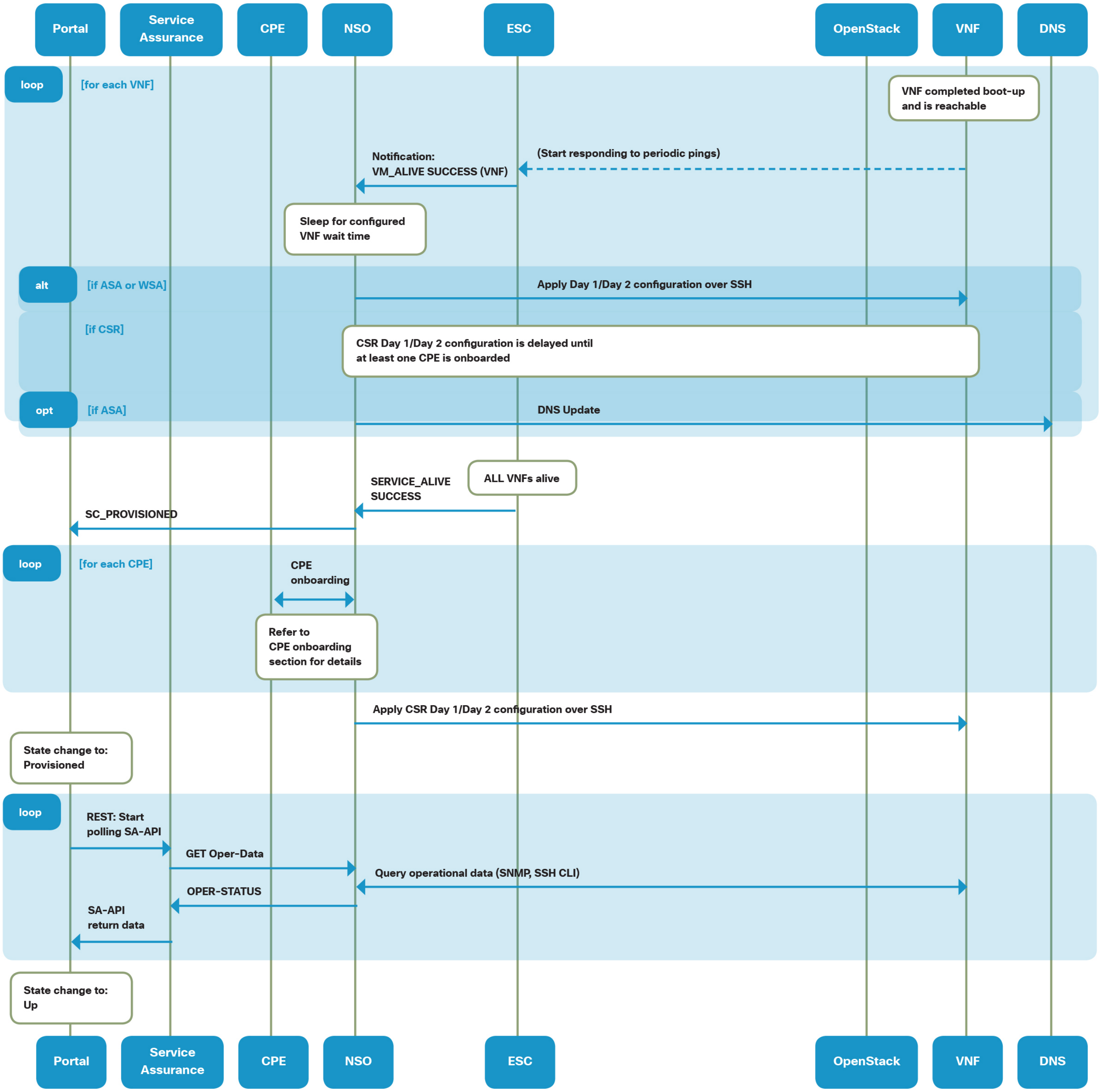

Portal
Service Assurance
CPE
NSO
ESC
OpenStack
VNF
DNS
User creates Service Chain
State: Provisioning
NETCONF edit-config (CloudVPN (...), Provider)
Select VNFM based on provider
Deploy Service Request
loop
[for each VNF]
HTTP GET request for Day 0 config
HTTP response
opt
[for FULL service chain only]
HTTP GET request for WSA license
HTTP response
Create Network
OK
Notification: CREATE_NETWORK SUCCESS
Create Subnet
OK
Notification: CREATE_SUBNET SUCCESS
loop
[for each network connected to First VNF]
Create Port
OK
Boot VM (First VNF)
Schedule VM on least utilised compute host
OK
Create Day 0 config
Rebuild VM (First VNF, Day 0 config)
OK
loop
[until First VNF ACTIVE]
GET First VNF status
First VNF status (BUILDING/SPAWNING or ACTIVE)
Notification: VM_DEPLOYED SUCCESS (First VNF)
(Start periodic ping to First VNF)
loop
[for each additional VNF]
loop
[for each network connected to VNF]
Create Port
OK
Boot VM (VNF, same host as First VNF)
Schedule VM on same compute host as First VNF
OK
Rebuild VM (VNF, Day 0 config)
OK
loop
[until VNF ACTIVE]
GET VNF status
VNF status (BUILDING/SPAWNING or ACTIVE)
Notification: VM_DEPLOYED SUCCESS (VNF)
(Start periodic ping to VNF)
Portal
Service Assurance
CPE
NSO
ESC
OpenStack
VNF
DNS

Diagram continues


Portal
Service Assurance
CPE
NSO
ESC
OpenStack
VNF
DNS
loop
[for each VNF]
VNF completed boot-up and is reachable
(Start responding to periodic pings)
Notification: VM_ALIVE SUCCESS (VNF)
Sleep for configured VNF wait time
alt
[if ASA or WSA]
Apply Day 1/Day 2 configuration over SSH
[if CSR]
CSR Day 1/Day 2 configuration is delayed until at least one CPE is onboarded
opt
[if ASA]
DNS Update
ALL VNFs alive
SERVICE_ALIVE SUCCESS
SC_PROVISIONED
loop
[for each CPE]
CPE onboarding
Refer to CPE onboarding section for details
Apply CSR Day 1/Day 2 configuration over SSH
State change to: Provisioned
loop
REST: Start polling SA-API
GET Oper-Data
Query operational data (SNMP, SSH CLI)
OPER-STATUS
SA-API return data
State change to: Up
Portal
Service Assurance
CPE
NSO
ESC
OpenStack
VNF
DNS

Service Chain Deletion Call Flow explained

A detailed sequence diagram of a Service Chain deletion operation is shown in Figure 2.7.

1. A Service Chain deletion operation starts by the end customer selecting the option to delete an existing Service from vMS Portal.

The Portal sends a NETCONF **delete-config** message to the NSO server.

2. Since ASAv and CSR1000v VNFs licensing is done using Cisco Smart Licensing, those licenses are released first.

- NSO sends commands to ASAv and CSR1000v that will unregister the VMs from Smart Licensing and release the license from the license pool.
- NSO sends a request via DNS-Updater NED to delete the DNS record for the ASAv

3. NSO sends a **Undeploy Service Request** message to ESC.

4. ESC starts the process of deleting the service chain.

4.1. ESC first stops periodic pings to all VNFs that are part of a Service Chain

4.2. ESC requests Nova to delete all VMs

4.3. ESC pools periodically check OpenStack for the status of the VNF that is in the deletion process

- ESC Sends get_status(VNF) message to OpenStack
- OpenStack replies with status(VNF)
- This process continues until ESC receives VM status: **deleted**

4.4. As each VNF is deleted, ESC sends a **VM_UNDEPLOYED**(VNF) message back to NSO

5. ESC now does the "clean-up" of the virtual network configurations related to the deleted Service chain by:

5.1. ESC deletes the Subnet:

- ESC sends: **Delete_subnet** message to OpenStack
- OpenStack replays with OK
- ESC Sends information about successful subnet deletion to NSO: using Notification: **DELETE_SUBNET_SUCCESS** message

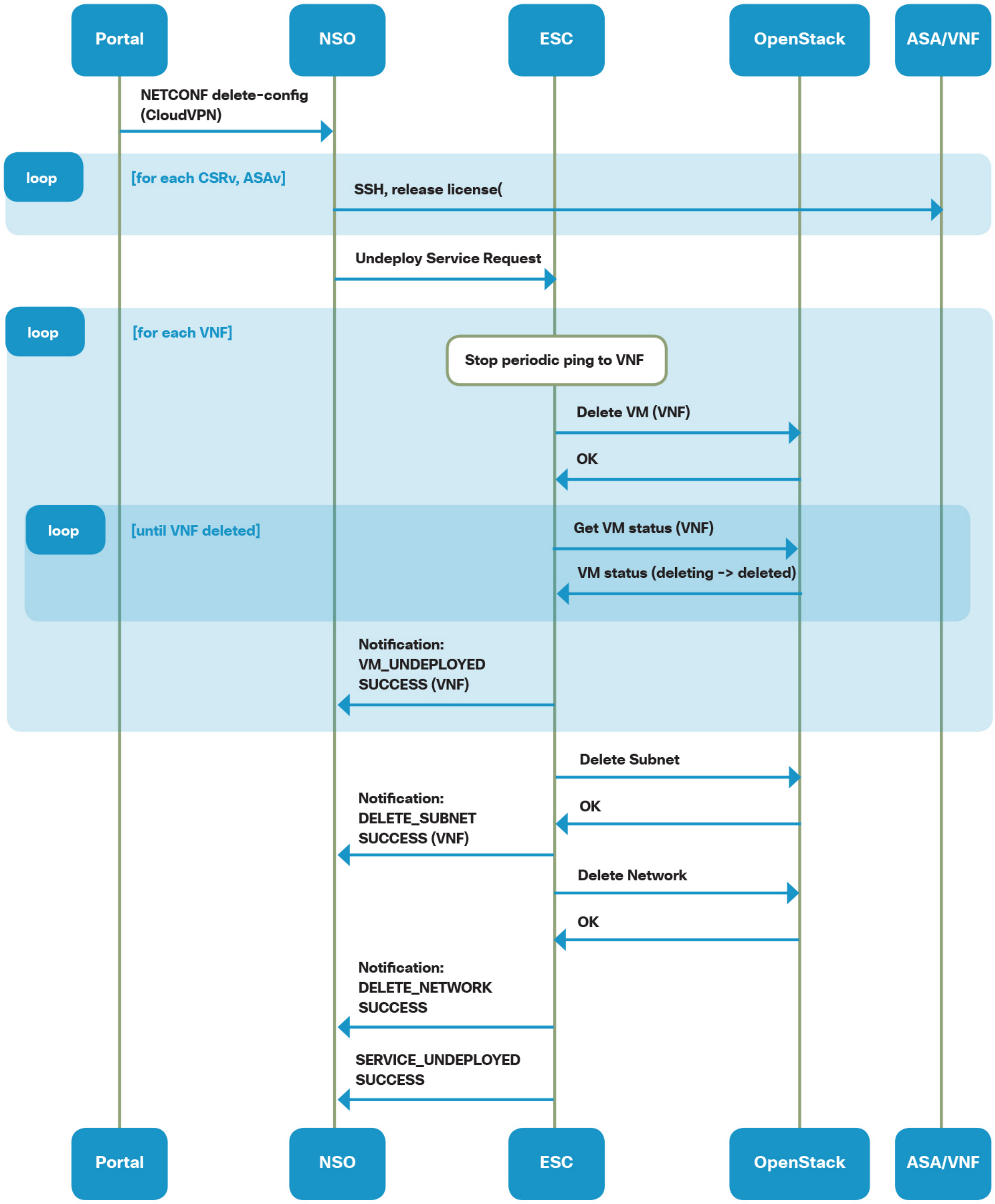
5.2 ESC deletes Network element that was part of deleted Service Chain

- ESC sends: **Delete_network** message to OpenStack

- OpenStack replays with OK
- ESC Sends information about successful subnet deletion to NSO, and information about Service being undeployed
  - ESC -> NSO: sends Notification: **DELETE_NETWORK_SUCCESS,**
  - ESC -> NSO: sends **SERVICE_UNDEPLOY_SUCCESS** message.

A **SERVICE_UNDEPLOY_SUCCESS** message, seen on NSO means that the Service Chain has been successfully undeployed.

Figure 2.7 Service Chain Deletion Call Flow



Service Chain Re-deploy Call Flow

Service Chain re-deploy procedure simply consists of a service undeploy followed by a deploy. Call flows of service chain redeploy operation are described in detail in sections: "Service Chain Deletion" and "Service Chain Creation" in this chapter.

Service Chain Upgrade and Downgrade Flows

Once end customers have the vMS Service Chain deployed and running, they could decide to upgrade or downgrade their level of service. For example, an end customer might start with a simple service, such as Basic, and then decide to purchase and migrate to a Full Service offering.
The process of changing a level of service in vMS terminology is called Upgrade/Downgrade process, and this section will describe detailed call flows that are the result of end customer's upgrade or downgrade request in the vMS Portal. A detailed sequence diagram of a Service Chain upgrade call flow is shown in Figure 2.8

Service Chain Upgrade Call Flow

1. An end customer configures the upgrade of the existing Service from either Basic Service to Medium or Full, or from Medium Service to Full. Depending on the actual upgrade path, NSO and ESC would need to add either ASAv VNF, or ASAv and WSAv VNFs.

The Portal sends NETCONF **edit-config** message to NSO with all the parameters reflecting the configuration change.

2. NSO communicates the update / adding a VNF to ESC by sending the "**Update Service Request**" message.

3. ESC starts the process of adding VNFs to the Service chain.

  3.1 ESC downloads Day 0 configuration via HTTPS from NSO

  3.2 ESC creates neutron ports in OpenStack for each network interface on the new VNF

  3.3 ESC boots new VM (VNF is scheduled to be booted on the same compute node as the first VNF in the Service Chain).

  3.4 ESC Rebuilds VM, and initializes new VNF with appropriate Day 0 configuration.

  3.5 ESC sends periodic requests to OpenStack checking the VM status. (BUILDING or SPAWNING or ACTIVE).

3.5 Once ESC receives **ACTIVE** state from OpenStack, it sends **VM_DEPLOYED SUCCESS** (VNF) message to NSO

3.6 ESC starts periodic ping requests towards the newly-created VNF

3.7 When pings start getting a response, ESC sends **VM_ALIVE** to NSO.

3.8 NSO deploys Day 1 and Day 2 configurations to the VNF

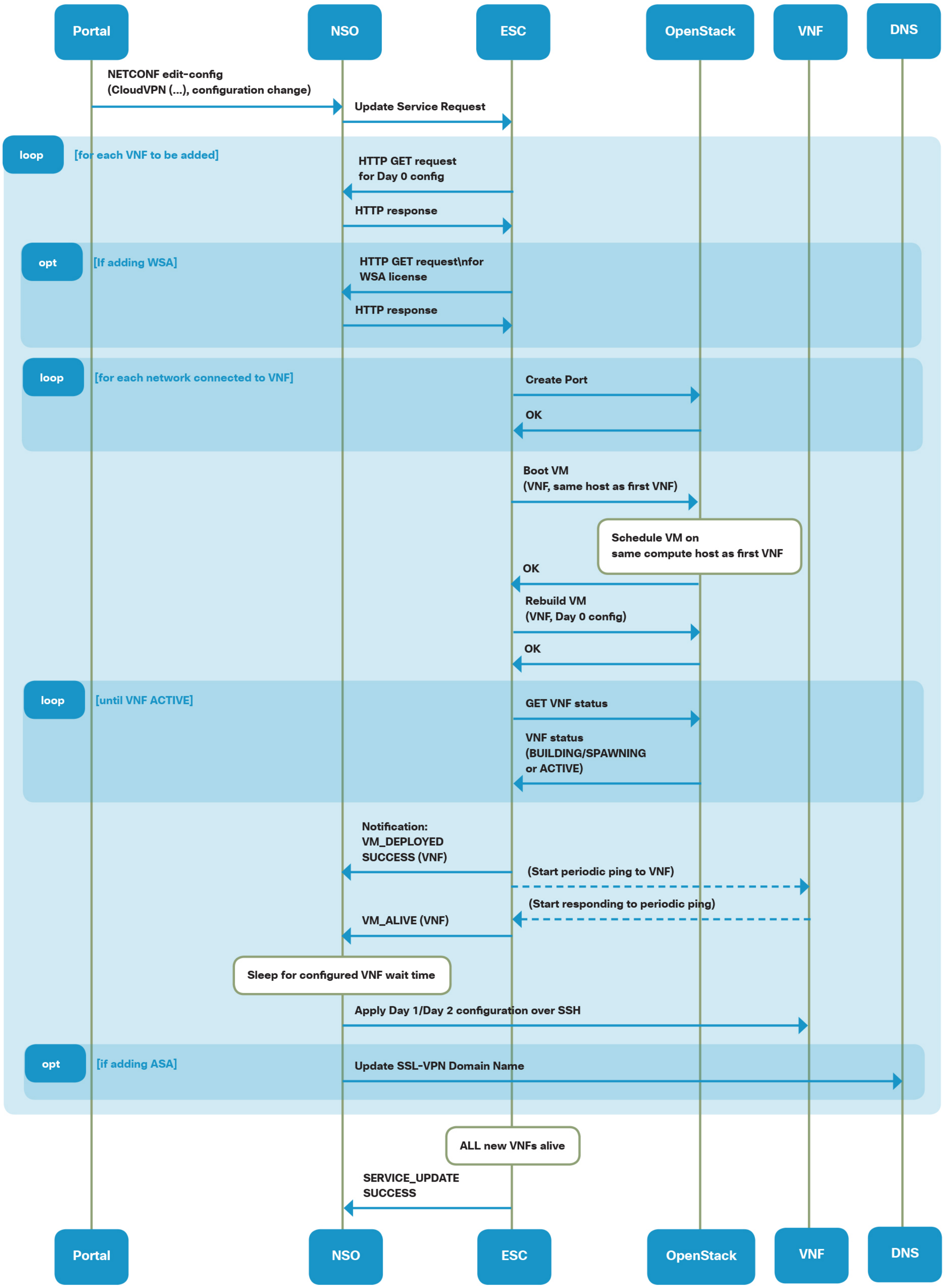
3.8.1 When adding an ASAv to a service chain, NSO additionally registers the DNS name using the DNS-Updater NED.

4. When all VNFs (including new VNF added to the Services) are in **VM_ALIVE** state, ESC sends **SERVICE_UPDATED SUCCESS** message back to NSO

This concludes Update Service Chain process.

Upgrade chain process is displayed in the following flow diagram.

Figure 2.8 Service Chain Upgrade Call Flow



Service Chain Downgrade Call Flow

The Service Chain downgrade scenario is somewhat similar to the upgrade scenario, and deals with changing the level of service from higher to lower (i.e. from Full back to Basic or Medium, or from Medium back to Basic).

A detailed sequence diagram of a Service Chain downgrade call flow is shown in Figure 2.9

1. An end customer selects a downgrade of the existing Service.

Depending on the actual downgrade path, NSO and ESC would need to delete WSAv or ASAv and WSAv VNFs.

Portal sends NETCONF **edit-config** message to NSO with all the parameters reflecting the configuration change.

2. NSO performs "Release License" configuration on deleted VNFs (ASAv), to release the license from Cisco Smart Licensing Server.

> 2.1 When removing an ASAv from a service chain, additionally NSO will also deregister the DNS name using the DNS-Updater NED.

3. NSO sends an "Update Service Request" message to ESC, containing Service Chain Changes.

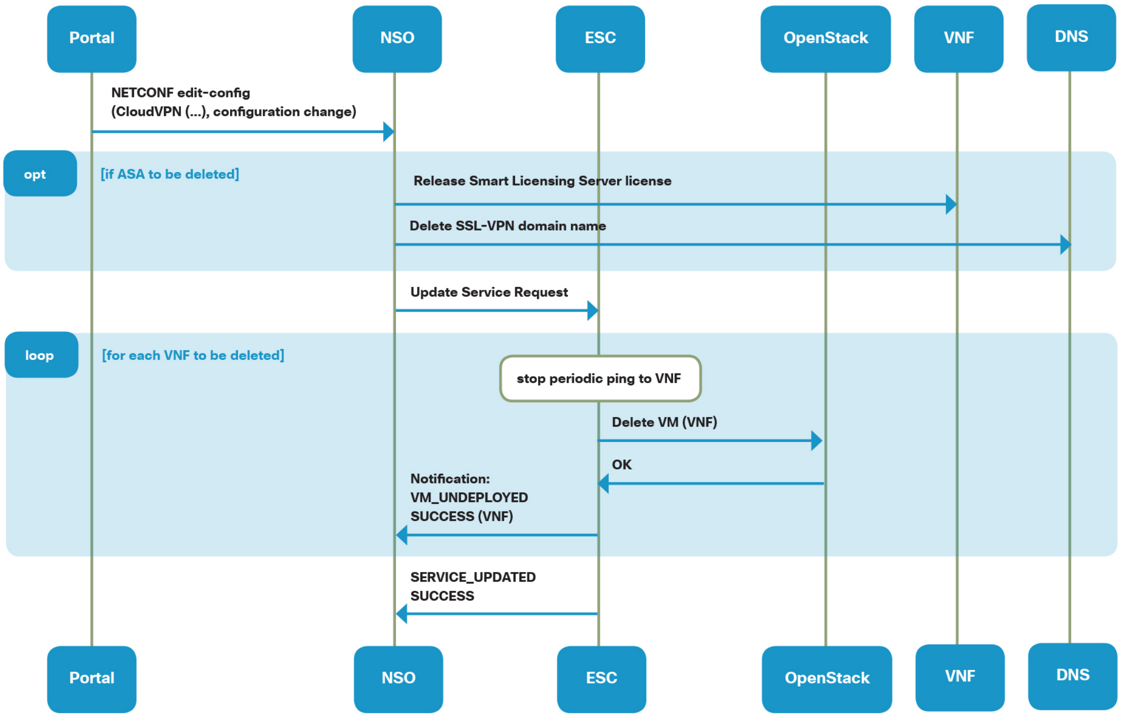
4. For each VNF that is deleted, ESC will:

> 4.1 Stop periodic health-check by stopping pings towards VNF scheduled for the deletion.
>
> 4.2 ESC deletes VNF from OpenStack.
>
> 4.3 ESC sends Notification message to NSO: **VM_UNDEPLOYED SUCCESS (VNF)**

5. Once all VNFs are undeployed successfully, ESC sends Notification message **SERVICE_UPDATED SUCCESS** to NSO.

This concludes the Service Chain Downgrade operation.

Figure 2.9 Service Chain Downgrade Call Flow



Service Chain Modification Flow

End customers of vMS-enabled services have the option to modify their existing Service level, without upgrading or downgrading. Based on the Service level that is used (Basic/Medium or Full), additional services such as Quality of Service, bandwidth settings, and different URL Filtering Security Levels can be selected through vMS Portal. This section describes actions / call flows that will happen in the background between vMS components when settings of an existing Service are changed. The flow diagram shows a general example, as the communication and call flows are the same for all Modification operations. The flow diagram shows a general example, without going into details of the VNF configurations actually changed.

A detailed sequence diagram of a Service Chain modification call flow is shown in Figure 2.10.

Service Chain Modification Flow Call Flow explained

1. The modification process starts WITH an end customer selecting additional details for an existing Service by modifying any of the following settings:

- Number of remote SSL VPN users (configuration change is reflected in ASAv configuration)
- Desired bandwidth and traffic prioritization (configuration change is reflected on CSR1000v and CPE)
- Web Security URL Filtering level (configuration change done on WSAv VNF - **applicable to Full Service offerings only**:

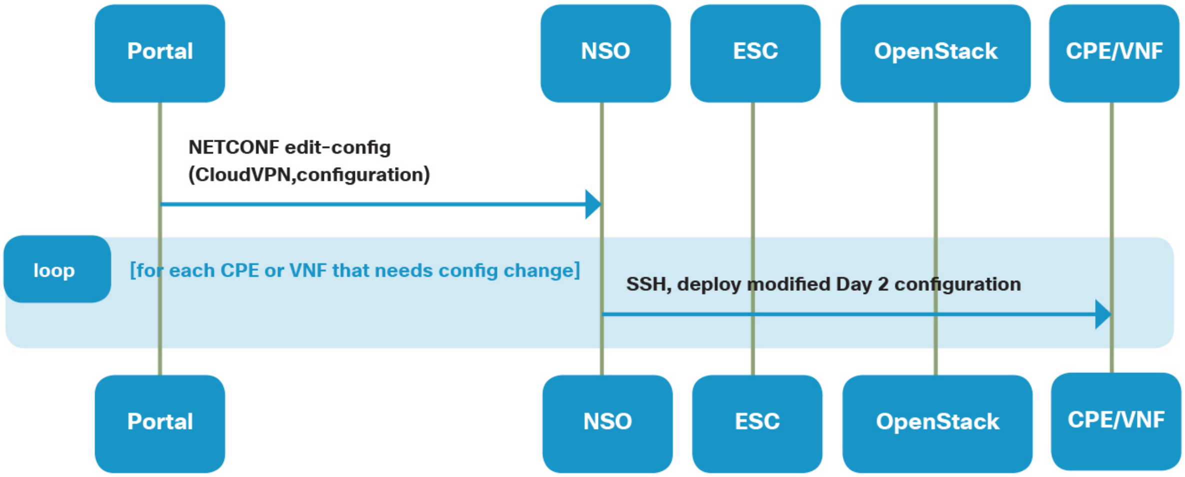
- Basic
- Enhanced
- High

2. vMS Portal sends NETCONF **edit-conf** message to NSO, with parameters of CloudVPN, and representation of configuration changes.

3. Based on configuration changes received by vMS Portal, NSO applies new Day 2 configuration templates to corresponding VNFs, and changes internal NSO configuration to reflect the changes (more details in chapter 5 "Network Services Orchestrator").

This ends the process of Service Chain modification.

Figure 2.10 Service Chain Modification Call Flow



3. vMS Use Cases

3.1 Introduction

The Cisco vMS solution is an open NFV service orchestration platform that can be utilized to orchestrate many cloud-based use-cases or deployment scenarios. Content can be created for the vMS orchestration components (Portal, NSO, ESC etc.) to support different use-cases. However, the vMS solution in each release will include out-of-box content to orchestrate specific use cases. This content includes the following:

- NSO - YANG service models, device configuration templates, NEDs, orchestration logic. This NSO content is also referred to as the NSO Function Pack.
- ESC - YANG data models, VNF deployment definitions
- Portal - UI and user experience for ordering and managing services and visualizing metrics

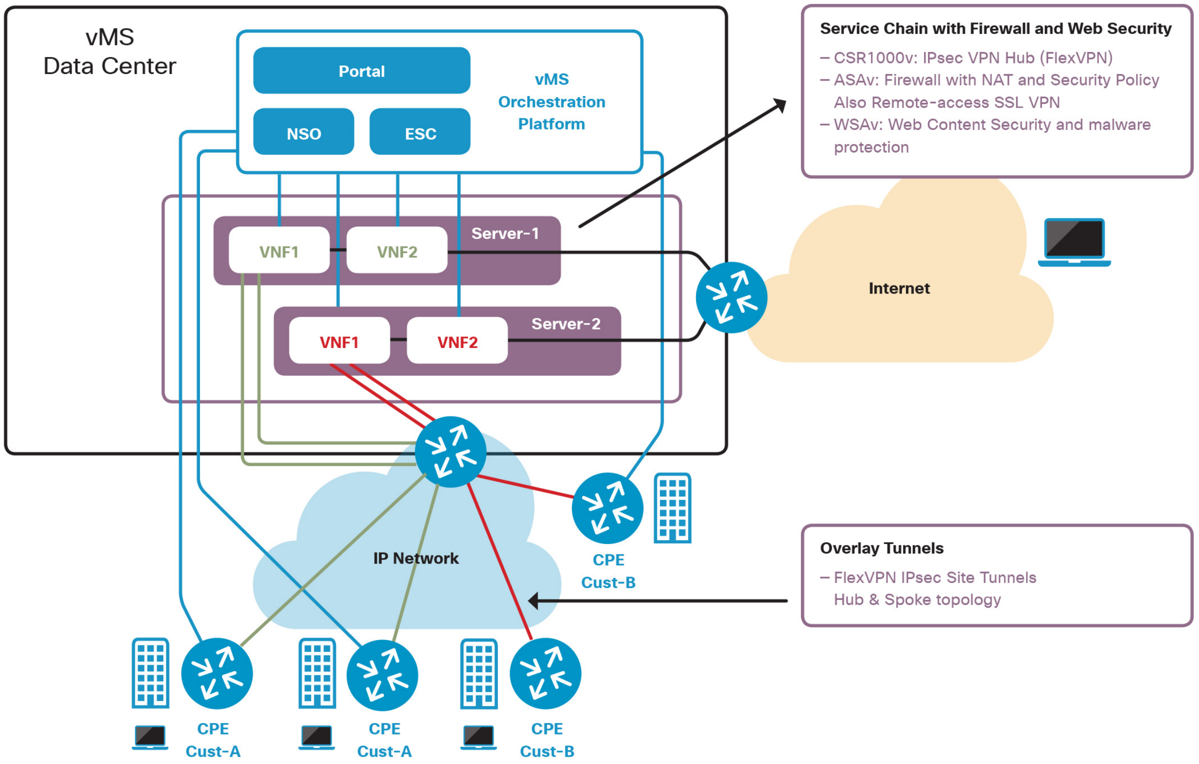
The overall content for the use-case is packaged as the vMS Service Package (along with the software for the NSO, ESC, Portal and other vMS components). The vMS 2.0/2.1 solution includes out-of-box support for the Cloud VPN use-case. Future versions of the vMS solution will include out-of-the-box content (Service Package and Function Packs) for additional services and use-cases.

This chapter will document the CloudVPN use-case and the different service chains or topologies possible with the vMS 2.0/2.1 deployment.

3.2 CloudVPN Service

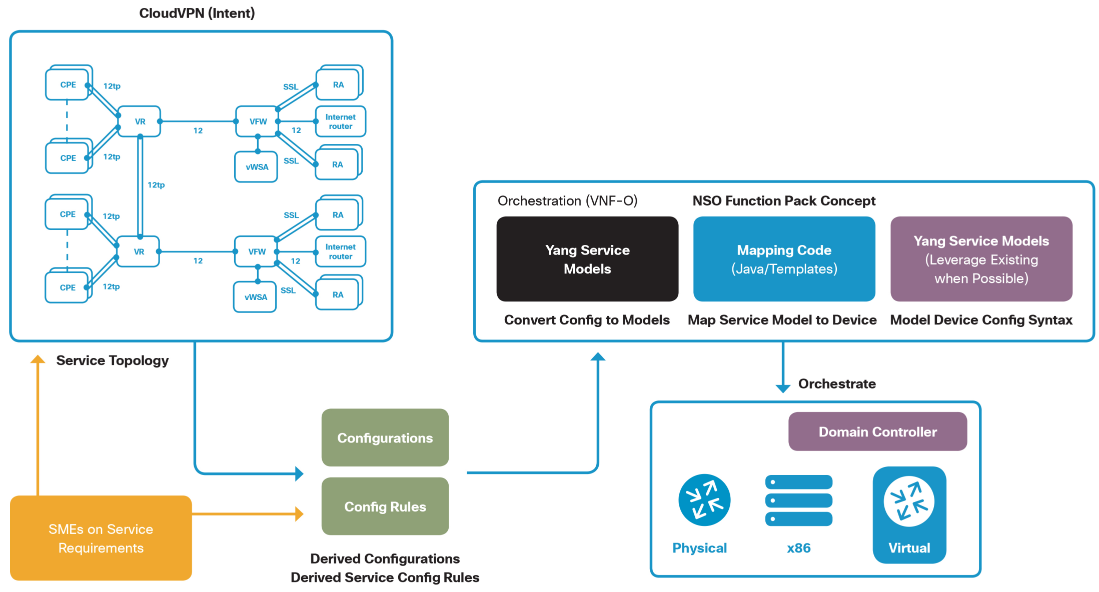
The vMS 2.0/2.1 solution release (current version at the time of writing of this book) supports the CloudVPN Function Pack. In this deployment model, the Service provider will use the vMS CloudVPN service to provide cloud-based managed services such as virtual IPsec VPN hub, SSL VPN hub, Firewall, Web content filtering, Anti-Malware and Anti-Virus capabilities. End customer CPEs connect via the IPsec tunnels (implemented with FlexVPN) to the CloudVPN hub in order to communicate with each other and also to securely access the Internet. The following figure illustrates the CloudVPN use case and deployment topology.

Figure 3.1 vMS CloudVPN Service Use Case


vMS
Data Center
Portal
NSO
ESC
vMS
Orchestration
Platform
Server-1
VNF1
VNF2
Server-2
VNF1
VNF2
Service Chain with Firewall and Web Security
– CSR1000v: IPsec VPN Hub (FlexVPN)
– ASAv: Firewall with NAT and Security Policy
Also Remote-access SSL VPN
– WSAv: Web Content Security and malware
protection
Internet
IP Network
CPE
Cust-B
Overlay Tunnels
– FlexVPN IPsec Site Tunnels
Hub & Spoke topology
CPE
Cust-A
CPE
Cust-A
CPE
Cust-B

The vMS CloudVPN Service Package includes all content necessary to provision the CloudVPN use case. This includes the NSO Function Pack for CloudVPN, which contains the YANG service/data models, device templates, NEDs, and orchestration logic. The following figure illustrates the NSO Function Pack concept.

Figure 3.2 NSO Function Pack



3.3 CloudVPN Service Chains

As described in Chapter 2.2 "vMS Overview", the CloudVPN Service Package includes two services - the CloudVPN Foundation (with CSR1000v), and CloudVPN Advanced (with CSR1000v, ASAv, WSAv). The CloudVPN Foundation package allows the end customer to deploy the CloudVPN Basic Service chain, which provides a secure encrypted site-to-site VPN between two or more customer sites.

The CloudVPN Advanced package allows the end customer to deploy one of two types of service chains - CloudVPN Medium Service Chain, or CloudVPN Full Service Chain. The CloudVPN Medium adds security options by adding a firewall that also offers secure access for mobile users. The CloudVPN Full adds content/URL filtering functionality, that enables controlling the access to illegal and/or business inappropriate content. The Full service also adds an additional level of security, by providing Anti-Malware and Anti-Virus services.

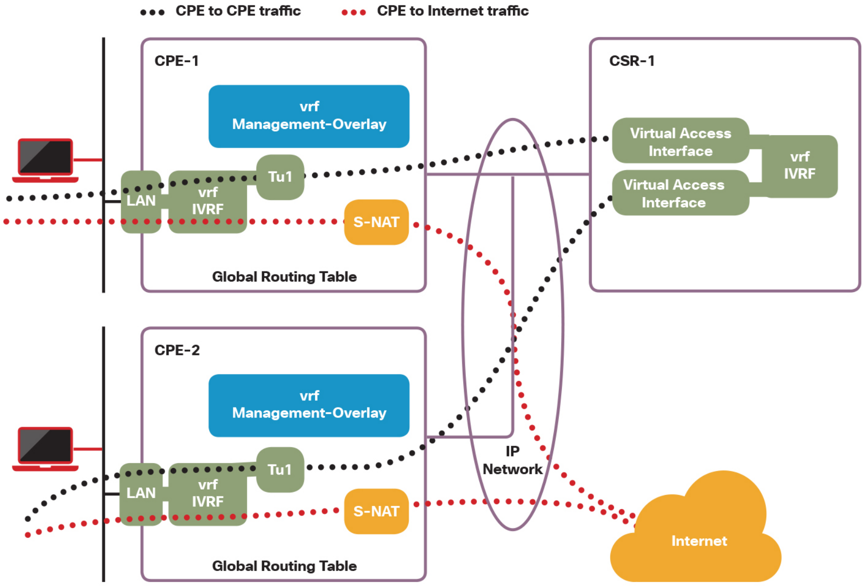
The CloudVPN service package and related NSO Function Pack provide a fully orchestrated, end-user driven FlexVPN and Internet firewall service. End customers order their CloudVPN service through a service portal, selecting optional security features before registering one or more CPEs to create a site-to-site VPN.

The next section will provide more details on the three variants of the CloudVPN service - Basic, Medium and Full.

CloudVPN Basic

The CloudVPN Basic service provides a secure site-to-site VPN service using Cisco FlexVPN technology. Through NSO orchestration, multiple end-customer site CPEs are connected together using IPsec tunnels via a Cisco CSR1000v hosted in the Service Provider data center. In the Basic service model, access outside of the secure VPN to the Internet is via a local NAT configuration on each CPE. The end customer has the option to add their own security between the CPE and an external network.

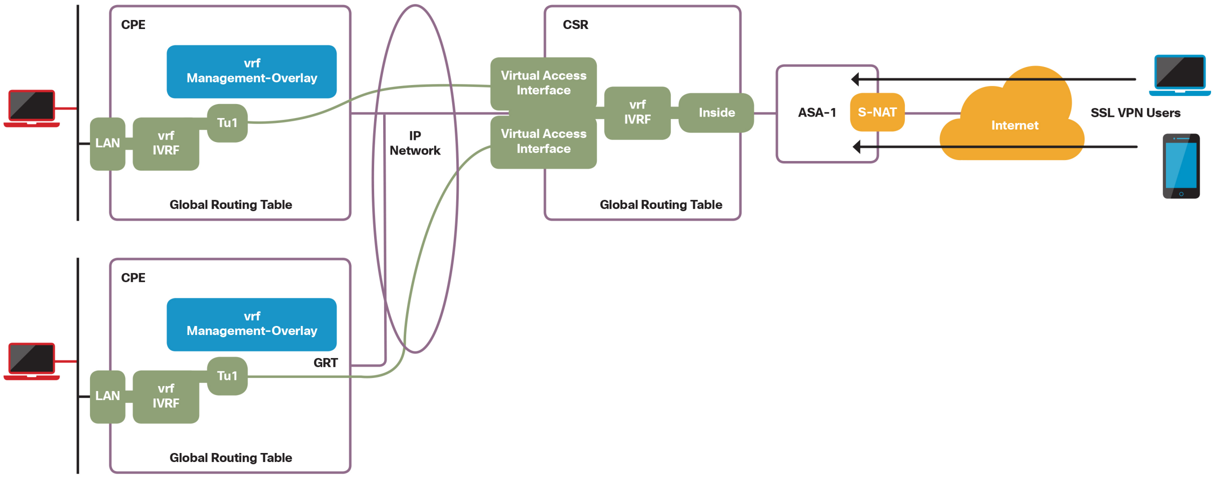
Figure 3.3 Basic Service Chain Topology



CloudVPN Medium

The CloudVPN Medium service provides a secure site-to-site VPN service using Cisco FlexVPN technology. This service adds a Cisco ASAv Firewall on top of the Basic model and routes all Internet traffic from the sites through the firewall. This model also adds the capability for remote user access via SSL VPN to the ASAv. Remote SSL VPN users are also able to access internal networks behind the CPE devices.

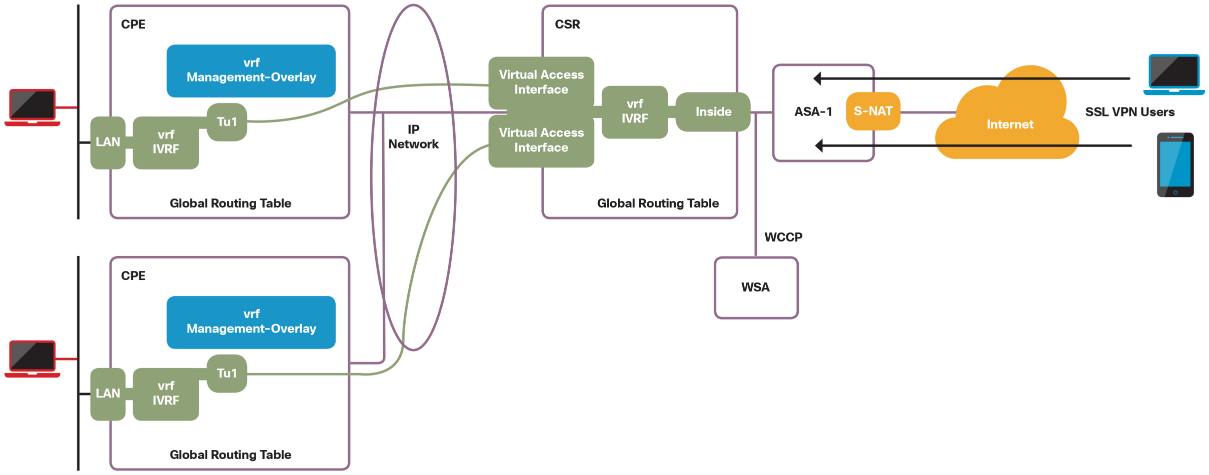
Figure 3.4 Medium Service Chain Topology



CloudVPN Full

The CloudVPN Full service provides a secure site-to-site VPN service using Cisco FlexVPN technology plus secure Internet break out with SSL VPN and Web URL filtering. HTTP traffic from CPEs to the internet is redirected using WCCP via a Cisco WSAv providing Web filtering, and Anti-Virus and Anti-Malware security services.

Figure 3.5 Full Service Chain Topology



4. Portal, Identity Management and Service Assurance

4.1 Introduction

This chapter provides details of operational and troubleshooting steps for the following vMS sub-systems:

- Portal - the user interface for ordering, managing and monitoring vMS services. The Portal sub-system is built on a Microservices framework.
- Identity Management (IDM) - for creating and managing user credentials for accessing the Portal
- Service Assurance - for service status and statistics

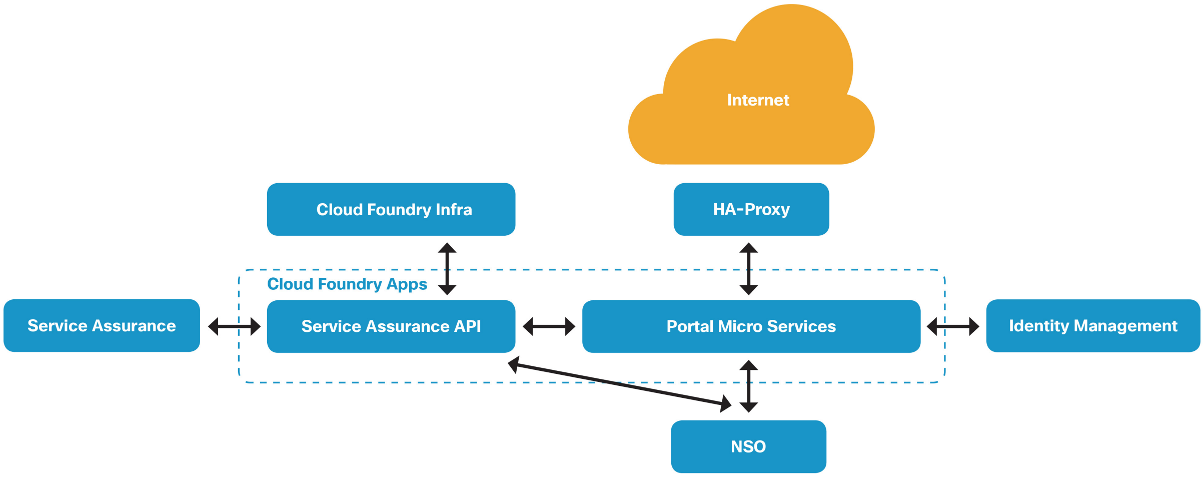
This chapter also provides an overview of the open source Cloud Foundry Platform-as-a-Service (PaaS) environment that is utilized to deploy and manage the above three sub-systems. In addition, this chapter also contains Portal flows for Service Provider and End Customer operations.

4.2 Portal, SA and IDM Sub-systems

Overview

The Portal sub-system is the vMS interface for the Provider administrator for creating and managing tenants (end customers). It is also the interface for end customers to order, manage and monitor vMS CloudVPN services. The user management and authentication for Single Sign-On (SSO) is enabled by the Identity Management (IDM) sub-system. Service status and statistics for deployed services are displayed on the Portal which obtains this information from the Service Assurance (SA) sub-system. Cloud Foundry PaaS is utilized in the vMS solution to deploy and manage the Portal, IDM and SA sub-systems. The following figure illustrates the relationships between Cloud Foundry, vMS Portal, vMS IDM and vMS Service Assurance.

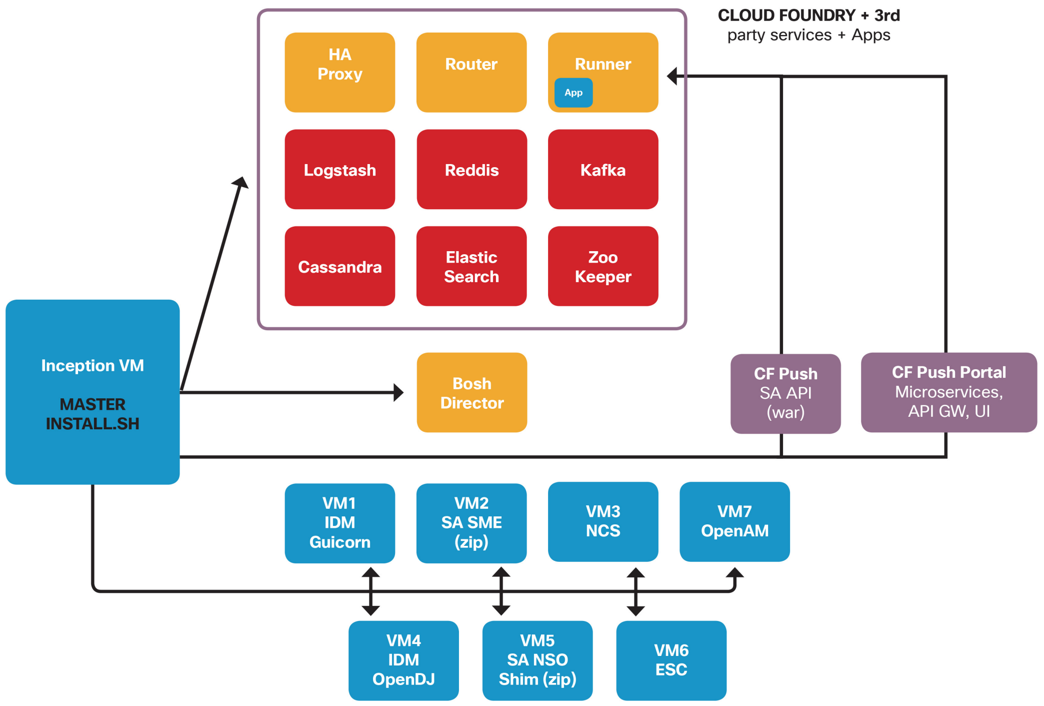
Figure 4.1 vMS Portal, IDM and SA Sub-systems



Cloud Foundry PaaS for Portal, IDM and SA

The open-source Cloud Foundry PaaS is not a vMS core component. It is utilized to streamline and simplify deployment and management of the Portal, ID Management, and Service Assurance sub-systems in the vMS solution. For applications built using a microservices framework (such as the Portal components and SA-API), Cloud Foundry can provide as-a-Service capabilities (discovery, monitoring, elasticity etc.). The Portal troubleshooting and log collection are done through the Cloud Foundry system. The following figure illustrates the Cloud Foundry infrastructure utilized within the vMS solution.

Figure 4.2 Cloud Foundry PaaS for deploying Portal, IDM, SA components.



Inception VM

The Inception VM, as seen in the high-level architecture diagram above, is the VM that will hold all the tools, software, and scripts needed to install the vMS solution. This includes the images and configuration files for NSO, ESC, Portal, Cloud Foundry, VNFs (CSR1000v, ASAv, WSAv, etc). The Inception VM is also the jumphost from which Bosh Director, Cloud Foundry and all other applications and third-party services will be deployed. The Inception VM is required to be able to run Ruby and BOSH CLI.

Cloud Foundry (CF) Components

The following VMs make up the Cloud Foundry infrastructure:

- **BOSH Director VM**: This VM controls and manages the various components in the deployment such as Director, BOSH Agent, Health Manager, etc
- **Cloud Foundry VMs**: When CF is deployed, the following VMs will be created:
  - BOSH Agent VM
  - HA Proxy VM

  - Droplet Execution Agent (DEA) VMs (2 instances). Applications are deployed on the DEA VMs when a CF push is done for each application
  - Router VM
- **Services VMs**: Following services will be deployed on VMs and all service VMs are in redundant mode
  - Cassandra
  - Redis
  - Kafka
  - Logstash

vMS Portal

vMS Portal acts as a front end to the vMS Solution where customers log in to order, monitor and use their managed services.

The vMS Portal Application consists of the following microservices:

- **Consume** - Ordering Service Chains through the Portal
- **Manage** - Manage Existing Service Chains on the Portal
- **Monitor** - Communicates with Service Assurance components
- **Notification** - Sends e-mail notifications
- **Orchestration** - Communicates with NCS
- **Identity Management** - Communicates with Identity Management VMs
- **Logconsumer** - Collect logs from all the microservices
- **Skyfall UI** - User facing interface
- **Administration**

There are multiple views for the Portal UI: Admin, Operator and Consumer/Tenant User.

vMS Identity Management(IDM)

Responsible for management of user credentials and mapping users to particular tenants. vMS Identity Management stack includes the following three VMs.

- IDM API - Exposes REST APIs to manage users and their mapping to tenants.
- IDM OpenDJ - Open source LDAP Server.
- IDM OpenAM - Open source AAA Server.

vMS Service Assurance (SA)

Responsible for Device and Service status, and statistics displayed on the Portal. vMS SA stack includes the following three VM's:

- SA-API - Service Assurance API
- Service Metrics Engine (SME) - Polls for and gathers status and statistics
- NSO-SHIM - Used to communicate with NSO

SA-API

Portal monitor microservice communicates with the SA-API via its exposed REST APIs to request data for display on the portal. Upon receiving a request for data, the SA-API in turn contacts the SME to fetch relevant data.

Service Metrics Engine (SME)

SME collects operational data from NSO via its NETCONF interface.

NSO-SHIM

NSO-SHIM resides between NSO and SME and provides a view of the services provisioned by NSO and configures SME for service chain monitoring.

Automatic Synchronization between NSO-SHIM and SME

NSO-SHIM will attempt to connect to the NSO NETCONF interface every 60 seconds. Once a connection to NSO is established, NSO-SHIM will attempt to connect to SME. When connections to both SME and NSO are established, NSO-SHIM will configure SME to monitor the services that are provisioned by NSO.

Upon receipt of a "cloudvpn-provisioned" notification, NSO-SHIM will call back to NSO to retrieve the service details of the service name given in the notification, and then configure SME to monitor the service.

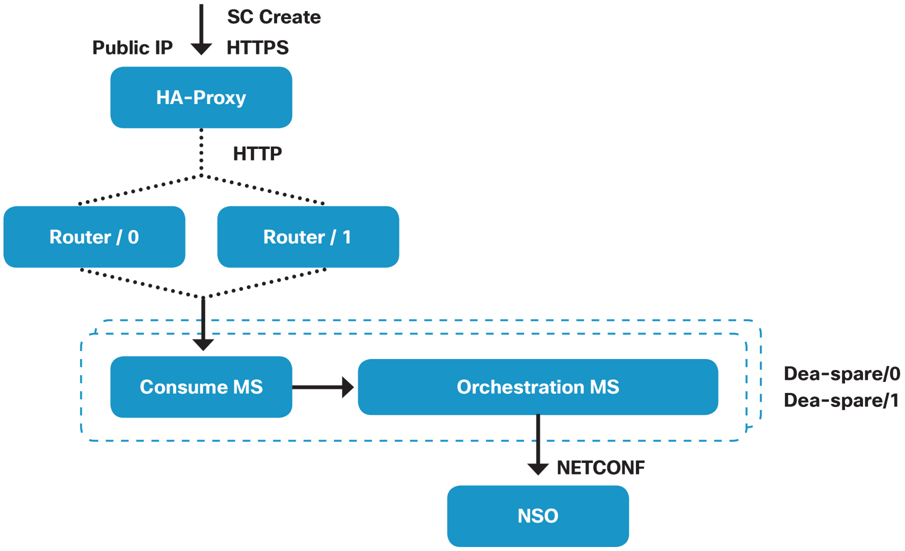
If the NETCONF connection is terminated, NSO-SHIM will attempt to reconnect every 60 seconds. Once the connection is reestablished, NSO-SHIM will perform a full synchronization with SME.

If an SME error prevents a full synchronization of NSO services, NSO-SHIM will back off to prevent overloading NSO with data requests. After several consecutive failures, the time between attempts will be more than one hour. In this case, resolve the issue with SME and then manually synchronize the services with the NSO-SHIM syncSME command.

Call Flows between Portal and Service Assurance

Service Chain Creation via Portal (high-level flow)

Figure 4.3 Service Chain Creation via Portal (high-level flow)



1 When an end customer (via web browser) tries to connect to the vMS Portal, he is actually connecting to the Public IP of the HA-Proxy using HTTPS. **Note**: In this high-level call flow, the intermediary call flows to the ID Management (IDM) subsystem to authenticate the end customer before he can login to the Portal, is not shown.

2 Internally using HTTP since the HA-Proxy terminates the SSL encryption. HA-Proxy sends the HTTP request to the service router, which will route the request to the appropriate Portal microservice. In the example shown above, for a new service chain create, the initial connection will be routed to the Portal Consume microservice

3 Consume microservice will in-turn trigger the Orchestration microservice.

4 Orchestration microservice will communicate with NSO via its NETCONF interface and pass on the request to it.

5 NSO will process the request to create the service chain (more detailed explanation in the Chapter 5 "Network Services Orchestrator").

6 NSO will notify success or failure northbound to orchestration microservice.

Retrieving Service Status and Statistics

When reviewing operational information, flows will differentiate between status and statistics:

- Status is an indication if a Service Chain is up or down.
- Statistics is Service Chain usage data including historical data.

On a high level, status and statistics are handled separately by the Portal and Service Assurance sub-system. The Portal will store status data but not statistics data. The Service Assurance sub-system will store both statistics and status data.

Figure 4.4 Service Assurance - Inter-component Call Flow

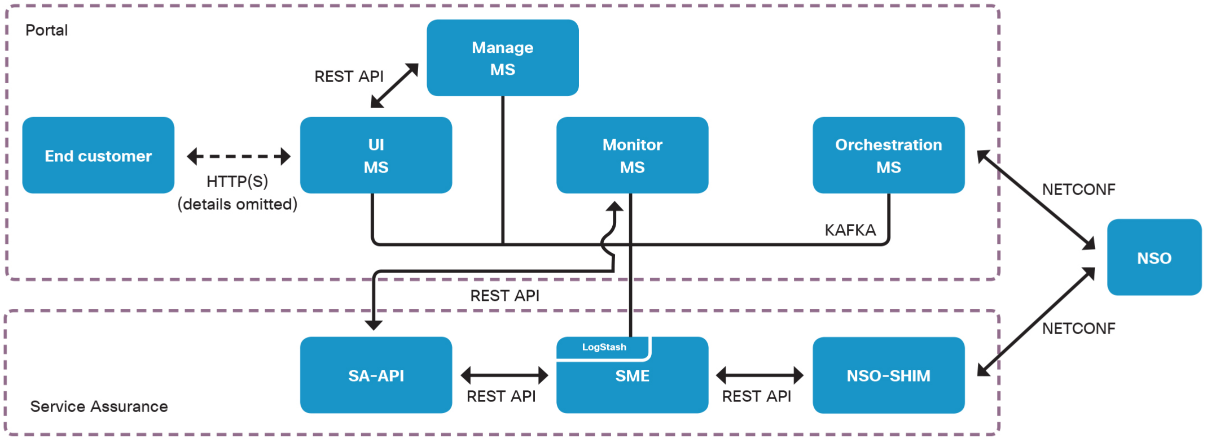
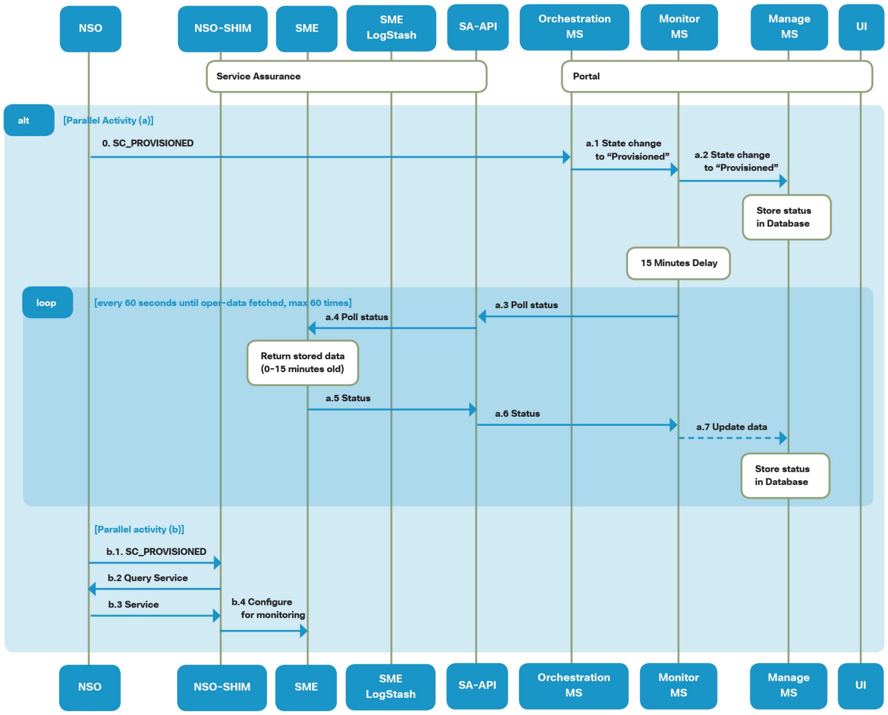


Figure 4.5 Data Collection Call Flow



Call Flow for Starting Data Collection begins with NSO sending notifications northbound to the orchestration microservice and NSO-SHIM in parallel via NETCONF. SME collects operational status and statistics from NSO every 15 minutes.

Orchestration microservice:

1. Orchestration microservice notifies monitor microservice of the change in service chain status from provisioning to provisioned through KAFKA distributed messaging bus.
2. Monitor service updates manage microservice of change in status from provisioning to provisioned for the Service Chain through KAFKA.
3. After a configured SME polling delay timer (default 15 mins), monitor microservice queries SA-API for operational status using REST API calls.
4. SA-API in turn queries SME for operational status using REST API calls.
5. SME returns stored data (0-15 minutes old) to SA-API with operational status through REST API calls.
6. SA-API updates monitor microservice with operational status through REST API.
7. Monitor microservice updates manage microservice with operational status through KAFKA.

Steps 3-7 repeats every 60 seconds until operational status and statistics are obtained. This process may not but not to exceed one hour.

NSO-SHIM:

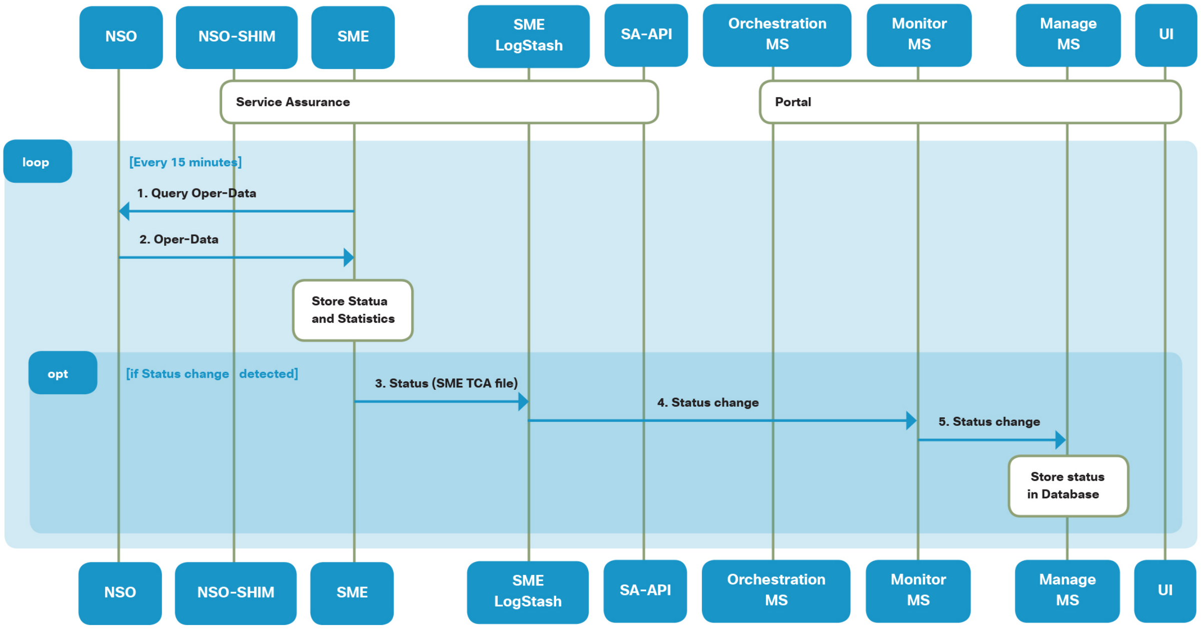
1 NSO sends provisioned notifications to NSO-SHIM through NETCONF.

2 NSO-SHIM queries NSO for topology (service and device information).

3 NSO updates NSO-SHIM with topology information.

4 NSO-SHIM then configures SME to monitor the service.

Figure 4.6 Periodic Status and Statistics Collection Call Flow



SME collects operational status and statistics from NSO every 15 minutes.

1 SME queries NSO for operational status and statistics via NETCONF.

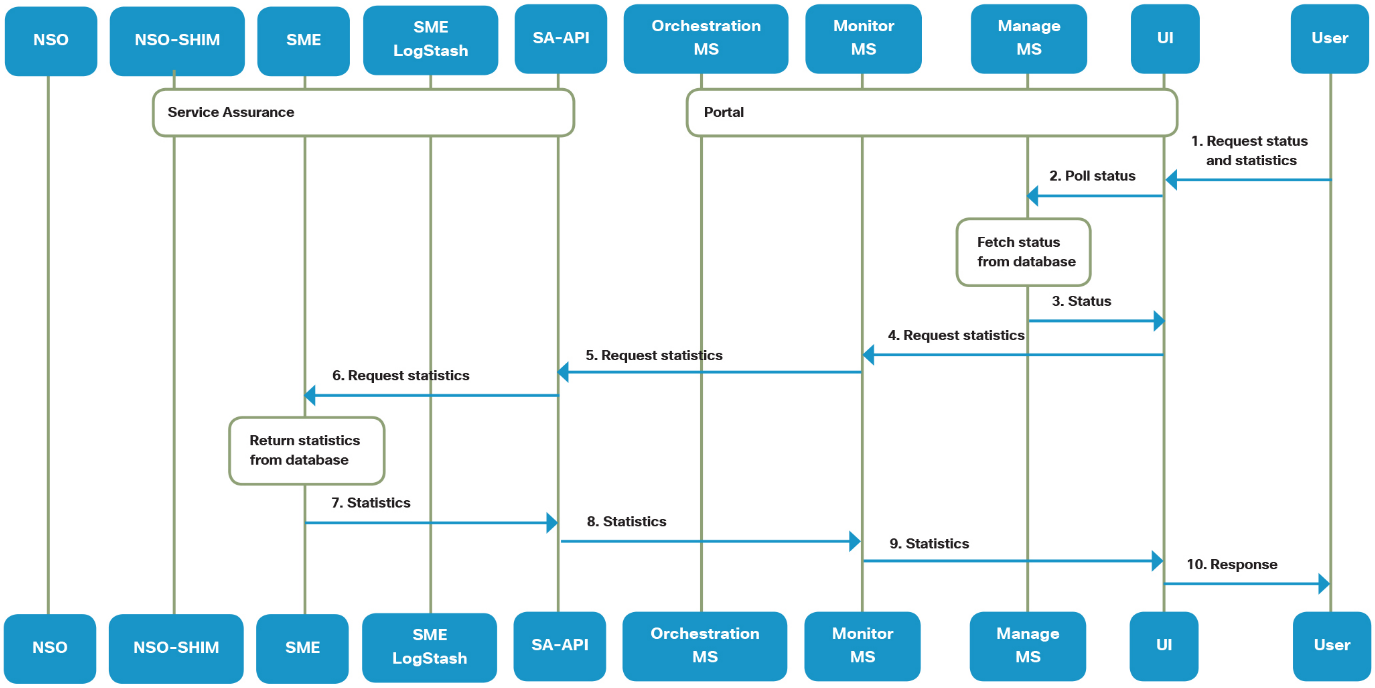
2 NSO responds with operational status and statistics via NETCONF, SME stores operational status and statistics locally

3 In the case of status change for devices or service chains, SME process updates the Threshold Crossing Alarm (TCA) log, which is detected by Logstash.

4 SME Logstash puts the message on KAFKA which is read by monitor microservice.

5 Monitor microservice updates manage microservice with change in status through KAFKA. Monitor microservice will store operational status only in its database.

Figure 4.7 End customer Status and Statistics Call Flow



1 End user logs into the user portal via the Portal user interface (HTTPS).

2 Portal UI initiates queries to manage microservice database for operational status through REST API.

3 Manage microservice returns operational status as stored in the database to portal UI.

4 Portal UI initiates queries to monitor microservice for statistics through REST API, passing the period for which it wants the data.

5 Monitor microservice queries SA-API requesting for statistics through REST API.

6 SA-API queries SME for operational status and statistics through REST API.

7 SME returns stored data (0-15 minutes old) to SA-API with statistics through REST API.

8 SA-API replies to monitor microservice with statistics through REST API.

9 Monitor microservice updates portal UI with statistics through REST API.

10 Portal UI reads manage microservice database and displays operational status and statistics.

Note: If the user queries for statistics and no data is available, user will not see any graphs.

4.3 Service Provider Operations

The Portal, SA and IDM components are made up of several different VMs and containers. Instead of exposing all hosts to the public Internet, the Inception VM is used as the jumphost to access CF and all microservices.

Connecting to CF VMs Using Bosh SSH

```
  ubuntu@vms-inception-p0:~$ sudo su -
  root@vms-inception-p0:/home/ubuntu# bosh ssh
  1. nfs/0
  2. cloud_controller/0
  3. loggregator/0
  4. loggregator_trafficcontroller/0
  5. api_worker/0
  6. dea-spare/0
  7. dea-spare/1
  8. router/0
  9. router/1
  10. haproxy/0
  11. cassandra_seed/0
  12. cassandra/0
  13. redis/0
  14. zookeeper/0
  15. zookeeper/1
  16. zookeeper/2
  17. kafka/0
  18. kafka/1
  19. kafka/2
  Choose an instance: 12
  Acting as user 'admin' on deployment 'cf-p0' on 'Symphony_VMSaaS_P0B'
  Enter password (use it to sudo on remote host): /usr/local/rvm/gems/ruby-1.9.3-
p551/gems/highline-1.6.21/lib/highline/system_extensions.rb:210: warning: Insecure world
writable dir /opt/cisco/vms-installer/tools in PATH, mode 040777
   (ENTER ADMIN PASSWORD HERE)
```

The admin password can be found in `/opt/cisco/vms-installer/scripts/script-constants` next to the COMMON_PASSWORD keyword. After login in, execute `sudo su -i` to gain root privileges.

```
BOSH_USERNAME=admin
COMMON_PASSWORD=PaSSW0Rd
```

Using SSH Keypair to Connect to the VMs in a Deployment

The keypair that is used to create all the VMs in the deployment can be located under `/opt/cisco/vms-installer/tenant-{SUBDOMAIN}/ssh` where "{SUBDOMAIN}" is a property that is set at install time in `/opt/cisco/vms-installer/conf/deploy.properties`.

VMs can be connected to via ssh and passing in the keypair.

To connect to Cloud Foundry VM `ssh -i <path to key file> vcap@<IP address>`

To connect to non-Cloud Foundry VMs like openam, opendj, nsoshim, gunicon and sme, `ssh -i <path to key file> cloud-user@<IP address>`

Microservice JARs and Configuration Files

Portal microservices run as Docker containers on a pair of Droplet Execution Agent VMs (dea-spare/0 and dea-spare/1). At any given time, a Portal microservice is running on only one of the DEA VMs.

The individual microservice JARs and the corresponding configuration (manifest) files are located in the `/opt/cisco/vms-installer/apps/skyfall/<microservice_name>` directory. Each microservice has a dedicated directory.

```
  root@vms-inception:/opt/cisco/vms-installer/apps/skyfall# ls
  administration  consume  idm  logconsumer  manage  monitor  notification  orchestration
skyfallui
```

Inspecting Portal Microservice Status

To view the status of each of the portal micro-services, execute the following on Inception VM as root user.

```
root@vms-inception-p0:/home/ubuntu# cf apps
Getting apps in org vms / space app as admin...
OK

name                requested state   instances   memory   disk   urls
administrationms    started           1/1         1G       1G     administrationms.vmsp0.io
logconsumerms       started           1/1         1G       1G     logconsumerms.vmsp0.io
assuranceapi        started           1/1         3G       1G     assuranceapi.vmsp0.io
```

```
  consumems          started          1/1         1G        1G      consumems.vmsp0.io
  managems           started          1/1         2G        1G      managems.vmsp0.io
  orchestrationms    started          1/1         2G        1G      orchestrationms.vmsp0.io
  monitorms          started          1/1         2G        1G      monitorms.vmsp0.io
  Skyfallui          started          1/1         2G        2G      skyfallui.vmsp0.io
  idmms              started          1/1         1G        1G      idmms.vmsp0.io
  notificationms     started          1/1         1G        1G      notificationms.vmsp0.io
```

To monitor the events of any one of the microservices, execute the `cf events <microservice_name>` command. The example below shows how to view events associated with the "consume" microservice. This is also a helpful command to monitor if at any given instant of time the service had crashed.

```
  root@vms-inception-2:/home/ubuntu# cf events consumems
  Getting events for app consumems in org vms / space app as admin...

  time                            event                 actor    description
  2015-12-08T00:33:04.00+0000    audit.app.update      admin    state: STARTED
  2015-12-08T00:32:57.00+0000    audit.app.update      admin    state: STOPPED
  2015-12-08T00:32:48.00+0000    audit.app.update      admin    instances: 1, memory: 1024,
environment_json: PRIVATE DATA HIDDEN
  2015-11-18T18:25:14.00+0000    audit.app.update      admin    state: STARTED
  2015-11-18T18:25:01.00+0000    audit.app.update      admin
  2015-11-18T18:24:59.00+0000    audit.app.map-route   admin
  2015-11-18T18:24:58.00+0000    audit.app.create      admin    instances: 1, memory: 1024,
state: STOPPED, environment_json: PRIVATE DATA HIDDEN
```

Portal Microservice Control

Stopping Portal Microservice

In order to restart any of the microservices, navigate to that particular directory where the microservice JAR and manifest is located `/opt/cisco/vms-installer/apps/skyfall/<microservice_name>` and execute a `cf stop <microservice_name>` command.

Starting Portal Microservice

In order to restart any of the microservices, navigate to that particular directory where the microservice JAR and manifest is located `/opt/cisco/vms-installer/apps/skyfall/<microservice_name>` and execute a `cf start <microservice_name>` command.

Restarting Portal Microservice

In order to restart any of the microservices, navigate to that particular directory where the microservice JAR and manifest is located `/opt/cisco/vms-installer/apps/skyfall/<microservice_name>` and execute a `cf restart <microservice_name>` command.

Applying New Configuration Changes to Portal Microservice

In case of any modifications to configurations for any of the microservice manifest files, the corresponding microservice will need to be deleted and redeployed. This can be achieved via the following commands:

```
cd /opt/cisco/vms-installer/apps/skyfall/<microservice_name>
cf delete <microservice_name>
cf push -p <microservice_jar_filename>
```

Portal Logs

Portal logs can be inspected using the following command:

```
root@vms-inception-p0:/home/ubuntu# cf files logconsumerms
Getting files for app logconsumerms in org vms / space app as admin...
OK

.bash_logout                              220B
.bashrc                                   3.6K
.profile                                  675B
app/                                         -
logs/                                        -
run.pid                                     3B
server.log                                1.8M
server.log.1.log.zip                    907.7K
server.log.2.log.zip                      1.3M
server.log.3.log.zip                      1.4M
server.log.4.log.zip                      1.4M
server.log.5.log.zip                      1.4M
server.log.6.log.zip                      1.3M
staging_info.yml                          619B
tmp/
```

All Portal microservice logs are aggregated in the `server.log` file.

Portal logs can be viewed as follows:

```
  root@vms-inception:/home/ubuntu# cf files logconsumerms server.log | more
  Getting files for app logconsumerms in org vms / space app as admin...
  OK

  2015-12-08 00:37:27 [task-scheduler-1] INFO c.c.s.o.c.NCSSubscribeTimer::getNCSSession - NCS
Connection Config :  Address:10.32.1.24, Port:830, LocalUser:admin, RemoteUser:admin, stream:
ncs-alarms
  2015-12-08 00:37:27 [task-scheduler-1] INFO c.c.s.o.c.NCSSubscribeTimer::getNCSSession - NCS
Connection Config :  Address:10.32.1.24, Port:830, LocalUser:admin, RemoteUser:admin, stream:
pnpnotif
  2015-12-08 00:37:28 [task-scheduler-2] INFO c.c.s.o.c.NCSSubscribeTimer::getNCSSession - NCS
Connection Config :  Address:10.32.1.24, Port:830, LocalUser:admin, RemoteUser:admin, stream:
cloudvpn-notif
  2015-12-08 00:37:28 [task-scheduler-1] INFO c.c.s.o.c.NCSSubscribeTimer::getNCSSession - NCS
Session : config:d07ce06c4f704b5da7ac1c5fb5f4c644 stream: ncs-alarms
  2015-12-08 00:37:28 [task-scheduler-3] INFO c.c.s.o.c.NCSSubscribeTimer::getNCSSession - NCS
Session : config:bfbc007ee79449d1a9269c67f9fae854 stream: pnpnotif
  2015-12-08 00:37:28 [task-scheduler-2] INFO c.c.s.o.c.NCSSubscribeTimer::run - >>>>
Successfully subscribed to stream : pnpnotif
  2015-12-08 00:37:29 [task-scheduler-4] INFO c.c.s.o.c.NCSSubscribeTimer::run - >>>>
Successfully subscribed to stream : ncs-alarms
  2015-12-08 00:37:29 [task-scheduler-1] INFO c.c.s.o.c.NCSSubscribeTimer::getNCSSession - NCS
Session : config:59710f65da4d4a7c8e70c8d89a92e20d stream: cloudvpn-notif
  2015-12-08 00:37:29 [task-scheduler-5] INFO c.c.s.o.c.NCSSubscribeTimer::run - .....Recieve
notification on stream....pnpnotif:  Address:10.32.1.24, Port:830, LocalUser:admin,
RemoteUser:admin
  2015-12-08 00:37:29 [task-scheduler-3] INFO c.c.s.o.c.NCSSubscribeTimer::run - .....Recieve
notification on stream....ncs-alarms:  Address:10.32.1.24, Port:830, LocalUser:admin,
RemoteUser:admin
  2015-12-08 00:37:29 [task-scheduler-3] INFO c.c.s.o.c.NCSSubscribeTimer::run - >>>>
Successfully subscribed to stream : cloudvpn-notif
  2015-12-08 00:37:30 [task-scheduler-3] INFO c.c.s.o.c.NCSSubscribeTimer::run - .....Recieve
notification on stream....cloudvpn-notif:  Address:10.32.1.24, Port:830, LocalUser:admin,
RemoteUser:admin
  2015-12-08 00:37:30 [task-scheduler-3] INFO c.c.s.a.w.c.ComponentVersionListener::execute -
Received message from BUILD_INFO_TOPIC is{"component":"Logconsumer
Microservice","version":"2.1.0","buildNumbe
  r":"4264","buildDateTime":"12/07/2015 05:44:06"}
  2015-12-08 00:37:30 [task-scheduler-3] INFO
c.c.s.a.s.i.ComponentVersionService::updateComponentVersion - updateComponentVersion - Version:
2.1.0, BuildDateTime: 12/07/2015 05:44:06, Component: Logconsu
  mer Microservice
  2015-12-08 00:37:30 [task-scheduler-3] INFO
```

```
c.c.s.a.r.i.ComponentVersionRepository::updateComponentVersion - updateComponentVersion()
  2015-12-08 00:38:17 [task-scheduler-7] INFO
c.c.s.a.c.CacheNotificationConfig::updateRedisCache -
{"eventName":"FORGOT_PASSWORD_ENDUSER","notificationType":"notificationType1","eventDisplayName
":"Reset
```

This includes any errors while communicating with NSO, including any errors during Service Chain deployment. NSO sends notifications northbound to the Portal orchestration microservice using two notification streams: pnpnotif (for devices status changes) and cloudvpn-notif (for service status changes).

SA-API Logs

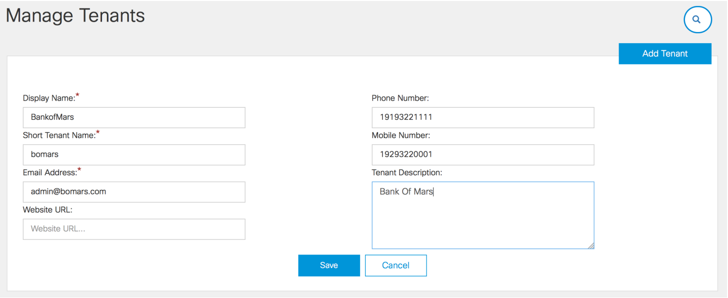
To view the SA-API logs:

```
  root@vms-inception:~# cf files assuranceapi app/.java-buildpack/tomcat/logs/AssuranceAPI.log
```

Creating a Tenant in Portal

Login to the portal as an Admin user. On the left-hand pane click on "Tenants". On the "Manage Tenants" pane on the right click on "Add Tenant". Fill in all the required details and click Save. Make a note of the Tenant's "Short Name" as this will be needed in the next step when associating users with this particular tenant.

Figure 4.8 Create Tenant in Portal



Manage Tenants

Display Name:* BankofMars

Short Tenant Name:* bomars

Email Address:* admin@bomars.com

Website URL: Website URL...

Phone Number: 19193221111

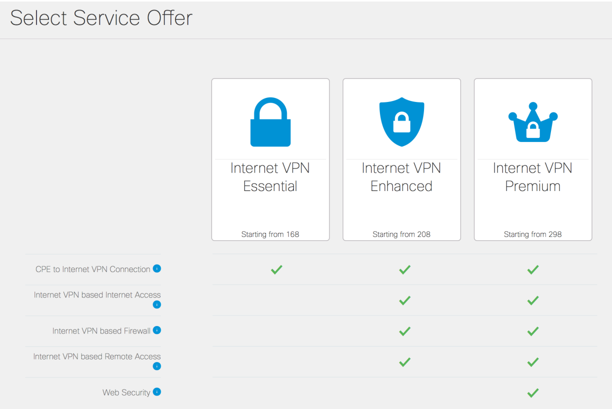
Mobile Number: 19293220001

Tenant Description: Bank Of Mars

Ordering a Service Chain via Portal

Login to the Portal as a Tenant Admin. Navigate to "Shop" and click on "Internet VPN". On the "Select Service Offering" page select the service to be purchased.

Figure 4.9 Selecting Service Bundle



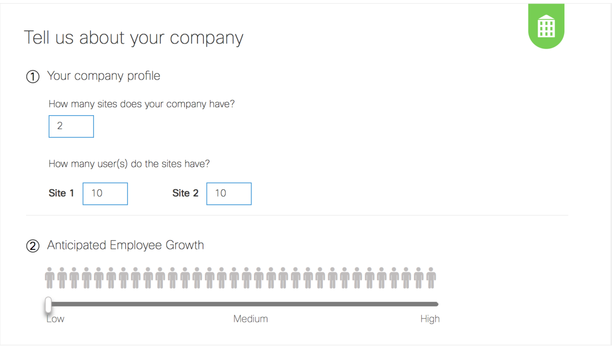
Select Service Offer

	Internet VPN Essential Starting from 168	Internet VPN Enhanced Starting from 208	Internet VPN Premium Starting from 298
CPE to Internet VPN Connection	✓	✓	✓
Internet VPN based Internet Access		✓	✓
Internet VPN based Firewall		✓	✓
Internet VPN based Remote Access		✓	✓
Web Security			✓

In this example, the Medium Service Chain has been selected. In the Portal, this corresponds to the "Internet VPN Enhanced" service offering. When the service flavor is clicked on, a new page opens to fill in details needed to customize the service to be purchased.

Enter the number of branch sites being requested and the number of user(s) at each branch site.

Figure 4.10 Sites associated with Service Chain



Tell us about your company

① Your company profile

How many sites does your company have?

2

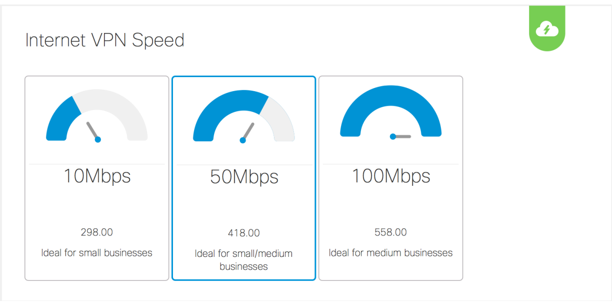
How many user(s) do the sites have?

Site 1 10 Site 2 10

② Anticipated Employee Growth

Select the Internet VPN Speed being requested.

Figure 4.11 Selecting Service Chain Throughput



Internet VPN Speed

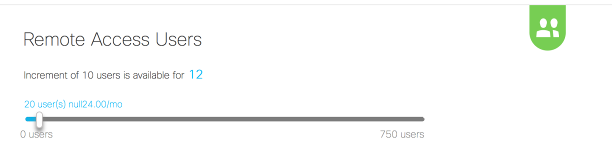
10Mbps
298.00
Ideal for small businesses

50Mbps
418.00
Ideal for small/medium businesses

100Mbps
558.00
Ideal for medium businesses

Select the number of Remote-Access VPN Users being requested.

Figure 4.12 Selecting Number of Remote Access VPN Users



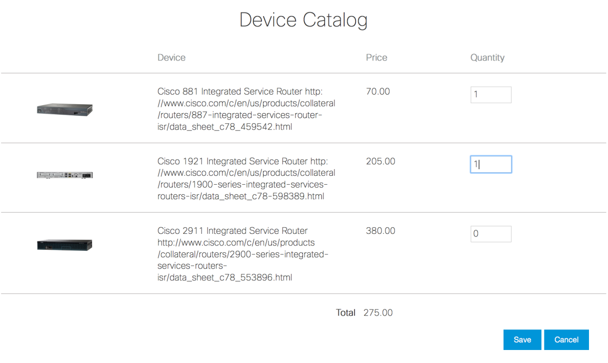
Remote Access Users

Increment of 10 users is available for 12

20 user(s) null24.00/mo

Select the number and type of each CPE being requested and click "Save".

Figure 4.13 Number and Type of CPE being Ordered



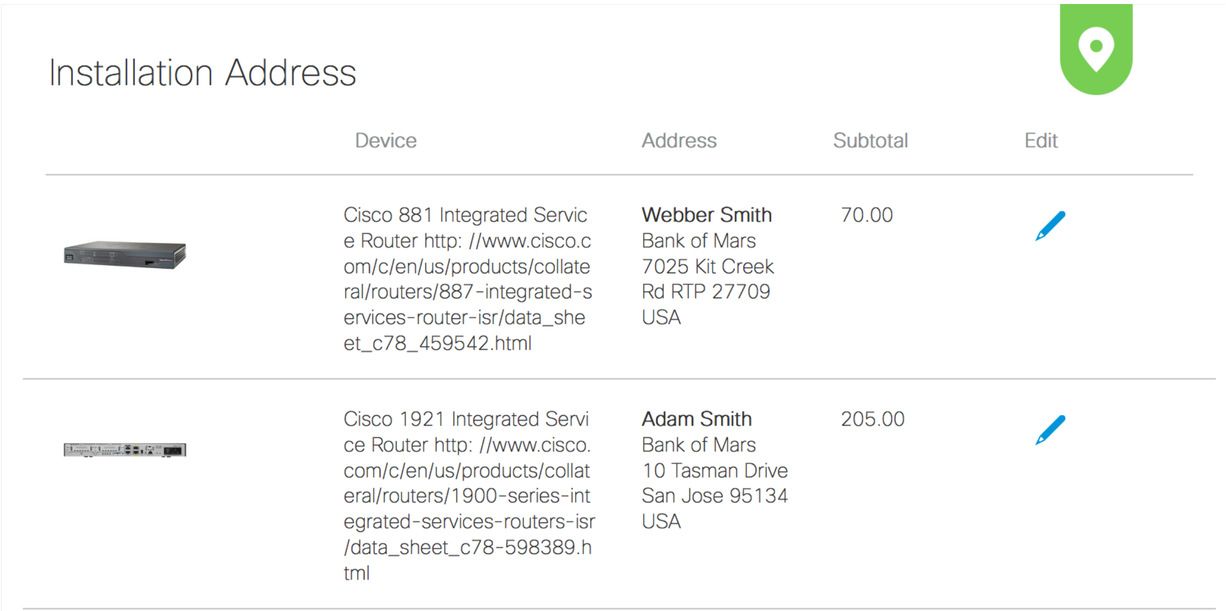
Device Catalog

	Device	Price	Quantity
	Cisco 881 Integrated Service Router http://www.cisco.com/c/en/us/products/collateral/routers/887-integrated-services-router-isr/data_sheet_c78_459542.html	70.00	1
	Cisco 1921 Integrated Service Router http://www.cisco.com/c/en/us/products/collateral/routers/1900-series-integrated-services-routers-isr/data_sheet_c78-598389.html	205.00	1
	Cisco 2911 Integrated Service Router http://www.cisco.com/c/en/us/products/collateral/routers/2900-series-integrated-services-routers-isr/data_sheet_c78_553896.html	380.00	0

Total 275.00

Provide the installation address for each of the branch sites where the CPEs need to be shipped to. Click on "Add new Address" next to each of the CPEs and enter an address.

Figure 4.14 CPE Shipping Address

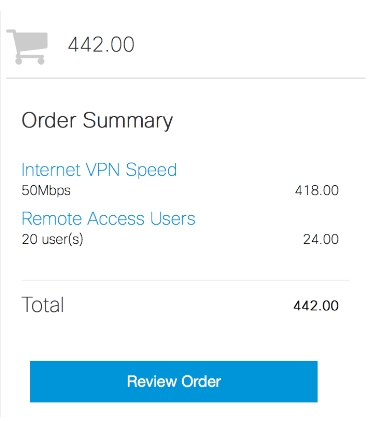


Installation Address

	Device	Address	Subtotal	Edit
	Cisco 881 Integrated Service Router http: //www.cisco.com/c/en/us/products/collateral/routers/887-integrated-services-router-isr/data_sheet_c78_459542.html	Webber Smith Bank of Mars 7025 Kit Creek Rd RTP 27709 USA	70.00	✎
	Cisco 1921 Integrated Service Router http: //www.cisco.com/c/en/us/products/collateral/routers/1900-series-integrated-services-routers-isr/data_sheet_c78-598389.html	Adam Smith Bank of Mars 10 Tasman Drive San Jose 95134 USA	205.00	✎

On the right-hand pane look at the Order Summary and click on "Review".

Figure 4.15 Order Summary



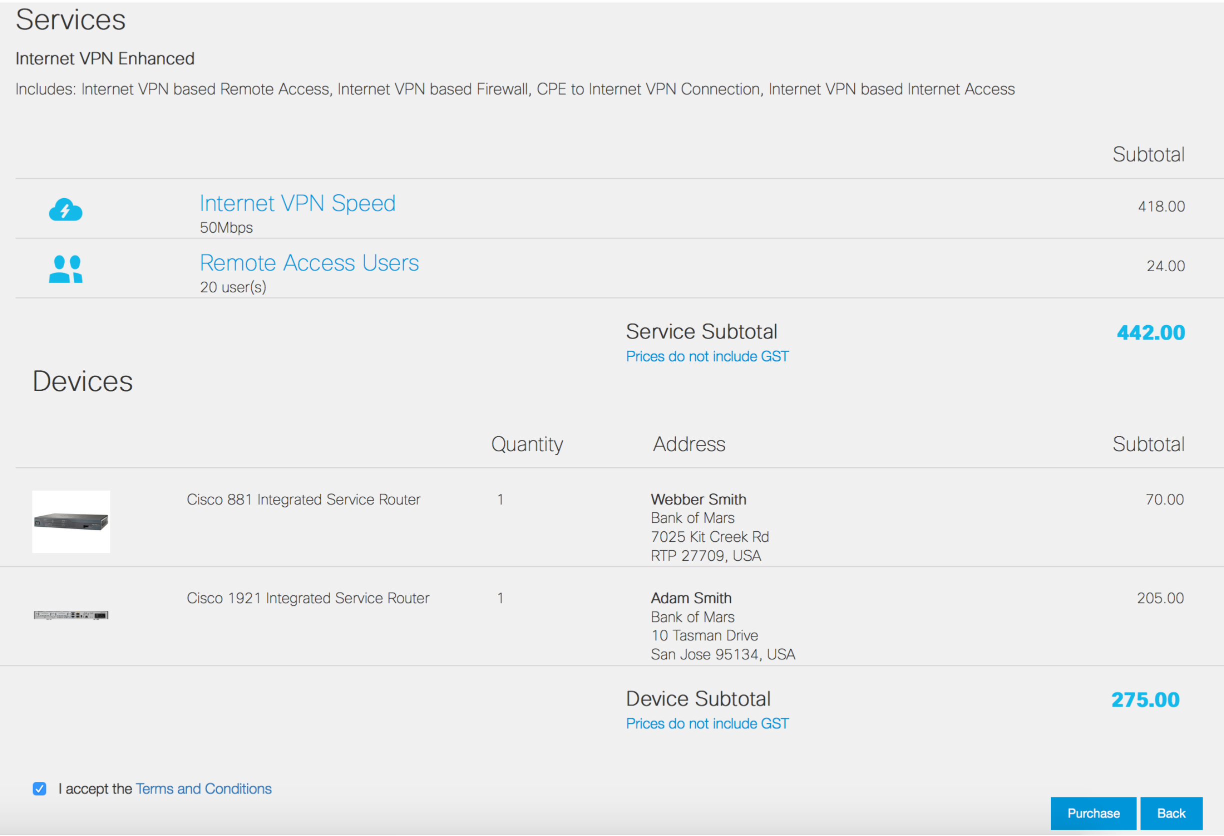
442.00

Order Summary

Internet VPN Speed 50Mbps	418.00
Remote Access Users 20 user(s)	24.00
Total	442.00

A new page is presented with the entire Order Summary for the Service Chain and the CPEs. Read through the "Terms and Conditions" and click "Accept".

Figure 4.16 Review Order and Purchase



Services

Internet VPN Enhanced

Includes: Internet VPN based Remote Access, Internet VPN based Firewall, CPE to Internet VPN Connection, Internet VPN based Internet Access

		Subtotal
	Internet VPN Speed 50Mbps	418.00
	Remote Access Users 20 user(s)	24.00

Service Subtotal 442.00
Prices do not include GST

Devices

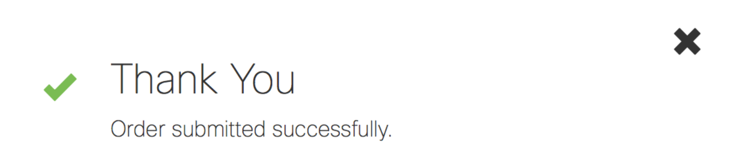
		Quantity	Address	Subtotal
	Cisco 881 Integrated Service Router	1	Webber Smith Bank of Mars 7025 Kit Creek Rd RTP 27709, USA	70.00
	Cisco 1921 Integrated Service Router	1	Adam Smith Bank of Mars 10 Tasman Drive San Jose 95134, USA	205.00

Device Subtotal 275.00
Prices do not include GST

☑ I accept the Terms and Conditions

Once reviewed, click "Purchase" A confirmation window will pop up.

Figure 4.17 Order Confirmation



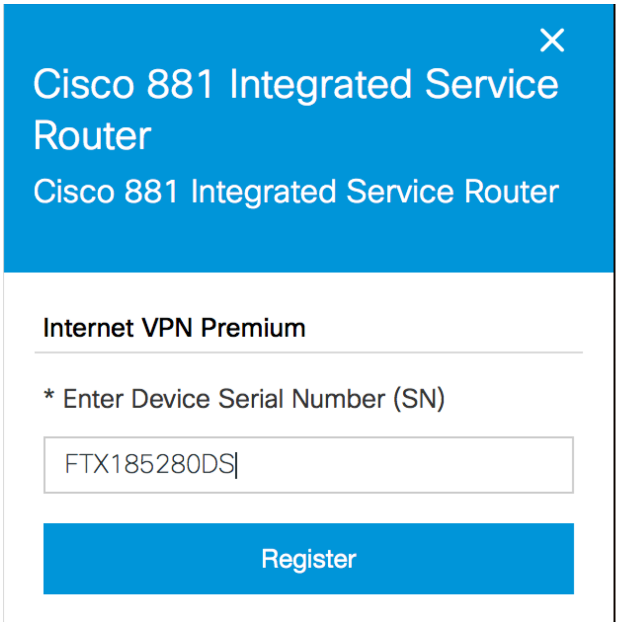
Thank You

Order submitted successfully.

Close the confirmation pop-up window. Navigate to the Devices tab on the left-hand pane.

Click on each of the "UNREGISTERED" devices, add the CPE S/N on the right-hand pane and click on "Register".

Figure 4.18 On-boarding CPE



Cisco 881 Integrated Service Router

Cisco 881 Integrated Service Router

Internet VPN Premium

* Enter Device Serial Number (SN)

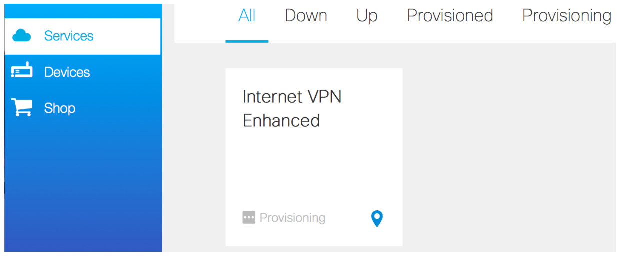
FTX185280DS

Once the Service Chain is alive and all of the CPEs have been successfully on-boarded, the Service and Devices should be online, and the entire service should be up and running.

Service Chain Status State Transition

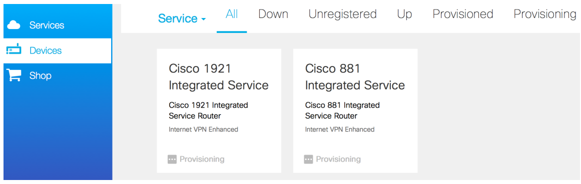
Once a Service Chain is purchased from the Portal, the service status is set to "Provisioning".

Figure 4.19 Service Chain in "Provisioning" state



Similarly, once the CPE devices are on-boarded via the Portal, they are set to the "Provisioning" state.

Figure 4.20 Devices in "Provisioning" state

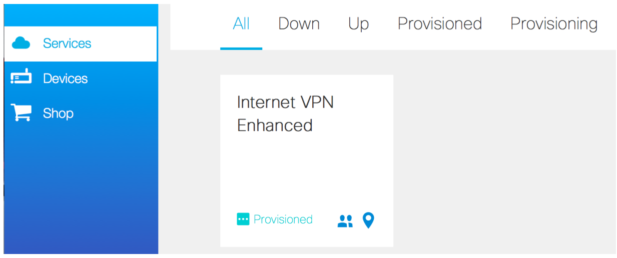


Once the orchestration microservices receives notifications from NSO, it updates the service status to "Provisioned".

The sample log below shows the notification received by the Orchestration microservice from NSO saying that the service has been "Provisioned" successfully.

```
  2015-12-08 23:43:39 [task-scheduler-2] INFO c.c.s.m.d.i.OrchestrationListener::execute -
@@@@@@@@@@@@@@@@@@@@@@@ In side ORCHESTRATION_NCS_RESPONSE_TOPIC :: OrchestrationListener ::
jsonString {"type":"response","messageId":"cns-soltest2b-test1-
cloudvpn","action":"provisionStatus","payload":{"vpnURL":
["https://128.107.38.87"],"time":"2015-12-08T23:43:07.89+00:00","message":"Provisioned
successfully", "status":"success"}}
  2015-12-08 23:43:40 [task-scheduler-2] WARN
c.c.s.m.o.i.OperatorSummaryManagerImpl::updateServiceInstanceSummaryFromStatus - Updating
service status for instance with ID cns-soltest2b-test1-cloudvpn with status Provisioned
  2015-12-08 23:43:40 [task-scheduler-2] WARN c.c.s.m.o.i.ServiceProvisionedState::doPreAction
- Transitioning state for service cns-soltest2b-test1-cloudvpn from PROVISIONING to Provisioned
  2015-12-08 23:43:46 [task-scheduler-2] WARN c.c.s.m.o.i.ServiceProvisionedState::doPreAction
-  Service state successfully transitioned from PROVISIONING to Provisioned for cns-soltest2b-
test1-cloudvpn
  2015-12-08 23:43:46 [task-scheduler-2] WARN
c.c.s.m.o.i.OperatorSummaryManagerImpl::updateServiceInstanceSummaryFromStatus - Successfully
updated service status for instance with ID cns-soltest2b-test1-cloudvpn with status
Provisioned
```

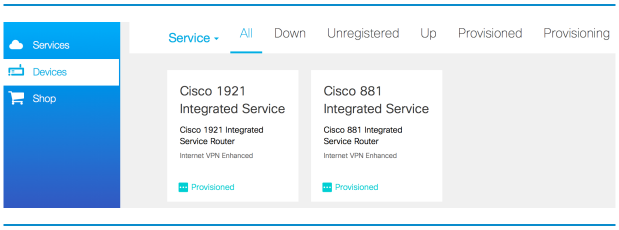
Figure 4.21 Service Chain in "Provisioned" state



Similarly for each of the CPE devices, the Portal receives notifications from NSO stating that the service state has now transitioned to the "Provisioned" state.

```
  2015-12-08 23:41:33 [task-scheduler-9] DEBUG
c.c.s.m.o.d.DeviceInstanceWithStatusRepository::updateOperationalState - Updating operational
state for device instance from deviceInstanceWithStatus table with id CPE-bfdb6585-1966-4d2f-
be8a-3d3bf67b7eb5 with status Provisioned
```

Figure 4.22 Devices in "Provisioned" state



For Medium and Full Service Chains, the NSO also informs the Portal about the VPN URL that needs to be used by Remote Access VPN users. This VPN-URL is displayed on the Portal.

```
  2015-12-08 23:43:39 [task-scheduler-2] INFO c.c.s.o.c.NCSSubscribeTimer::run - == publish
stream notification....{"type":"response","messageId":"cns-soltest2b-test1-
cloudvpn","action":"provisionStatus","payload":{"vpnURL":
["https://128.107.38.87"],"time":"2015-12-08T23:43:07.89+00:00","message":"Provisioned
successfully","status":"success"}}
```

However the Service and Device UP / DOWN status displayed on the Portal comes from the Service Assurance components. About 30minutes after the CPEs are on-boarded, the Device and Service Status should show as "UP" on the Portal. If that is not that case and they show as either stuck in the "PROVISIONED" state or "DOWN" on the Portal, this will need some investigation and troubleshooting.

Figure 4.23 Devices in "UP" state

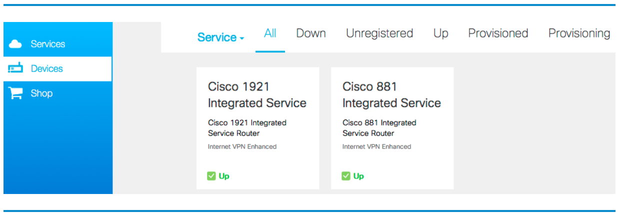
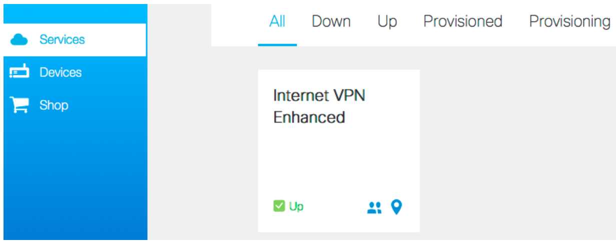

Services
Devices
Shop
Service
All
Down
Unregistered
Up
Provisioned
Provisioning
Cisco 1921 Integrated Service
Cisco 1921 Integrated Service Router
Internet VPN Enhanced
Up
Cisco 881 Integrated Service
Cisco 881 Integrated Service Router
Internet VPN Enhanced
Up

Figure 4.24 Service Chain in "UP" state



Viewing Portal Microservice Versions

When logged in to the Portal as an admin user, click on the Component Versions link on the bottom left to view the currently installed version of all the Portal microservices.

Figure 4.25 Viewing Portal Component Version

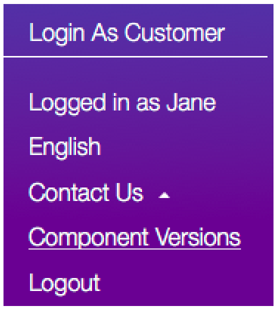
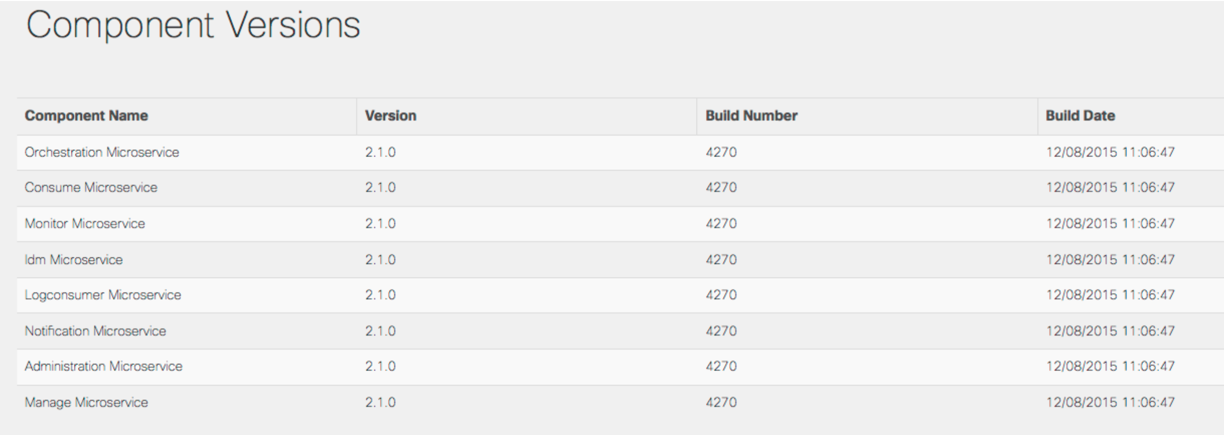


Figure 4.26 Portal Component Versions



Component Versions

Component Name	Version	Build Number	Build Date
Orchestration Microservice	2.1.0	4270	12/08/2015 11:06:47
Consume Microservice	2.1.0	4270	12/08/2015 11:06:47
Monitor Microservice	2.1.0	4270	12/08/2015 11:06:47
Idm Microservice	2.1.0	4270	12/08/2015 11:06:47
Logconsumer Microservice	2.1.0	4270	12/08/2015 11:06:47
Notification Microservice	2.1.0	4270	12/08/2015 11:06:47
Administration Microservice	2.1.0	4270	12/08/2015 11:06:47
Manage Microservice	2.1.0	4270	12/08/2015 11:06:47

Viewing Event Logs

Once logged in to the Portal as a user, click on the Event Logs link on the bottom left to view all the events associated with that particular customer:

Figure 4.27 Viewing Event Logs

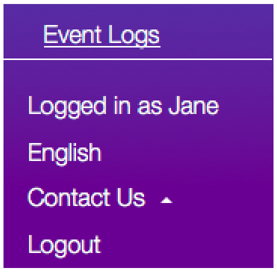
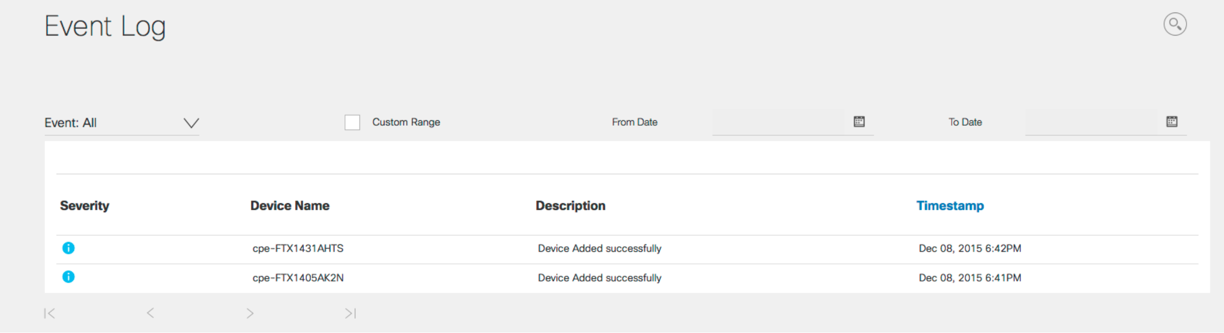


Figure 4.28 Sample Event Log

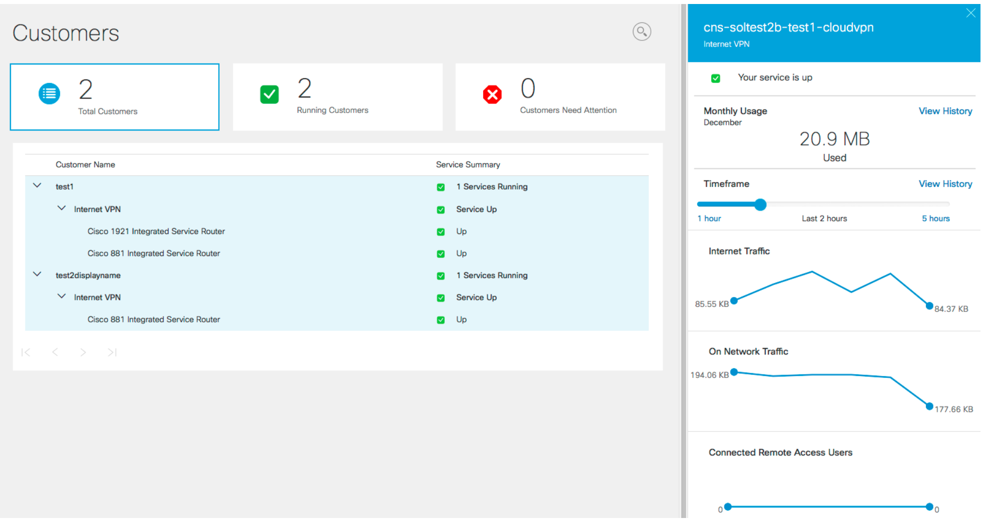


Severity	Device Name	Description	Timestamp
ℹ	cpe-FTX1431AHTS	Device Added successfully	Dec 08, 2015 6:42PM
ℹ	cpe-FTX1405AK2N	Device Added successfully	Dec 08, 2015 6:41PM

Portal Operator View

For day-to-day operations and troubleshooting, log in as an Operator on the portal. Service Chain status, statistics and errors can be viewed.

Figure 4.29 Portal Operator View



Customers

2 Total Customers

2 Running Customers

0 Customers Need Attention

Customer Name	Service Summary
test1	✓ 1 Services Running
Internet VPN	✓ Service Up
Cisco 1921 Integrated Service Router	✓ Up
Cisco 881 Integrated Service Router	✓ Up
test2displayname	✓ 1 Services Running
Internet VPN	✓ Service Up
Cisco 881 Integrated Service Router	✓ Up

cns-soltest2b-test1-cloudvpn

Internet VPN

✓ Your service is up

Monthly Usage
December

20.9 MB
Used

Admin or Operator logging as End Customer

When logged in as an Admin or Operator for operations and troubleshooting, it is useful to have a view of what any customer is seeing. This can be done by clicking on the "Login as Customer" on the lower left. From the drop-down list provided, select the customer to view and the user to login as. Click on "Start". This provides a central location for the Service Provider Admin to login as any customer without having to remember the user login credentials for all customers.

Figure 4.30 Admin or Operator Logging in as Customer

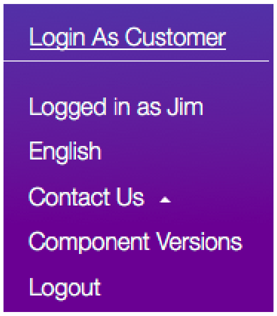
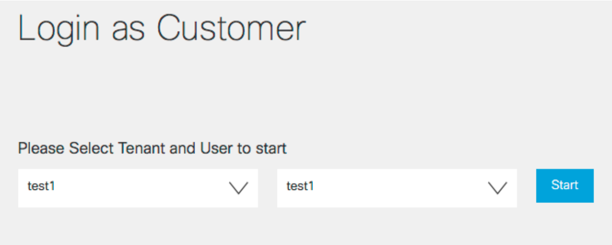


Figure 4.31 Operator selection of Tenant and user

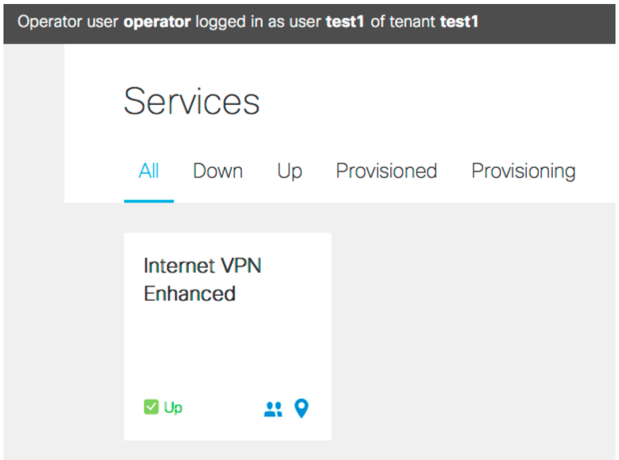


Login as Customer

Please Select Tenant and User to start

test1 test1

Figure 4.32 Operator logged in as Tenant User



Operator user **operator** logged in as user **test1** of tenant **test1**

To return to the Operator view, click "Return to operator view" at the top of the window.

Tenant Users Create, Read, Update and Delete (CRUD) Operations

Create User

To associate users with this particular tenant, CURL commands need to be executed against the IDM API:

```
  curl --header "Content-Type: application/json" --user "admportal:NiceToBe#Admin4Portal" --
data '{"_id":"bomadmin","first_name":"Bom","last_name":"Admin","display_name":"Bom
Admin","email":"admin@bomars.com","password":"ABCD1234","tenant_id":"bomars","phone":"191932211
11","roles":["CONSUMER"]}' http://<IDM_API_IP>/api/v1/users
```

- `--user` needs to be supplied with admportal uid and password.
- `_id` is the user-id that will be used by the user to login to the portal.

- `first_name` refers to the first name of the user.
- `last_name` refers to the last name of the user.
- `display_name` refers to the Display Name of the User.
- `email` refers to the e-mail of the user.
- `password` - specify a strong password for the user
- `tenant_id` needs to match the "Short Name" give while creating the tenant in the portal.
- `phone` - provide phone number of the user
- `roles` - The role needs to be set to "CONSUMER"

Read User

Query for a user using GET

```
  curl --user "admportal:NiceToBe#Admin4Portal" http://<IDM_API_IP>/api/v1/users/bomars
```

Update User

```
  curl --user "admportal:NiceToBe#Admin4Portal" --request PUT --header "Content-Type:
application/json" --data '{ "first_name": "Bom","last_name": "Admin","display_name": "Bom
Admin","email": "Administrator@bomars.com","password": "cUk39cld!%","tenant_id":
"bomars","phone": "123-123-1234","roles": ["Developer", "Designer"]}'
http://10.203.29.254/api/v1/users/bomars
```

Delete User

```
  curl --user "admportal:NiceToBe#Admin4Portal" --request DELETE
http://<IDM_API_IP>/api/v1/users/bomars
```

Updating User Password

```
  curl --header "Content-Type: application/json" --data '{"new_password": "newpassword123"}'
http://<IDM_API_IP>/api/v1/update_password/bomars
```

NSO-SHIM CLI Usage

```
  Usage:
         nso-shim {start | stop | restart | status}
           nso-shim {get services [service] | search [device]]}
           nso-shim {get devices [device]}
```

```
       nso-shim {syncSME [service]}
       nso-shim {security ssl [enable | disable | status]
       nso-shim {security httpauth [enable | disable | status]
Examples:
nso-shim get services
nso-shim get services Telco-ACME-VIPS-Medium
nso-shim get services search cpe-CPE_Telco_ACME_V_2
nso-shim get devices
nso-shim get devices cpe-CPE_Telco_ACME_V_2
nso-shim syncSME
nso-shim syncSME Telco-ACME-VIPS-Medium
(hidden option) -d enables debug mode
nso-shim -d start
nso-shim -d get services
```

SME Operations

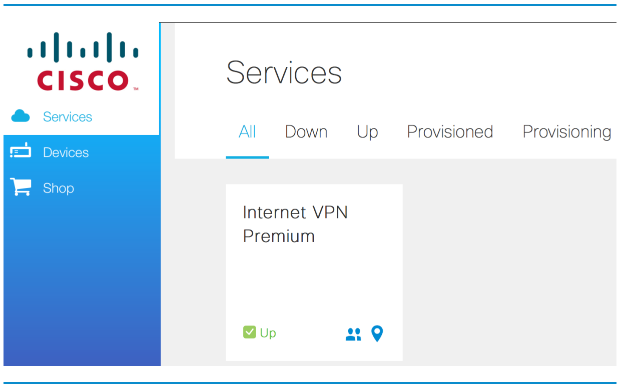
SME process can be started, stopped, restarted and status can be checked using the following commands:

```
/opt/CSCOppm-gw/bin/ppm status
/opt/CSCOppm-gw/bin/ppm start
/opt/CSCOppm-gw/bin/ppm stop
/opt/CSCOppm-gw/bin/ppm restart
```

4.4 End Customer Operations

When an end customer logs into the CloudVPN Portal, they are presented with a layout similar to that below. The three menu options on the left-hand side are Services, Devices, and Shop. The existing service that the end customer is subscribed to is represented by the box "Internet VPN Premium".

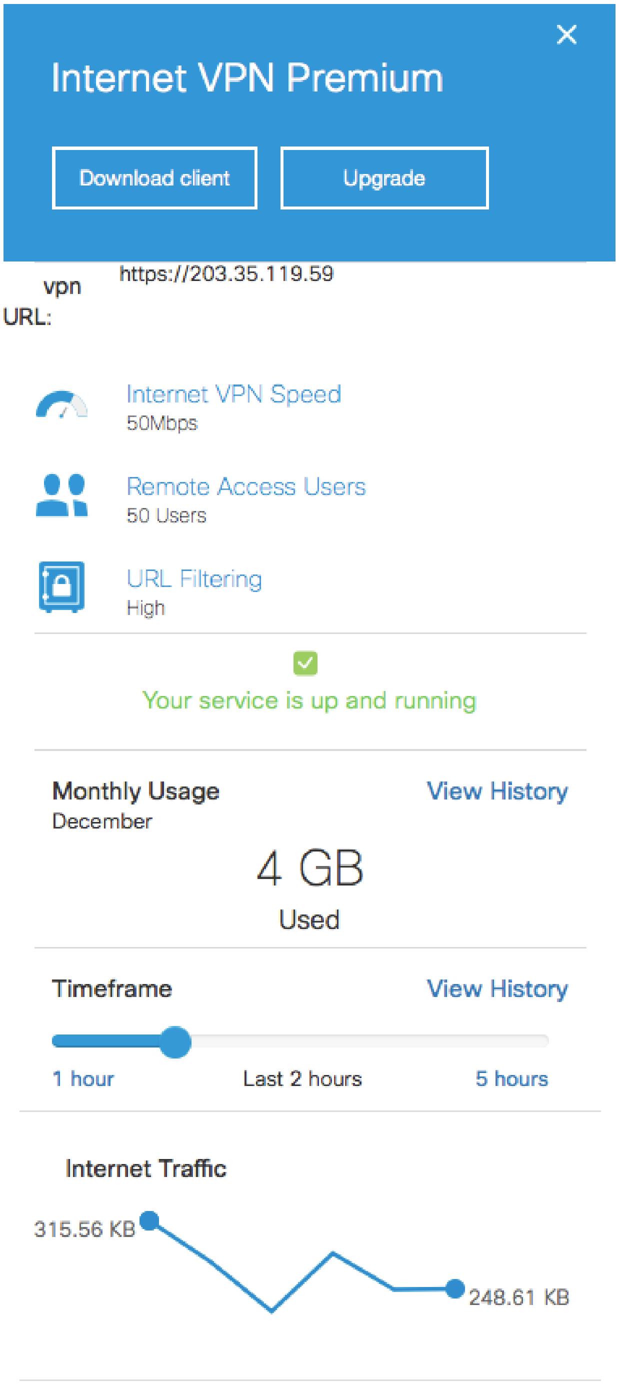
Figure 4.33 Customer Initial Portal View



QoS

To change QoS settings, an end customer needs to click on the box "Internet VPN Premium". On the right-hand side of the Web browser a new box will appear similar to this:

Figure 4.34 Existing Service Options



Internet VPN Premium

vpn URL: https://203.35.119.59

Internet VPN Speed
50Mbps

Remote Access Users
50 Users

URL Filtering
High

Your service is up and running

Monthly Usage
December

4 GB
Used

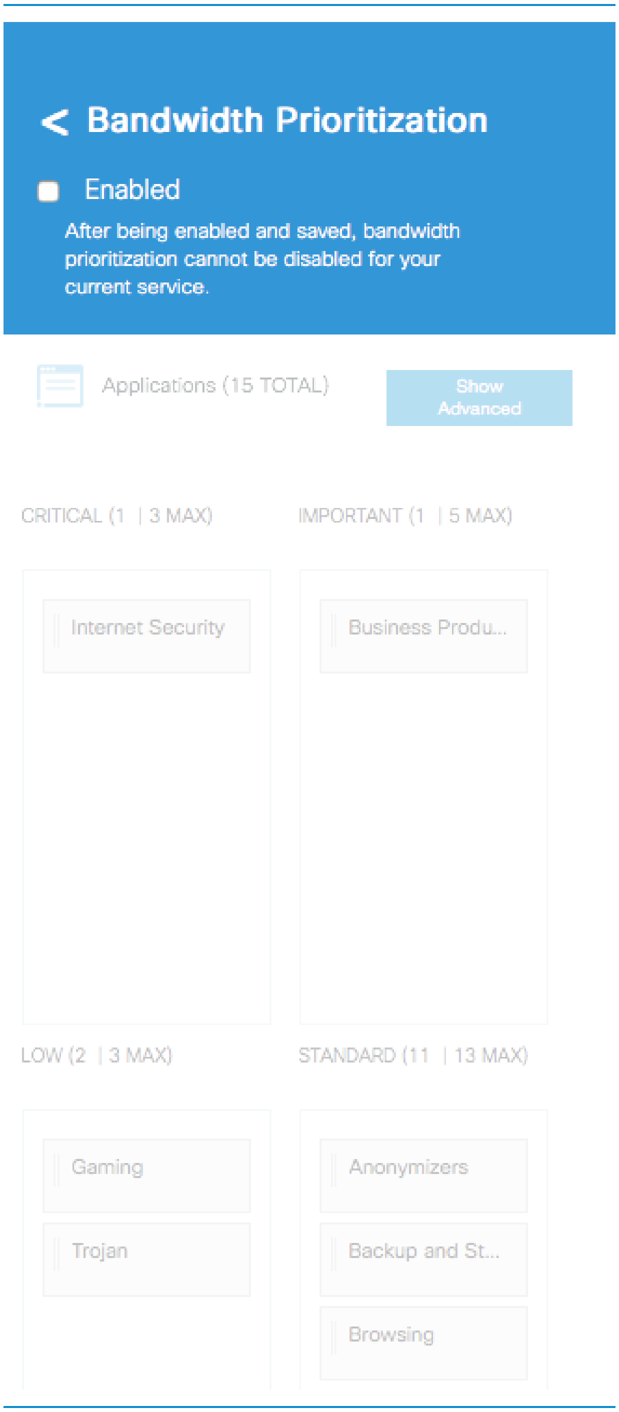
Timeframe

Last 2 hours

Internet Traffic

The box will show the configured service options, the service status, monthly usage, and traffic graphs. If the user scrolls further down in this box and clicks on the edit (pencil) icon of the "Bandwidth Prioritization" section, the vMS Portal will display the "Bandwidth Prioritization" menu. The options will be grayed out until the "Bandwidth Prioritization" is enabled by clicking on the checkbox.

Figure 4.35 Enabling Bandwidth Prioritization



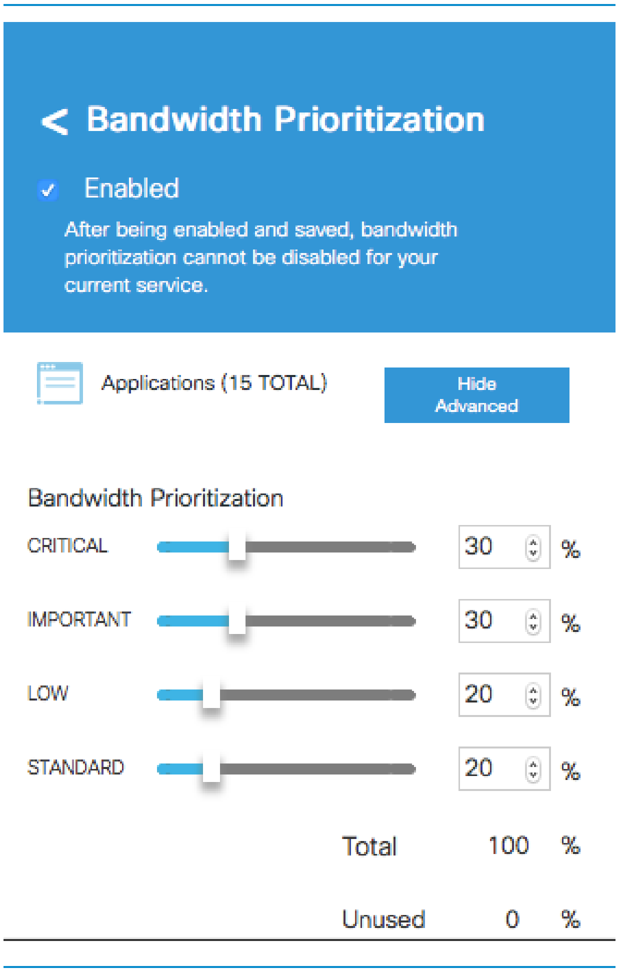
< Bandwidth Prioritization

□ Enabled

After being enabled and saved, bandwidth prioritization cannot be disabled for your current service.

With Bandwidth Prioritization enabled, the customer will now have the option to drag and drop pre-configured traffic classification classes into the queue priority of their choosing. There is a "Save" button here to save the selections. Additionally, there is an "Advanced Bandwidth Prioritization" option that will open an additional window to adjust the queue percentages as seen below.

Figure 4.36 Configuring Bandwidth Prioritization



< Bandwidth Prioritization

☑ Enabled

After being enabled and saved, bandwidth prioritization cannot be disabled for your current service.

Applications (15 TOTAL)

Bandwidth Prioritization

CRITICAL 30 %

IMPORTANT 30 %

LOW 20 %

STANDARD 20 %

Total 100 %

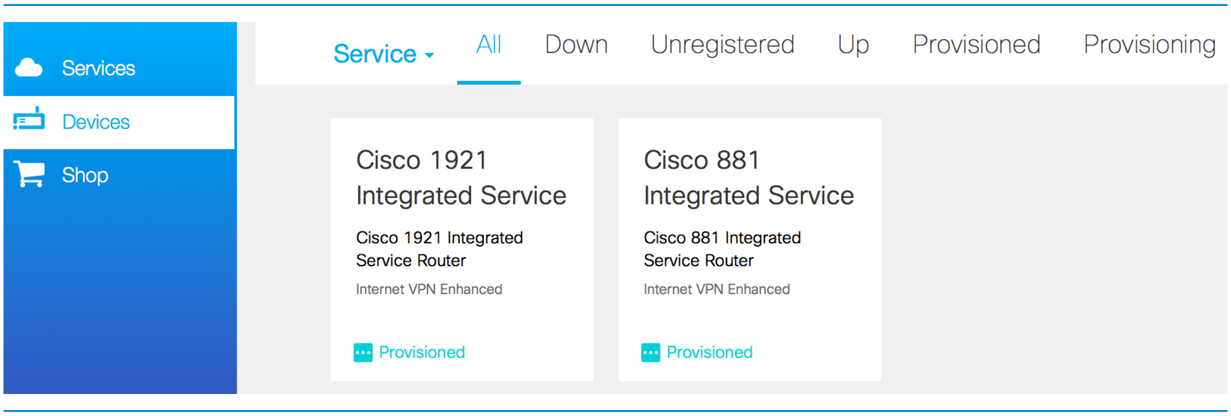
Unused 0 %

The end customer can either use the slider bar to adjust the bandwidth percentage per queue or manually enter a value. The end total cannot exceed 100%. This too must also be saved using the Save button located at the bottom.

Custom LAN subnet

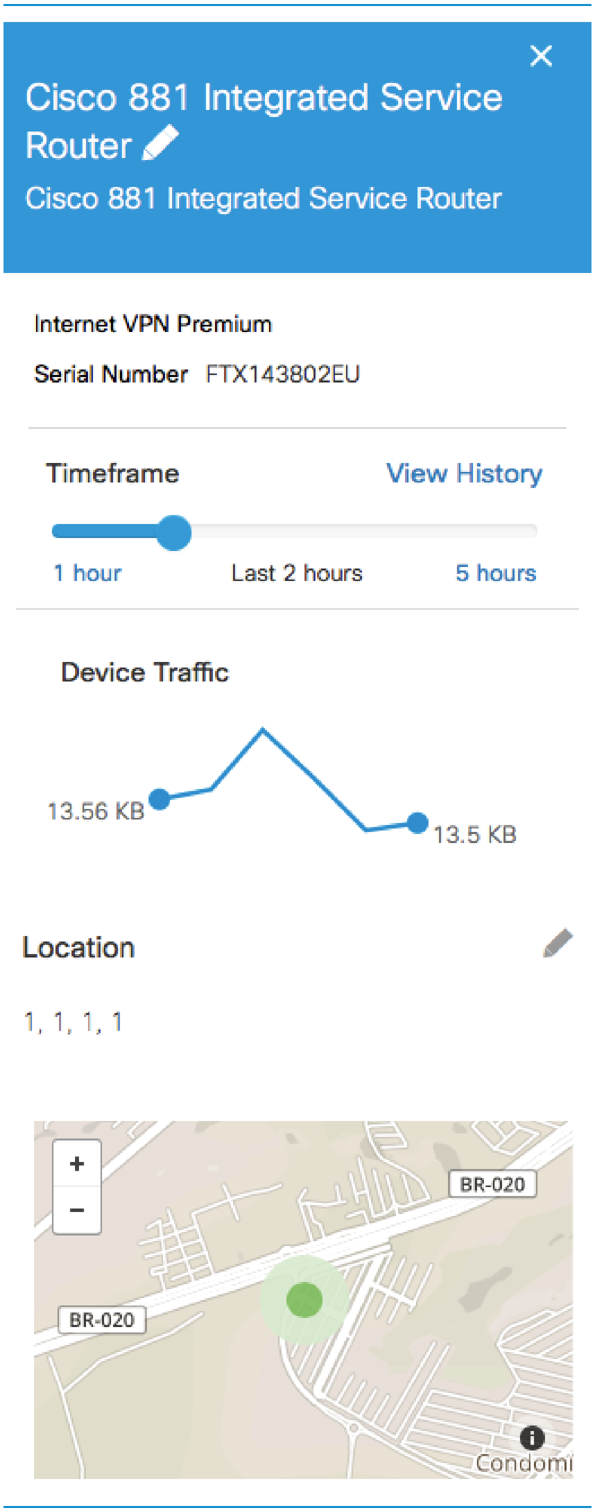
An end customer can also define the local IP address range for the LAN network served by a CPE device. To change the local IP address range, the end customer needs to select "Devices" from the menu on the left-hand side of the vMS Portal. This will display the CPE routers configured for their CloudVPN service as seen below.

Figure 4.37 Navigating to Devices Tab



When the end customer clicks on one of the routers, a new box will appear on the right-hand side of the browser similar to the one below.

Figure 4.38 Device Details



Cisco 881 Integrated Service Router

Cisco 881 Integrated Service Router

Internet VPN Premium

Serial Number FTX143802EU

Timeframe View History

1 hour Last 2 hours 5 hours

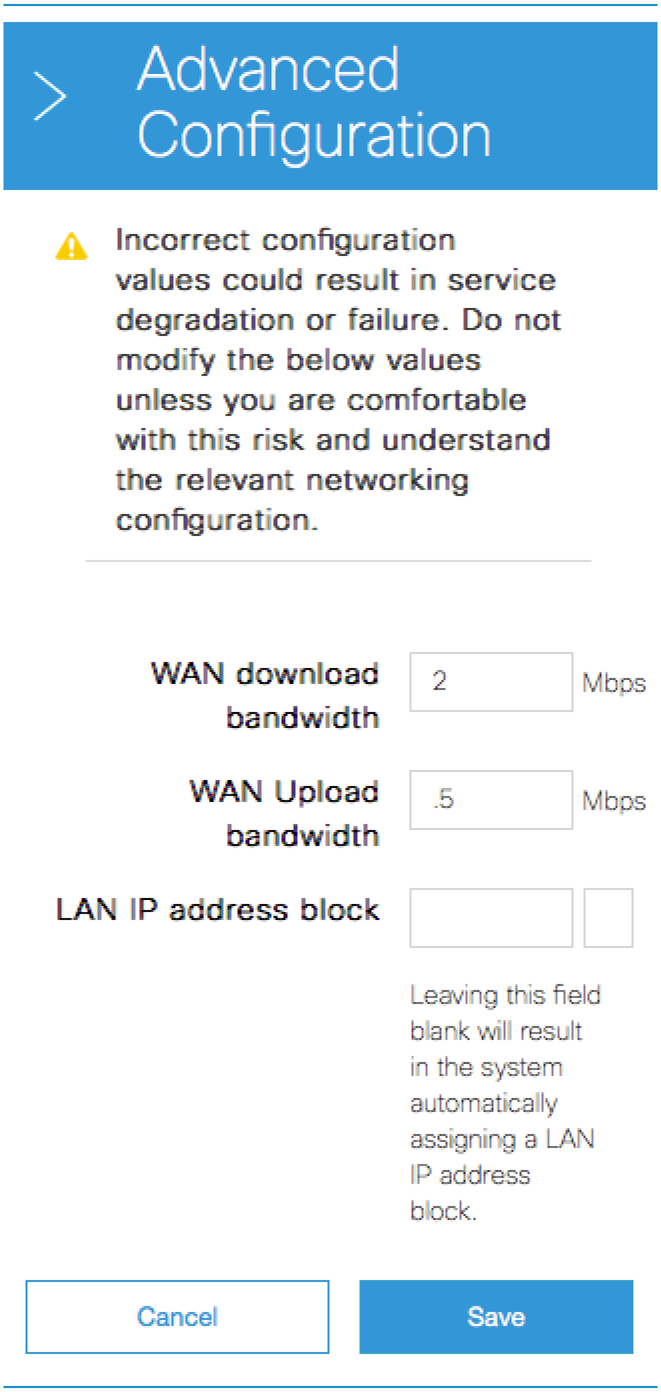
Device Traffic

Location

1, 1, 1, 1

The end customer can scroll further down to get to the "Advanced Configuration" and clicks on the edit (pencil) icon to open the "Advanced Configuration" box as seen below. The local LAN network IP address range can be changed in the boxes for LAN IP address blocks. The first box is for an IPv4 subnet and the second is for the subnet mask (ex. 192.168.1.0 24). To save the configuration, the end customer needs to click on the "Save" button.

Figure 4.39 Device Advanced Configuration



Advanced Configuration

⚠ Incorrect configuration values could result in service degradation or failure. Do not modify the below values unless you are comfortable with this risk and understand the relevant networking configuration.

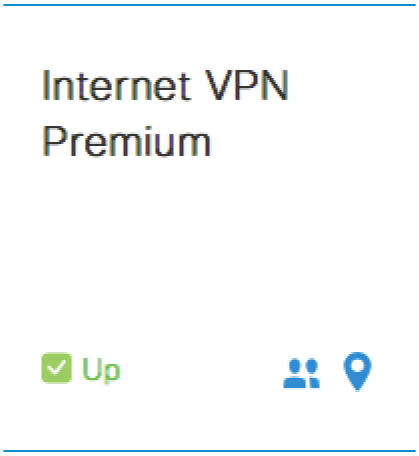
WAN download bandwidth	2	Mbps
WAN Upload bandwidth	.5	Mbps
LAN IP address block		

Leaving this field blank will result in the system automatically assigning a LAN IP address block.

RA User add/delete/password change

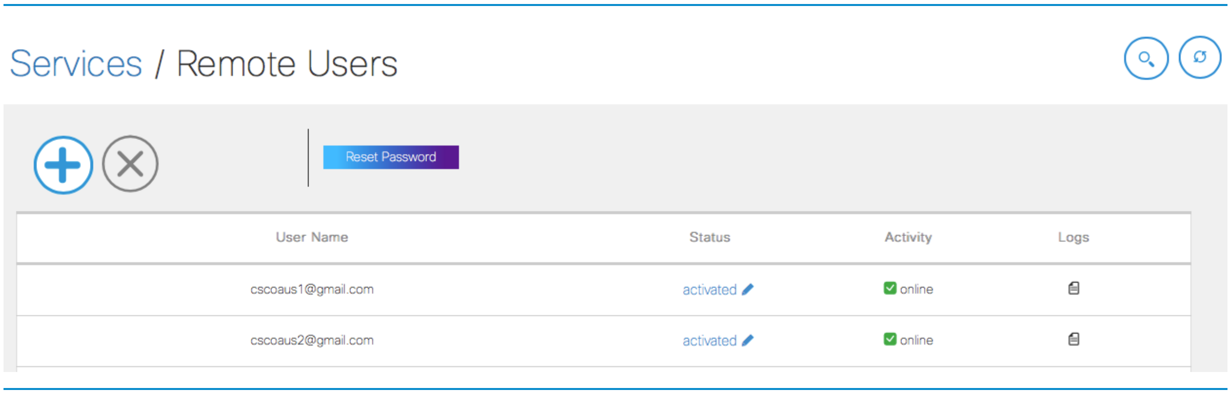
To administer Remote Access users, the end customer needs to select Services on the left-hand menu and then select the "people icon".

Figure 4.40 Navigating to Service View



This will bring the end customer to the Remote Access user Administration part of the portal for their CloudVPN service.

Figure 4.41 Managing Remote Access VPN Users



User Name	Status	Activity	Logs
cscoaus1@gmail.com	activated	☑ online	
cscoaus2@gmail.com	activated	☑ online	

A Remote Access user can be added by clicking on the Plus button. The end customer needs to enter a username for the remote access user account that needs to be created. A randomly-generated password will be created for the newly added user. The new remote access user will be notified via email of the login credentials, which also contains the URL for remote access. This is the URL that was created by the DNS-Updater when the original CloudVPN service was provisioned. The new user is added to the tenant's ASAv running configuration.

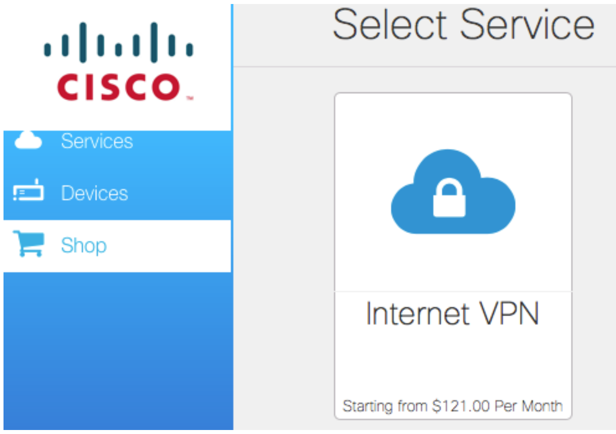
Within this same Remote Access Users page, a user account can be removed by selecting the remote user account and clicking on the X button.

A Remote Access User account can also have the password reset by selecting the Remote Access User account and clicking on the Reset Password button.

URL filtering levels, Bandwidth settings, and Additional Locations

All of the above settings can be changed by selecting the Shop option from the menu item selection on the left and by clicking on the desired service box.

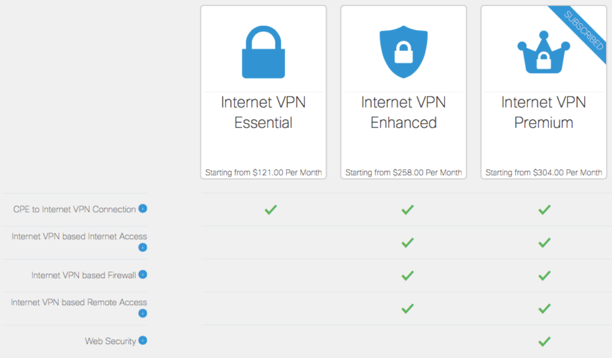
Figure 4.42 Navigating to Shop > Select Service



The end customer will be presented with 3 different options for their CloudVPN service. The end customer needs to select the current service level for which the bandwidth needs to be changed.

Note: Please note that the level of URL Filtering is only available with the Internet VPN Premium service level.

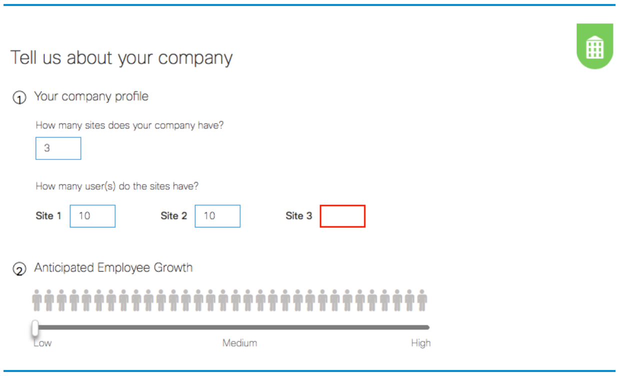
Figure 4.43 Select Subscribed Service Bundle to Modify



	Internet VPN Essential Starting from $121.00 Per Month	Internet VPN Enhanced Starting from $258.00 Per Month	SUBSCRIBED Internet VPN Premium Starting from $304.00 Per Month
CPE to Internet VPN Connection	✓	✓	✓
Internet VPN based Internet Access		✓	✓
Internet VPN based Firewall		✓	✓
Internet VPN based Remote Access		✓	✓
Web Security			✓

To add a new location, the end customer needs to change the number of sites to reflect their total number of sites. If they previously had 2 sites and wanted to add a 3rd site, the end customer would need to change the number of sites to 3.

Figure 4.44 Modify Number of Sites



Tell us about your company

① Your company profile

How many sites does your company have?

3

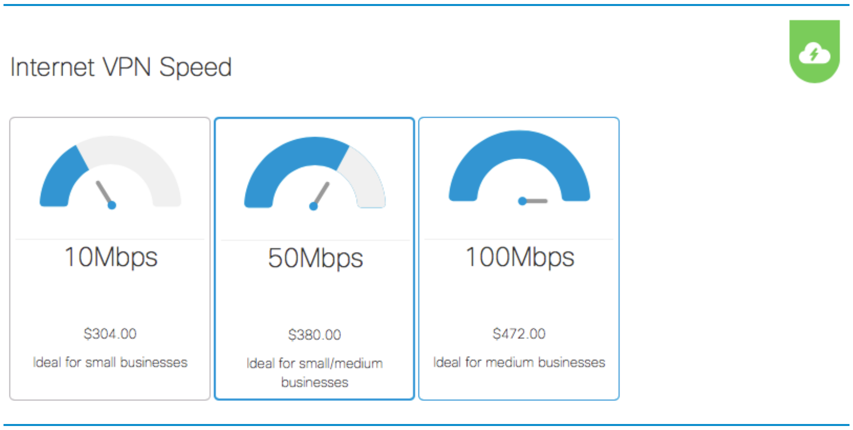
How many user(s) do the sites have?

Site 1: 10
Site 2: 10
Site 3:

② Anticipated Employee Growth

In order to change the Internet VPN speed, the end customer needs to select one of the presented options in the box below Previous.

Figure 4.45 Modifying Internet VPN Speed



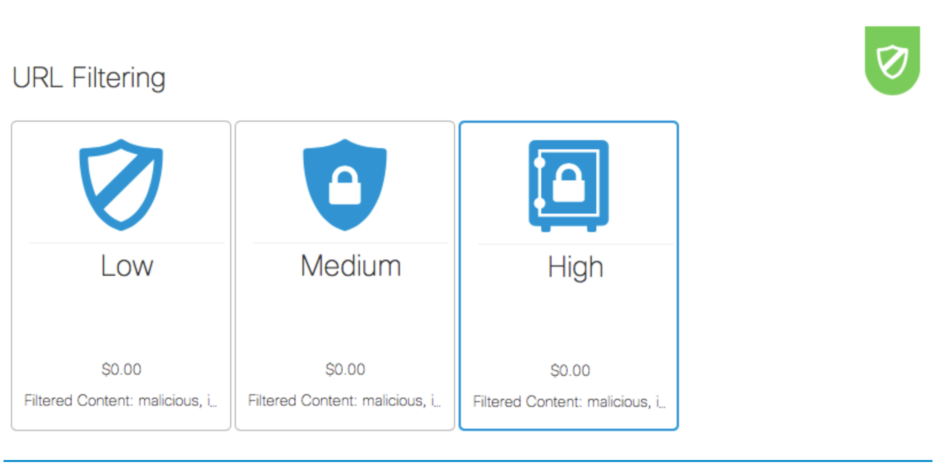
Internet VPN Speed

10Mbps	50Mbps	100Mbps
$304.00	$380.00	$472.00
Ideal for small businesses	Ideal for small/medium businesses	Ideal for medium businesses

The next box allows the end user to change the level of URL filtering out of three possible choices:

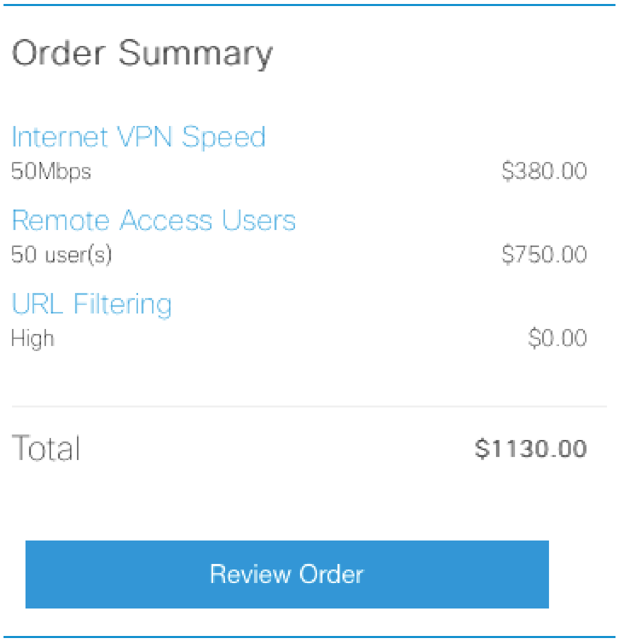
- Low - Filtered Content: Malicious, illegal
- Medium - Filtered Content: Malicious, illegal, adult, involving illegal substances
- High - Filtered Content: Malicious, illegal, adult, involving illegal substances, entertainment, sports, non-business related

Figure 4.46 Modifying Web-URL Filtering Level


URL Filtering
Low
$0.00
Filtered Content: malicious, i...
Medium
$0.00
Filtered Content: malicious, i...
High
$0.00
Filtered Content: malicious, i...

After completing the desired service modifications, the end customer needs to select the Review Order button on the right.

Figure 4.47 Order Summary

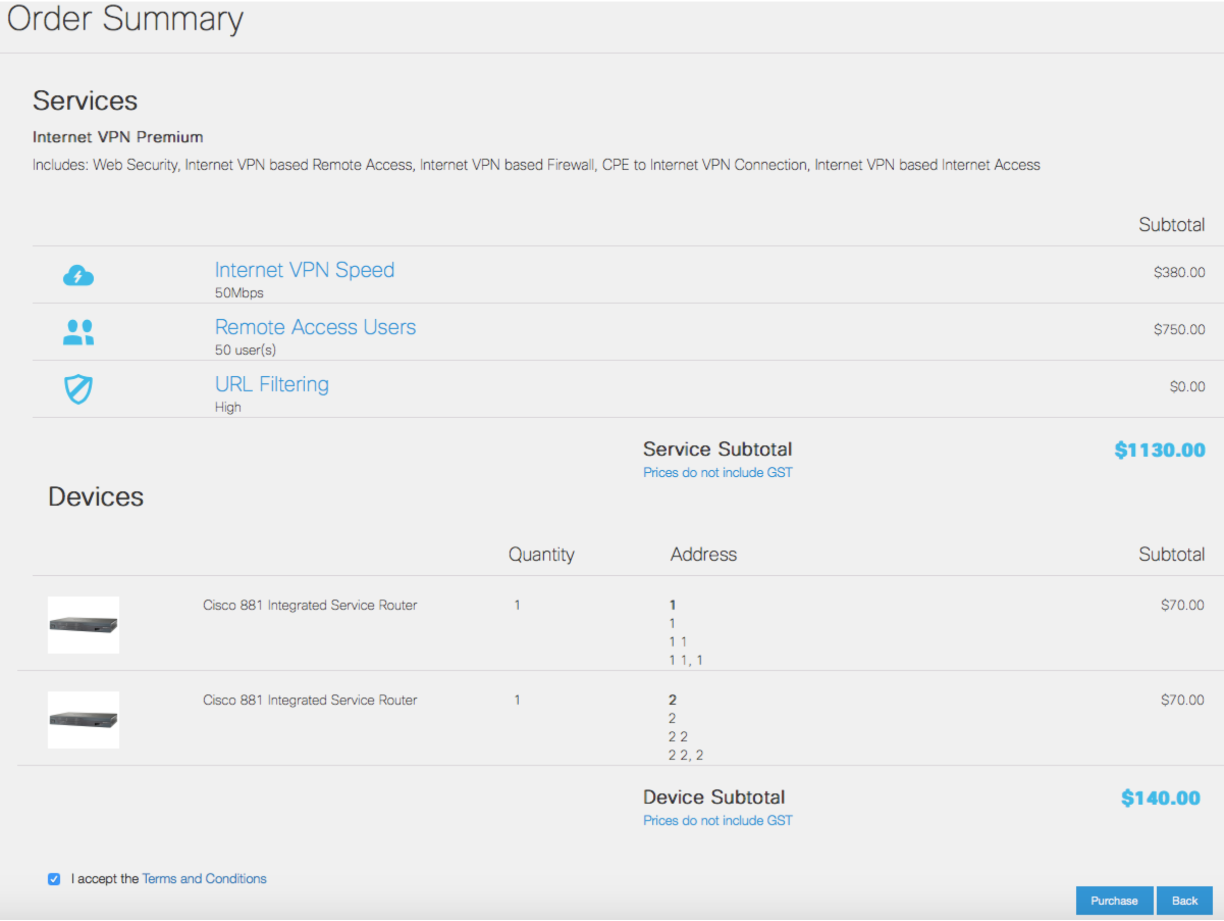


Order Summary

Internet VPN Speed 50Mbps	$380.00
Remote Access Users 50 user(s)	$750.00
URL Filtering High	$0.00
Total	**$1130.00**

After clicking on Review Order, the end customer will be taken to the Order Summary Page where a review of the service modifications and the service price are displayed. The changes will not take effect until the customer checks the box "I accept the Terms and Conditions" and then clicks on the "Purchase" button.

Figure 4.48 Reviewing Order Summary and Purchasing



Order Summary

Services

Internet VPN Premium

Includes: Web Security, Internet VPN based Remote Access, Internet VPN based Firewall, CPE to Internet VPN Connection, Internet VPN based Internet Access

		Subtotal
	Internet VPN Speed 50Mbps	$380.00
	Remote Access Users 50 user(s)	$750.00
	URL Filtering High	$0.00

Service Subtotal **$1130.00**
Prices do not include GST

Devices

		Quantity	Address	Subtotal
	Cisco 881 Integrated Service Router	1	**1** 1 1 1 1 1, 1	$70.00
	Cisco 881 Integrated Service Router	1	**2** 2 2 2 2 2, 2	$70.00

Device Subtotal **$140.00**
Prices do not include GST

☑ I accept the Terms and Conditions

4.5 Troubleshooting

Verifying Portal Microservices State

Login to the Inception VM and execute: `cf app <microservice-name>`

```
root@vms-inception-2:/home/ubuntu# cf app Skyfallui
Showing health and status for app Skyfallui in org vms / space app as admin...
OK

requested state: started
instances: 1/1
usage: 2G x 1 instances
urls: skyfallui.vmssol1.io
last uploaded: Wed Dec 9 07:09:38 UTC 2015
stack: cflinuxfs2
buildpack: httpd_buildpack_40

     state     since                    cpu    memory         disk      details
#0   running   2015-12-09 07:10:30 AM   0.3%   187.2M of 2G   0 of 2G
```

Portal URL Inaccessible

- Check DNS resolution of the URL
- Check reachability to the vMS Portal IP
- Verify HA-Proxy service is UP
- Verify HA-Proxy VM is up and running in OpenStack, and also associated with vMS Portal IP
- Login to Inception VM and verify haproxy/0, router/0 and router/1 Bosh VMs are up and running

```
root@vms-inception-2:~# bosh vms
Acting as user 'admin' on 'cns-soltest1b-cp'
Deployment `cf-soltest1b'

Director task 59

Task 59 done

+----------------------------------+---------+--------------------+-------------+
| Job/index                        | State   | Resource Pool      | IPs         |
+----------------------------------+---------+--------------------+-------------+
| api_worker/0                    | running | large              | 10.20.0.176 |
| cassandra/0                     | running | service-net-large  | 10.20.0.184 |
| cassandra_seed/0                | running | service-net-large  | 10.20.0.180 |
| cloud_controller/0              | running | large              | 10.20.0.185 |
| dea-spare/0                     | running | cf-xlarge          | 10.20.0.175 |
| dea-spare/1                     | running | cf-xlarge          | 10.20.0.179 |
| haproxy/0                       | running | small              | 10.20.0.173 |
| kafka/0                         | running | service-net-large  | 10.20.0.186 |
| kafka/1                         | running | service-net-large  | 10.20.0.178 |
| kafka/2                         | running | service-net-large  | 10.20.0.182 |
| loggregator/0                   | running | small              | 10.20.0.172 |
| loggregator_trafficcontroller/0 | running | small              | 10.20.0.171 |
| nfs/0                           | running | medium             | 10.20.0.181 |
| redis/0                         | running | service-net-large  | 10.20.0.187 |
| router/0                        | running | medium             | 10.20.0.189 |
| router/1                        | running | medium             | 10.20.0.174 |
| zookeeper/0                     | running | service-net-medium | 10.20.0.188 |
| zookeeper/1                     | running | service-net-medium | 10.20.0.177 |
| zookeeper/2                     | running | service-net-medium | 10.20.0.183 |
+---------------------------------+---------+--------------------+-------------+

VMs total: 19
```

If there are any issues with any of the Bosh VM (e.g. haproxy/0) the Bosh Director will try to recover automatically.

Starting and Stopping Services Deployed in Cloud Foundry

If there are issues with starting services, for eg: Cassandra, log into the VM where the service is running and execute the `monit status` command to view the status of the service and attempt to manually stop/start the service. To gain root privileges, please review section "Connecting to CF VMs Using Bosh SSH" in this chapter.

```
root@9b07b742-3e3b-4250-b162-ce2c81102469:/var/vcap/bosh_ssh/bosh_59idbxx3j# monit status
The Monit daemon 5.2.4 uptime: 3d 18h 6m
Process 'cassandra'
status                                    running
monitoring status                         monitored
pid                                       4311
parent pid                                1
uptime                                    3d 18h 6m
children                                  0
memory kilobytes                          2391452
memory kilobytes total                    2391452
memory percent                          29.2%
memory percent total                    29.2%
cpu percent                             0.4%
cpu percent total                       0.4%
data collected                          Mon Oct 19 21:19:45 2015
  monit stop <service name>
  monit start <service name>
```

Problem Creating Tenants

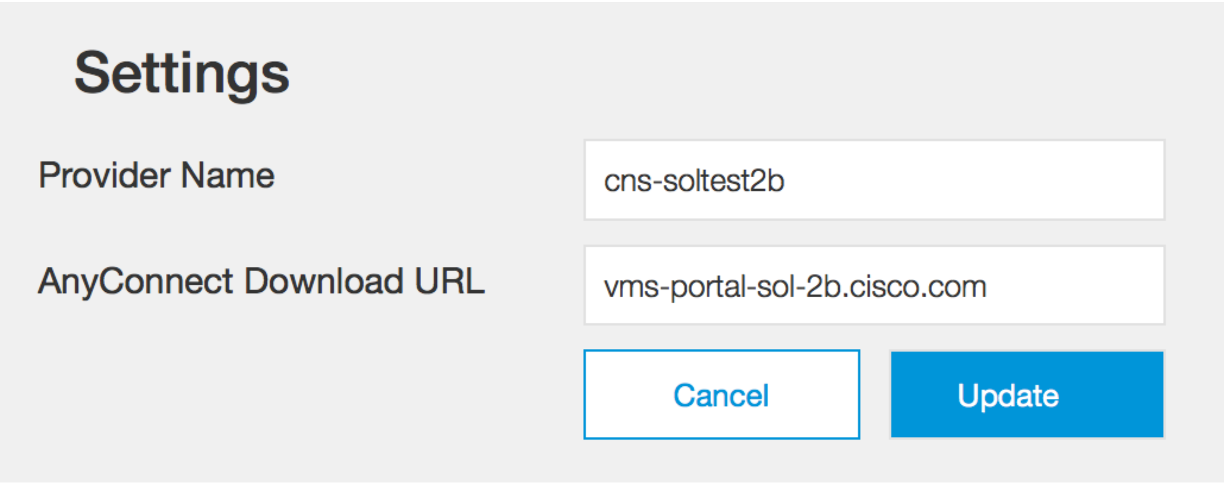
- Verify if Cassandra/0 VM is in running state.
- Access Cassandra DB from Inception VM
  - `cqlsh <CASSANDRA_DB_IP>` (capture from bosh vms output)
  - Execute "DESCRIBE keyspace;" and verify if the "skyfall_idm" keyspace has been created

```
root@vms-inception-2:~# bosh vms
Acting as user 'admin' on 'cns-soltest1b-cp'
Deployment `cf-soltest1b'

Director task 59

Task 59 done

+---------------------------------+---------+--------------------+-------------+
| Job/index                       | State   | Resource Pool      | IPs         |
+---------------------------------+---------+--------------------+-------------+
| api_worker/0                    | running | large              | 10.20.0.176 |
| cassandra/0                     | running | service-net-large  | 10.20.0.184 |
| cassandra_seed/0                | running | service-net-large  | 10.20.0.180 |
| cloud_controller/0              | running | large              | 10.20.0.185 |
| dea-spare/0                     | running | cf-xlarge          | 10.20.0.175 |
| dea-spare/1                     | running | cf-xlarge          | 10.20.0.179 |
| haproxy/0                       | running | small              | 10.20.0.173 |
| kafka/0                         | running | service-net-large  | 10.20.0.186 |
| kafka/1                         | running | service-net-large  | 10.20.0.178 |
| kafka/2                         | running | service-net-large  | 10.20.0.182 |
| loggregator/0                   | running | small              | 10.20.0.172 |
| loggregator_trafficcontroller/0 | running | small              | 10.20.0.171 |
| nfs/0                           | running | medium             | 10.20.0.181 |
| redis/0                         | running | service-net-large  | 10.20.0.187 |
| router/0                        | running | medium             | 10.20.0.189 |
| router/1                        | running | medium             | 10.20.0.174 |
| zookeeper/0                     | running | service-net-medium | 10.20.0.188 |
| zookeeper/1                     | running | service-net-medium | 10.20.0.177 |
| zookeeper/2                     | running | service-net-medium | 10.20.0.183 |
+---------------------------------+---------+--------------------+-------------+
  root@vms-inception-2:~# cqlsh 10.20.0.184
Connected to vms at 10.20.0.184:9160.
[cqlsh 4.1.1 | Cassandra 2.1.0 | CQL spec 3.1.1 | Thrift protocol 19.39.0]
Use HELP for help.
cqlsh> DESCRIBE KEYSPACES;

skyfall_monitor  skyfall_consume  skyfall_orchestration  skyfall_manage
system           skyfall_idm      system_traces          skyfall_administration
```

Service Chain not Getting Created

- Verify consume microservice is running
- Verify orchestration microservice is running
- Verify orchestration microservice is sending the appropriate provider name to NSO. Confirm "Provider Name" is populated correctly by navigating to Portal > Settings as Admin.

Figure 4.49 Configuring Provider Name



Settings

Provider Name: cns-soltest2b

AnyConnect Download URL: vms-portal-sol-2b.cisco.com

Check NSO `netconf-north.log` for portal requests. If not, check connectivity between the Portal and the NSO.

vMS Service Assurance Troubleshooting

SME - NSO reports not getting generated

NSO reports on the SME are collected via the NSO NETCONF interface. If these are not getting generated:

- Ensure that the paramiko and libxml2-utils packages and dependencies are installed on VM hosting the NSO.
- Confirm that SME has been configured with the correct SSH and NETCONF credentials for NSO.

```
  root@vms-inception-2:~# ssh -i /opt/cisco/vms-installer/tenant-soltest1b/ssh/admin-key-
soltest1b cloud-user@<sme_ip>

  [root@sme-instance-soltest1b ~]# /opt/CSCOppm-gw/bin/ppm maskcreds status

  Password masking is enabled.

  [root@sme-instance-soltest1b ~]# /opt/CSCOppm-gw/bin/ppm maskcreds disable

  Password masking feature is disabled.

  ****************************************************************
  NOTE: This will cause passwords to be displayed in plain text
        in the user interface and on some command line operations.
  ****************************************************************
  [root@sme-instance-soltest1b ~]#

  [root@sme-instance-soltest1b ~]# /opt/CSCOppm-gw/bin/ppm showcreds
  Hostname : 10.32.1.24
  Username : admin
  Password : gXg7ZZgGujWEJTwV&#10;  Secondary Login Type : ENABLE
  Secondary Username : admin
  Secondary Password : \g\X\g\7\Z\Z\g\G\u\j\W\E\J\T\w\V&#10;  Protocol : SSH_V2
  Port : 22
  Path :
  Client Auth Type : password
  Client Private Key :
  Client Private Key :
```

- Resolve the issue by updating credentials using the command below:

```
  /opt/CSCOppm-gw/bin/ppm modifycreds -i <nso-ip-address> -r <connection-protocol> -u <ssh-
username> -p <ssh-password> -n <netconf-username> -e <netconf-password>
```

- Verify NSO status in SME is "Active". If not, verify SNMP is enabled on NSO, and also inspect `alarmLog.txt` logfile on SME.

> **Note:** Do not edit or vi the file. The file will get CORRUPTED and also block oper status changes to Portal.

```
  [root@sme-instance-soltest1b ~]# tail -f /opt/CSCOppm-gw/logs/alarmLog.txt.0
  3269001 | Unknown | 10.32.1.24 | 10.32.1.24 | Critical | NodeUnreachable | NEW | Device
10.32.1.24 is unreachable. | Node=10.32.1.24 | Unknown
  3269001 | Unknown | 10.32.1.24 | 10.32.1.24 | Normal | NodeUnreachable | UPDATE | Device
10.32.1.24 is reachable. | Node=10.32.1.24 | Unknown
  3534001 | Unknown | 10.32.1.24 | 10.32.1.24 | Critical | Symphony_NSO_Oper_State | NEW |
Threshold : 'Symphony_NSO_Oper_State' - 'Node=10.32.1.24,serviceId=cns-soltest1b-test1-
cloudvpn' crossed threshold for 'NSO Operational State 15 Minute/Operational State' - value
'down' onset 'Equals [down]'. Severity: Critical | Node=10.32.1.24,serviceId=cns-soltest1b-
test1-cloudvpn | NSO Operational State 15 Minute/Operational State
  3534001 | Unknown | 10.32.1.24 | 10.32.1.24 | Normal | Symphony_NSO_Oper_State | UPDATE |
Threshold : 'Symphony_NSO_Oper_State' - 'Node=10.32.1.24,serviceId=cns-soltest1b-test1-
cloudvpn' abated threshold for 'NSO Operational State 15 Minute/Operational State' - value 'up'
Original threshold 'Equals [down]'. Severity: Normal | Node=10.32.1.24,serviceId=cns-soltest1b-
test1-cloudvpn | NSO Operational State 15 Minute/Operational State
  3544001 | Unknown | 10.32.1.24 | 10.32.1.24 | Critical | Symphony_NSO_Node_Oper_State | NEW |
Threshold : 'Symphony_NSO_Node_Oper_State' - 'Node=10.32.1.24,serviceId=cns-soltest1b-test1-
cloudvpn,nodeId=cpe-FTX1826ALL9' crossed threshold for 'NSO Node Operational State 15
Minute/Operational State' - value 'down' onset 'Equals [down]'. Severity: Critical |
Node=10.32.1.24,serviceId=cns-soltest1b-test1-cloudvpn,nodeId=cpe-FTX1826ALL9 | NSO Node
Operational State 15 Minute/Operational State
```

Service or Device Status Remains in"Provisioned" State

If the service or device status on the portal persistently remains in the "Provisioned" state and not transitioning to the "UP" state:

- Verify NSO `netconf-north.log` to check if incoming requests for oper-data are being received from the SME. Requests for the following oper-data should be seen. If these are not being seen then verify SME to NSO communication.

```
   /cloudvpn-data/oper-data/FW-INET
   /cloudvpn-data/oper-data/connected_users
   /cloudvpn-data/oper-data/internet_traffic
   /cloudvpn-data/oper-data/nodes/node/oper-state
   /cloudvpn-data/oper-data/onnet_traffic
   /cloudvpn-data/oper-data/oper-state
```

```
  /cloudvpn-data/oper-data/RA
  /cloudvpn-data/oper-data/S2S
```

- Verify if SME has been configured with the correct SSH and NETCONF credentials for NSO.

```
 root@vms-inception-2:~# ssh -i /opt/cisco/vms-installer/tenant-soltest1b/ssh/admin-key-soltest1b cloud-user@<sme_ip>

 [root@sme-instance-soltest1b ~]# /opt/CSCOppm-gw/bin/ppm maskcreds status

 Password masking is enabled.

 [root@sme-instance-soltest1b ~]# /opt/CSCOppm-gw/bin/ppm maskcreds disable

 Password masking feature is disabled.

 ****************************************************************
 NOTE: This will cause passwords to be displayed in plain text
       in the user interface and on some command line operations.
 ****************************************************************
 [root@sme-instance-soltest1b ~]#

 [root@sme-instance-soltest1b ~]# /opt/CSCOppm-gw/bin/ppm showcreds
 Hostname : 10.32.1.24
 Username : admin
 Password : gXg7ZZgGujWEJTwV&#10;  Secondary Login Type : ENABLE
 Secondary Username : admin
 Secondary Password : \g\X\g\7\Z\Z\g\G\u\j\W\E\J\T\w\V&#10;  Protocol : SSH_V2
 Port : 22
 Path :
 Client Auth Type : password
 Client Private Key :
 Client Private Key :
```

- Resolve the issue by updating credentials using the command below:

```
 /opt/CSCOppm-gw/bin/ppm modifycreds -i <nso-ip-address> -r <connection-protocol> -u <ssh-username> -p <ssh-password> -n <netconf-username> -e <netconf-password>
```

- Verify NSO status in SME is "Active". If not, verify SNMP is enabled on NSO, and also inspect alarmLog.txt logfile on SME.

> **Note:** Do not edit or vi the file. The file will get CORRUPTED and also block oper status changes to Portal.

```
  [root@sme-instance-soltest1b ~]# tail -f /opt/CSCOppm-gw/logs/alarmLog.txt.0
  3269001 | Unknown | 10.32.1.24 | 10.32.1.24 | Critical | NodeUnreachable | NEW | Device
10.32.1.24 is unreachable. | Node=10.32.1.24 | Unknown
  3269001 | Unknown | 10.32.1.24 | 10.32.1.24 | Normal | NodeUnreachable | UPDATE | Device
10.32.1.24 is reachable. | Node=10.32.1.24 | Unknown
  3534001 | Unknown | 10.32.1.24 | 10.32.1.24 | Critical | Symphony_NSO_Oper_State | NEW |
Threshold : 'Symphony_NSO_Oper_State' - 'Node=10.32.1.24,serviceId=cns-soltest1b-test1-
cloudvpn' crossed threshold for 'NSO Operational State 15 Minute/Operational State' - value
'down' onset 'Equals [down]'. Severity: Critical | Node=10.32.1.24,serviceId=cns-soltest1b-
test1-cloudvpn | NSO Operational State 15 Minute/Operational State
  3534001 | Unknown | 10.32.1.24 | 10.32.1.24 | Normal | Symphony_NSO_Oper_State | UPDATE |
Threshold : 'Symphony_NSO_Oper_State' - 'Node=10.32.1.24,serviceId=cns-soltest1b-test1-
cloudvpn' abated threshold for 'NSO Operational State 15 Minute/Operational State' - value 'up'
Original threshold 'Equals [down]'. Severity: Normal | Node=10.32.1.24,serviceId=cns-soltest1b-
test1-cloudvpn | NSO Operational State 15 Minute/Operational State
  3544001 | Unknown | 10.32.1.24 | 10.32.1.24 | Critical | Symphony_NSO_Node_Oper_State | NEW |
Threshold : 'Symphony_NSO_Node_Oper_State' - 'Node=10.32.1.24,serviceId=cns-soltest1b-test1-
cloudvpn,nodeId=cpe-FTX1826ALL9' crossed threshold for 'NSO Node Operational State 15
Minute/Operational State' - value 'down' onset 'Equals [down]'. Severity: Critical |
Node=10.32.1.24,serviceId=cns-soltest1b-test1-cloudvpn,nodeId=cpe-FTX1826ALL9 | NSO Node
Operational State 15 Minute/Operational State
```

- Verify reports are getting generated in SME.
- Verify NSO-SHIM to NSO communication.

NSO-SHIM Troubleshooting

The configuration file can be checked to verify that all settings are appropriate based on the setup.

Configuration file: `/opt/cisco/nsoshim/CSCOnso-shim/etc/config.json`

```
  {
    "rest_bind": "0.0.0.0",     # nso shim rest server address
    "rest_port": 4450,          # nso shim rest server port
    "rest_ssl": "false",        # nso shim rest server SSL enabled
    "rest_httpauth": "false",   # nso shim rest server HTTP Auth enabled
    "rest_username": "admin",   # nso shim rest server HTTP Auth username
    "rest_password": "admin",   # nso shim rest server HTTP Auth password
    "ncs_host": "",             # nso host address
    "ncs_port": 830,             # nso host netconf port
    "ncs_username": "admin",     # nso host netconf username
    "ncs_password": "admin",     # nso host netconf password
    "ncs_notif": "true",         # nso notification support enabled
    "sme_host": "",              # sme host address
    "sme_port": "4440",          # sme host rest/soap API port
    "sme_ssl": "false",          # sme host SSL enabled
    "sme_username": "",          # sme host HTTP Auth username
    "sme_password": "",          # sme host HTTP Auth password
    "bus_timeout": 180           # nso shim message bus timeout
}
```

Inspecting NSO-SHIM Log files

1. `/opt/cisco/nsoshim/CSCOnso-shim/logs/nso-shim.out`

This is the log file for capturing standard out.

2. `/opt/cisco/nsoshim/CSCOnso-shim/logs/nso-shim.log`

This is the regular log file.

- 100MB max.
- At startup, the old logs are zipped, timestamped, and stored in the `logs/backup` directory.
- The log level is INFO - which is logging pretty much everything.

Configuration: `/opt/cisco/nsoshim/CSCOnso-shim/logging.properties`

```
handlers= java.util.logging.FileHandler
# LEVELS: SEVERE, WARNING, INFO, CONFIG, FINE, FINER ,FINEST
.level=INFO
java.util.logging.FileHandler.pattern = %h/logs/nso-shim.log
java.util.logging.FileHandler.limit = 100000000
java.util.logging.FileHandler.count = 1
java.util.logging.FileHandler.formatter = java.util.logging.SimpleFormatter
java.util.logging.SimpleFormatter.format=%4$s: %1$tc: %5$s%6$s%n
```

5. Network Services Orchestrator (NSO)

5.1 Introduction

This chapter provides details on the Cisco Network Services Orchestrator (NSO) - how NSO provisions services, how it is configured and how it integrates with the other vMS solution components. This chapter also provides NSO troubleshooting guidelines.

The chapter is divided into the following sections:

- Overview
- Configuration
- Operations
- ESC Registration
- Backup and Restore
- Upgrade
- Troubleshooting

5.2 Overview

Purpose

The Network Services Orchestrator (NSO) forms the orchestration engine of the vMS, taking service models described in the YANG modeling language and transforming them, using Reactive Fastmap technology, into low-level device configurations that are then delivered across both virtual and physical infrastructure, provisioning end-to-end network services.

In the vMS solution, NSO works in conjunction with a Virtual Network Function Manager (VNFM) also known as the Elastic Services Controller (ESC) to instantiate service chains made up of Virtual Network Functions (VNF) on top of a virtualized infrastructure such as OpenStack. On top of NSO, vMS provides different function packs, each delivering their own fully orchestrated managed service such as CloudVPN. NSO also provides a northbound NETCONF interface allowing for an external system such as a user portal to request and manage these services.

Finally, through the use of its PnP service, NSO can deliver Zero Touch Deployment (ZTD) to Customer Premise Equipment (CPE) devices located on remote sites, bringing them into service automatically.

NSO Orchestration Model

NSO enables Service Providers to deploy orchestration solutions according to changes in the offered service portfolio. This is enabled by using a model-driven architecture where service definitions can be changed on the fly. NSO uses YANG service model definitions-derived service configurations rather than hard-coded workflows. Service models are written in the YANG modeling language as described in RFC 6020.

NSO delivers an automated orchestration solution towards a hybrid multi-vendor network. The network can be a mix of traditional equipment, virtual devices, and SDN Controllers. This flexibility is managed by a Network Element Driver (NED) layer that abstracts the device interfaces and the Device Manager which enables generic device configuration functions.

At the core of NSO is the configuration data store (CDB), that synchronises with the actual device and service configurations. CDB also manages relationships between services and devices and can handle multiple revisions of device interfaces.

NSO has two main layers: the Device Manager and the Service Manager. They serve different purposes but are tightly integrated into one transactional engine and database.

The Service Manager makes it possible for an operator to manage high-level aspects of the network that are not supported by the devices directly, or which are supported in a cumbersome way. With the appropriate service definition running in the Service Manager, an operator could, for example, configure the VLANs that should exist in the network in a single place, and the Service Manager would compute the specific configuration changes required for each device in the network and push them out.

This covers the whole life-cycle for a service: creation, modification, and deletion.

NSO has an intelligent, easy-to-use mapping layer, so network engineers can define how a service should be deployed in the network.

The Service Manager addresses the following challenges:

- Transaction-safe activation of services across different multi-vendor devices
- What-if scenarios, (dry-run), showing the effects on the network for a service creation/change
- Maintaining relationships between services and corresponding device configurations and vice versa
- Modeling of services
- Short development and turn-around time for new services
- Mapping the service model to device models.

The purpose of the Device Manager is to manage device configurations in a transactional manner. It supports features such as fine-grained configuration commands, bi-directional device configuration synchronization, device groups and templates, and compliance reporting.

The Device Manager supports the following overall features:

- Deploy configuration changes to multiple devices in a fail-safe way using distributed transactions.
- Validate the integrity of configurations before deploying to the network.
- Apply configuration changes to named device groups.
- Apply templates (with variables) to named device groups.
- Easily roll back changes, if needed.
- Configuration audits: Check whether device configurations are in sync with the NSO database. If they are not, what is the diff?
- Synchronize the NSO database and the configurations on devices, in case they are not in sync. This can be done in either direction (import the diff to the NSO database or deploy the diff on devices).

The NSO provides user interfaces as well as northbound APIs for integration to other systems. The main user interface is the NSO network-wide CLI which gives a unified CLI towards the complete network including the network services.

The northbound APIs are available in different language bindings (Java, Python), and as protocols, like NETCONF and REST.

In order to support dynamic updates of functionality as added or modified service models, support for a new device type etc, NSO manages extensions as well-defined packages. Every NED is its own package with its own release life-cycle. Every service model with corresponding mapping is also a package of its own. These can be upgraded without upgrading NSO itself.

Fastmap (Internal Call Flow)

FASTMAP is a service-to-device mapping solution that minimizes the complexity of specifying how service configuration changes are reflected in corresponding device configurations. The Cisco NSO reduces the mapping challenge to a bare minimum. The solution covers the complete service lifecycle so services can be changed and deleted and NSO will automatically calculate the minimum change to be applied to the network.

How does the NSO solve this? Service and device models are the foundation for the solution. The NSO uses the IETF standardized YANG [RFC 6020] modeling language to specify service models as well as device models. This is true even for devices that do not support YANG natively, such as CLI- or SNMP- based devices. The NSO tool-chain automatically generates YANG from SNMP MIBs and for CLI devices the declarative YANG model can automatically render the corresponding CLI commands. This collapses the first problem; the mapping is reduced to a YANG-to-YANG mapping problem rather than the normal mapping from informal service models to sequences of CLI commands or sequences of SNMP SETs.

How can NSO calculate the minimum difference towards the network? NSO stores the service instances and the device configurations in the embedded configuration data store, CDB. In contrast to traditional "offline" inventory systems, the NSO focuses on transactional integrity between the data store and the network. The core technology of the NSO is the capability to read and write configurations to/from the devices. This leads to a "real-time" database that is always in sync with the network. Therefore, NSO can look at the desired device configuration and generate the diff versus current configuration at the model level. Second, NSO can automatically render the underlying device commands for various interfaces like Cisco-style CLI, NETCONF, SNMP etc. No programming is required for this. Note that this also includes tricky problems like sending CLI commands or SNMP PDUs in the correct order, depending on what has changed.

How is the transformation from service models to device models defined? This is normally a complex problem. It might look easy from a thousand feet above, but when it comes to the actual configuration values for a router, there are configuration details that need to be well defined. NSO simplifies this drastically by use of the FASTMAP algorithm. Typically, an engineer defines a higher level service model representing, for example, the VPN service itself, and this service shall result in actual device configurations.

FASTMAP enables automatic management of any kind of change and deletion of a service. All change scenarios and deletions are inferred from a single definition of the creation of the service. The mapping from the service created to the corresponding device configuration is defined in a high-level API or through a template. The programmer only has to define the service creation method. If later, a user changes or deletes an existing service, the NSO calculates the changes. The result of this is close to magic - at any point in time a user or external client can modify a service and the NSO will automatically calculate the required minimum diff to the network. Also, deleting a service from the network will automatically clean up all configuration data on the devices. On top of this, FASTMAP automatically enables the following functions:

1. Service dry-run: if I apply this service to the network, what will happen? The output can be shown in a data-model general way as well as a native CLI, NETCONF etc output.

2. Service check-sync: is the service configuration in sync with the actual device configuration? This can detect the case that a network engineer has modified the device configuration by using the native CLI and that the change is in conflict with a desired service configuration. It can also show the minimum diff to restore the service

3. Service redeploy: generate the minimum diff to restore a service and make it happen in the network.

Function Packs and Network Element Drivers

vMS provides various function packs such as CloudVPN which is a specific set of packages loaded into NSO upon the program startup. These packages are a combination of YANG models and Java code that extend NSO's capabilities by providing additional features to the NSO. The following packages are loaded by NSO for vMS functionality from the overall CloudVPN function pack. It loads a combination of NEDs and services as seen below: In order to communicate with the IOS devices in the CloudVPN solution, corresponding NEDs need to be loaded by the NSO. A NED contains the YANG model for the device and corresponding driver code to talk CLI, REST, SNMP, etc. NEDs are distributed as packages.

Table 5.1 NSO Packages included in CloudVPN Function Pack

cisco-asa	NED package for Cisco ASA
cisco-ios	NED package for Cisco IOS
cisco-iosxr	NED package for Cisco IOS XR
cisco-pnp	NED package for supporting Cisco PNP (Plug-n-Play)
cisco-vips_manager	Based on: NED package for Cisco vIPS Manager (not currently used in production)
cisco-vips_sensor	Based on: NED package for Cisco vIPS Sensor (not currently used in production)
cisco-wsa	NED package for the Cisco WSA
cloudvpn	Skeleton for a resource-facing service - RFS
day0	Day 0 template-based service
dns-updater	NED package for updating DNS server entries
esc	netconf package for ESC
id-allocator	Resource allocator for IDs
iosstats	Generated SNMP package
ipaddress-allocator	Resource allocator for IP addresses
perfdata	Performance/Service Assurance data for NSO.
quagga-bgp	NED package for Quagga BGP daemon

resource-manager	Resource manager package
tailf-hcc	NED package for Tail-f HA Cluster Control Interface
vm-manager	VM manager interface package
vm-manager-esc	ESC VM manager package; dependent on the vm-manager package

Communication (Northbound/Southbound)

NSO uses two communication interfaces for communication with other vMS solution components:

- Northbound communication interface: for communication between NSO and vMS Portal
- Southbound communication interface: for communication between NSO and NSO managed devices

Northbound Interface

Northbound communication to the NSO is from the vMS portal and uses the NETCONF protocol (RFC 6241) which by default runs on TCP port 830 and uses SSH for secure transport. The NSO logs activity on this interface to `/var/log/ncs/netconf-north.log` and keeps track of all the communication between NSO and the portal. Examples of this communication would be a service creation request from the portal to NSO.

Southbound Interface

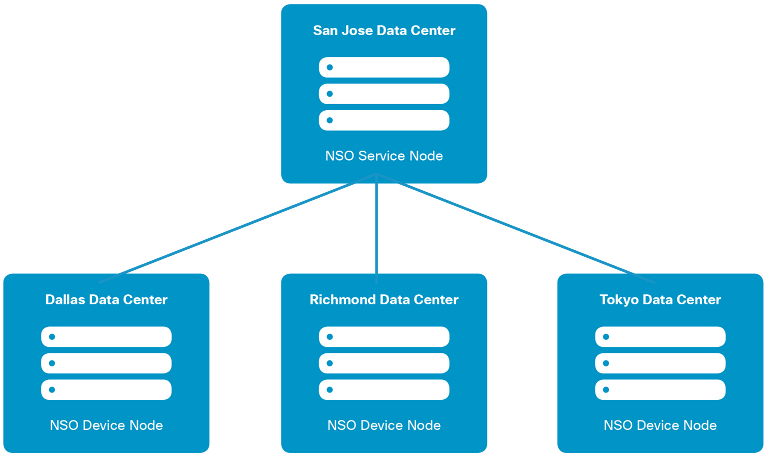
Southbound communication from NSO is to end-devices that are managed by the NSO. All southbound communication is logged in trace logs per device that are also located in `/var/log/ncs`. Examples of this type of communication would be NSO requesting devices status or changing the configuration on the device.

Clustering (Service Node - Device Node)

There can be many reasons to use clustering: scaling (handling more devices than a single node can handle), partitioning considerations (for example delegating Device Manager responsibility according to geographical or network topological considerations). Another reason can be to spread out services (services can be made to run on different cluster nodes). Clustering (not to be confused with HA-clustering) allows the assignment of the roles to servers.

In vMS, the Service Manager and Device Manager functions of NSO are split across multiple servers and possibly data centers, depending on the setup. This is accomplished with the concept of clustering. In this configuration, NSO assigns the Service Manager role to a Service node(s). And the Device Manager roles are assigned to device node(s). An example diagram of NSO clustering in vMS is shown below.

Figure 5.1 NSO Clustering



HA-Clustering (Quagga/BGP)

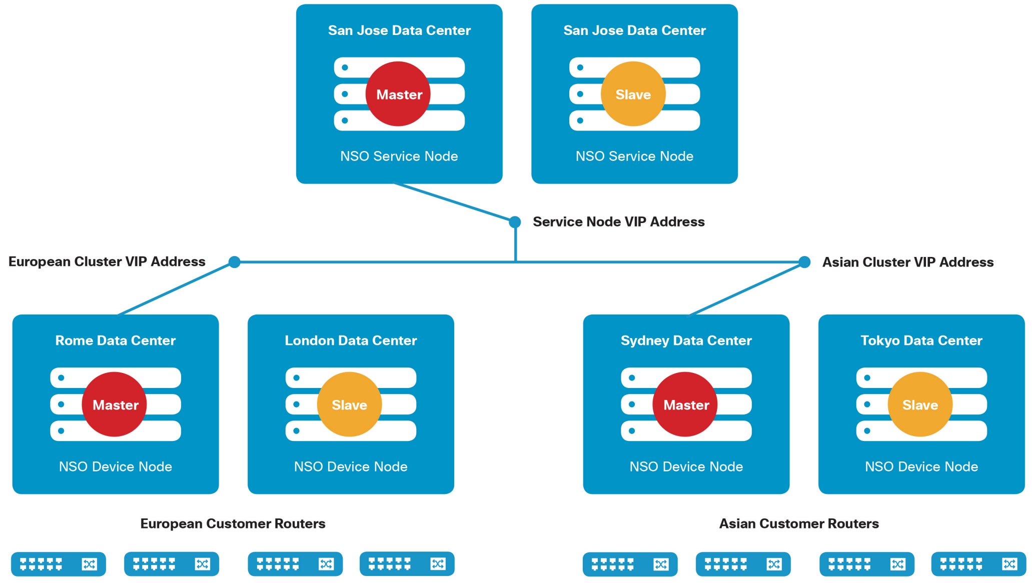
NSO supports a one-to-many high-availability mode. One NSO system can be Master and it can have any number of Slaves. Any configuration write has to go through the Master node. The configuration changes are replicated to the read-only Slaves. The replication can be done in asynchronous or synchronous mode. NSO supports replication of the CDB configuration as well as of the operational data kept in CDB. The replication architecture is that of one active Master and a number of passive Slaves. A group of NSO hosts consisting of a Master, and one or more Slaves, is referred to as an HA group (and sometimes as an HA cluster - however, this is completely independent and separate from the clustering feature. All configuration write operations must occur at the Master, and NSO will automatically distribute the configuration updates to the set of live Slaves. Operational data in CDB may be replicated or not, based on the "tailf:persistent" statement in the data model. All write operations for replicated operational data must also occur at the Master, with the updates distributed to the live slaves, whereas non-replicated operational data can also be written on the Slaves. The only thing NSO does is to replicate the CDB data amongst the members in the HA group. It doesn't perform any of the otherwise High-Availability related tasks such as running election protocols in order to elect a new master.

HA-Clustering in vMS is utilizing the tailf-hcc and quagga-bgp packages to operate in NSO. HA-Clustering if configured to have High Availability based on node functions. An example of this would be two Service Nodes in an HA configuration or two Device Nodes in an HA configuration.

HA is achieved via a Virtual IP (VIP) address that represents the nodes configured in the HA-Cluster. Communication from HA-Cluster Service Node <-> HA-Cluster Device Node <-> HA-ESC is to the VIP addresses of each and managed via the Quagga-BGP router.

The following figure shows the concepts of HA-Clustering.

Figure 5.2 NSO HA-Clustering



Plug and Play (PnP) Package

The Cisco Network Plug and Play solution provides a simple, secure, unified, and integrated offering for Service Providers to provision CPE to new CloudVPN service chains. The solution allows Service Providers to deploy CPE in the field with a near zero touch deployment experience.

It reduces the burden on Service Providers by greatly simplifying the process of deploying new devices. An end customer at the site can deploy a new device without any CLI knowledge, while a network administrator centrally manages device configuration.

Verify NSO Health Status

There are multiple ways to determine the health status of the NSO server.

The first check is to make sure that the CloudVPN release of NSO is installed. This can be verified with the `show cloudvpn-release` command. If this command is not available on the CLI then NSO does not have the correct packages loaded. The expected output should be similar to the one shown below:

```
admin@ncs-vm> show cloudvpn-release
cloudvpn-release cvpn-version vMS-2.1
cloudvpn-release build-date 2015-10-05-01-15
cloudvpn-release ncs-build-version 3.4.2.1
```

```
  cloudvpn-release git-revision 717cddb
  cloudvpn-release git-branch dev
  cloudvpn-release build-user px-build
  cloudvpn-release build-host bxb-bld10-lnx
  cloudvpn-release build-host-uname Linux
  cloudvpn-release build-host-java-version 1.7.0_25-Java(TM)
  [ok][2015-12-08 14:32:55]
```

The second check is to ensure that all of the required packages for the vMS are loaded into the NSO. The command to run is `show packages package oper-status`. All of the packages are expected to be loaded in an UP state. The packages are a combination of function packs and NEDs.

```
  admin@ncs-vm> show packages package oper-status
                                                                            PACKAGE
                                                                            META           FILE
                         JAVA          BAD NCS  PACKAGE  PACKAGE  CIRCULAR   DATA       JAVA  LOAD
ERROR
  NAME                UP UNINITIALIZED VERSION  NAME     VERSION  DEPENDENCY ERROR      ERROR ERROR
INFO
  ----------------------------------------------------------------------------------------------
---------
  cisco-asa           X  -             -        -        -       -          -        -     -
-
  cisco-ios           X  -             -        -        -       -          -        -     -
-
  cisco-iosxr         X  -             -        -        -       -          -        -     -
-
  cisco-pnp           X  -             -        -        -       -          -        -     -
-
  cisco-vips_manager  X  -             -        -        -       -          -        -     -
-
  cisco-vips_sensor   X  -             -        -        -       -          -        -     -
-
  cisco-wsa           X  -             -        -        -       -          -        -     -
-
  cloudvpn            X  -             -        -        -       -          -        -     -
-
  day0                X  -             -        -        -       -          -        -     -
-
  dns-updater         X  -             -        -        -       -          -        -     -
-
  esc                 X  -             -        -        -       -          -        -     -
-
  id-allocator        X  -             -        -        -       -          -        -     -
```

```
-
  iosstats             X   -              -        -        -        -           -        -      -
-
  ipaddress-allocator  X   -              -        -        -        -           -        -      -
-
  perfdata             X   -              -        -        -        -           -        -      -
-
  quagga-bgp           X   -              -        -        -        -           -        -      -
-
  resource-manager     X   -              -        -        -        -           -        -      -
-
  tailf-hcc            X   -              -        -        -        -           -        -      -
-
  vm-manager           X   -              -        -        -        -           -        -      -
-
  vm-manager-esc       X   -              -        -        -        -           -        -      -
-

  [ok][2015-12-08 16:55:41]
```

If any of the packages are in a state other than UP, an individual package can be re-deployed with the `request packages redeploy` command. An example is shown below:

```
admin@ncs-vm> request packages package cisco-asa redeploy
result true
[ok][2015-12-08 14:25:29]
```

All packages can be reloaded by using the `request packages reload` command. If a package is missing, it is needed to verify that all packages that are necessary to run vMS are on the NSO server. The packages can be found at `/var/opt/ncs/packages`.

If neither of the options above resolve the issue, then ncs.log, devel.log, debug dump files in the `/var/log/ncs` dir need to be collected and uploaded to a TAC case with the Cisco NSO team for further troubleshooting. If the problem is related to the NSO startup, starting NSO with the flags `--verbose` and `--foreground` will print additional debug messages to stdout.

If NSO is in a hung state or behaving in an abnormal way, status can be found with the command `ncs --status`. This command is run on the NSO server when users are not logged into the NSO CLI.

Sample output of this command is below. This command provides a lot of data about the NSO application installed on this server. The first part of the output of the command provides the version and running status of NSO as seen here:

```
ubuntu@ncs-vm:~$ ncs --status
vsn: 3.4.2.1
SMP support: yes, using 4 threads
Using epoll: yes
available modules: backplane,netconf,cdb,cli,snmp,webui
running modules: backplane,netconf,cdb,cli,snmp,webui
status: started
cluster status:
  mode: none
```

Note: The above output contains only the portion of the command output.

The rest of the output will show the loaded packages, YANG models, callpoints, etc. The key item of data that could indicate a problem would be a lock set on the database. `ncs --status | grep lock` can show this information quickly. If there are locks on the database this is the way to check which part(s) are locked:

```
ubuntu@ncs-vm:~$ ncs --status | grep lock
       no locks set
       no locks set
partial running locks:
partial candidate locks:
partial startup locks:
      read locks: 0
      write lock: unset
      subscription lock: unset
      subscription-lock: false
      read-lock: false
      subscription-lock: false
      read-lock: false
      subscription-lock: false
      subscription-lock: false
      subscription-lock: false
      subscription-lock: false
      read-lock: false
      read-lock: false
```

5.3 Configuration

In the NSO configuration, a "provider" is an organizational unit which groups service resources for each Service Provider tenant comprising a vMS deployment. In the remainder of this chapter, the term provider will be used to reference an individual Service Provider in the context of the NSO configuration.

IP Address Allocation for VNFs

IP addresses can be allocated to an external-facing interface of the VNF either by the NSO or by a virtual infrastructure manager such as OpenStack. When the leaf-list below is set to true, NSO will be used to assign IP addresses to the outside interfaces of the VNF. Available IP addresses are allocated from the resource pools and are identified through the use of tags. When the leaf-list below is set to false, NSO will use IP addresses that are assigned by a virtual infrastructure manager such as OpenStack.

The default value is false.

```
admin@vMS-ncs-sm> show configuration cloudvpn-deployment
allocate-ip-resources false;
[ok][2015-12-08 17:15:19]
```

Global Variables

NSO requires configuration of global variables that are further used in the provisioning process. A snippet of global variable configuration is shown below.

```
admin@vMS-ncs-sm> show configuration globals
any-connect-client-url {
    register-dns-entry false;
    protocol           https;
    append-domain      false;
}
ncs-service-node    10.32.1.101;
ip-address          100.1.1.101;
mgmt-ipv6-type      false;
esc-undep-wait-time 90;
provider Symphony_VNF_P-1B_IntegrationTest {
    variable CPE_ALLOWED_MGMT_NETWORK_AND_WILDMASK {
        value "10.0.0.0 0.255.255.255";
    }
.........................................
.........................................
}
[ok][2015-12-10 20:38:23]
```

All global variables that have to be configured are listed below with a description and sample values in tables 1-7:

- Tables 4.2 and 4.3 contain global variables common to all providers.
- Tables 4.4 through 4.8 contain global variables specific to a provider.
- Global variables marked with "*" are not essential to configure as they have default values.

Table 5.2 Global Variables Common to All Providers

Variable	Description	Value
ncs-service-node	IP address/hostname of the NSO service node.	127.0.0.1
ip-address	pnp server IP-address	10.1.0.0
rest-ssl*	default true (Boolean value)	true
mgmt-ipv6-type*	Flag that indicates the type of Management Network used. **true**: if IPv6 is used (Default) **false**: If IPv4 is used for Management Network type.	true
host-key-verification-type*	**none:** ***This is Default**. Accept any host key. With this setting, no SSH host key verification is done - the key provided by the device or cluster node may be either unknown or different from a 'known' key for the same key algorithm. **reject-mismatch:** Reject host keys that do not match the stored key. With this setting, the SSH host key provided by the device or cluster node may be unknown, but it must not be different from a 'known' key for the same key algorithm. **reject-unknown:** Reject unknown host keys. With this setting, the SSH host key provided by the device or cluster node must already be known.	reject-unknown
cloudvpn-max-retries*	Default value 5. Maximum undeploy-redeploy or redeploy action retries for a given cloudvpn service on ESC events.	5
esc-undep-wait-time*	Sleep time in seconds to wait for ESC to undeploy service and reply with SERVICE_UNDEPLOYED event, before NSO moves ahead with redeploying the service. ***Default value 90**	90
provider	maps to a tenant created in OpenStack	ex: vMS

Table 5.3 Global Variables for Anyconnect Client URL

Variable	Description	Value
register-dns-entry	Default value is set to false. This can be used to control DNS entry. DNS entry will not be made if set to false. Further, when register-dns-entry is false, URL will contain IP-address.	true/false
protocol	Default value is set to "https". The protocol to be used for any-connect client URL https / http.	http/https
append-domain	Default value is set to true. Domain name available in "/glob:globals/provider/variable/VFW_CUSTOMER_DOMAIN_NAME" will be appended to the AnyConnect client URL.	true/false

Table 5.4 Provider-specific Global Variables for CPE

Variable	Description	Sample Value
CPE_CUSTOMER_DNS_1 CPE_CUSTOMER_DNS_2	CPE Domain Name Server's IP address.	172.20.1.21 172.20.1.22
CPE_CUSTOMER_DOMAIN_NAME	CPE domain name. The value should be same as the value of **VR_CUSTOMER_DOMAIN_NAME** of **CSR**.	cloduvpn.com
CPE_MGMT_HUB1	CPE Management Hub1 IP address.	192.168.1.21
CPE_MGMT_HUB2*	CPE Management Hub2 IPaddress. ***If CPE_MGMT_HUB1 is present then this variable need not be set, unless management hub redundancy is required.**	192.168.1.22
CPE_MGMT_LOCAL_KEY	CPE MGMT-OVERLAY local password key.	0fGyjTQrR1qX
CPE_MGMT_REMOTE_IDENTIY	CPE MGMT-OVERLAY remote identity domain name.	cloduvpn.com
CPE_MGMT_REMOTE_KEY	CPE MGMT-OVERLAY remote password key.	plB6GcyLi5aZ
CPE_MGMT_ROUTE	CPE management route IP address.	::/0
CPE_MGMT_TUNNEL_INTERFACE	CPE management Tunnel interface index. In CPE Day 0 config, interface Tunnel**0**	0
CPE_TRUSTED_NETWORKS	CPE WAN interface security access list rule.	any
CPE_NTP_1	NTP servers used by CPE.	192.168.30.21

CPE_NTP_2		192.168.30.22
CPE_HOUR	CPE time zone hour.	1
CPE_ZONE	CPE time zone name.	CET
CPE_MINUTES	CPE time zone minute.	0
CPE_START_TIME	CPE location summer starting time.	02:00
CPE_END_MONTH	CPE location summer ending month.	Mar
CPE_START_MONTH	CPE location summer starting month.	Mar
CPE_START_DAY	CPE location summer starting day.	Sun
CPE_START	CPE location summer starting first/last week of month or the specific week number ranging 1 to 4 of the month.	last
CPE_END_DAY	CPE location summer ending day.	Sun
CPE_END	CPE location summer ending first/last week of month or the specific week number ranging 1 to 4 of the month.	last
CPE_END_TIME	CPE location summer ending time.	03:00
CPE_LOGGING_HOST_1 CPE_LOGGING_HOST_2	CPE logging host IP address or hostname	172.21.1.21 172.21.1.22
CPE_SNMP_COMMUNITY	SNMP community name.	cisco
CPE_SNMP_HOST1	Specify the recipient of an SNMP notification, indicate the interface from which traps are sent, and identify the name and IP address of the NMS or SNMP manager that can connect to the CPE. SNMP host address.	172.22.1.21
CPE_SNMP_LOCATION	SNMP server location.	Customer test lab
CPE_SNMP_NETWORK_WILDMASK	SNMP-ACL standard access list rule permit/deny network and mask.	any
CPE_SNMP_TRAPS	SNMP server enable traps.	config
CPE_SNMP_CONTACT	SNMP server contact information.	cisco
VNF_ALLOWED_MGMT_NETWORK_AND_MASK_1 VNF_ALLOWED_MGMT_NETWORK_AND_MASK_2 VNF_ALLOWED_MGMT_NETWORK_AND_MASK_30	SNMP IPV6 queries restricted to addresses in this range	fd00:0:0:1001::/64 fd00:0:0:1002::/64 fd00:0:0:1030::/64

VNF_ALLOWED_MGMT_NETWORK_AND_INV_MASK_1
VNF_ALLOWED_MGMT_NETWORK_AND_INV_MASK_2
VNF_ALLOWED_MGMT_NETWORK_AND_INV_MASK_30
	SNMP IPV4 queries restricted to addresses in this range	169.254.78.0 0.0.0.255
		169.254.38.0 0.0.0.255
		169.254.30.0 0.0.0.255

Table 5.5 Provider-specific Global Variables for Virtual Router (CSR1000v)

Variable	Description	Sample Value
VR_CUSTOMER_SECRET_PASSWORD_ENCRYPTED	CSR MD5 HASH of privilege exec password	${DS}1${DS}.eRw${DS}3Cfa aRelBM5 6qRWgta7SS0
VR_MANAGEMENT_INTERFACE_NAME	CSR management interface name	Mgmt
VR_CUSTOMER_USERNAME	Customer username.	admin
VR_CUSTOMER_DOMAIN_NAME	Customer domain name. The value should be same as the value of **CPE_CUSTOMER_DOMAIN_NAME** of **CPE**.	cloudvpn.com
VR_CUSTOMER_PASSWORD_ENCRYPTED	CSR MD5 hash of exec mode password	${DS}1${DS}B8ex${DS}LeTn KaOVcJrogiWHiTWeY1
VR_CUSTOMER_DNS_1 VR_CUSTOMER_DNS_2	CSR DNS IP addresses.	172.20.1.21 172.20.1.22
VR_CSR_G2_INTERFACE_NAME VR_CSR_G3_INTERFACE_NAME	GigabitEthernet2 interface description. GigabitEthernet3 interface description.	to-gateway router to-ASAv
VR_NTP_1 VR_NTP_2	CSR Mgmt-intf peer NTP IP address.	192.168.30.21 192.168.30.22
VR_HOUR	CSR time zone hour.	1
VR_ZONE	CSR time zone name.	CET
VR_MINUTES	CSR time zone minute.	0
VR_START_TIME	CSR location summer starting time.	02:00
VR_END_MONTH	CSR location summer ending month.	Mar
VR_START_MONTH	CSR location summer starting month.	Mar
VR_START_DAY	CSR location summer starting day.	Sun
VR_START	CSR location summer starting first/last week of month or the specific week number ranging 1 to 4 of the month.	last
VR_END_DAY	CSR location summer ending day.	Sun
VR_END	CSR location summer ending first/last week of month or the specific week number ranging 1 to 4 of the month.	last

VR_END_TIME	CSR location summer ending time.	03:00
VR_LOGGING_HOST_1 VR_LOGGING_HOST_2	CSR logging host IP addresses.	172.21.1.21 172.21.1.22
VR_SNMP_COMMUNITY	SNMP community name.	public
VR_SNMP_CONTACT	SNMP server contact information.	cisco
VR_SNMP_HOST1	Specify the recipient of an SNMP notification, indicate the interface from which traps are sent, and identify the name and IP address of the NMS or SNMP manager that can connect to the CSR. SNMP server host address.	172.40.1.21
VR_SNMP_LOCATION	SNMP server location.	Customer location
VR_SNMP_NETWORK_WILDMASK	SNMP-ACL standard access list rule permit/deny network and mask.	172.0.0.0 0.255.255.255
VR_SNMP_TRAPS	SNMP server enable traps.	config
VR_MANAGEMENT_PROXY	http proxy IP address	172.100.1.21
VR_MANAGEMENT_PROXY_PORT	http proxy port.	8888
VR_LICENSE_TOKEN	CSR smart license token.	<Smart license token>
VR_INTERNET_MONITORED_IP	CSR internet service monitoring IP address to ping.	8.8.8.8
VR_VNFWAITIME*	CSR wait time to be ready for Day1 and Day2 configurations to be pushed. ***Default is 15 seconds**.	30
VR_SI_LI_FLAG	Flag to turn on/off the LI feature per provider	true/false
VR_INTERCEPT_USER	The LI username	LI_USER
VR_INTERCEPT_GROUP	The group to which the user belongs to	INTERCEPT_GROUP
VR_INTERCEPT_VIEW	The LI view name	INTERCEPT_VIEW
VR_INTERCEPT_PASSWORD	password	INTERCEPT_VIEW

Table 5.6 Provider-specific Global Variables for Virtual Firewall (ASAv)

Variable	Description	Sample Value
VFW_CUSTOMER_USERNAME	Customer username.	admin

VFW_CUSTOMER_PASSWORD_ENCRYPTED	ASA encrypted password.	e1z89R3cZe9Kt6lb
VFW_CUSTOMER_DOMAIN_NAME	Customer domain name.	cloudvpn.com
VFW_CUSTOMER_ENABLED_PASSWORD_ENCRYPTED	ASA privilege exec mode encrypted password.	9jNfZuG3TC5tCVH0
VFW_MANAGEMENT_NTP_SERVER	ASA NTP server IP address.	192.168.30.21
VFW_MANAGEMENT_TIMEZONE	ASA time zone.	CET 1
VFW_MANAGEMENT_SNMP_SERVER	Specify the recipient of an SNMP notification, indicate the interface from which traps are sent, and identify the name and IP address of the NMS or SNMP manager that can connect to the ASA. SNMP server management host address.	172.40.1.21
VFW_MANAGEMENT_SNMP_LOCATION	SNMP server location.	Customer location
VFW_MANAGEMENT_SNMP_CONTACT	SNMP server contact information.	contact
VFW_MANAGEMENT_SNMP_COMMUNITY	SNMP server community name.	public
VFW_MANAGEMENT_DNS_SERVER	ASA domain name server address.	172.20.1.21
VFW_MANAGEMENT_PROXY	ASA license http proxy address.	172.100.1.21
VFW_MANAGEMENT_PROXY_PORT	ASA license http proxy port.	8888
VFW_LICENSE_TOKEN	ASA smart license token.	<Smart License token>
VFW_CUSTOMER_DNS_SERVER_1 VFW_CUSTOMER_DNS_SERVER_2	ASA domain name server address.	172.20.1.21 172.20.1.22
VFW_G0_INTERFACE_NAME VFW_G1_INTERFACE_NAME	ASA inside interface. ASA outside interface.	inside outside
VFW_SSL_REMOTE_POOL	SSL VPN user address pool IP.	172.31.248.1-172.31.255.255
VFW_SSL_REMOTE_POOL_MASK	SSL VPN user address pool mask.	255.255.248.0

VFW_SSL_CERT_NAME	ASA SSL certificate subject name.	cn=SSL VPN,ou=ne8,c=DE
VFW_PKCS12_CERT	Signed certificate from a certificate authority	<CA signed certificate>
VFW_PKCS12_PASSWORD	Password for signed certificate from CA	
VFW_SSL_CA_TRUSTPOINT_NAME	Trustpoint name for the SSL VPN	VMS
VFW_SSL_DNS_SERVER1 VFW_SSL_DNS_SERVER2	SSL DNS IP addresses.	172.20.1.21 172.20.1.22
VFW_RULE_NAME	Firewall rule name	outside-to-inside
VFW_VNFWAITIME*	ASA wait time to be ready for Day 1 and Day 2 configurations to be pushed. ***Default is 15 seconds**.	30
VFW_ANYCONNECT_LINUX_IMG	Points to anyconnect linux image on the device	" disk0:/anyconnect/anyconnect-linux-64-4.1.06020-k9.pkg 3"
VFW_ANYCONNECT_MACOSX_IMG	Points to anyconnect Mac image on the device	" disk0:/anyconnect/anyconnect-macosx-i386-4.1.06020-k9.pkg 2"
VFW_ANYCONNECT_WIN_IMG	Points to anyconnect windows image on the device	" disk0:/anyconnect/anyconnect-win-4.1.06020-k9.pkg 1"

Table 5.7 Provider specific global variables for virtual Web security appliance (WSAv)

Variable	Description	Sample Value
WSA_DNS_1 WSA_DNS_2	WSA DNS IPs.	172.20.1.21 172.20.1.22
WSA_CPE_ROUTE_1 WSA_CPE_ROUTE_2 WSA_CPE_ROUTE_3	WSA Data routing table routing destination addresses.	10.0.0.0/8 172.16.0.0/12 192.168.0.0/16
WSA_CPE_ROUTE_NAME_1 WSA_CPE_ROUTE_NAME_2 WSA_CPE_ROUTE_NAME_3	WSA Data routing table route names.	RFC1918-1 RFC1918-2 RFC1918-3
WSA_USERNAME	WSA username	admin
WSA_ENCRYPTED_ADMIN_PASSWORD	WSA admin encrypted password.	${DS}1${DS}h1VxC40U${DS}u f2qLUwGTjHgZplkP78xA.
WSA_CUSTOMER_DOMAIN_NAME	WSA customer domain	cloudvpn.com

	name	
WSA_LICENSE_FILE	WSA license file path.	/home/localadmin/wsa-license.txt
WSA_VNFWAITIME*	Time in seconds NSO waits for WSA to setup Day2 configuration (URL filters / security policies) ***Default is 480 seconds**.	240
WSA_HOSTNAME_DATA	WSA P1 interface host name.	wsa-01-data.cloudvpn.com
WSA_HOSTNAME_MGMT	WSA Management interface host name for IPV4.	wsa-01-mgmt.cloudvpn.com
WSA_SNMP_COMMUNITY	WSA SNMP trap community encrypted name.	$4$Z6rqKkjTRQ+0Rs4icifw4A==
WSA_SNMP_CONTACT	WSA SNMP contact information.	cisco
WSA_SNMP_LOCATION	WSA SNMP location information	vMS test bed
WSA_TIMEZONE	WSA local time zone.	GMT-4
VNF_ALLOWED_MGMT_NETWORK_AND_PREFIX_1 VNF_ALLOWED_MGMT_NETWORK_AND_PREFIX_2 VNF_ALLOWED_MGMT_NETWORK_AND_PREFIX_30	IPv4 and IPv6 prefixes that are allowed to connect to WSA Management IP address.	10.0.0.0/8 10.1.0.0/8 " "

Table 5.8 Provider-specific global variables for Service assurance

Variable	Description	Sample Value
VFW_LOGGING_HOST	Firewall logging host name	localhost
VFW_SNMP_HOST	Firewall SNMP host IP address	10.10.10.1
USER_DEFINED_ID	User defined identifying string for CPE	VNF_CPE

Infrastructure

Network resources that are managed by a virtual infrastructure manager such as OpenStack are specified in the configuration below.

```
admin@vMS-ncs-sm> show configuration infrastructure
esc esc-device {
    network vnf-management {
        network-type mgmt;
    }
    network vnf-outside {
        network-type outside;
    }
    is-active true;
}
[ok][2015-12-08 16:17:55]
```

Resource Pools

NSO manages multiple IP-address-pools that are used during service chain provisioning.

- **CPE-Superpool**: This pool specifies the LAN IP subnet block that will be assigned to a LAN-facing interface on a CPE.
- **br-inside:** This pool specifies the IP addresses that are assigned to inside interfaces of VNFs
- **csr-to-csr:** This pool specifies IP addresses that are assigned to tunnel endpoints between virtual routers(CSR1000v) in a dual data center deployment.
- **loopback:** This pool specifies IP addresses that are assigned to the tunnel endpoints between CPE and virtual router (CSR1000v).

NSO resource pools can be shown in NSO CLI by executing the following command.

```
admin@vMS-ncs-sm> show configuration resource-pools
ip-address-pool CPE-Superpool {
    owner Shared;
    tags  [ cpe-lan-pool ];
    subnet 10.128.0.0 16;
}
ip-address-pool br-inside {
    owner Shared;
    tags  [ br-inside esc-openstack ];
    subnet 10.192.0.0 10;
}
ip-address-pool csr-to-csr {
    owner Shared;
    tags  [ csr-csr ];
    subnet 10.64.0.0 10;
}
ip-address-pool loopback {
    owner Shared;
    tags  [ cpe-csr loopback ];
    subnet 10.0.0.0 11;
}
[ok][2015-12-08 16:24:32]
```

NSO has the capability to assign IP addresses to virtual instances spun up on a compute device from IP-address-pools. Tags are used to identify such pools. Individual IP addresses can also be excluded from the subnet assigned to such IP pools.

Following output shows NSO CLI command that lists IP address pools on a specific compute node.

```
admin@vMS-ncs-sm> show configuration resource-pools ip-address-pool compute1
    ip-address-pool compute1 {
        owner vMS;
        tags [ br-outside-compute1 ];
        subnet 172.32.100.0 24;
        exclude 172.32.100.0 32;
        exclude 172.32.100.51 32;
        exclude 172.32.100.75 32;
        exclude 172.32.100.185 32;
        exclude 172.32.100.255 32;
    }
}
```

Occupancy

Occupancy container reflects the organizational structure and is hierarchical. Resources such as infrastructure and resource pools are attached to occupancy structure.

```
admin@vMS-ncs-sm> show configuration occupancy
provider vMS-lab {
    vnfm esc-device;
    ip-resource-pool CPE-Superpool;
    ip-resource-pool br-inside;
    ip-resource-pool csr-to-csr;
    ip-resource-pool loopback;
}
[ok][2015-12-08 16:19:40]
```

Day 0 Server

NSO hosts a Day 0 configuration server which enables southbound devices to fetch initial configurations that are required during VNF boot up. This is required by the ESC to boot new VNFs.

```
admin@vMS-ncs-sm> show configuration day0
```

```
server {
    port    4343;
    use-ssl true;
}
[ok][2015-12-08 16:02:34]
```

Plug-n-Play(PnP)

NSO hosts a PnP server which enables Cisco devices running a PnP client agent to communicate with the server. The configuration below is required to enable the PnP server with proper functionality.

PnP Server

The following NSO CLI command lists PnP Server configuration.

```
admin@vMS-ncs-sm> show configuration pnp server
port    443;
use-ssl true;
[ok][2015-12-08 16:08:02]
```

PnP Common Configuration

PnP makes use of cisco-ios NED to configure Cisco ISR CPEs. The configuration below specifies NSO to use cisco-ios NED.

```
admin@vMS-ncs-sm> show configuration pnp ned-map
ned-map "Cisco IOS" {
    device-type cli;
    ned-id      cisco-ios;
}
[ok][2015-12-08 16:12:12]
```

The configuration below specifies the start of the management tunnel IP address range to be assigned to CPEs. For every subsequent CPE management tunnel, the next IP address from this range will be used.

```
admin@vMS-ncs-sm> show configuration pnp mgmt-address-start
mgmt-address-start 10.254.0.2;
[ok][2015-12-08 16:12:18]
```

NSO will generate random passwords (exec mode and privilege exec mode) for CPEs when the below configuration is set to true. Default setting is false.

```
admin@vMS-ncs-sm> show configuration pnp random-passwords
```

```
random-passwords false;
[ok][2015-12-08 16:12:22]
```

Alternately, NSO will use the password configured (exec mode and privileged exec mode) below to login to the CPE when random-passwords are not set.

```
admin@vMS-ncs-sm> show configuration pnp cpe-password
cpe-password FiHM2U83gNRtarj;
[ok][2015-12-08 16:12:29]
```

PnP Interface Map

PnP interface-map specifies the WAN-facing interface and LAN-facing interface for different models of Cisco ISR CPEs. PnP interface-map configuration is shown in the output below.

```
admin@vMS-ncs-sm> show configuration pnp interface-map
interface-map "(C19[0-9][0-9])|(CISCO19[0-9][0-9])" {
    wan GigabitEthernet0/1;
    lan GigabitEthernet0/0;
}
interface-map "(C29[0-9][0-9])|(CISCO29[0-9][0-9])" {
    wan GigabitEthernet0/1;
    lan GigabitEthernet0/0;
}
interface-map "(C39[0-9][0-9])|(CISCO39[0-9][0-9])" {
    wan GigabitEthernet0/1;
    lan GigabitEthernet0/0;
}
interface-map C819 {
    wan GigabitEthernet0;
    lan FastEthernet0;
}
interface-map C881 {
    wan FastEthernet4;
    lan FastEthernet0;
}
interface-map C891 {
    wan GigabitEthernet0;
    lan FastEthernet0;
}
[ok][2015-12-08 16:08:56]
```

PnP Logging

PnP messages exchanged between server and client agent can be enabled for each CPE by specifying the serial number of a particular CPE or for all CPEs.

```
admin@vMS-ncs-sm> show configuration pnp logging
enabled;
directory /var/log/ncs;
serial    all;
[ok][2015-12-08 16:10:39]
```

PnP Restore Configuration

PnP can restore Day -1 configuration on a CPE by copying a file from flash directory to `startup-config` on CPE and reloading the CPE. Default value for this is `flash:day--1-config`. This can be changed as shown in the example below. All CPEs must have a file in the flash directory containing a valid Day -1 configuration.

```
admin@ncs-vm> show configuration pnp config-restore-file
config-restore-file flash:day--1-config;
[ok][2015-12-10 14:36:24]
admin@ncs-vm>
  admin@ncs-vm> show configuration pnp config-restore-file
config-restore-file flash:ZTD-config;
[ok][2015-12-10 14:37:25]
admin@ncs-vm>
```

High-Availability

This section outlines NSO configuration to provide High Availability (HA) through 1:1 NSO redundancy. Relevant NSO configuration is shown below.

```
admin@ncs-vm-1> show configuration ha-cluster
  cluster-name HA-CLUSTER;
  auto-start disable;
  failure-limit 2;
  bgp-anycast-failover {
   bgp-anycast-prefix 2a00:a24:b5fe:fff::ab01/128;
   bgp-anycast-path-min 2;
   bgp-clear-enabled true;
  }
  members ncs-vm-1 {
   address 2a00:a24:b5e4:17::1:1;
   role slave;
   transition-enabled true;
   bgp-anycast true;
   master-capable true;
   device devs-quagga;
  }
  members ncs-vm-2 {
   address 2a00:a24:b5e2:17::1:2;
   role master;
   bgp-anycast true;
   device devm-quagga;
  }
```

Devices

Authgroups

Authgroups are used to store the access credential required to access the devices (ESC, ASAv, CSR1000v, WSAv, quagga etc) from NSO.

A list of configured devices can be seen by executing the following NSO CLI command.

```
admin@vMS-ncs-sm> show configuration devices authgroups
group asa {
    default-map {
        remote-name               admin;
        remote-password           $4$e+nZKEjGLkzRNfZ7Q02k4A==;
        remote-secondary-password $4$IpbEJGf3dGfwLPlNS1OG4A==;
    }
}
group csr {
    default-map {
        remote-name               admin;
        remote-password           $4$bZbNFkzkHXnbdapyPRKS4A==;
        remote-secondary-password $4$fevYH1fZFTb3MZQUPn/C4A==;
    }
}
group dnsupdater {
    default-map {
        remote-name cvpn-zones-maintainer;
        remote-password $4$c7H7dxODRQ+0Rs4icifw4A==;
    }
}
group esc {
    default-map {
        remote-name     admin;
```

```
            remote-password $4$JL3bDkWDfEXidJloAE6q4A==;
        }
    }
    group wsa {
        default-map {
            remote-name     admin;
            remote-password $4$e6n7A1Pkc17RLYhNGn2T0pk2oMA9yd7XaQ9+j7/Ctwo=;
        }
    }
    group quaagaAuthGrp {
        default-map {
            remote-name           admin;
            remote-password       $4$bbrqNECwRQ+0Rs4icifw4A==;
            remote-secondary-password $4$bbrqNECwRQ+0Rs4icifw4A==;
        }
    }
    [ok][2015-12-08 16:41:34]
```

ESC

This section outlines the configuration required to add ESC as a device managed by NSO. The IP address of the ESC, port, authgroups and device state are required configuration to add ESC as a device. ESC device state should be unlocked from an NSO perspective.

```
admin@vMS-ncs-sm> show configuration devices device esc-device
 device esc-device {
    address 10.60.73.137;
    port  830;
    authgroup esc;
    device-type {
    netconf;
    }
    trace pretty;
    state {
    admin-state unlocked;
    }
}
```

Quagga

This section outlines the configuration to add quagga instances on NSO nodes which are part of HA mode. IP address, port, authgroup, NED and the protocol are required configuration to add a quagga device. Quagga configuration of both nodes in HA should exist on both NSOs. All quagga device states should be unlocked from an NSO perspective.

```
admin@vMS-ncs-sm> show configuration device device srvm-quagga
device srvm-quagga {
       address 2003:1f0b:ffe5:101::1:10;
       port    2605;
       description srvm-quagga;
       authgroup quaagaAuthGrp;
       device-type {
           cli {
               ned-id quagga-bgp;
               protocol telnet;
            }
        }
       state {
           admin-state unlocked;
        }
    }
   device srvs-quagga {
       address 2003:1f0b:ffe4:101::1:15;
       port    2605;
       description srvs-quagga;
       authgroup quaagaAuthGrp;
       device-type {
           cli {
               ned-id quagga-bgp;
               protocol telnet;
            }
        }
       state {
           admin-state unlocked;
        }
    }
```

DNS Updater

NSO provides the functionality to update DNS entries on a DNS server when new service chains are created.

To enable NSO to provide this functionality, a DNS server is added as a device managed by the NSO. IP address, ports, authgroups and dns-updater NED are the required configuration for the functionality to be enabled. The DNS device state should be unlocked from an NSO perspective.

```
admin@vMS-ncs-sm> show configuration devices device dns
     device dns {
        address 172.20.1.21;
        port  22;
        authgroup dnsupdater;
        device-type {
            ncs:generic {
               ncs:ned-id dns-updater;
            }
        }
        ncs:state {
            ncs:admin-state unlocked;
        }
    }
```

The dns-updater will invoke scripts to update and delete DNS entries on the DNS server. Default scripts invoked by dns-updater are `./update-name.sh` and `./delete-name.sh`. The file name for the scripts can be changed as shown below. NSO will only invoke the script. The administrator is responsible to verify that scripts exist on the DNS server, consume the parameters provided by NCS, and update/delete the DNS entries correctly.

```
admin@vMS-ncs-sm> show configuration devices device dns update-cmd
update-cmd ./update-name.sh;
[ok][2015-12-10 14:50:04]
admin@vMS-ncs-sm> show configuration devices device dns delete-cmd
delete-cmd ./delete-name.sh;
[ok][2015-12-10 14:50:26]
    admin@vMS-ncs-sm> show configuration devices device dns update-cmd
update-cmd ./add-dns.sh;
[ok][2015-12-10 14:52:40]
admin@vMS-ncs-sm> show configuration devices device dns delete-cmd
delete-cmd ./del-dns.sh;
[ok][2015-12-10 14:52:50]
```

ESC Day 0

Out-of-band Image and Flavor Management

Images and Flavors are pre-created in a hosted environment and NSO will treat them as unmanaged images and flavors. Out-of-band image and flavor configuration is shown below.

```
admin@ncs-vm> show escday0 image-registration
escday0 {
    image-registration vFirewall 1.2 {
        image-id <Image UUID in openstack>;
        flavor-id <Flavor UUID in openstack>;
        unmanaged;
        devices [ esc-device ];
    }
    image-registration vRouter 1.2 {
        image-id <Image UUID in openstack>;
        flavor-id <Flavor UUID in openstack>;
        unmanaged;
        devices [ esc-device ];
    }
    image-registration vWSA 1.2 {
        image-id <Image UUID in openstack>;
        flavor-id <Flavor UUID in openstack>;
        unmanaged;
        devices [ esc-device ];
    }
}
```

In-band Flavor and Image Management

Tenants, images and flavors are created in-band when managed by the NSO. The configuration is shown below.

```
admin@vMS-ncs-sm> show configuration escday0
  escday0 {
    tenants;
    registration 1.1 {
        csr-url https://[IP address]/images/csr_latest.qcow2;
        asa-url https://[IP address]/images/asa_latest.qcow2;
        ewsa-url https://[IP address]/images/wsa_latest.qcow2;
    }
}
```

Cluster

Configuration

The NSO has the ability to distribute management of devices across multiple NSO instances. This is called clustering.

IP address, port and authgroup configurations are required to establish clustering.

```
vMS-ncs-sm>show configuration cluster
cluster {
    global-settings {
        caching {
            full-device-config {
                max-devices 100;
            }
        }
    }
    remote-node vMS-ncs-dm {
        address 10.60.73.136;
        port  830;
        authgroup NSOcluster;
        trace pretty;
    }
    authgroup ncscluster {
        default-map {
            remote-name admin;
            remote-password $4$cbTAcEeBK1vWdv1sPx634A==;
        }
    }
}
```

Map NSO Device Node with ESC

VNFs that are spun up by Elastic Service Controllers are managed by the NSO device node. Multiple ESCs can be managed by an NSO device node.

```
admin@vMS-ncs-sm> show configuration esc-device-ncs-map
esc-device-ncs-map esc-dc1 {
    device-ncs vMS-ncs-dm;
}
esc-device-ncs-map esc-dc2 {
    device-ncs vMS-ncs-dm;
}
[ok][2015-12-08 17:13:22]
```

Logging

CloudVPN Logs

CloudVPN service logging is enabled with the configuration below.

```
admin@vMS-ncs-sm> show configuration logging
enabled;
directory /var/log/ncs/cloudvpn-logs;
[ok][2015-12-08 16:57:13]
```

NSO Java Logs

NSO Java logs with different log level settings are enabled with the configuration below.

```
admin@vMS-ncs-sm> show configuration java-vm
stdout-capture {
    file /var/log/ncs/ncs-java-vm.log;
}
java-logging {
    logger com.cisco {
        level level-all;
    }
    logger com.cisco.perfdata {
        level level-warn;
    }
    logger com.tailf.ns.tcm {
        level level-all;
    }
}
[ok][2015-12-08 16:57:20]
```

5.4 Operations

Start, Stop, Restart NSO

The NSO runs as a process under the host Linux OS and can be started, restarted and stopped using standard Linux commands. In both Ubuntu and RedHat use the `service` command to run a System V init script under `/etc/init.d`.

E.g. `service ncs [start | restart | stop | status ]`

SSL Certificates

The NSO runs two web services (HTTP/HTTPS) for the Zero Touch Deployment (PnP) service and the Day 0 configuration service. Although these servers can be run as plain HTTP, it is recommended to enable HTTPS for both.

For the PnP service, HTTPS is enabled in the `pnp` configuration section of the NSO as shown below.

```
admin@ncs-vm> show configuration pnp server
port    443;
use-ssl true;
```

For the Day 0 service, HTTPS is enabled in the `day0` configuration section of the NSO as shown below:

```
admin@ncs-vm> show configuration day0
server {
    port    443;
    use-ssl true;  }
```

The `/etc/ncs/ncs.conf` file defines where the required certificate files are located for SSL. By default, the NSO will use the self-signed certificates found in `${NCS_CONFIG_DIR}/ssl/cert/` which translates to `/etc/ncs/ssl/cert`. For a production system, it is highly recommended that these certificates be replaced by actual CA signed certificates.

```
ubuntu@NSO-vm:~$ ls -l /etc/ncs/ssl/cert/*2.*
-rw-r--r-- 1 root root 1822 Nov 26 11:32 /etc/ncs/ssl/cert/host2.ca-cert
-rw-r--r-- 1 root root 1822 Nov 26 11:32 /etc/ncs/ssl/cert/host2.cert
-rw-r--r-- 1 root root 1651 Nov 26 11:32 /etc/ncs/ssl/cert/host2.csr
-rw-r--r-- 1 root root 3243 Nov 26 11:32 /etc/ncs/ssl/cert/host2.key
```

Either overwrite the existing cert files with CA-signed certs, or change the path in the `ncs.conf` file to point to the new certificate files. Once the PnP server is enabled for HTTPS, a CA-signed certificate needs to be imported to the CPE Day -1 configuration. The final certificate that is imported into CPE Day -1 configuration has to be in Base64-encoded PKCS12 format. If the input certificate is in PEM format, OpenSSL tool can be used to convert it to Base64-encoded PKCS12 certificate format.

An example of how a certificate might be converted to this format can be found below.

```
  openssl pkcs12 -export -out ios-client.p12 -in ios-client.crt -inkey ios-client.key -chain -
CAfile ca-all.crt -password pass:cisco123
  openssl base64 -e -in ios-client.p12 -out ios-client.p12.b64
```

Then on the IOS CPE device, create a trustpoint and import the certificate:

```
router#conf t
router(config)#crypto pki import NSO-PNP pkcs12 terminal password cisco123
    <-| paste-contents-of cunningr.p12.b64 or file |->
  quit
router(config)#crypto pki trustpoint NSO-PNP
router(config)#revocation-check none
```

The resulting IOS configuration should be included as part of the CPE Day -1 configuration and deployed to all existing CPEs.

IP Address Allocation

NSO manages IP address pools used for provisioning services. For example, in a CloudVPN service, the NSO can be used to manage the public IP address pools used for the VNFs, as well as addressing used for internal IP addressing of the service (e.g. CPE Management Loopback and br-inside).

The commands below can be used to display current IP pools managed by NSO and any allocations that have been made.

Show Available Address Pools

```
admin@ncs-vm> show resource-pools ip-address-pool
NAME           ID             ERROR  SUBNET         FROM
-------------------------------------------------------------------
CPE-Superpool
br-inside      my-first-cvpn  -      10.192.0.0/24  10.192.0.0/10
               my-new-cvpn    -      10.192.1.0/24  10.192.0.0/10
csr-to-csr
loopback
```

```
[ok][2015-12-08 16:11:17]
```

Show Specific IP Allocations

```
admin@ncs-vm> show ip-allocator pool allocation
NAME       ADDRESS     CIDRMASK  OWNER                                REQUEST ID
-----------------------------------------------------------------------------------
br-inside  10.192.0.0  24        /cloudvpn:cloudvpn{my-first-cvpn}  my-first-cvpn
           10.192.1.0  24        /cloudvpn:cloudvpn{my-new-cvpn}    my-new-cvpn

[ok][2015-12-08 16:11:37]
```

Show Unallocated Prefixes

```
admin@ncs-vm> show ip-allocator pool available
NAME           ADDRESS       CIDRMASK
---------------------------------------
CPE-Superpool  10.128.0.0    10
br-inside      10.192.2.0    23
               10.192.4.0    22
               10.192.8.0    21
               10.192.16.0   20
               10.192.32.0   19
               10.192.64.0   18
               10.192.128.0  17
               10.193.0.0    16
               10.194.0.0    15
               10.196.0.0    14
               10.200.0.0    13
               10.208.0.0    12
               10.224.0.0    11
csr-to-csr     10.64.0.0     10
loopback       10.0.0.0      11

[ok][2015-12-08 16:12:45]
```

View the Resource Pool Configurations

```
admin@ncs-vm> show configuration resource-pools
ip-address-pool CPE-Superpool {
    owner Shared;
    tags  [ cpe-lan-pool ];
    subnet 10.128.0.0 10;
}
ip-address-pool br-inside {
    owner Shared;
    tags  [ br-inside esc-openstack ];
    allocation my-first-cvpn {
        allocating-service /cloudvpn[name='my-first-cvpn'];
        request {
            subnet-size 24;
        }
    }
    allocation my-new-cvpn {
        allocating-service /cloudvpn[name='my-new-cvpn'];
        request {
            subnet-size 24;
        }
    }
    subnet 10.192.0.0 10;
}
ip-address-pool csr-to-csr {
    owner Shared;
    tags  [ csr-csr ];
    subnet 10.64.0.0 10;
}
ip-address-pool loopback {
    owner Shared;
    tags  [ cpe-csr loopback ];
    subnet 10.0.0.0 11;
}
[ok][2015-12-08 16:13:05]
```

In some environments, it is necessary to manage IP address allocation based on specific compute-hosts of the underlying virtual infrastructure. In this case, the actual name given to the pool is relevant and will describe the compute-host by its host name and the specific neutron network to which the resource-pool belongs.

For example the definition below shows a resource pool for the neutron network "net_vnf_outside" on compute-host "ny-dc-1-us".

```
ip-address-pool ny-dc-1-us-net_vnf_outside {
    owner VMS;
    tags  [ br-outside-ny-dc-1-us ];
    subnet 20.100.0.0 24;
    exclude 20.100.0.1 32;


}
```

Service Chain Status Verification

The status of the CloudVPN Service Chains can be verified using the `show cloudvpn-data devices-list` as shown below. This command also shows the status of each device.

```
admin@NSO-vm> show cloudvpn-data device-list
NAME           DEVICE                REMOTE  READY           PROVISIONED
---------------------------------------------------------------------------
my-first-cvpn  my-first-cvpn-ASA-esc  local   ready           true
               my-first-cvpn-CSR-esc  local   ready           true
               my-first-cvpn-WSA-esc  local   ready           true
               cpe-9CANYW0SMDU        local   ready           true

[ok][2015-12-07 22:16:04]
```

Column explanation is as follows:

- NAME - The full unique name of the CloudVPN service.
- DEVICE - The name of the individual devices that make up the CloudVPN service.
- REMOTE - Whether or not the device is local or remote to this NSO instance (See NSO clustering). If it is not local, the name of the remote NSO device node will be displayed here.
- READY - The current status of the device in NSO. Other values here could be "not-existing" or "not-ready".
- PROVISIONED - Shows the current provisioning status within the service e.g "true" or "false".

Service Chain Operational Data

The operational data relating to a CloudVPN service with the command `show cloudvpn-oper`, as shown in the output below.

```
admin@ncs-vm> show cloudvpn-oper my-first-cvpn
cloudvpn-oper my-first-cvpn
 counter              10
 current-state        CLOUDVPN_PROVISION_WSA
 state-details        "Caused By:Provisioning WSA"
 last-redeploy-reason CLOUDVPN_PROVISION_VFIREWALL
 waiting-redeploy     yes
 vnfs-compute-host esc:jungfrau
 image-registration-version vFirewall
  version 1.1
 image-registration-version vRouter
  version 1.1
 image-registration-version vWSA
  version 1.1
 escList esc
```

The "current-state" and "state-details" show the last reported status from the provisioning workflow. Since the WSAv is usually the last device to get provisioned in a CloudVPN service, "CLOUDVPN_PROVISION_WSA" will normally be displayed for a fully provisioned CloudVPN service.

HA Status

In order to check the current HA status for an NSO HA pair use the `show ha-cluster status`:

```
admin@ncs-vms102> show ha-cluster members
NAME                      STATUS
----------------------------------
svz-op-bdc-NSOf-1-vms102  slave
szg-dr-bdc-NSOf-1-vms102  master
```

Cluster Status

To check the status of the NSO device nodes in a cluster and all the associated devices, the `show cluster remote-node` command should be used.

```
  ubuntu@ncs-vm> show cluster remote-node
                         LAST   RECEIVED
  NAME     NAME  STATUS  EVENT  NOTIFICATIONS  NAME
  ---------------------------------------------------------------------------------------------
-----------------
  TEXAS                                     my-first-cvpn-ASA-esc
                        my-first-cvpn-CSR-esc
                      my-first-cvpn-WSA-esc
                                              cpe-FTX1822829Y
  TEXAS                                     my-second-cvpn-ASA-esc
                       my-second-cvpn-CSR-esc
                         my-second-cvpn-WSA-esc
                                              cpe-FTX1822836B
```

ESC Status

The current operational state of ESC can be verified with the `show devices device esc state` command.

```
admin@ncs-vm> show devices device esc state
state oper-state    enabled
state transaction-mode running-only
state last-transaction-id 1448-462418-418575
[ok][2015-12-07 23:13:55]
```

If the status is showing "enabled" the ESC device is up and running.

CPE Status [PnP]

The status of each CPE device can be checked by the following NSO CLI command. Replace `$SERIAL_NUMBER` with the actual Serial Number of CPE device.

```
admin@ncs-vm> show pnp-state device $SERIAL_NUMBER
pnp-state device 9AR13N53RHC
 udi           PID:ISR881,VID:V00,SN:9AR13N53RHC
 device-info   15.6(20150712:154917)
 ip-address    192.168.11.16
 mgmt-ip       10.254.0.10
 port          22
 name          cpe-9AR13N53RHC
 username      ""
 password      ""
 sec-password  ""
 salt          ""
 remote-node   ""
 wan-interface GigabitEthernet1
 lan-interface GigabitEthernet2
 configured    false
 request       device-info
 added         false
 synced        false
 is-netsim     false
 need-clean    false
 pending-exec  ""
 last-contact  2015-12-09 23:58:50
 last-clean    1449705525
[ok][2015-12-11 00:01:08]
```

NSO Alarms

NSO Alarms can be used for troubleshooting basic NSO related issues.

To view NSO alarm summary `show alarms summary` can be used command as shown in the example output below.

```
  admin@ncs-vm> show alarms summary
  alarms summary indeterminates 0
  alarms summary criticals 0
  alarms summary majors 3
  alarms summary minors 0
  alarms summary warnings 0
```

To view NSO alarms in more detail, the `show alarms alarm-list` command should be used.

```
  admin@ncs-vm> show alarms alarm-list
  alarms alarm-list number-of-alarms 8
  alarms alarm-list last-changed 2015-12-01T16:06:52.41966+00:00
  alarms alarm-list alarm cpe-9CANYW0SMDU connection-failure /devices/device[name='cpe-
9CANYW0SMDU'] ""
   is-cleared              true
   last-status-change      2015-12-01T16:00:10.704501+00:00
   last-perceived-severity major
   last-alarm-text         "Failed to connect to device cpe-9CANYW0SMDU: connection refused:
The kexTimeout (20000 ms) expired."
   status-change 2015-12-01T15:53:48.471327+00:00
    received-time      2015-12-01T15:53:48.471327+00:00
    perceived-severity major
    alarm-text         "Failed to connect to device cpe-9CANYW0SMDU: connection refused: The
kexTimeout (20000 ms) expired."
   status-change 2015-12-01T16:00:10.704501+00:00
    received-time      2015-12-01T16:00:10.704501+00:00
    perceived-severity cleared
    alarm-text         "Connected as admin"
```

Dual-DC Service Chain Provisioning

NSO can support multiple vm-managers (ESC) and distribute CloudVPN deployments across multiple data centers. When NSO is configured to do this it can be seen which ESC was used last using the command below.

ESC selection uses the round-robin algorithm.

```
admin@ncs-vm> show provider-esc-selection
                        LAST
                        SELECTED
PROVIDER  LAST          REDUNDANT
ID        SELECTED ESC  ESC
-----------------------------------
VMS        intra-ht-ny   -
```

Requesting SSH Host Keys for NSO Devices

When a vMS orchestration component is added, NSO needs to retrieve its SSH host keys before being able to access to it. The following output shows how to request SSH host keys for a newly deployed ESC.

```
admin@ncs-vm> request devices device esc ssh fetch-host-keys
result changed
fingerprint {
algorithm ssh-dss
value ff:d2:cc:da:b9:99:50:db:a6:56:f5:5f:af:ec:a1:fb
}
[ok][2015-12-10 15:42:05]
```

After the new SSH host keys are retrieved, it is recommended to check if NSO is able to connect to the device, as shown in the example below.

```
admin@ncs-vm> request devices device esc connect
result true
info (admin) Connected to esc - 192.168.15.20:830
[ok][2015-12-10 15:42:17]
```

5.5 ESC Registration

ESC must be registered with NSO in order to be able to deploy Service Chains. It follows the sequence of the Figure 2.4 "ESC registration call flow" in section 2.4 "vMS Call Flows"<refdone>.

First check the ESC device settings in the NSO.

```
admin@vms-ncs-sm> show configuration devices device esc-device address
address 10.60.73.137;
[ok][2015-12-08 12:33:12]
admin@vms-ncs-sm> show configuration devices device esc-device port
port 830;
[ok][2015-12-08 12:33:17]
admin@vms-ncs-sm> show configuration devices device esc-device remote-node
No entries found.
[ok][2015-12-08 12:33:27]
admin@vms-ncs-sm> show configuration devices device esc-device authgroup
authgroup esc;
[ok][2015-12-08 12:33:45]
admin@vms-ncs-sm> show configuration devices authgroups group esc
default-map {
    remote-name     admin;
    remote-password XXXXYYYYYZZZZZ;
}
[ok][2015-12-08 12:36:22]
```

The log file `/var/log/ncs/netconf-esc-device.trace`, should show an outbound hello and a return hello with capabilities, as shown in the section below.

```
>>>>out 23-Nov-2015::20:34:22.181 device=esc-device
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.0</capability>
    <capability>urn:ietf:params:netconf:base:1.1</capability>
  </capabilities>
</hello>
<<<<in 23-Nov-2015::20:34:22.207 device=esc-device
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.0</capability>
    <capability>urn:ietf:params:netconf:base:1.1</capability>
```

5.6 Backup and Restore

NSO Backup

The NSO stores all its configuration in the Configuration Database (CDB) which is located in `/var/opt/ncs/cdb` directory.

All parts of the NSO installation can be backed up with standard file system backup procedures. The following sections discuss the overall backup procedure and provide recommendations for daily cron jobs to automate the backup task.

The recommended way to do a backup is to use the `ncs-backup` script (utility) located in the `/opt/ncs/current/bin` directory.

The backup files will be saved in the directory `/var/opt/ncs/backups` and will be in the format `<ncs-build-name>@<date>.backup`.

```
ubuntu@ncs-vm:~$ sudo /opt/ncs/current/bin/ncs-backup
INFO  Backup /var/opt/ncs/backups/ncs-3.4.2.1@2015-12-08T14:36:58.backup created successfully
ubuntu@nso-vm:~$ ls -la /var/opt/ncs/backups/
total 115452
drwxr-xr-x  2 root root      4096 Dec  8 14:36 .
drwxr-xr-x 10 root root      4096 Dec  8 14:36 ..
-rw-r--r--  1 root root   1730560 Nov 26 11:38 ncs-3.4.2.1@2015-11-26T11:38:07.backup
-rw-r--r--  1 root root 116480000 Dec  8 14:36 ncs-3.4.2.1@2015-12-08T14:36:58.backup
```

In order to take regular backups every 24 hours, for example, the administrator can set the above command as a cron job.

It is highly recommended that these backup files be copied to a storage location with disaster recovery in the event that the NSO VM is lost entirely. It is also highly recommended that old backup files be removed regularly to prevent unnecessary usage and potential system resource deadlock. A similar script to the one shown below can be used to achieve this.

```
#!/bin/sh

/opt/ncs/current/bin/ncs-backup

find /var/opt/ncs/backups/* -mtime +15 -exec rm {} \;
```

Saving the NSO Configuration to a file

In addition to the usage of NSO backup utility, it is possible to save the NSO configuration to a text file, and use this to perform a system restore. This method also has the advantage that it can backup specific sections of the system configuration such as the CloudVPN services.

From NSO CLI, the `save` command can be used in configuration mode. The example below shows how the CloudVPN services can be saved to a file called `my services`. The file will be created in the same folder in which the command was executed. If the command is executed from the Home Directory, that is where the file will be located.

```
admin@ncs-vm% save my-services cloudvpn
Saving parts of the configuration.
[ok][2015-12-08 15:06:16]
```

The configuration segments can be loaded from a file and merged with existing configuration from the NSO CLI configuration mode.

```
admin@ncs-vm% load merge my-services
[ok][2015-12-08 15:10:10]

[edit]
admin@ncs-vm% show | compare
+cloudvpn my-new-cvpn {
+    provider vms-lab-pod;
+    tenant my-new-cvpn;
+    license-bandwidth 50;
+    cpe cpe_1444911212046 {
+        allocate {
+            ip-type ipv4;
+            prefix-size 24;
+        }
+    }
+    cpe cpe_1444911212048 {
+        allocate {
+            ip-type ipv4;
+            prefix-size 24;
+        }
+    }
+    cpe cpe_50 {
+        allocate {
+            ip-type ipv4;
+            prefix-size 24;
+        }
+    }
+    firewall {
+        ssl-vpn {
+            max-users 50;
+            certificate-name cn=SSL-VPN,ou=LAB,c=BE;
+        }
+        rule 10 {
+            match-type any;
+            action allow;
+        }
+    }
+    wsa {
+        security-protection-level security-high;
+    }
+}
[ok][2015-12-08 15:10:18]
```

If configuration changes viewed with `show | compare` are correct, in order to finalize the configuration, please enter the `commit` command to save the changes.

NSO Restore

Due to the complexity of overall vMS solution, NSO restore may not be enough to restore all the vMS services, as other vMS solution components might also need to be restored in order for an end-to-end solution to be completely restored and functional.

> **Note:** It is highly recommended to consult Cisco Technical Services or Cisco Advanced Services for performing the NSO restoration procedure.

In order to restore the NSO from a backup file, the NSO must be stopped before loading the backup file.

NSO restore is done in three steps, as shown in the output below:

- Stop NCS services
- Restore NSO from NSO backup file
- Re-start NCS services

```
  ubuntu@ncs-vm:~$ sudo service ncs stop
  Stopping NCS: .
  ubuntu@ncs-vm:~$ sudo /opt/ncs/current/bin/ncs-backup --restore /var/opt/ncs/backups/ncs-
3.4.2.1@2015-12-08T14\:36\:58.backup
  Restore /etc/ncs from the backup (y/n)? y
  Restore /var/opt/ncs from the backup (y/n)? y
  INFO  Restore completed successfully
  ubuntu@ncs-vm:~$ sudo service ncs start
  Starting NCS: .
```

5.7 Upgrade

The NSO package installation and upgrade are performed using Ansible scripts that ship with the NSO software package. NSO system installation is located in `/opt/ncs` directory, where `/data/ncs/current` represents a symbolic link, pointing to the currently installed software package directory, as shown in the following output.

```
ubuntu@ncs-vm:~$ ls -la /opt/ncs/current
lrwxrwxrwx 1 root root 11 Nov 26 11:18 /opt/ncs/current -> ncs-3.4.2.1
```

NSO Upgrade Procedure

1. Download of required NSO Package and unpacking

In order to upgrade the current NSO system, download the required NSO package to the `/opt` directory on the NSO VM, and use tar to unpack the archive:

```
tar -xvf NCS-2.0-2015-10-05-59.tar
```

2. Unpacking cvpn.tgz archive in new NSO Package

Change directory in the new "NSC-x.x..." directory and unpack the `cvpn.tgz` archive again, moving into the newly created directory:

```
cd NCS-2.0-2015-10-05-59
tar -zxvf cvpn.tgz
cd cvpn-rel-2015-10-05-01-15
```

3. Setting up Ansible environment variables

Set the "NCS_ANSIBLE" environment variable to this current directory:

```
export NCS_ANSIBLE=`pwd`/ansible
```

4. NSO Core Patch Upgrade (Optional)

Upgrading of the NSO covers two components; the core NSO binary and any patches, and the NSO function packs themselves. Before proceeding with the following upgrade steps, please refer to the release notes to determine if there are any patches to the core NSO software.

Upgrade/patch NSO core (optional) using the following command:

```
  ansible-playbook --sudo ansible/playbooks/ncs_sys_files.yml
```

5. NSO Functional Pack Upgrade

Upgrade the NSO functional packs, using `ncs_package_upgrade.sh` script.

This command takes the form: `ncs_package_upgrade.sh <packages-dir> <username> <cvpn-hosts-file>`

```
  admin@ncs-vm> $NCS_ANSIBLE/scripts/ncs_package_upgrade.sh `pwd`/cvpn-packages ubuntu
/etc/ansible/cvpn-hosts
```

6. NSO Installation Verification

To see the version information of the current CloudVPN packages loaded in NSO, run the command `show cloudvpn-release` from the NSO CLI, as shown below.

```
  admin@ncs-vm> show cloudvpn-release
  cloudvpn-release cvpn-version VMS-2.1
  cloudvpn-release build-date 2015-10-05-01-15
  cloudvpn-release ncs-build-version 3.4.2.1
  cloudvpn-release git-revision 717cddb
  cloudvpn-release git-branch dev
  cloudvpn-release build-user px-build
  cloudvpn-release build-host bxb-bld10-lnx
  cloudvpn-release build-host-uname Linux
  cloudvpn-release build-host-java-version 1.7.0_25-Java(TM)
  [ok][2015-12-08 15:45:06]
```

To check the operational status of the CloudVPN packages use the command `show packages package oper-status` from the CLI.

```
  admin@ncs-vm> show packages package oper-status
                                                                                 PACKAGE
                                                                                 META            FILE
                           JAVA           BAD NCS  PACKAGE  PACKAGE  CIRCULAR    DATA     JAVA   LOAD   ERROR
  NAME                 UP  UNINITIALIZED  VERSION  NAME     VERSION  DEPENDENCY  ERROR    ERROR  ERROR  INFO
  -----------------------------------------------------------------------------------------------------------
--
  cisco-asa            X   -              -        -        -        -           -        -      -      -
  cisco-ios            X   -              -        -        -        -           -        -      -      -
  cisco-iosxr          X   -              -        -        -        -           -        -      -      -
  cisco-pnp            X   -              -        -        -        -           -        -      -      -
  cisco-vips_manager   X   -              -        -        -        -           -        -      -      -
  cisco-vips_sensor    X   -              -        -        -        -           -        -      -      -
  cisco-wsa            X   -              -        -        -        -           -        -      -      -
  cloudvpn             X   -              -        -        -        -           -        -      -      -
  day0                 X   -              -        -        -        -           -        -      -      -
  dns-updater          X   -              -        -        -        -           -        -      -      -
  esc                  X   -              -        -        -        -           -        -      -      -
  id-allocator         X   -              -        -        -        -           -        -      -      -
  iosstats             X   -              -        -        -        -           -        -      -      -
  ipaddress-allocator  X   -              -        -        -        -           -        -      -      -
  perfdata             X   -              -        -        -        -           -        -      -      -
  quagga-bgp           X   -              -        -        -        -           -        -      -      -
  resource-manager     X   -              -        -        -        -           -        -      -      -
  tailf-hcc            X   -              -        -        -        -           -        -      -      -
  vm-manager           X   -              -        -        -        -           -        -      -      -
  vm-manager-esc       X   -              -        -        -        -           -        -      -      -

  [ok][2015-12-08 15:44:57]
```

If all the packages are installed successfully, operational status will be shown as **UP** in the previous output.

5.8 Troubleshooting

NSO Log Overview

NSO logs are stored in the following directory: `/var/log/ncs`.
Most of the log files are enabled in the `ncs.conf` file under the `/ncs-config/logs` section unless specified otherwise.

Please refer to the "Logging" section in chapter 5.3 "NSO Configuration" for details on how to enable/disable logs.

> **Note:** If NSO is deployed in a clustered configuration, some of the following log and trace files are specific to NSO Service or Device nodes, while others exist on both nodes.

Audit Log

NSO Audit Log `audit.log` exists on both Service and Device nodes. It contains audit trails which track changes that have been made to an NSO device.

Day 0 Server Access Logs

Day 0 Server Access Logs are stored in `day0server:<devicename>.access` file that contains Day 0 HTTP server access logs.

It exists on both Service and Device nodes.

Developer Log

Developer log `devel.log` is a debug log for troubleshooting user-written Java code. It exists on both Service and Device nodes. Check this log for problems with user- written code, etc. This log is enabled by default. Developer log is used to troubleshoot HA-clustering issues.

NCS Error Log

NCS error log is used for internal logging from the NCS daemon. It is used for troubleshooting the NCS daemon itself. Enable this log when NSO crashes, reports an "Internal error," or the error is related to the NCS daemon. This log creates at least three different files; all these files are needed to be able to read it. This log is particularly useful for Cisco developers. It exists on both Service and Device nodes and is stored in `ncserr.log.1` in a binary format.

In order to view the contents of the log, the following command is used `--printlog /var/log/ncs/ncserr.log.1`

NCS Java VM Log

NCS Java VM starts a number of internal processes or threads. One of these threads executes a service called NcsLogger which handles the dynamic configurations of the logging framework for NCS Java API. Any function pack written in Java will write its log to this file. It exists on both Service and Device nodes. As CloudVPN function packs are Java-based, all the relevant call-flow specific logs will also be written into `ncs-java-vm.log`.

> **Note**: Thus, this log is of a great importance when troubleshooting CloudVPN and CloudVPN provisioning issues.

NCS Log

NCS daemon logs are located in `ncs.log` file.

Check this log for startup problems of the ncs daemon itself. It exists on both Service and Device nodes.

NED Trace Log

NED trace logs are named `ned-<vnf_or_cpe>.trace` and contain the southbound NETCONF/NED traffic flows between NSO and the VNFs or the CPEs.

It exists on the Device Node only.

NED DNS Updater Trace Log

NED DNS updater trace logs `(ned-dns-updater-dns.trace`) contain specific information about NSO DNS updater component.

It exists on the Service Node only.

NED Quagga BGP Device Trace Log

NED Quagga BGP Device Trace Log (`ned-quagga-bgp-devs-quagga.trace`) contains logs relevant to BGP Quagga component of the solution.

Logs exist on both Service and Device nodes.

NETCONF Cluster Trace Logs

NETCONF Cluster Trace Logs (`netconf-cluster-<name>.trace`) log contains NETCONF traces specific to cluster communication.

This log exists on Service node only.

NETCONF ESC Trace Logs (netconf-<esc_name>.trace**)**

NETCONF ESC Trace Logs (`netconf-<esc_name>.trace`) contains NETCONF traces specific to NSO to ESC communication.

Log exists on Service node only.

NETCONF Log

NETCONF Log (`netconf.log`) is used for troubleshooting NETCONF operations on both Service and Device nodes.

NETCONF North Log

File Netconf-north log is useful for understanding and troubleshooting northbound NETCONF protocol interactions between the Portal and NSO. When this log is enabled, all NETCONF traffic to and from NSO is stored to a file. It exists on the Service Node only.

CPE PID Trace Log

PID trace are files named `PID_<cpe_pid>.trace` and contains the southbound PnP https traffic flow between PnP and the CPEs.

It exists on the Service Node only.

Unlike other logs discussed in this chapter, PID trace logs are not enabled or disabled in `ncs.conf`, but instead, can be enabled and their output directory can be configured in NCS "pnp" configuration section:

```
admin@ncs-vm> show configuration pnp
logging {
    enabled;
    directory /var/log/ncs;
}
```

PnP Server Access Logs

PnP Server Access Logs (`pnpserver:<port>.access`) log contains PnP server access log events.

It exists on the Service Node only.

SNMP VNF/CPE Trace Log

SNMP trace logs (named `snmp-<vnf_or_cpe_id>.trace`) contain SNMP-related traffic flows for specific VNF or CPEs.

SNMP Log

The SNMP log `snmp.log` contains SNMP-related operations of the local SNMP daemon towards the configured SNMP trap collector (NMS).

WebUI Browser Log

The WebUI browser log `webui-browser.log` makes it possible to log JavaScript errors/exceptions in a log file on the target device instead of just in the browser's error console. It exists on both Service and Device nodes.

Collecting NCS error logs & debug dumps

NCS error logs and debug dumps can be requested by Cisco TAC for troubleshooting purposes.

Use the following CLIs to collect ncs error logs and debug dumps.

```
ncs --printlog ncserr.log
ncs --debug-dump /home/localadmin/debugdump
```

NCS Log Reader

This is a tool created to make searching through logs easier and more efficient. The script is called ncs-log-reader and it is located in `/usr/local/bin` directory of the of the NSO server.

By passing specific arguments, the tool sorts through the logs and prints out specific segments. Please refer to the help description below, as shown below.

```
ncs-log-reader [args..]
ncs-log-reader [-m | --minutes ] <mins>
               [-h | --hours ]   <hours>
               [-s | --since ]   <time>
ncs-log-reader [--exclude RegEx] ......
ncs-log-reader [--include RegEx] ......
ncs-log-reader --show-files
```

Examples:
-m 30 -- the last 30 minutes
-h 2 -- the last 2 hours
-s 2015-01-22 22:22:00 -- Everything since that date
-s 22:22:00 -- Everything since that 10 pm today
-s 14 -- Everything since 14.00 today

Ncs-log-reader merges all ncs logs in the current directory and prints to stdout - including the ncs error log.

The following are some examples of how `ncs-log-reader` can be used.

1. Show the logs updated in the last 5 minutes

```
  localadmin@ncs-vm:/var/log/ncs$ ncs-log-reader --show-files -m 5
  netconf-cluster-croatia.trace
  PID_C881_K9_VID__SN_FCZ1909719M.trace
  PID_C881_K9_VID__SN_FCZ190961QH.trace
  localhost.access
  PID_CISCO2911_K9_VID__SN_FCZ174770XL.trace
  netconf-north.log
  PID_C881_K9_VID__SN_FCZ1823C444.trace
  netconf-cluster-germany.trace
  PID_C881_K9_VID__SN_FCZ1909719B.trace
  netconf.log
  PID_CISCO1921_K9_VID__SN_FCZ1820C030.trace
  PID_C881_K9_VID__SN_FCZ190961QJ.trace
  PID_C881_K9_VID__SN_FCZ1909719J.trace
  PID_C881_K9_VID__SN_FCZ190961QE.trace
  PID_C881_K9_VID__SN_FCZ1909719Q.trace
  PID_C881_K9_VID__SN_FCZ19097199.trace
  PID_C881_K9_VID__SN_FCZ190961QT.trace
  PID_C881_K9_VID__SN_FCZ1909719C.trace
  PID_C881_K9_VID__SN_FCZ1823C471.trace
  PID_C881_K9_VID_V01_SN_FCZ1909719E.trace
  PID_CISCO1921_K9_VID__SN_FCZ190391KQ.trace
  PID_CISCO1941_K9_VID__SN_FCZ1504C0FL.trace
  audit.log
  PID_C881_K9_VID__SN_FCZ1925C1QW.trace
```

2. Collect the logs related to a device for the last 30 minutes and write them to a file in the Home Directory.

```
  localadmin@ncs-vm:/var/log/ncs$ ncs-log-reader -m 30 --include FGL154325HG | tee
~/FGL154325HG.log
```

Service Chain Undeploy-Redeploy

In some circumstances it may be necessary to redeploy a service chain.

The following commands represent a list of common use-cases where service chain undeploy-redeploy is required.

Next command is used to undeploy and the redeploy a service chain.

```
admin@ncs-vm> request cloudvpn-action undeploy-redeploy cloudvpn <cloudvpn_name>
```

Similar to the previous command with the addition of the "force" keyword, used when the connection to the ESC which currently hosts the service chain is lost.

```
admin@ncs-vm> request cloudvpn-action undeploy-redeploy cloudvpn <cloudvpn_name> force
```

The following command is used to undeploy/redeploy all the service chains managed by a specific ESC. For example, this command can be used to evacuate all VNFs from a virtual infrastructure/datacenter:

```
admin@ncs-vm> request cloudvpn-action undeploy-redeploy vnfm <esc_name>
```

Re-establish Connection with ESC

NSO will register with ESC for netconf-notifications to receive return notifications on VM status events during service instantiation, for example. Sometimes this session can timeout/terminate at the ESC side, however, the NSO still has the session 'open' locally. This session is owned by the NSO and will not be re-established by ESC. The NSO tries to re-establish a netconf-notifications session to the ESC every 5 seconds.

To check if the ESC is enabled in NSO, use following command:

```
admin@ncs-vm> show devices device <esc_name> state oper-state
state oper-state enabled
```

To check if the netconf-notifications session between NSO and ESC is up, use the following NSO command.

```
admin@ncs-vm> show devices device <esc_name> netconf-notifications subscription status
NAME  STATUS
---------------
s     running
```

If the ESC is down, or it is no longer reachable by the NSO, the oper-state status changes to "disabled" and the netconf-notification subscription status toggles between "failed" and "connecting", as shown in the following outputs.

```
  admin@ncs-vm> show devices device <esc_name> state oper-state
  state oper-state disabled
```

```
  admin@ncs-vm> show devices device <esc_name> netconf-notifications subscription status
  NAME   STATUS
  --------------
  s      failed
```

```
  admin@ncs-vm> show devices device <esc_name> netconf-notifications subscription status
  NAME   STATUS
  ------------------
  s      connecting
```

When the ESC is reachable again, the netconf-notifications session should be re-established automatically.

If this is not the case, ESC must be manually be reconnected by using the following NSO CLI command.

```
  admin@ncs-vm> request devices device <esc_name> netconf-notifications subscription s
reconnect
```

In some cases it can happen that ESC sends notifications, however, NSO does not receive them and devices marked as 'VM_ALIVE' by ESC will never be added to the NSO CDB. In order to verify this condition, ESC logs can be correlated with the output of `show devices device esc netconf-notifications received-notifications`.

In case that NSO is missing notifications sent by ESC, the replay keyword can be used to force the ESC to re-send the notifications that have not been acknowledged by NSO.

```
  admin@ncs-vm> request devices device <esc_name> netconf-notifications subscription s replay
```

HA-cluster Restoration

In the current NSO HA Clustering implementation, NSO HA framework currently doesn't have the functionality to automatically sense and initiate a revert back to the original HA Cluster configuration.

This means that manual intervention to restore HA Clustering functionality is needed after the NSO node (the VM) or the NSO process are restarted.

Technical Background

NSO HA Clustering functionality relies on two distinct underlying frameworks: the NSO HA framework - which is responsible for CDB replication from a Master node to one or more Slave nodes; and the High

Availability Cluster Communications (tailf-hcc) - which coordinates the master-slave relationships ensuring that a Slave node takes over the Master role when the HA Master is no longer available (keepalive communication to the Master has failed).

The High Availability Cluster Communication system is neither responsible nor aware of the CDB replication status across HA nodes, therefore it should be made sure that the original Master node coming back to life after VM or NSO process restart doesn't assume its original responsibility, as it cannot be sure how long the original Master was down.

The original Slave (now the new Master) node may have processed updates that the original Master is not aware of. Until the original Master synchronizes its CDB database from the new Master, it should not take its original role.

To achieve this, the 'auto-start' parameter of the ha-cluster global configuration is set to disabled, which ensures that on startup all cluster member nodes will NOT automatically assume the configured HA roles.

```
admin@ncs-vm-1> show configuration ha-cluster
cluster-name HA-CLUSTER;
auto-start disable;
failure-limit 2;
bgp-anycast-failover {
 bgp-anycast-prefix 2a00:a24:b5fe:fff::ab01/128;
 bgp-anycast-path-min 2;
 bgp-clear-enabled true;
}
members ncs-vm-1 {
 address 2a00:a24:b5e2:17::1:2;
 role master;
 bgp-anycast true;
 device devm-quagga;
}
members ncs-vm-2 {
 address 2a00:a24:b5e4:17::1:1;
 role slave;
 transition-enabled true;
 bgp-anycast true;
 master-capable true;
 device devs-quagga;
}
```

Procedure to Restore HA Clustering

After rectifying the cause of the Master-Slave communication failure, the approach is to bring the original Master back on-line initially as a Slave, which will then connect to the new Master to complete a CDB re-sync prior to being transitioned to the operational Master.

The following procedure is based on the ha-cluster configuration shown above, where ncs-vm-1 is the configured Master node and ncs-vm-2 is Slave. It assumes that ncs-vm-1 at a given point in time was restarted (the entire server or just the NSO process) and it is now back online.

1. Override the role of ncs-vm-1 to Slave for connection and CDB Sync with new Master (ncs-vm-2).

```
admin@ncs-vm-1> request ha-cluster commands role-override role slave
status override
[ok]
```

2. Re-activate HA-Cluster on Original Master

Allow adequate time for CDB on re-activated ncs-vm-1 to sync with CDB on the failed-over Master. This time depends on the CDB size and how much of the CDB needs to be updated.

Database synchronization status can be checked with the command below. Note that replication is finished once "waiting-for-replication-sync" flag is set to "false".

```
admin@ncs-vm-1> show status ncs-state internal cdb datastore
datastore running {
    transaction-id               1449-586682-571214@NSO-vm-2;
    filename                     /var/opt/ncs/cdb/A.cdb;
    disk-size                    "31.75 MiB";
    ram-size                     "26.11 MiB";
    read-locks                   0;
    write-lock-set               false;
    waiting-for-replication-sync false;
}
datastore operational {
    transaction-id        1449-601896-460758@NSO-vm-1;
    filename              /var/opt/ncs/cdb/O.cdb;
    disk-size             "703.05 KiB";
    ram-size              "2.90 MiB";
    subscription-lock-set false;
}
[ok][2015-12-08 19:11:36]
```

3. When CDB sync is complete, first revert failed-over Master ncs-vm-2 back to Slave as configured.

```
admin@ncs-vm-2> request ha-cluster commands role-revert
 status reverted
[ok]
```

4. Revert ncs-vm-1 to Master as configured.

```
admin@ncs-vm-1> request ha-cluster commands role-revert
 status reverted
[ok]
```

5. Verify that ncs-vm-2 Slave re-connects to the Master and synchronizes its CDB data to that on the newly reinstated Master.

```
admin@ncs-vm-1> show ha-cluster
NAME           STATUS
-----------------------
ncs-vm-2  slave
ncs-vm-1  master

admin@ncs-vm-1> show ncs-state ha
ncs-state ha mode master
ncs-state ha node-id ncs-vm-1
ncs-state ha connected-slave [ ncs-vm-2 ]

admin@ncs-vm-2> show ha-cluster
NAME           STATUS
-----------------------
ncs-vm-2  slave
ncs-vm-1  master
```

6. Elastic Services Controller (ESC)

6.1 Introduction

This chapter describes the ESC (Elastic Services Controller) role in vMS, how it operates, and where to begin troubleshooting if problems arise. This chapter includes the following topics:

- Overview - a brief overview of the role of ESC in vMS
- High-Availability - how ESC operates in High Availability mode
- Operations - ESC health checks and other useful commands
- Logs - where the logs are located, which ones are important, and what to look for in them
- VNF Recovery - how the ESC recovers a failed VNF
- Troubleshooting - detailed vMS Call Flow and where to begin troubleshooting

6.2 Overview

The Elastic Service Controller performs the role of the Virtual Network Functions Manager (VNFM) in the VMS Architecture. ESC receives requests from the NSO to create service chains that consist of networks, subnets, and VMs to be deployed in OpenStack. It also monitors the VNF instances for availability via their management interface and can reboot or redeploy VNFs if needed. ESC sends notifications to the NSO when the VNFs and the entire service chain are ready. In the vMS Architecture, ESC utilizes NETCONF protocol for northbound communication to NSO, and uses REST API calls southbound to OpenStack.

- Cisco Elastic Services Controller (ESC) is a Virtual Network Functions Manager (VNFM) and it performs life cycle management of Virtual Network Functions (VNFs).
- ESC provides agentless VNF management by provisioning the virtual services and monitoring their health.
- ESC promotes agility, flexibility, and programmability in Network Function Virtualization (NFV) environments.
- ESC provides flexibility to define rules for monitoring and associated actions that are triggered based on the outcome of these rules.
- In the event of a VNF failure, ESC also supports automatic VNF recovery.
- Complex services include multiple VNFs that are orchestrated as a single service with dependencies between them. These multiple VNFs are managed as a single entity.

Key ESC Features

- Provides end-to-end dynamic provisioning and monitoring of virtualized services.
- Provides customization across different phases of life cycle management; while monitoring the VNF, service advertisement, and custom actions.
- Provides agentless monitoring with an integrated Monitoring Actions (MONA) engine. The monitoring engine provides simple and complex rules to monitor VNFs
- Deploys or removes VNFs based on the monitoring errors and threshold conditions detected as part of healing (also called as recovery).
- Supports multi-tenant environments.
- Supports REST and NETCONF/YANG interfaces to provide better hierarchical configuration and data modularity.

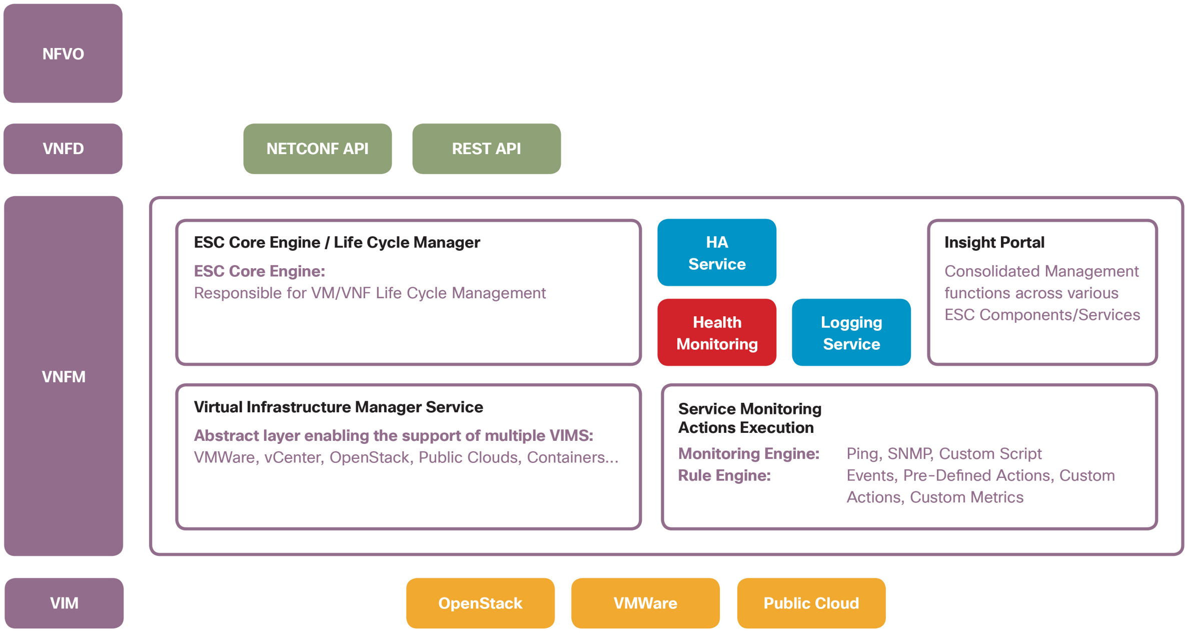
ESC Architecture

Cisco Elastic Services Controller (ESC) is built as an open and modular architecture, that allows OSS and multi-vendor support. It performs life cycle management of the VNFs, that is, VNF on-boarding, configuring the VNFs, monitoring them, and making VNF level life cycle decisions such as healing and scaling based on the KPI requirements. ESC and its managed VNFs are deployed as VMs running within a Virtual Infrastructure Manager (VIM). Currently, OpenStack and vCenter 5.5 are the supported VIMs. In the vMS solution, the VIM is OpenStack.

The ESC core engine manages transactions, validations, policies, workflows, VM state machines and rollbacks. The monitoring and actions service engine in ESC performs monitoring based on several monitoring methods. Events are triggered based on the monitoring actions. The monitoring engine also supports custom monitoring plugins. ESC can be configured for High Availability.

ESC interacts with the top orchestration layer using the REST and NETCONF/YANG NB APIs. The orchestration layer can be a Cisco NSO or any third party OSS. The ConfD agent in ESC enables integration with NSO by adding NETCONF/YANG northbound interface support. A configuration template, Virtual Network Function Descriptor (VNFD) file is used to describe the deployment parameters and operational behaviors of the VNFs. The VNFD file is used in the process of on-boarding a VNF and managing the life cycle of a VNF instance.

Figure 6.1 Elastic Services Controller Architecture


NFVO
VNFD
NETCONF API
REST API
VNFM
ESC Core Engine / Life Cycle Manager
ESC Core Engine:
Responsible for VM/VNF Life Cycle Management
HA Service
Health Monitoring
Logging Service
Insight Portal
Consolidated Management functions across various ESC Components/Services
Virtual Infrastructure Manager Service
Abstract layer enabling the support of multiple VIMS:
VMWare, vCenter, OpenStack, Public Clouds, Containers...
Service Monitoring Actions Execution
Monitoring Engine:
Rule Engine:
Ping, SNMP, Custom Script
Events, Pre-Defined Actions, Custom Actions, Custom Metrics
VIM
OpenStack
VMWare
Public Cloud

ESC Interfaces

- Elastic Services Controller (ESC) supports REST and NETCONF northbound interfaces for operations and transactions.
- Northbound ESC interacts with NSO, and southbound ESC interacts with the VIM layer, which in this case is OpenStack.
- In the vMS Solution, ESC is essentially acting as a gateway accepting NETCONF service requests from NSO and translating them to OpenStack REST API calls.
- NSO communicates with ESC using the NETCONF protocol and YANG-based data models. ESC manages Virtual Network Functions at a device level, and NSO manages the entire network service lifecycle. Together, they make it a complete orchestration solution that spans across both physical and virtual infrastructure.

6.3 High-Availability

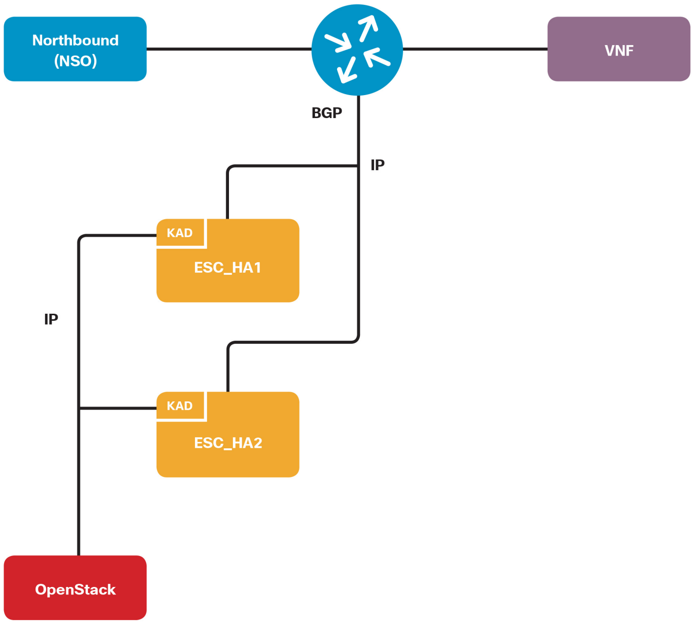
ESC has the capability to provide high availability in a one-to-one configuration. ESC HA deployment has two network variations and two database variations.

Simple Network HA

The first of the network variations is the Simple Network HA. In this scenario, both the ESC VMs are on the same network and use keep-alive messages to check on health and determine which ESC instance is MASTER and which one is BACKUP.

Keep-alive messages run on the southbound interface of the ESC.

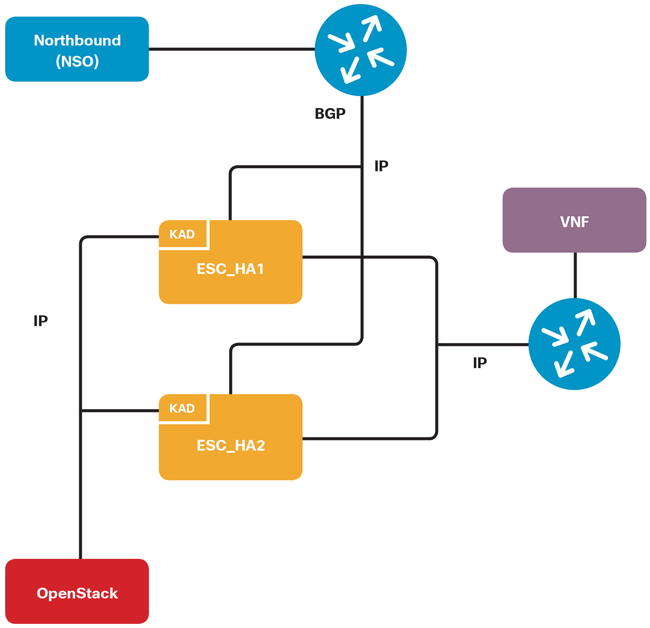
Figure 6.2 ESC High-Availability - Simple Network HA



HA with BGP

In a second network variation, each ESC has northbound interfaces towards NSO on separate L2 domains with separate subnets. BGP is used to advertise an Anycast address for the NSO to use to connect to the ESC.

Figure 6.3 ESC High-Availability - with BGP



Database on OpenStack Cinder Volume

In the first database scenario, ESC stores its data in an OpenStack Cinder volume that is created during ESC deployment. The Cinder volume is shared between the two ESC instances. Only the MASTER ESC has a Cinder volume mounted. The Cinder volume is not attached to the BACKUP ESC. In the case of a switchover, the Cinder volume gets attached to the new MASTER ESC.

DRDB for Database

In the second database scenario, each ESC instance has its own database and the database is replicated between the two nodes. This deployment is not dependent on usage of Cinder volumes. Databases are synchronized at regular intervals between the paired ESCs.

Troubleshooting Failovers

The current role of the ESC in high availability is displayed upon login, and an example is shown in the output below.

```
####################################################################
    ESC on vms-esc-p0-2 is in BACKUP state.
####################################################################
```

The current status of ESC can be checked by displaying `/cisco/keepalived_state` file. File timestamp indicates the last time ESC changed the status - the last time ESC failover happened.

```
 [admin@vms-esc-1 cisco]$ cat /cisco/keepalived_state
 MASTER
 [admin@vms-esc-1 cisco]$ ls keepalived_state -l
 -rw-r--r--. 1 root root 7 Dec  9 20:22 keepalived_state
```

In an ESC pair, there should be one MASTER and one BACKUP. During a failover event, one or both ESCs may show the status as SWITCHING for a short period of time. Switching state should last less than 2 minutes. If state remains the same for 5 minutes, further investigation should begin by looking at the logs.

Logs are stored in `/var/log/esc`. The `esc_monitor.log` contains helpful information.

To check ESC health use `/cisco/health.sh` script. Sample output of the script is shown below.

If ESC is running without problems, there will be a "ESC HEALTH PASSED" line at the bottom on the output.

```
  [admin@vms-esc-1 ~]$ /cisco/health.sh
  esc_monitor start/running, process 30520
  esc_mona is up and running ...
  postgresql-9.4 (pid  30807) is running...
  esc ui service is not configured to run.  To enable use command: /cisco/esc-
init/enable_esc_ui
  Status Running: 30915
  tomcat6 (pid 30880) is running...                          [  OK  ]
  postgresql-9.4 (pid  30807) is running...
  ESC service is running...
  tcp        0      0 ::ffff:127.0.0.1:8080       :::*                        LISTEN
  /mnt/esc_database is a mountpoint
  ============== ESC HA (MASTER) with DRBD =================
  bDRBD_ROLE_CHECK=0
  MNT_ESC_DATABSE_CHECK=0
  ESC_CHECK=0
  STORAGE_CHECK=0
  ESC_SERVICE_RET=0
  NOVA_CHECK=0
  DB_CHECK=0
  MONA_RET=0
  ESC_MONITOR_RET=0
  =======================================
  ESC HEALTH PASSED
```

Here is sample output from the same commands on a BACKUP ESC.

Even though this ESC is in BACKUP mode, it has still passed the health checks, but without the detailed checking that can be done while a node is active.

```
[admin@vms-esc-2 ~]$ cat /cisco/keepalived_state
BACKUP
[admin@vms-esc-2 ~]$ /cisco/health.sh
============== ESC HA (BACKUP) =================
=======================================
ESC HEALTH PASSED
```

To check the history of the ESC status, use the following command `/var/log/esc/esc_haagent.log`. Additional details can be found by reviewing the entire log.

```
[admin@vms-esc-2 esc]$ grep "switching to" /var/log/esc/esc_haagent.log
[Mon Nov 23 20:27:57 2015] switching to SWITCHING_TO_BACKUP
[Mon Nov 23 20:27:59 2015] switching to BACKUP
[Fri Dec  4 21:35:25 2015] switching to SWITCHING_TO_FAULT
[Fri Dec  4 21:35:25 2015] switching to BACKUP
[Fri Dec  4 21:35:25 2015] switching to FAULT
[Fri Dec  4 21:36:12 2015] switching to SWITCHING_TO_BACKUP
[Fri Dec  4 21:36:12 2015] switching to BACKUP
[Fri Dec  4 21:36:28 2015] switching to SWITCHING_TO_MASTER
[Fri Dec  4 21:36:58 2015] switching to MASTER
[Fri Dec  4 21:42:48 2015] switching to SWITCHING_TO_FAULT
[Fri Dec  4 21:43:21 2015] switching to BACKUP
[Fri Dec  4 21:43:21 2015] switching to FAULT
[Fri Dec  4 21:43:22 2015] switching to SWITCHING_TO_STOP
[Fri Dec  4 21:43:22 2015] switching to BACKUP
[Fri Dec  4 21:43:22 2015] switching to STOP
[Fri Dec  4 21:44:10 2015] switching to SWITCHING_TO_BACKUP
[Fri Dec  4 21:44:11 2015] switching to BACKUP
[Fri Dec  4 21:45:14 2015] switching to SWITCHING_TO_MASTER
[Fri Dec  4 21:45:44 2015] switching to MASTER
[Fri Dec  4 21:46:14 2015] switching to SWITCHING_TO_FAULT
[Fri Dec  4 21:47:11 2015] switching to BACKUP
[Fri Dec  4 21:47:11 2015] switching to FAULT
[Fri Dec  4 21:47:11 2015] switching to SWITCHING_TO_BACKUP
[Fri Dec  4 21:47:11 2015] switching to BACKUP
```

6.4 Operations

ESC offers a CLI interface that allows performing basic configuration and operational tasks.

It is possible to SSH to the ESC and then invoke the ConfD CLI to administer and configure the ESC. This is not needed during normal vMS operations and would only be used during troubleshooting or reconfiguration of passwords.

ESC ConfD CLI Access

To access ESC ConfD CLI please use the following command.

```
[admin@esc ~]$ /opt/confd/bin/confd_cli
```

Once connected `?` can be executed to preview all the possible command options.

```
admin connected from 10.32.1.23 using ssh on esc-test-1
admin@esc-test-1> ?
Possible completions:
  compare    - Compare running configuration to another configuration or a file
  configure   - Manipulate software configuration information
  describe     - Display transparent command information
  exit            - Exit the management session
  file             - Perform file operations
  help           - Provide help information
  id               - Show user id information
  monitor     - Real-time debugging
  ping          - Ping a host
  quit           - Exit the management session
  request      - Make system-level requests
  script         - Script actions
  set             - Set CLI properties
  set-path    - Set relative show path
  show        - Show information about the system
  source      - File to source
  ssh           - Open a secure shell on another host
  telnet       - Open a telnet session to another host
  top           - Exit to top level and optionally run command
  traceroute - Trace the route to a remote host
  up           - Exit one level of configuration
```

Useful ESC ConfD Commands

To inspect the configuration of a particular Service Chain execute.

```
  admin@esc> show configuration esc_datamodel tenants tenant cns-auto1b deployments deployment
basic1
  policies {
      placement CSR {
          type         affinity;
          enforcement  strict;
          vm_group_ref [ CSR ];
      }
  }
  networks {
      network BR-INSIDE-01-basic1 {
          shared      false;
          admin_state true;
          subnet BR-INSIDE-01-basic1-BOGUS-CVPN-SUBNET {
              ipversion ipv4;
              dhcp      false;
              address   10.192.4.0;
              netmask   255.255.255.0;
              gateway   10.192.4.1;
          }
      }
  }
  vm_group CSR {
      bootup_time        600;
      recovery_wait_time 600;
      interfaces {
          interface 0 {
              network VNF-Mgmt;
          }
          interface 1 {
              network private-direct-iv604;
              allowed_address_pairs {
                  address 0.0.0.0 {
                      netmask 0.0.0.0;
                  }
              }
          }
          interface 2 {
              network BR-INSIDE-01-basic1;
              allowed_address_pairs {
                  address 0.0.0.0 {
                      netmask 0.0.0.0;
                  }
              }
          }
      }
      kpi_data {
          kpi VM_ALIVE {
              metric_value             79;
              metric_cond              GT;
              metric_type              UINT32;
              metric_occurrences_true  3;
              metric_occurrences_false 3;
              metric_collector {
                  type             ICMPPing;
                  nicid            0;
                  poll_frequency   15;
                  polling_unit     seconds;
                  continuous_alarm false;
              }
          }
      }
      rules {
          admin_rules {
              rule VM_ALIVE {
                  action [ "ALWAYS log" "FALSE recover autohealing" "TRUE servicebooted.sh" ];
              }
          }
      }
      config_data {
          configuration iosxe_config.txt {
              file https://10.32.1.23:443/day0/cfg/vRouter/node/basic1-csr/provider/cns-auto1b;
          }
      }
      scaling {
          min_active 1;
          max_active 1;
          elastic    true;
      }
      image              679b8db9-a1a2-48cd-a270-793036ef341d;
      flavor             b1ed0ffd-55e4-47b8-a1a7-72ec5f2efe44;
  }
  [ok][2015-12-07 21:54:42]
```

The above shows the sample configuration of a Basic Service Chain. In the output above, it shows that the Basic Service Chain is comprised of a single CSR VNF with three interfaces. One interface is connected to the

VNF-Mgmt network, the second interface to a bridge-inside network, and the third interface to a public private-direct network.

You can also see the image and flavor ID that will be used.

Checking ESC HA State

To inspect if a particular instance of ESC is in MASTER or BACKUP state, execute the following:

```
  [admin@esc-1 ~]$ cat /cisco/keepalived_state
  MASTER
```

From the output above it shows that esc-1 is acting as the "MASTER" and esc-2 is acting as the "BACKUP".

```
  [admin@esc-2 ~]$ cat /cisco/keepalived_state
  BACKUP
```

Checking ESC Service Health

To check ESC health use `/cisco/health.sh` script. Sample output of the script is shown below.

If ESC is running without problems, "ESC HEALTH PASSED" will be displayed on the last line.

```
  [admin@vms-esc-1 ~]$ /cisco/health.sh
  esc_monitor start/running, process 30520
  esc_mona is up and running ...
  postgresql-9.4 (pid  30807) is running...
  esc ui service is not configured to run.  To enable use command: /cisco/esc-
init/enable_esc_ui
  Status Running: 30915
  tomcat6 (pid 30880) is running...                          [  OK  ]
  postgresql-9.4 (pid  30807) is running...
  ESC service is running...
  tcp        0      0 ::ffff:127.0.0.1:8080       :::*                        LISTEN
  /mnt/esc_database is a mountpoint
  ============== ESC HA (MASTER) with DRBD =================
  bDRBD_ROLE_CHECK=0
  MNT_ESC_DATABSE_CHECK=0
  ESC_CHECK=0
  STORAGE_CHECK=0
  ESC_SERVICE_RET=0
  NOVA_CHECK=0
  DB_CHECK=0
```

```
MONA_RET=0
ESC_MONITOR_RET=0
=======================================
ESC HEALTH PASSED
```

Restarting ESC Service

```
[admin@esc ~]$ sudo /etc/init.d/esc_service restart
```

Please note that in an ESC-HA deployment, this will most likely trigger a failover event, in which case the backup ESC will become the Master.

6.5 Logs

This section covers logs available on ESC for monitoring its functionality, communication, debugging and real-time troubleshooting.

ESC has a built-in log collection script `/cisco/esc-init/collect_esc_log.sh`. The ESC log collector utility will collect all log files and create a tarball of ESC logs that can be sent to Cisco TAC for assistance. Here is an example of the command being run. Note that at the end, it prints out where the log file is stored in the `/tmp` directory and that it should be removed to clean up disk space. Important log files used for troubleshooting are listed below.

```
[admin@esc-2 esc-init]$ /cisco/esc-init/collect_esc_log.sh

Copying files to temporary folder: /tmp/esc_log-2015-12-08_19.57.46_UTC

Creating log tarball /tmp/esc_log-2015-12-08_19.57.46_UTC.tar.bz2
manager.2015-12-04.log
rules_mona.log
debug_yangesc.log
<<< ABBREVIATED OUTPUT >>>
netconf.log
mona.2015-11-21.0.log
catalina.out
mona.2015-11-26.0.log
tar: Removing leading `/' from member names
/media/cdrom/openstack/2012-08-10/meta_data.json
/media/cdrom/openstack/2013-04-04/meta_data.json
/media/cdrom/openstack/2013-10-17/meta_data.json
/media/cdrom/openstack/2013-10-17/vendor_data.json
/media/cdrom/openstack/latest/meta_data.json
/media/cdrom/openstack/latest/vendor_data.json
/media/cdrom/openstack/content/0000
/media/cdrom/openstack/content/0001
/media/cdrom/openstack/content/0002

Tarball created:
  /tmp/esc_log-2015-12-08_19.57.46_UTC.tar.bz2
Suggestions:
 1. Transfer the tarball file from the esc vm to your storage location.
 2. Remove the tarball from the esc vm disk.
 3. List contents of tarball:
     tar jtvf esc_log-2015-12-08_19.57.46_UTC.tar.bz2
```

```
    4. Extract the logs:
        tar jxf esc_log-2015-12-08_19.57.46_UTC.tar.bz2
```

Logs for ESC are stored in /`var/log/esc/` directory. This section lists the main logs that are used for troubleshooting and what can be found in each of those logs. The three main logs are: `esc_monitor.log`, `escmanager.log`, `yangesc.log` and they are explained in more details in the following sections.

ESC Monitor Log

This log is written to `esc_monitor.log` file and contains the output of ESC performing its own health checks on the environment. It checks its own processes every 10-20 seconds and it checks basic OpenStack keystone API access every minute.

Here is an example of an ESC status check. Notice that the last line at the bottom is OPER_UP. If it is OPER_DOWN, one of the other components will be down and needs further investigation on the failed component.

```
127.0.0.1 - - [04/Dec/2015 00:00:39] "POST /status HTTP/1.1" 200 0
{
    "machine_name": "esc-2",
    "msgid": "96451b46-4f2b-4ffa-a852-2d82a4f37826",
    "operational_info": [
        {
            "id": "c55a4014-dc7a-4466-a762-2e72535bb960",
            "message": "idle: 97.28%",
            "type": "cpu"
        },
        {
            "id": "81ab8021-d709-4498-aa82-e20de744c8bd",
            "message": "OPER_UP",
            "operational_info": [],
            "type": "esc"
        },
        {
            "id": "4915bb7c-6510-453a-88bf-d618bd288c96",
            "message": "Status Running: 2911",
            "type": "esc_confd"
        },
        {
            "id": "a4c8e10d-0788-45ff-8e94-40daa73a7502",
            "message": "OPER_UP",
            "type": "network"
        },
          {
              "id": "36d98559-6298-4f2a-9a6c-119b1296c40f",
              "message": "Total: 32241m, Used: 19.2643%",
              "type": "ram"
          },
          {
              "id": "f38a3649-418e-4d9c-8f50-73c50327e703",
              "message": "/dev/vda2 15% /,/dev/vda1 14% /boot,/dev/vdb 1%
/media/cdrom,/dev/drbd1 3% /mnt/esc_database",
              "type": "storage"
          },
          {
              "id": "95c60e15-e84b-401c-8cf5-b2baf1285e4a",
              "message": "tomcat6 (pid 2869) is running...[  OK  ]",
              "type": "tomcat6"
          }
      ],
      "operational_message": null,
      "operational_period": 15000,
      "operational_state": "OPER_UP"
  }
```

`esc_monitor.log` also contains OpenStack API check, as shown in the output below.

```
127.0.0.1 - - [04/Dec/2015 00:01:17] "POST /status HTTP/1.1" 200 0
POST for tokens was successful. Code: 200 = OK
URLS:
    NAME: nova URL: https://pod1.cloud.cisco.com:8774/v2/d861db1b6aae436fb52e9cdd8a9caf4f
    NAME: neutron URL: https://pod1.cloud.cisco.com:9696/
    NAME: cinderv2 URL: https://pod1.cloud.cisco.com:8776/v2/d861db1b6aae436fb52e9cdd8a9caf4f
    NAME: novav3 URL: https://pod1.cloud.cisco.com:8774/v3
    NAME: psc URL: None
    NAME: glance URL: https://pod1.cloud.cisco.com:9292
    NAME: ceilometer URL: https://pod1.cloud.cisco.com:8777
    NAME: heat-cfn URL: https://pod1.cloud.cisco.com:8000/v1/
    NAME: cinder URL: https://pod1.cloud.cisco.com:8776/v1/d861db1b6aae436fb52e9cdd8a9caf4f
    NAME: nova_ec2 URL: https://pod1.cloud.cisco.com:8773/services/Cloud
    NAME: heat URL: https://pod1.cloud.cisco.com:8004/v1/d861db1b6aae436fb52e9cdd8a9caf4f
    NAME: swift URL: https://storage.pod1.cloud.cisco.com:443/swift/v1
    NAME: keystone URL: https://pod1.cloud.cisco.com:5000/v2.0
GET https://pod1.cloud.cisco.com:5000/v2.0/tenants
GET for /tenants was successful. Code: 200 = OK
```

ESC Manager Log

This log is written to the `escmanager.log` file contains the internal actions taken by ESC and outbound requests to the OpenStack API. It also contains ESC state machine changes and other details. Use `yangesc.log` to narrow the timeframe and then find that timestamp in `escmanager.log`. Going backwards from there is usually the most effective way to find the actual error.

YANG ESC Log

This log is written to the `yangesc.log` primarily shows the NETCONF interaction between the NSO and ESC. Log provided below is a working example of ESC deploying a service chain with an ASAv and a CSR1000v in it. The important steps are:

- CREATE_NETWORK
- CREATE_SUBNET
- 2xVM_DEPLOYED
- 2x VM_ALIVE
- SERVICE_ALIVE

```
  20:59:36,528 19-Nov-2015 INFO  Order size: 1
  20:59:36,528 19-Nov-2015 INFO  Order: DEPLOY_SERVICE
  20:59:38,628 19-Nov-2015 INFO
  20:59:38,628 19-Nov-2015 INFO  ===== DEPLOY SERVICE REQUEST RECEIVED (UNDER TENANT) =====
  20:59:38,628 19-Nov-2015 INFO  Tenant name: Tenant1
  20:59:38,628 19-Nov-2015 INFO  Deployment name: Deployment1
  20:59:39,412 19-Nov-2015 INFO  =====  DEPLOY SERVICE REQUEST ACCEPTED  (UNDER TENANT)  =====
  20:59:40,159 19-Nov-2015 INFO
  20:59:40,159 19-Nov-2015 INFO  ===== SEND NOTIFICATION STARTS =====
  20:59:40,159 19-Nov-2015 INFO  Type: CREATE_NETWORK
  20:59:40,160 19-Nov-2015 INFO  Status: SUCCESS
  20:59:40,160 19-Nov-2015 INFO  Status Code: 200
  20:59:40,160 19-Nov-2015 INFO  Status Msg: Network creation for [BR-INSIDE-01-Deployment1]
completed successfully!
  20:59:40,160 19-Nov-2015 INFO  Tenant: Tenant1
  20:59:40,160 19-Nov-2015 INFO  Network: Tenant1Deployment1BR-INSIDE-01-Deployment1
  20:59:40,160 19-Nov-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
  20:59:40,525 19-Nov-2015 INFO
  20:59:40,525 19-Nov-2015 INFO  ===== SEND NOTIFICATION STARTS =====
  20:59:40,525 19-Nov-2015 INFO  Type: CREATE_SUBNET
  20:59:40,525 19-Nov-2015 INFO  Status: SUCCESS
  20:59:40,525 19-Nov-2015 INFO  Status Code: 200
  20:59:40,526 19-Nov-2015 INFO  Status Msg: Subnet creation completed successfully, subnet ID:
[Tenant1Deployment1BR-INSIDE-01-Deployment1BR-INSIDE-01-Deployment1-BOGUS-CVPN-SUBNET]
  20:59:40,526 19-Nov-2015 INFO  Tenant: Tenant1
  20:59:40,526 19-Nov-2015 INFO  Network: Tenant1Deployment1BR-INSIDE-01-Deployment1
  20:59:40,526 19-Nov-2015 INFO  Subnet: Tenant1Deployment1BR-INSIDE-01-Deployment1BR-INSIDE-
01-Deployment1-BOGUS-CVPN-SUBNET
  20:59:40,526 19-Nov-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
  21:00:41,393 19-Nov-2015 INFO
  21:00:41,393 19-Nov-2015 INFO  ===== SEND NOTIFICATION STARTS =====
  21:00:41,393 19-Nov-2015 INFO  Type: VM_DEPLOYED
  21:00:41,393 19-Nov-2015 INFO  Status: SUCCESS
  21:00:41,393 19-Nov-2015 INFO  Status Code: 200
  21:00:41,393 19-Nov-2015 INFO  Status Msg: OpenStack Driver: VM successfully created, VM
Name: [Deployment1__78760__Tenant1__SystemAdminTenantIdDeployment1Deployment10.0__0__CSR__0]
  21:00:41,393 19-Nov-2015 INFO  Tenant: Tenant1
  21:00:41,393 19-Nov-2015 INFO  Service name: Deployment1
  21:00:41,393 19-Nov-2015 INFO  Service version: 0.0
  21:00:41,393 19-Nov-2015 INFO  Deployment ID: d96e5a87-cbed-4f81-ad01-d916d7e6e2ae
  21:00:41,393 19-Nov-2015 INFO  Deployment name: Deployment1
  21:00:41,393 19-Nov-2015 INFO  VM group name: CSR
  21:00:41,393 19-Nov-2015 INFO  VM Source:
  21:00:41,393 19-Nov-2015 INFO      VM ID: 75599eac-2a74-4c6b-b5a8-994ba3527d62
  21:00:41,393 19-Nov-2015 INFO      Host ID:
61ed5e10ace03237d0924c16e88330c1a5f45603b0adaebc7aea65bc
  21:00:41,393 19-Nov-2015 INFO      Host Name: none
  21:00:41,393 19-Nov-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
  21:01:40,774 19-Nov-2015 INFO
  21:01:40,774 19-Nov-2015 INFO  ===== SEND NOTIFICATION STARTS =====
  21:01:40,774 19-Nov-2015 INFO  Type: VM_DEPLOYED
  21:01:40,774 19-Nov-2015 INFO  Status: SUCCESS
  21:01:40,774 19-Nov-2015 INFO  Status Code: 200
  21:01:40,774 19-Nov-2015 INFO  Status Msg: OpenStack Driver: VM successfully created, VM
Name: [Deployment1__72075__Tenant1__SystemAdminTenantIdDeployment1Deployment10.0__0__ASA__0]
  21:01:40,774 19-Nov-2015 INFO  Tenant: Tenant1
  21:01:40,774 19-Nov-2015 INFO  Service name: Deployment1
  21:01:40,774 19-Nov-2015 INFO  Service version: 0.0
  21:01:40,774 19-Nov-2015 INFO  Deployment ID: d96e5a87-cbed-4f81-ad01-d916d7e6e2ae
  21:01:40,775 19-Nov-2015 INFO  Deployment name: Deployment1
  21:01:40,775 19-Nov-2015 INFO  VM group name: ASA
  21:01:40,775 19-Nov-2015 INFO  VM Source:
  21:01:40,775 19-Nov-2015 INFO      VM ID: 77e3f50d-7a8a-4fe8-bfc0-0081c58df69a
  21:01:40,775 19-Nov-2015 INFO      Host ID:
61ed5e10ace03237d0924c16e88330c1a5f45603b0adaebc7aea65bc
  21:01:40,775 19-Nov-2015 INFO      Host Name: none
  21:01:40,775 19-Nov-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
  21:03:15,263 19-Nov-2015 INFO
  21:03:15,263 19-Nov-2015 INFO  ===== SEND NOTIFICATION STARTS =====
  21:03:15,263 19-Nov-2015 INFO  Type: VM_ALIVE
  21:03:15,263 19-Nov-2015 INFO  Status: SUCCESS
  21:03:15,263 19-Nov-2015 INFO  Status Code: 200
  21:03:15,263 19-Nov-2015 INFO  Status Msg: VM_Alive event received, VM ID: 
[Deployment1__72075__Tenant1__SystemAdminTenantIdDeployment1Deployment10.0__0__ASA__0]
  21:03:15,263 19-Nov-2015 INFO  Tenant: Tenant1
  21:03:15,263 19-Nov-2015 INFO  Service name: Deployment1
  21:03:15,263 19-Nov-2015 INFO  Service version: 0.0
  21:03:15,263 19-Nov-2015 INFO  Deployment ID: d96e5a87-cbed-4f81-ad01-d916d7e6e2ae
  21:03:15,263 19-Nov-2015 INFO  Deployment name: Deployment1
  21:03:15,263 19-Nov-2015 INFO  VM group name: ASA
  21:03:15,263 19-Nov-2015 INFO  VM Source:
  21:03:15,263 19-Nov-2015 INFO      VM ID: 77e3f50d-7a8a-4fe8-bfc0-0081c58df69a
  21:03:15,263 19-Nov-2015 INFO      Host ID: 
61ed5e10ace03237d0924c16e88330c1a5f45603b0adaebc7aea65bc
  21:03:15,263 19-Nov-2015 INFO      Host Name: none
  21:03:15,263 19-Nov-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
  21:04:00,177 19-Nov-2015 INFO
  21:04:00,177 19-Nov-2015 INFO  ===== SEND NOTIFICATION STARTS =====
  21:04:00,177 19-Nov-2015 INFO  Type: VM_ALIVE
  21:04:00,177 19-Nov-2015 INFO  Status: SUCCESS
  21:04:00,177 19-Nov-2015 INFO  Status Code: 200
  21:04:00,178 19-Nov-2015 INFO  Status Msg: VM_Alive event received, VM ID: 
[Deployment1__78760__Tenant1__SystemAdminTenantIdDeployment1Deployment10.0__0__CSR__0]
  21:04:00,178 19-Nov-2015 INFO  Tenant: Tenant1
  21:04:00,178 19-Nov-2015 INFO  Service name: Deployment1
  21:04:00,178 19-Nov-2015 INFO  Service version: 0.0
  21:04:00,178 19-Nov-2015 INFO  Deployment ID: d96e5a87-cbed-4f81-ad01-d916d7e6e2ae
  21:04:00,178 19-Nov-2015 INFO  Deployment name: Deployment1
  21:04:00,178 19-Nov-2015 INFO  VM group name: CSR
  21:04:00,178 19-Nov-2015 INFO  VM Source:
  21:04:00,178 19-Nov-2015 INFO      VM ID: 75599eac-2a74-4c6b-b5a8-994ba3527d62
  21:04:00,178 19-Nov-2015 INFO      Host ID: 
61ed5e10ace03237d0924c16e88330c1a5f45603b0adaebc7aea65bc
  21:04:00,178 19-Nov-2015 INFO      Host Name: none
  21:04:00,178 19-Nov-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
  21:04:00,251 19-Nov-2015 INFO
  21:04:00,251 19-Nov-2015 INFO  ===== SEND NOTIFICATION STARTS =====
  21:04:00,251 19-Nov-2015 INFO  Type: SERVICE_ALIVE
  21:04:00,251 19-Nov-2015 INFO  Status: SUCCESS
  21:04:00,251 19-Nov-2015 INFO  Status Code: 200
  21:04:00,251 19-Nov-2015 INFO  Status Msg: Service group deployment completed successfully!
  21:04:00,251 19-Nov-2015 INFO  Tenant: Tenant1
  21:04:00,251 19-Nov-2015 INFO  Service name: Deployment1
  21:04:00,251 19-Nov-2015 INFO  Service version: 0.0
  21:04:00,251 19-Nov-2015 INFO  Deployment ID: d96e5a87-cbed-4f81-ad01-d916d7e6e2ae
```

```
21:04:00,251 19-Nov-2015 INFO  Deployment name: Deployment1
21:04:00,251 19-Nov-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
```

If an ESC instance is deployed with HA, there will also be an `esc_hahealth.log`. There are other logs in the directory that may be useful for troubleshooting in some situations.

NETCONF Logs

NETCONF logs are written to the `netconf.log` file and trace files are available on the ESC at `/cisco/esc-confd/` directory. NETCONF trace files log all RPC requests and replies to and from NSO. Example of NETCONF log are shown in the code below.

```
19-Nov-2015::00:55:52.299 **> sess:15 read:
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.0</capability>
    <capability>urn:ietf:params:netconf:base:1.1</capability>
  </capabilities>
</hello>
]]>]]>
#251
<rpc message-id="1" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-config>
    <source>
      <running/>
    </source>
    <filter>
      <esc_datamodel xmlns="http://www.cisco.com/esc/esc"/>
    </filter>
  </get-config>
</rpc>
```

```
19-Nov-2015::00:55:52.397 **< sess:15 write:
<rpc-reply message-id="1" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <esc_datamodel xmlns="http://www.cisco.com/esc/esc">
      <tenants>
        <tenant>
          <name>Symphony_VNF_P-1B_IntegrationTest</name>
        </tenant>
      </tenants>
    </esc_datamodel>
  </data>
</rpc-reply>
```

6.6 VNF Recovery

Overview

In the ESC VNF recovery workflow, the recovery policy in the data model can be configured by the user with three options:

- REBOOT_THEN_REDEPLOY (default)
- REDEPLOY_ONLY
- REBOOT_ONLY

Recovery workflow performs with one of the options. VNF lifecycle management is performed by ESC. As part of this functionality, ESC will try to recover VNFs when they are unreachable in three ways:

- In "REBOOT_THEN_REDEPLOY" scenario, ESC will try to recover VNFs by rebooting them as the first. If the VNFs are still unreachable, then ESC will try to recover VNFs by redeploying the VNF.
- In "REDEPLOY_ONLY" scenario, ESC will try to recover VNFs by re-deploying the VNF.
- In "REBOOT_ONLY" scenario, ESC will try to recover VNFs by rebooting the VNF.

Following output shows the part of XML ESC configuration that shows configured Recovery mode.

```
<vm_group>
......

 <recovery_policy>
  <action_on_recovery>
  REBOOT_THEN_REDEPLOY
  </action_on_recovery>
 </recovery_policy>
</vm_group>
```

Pre-requisites

Three pre-requistets must be met:

1. Reboot only option: before recovery workflow is triggered, the VM should be in the VM_ALIVE state.
2. If a recovery is triggered before the VM goes alive, ESC will directly try to redeploy the VM, since, in this case, rebooting the VM is not likely to fix the issue and no user-made configs are in place.
3. Recovery will only be triggered if the XML has the following configuration.

```
<rules>
...
<admin_rules>
<rule>
<event_name>VM_ALIVE</event_name>
<action>"ALWAYS log"</action>
<action>"FALSE recover autohealing"</action>
<action>"TRUE servicebooted.sh"</action>
</rule>
</admin_rules>
</rules>
```

ESC VNF Recovery Workflow

The detailed algorithm of ESC VNF recovery, for all three recovery options, is shown below.

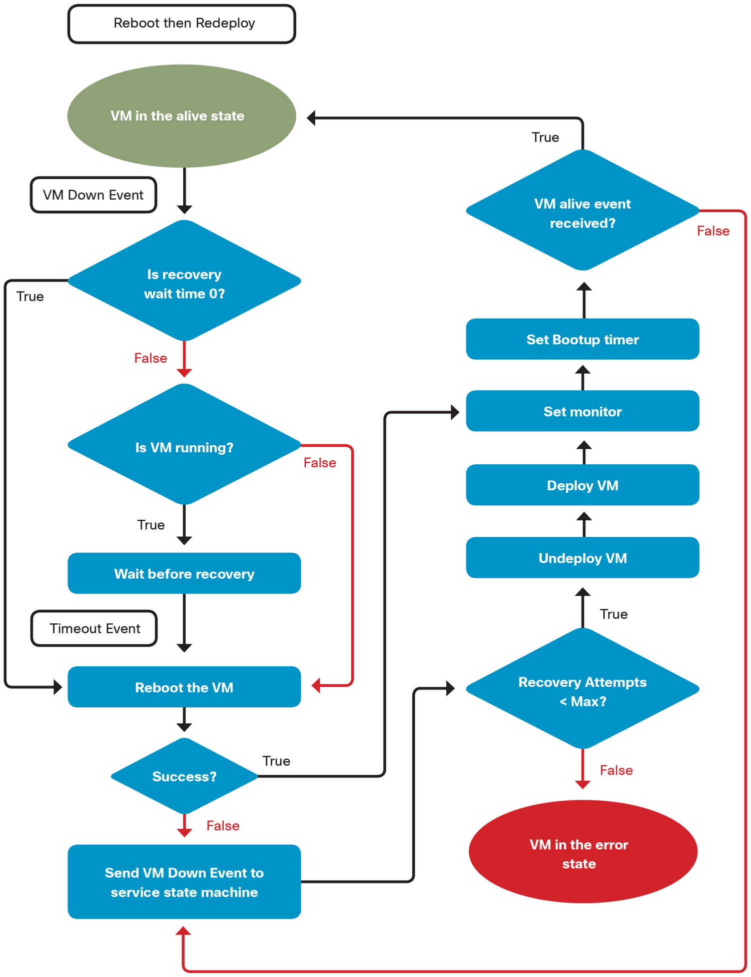
REBOOT_THEN_REDEPLOY ESC Recovery Option

If REBOOT_THEN_REDEPLOY ESC option is used:

- ESC manager tries to reboot the VM first
- if this fails, then the VNF will be redeployed.

Detailed ESC Reboot then Redeploy recovery algorithm shown in the following figure.

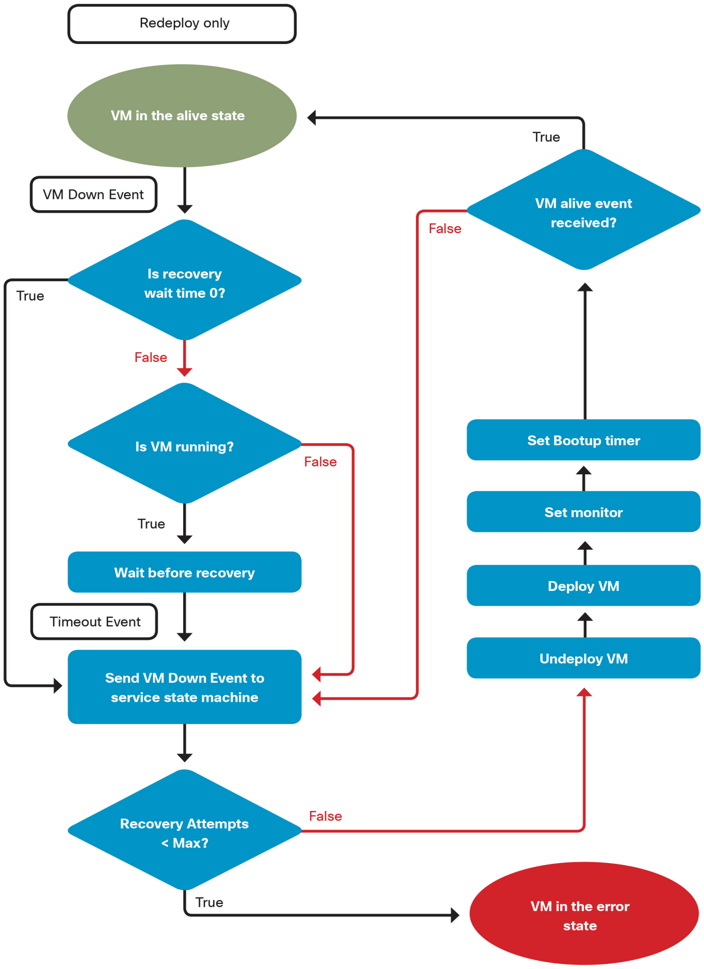
Figure 6.4 ESC VNF Recovery: Reboot then Redeploy Option



REDEPLOY_ONLY ESC Recovery Option

If REDEPLOY_ONLY recovery option is used, ESC manager redeploys the VM directly. The detailed ESC Redeploy Only recovery algorithm is shown in the following figure.

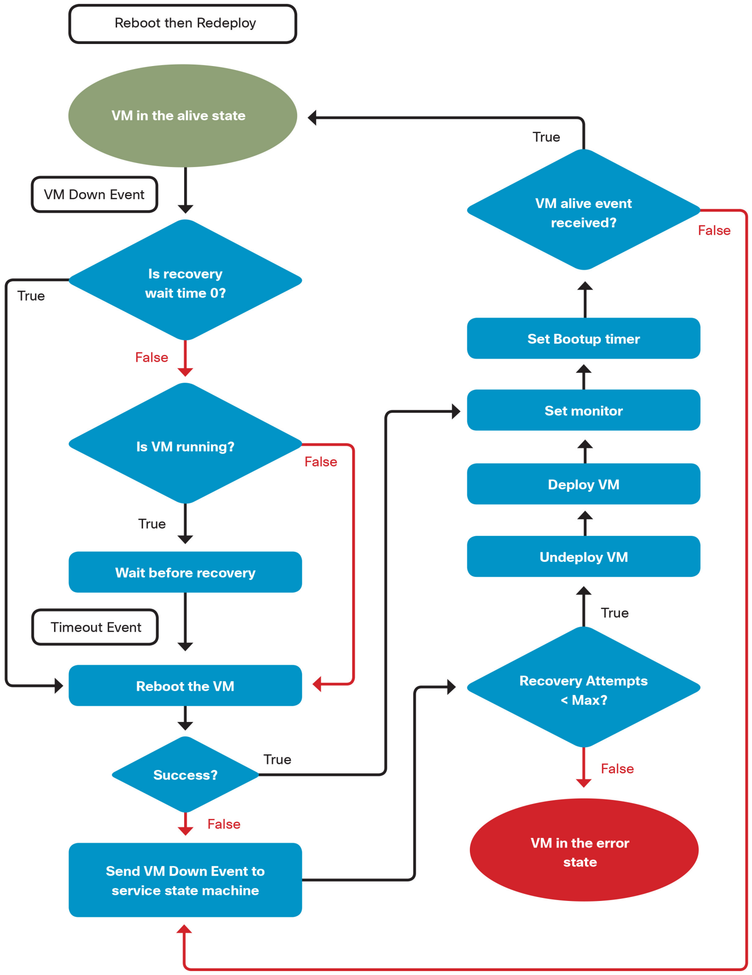
Figure 6.5 ESC VNF Recovery: Redeploy Only Option



REBOOT_ONLY ESC Recovery Option

If REBOOT_ONLY option is used, ESC manager reboots the VM, if failed, the VM goes to the error state, and recovery workflow goes to the error state. Detailed ESC Reboot Only recovery algorithm is shown in the following figure.

Figure 6.6 ESC VNF Recovery: Reboot Only Option



Note: VM running means that VM is powered up. Reboot is only for VM which is either powered up, or powered off and shut off.

6.7 Troubleshooting

To effectively troubleshoot the vMS Orchestration stack, an understanding of the high-level call-flow between NSO and ESC, and ESC and OpenStack is essential.

Figure 6.7 End-to-end Service Chain Creation Call Flow

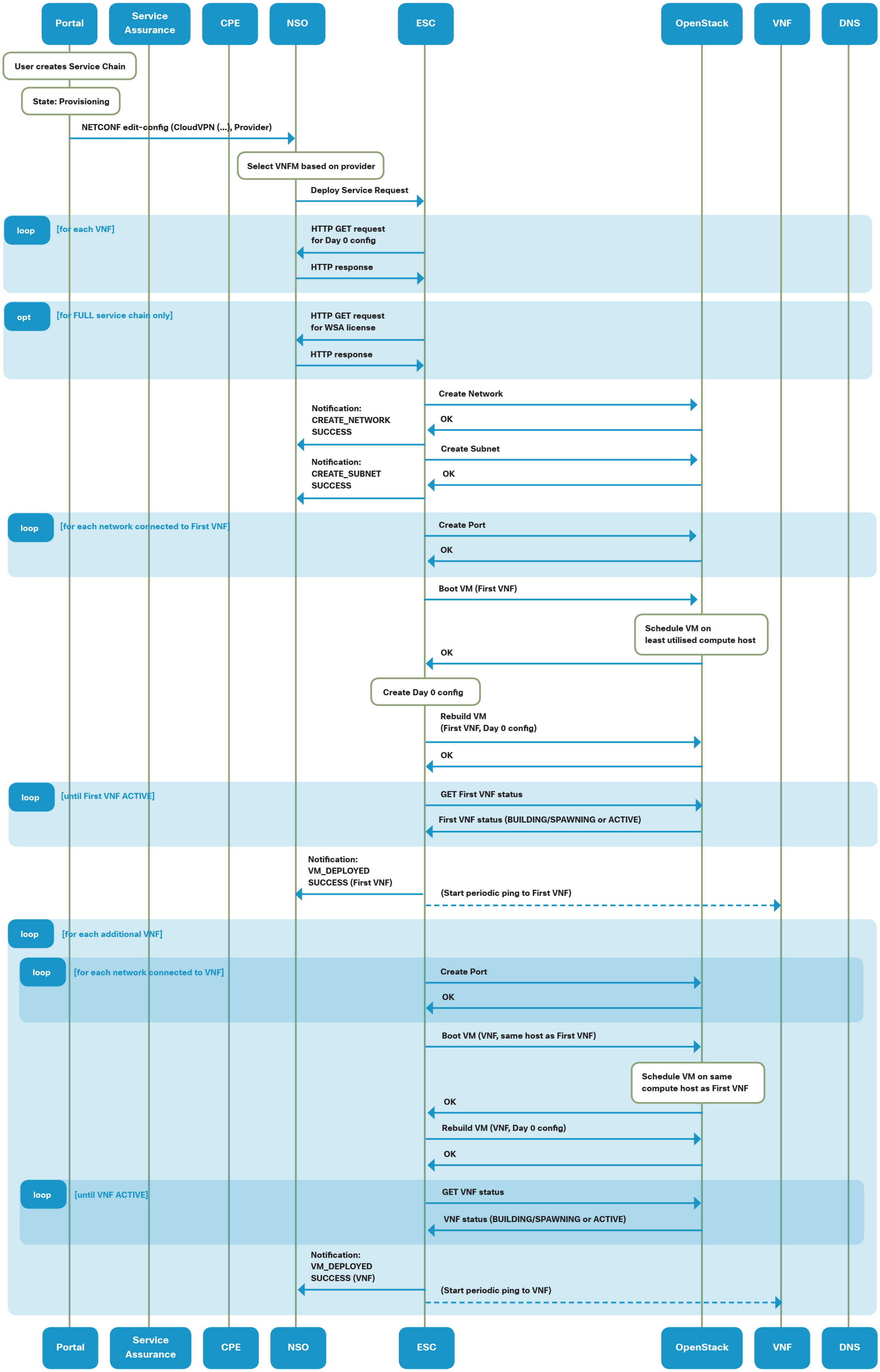
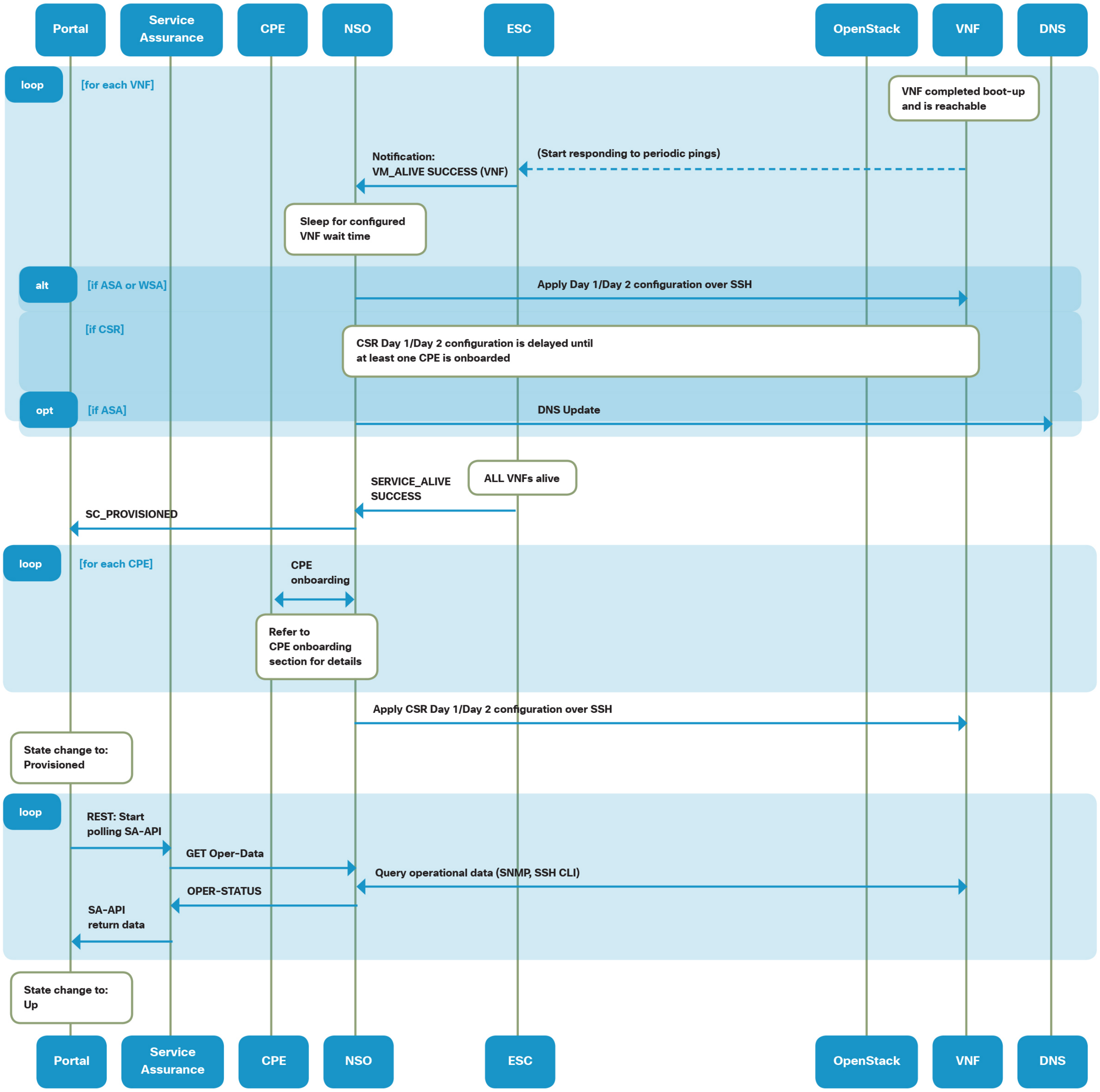

Portal
Service Assurance
CPE
NSO
ESC
OpenStack
VNF
DNS
User creates Service Chain
State: Provisioning
NETCONF edit-config (CloudVPN (...), Provider)
Select VNFM based on provider
Deploy Service Request
loop
[for each VNF]
HTTP GET request for Day 0 config
HTTP response
opt
[for FULL service chain only]
HTTP GET request for WSA license
HTTP response
Create Network
OK
Notification: CREATE_NETWORK SUCCESS
Create Subnet
OK
Notification: CREATE_SUBNET SUCCESS
loop
[for each network connected to First VNF]
Create Port
OK
Boot VM (First VNF)
Schedule VM on least utilised compute host
OK
Create Day 0 config
Rebuild VM (First VNF, Day 0 config)
OK
loop
[until First VNF ACTIVE]
GET First VNF status
First VNF status (BUILDING/SPAWNING or ACTIVE)
Notification: VM_DEPLOYED SUCCESS (First VNF)
(Start periodic ping to First VNF)
loop
[for each additional VNF]
loop
[for each network connected to VNF]
Create Port
OK
Boot VM (VNF, same host as First VNF)
Schedule VM on same compute host as First VNF
OK
Rebuild VM (VNF, Day 0 config)
OK
loop
[until VNF ACTIVE]
GET VNF status
VNF status (BUILDING/SPAWNING or ACTIVE)
Notification: VM_DEPLOYED SUCCESS (VNF)
(Start periodic ping to VNF)
Portal
Service Assurance
CPE
NSO
ESC
OpenStack
VNF
DNS

Diagram continues


Portal
Service Assurance
CPE
NSO
ESC
OpenStack
VNF
DNS
loop
[for each VNF]
VNF completed boot-up and is reachable
(Start responding to periodic pings)
Notification: VM_ALIVE SUCCESS (VNF)
Sleep for configured VNF wait time
alt
[if ASA or WSA]
Apply Day 1/Day 2 configuration over SSH
[if CSR]
CSR Day 1/Day 2 configuration is delayed until at least one CPE is onboarded
opt
[if ASA]
DNS Update
ALL VNFs alive
SERVICE_ALIVE SUCCESS
SC_PROVISIONED
loop
[for each CPE]
CPE onboarding
Refer to CPE onboarding section for details
Apply CSR Day 1/Day 2 configuration over SSH
State change to: Provisioned
loop
REST: Start polling SA-API
GET Oper-Data
Query operational data (SNMP, SSH CLI)
OPER-STATUS
SA-API return data
State change to: Up
Portal
Service Assurance
CPE
NSO
ESC
OpenStack
VNF
DNS

The first thing that ESC does when it receives a deployment request from the NSO is to go ahead and fetch the required Day 0 configurations for each of the VNFs that are going to be part of this service chain. In our example flow above, it will go ahead and fetch the Day 0 configurations for the CSR1000v, the ASAv and the WSAv VNFs.

Once that is successful, ESC will go ahead and authenticate with OpenStack and ask Nova Neutron to create the Bridge-Inside private network and subnet for this particular tenant Service Chain. Once ESC has successfully created the network and the corresponding subnet in OpenStack, it will send a northbound notification to NSO. Even in the case of a failure to do so, a failure message is sent northbound to NSO.

Due to specific VNF placement and affinity required as part of the Service Chain deployment, all VNFs that are part of a particular Service Chain need to be deployed on the same OpenStack compute host. ESC first boots-up only the CSR1000v VNF. It makes a note of which compute- host was used, and then boots up all of the VNFs on that particular compute node.

Once the network and subnet are successfully created, ESC will start booting up each of the VNFs in Openstack using Nova and Neutron APIs. For each of VNFs, a VM_Deployed success or failure message is sent northbound to NSO. Once each of the VNFs are deployed, ESC then starts an ICMP ping against each of the VNF's Management interface. As soon as ESC is able to ping the management interface of a particular VNF, a VM_ALIVE with status SUCCESS message is sent northbound to NSO.

ESC sends 5 pings in 15-second intervals, >79% of the pings should be successful for VM_ALIVE to be successful. This is specified in `dep.xml` and can be modified as needed. After success, ESC reports VM_ALIVE with status SUCCESS message for that VNF to the NSO.

If a VM_ALIVE is not observed for a specific VNF, verify IP connectivity between ESC and the VNF, check to see if the VNF booted successfully (having consumed its Day 0 config) by checking the VNF's console log (`nova console-log <vm id>` in OpenStack) and, if available, by accessing the VNF's VNC console in OpenStack Horizon.

If VM_ALIVE does not occur within 10 minutes, ESC deletes and recreates the VM in OpenStack. If, after several iterations, ESC still cannot create a VM that responds to a ping, ESC aborts the deployment, and sends both a VM_ALIVE and SERVICE_ALIVE with status FAILURE to the NSO.

Once all the particular VNFs needed for the Service Chain are deployed successfully and reachable by ESC, it triggers a northbound SERVICE_ALIVE message to the NSO. When the NSO receives a SERVICE_ALIVE with status SUCCESS message from ESC, this is a trigger for the NSO to then go ahead and apply the Day 1/Day 2 configurations for each of the VNFs.

Use the `yangesc.log` to search and follow the message sequence between the NSO and ESC (excluding the HTTP Day 0 configuration and license file requests, not shown in this file). If an expected response is missing, or FAILURE vs. SUCCESS flags appear, refer to the `escmanager.log` at the timestamp in question to isolate error origination.

Troubleshooting ESC OpenStack related issues

As previously shown in the Service Chain Creation flow diagrams, the final step of deploying a Service Chain is done by ESC and OpenStack solution components.

If Service Chain manipulation fails due to ESC-OpenStack communication, there are two available ESC tools to troubleshoot the issues further:

- Nova and Neutron Python clients
- ESC manager logs

Nova and Neutron Python Clients

OpenStack Python clients can be used to verify basic OpenStack API availability and functionality. The usage of the clients is covered in more details in Chapter 9.2 "OpenStack Health Check".

ESC manager logs

The ESC manager logs can be found at `/var/log/esc/escmanager.log` and contain all communication between ESC and OpenStack APIs. For each API call made by ESC it records:

1. A line that shows the API URL and request body (if any)
2. A line that shows the response status code
3. Lines showing the response contents

An example is shown below.

```
  10:30:20,769 26-Nov-2015 Thread-7 INFO  [OpenStackDriver.java:getOpenStackObject():587]
[tid=escmanager] Sending GET to http://controller:9696/v2.0/networks?name=internet

  10:30:20,787 26-Nov-2015 Thread-7 INFO  [OpenStackDriver.java:getOpenStackObject():602]
[tid=escmanager] GET status code returned from Openstack: 200

  10:30:20,787 26-Nov-2015 Thread-7 INFO  [OpenStackDriver.java:jsonToNetworkList():1004]
[tid=escmanager] Network list response from Openstack: {"networks":
[{"tenant_id":"931829a9d81546669b8b879e623f0238","provider:physical_network":"physnet1","shared
":true,"router:external":false,"provider:segmentation_id":110,"admin_state_up":true,"name":"int
ernet","provider:network_type":"vlan","subnets":["3a10abd2-d6fb-4b8c-a7c0-
fd13ed8b00b1"],"id":"5903884c-6e36-4d61-ab84-9867b513bf92","status":"ACTIVE","mtu":0}]}

  10:30:20,787 26-Nov-2015 Thread-7 INFO  [OpenStackDriver.java:jsonToNetwork():945]
[tid=escmanager] Network response from Openstack: {"network":
{"tenant_id":"931829a9d81546669b8b879e623f0238","provider:physical_network":"physnet1","shared"
:true,"router:external":false,"provider:segmentation_id":110,"admin_state_up":true,"name":"inte
rnet","provider:network_type":"vlan","subnets":["3a10abd2-d6fb-4b8c-a7c0-
fd13ed8b00b1"],"id":"5903884c-6e36-4d61-ab84-9867b513bf92","status":"ACTIVE","mtu":0}}
```

One of the ways to filter out the API call log lines is to apply the following grep filter:

```
  $ grep OpenStackDriver /var/log/esc/escmanager.log | egrep "Sending|from Openstack" | less
```

Using the `less` Linux tool, a particular object name can be searched (ex. Neutron network name) or response status codes (ex. 400 or 500). OpenStack API status codes are standard HTTP status codes which can be found at https://en.wikipedia.org/wiki/List_of_HTTP_status_codes.

The following three groups are of most interesting for ESC operation:

- 2xx - Success
- 4xx - Client side error
- 5xx - Server side error

Sample ESC yangesc.log snippet for a successful service chain deployment

The `/var/log/yangesc.log` logfile captures the communication between the NSO and the ESC. In the example below, ESC has received a new DEPLOY_SERVICE request from the NSO. This request has been accepted by ESC.

```
  07:11:22,485 08-Dec-2015 INFO  Order size: 1
  07:11:22,485 08-Dec-2015 INFO  Order: DEPLOY_SERVICE
  07:11:32,798 08-Dec-2015 INFO
  07:11:32,798 08-Dec-2015 INFO  ===== DEPLOY SERVICE REQUEST RECEIVED (UNDER TENANT) =====
  07:11:32,798 08-Dec-2015 INFO  Tenant name: cns-auto1b
  07:11:32,798 08-Dec-2015 INFO  Deployment name: cns-auto1b-rtpco-cloudvpn
  07:11:37,951 08-Dec-2015 INFO  =====  DEPLOY SERVICE REQUEST ACCEPTED  (UNDER TENANT)  =====
```

ESC then goes ahead and creates the BR-INSIDE network and subnet and sends notification upstream to the NSO for both.

```
  07:11:44,419 08-Dec-2015 INFO  ===== SEND NOTIFICATION STARTS =====
  07:11:44,419 08-Dec-2015 INFO  Type: CREATE_NETWORK
  07:11:44,419 08-Dec-2015 INFO  Status: SUCCESS
  07:11:44,419 08-Dec-2015 INFO  Status Code: 200
  07:11:44,419 08-Dec-2015 INFO  Status Msg: Network creation for [BR-INSIDE-01-cns-auto1b-
rtpco-cloudvpn] completed successfully!
  07:11:44,419 08-Dec-2015 INFO  Tenant: cns-auto1b
  07:11:44,419 08-Dec-2015 INFO  Network: cns-auto1bcns-auto1b-rtpco-cloudvpnBR-INSIDE-01-cns-
auto1b-rtpco-cloudvpn
  07:11:44,419 08-Dec-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
  07:11:47,360 08-Dec-2015 INFO  ===== SEND NOTIFICATION STARTS =====
  07:11:47,360 08-Dec-2015 INFO  Type: CREATE_SUBNET
  07:11:47,360 08-Dec-2015 INFO  Status: SUCCESS
  07:11:47,360 08-Dec-2015 INFO  Status Code: 200
  07:11:47,360 08-Dec-2015 INFO  Status Msg: Subnet creation completed successfully, subnet ID:
[cns-auto1bcns-auto1b-rtpco-cloudvpnBR-INSIDE-01-cns-auto1b-rtpco-cloudvpnBR-INSIDE-01-cns-
auto1b-rtpco-cloudvpn-BOGUS-CVPN-SUBNET]
  07:11:47,360 08-Dec-2015 INFO  Tenant: cns-auto1b
  07:11:47,360 08-Dec-2015 INFO  Network: cns-auto1bcns-auto1b-rtpco-cloudvpnBR-INSIDE-01-cns-
auto1b-rtpco-cloudvpn
  07:11:47,360 08-Dec-2015 INFO  Subnet: cns-auto1bcns-auto1b-rtpco-cloudvpnBR-INSIDE-01-cns-
auto1b-rtpco-cloudvpnBR-INSIDE-01-cns-auto1b-rtpco-cloudvpn-BOGUS-CVPN-SUBNET
  07:11:47,360 08-Dec-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
```

Once the network and subnet are successfully created, ESC starts deploying each of the VNFs. Then ESC informs NSO that it was able to successfully deploy the CSR1000v VNF.

```
  07:13:34,863 08-Dec-2015 INFO  ===== SEND NOTIFICATION STARTS =====
  07:13:34,863 08-Dec-2015 INFO  Type: VM_DEPLOYED
  07:13:34,863 08-Dec-2015 INFO  Status: SUCCESS
  07:13:34,863 08-Dec-2015 INFO  Status Code: 200
  07:13:34,863 08-Dec-2015 INFO  Status Msg: OpenStack Driver: VM successfully created, VM
Name: [cns-auto1b-rtpco-cloudvpn__83471__cns-auto1b__SystemAdminTenantIdcns-auto1b-rtpco-
cloudvpncns-auto1b-rtpco-cloudvpn0.0__0__CSR__0]
  07:13:34,863 08-Dec-2015 INFO  Tenant: cns-auto1b
  07:13:34,863 08-Dec-2015 INFO  Service name: cns-auto1b-rtpco-cloudvpn
  07:13:34,864 08-Dec-2015 INFO  Service version: 0.0
  07:13:34,864 08-Dec-2015 INFO  Deployment ID: 8a89e619-9908-474a-b8fc-7129abf50360
  07:13:34,864 08-Dec-2015 INFO  Deployment name: cns-auto1b-rtpco-cloudvpn
  07:13:34,864 08-Dec-2015 INFO  VM group name: CSR
  07:13:34,864 08-Dec-2015 INFO  VM Source:
  07:13:34,864 08-Dec-2015 INFO      VM ID: 113dbc97-ce3a-4c77-9544-2a3fee44e241
  07:13:34,864 08-Dec-2015 INFO      Host ID:
c0e8140a0a2f079759edc9f22dcab6adaf3e18a1f4c7b9dbf592dbc9
  07:13:34,864 08-Dec-2015 INFO      Host Name: none
  07:13:34,864 08-Dec-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
```

Following the CSR1000v, ESC was successfully able to deploy the ASAv VNF.

```
  07:14:36,803 08-Dec-2015 INFO  ===== SEND NOTIFICATION STARTS =====
  07:14:36,803 08-Dec-2015 INFO  Type: VM_DEPLOYED
  07:14:36,803 08-Dec-2015 INFO  Status: SUCCESS
  07:14:36,803 08-Dec-2015 INFO  Status Code: 200
  07:14:36,803 08-Dec-2015 INFO  Status Msg: OpenStack Driver: VM successfully created, VM
Name: [cns-auto1b-rtpco-cloudvpn__54846__cns-auto1b__SystemAdminTenantIdcns-auto1b-rtpco-
cloudvpncns-auto1b-rtpco-cloudvpn0.0__0__ASA__0]
  07:14:36,803 08-Dec-2015 INFO  Tenant: cns-auto1b
  07:14:36,804 08-Dec-2015 INFO  Service name: cns-auto1b-rtpco-cloudvpn
  07:14:36,804 08-Dec-2015 INFO  Service version: 0.0
  07:14:36,804 08-Dec-2015 INFO  Deployment ID: 8a89e619-9908-474a-b8fc-7129abf50360
  07:14:36,804 08-Dec-2015 INFO  Deployment name: cns-auto1b-rtpco-cloudvpn
  07:14:36,804 08-Dec-2015 INFO  VM group name: ASA
  07:14:36,804 08-Dec-2015 INFO  VM Source:
  07:14:36,804 08-Dec-2015 INFO      VM ID: 98e9a8cd-5bf6-46f3-9c12-fbccd2286cb7
  07:14:36,804 08-Dec-2015 INFO      Host ID:
c0e8140a0a2f079759edc9f22dcab6adaf3e18a1f4c7b9dbf592dbc9
  07:14:36,804 08-Dec-2015 INFO      Host Name: none
  07:14:36,804 08-Dec-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
```

Once ESC deploys a VNF, it starts to ping the management interface to test whether it can reach the VNF, and it uses this to describe the VM as ALIVE. As ESC is able to successfully reach the CSR1000v management interface, it sends a northbound VM_ALIVE (CSR1000v) message to the NSO.

```
  07:16:30,246 08-Dec-2015 INFO  ===== SEND NOTIFICATION STARTS =====
  07:16:30,246 08-Dec-2015 INFO  Type: VM_ALIVE
  07:16:30,246 08-Dec-2015 INFO  Status: SUCCESS
  07:16:30,246 08-Dec-2015 INFO  Status Code: 200
  07:16:30,246 08-Dec-2015 INFO  Status Msg: VM_Alive event received, VM ID: [cns-auto1b-rtpco-
cloudvpn__83471__cns-auto1b__SystemAdminTenantIdcns-auto1b-rtpco-cloudvpncns-auto1b-rtpco-
cloudvpn0.0__0__CSR__0]
  07:16:30,247 08-Dec-2015 INFO  Tenant: cns-auto1b
  07:16:30,247 08-Dec-2015 INFO  Service name: cns-auto1b-rtpco-cloudvpn
  07:16:30,247 08-Dec-2015 INFO  Service version: 0.0
  07:16:30,247 08-Dec-2015 INFO  Deployment ID: 8a89e619-9908-474a-b8fc-7129abf50360
  07:16:30,247 08-Dec-2015 INFO  Deployment name: cns-auto1b-rtpco-cloudvpn
  07:16:30,247 08-Dec-2015 INFO  VM group name: CSR
  07:16:30,247 08-Dec-2015 INFO  VM Source:
  07:16:30,247 08-Dec-2015 INFO      VM ID: 113dbc97-ce3a-4c77-9544-2a3fee44e241
  07:16:30,247 08-Dec-2015 INFO      Host ID:
c0e8140a0a2f079759edc9f22dcab6adaf3e18a1f4c7b9dbf592dbc9
  07:16:30,247 08-Dec-2015 INFO      Host Name: none
  07:16:30,247 08-Dec-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
```

A VM_ALIVE (ASAv) message is sent northbound from ESC to NSO.

```
  07:17:01,964 08-Dec-2015 INFO  ===== SEND NOTIFICATION STARTS =====
  07:17:01,964 08-Dec-2015 INFO  Type: VM_ALIVE
  07:17:01,964 08-Dec-2015 INFO  Status: SUCCESS
  07:17:01,964 08-Dec-2015 INFO  Status Code: 200
  07:17:01,964 08-Dec-2015 INFO  Status Msg: VM_Alive event received, VM ID: [cns-auto1b-rtpco-
cloudvpn__54846__cns-auto1b__SystemAdminTenantIdcns-auto1b-rtpco-cloudvpncns-auto1b-rtpco-
cloudvpn0.0__0__ASA__0]
  07:17:01,964 08-Dec-2015 INFO  Tenant: cns-auto1b
  07:17:01,964 08-Dec-2015 INFO  Service name: cns-auto1b-rtpco-cloudvpn
  07:17:01,964 08-Dec-2015 INFO  Service version: 0.0
  07:17:01,964 08-Dec-2015 INFO  Deployment ID: 8a89e619-9908-474a-b8fc-7129abf50360
  07:17:01,964 08-Dec-2015 INFO  Deployment name: cns-auto1b-rtpco-cloudvpn
  07:17:01,964 08-Dec-2015 INFO  VM group name: ASA
  07:17:01,964 08-Dec-2015 INFO  VM Source:
  07:17:01,964 08-Dec-2015 INFO      VM ID: 98e9a8cd-5bf6-46f3-9c12-fbccd2286cb7
  07:17:01,964 08-Dec-2015 INFO      Host ID:
c0e8140a0a2f079759edc9f22dcab6adaf3e18a1f4c7b9dbf592dbc9
  07:17:01,964 08-Dec-2015 INFO      Host Name: none
  07:17:01,964 08-Dec-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
```

ESC informs NSO that it was able to successfully deploy the last VNF (WSAv) as part of the Service Chain.

```
  07:23:25,693 08-Dec-2015 INFO  ===== SEND NOTIFICATION STARTS =====
  07:23:25,693 08-Dec-2015 INFO  Type: VM_DEPLOYED
  07:23:25,693 08-Dec-2015 INFO  Status: SUCCESS
  07:23:25,693 08-Dec-2015 INFO  Status Code: 200
  07:23:25,693 08-Dec-2015 INFO  Status Msg: OpenStack Driver: VM successfully created, VM
Name: [cns-auto1b-rtpco-cloudvpn__53637__cns-auto1b__SystemAdminTenantIdcns-auto1b-rtpco-
cloudvpncns-auto1b-rtpco-cloudvpn0.0__0__WSA__0]
  07:23:25,694 08-Dec-2015 INFO  Tenant: cns-auto1b
  07:23:25,694 08-Dec-2015 INFO  Service name: cns-auto1b-rtpco-cloudvpn
  07:23:25,694 08-Dec-2015 INFO  Service version: 0.0
  07:23:25,694 08-Dec-2015 INFO  Deployment ID: 8a89e619-9908-474a-b8fc-7129abf50360
  07:23:25,694 08-Dec-2015 INFO  Deployment name: cns-auto1b-rtpco-cloudvpn
  07:23:25,694 08-Dec-2015 INFO  VM group name: WSA
  07:23:25,694 08-Dec-2015 INFO  VM Source:
  07:23:25,694 08-Dec-2015 INFO      VM ID: 01a84f2a-68cc-4b06-8c08-de3d80b0835f
  07:23:25,694 08-Dec-2015 INFO      Host ID:
c0e8140a0a2f079759edc9f22dcab6adaf3e18a1f4c7b9dbf592dbc9
  07:23:25,694 08-Dec-2015 INFO      Host Name: none
  07:23:25,694 08-Dec-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
```

Once ESC is able to reach the management interface of WSAv, it sends a VM_ALIVE (WSAv) northbound to the NSO.

```
  07:24:21,373 08-Dec-2015 INFO  ===== SEND NOTIFICATION STARTS =====
  07:24:21,373 08-Dec-2015 INFO  Type: VM_ALIVE
  07:24:21,373 08-Dec-2015 INFO  Status: SUCCESS
  07:24:21,373 08-Dec-2015 INFO  Status Code: 200
  07:24:21,373 08-Dec-2015 INFO  Status Msg: VM_Alive event received, VM ID: [cns-auto1b-rtpco-
cloudvpn__53637__cns-auto1b__SystemAdminTenantIdcns-auto1b-rtpco-cloudvpncns-auto1b-rtpco-
cloudvpn0.0__0__WSA__0]
  07:24:21,373 08-Dec-2015 INFO  Tenant: cns-auto1b
  07:24:21,373 08-Dec-2015 INFO  Service name: cns-auto1b-rtpco-cloudvpn
  07:24:21,373 08-Dec-2015 INFO  Service version: 0.0
  07:24:21,373 08-Dec-2015 INFO  Deployment ID: 8a89e619-9908-474a-b8fc-7129abf50360
  07:24:21,373 08-Dec-2015 INFO  Deployment name: cns-auto1b-rtpco-cloudvpn
  07:24:21,373 08-Dec-2015 INFO  VM group name: WSA
  07:24:21,373 08-Dec-2015 INFO  VM Source:
  07:24:21,374 08-Dec-2015 INFO      VM ID: 01a84f2a-68cc-4b06-8c08-de3d80b0835f
  07:24:21,374 08-Dec-2015 INFO      Host ID:
c0e8140a0a2f079759edc9f22dcab6adaf3e18a1f4c7b9dbf592dbc9
  07:24:21,374 08-Dec-2015 INFO      Host Name: none
  07:24:21,374 08-Dec-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
```

Since all the three VNFs (CSR1000v, ASAv and WSAv) required for this particular deployment were deployed successfully and reachable via their management interface, the final message sent by ESC to NSO is a SERVICE_ALIVE notification.

```
  07:24:21,433 08-Dec-2015 INFO  ===== SEND NOTIFICATION STARTS =====
  07:24:21,433 08-Dec-2015 INFO  Type: SERVICE_ALIVE
  07:24:21,433 08-Dec-2015 INFO  Status: SUCCESS
  07:24:21,433 08-Dec-2015 INFO  Status Code: 200
  07:24:21,433 08-Dec-2015 INFO  Status Msg: Service group deployment completed successfully!
  07:24:21,433 08-Dec-2015 INFO  Tenant: cns-auto1b
  07:24:21,433 08-Dec-2015 INFO  Service name: cns-auto1b-rtpco-cloudvpn
  07:24:21,433 08-Dec-2015 INFO  Service version: 0.0
  07:24:21,433 08-Dec-2015 INFO  Deployment ID: 8a89e619-9908-474a-b8fc-7129abf50360
  07:24:21,433 08-Dec-2015 INFO  Deployment name: cns-auto1b-rtpco-cloudvpn
  07:24:21,433 08-Dec-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
```

Common ESC Errors and Resolution

ESC HA Failure

The most common reasons of ESC HA failure occur when PostgreSQL fails to start.

Symptoms:

1. ESC is running in HA mode
2. ESC Master Failure and switchover to Backup ESC, however backup ESC is not functioning as Master

Inspect the `esc_haagent.log` to see if by any chance it failed to start PostgreSQL service.

```
  Starting postgresql-9.4 service: [FAILED]
```

The workaround is to manually start PostgreSQL, as shown below.

```
  $ sudo /etc/init.d/postgresql-9.4 start
```

Day 0 Configuration Failure

Once in a while, issues are encountered when the Day 0 configuration is not applied to a VNF or VNF(s). This could occur for several reasons. One of the issues could be due to ESC not being able to fetch the Day 0 configurations from the NSO. This could be because the configurations are not present at the specified location, or the specified location is incorrect.

1. Obtain location of Day 0 configuration

To determine the location of where ESC should fetch the Day 0 configuration for a particular VNF, execute the following command in ESC:

```
  [admin@esc ~]$ /opt/confd/bin/confd_cli
  admin@esc> show configuration esc_datamodel tenants tenant <tenant_name> deployments
deployment <deployment_name>
```

From the output, inspect the following configuration:

```
  config_data {
          configuration iosxe_config.txt {
              file https://10.32.1.23:443/day0/cfg/vRouter/node/basic1-csr/provider/cns-auto1b;
          }
```

2. Check basic network connectivity between ESC server and HTTPS server that is hosting Day 0 configuration, by executing a ping test to the IP address in URL:

```
  [admin@esc-test-2 ~]$ ping -c 5 -n 10.32.1.23
  PING 10.32.1.23 (10.32.1.23) 56(84) bytes of data.
  64 bytes from 10.32.1.23: icmp_seq=1 ttl=64 time=0.387 ms
  64 bytes from 10.32.1.23: icmp_seq=2 ttl=64 time=0.470 ms
  64 bytes from 10.32.1.23: icmp_seq=3 ttl=64 time=0.234 ms
  64 bytes from 10.32.1.23: icmp_seq=4 ttl=64 time=0.306 ms
  64 bytes from 10.32.1.23: icmp_seq=5 ttl=64 time=0.213 ms

  --- 10.32.1.23 ping statistics ---
  5 packets transmitted, 5 received, 0% packet loss, time 3998ms
  rtt min/avg/max/mdev = 0.213/0.322/0.470/0.095 ms
```

Additionally, connectivity to Web server hosting Day 0 configurations can be checked using `wget` as shown in the example output below.

```
  [admin@esc-test-2 ~]$ wget --no-check-certificate
https://10.32.1.23:443/day0/cfg/vRouter/node/basic1-csr/provider/cns-auto1b
  --2015-12-08 16:13:00--  https://10.32.1.23/day0/cfg/vRouter/node/basic1-csr/provider/cns-
auto1b
  Connecting to 10.32.1.23:443... connected.
  WARNING: cannot verify 10.32.1.23's certificate, issued by "/C=US/ST=California/O=Internet
Widgits Pty Ltd/CN=John Smith":
    Unable to locally verify the issuer's authority.
      WARNING: certificate common name "" doesn't match requested host name "10.32.1.23".
  HTTP request sent, awaiting response... 200 OK
  Length: 3157 (3.1K) [text/plain]
  Saving to: "cns-auto1b"

  100%[====================================================>] 3,157       --.-K/s   in 0s

  2015-12-08 16:13:00 (325 MB/s) - "cns-auto1b" saved [3157/3157]
```

In case of a success, the Day 0 configuration is successfully downloaded to the ESC. If not, please check either the path specified is correctly, and whether the template files areactually present at the specified location.

ESC will query the following URLs on NSO (NSO-SM for clustered setup) through HTTPS to retrieve the configurations:

- For the ASAv: `https://<NSO-IP>:443/day0/cfg/vFirewall/node/<tenant>-<cloudvpn>-ASA-<esc-device>/provider/<provider>`
- For the CSR1000v: `https://<NSO-IP>:443/day0/cfg/vRouter/node/<tenant>-<cloudvpn>-CSR-<esc-device>/provider/<provider>`
- For the WSAv: `https://<NSO-IP>:443/day0/cfg/vWSA/node/<tenant>-<cloudvpn>-WSA-<esc-device>/provider/<provider>`

3. ESC pushing the Configuration to OpenStack

Once it is verified that ESC is getting Day 0 configuration from NSO, next step would be to check that ESC is pushing those configurations to OpenStack.

This data is present in the `/var/log/esc/escmanager.log` file and for each VNF, binary data will be seen in the logs and this would be the configuration sent to OpenStack:

```
  [localadmin@esc esc]$ grep imageRef escmanager.log | tail
  08:17:36,308 08-Dec-2015 VM_STATE_MACHINE-ESC_Day0-3__64969__bru-lab1-tenant__bru-lab1-
tenanttest1ESC_Day0-31.1__0__ASA__0 INFO
  [OpenStackDriver.java:createOpenStackObject():583] [tid=ESC_Day0-3__64969__bru-lab1-
tenant__bru-lab1-tenanttest1ESC_Day0-31.1__0__ASA__0]
  Sending POST to http://10.20.0.154:8774/v2/3cf9f12d07e04eb4ac07889295a9448b/servers/fad4202c-
38ba-495b-9df3-11ad1931880a/action with body
  {"rebuild" : {  "imageRef" : "1845a570-dad5-4083-9e2a-673effdb3d35", "personality":
[{"path":"asa_config.txt","contents":"QVNBIFZlcnNpb24gOS4zKD
  <Binary data>
  lbW9yeQo="}]}}
```

If a similar entry is shown by `escmanager.log`, for each VNF, it means the configuration is pushed by ESC to OpenStack successfully.

WSAv Day 0 Configuration Issues due to OpenStack Nova quota

If there are issues with WSAv boot-up and Day 0 configuration, one of the potential reasons could be due to the default nova quota settings in OpenStack. Default value for injected_file_content_bytes is **10240.**

Fetch Day 0 configuration for the VNF and check file size. If WSA Day 0 configuration file size is greater than **10240,** the quota needs to be increased in OpenStack.

```
  $ keystone tenant-list
    +----------------------------------+-------------+---------+
  |                id                |     name    | enabled |
  +----------------------------------+-------------+---------+
  | 931829a9d81546669b8b879e623f0238 |    admin    |   True  |
  | 80ec1304cb774393a5d2afe1fd832a78 |   service   |   True  |
  | 6b2e7e73c0954fe9b17adf84311db980 |     test    |   True  |
  | 5a3a0fe65b4747d19ea05476a1220ea4 | vms-lab-pod |   True  |
  +----------------------------------+-------------+---------+
```

Check Nova Neutron quotas for this particular tenant.

```
$ nova quota-show --tenant 5a3a0fe65b4747d19ea05476a1220ea4
+-----------------------------+--------+
| Quota                       | Limit  |
+-----------------------------+--------+
| instances                   | 100    |
| cores                       | 200    |
| ram                         | 512000 |
| floating_ips                | 10     |
| fixed_ips                   | -1     |
| metadata_items              | 128    |
| injected_files              | 5      |
| injected_file_content_bytes | 10240  |
| injected_file_path_bytes    | 255    |
| key_pairs                   | 100    |
| security_groups             | 10     |
| security_group_rules        | 20     |
| server_groups               | 10     |
| server_group_members        | 10     |
+-----------------------------+--------+
```

Once you have the ID, you can increase the value of the injected file content bytes.

```
$ nova quota-update --injected_file_content_bytes 65535 5a3a0fe65b4747d19ea05476a1220ea4
```

Verify that nova quota is updated with the new value.

```
$ nova quota-show --tenant 5a3a0fe65b4747d19ea05476a1220ea4
+-----------------------------+--------+
| Quota                       | Limit  |
+-----------------------------+--------+
| instances                   | 100    |
| cores                       | 200    |
| ram                         | 512000 |
| floating_ips                | 10     |
| fixed_ips                   | -1     |
| metadata_items              | 128    |
| injected_files              | 5      |
| injected_file_content_bytes | 65535  |
| injected_file_path_bytes    | 255    |
| key_pairs                   | 100    |
| security_groups             | 10     |
| security_group_rules        | 20     |
| server_groups                | 10     |
| server_group_members         | 10     |
+-----------------------------+--------+
```

VNF not accepting Day 0 Configuration

Even when the configuration is properly attached to the device, there can be cases where the device is simply rejecting Day 0 configuration because it is not properly formatted.

Another reason for the Day 0 configuration not being applied is bthat ESC could not correctly replace the variables called "Token Replacement" or else some variables passed to the Day 0 configuration are not properly formatted.

VNF logs can provide more information on any errors generated during application of the Day 0 configuration.

7. Customer Premise Equipment (CPE)

7.1 Introduction

This chapter discusses the Customer Premise Equipment (CPE) used in CloudVPN. The topics covered include - how to on-board a CPE into a CloudVPN service, operational call flows and troubleshooting steps.

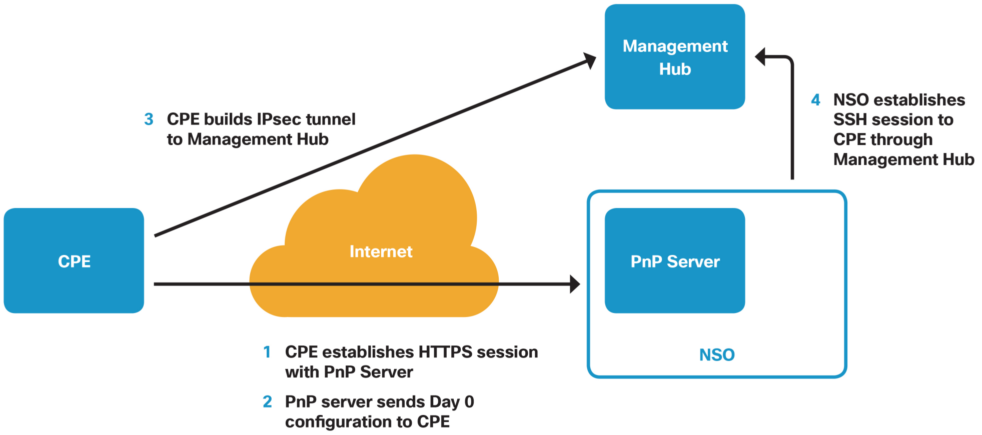
This chapter is divided into seven sections that cover the following topics in detail:

- Overview
- Requirements
- Preparation and Verification
- On-boarding Call Flow
- Delete Call Flow
- Replacement Procedure
- Troubleshooting

7.2 Overview

Customer Premise Equipment supported in the vMS solution currently comes from the Cisco ISR-G2 family of IOS-based routers. These include the 800, 1900, 2900 and 3900 series ISR-G2 routers.CPE is deployed at customer sites and provides secure tunnels from an end customer network into a CloudVPN service chain. Provisioning of the CPE is automated by the vMS solution. The key IOS features utilized on these routers are: PnP (Plug and Play), FlexVPN, BGP, NBAR2 and QoS. Figure 7.1 below illustrates the Zero-Touch Deployment (ZTD) of the CPE using the PnP process.

Figure 7.1 PnP functionality and CPE on-boarding


Management Hub
3 CPE builds IPsec tunnel to Management Hub
4 NSO establishes SSH session to CPE through Management Hub
CPE
Internet
PnP Server
NSO
1 CPE establishes HTTPS session with PnP Server
2 PnP server sends Day 0 configuration to CPE

7.3 Requirements

The key requirements for the CPE to be orchestrated and function properly in the vMS solution are:

IOS version and Feature set - vMS requires a minimum IOS version of 15.5(1)T or newer. The use of FlexVPN and IPsec requires that the IOS has a feature set that can support IPsec. This is usually designated with a "K9" in the image name, and the IOS license type should be "Advanced Security".

DHCP - The CPE needs to have an IP address on its WAN interface to connect to the PnP server (NSO) and get provisioned, and this is typically obtained via Dynamic Host Configuration Protocol (DHCP) from the Internet Service Provider (ISP). DHCP should also provide a default gateway for the CPE that allows packets to be sent across the Internet.

Internet Access - The CPE connects to the PnP server across the open Internet. The Internet connection must be capable of routing traffic across the Internet to the PnP server.

WAN port vs LAN port - Each CPE has 2 or more Ethernet connections. One of them is specifically designated as the WAN port and must be connected to the Internet. Additional information about this can be found in Chapter 7.8 "CPE Troubleshooting".

NAT - The CPE can be behind a NAT (Network Address Translation) device, but the device ***MUST support either IPsec NAT Traversal or IPsec passthrough*** if NAT will be used. In addition, it must be capable of supporting two IPsec tunnels. Some NAT devices can support only one tunnel. Many end-customer devices might be capable, but don't have the feature enabled by default. These devices must have the feature enabled before the CPE will be able to establish both IPsec tunnels. The configuration of this feature is critical for a CPE to work properly. Each device or Internet connection might be different. This aspect should be investigated if the IPsec tunnels fail to come up. It is best to check on the capabilities of the specific device a customer has to ensure that it can perform this function before deploying a CPE to the site.

IPsec - When fully provisioned, each CPE establishes two IPsec tunnels. The first tunnel is provisioned via the Day 0 configuration pushed by the PnP server, and is terminated on a Management Hub inside of the vMS infrastructure. This Management tunnel is used for subsequent management and provisioning of the CPE by NSO. The second tunnel is for carrying end-customer traffic to the CloudVPN service chain and is terminated on a Cloud Services Router (CSR1000v) that is part of the customer service chain. IPsec uses IP protocols 50 and 51 and UDP ports 500 and 4500. The use of these protocols and ports is important to the successful deployment of a CPE when NAT is involved. Most Internet connections are TCP-based and NAT devices typically handle TCP-based NAT very well.

Firewall - If there is a firewall in the path between the CPE and vMS infrastructure, it must allow IP protocols 50/51, UDP ports 500/4500, and TCP port 443 to pass through it and not block them.

Day 1 Configuration Requirements

The Day 1 Startup Configuration loaded on the CPE device will include the following items before the CPE is shipped to the end customer.

HTTPS - PnP protocol uses HTTPS on TCP port 443 to contact the PnP server that runs on NSO. The CPE must have an IP address in order to originate this HTTPS request. This is the first point of contact between the CPE and PnP server on NSO.

DNS - The CPE requires DNS lookup to work for certificate validation. If DNS is not working, the CPE will not be able to establish a PnP session because it will reject the certificate.

NTP - The CPE requires NTP to be working and have its internal clock be relatively close to actual time to ensure certificate validation. If NTP is not working and the CPE clock is not accurate, it might think the certificate is not valid. Most certificates are valid only during a certain date range.

Certificate - The CPE needs to have a certificate in its configuration to properly establish HTTPS session with the PnP server.

7.4 Preparation and Verification

All CPEs delivered to a customer must have the Day -1 config as the startup-config, and the same configuration should also be stored in the device flash storage as `day--1-config` file. If the Day -1 configuration is missing, then the CPE will never attempt to connect to the PnP server until the configuration is loaded. The same configuration is stored as `flash:day--1-config` so that it can be reset back to "factory defaults" once it has been de-commissioned from the vMS service. If the `flash:day--1-config` is not present, then the CPE will fail to reconnect to the PnP server after being reset or removed from a service chain. Each CPE type needs a `day--1-config` file that is specific to the CPE type and deployment. It should be tested in a lab before the configurations are finalized and placed on multiple CPEs.

To configure a CPE requires a console connection. Once connected to the CPE console, enter enable mode, and then config mode, and then paste the day--1-config into the terminal session.

```
router>enable
router#config t
router-config# < now paste the day--1-config  into the terminal >
router-config#end

router#copy running-config startup-config
router#copy running-config flash:day--1-conf
```

Here is a sample configuration for `day--1-config`:

```
!!!
!!! Common for all CPE models
!!!
!
hostname router
!
ip domain name <a local domain name like cisco.com>t
ip domain lookup
!
aaa new-model
!
aaa authentication login default none
!
crypto key generate rsa general-keys modulus 2048
!
ip host <pnp servername from certificate> <pnp server IP address>
ip name-server <public IP address of DNS server like 8.8.8.8>
ip name-server <public IP address of secondary DNS server like 8.8.4.4
!
ntp server <public IP address or name of NTP server 1 like 0.pool.ntp.org>
ntp server <public IP address or name of NTP server 2 like 1.pool.ntp.org>
!
crypto pki trustpoint <Deployment specific CA root name>
 enrollment mode ra
 enrollment terminal
 usage ssl-client
 revocation-check crl
!
!
crypto pki certificate chain <Deployment specific CA root name>
 certificate ca 26
< CERTIFICATE TEXT GOES IN HERE >
  quit
!
pnp profile zero-touch
 transport https ipv4 <IP Address of PnP server> port 443 remotecert <Deployment specific CA root name>
!
!
!
!!!
!!! interfaces varies between CPE models
!! the interface in this config should be the WAN interface
!! as shown in show pnp interface-map on NSO
!! There are CPE model specific variations
!! There are deployment specific variations
!!!
!
!! C881 / 881G2
interface FastEthernet4
  no shutdown
  ip address dhcp
!
!
!! C1900/C2900/C3900
interface GigabitEthernet0/1
  no shutdown
  ip address dhcp
!
!
!! C892
interface GigabitEthernet8
```

```
  no shutdown
  ip address dhcp
!
!
end
```

The configuration can be checked with the following commands. Both commands should have the same output, and should display all of the configuration needed for PnP to work.

```
more flash:day--1-config
more nvram:startup-config
```

The serial number of the CPE can be displayed by executing the following command. The serial number of the CPE should be saved so that it can be referenced later in case of difficulty. Notice that the `show version` command also displays the IOS feature set that is enabled.

For new combinations of CPE device types and Day -1 configurations, it is recommended to verify that the IOS version, feature set, and configuration works before deploying to the field. It is suggested to test it by the normal CPE on-boarding process into a test service chain. This can prevent troubleshooting sessions at an end customer location.

7.5 On-boarding Call Flow

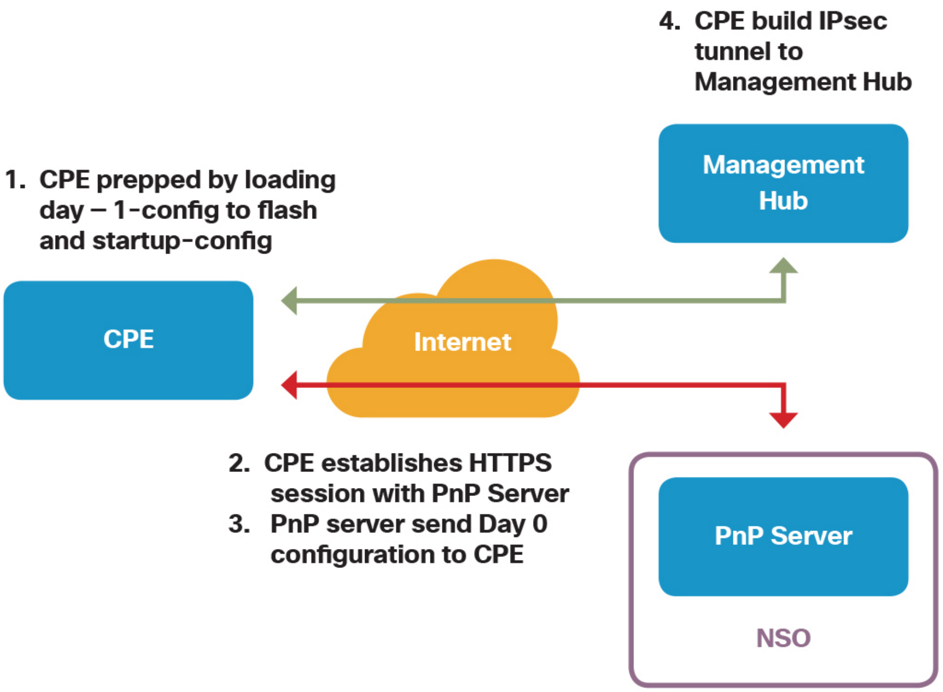
CPE on-boarding is the process of adding a CPE into an existing service chain. This can be done when the service chain is originally created, but often times, the CPE serial number is not known when the service chain is created. The CPE serial number is a required entry for on-boarding it into the vMS CloudVPN service. A geographical address can also be entered, for easier graphical display of CPE on a map, but is not required for the CPE to work. The CPE can be on-boarded in the portal either before or after it is connected to the Internet. Once a CPE serial number has been associated with a service chain from the portal, and CPE establishes connectivity to the NSO, the CPE will get provisioned by the NSO.

The CPE on-boarding procedure in the Portal is documented in Chapter 4.3 "Service Provider Operations" earlier in this document. An explanation of how to on-board a CPE into a service chain can also be found in the "vMS Portal Administration and Operations Guide" www.cisco.com/c/en/us/td/docs/net_mgmt/vMS/admin_operations_guide/vMS_portal_2_0_admin_operation_guide.html. The CPE related configuration options exposed in the Portal are "**Local LAN IP Subnet**" and "**WAN Bandwidth Upload and Download**" settings.

CPE On-boarding Call Flow

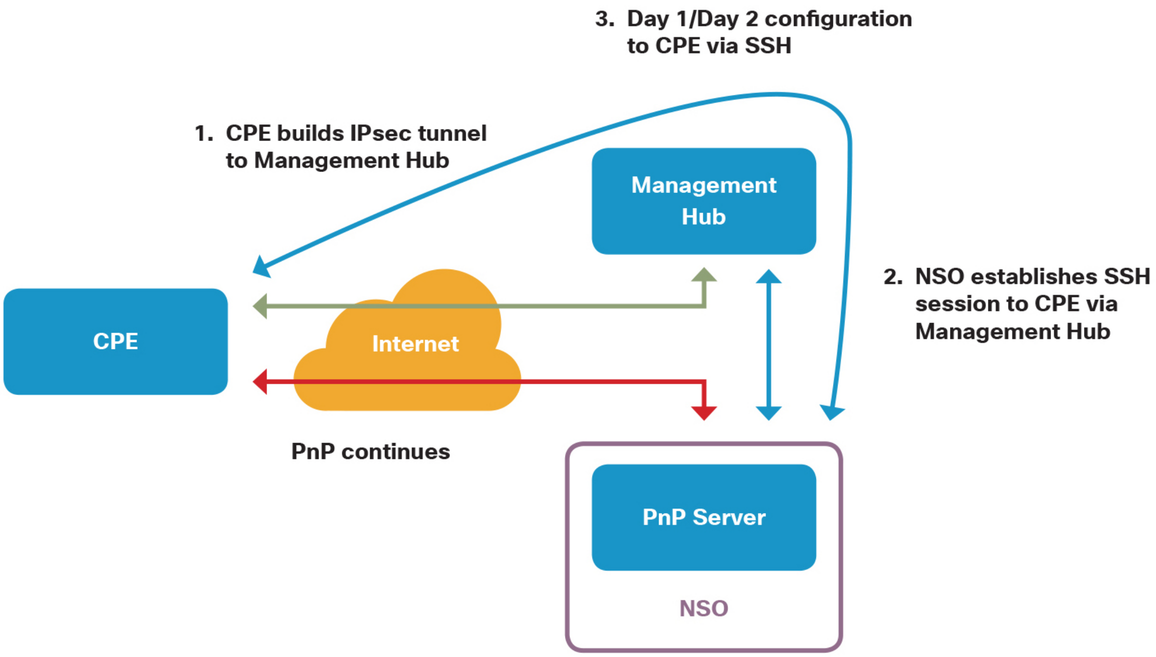
When the end customer receives the physical CPE and connects it to the Internet, the CPE goes through the on-boarding process, to become an active component of the customer's service chain. The first diagram shows the PnP flow of a CPE from boot to obtaining a Day 0 configuration.

Figure 7.2 CPE Day 0 Provisioning


4. CPE build IPsec tunnel to Management Hub
Management Hub
1. CPE prepped by loading day – 1-config to flash and startup-config
CPE
Internet
2. CPE establishes HTTPS session with PnP Server
3. PnP server send Day 0 configuration to CPE
PnP Server
NSO

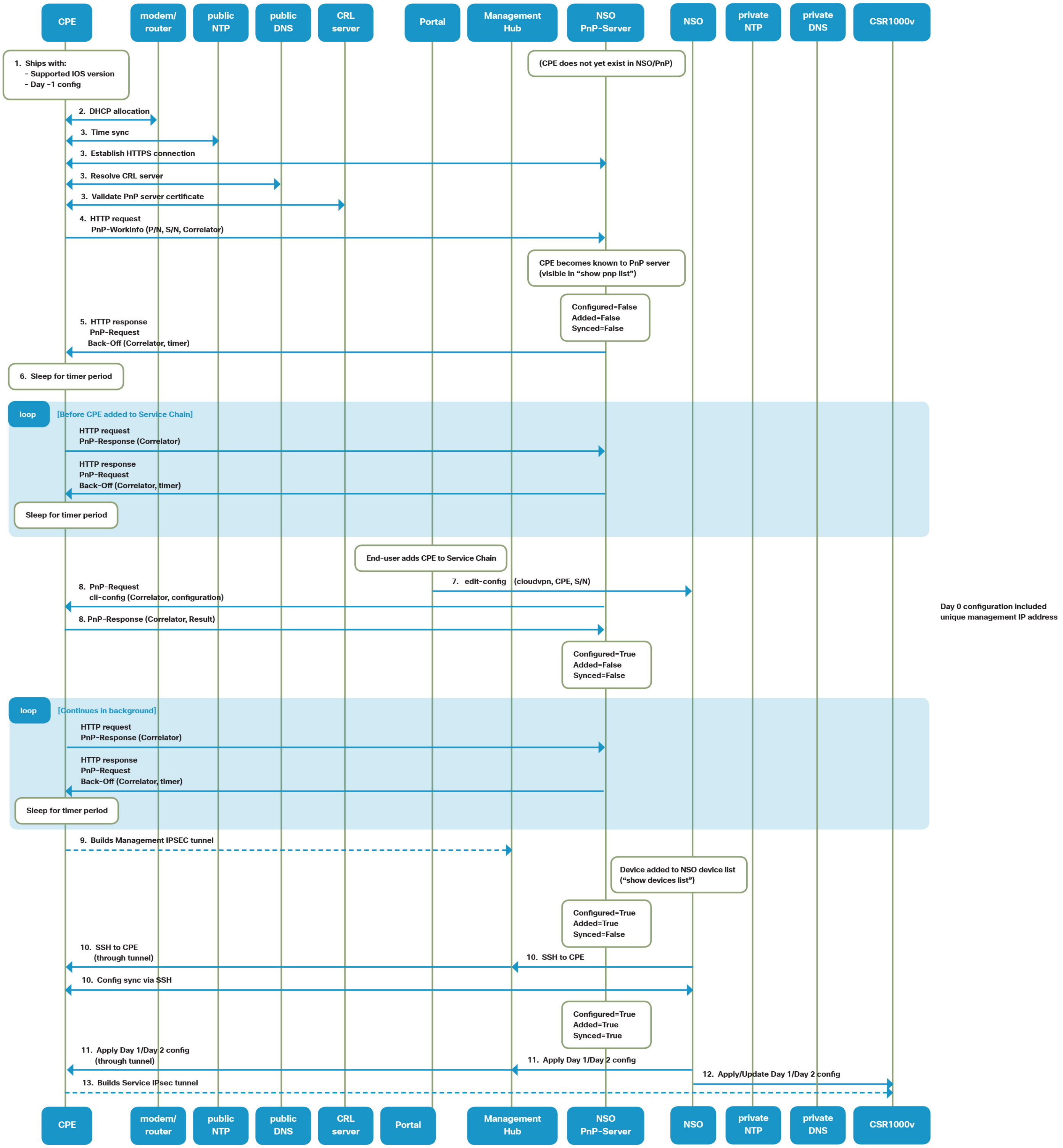
This next diagram shows the flow of the CPE on-boarding process from Day 0 to automated deployment of the Day 1/Day 2 configuration.

Figure 7.3 CPE Day 1/Day 2 Provisioning


3. Day 1/Day 2 configuration to CPE via SSH
1. CPE builds IPsec tunnel to Management Hub
Management Hub
2. NSO establishes SSH session to CPE via Management Hub
CPE
Internet
PnP continues
PnP Server
NSO

This section describes in detail the call flow to on-board a new CPE into the vMS CloudVPN service. A detailed sequence diagram of a CPE on-boarding call flow is shown in Figure 7.4

Figure 7.4 CPE On-boarding Call Flow


CPE
modem/ router
public NTP
public DNS
CRL server
Portal
Management Hub
NSO PnP-Server
NSO
private NTP
private DNS
CSR1000v
1. Ships with:
- Supported IOS version
- Day -1 config
(CPE does not yet exist in NSO/PnP)
2. DHCP allocation
3. Time sync
3. Establish HTTPS connection
3. Resolve CRL server
3. Validate PnP server certificate
4. HTTP request
PnP-Workinfo (P/N, S/N, Correlator)
CPE becomes known to PnP server
(visible in "show pnp list")
Configured=False
Added=False
Synced=False
5. HTTP response
PnP-Request
Back-Off (Correlator, timer)
6. Sleep for timer period
loop
[Before CPE added to Service Chain]
HTTP request
PnP-Response (Correlator)
HTTP response
PnP-Request
Back-Off (Correlator, timer)
Sleep for timer period
End-user adds CPE to Service Chain
7. edit-config (cloudvpn, CPE, S/N)
8. PnP-Request
cli-config (Correlator, configuration)
Day 0 configuration included
unique management IP address
8. PnP-Response (Correlator, Result)
Configured=True
Added=False
Synced=False
loop
[Continues in background]
HTTP request
PnP-Response (Correlator)
HTTP response
PnP-Request
Back-Off (Correlator, timer)
Sleep for timer period
9. Builds Management IPSEC tunnel
Device added to NSO device list
("show devices list")
Configured=True
Added=True
Synced=False
10. SSH to CPE
(through tunnel)
10. SSH to CPE
10. Config sync via SSH
Configured=True
Added=True
Synced=True
11. Apply Day 1/Day 2 config
(through tunnel)
11. Apply Day 1/Day 2 config
12. Apply/Update Day 1/Day 2 config
13. Builds Service IPsec tunnel
CPE
modem/ router
public NTP
public DNS
CRL server
Portal
Management Hub
NSO PnP-Server
NSO
private NTP
private DNS
CSR1000v

CPE On-boarding Flow Description

A detailed description of the flow, including verification steps, is below:

1. The end customer receives a CPE that has the correct and supported Cisco IOS version, as well as appropriate Day -1 configuration (named `day--1-config` on the CPE's flash device), from their Service Provider.

At this initial stage, the CPE is not yet known to the PnP server. NSO `show pnp list or show pnp-state` CLI will not list the CPE serial number in the output.

2. Once the CPE's WAN port is connected to the Internet, it will receive an IP address via DHCP. If connected to the CPE via serial console, this can be verified using the `show ip interface brief` command.

3. The CPE will obtain current date and time from the configured NTP server, and will attempt to establish a secure connection to the PnP server using HTTPS protocol. Depending on the specific configuration and certificate being used, these steps may involve:

- resolving the PnP server name to an IP address using DNS,
- verifying that the PnP server certificate is trusted by the CA-Certificate that is included in the CPE's Day -1 configuration,
- resolving a Certificate Revocation List (CRL) server using DNS, downloading a CRL, and validating the PnP server certificate against the CRL
- verifying that the current date and time is within the PnP server certificate's valid time

The `show pnp profile` command can be used to verify that the CPE is connected to the PnP server, and it should show a https connection in "Established" state:

```
  router#show pnp profile&#10;
  Initiator Profile zero-touch: 1 open connections: 0 closing connections
    Encap: pnp
     WSSE header is not required. Configured authorization level is 1
   Work-Request Tracking: Validation Yes, Violation 0, UDI
[PID:CISCO1941/K9,VID:,SN:FTX00000000], Correlator [CiscoPnP-1.0-1111-22222222]
    PnP Response Pending: Retry Allowed 0, UDI [none], Correlator [none]
   Countdown: Security Unlock 0, Service Lock: 72, Service Req Wait: 0, Prxoy Req Wait 0,
Service Resp Ack: 0
     Max message (RX) is 50 Kbytes
      XEP Faults are sent
    Idle timeout infinite
      Keepalive not configured
   Reconnect time 60 seconds
      Primary transport: https to host 192.168.192.168, port 443, URL pnp/WORK-REQUEST
    Connected to the primary transport via https

    Remote connection via HTTP client.  URL https://192.168.192.168:443/pnp/WORK-REQUEST, post
    Established at 13:06:12.643 CET Wed Nov 25 2015
    Tx 2753052 bytes (11086 msg), Tx 0 errors,
     Last message sent at 01:14:01.156 CET Wed Dec 9 2015
       Rx 3332284 bytes (11086 msg), 0 empty msg
      Last message received at 01:14:01.532 CET Wed Dec 9 2015
```

4. Once the underlying HTTPS session is up, CPE will contact the PnP server with an HTTP request PnP-Workinfo message, including:

- the CPE's UDI (which, in turn, contains the Serial Number and Part ID; "PID:CISCO1941/K9,VID:,SN:FTX00000000" in the example above)
- A correlator, which is a quasi-random identifier that will change periodically and aid in matching requests and responses.

This is the first sign of life of the CPE being visible to the NSO PnP server. At this stage:

- NSO `show pnp list` CLI will include the CPE's serial number
  - The "CONFIGURED", "ADDED" and "SYNCED" state columns will all show "false", as indicated by the diagram above
- NSO `show pnp-state device <serial number>` will be aware of the CPE, and will show more detailed information about the CPE's state.
- If PnP logging is configured, there will be a PnP trace log (with a file name derived from the CPE's UDI, e.g. for PID "PID:CISCO1941/K9,VID:,SN:FTX00000000" the log file name would be PID_CISCO1941_K9_VID__SN_FTX00000000.trace) created in the configured log directory.

5. NSO PnP Server responds by sending back HTTP response, PnP-Request message with correlator and a backoff timer to CPE, requesting the CPE to back off and re-establish contact to the PnP server after the specified period. The backoff timer is configurable but defaults to 3 minutes and 30 seconds.

6. The CPE's PnP client will "sleep" for the requested period of time, and then re-establish contact with the PnP server by sending an HTTP request PnP-Response message. Unless the CPE has been added to a service chain in the meantime, the PnP server will again instruct the CPE to back off for a given period of time.

> **Note:** This cycle of CPE/PnP client request, response and " sleep" will continue indefinitely for as long as the CPE is connected to the internet. It will also continue after the CPE has received a configuration and/or has become part of a service chain.

> **Note:** Instead of PnP-Request and PnP-Response messages, occasionally PnP-Workinfo messages are seen being exchanged in either direction. For the purpose of this document, the specific message type (PnP Request, Response or Workinfo) can be considered as irrelevant; it is merely required that there is **any** exchange of messages after every backoff interval. A CPE can be considered alive and responsive to PnP requests as long as NSO `show pnp list` or `show pnp-state` CLI shows a " LAST CONTACT" timestamp that is less than <backoff-timer> ago, and the trace log contains no errors.

Subsequent steps happen **after the CPE has been added to a service chain via the portal**:

7. vMS Portal will communicate to NSO by sending **edit-config** message with the following parameters:

- **cloudvpn** - cloudvpn name
- **CPE** - CPE-identifier
- **S/N** - Serial Number of CPE

The service chain configuration (as seen in `show configuration cloudvpn <service-chain-name>` in NSO CLI) now contains the CPE serial number:

```
admin@ncs> show configuration cloudvpn <service-chain-name> cpe
cpe cpe_1 {
    serial FTX00000000;
    allocate {
        ip-type     ipv4;
        prefix-size 24;
    }
}
```

The service chain's device list will also include the CPE, but in non-existing state in the "READY" column:

```
admin@ncs-vm> show cloudvpn-data <service-chain-name> device-list
DEVICE                 REMOTE  READY           PROVISIONED
------------------------------------------------------------
...
cpe-FTX00000000        -       not-existing    false

[ok][2015-12-09 15:50:08]
```

8. After the next backoff-timer interval has expired and the CPE re-establishes contact with the PnP server, PnP server will send a Day 0 configuration to the CPE using the PnP protocol.

This Day 0 configuration includes:

- A Management VRF
- A Management IPsec tunnel which will terminate on the Management IPsec Hub
- A unique Management IPv4 or IPv6 address (determined by NSO setting) assigned by NSO
- SSH server configuration that allows inbound SSH access to the CPE via the Management IPsec tunnel only
- (Potentially different) DNS and NTP servers, reachable through the Management IPsec tunnel and potentially replacing those from the Day -1 configuration
- Access-lists to secure the WAN and management interfaces

The Day 0 configuration will be sent to the CPE using PnP protocol in a "cli-config" message, in response to the PnP Request message that the CPE will send upon backoff timer expiry. The CPE will respond again reporting the outcome of each command executed, and then the periodic backoff-timer request/response/sleep cycle will continue.

The complete Day 0 configuration that is being sent to the CPE, as well as the CPE's response to each command execution, can be seen in the PnP trace log (examples abbreviated):

```
  To device 2015-11-19 18:43:31
  -------------------------------------------
  [{pnp,
       [{xmlns,"urn:cisco:pnp"},
        {version,"1.0"},
        {udi,"PID:CISCO3945-CHASSIS,VID:,SN:FTX00000000"},
        {usr,"admin"},
        {pwd,"cisco123"}],
       [{request,
            [{correlator,"CiscoPnP-1.0-891-2F53DD0"},
             {version,"1.0"},
             {xmlns,"urn:cisco:pnp:cli-config"}],
            [{configApply,
                 [{details,"all"}],
                 [{'config-data',[],
                      [{'cli-config-data-block',[],
                           "no ntp\nservice timestamps debug datetime msec localtime show-
timezone\nservice timestamps log datetime msec localtime show-timezone...
    ...


  From device 2015-11-19 18:43:32
  -------------------------------------------
  [{pnp,
       [{xmlns,"urn:cisco:pnp"},
        {version,"1.0"},
        {udi,"PID:CISCO3945-CHASSIS,VID:,SN:FTX00000000"}],
       [{response,
            [{xmlns,"urn:cisco:pnp:cli-config"},
             {correlator,"CiscoPnP-1.0-891-2F53DD0"},
             {success,"1"}],
            [{resultentry,
                 [{linenumber,"1"},{clistring,"no ntp"}],
                 [{success,[],[]}]},
             {resultentry,
                 [{linenumber,"2"},
                  {clistring,
                      "service timestamps debug datetime msec localtime show-timezone"}],
                 [{success,[],[]}]},
             {resultentry,
                 [{linenumber,"3"},
                  {clistring,
                      "service timestamps log datetime msec localtime show-timezone"}],
                 [{success,[],[]}]},
  ...
```

At this stage, CPE has its dedicated management IP address.

On NSO PnP server, `show pnp list` and `show pnp-state` commands will now show "true" for the CPE in question under the "CONFIGURED" column:

```
admin@ncs> show pnp list
SERIAL        IP ADDRESS       CONFIGURED  ADDED  SYNCED  LAST CONTACT
-------------------------------------------------------------------------------
FTX00000000   192.168.192.168  true        false  false   2015-12-09 02:51:52
```

Also, at this stage, `show pnp-state device <serial number>` will show additional data that has been constructed or assigned or the CPE, including:

- the management IP address (allocated based on the configured pool of Management IP addresses)
- the administrator password (typically randomly created, except in sandboxed testing environments where a static password may be used)
- the LAN and WAN interface (selected by matching the device UDI against the configured PnP interface maps).

```
admin@f-ncsf-2a-de> show pnp-state device FTX00000000
pnp-state device FTX00000000
 udi           PID:CISCO3945-CHASSIS,VID:,SN:FTX00000000
 device-info   "Cisco IOS Software, C3900 Software (C3900-UNIVERSALK9-M), Version 15.5(1)T,
RELEASE SOFTWARE (fc2)\nTechnical Support: http://www.cisco.com/techsupport\nCopyright (c)
1986-2014 by Cisco Systems, Inc.\nCompiled Wed 19-Nov-14 05:37 by prod_rel_team"
   ip-address    192.168.192.168
   mgmt-ip       fc00:1:2:3:4:5:6:7
   port          22
   name          cpe-FTX00000000
   username      admin
   password      random-password-here
   sec-password  random-password-here
   wan-interface GigabitEthernet0/1
   lan-interface GigabitEthernet0/0
   configured    true
   request       backoff
   added         true
   synced        true
   is-netsim     false
   need-clean    false
   pending-exec  ""
   last-contact  2015-12-09 15:23:29
   last-clean    0
  [ok][2015-12-09 15:26:23]
```

9. Having received the Day 0 config, the CPE will establish the newly configured Management IPsec tunnel to the Management Hub.

In parallel, NSO will create a device entry for the CPE.

Once the device entry is created:

- The CPE would appear in `show devices list`
- `show cloudvpn-data <service-chain-name> device-list`, "READY" column, will show the CPE in not-yet-ready status
- `show pnp list` and `show pnp-state device <serial number>` will show the CPE in "ADDED"=true status
- NSO will attempt to establish contact with the CPE using SSH (for configuration) and SNMP (for statistics polling).

Note: At this stage, NSO's attempt to establish contact with the CPE may or may not be successful, depending on whether or not the CPE is successful in establishing the Management IPsec tunnel. If the CPE is unable to establish the Management IPsec tunnel for any reason, then CPE onboarding will be unable to progress beyond this point. Refer to Chapter 7.8 " CPE Troubleshooting" for additional details on how to troubleshoot the Management IPsec tunnel.

Subsequent steps happen **after the CPE has established the Management IPsec tunnel and has IP reachability from NSO through the tunnel**:

10. NSO successfully establishes a connection to the CPE and synchronises the configuration between CPE and NSO's configuration database.

At this stage:

- `show cloudvpn-data <service-chain-name> device-list`, "READY" column, will show the CPE in ready status
- `show pnp list` and `show pnp-state device <serial number>` will show the CPE in "SYNCED"=true status

11. NSO will generate the Day 1 and Day 2 configuration appropriate for the CPE (including the LAN interface configuration and the Service IPsec tunnel to the service chain's CSR1000v), and apply that configuration to the CPE.

Once the Day 1 and Day 2 configuration has been successfully applied, the "PROVISIONED" value for the CPE as seen in `show cloudvpn-data device-list` will change to "true".

Details of the Day 1/Day 2 configuration being provisioned to the CPE, can be seen in the CPE's NED trace.

12. If this is the **first** CPE being added to the service chain, then the service chain CSR1000v will be provisioned with its Day 1/Day 2 configuration at the same time, and the "PROVISIONED" value for the CSR1000v (as seen in `show cloudvpn-data device-list`) will change from "false" to "true". If any other CPEs are already provisioned for the same service chain, then the CSR1000v's configuration will merely be updated to accept this additional CPE.

Details of the Day 1/Day 2 configuration or configuration update being provisioned to the CSR1000v, can be seen in the CSR1000v's NED trace.

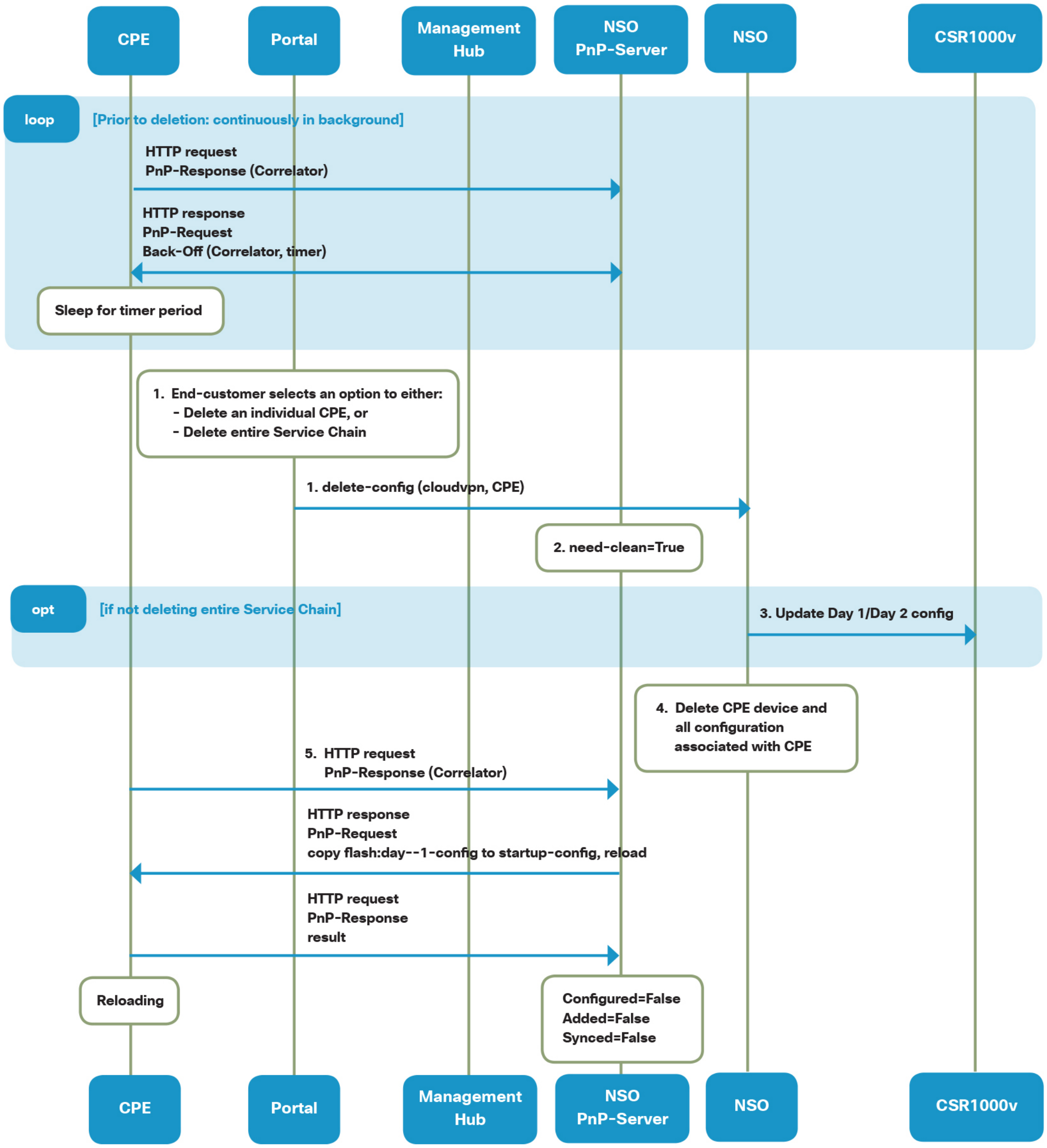
13. Having received the Day 1/Day 2 config, the CPE will initiate the Service IPsec tunnel to the CSR1000v.

7.6 Delete Call Flow

End customers sometimes have a need to remove a CPE device from their service chain. This could be because the CPE needs to be removed from the customer's remote location, or because the customer will not be using their service anymore.

The process of removing the CPE from the service chain is referred to as the "CPE off-boarding process", and will be described in detail in this section. A detailed sequence diagram of a CPE off-boarding call flow is shown in Figure 7.5

Figure 7.5 CPE Off-boarding Call Flow


CPE
Portal
Management Hub
NSO PnP-Server
NSO
CSR1000v
loop
[Prior to deletion: continuously in background]
HTTP request
PnP-Response (Correlator)
HTTP response
PnP-Request
Back-Off (Correlator, timer)
Sleep for timer period
1. End-customer selects an option to either:
- Delete an individual CPE, or
- Delete entire Service Chain
1. delete-config (cloudvpn, CPE)
2. need-clean=True
opt
[if not deleting entire Service Chain]
3. Update Day 1/Day 2 config
4. Delete CPE device and all configuration associated with CPE
5. HTTP request
PnP-Response (Correlator)
HTTP response
PnP-Request
copy flash:day--1-config to startup-config, reload
HTTP request
PnP-Response
result
Reloading
Configured=False
Added=False
Synced=False
CPE
Portal
Management Hub
NSO PnP-Server
NSO
CSR1000v

The process of off-boarding a CPE consists of the NSO sending configuration commands to the CPE to install Day -1 configuration, and remove CPE from the existing service chain.

1. The end customer marks CPE for deletion using vMS Portal.

The Portal communicates with NSO and sends NETCONF **delete-config** message with parameters: cloudvpn and CPE that needs to be deleted.

2. NSO marks CPE for the deletion by setting a PnP CPE state flag, "need-clean", to True. This can be seen in NSO `show pnp-state device` CLI:

```
  admin@ncs> show pnp-state device FTX00000000
  pnp-state device FTX00000000
   udi           PID:CISCO1941/K9,VID:,SN:FTX00000000

   device-info   "Cisco IOS Software, C1900 Software (C1900-UNIVERSALK9-M), Version 15.5(1)T,
RELEASE SOFTWARE (fc2)\nTechnical Support: http://www.cisco.com/techsupport\nCopyright (c)
1986-2014 by Cisco Systems, Inc.\nCompiled Wed 19-Nov-14 00:44 by prod_rel_team"
  ...
   need-clean    true
  ...
```

3. Unless the entire service chain is being deleted, NSO will update the service chain's CSR1000v configuration to remove the service tunnel associated with this CPE.

4. NSO deletes the CPE from the service chain and from the device database, and deletes any other configuration items associated with the CPE in NSO configuration. The CPE will no longer appear ina show devices list or similar commands, or anywhere in NSO `show configuration`. It will, however, continue to be known to the NSO PnP server component, and will continue to appear in `show pnp list` and `show pnp-state` CLI.

5. Next time the NSO PnP server receives an HTTP Request from this CPE, PnP will:

- Instruct the CPE to copy `flash:day--1-config` to the startup-config, and
- Instruct the CPE to reload.

This will reset the CPE to the same state in which it was initially shipped to the end customer.

After the CPE has been instructed to reload, NSO PnP server will reset its state associated with the CPE. `Show pnp list` and `show pnp-state` will show all flags (Configured, Added, Synced, Need-Clean) as "false" and passwords removed. Some of the CPE information (e.g. Management IP address, Device-Info) is retained so that if the CPE were to be re-connected in the future, the same address would be used.

> **Note 1:** Assuming that the CPE is continuously connected to the public Internet, then the CPE's PnP request/response/backoff cycle will have continued in the background, and step 5 will be executed whenever the next backoff timer has expired. If the CPE was offline at the time it was being deleted, then step 5 will be executed whenever the CPE is re-connected to the public Internet.

> **Note 2**: After the CPE completes rebooting and comes up with a clean copy of the Day -1 configuration, NSO/PnP will again learn of its existence and treat it as a new CPE that may be on-boarded at a later time.

7.7 Replacement Procedure

If a CPE needs to be replaced following a hardware failure, off-board the old CPE and onboard a new CPE. Off-boarding an old CPE and on-boarding a new CPE is covered in the previous two sections.

7.8 Troubleshooting

Obtaining CPE Password

If a CPE is in True/True/True state, then it should be possible to SSH from the NSO to the CPE. Required information (CPE Management IP Address, username, password) can be obtained from NSO by executing the `show pnp-state device <cpe_serial_number>` command as shown below.

```
admin@vms-ncs-sm> show pnp-state device XXX194326WW
pnp-state device XXX194326WW
 udi            PID:C881-K9,VID:V01,SN:XXX194326WW
 device-info    15.5(3)M1
 ip-address     11.156.141.167
 mgmt-ip        10.254.0.29
 port           22
 name           cpe-XXX194326WW
 username       admin
 password       cpe_password
 sec-password   cpe_password
 salt           ABCD
 remote-node    vms-ncs-dm
 wan-interface  FastEthernet4
 lan-interface  FastEthernet0
 configured     true
 request        backoff
 added          true
 synced         true
 is-netsim      false
 need-clean     false
 pending-exec   ""
 last-contact   2015-12-09 01:53:33
 last-clean     0
[ok][2015-12-09 01:54:14]
```

From NSO, establish an SSH session to the CPE.

```
admin@vms-ncs-sm> ssh 10.254.0.29
Password:cpe_password

router line 11

router#
```

Plug-and-Play (PnP)

There are several commands that can provide the state of the PnP server and individual CPEs.

The `show pnp list` is a frequently used command. It has several nice details that can be used to determine the state of the system and the CPE. This list shows three CPEs that are in various states. The first one is not configured, added, or synced and most importantly, notice the "LAST CONTACT" column. In this case, the last two CPEs have talked within the last 210 seconds (backoff-timer), while the first CPE has not contacted the PnP server for over 24 hours. PnP has a configurable back-off timer to tell the CPE how long to wait before the next time it contacts the PnP server. Each time a CPE contacts the PnP server, it asks for updates (WORK-REQUEST) and if there are updates or configuration changes to be done, the PnP server sends them to the CPE via the PnP protocol.

```
admin@ncs-vm> show pnp list
SERIAL        IP ADDRESS     CONFIGURED  ADDED  SYNCED  LAST CONTACT
-----------------------------------------------------------------------------
9CANYW0SMDU   192.168.11.13  false       false  false   2015-12-07 22:27:42
9N989LLRWSN   192.168.11.14  true        true   true    2015-12-09 02:02:26
9YLMGXIQ0ON   192.168.11.15  true        true   true    2015-12-09 02:02:47

[ok][2015-12-09 02:05:07]
admin@ncs-vm>
```

PnP/PID traces are available at `/var/log/ncs/` by default or the configured log directory. There is one trace file per CPE (serial number). PnP trace files have the naming format of `PID*<SN>.trace`. It contains all of the interactions between the PnP server and the PnP agent on the CPE device. The most common thing that happens in the `PID*trace` files is for a CPE to contact the PnP server and ask for work, if the PnP server does not have any new configuration changes for the CPE, it tells the CPE to backoff for a period of time. This is a normal workflow for the CPE to ask PnP server for work and it goes in a continuous timed loop.

Troubleshooting PnP HTTP session establishment

If a PnP session between the CPE and the PnP server is not getting established, troubleshooting requires a serial console connection. The following steps can be used to narrow down the root cause:

- `show running-config`: Verify PnP configuration is present and correct. Also verify startup-config and `flash:day--1-config` are correct.
- `show ip interface brief`: Verify the CPE WAN interface obtained an IP address and default gateway and is "UP".
- Verify that the default gateway is pointing to the WAN interface and not the LAN interface.
- Attempt to ping the PnP server, and/or other known reachable devices on the Internet.
- Attempt to ping a host by name, to verify DNS resolution
- `show clock`, `show ntp status` to verify NTP time synchronization

- `show pnp profile`: Verify the PnP client status
- `debug pnp all`, `debug xmpp profile` to debug PnP transactions
- `debug ip http ssl error`, `debug ip http client`, `debug ip http error`, `debug ip http transactions`, and `debug ip http url` (or `debug ip http all`), and `debug crypto pki validatio`n, to debug any errors in HTTP session establishment or certificate validation

PID trace example of work-request and backoff with timer

```
  From device 2015-12-09 14:04:24
  -------------------------------------------
  [{pnp,
       [{xmlns,"urn:cisco:pnp"},
        {version,"1.0"},
        {udi,"PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"}],
       [{info,
            [{xmlns,"urn:cisco:pnp:work-info"},
             {correlator,"CiscoPnP-1.0-416-582-7F987E6371F0-416"}],
            [{deviceid,[],
                 [{udi,[],["PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"]},
                  {hostname,[],["router"]},
                  {authrequired,[],["true"]},
                  {viaproxy,[],["false"]},
                  {securityadvise,[],
                      ["Password in clear text in unsecured transport"]}]}]}]}]
  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"><info
xmlns="urn:cisco:pnp:work-info" correlator="CiscoPnP-1.0-416-582-7F987E6371F0-416"><deviceId>
<udi>PID:ISR880,VID:V00,SN:9YLMGXIQ0ON</udi><hostname>router</hostname>
<authRequired>true</authRequired><viaProxy>false</viaProxy><securityAdvise>Password in clear
text in unsecured transport</securityAdvise></deviceId></info></pnp>
    To device 2015-12-09 14:04:24
  -------------------------------------------
  [{pnp,
       [{xmlns,"urn:cisco:pnp"},
        {version,"1.0"},
        {udi,"PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"},
        {usr,"admin"},
        {pwd,"cisco123"}],
       [{request,
            [{correlator,"CiscoPnP-1.0-416-582-7F987E6371F0-416"},
             {version,"1.0"},
             {xmlns,"urn:cisco:pnp:backoff"}],
            {backoff,[],
                [{reason,[],"take it easy"},
                 {callbackAfter,[],
                     [{hours,[],"0"},{minutes,[],"3"},{seconds,[],"30"}]}]}]}]}]

  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9YLMGXIQ0ON" usr="admin"
pwd="cisco123">
  <request correlator="CiscoPnP-1.0-416-582-7F987E6371F0-416" version="1.0"
xmlns="urn:cisco:pnp:backoff">
  <backoff>
  <reason>take it easy</reason>
  <callbackAfter>
  <hours>0</hours>
  <minutes>3</minutes>
  <seconds>30</seconds></callbackAfter></backoff></request></pnp>
    From device 2015-12-09 14:04:24
  -------------------------------------------
  [{pnp,[{xmlns,"urn:cisco:pnp"},
         {version,"1.0"},
         {udi,"PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"}],
        [{response,[{correlator,"CiscoPnP-1.0-416-582-7F987E6371F0-416"},
                    {success,"1"},
                    {xmlns,"urn:cisco:pnp:backoff"}],
                   []}]}]
  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"><response
correlator="CiscoPnP-1.0-416-582-7F987E6371F0-416" success="1" xmlns="urn:cisco:pnp:backoff">
</response></pnp>
    To device 2015-12-09 14:04:24
  -------------------------------------------
  [{pnp,[{xmlns,"urn:cisco:pnp"},
         {version,"1.0"},
         {udi,"PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"},
         {usr,"admin"},
         {pwd,"cisco123"}],
        [{info,[{correlator,"CiscoPnP-1.0-416-582-7F987E6371F0-416"},
                {xmlns,"urn:cisco:pnp:work-info"}],
               {workInfo,[],[{bye,[]}]}]}]}]

  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9YLMGXIQ0ON" usr="admin"
pwd="cisco123">
  <info correlator="CiscoPnP-1.0-416-582-7F987E6371F0-416" xmlns="urn:cisco:pnp:work-info">
  <workInfo>
  <bye></bye></workInfo></info></pnp>
```

Device-info PnP trace

PID trace file example of an initial contact from CPE to PnP server, and the PnP server request for "deviceinfo".

```
  From device 2015-12-08 13:59:01
  -------------------------------------------
  [{pnp,
       [{xmlns,"urn:cisco:pnp"},
        {version,"1.0"},
        {udi,"PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"}],
       [{info,
            [{xmlns,"urn:cisco:pnp:work-info"},
             {correlator,"CiscoPnP-1.0-1-17-7F987E63BF88-2"}],
            [{deviceid,[],
                 [{udi,[],["PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"]},
                  {hostname,[],["host-192-168-11-15"]},
                  {authrequired,[],["false"]},
                  {viaproxy,[],["false"]},
                  {securityadvise,[],
                      ["Password in clear text in unsecured transport"]}]}]}]}]
  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"><info
xmlns="urn:cisco:pnp:work-info" correlator="CiscoPnP-1.0-1-17-7F987E63
  BF88-2"><deviceId><udi>PID:ISR880,VID:V00,SN:9YLMGXIQ0ON</udi><hostname>host-192-168-11-
15</hostname><authRequired>false</authRequired><viaProxy>false</viaPr
  oxy><securityAdvise>Password in clear text in unsecured transport</securityAdvise></deviceId>
</info></pnp>
    To device 2015-12-08 13:59:01
  -------------------------------------------
  [{pnp,[{xmlns,"urn:cisco:pnp"},
         {version,"1.0"},
         {udi,"PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"},
         {usr,"admin"},
         {pwd,"admin"}],
        [{request,[{correlator,"CiscoPnP-1.0-1-17-7F987E63BF88-2"},
                   {version,"1.0"},
                   {xmlns,"urn:cisco:pnp:device-info"}],
                  [{deviceInfo,[{type,"all"}]}]}]}]

  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9YLMGXIQ0ON" usr="admin"
pwd="admin">
  <request correlator="CiscoPnP-1.0-1-17-7F987E63BF88-2" version="1.0"
xmlns="urn:cisco:pnp:device-info">
  <deviceInfo type="all"></deviceInfo></request></pnp>
    From device 2015-12-08 13:59:06
```

```
-------------------------------------------
[{pnp,
     [{xmlns,"urn:cisco:pnp"},
      {version,"1.0"},
      {udi,"PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"}],
     [{response,
          [{correlator,"CiscoPnP-1.0-1-17-7F987E63BF88-2"},
           {success,"1"},
           {xmlns,"urn:cisco:pnp:device-info"}],
          [{udi,[],
               [{'primary-chassis',[],
                    ["PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"]}]},
           {imageinfo,[],
               [{versionstring,[],["15.6(20150712:154917)"]},
                {imagefile,[],["bootflash:packages.conf"]},
                {imagehash,[],[]},
                {returntoromreason,[],["reload"]},
                {bootvariable,[],[]},
                {bootldrvariable,[],[]},
                {configvariable,[],[]},
                {configreg,[],["0x2102"]},
                {configregnext,[],[]}]},
           {hardwareinfo,[],
               [{hostname,[],["host-192-168-11-15"]},
                {vendor,[],["cisco"]},
                {platformname,[],["ISR880"]},
                {processortype,[],["VXE"]},
                {hwrevision,[],["VXE"]},
                {mainmemsize,[],["2105633648"]},
                {iomemsize,[],["6295128"]},
                {boardid,[],["9YLMGXIQ0ON"]},
                {boardreworkid,[],[]},
                {processorrev,[],[]},
                {midplaneversion,[],[]},
                {location,[],[]}]},
           {filesystemlist,[],[{filesystem,[],[]},{filesystem,[],[]}]},
           {profileinfo,[],
               [{profile,
                    [{'profile-name',"zero-touch"},
                     {'discovery-created',"false"}],
                    [{'primary-server',[],
                         [{protocol,[],["http"]},
                          {'server-address',[],
                              [{ipv4,[],["192.168.12.11"]},
                               {port,[],["80"]}]}]}]}]}]}]}]}]
```

```
  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"><response
correlator="CiscoPnP-1.0-1-17-7F987E63BF88-2" success="1" xmlns="u
  rn:cisco:pnp:device-info"><udi><primary-chassis>PID:ISR880,VID:V00,SN:9YLMGXIQ0ON</primary-
chassis></udi><imageInfo><versionString>15.6(20150712:154917)</ver
  sionString><imageFile>bootflash:packages.conf</imageFile><imageHash />
<returnToRomReason>reload</returnToRomReason><bootVariable /><bootLdrVariable /><configVa
  riable /><configReg>0x2102</configReg><configRegNext /></imageInfo><hardwareInfo>
<hostname>host-192-168-11-15</hostname><vendor>cisco</vendor><platformName>CSR
  1000V</platformName><processorType>VXE</processorType><hwRevision>VXE</hwRevision>
<mainMemSize>2105633648</mainMemSize><ioMemSize>6295128</ioMemSize><boardId>9
  YLMGXIQ0ON</boardId><boardReworkId /><processorRev /><midplaneVersion /><location />
</hardwareInfo><fileSystemList><fileSystem name="bootflash" type="disk" siz
  e="7835619328" freespace="6533558272" readable="true" writable="true" /><fileSystem
name="nvram" type="nvram" size="33554432" freespace="33535742" readable="tr
  ue" writable="true" /></fileSystemList><profileInfo><profile profile-name="zero-touch"
discovery-created="false"><primary-server><protocol>http</protocol><serv
  er-address><ipv4>192.168.12.11</ipv4><port>80</port></server-address></primary-server>
</profile></profileInfo></response></pnp>
    To device 2015-12-08 13:59:06
  -------------------------------------------
  [{pnp,[{xmlns,"urn:cisco:pnp"},
         {version,"1.0"},
         {udi,"PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"},
         {usr,"admin"},
         {pwd,"admin"}],
        [{info,[{correlator,"CiscoPnP-1.0-1-17-7F987E63BF88-2"},
                {xmlns,"urn:cisco:pnp:work-info"}],
               {workInfo,[],[{bye,[]}]}}]}]

  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9YLMGXIQ0ON" usr="admin"
pwd="admin">
  <info correlator="CiscoPnP-1.0-1-17-7F987E63BF88-2" xmlns="urn:cisco:pnp:work-info">
  <workInfo>
  <bye></bye></workInfo></info></pnp>
    From device 2015-12-08 13:59:11
  -------------------------------------------
  [{pnp,
       [{xmlns,"urn:cisco:pnp"},
        {version,"1.0"},
        {udi,"PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"}],
       [{info,
            [{xmlns,"urn:cisco:pnp:work-info"},
             {correlator,"CiscoPnP-1.0-2-582-7F987E63D1D0-3"}],
            [{deviceid,[],
                 [{udi,[],["PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"]},
                  {hostname,[],["host-192-168-11-15"]},
                  {authrequired,[],["false"]},
                  {viaproxy,[],["false"]},
                  {securityadvise,[],
                      ["Password in clear text in unsecured transport"]}]}]}]}]
  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9YLMGXIQ0ON"><info
xmlns="urn:cisco:pnp:work-info" correlator="CiscoPnP-1.0-2-582-7F987E6
  3D1D0-3"><deviceId><udi>PID:ISR880,VID:V00,SN:9YLMGXI</udi><hostname>host-192-168-11-
15</hostname><authRequired>false</authRequired><viaProxy>false</viaP
  roxy><securityAdvise>Password in clear text in unsecured transport</securityAdvise>
</deviceId></info></pnp>
```

Day 0 configuration example

Below is a PID trace example with the PnP server sending a configuration to CPE. The important steps are that the PnP server sends the configuration, the CPE parses the configuration and sends the configuration back to the PnP server to ensure it matches what the PnP server sent.

```
  From device 2015-12-08 23:24:08
  -------------------------------------------
  [{pnp,
       [{xmlns,"urn:cisco:pnp"},
        {version,"1.0"},
        {udi,"PID:CSR1000V,VID:V00,SN:9YLMGXIQ0ON"}],
       [{info,
            [{xmlns,"urn:cisco:pnp:work-info"},
             {correlator,"CiscoPnP-1.0-164-582-7F987E641C28-164"}],
            [{deviceid,[],
                 [{udi,[],["PID:CSR1000V,VID:V00,SN:9YLMGXIQ0ON"]},
                  {hostname,[],["host-192-168-11-15"]},
                  {authrequired,[],["false"]},
                  {viaproxy,[],["false"]},
                  {securityadvise,[],
                      ["Password in clear text in unsecured transport"]}]}]}]}]
  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:CSR1000V,VID:V00,SN:9YLMGXIQ0ON"><info
xmlns="urn:cisco:pnp:work-info" correlator="CiscoPnP-1.0-164-582-7F987E641C28-164"><deviceId>
<udi>PID:CSR1000V,VID:V00,SN:9YLMGXIQ0ON</udi><hostname>host-192-168-11-15</hostname>
<authRequired>false</authRequired><viaProxy>false</viaProxy><securityAdvise>Password in clear
text in unsecured transport</securityAdvise></deviceId></info></pnp>
   To device 2015-12-08 23:24:09
  -------------------------------------------
  [{pnp,
       [{xmlns,"urn:cisco:pnp"},
        {version,"1.0"},
        {udi,"PID:CSR1000V,VID:V00,SN:9YLMGXIQ0ON"},
        {usr,"admin"},
        {pwd,"cisco123"}],
       [{request,
            [{correlator,"CiscoPnP-1.0-164-582-7F987E641C28-164"},
             {version,"1.0"},
             {xmlns,"urn:cisco:pnp:cli-config"}],
            [{configApply,
                 [{details,"all"}],
                 [{'config-data',[],
                      [{'cli-config-data-block',[],
                           "no ntp\nservice timestamps debug datetime msec localtime show-
timezone\n
                           [...]
                           snmp-server community cisco RO SNMP-ACL"}]}]}]}]}]

  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:CSR1000V,VID:V00,SN:9YLMGXIQ0ON"
usr="admin" pwd="cisco123">
  <request correlator="CiscoPnP-1.0-164-582-7F987E641C28-164" version="1.0"
xmlns="urn:cisco:pnp:cli-config">
  <configApply details="all">
  <config-data>
  <cli-config-data-block>no ntp
  service timestamps debug datetime msec localtime show-timezone
  [...]
  snmp-server community cisco RO SNMP-ACL</cli-config-data-block></config-data></configApply>
</request></pnp>
    From device 2015-12-08 23:24:10
  -------------------------------------------
  [{pnp,
       [{xmlns,"urn:cisco:pnp"},
        {version,"1.0"},
        {udi,"PID:CSR1000V,VID:V00,SN:9YLMGXIQ0ON"}],
       [{response,
            [{xmlns,"urn:cisco:pnp:cli-config"},
             {correlator,"CiscoPnP-1.0-164-582-7F987E641C28-164"},
             {success,"1"}],
            [{resultentry,
                 [{linenumber,"1"},{clistring,"no ntp"}],
                 [{success,[],[]}]},
             {resultentry,
                 [{linenumber,"2"},
                  {clistring,
                      "service timestamps debug datetime msec localtime show-timezone"}],
                 [{success,[],[]}]},
  [...]
```

```
             {resultentry,
                 [{linenumber,"82"},
                  {clistring,"snmp-server community cisco RO SNMP-ACL"}],
                 [{success,[],[]}]}]}]}]}]
  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:CSR1000V,VID:V00,SN:9YLMGXIQ0ON"><response
xmlns="urn:cisco:pnp:cli-config" correlator="CiscoPnP-1.0-164-582-7F987E641C28-164"
success="1"><resultEntry lineNumber="1" cliString="no ntp"><success change="CHANGE"
mode="IMMEDIATE" /></resultEntry><resultEntry lineNumber="2" cliString="service timestamps
debug datetime msec localtime show-timezone"><success change="CHANGE" mode="IMMEDIATE" />
</resultEntry>[...]<resultEntry lineNumber="82" cliString="snmp-server community cisco RO SNMP-
ACL"><success change="CHANGE" mode="IMMEDIATE" /></resultEntry></response></pnp>
    To device 2015-12-08 23:24:11
  -------------------------------------------
  [{pnp,[{xmlns,"urn:cisco:pnp"},
         {version,"1.0"},
         {udi,"PID:CSR1000V,VID:V00,SN:9YLMGXIQ0ON"},
         {usr,"admin"},
         {pwd,"cisco123"}],
        [{info,[{correlator,"CiscoPnP-1.0-164-582-7F987E641C28-164"},
                {xmlns,"urn:cisco:pnp:work-info"}],
               {workInfo,[],[{bye,[]}]}]}]}]
```

Reset CPE to Day -1 example

The CPE being reset, cleaned or removed from a service chain results in the PnP server telling the CPE to `copy flash:day--1-config startup-config` and then reload the CPE.

```
  From device 2015-12-07 22:27:37
  -------------------------------------------
  [{pnp,
       [{xmlns,"urn:cisco:pnp"},
        {version,"1.0"},
        {udi,"PID:ISR880,VID:V00,SN:9CANYW0SMDU"}],
       [{info,
            [{xmlns,"urn:cisco:pnp:work-info"},
             {correlator,"CiscoPnP-1.0-2594-346-7FDEC7DDF6F8-2595"}],
            [{deviceid,[],
                 [{udi,[],["PID:ISR880,VID:V00,SN:9CANYW0SMDU"]},
                  {hostname,[],["router"]},
                  {authrequired,[],["true"]},
                  {viaproxy,[],["false"]},
                  {securityadvise,[],
                      ["Password in clear text in unsecured transport"]}]}]}]}]}]
  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9CANYW0SMDU"><info
xmlns="urn:cisco:pnp:work-info" correlator="CiscoPnP-1.0-2594-346-7FDE
```

```
  C7DDF6F8-2595"><deviceId><udi>PID:ISR880,VID:V00,SN:9CANYW0SMDU</udi>
<hostname>router</hostname><authRequired>true</authRequired><viaProxy>false</viaProxy><s
  ecurityAdvise>Password in clear text in unsecured transport</securityAdvise></deviceId>
</info></pnp>
    To device 2015-12-07 22:27:37
  --------------------------------------------
  [{pnp,
       [{xmlns,"urn:cisco:pnp"},
        {version,"1.0"},
        {udi,"PID:ISR880,VID:V00,SN:9CANYW0SMDU"},
        {usr,"admin"},
        {pwd,"cisco123"}],
       [{request,
            [{correlator,"CiscoPnP-1.0-2594-346-7FDEC7DDF6F8-2595"},
             {version,"1.0"},
             {xmlns,"urn:cisco:pnp:config-upgrade"}],
            [{config,[],
                 [{copy,[],
                      [{source,[],[{location,[],"flash:day--1-config"}]},
                       {applyTo,[],"startup"}]}]},
             {reload,[],
                 [{reason,[],"scheduled reload for cVPN undeployment"},
                  {delayIn,[],"20"},
                  {user,[],"admin"}]}]}]}]

  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9CANYW0SMDU" usr="admin"
pwd="cisco123">
  <request correlator="CiscoPnP-1.0-2594-346-7FDEC7DDF6F8-2595" version="1.0"
xmlns="urn:cisco:pnp:config-upgrade">
  <config>
  <copy>
  <source>
  <location>flash:day--1-config</location></source>
  <applyTo>startup</applyTo></copy></config>
  <reload>
  <reason>scheduled reload for cVPN undeployment</reason>
  <delayIn>20</delayIn>
  <user>admin</user></reload></request></pnp>
    From device 2015-12-07 22:27:42
  --------------------------------------------
  [{pnp,[{xmlns,"urn:cisco:pnp"},
         {version,"1.0"},
         {udi,"PID:ISR880,VID:V00,SN:9CANYW0SMDU"}],
        [{response,[{correlator,"CiscoPnP-1.0-2594-346-7FDEC7DDF6F8-2595"},
                    {success,"1"},
                    {xmlns,"urn:cisco:pnp:config-upgrade"}],
                   []}]}]
  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9CANYW0SMDU"><response
correlator="CiscoPnP-1.0-2594-346-7FDEC7DDF6F8-2595" success="1" xmlns="urn:cisco:pnp:config-
upgrade"></response></pnp>
    To device 2015-12-07 22:27:42
  -------------------------------------------
  [{pnp,[{xmlns,"urn:cisco:pnp"},
         {version,"1.0"},
         {udi,"PID:ISR880,VID:V00,SN:9CANYW0SMDU"},
         {usr,"admin"},
         {pwd,"admin"}],
        [{info,[{correlator,"CiscoPnP-1.0-2594-346-7FDEC7DDF6F8-2595"},
                {xmlns,"urn:cisco:pnp:work-info"}],
               {workInfo,[],[{bye,[]}]}]}]

  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9CANYW0SMDU" usr="admin"
pwd="admin">
  <info correlator="CiscoPnP-1.0-2594-346-7FDEC7DDF6F8-2595" xmlns="urn:cisco:pnp:work-info">
  <workInfo>
  <bye></bye></workInfo></info></pnp>
```

CPE Remote Command Execution

It is possible to send commands to the CPE via PnP and then observe the CPE output in the `PID*<serial_number>.trace` file. The example below shows an administrator logging into the NSO CLI and executing an exec command to the CPE, which gets run the next time that the CPE contacts the PnP server, and the output is placed into the `PID*<serial_number>.trace` file. The output shows a tail on that file until it shows the expected output.

```
ubuntu@ncs-vm:/var/log/ncs$ ncs_cli -u admin

admin connected from 10.0.101.1 using ssh on ncs-vm
admin@ncs-vm>
admin@ncs-vm> request pnp exec command "do show int Tunnel0" serial 9AR13N53RHC
[ok][2015-12-09 16:06:12]
admin@ncs-vm> exit
ubuntu@ncs-vm:/var/log/ncs$ tail -f -n 100 /var/log/ncs/PID*RHC.trace
  From device 2015-12-09 16:04:43
-------------------------------------------
[{pnp,[{xmlns,"urn:cisco:pnp"},
       {version,"1.0"},
       {udi,"PID:ISR880,VID:V00,SN:9AR13N53RHC"}],
      [{response,[{correlator,"CiscoPnP-1.0-24-583-7FAECFC2F4B8-25"},
                    {success,"1"},
                    {xmlns,"urn:cisco:pnp:backoff"}],
                   []}]}]
  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9AR13N53RHC"><response
correlator="CiscoPnP-1.0-24-583-7FAECFC2F4B8-25" success="1" xmlns="urn:cisco:pnp:backoff">
</response></pnp>
    To device 2015-12-09 16:04:43
  -------------------------------------------
  [{pnp,[{xmlns,"urn:cisco:pnp"},
         {version,"1.0"},
         {udi,"PID:ISR880,VID:V00,SN:9AR13N53RHC"},
         {usr,"admin"},
         {pwd,"cisco123"}],
        [{info,[{correlator,"CiscoPnP-1.0-24-583-7FAECFC2F4B8-25"},
                {xmlns,"urn:cisco:pnp:work-info"}],
               {workInfo,[],[{bye,[]}]}]}]}]

  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9AR13N53RHC" usr="admin"
pwd="cisco123">
  <info correlator="CiscoPnP-1.0-24-583-7FAECFC2F4B8-25" xmlns="urn:cisco:pnp:work-info">
  <workInfo>
  <bye></bye></workInfo></info></pnp>
    From device 2015-12-09 16:08:13
  -------------------------------------------
  [{pnp,
       [{xmlns,"urn:cisco:pnp"},
        {version,"1.0"},
        {udi,"PID:ISR880,VID:V00,SN:9AR13N53RHC"}],
       [{info,
            [{xmlns,"urn:cisco:pnp:work-info"},
             {correlator,"CiscoPnP-1.0-25-583-7FAECFC2F1E0-26"}],
            [{deviceid,[],
                 [{udi,[],["PID:ISR880,VID:V00,SN:9AR13N53RHC"]},
                  {hostname,[],["router"]},
                  {authrequired,[],["true"]},
                  {viaproxy,[],["false"]},
                  {securityadvise,[],
                      ["Password in clear text in unsecured transport"]}]}]}]}]
  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9AR13N53RHC"><info
xmlns="urn:cisco:pnp:work-info" correlator="CiscoPnP-1.0-25-583-7FAECFC2F1E0-26"><deviceId>
<udi>PID:ISR880,VID:V00,SN:9AR13N53RHC</udi><hostname>router</hostname>
<authRequired>true</authRequired><viaProxy>false</viaProxy><securityAdvise>Password in clear
text in unsecured transport</securityAdvise></deviceId></info></pnp>
    To device 2015-12-09 16:08:13
  -------------------------------------------
  [{pnp,
       [{xmlns,"urn:cisco:pnp"},
        {version,"1.0"},
        {udi,"PID:ISR880,VID:V00,SN:9AR13N53RHC"},
        {usr,"admin"},
        {pwd,"cisco123"}],
       [{request,
            [{correlator,"CiscoPnP-1.0-25-583-7FAECFC2F1E0-26"},
             {version,"1.0"},
             {xmlns,"urn:cisco:pnp:cli-config"}],
            [{configApply,
                 [{details,"all"}],
                 [{'config-data',[],
                      [{'cli-config-data-block',[],
                           "do show int Tunnel0\n\n"}]}]}]}]}]

  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9AR13N53RHC" usr="admin"
pwd="cisco123">
  <request correlator="CiscoPnP-1.0-25-583-7FAECFC2F1E0-26" version="1.0"
xmlns="urn:cisco:pnp:cli-config">
  <configApply details="all">
  <config-data>
  <cli-config-data-block>do show int Tunnel0

  </cli-config-data-block></config-data></configApply></request></pnp>
    From device 2015-12-09 16:08:13
  -------------------------------------------
  [{pnp,[{xmlns,"urn:cisco:pnp"},
         {version,"1.0"},
         {udi,"PID:ISR880,VID:V00,SN:9AR13N53RHC"}],
        [{response,[{xmlns,"urn:cisco:pnp:cli-config"},
                    {correlator,"CiscoPnP-1.0-25-583-7FAECFC2F1E0-26"},
                    {success,"1"}],
                   [{resultentry,[{linenumber,"1"},
                                  {clistring,"do show int Tunnel0"}],
                                 [{invalid,[],[]},
                                  {text,[],
                                        ["\n**CLI Line # 1: Tunnel0 is up, line protocol is
down \r\n**CLI Line # 1:   Hardware is Tunnel\r\n**CLI Line # 1:   Internet address is
10.254.0.10/32\r\n**CLI Line # 1:   MTU 9976 bytes, BW 100 Kbit/sec, DLY 50000 usec, \r\n**CLI
Line # 1:      reliability 255/255, txload 1/255, rxload 1/255\r\n**CLI Line # 1:
Encapsulation TUNNEL, loopback not set\r\n**CLI Line # 1:   Keepalive not set\r\n**CLI Line #
1:   Tunnel linestate evaluation down - linestate protection reg down\r\n**CLI Line # 1:
Tunnel source 192.168.11.16 (GigabitEthernet1), destination 192.168.12.50\r\n**CLI Line # 1:
Tunnel Subblocks:\r\n**CLI Line # 1:       src-track:\r\n**CLI Line # 1:          Tunnel0
source tracking subblock associated with GigabitEthernet1\r\n**CLI Line # 1:           Set of
tunnels with source GigabitEthernet1, 1 member (includes iterators), on interface ",
    ...........................
{resultentry,[{linenumber,"2"}],[{success,[],[]}]}]}]}]
  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9AR13N53RHC"><response
xmlns="urn:cisco:pnp:cli-config" correlator="CiscoPnP-1.0-25-583-7FAECFC2F1E0-26" success="1">
<resultEntry lineNumber="1" cliString="do show int Tunnel0"><invalid /><text>
  **CLI Line # 1: Tunnel0 is up, line protocol is down
  **CLI Line # 1:   Hardware is Tunnel
  **CLI Line # 1:   Internet address is 10.254.0.10/32
  **CLI Line # 1:   MTU 9976 bytes, BW 100 Kbit/sec, DLY 50000 usec,
  **CLI Line # 1:      reliability 255/255, txload 1/255, rxload 1/255
  **CLI Line # 1:   Encapsulation TUNNEL, loopback not set
  **CLI Line # 1:   Keepalive not set
  **CLI Line # 1:   Tunnel linestate evaluation down - linestate protection reg down
  **CLI Line # 1:   Tunnel source 192.168.11.16 (GigabitEthernet1), destination 192.168.12.50
  **CLI Line # 1:    Tunnel Subblocks:
  **CLI Line # 1:       src-track:
  **CLI Line # 1:          Tunnel0 source tracking subblock associated with GigabitEthernet1
  **CLI Line # 1:           Set of tunnels with source GigabitEthernet1, 1 member (includes
iterators), on interface <OK>
  ..............................................  **CLI Line # 1:      0 output buffer
failures, 0 output buffers swapped out</text></resultEntry><resultEntry lineNumber="2"><success
change="NO_CHANGE" mode="IMMEDIATE" /></resultEntry></response></pnp>
    To device 2015-12-09 16:08:13
  -------------------------------------------
  [{pnp,[{xmlns,"urn:cisco:pnp"},
         {version,"1.0"},
         {udi,"PID:ISR880,VID:V00,SN:9AR13N53RHC"},
         {usr,"admin"},
         {pwd,"cisco123"}],
        [{info,[{correlator,"CiscoPnP-1.0-25-583-7FAECFC2F1E0-26"},
                {xmlns,"urn:cisco:pnp:work-info"}],
               {workInfo,[],[{bye,[]}]}]}]}]

  <pnp xmlns="urn:cisco:pnp" version="1.0" udi="PID:ISR880,VID:V00,SN:9AR13N53RHC" usr="admin"
pwd="cisco123">
  <info correlator="CiscoPnP-1.0-25-583-7FAECFC2F1E0-26" xmlns="urn:cisco:pnp:work-info">
  <workInfo>
  <bye></bye></workInfo></info></pnp>
```

Troubleshooting IPsec tunnel establishment

Troubleshooting the Management IPsec tunnel

If the Day 0 config has been pushed by PnP to the CPE, but the CPE is not reachable via ping or SSH from the Management Hub or from NSO, the state of that CPE will not transition beyond the "True/True/False" state in `show pnp list`. In this case, most likely the CPE Management IPsec tunnel (Tunnel0) is not coming up.

In this case, troubleshooting can be done:

- on the CPE, either using a serial console or using the PnP remote command execution described above
- on the Management Hub (keeping in mind that the Management Hub may be terminating other CPE Management IPsec tunnels at the same time, so any debugging done on the Management Hub may be chatty and should be filtered to limit output to the CPE in question)

As a first step, it is useful to get an understanding of the network topology at the CPE remote site. Does the CPE directly connect to the public Internet? Are there any devices such as firewalls, or home routers (including Cable, DSL or Mobile Internet routers) along the path that may either restrict access to the Internet, and/or perform Network Address Translation (NAT)? If there are devices along the network path performing packet filtering or NAT, then it may be useful to get details of these devices, and verify they are configured to meet the requirements documented in Chapter 7.3 "CPE Requirements" in this section. As a reminder, several NAT routers may need specific configuration to permit IPsec traffic.

For further troubleshooting, it is useful to note three IP addresses associated with the CPE:

1. WAN IP address: This is the IP address the CPE has received, from the DHCP server, on its WAN interface.
2. Outside IP address: This IP address is not known to the CPE itself, but is the IP address the CPE would be visible with to the outside world if there is NAT along the network path. If there is no NAT, then the CPE's WAN IP address and Outside IP address are identical.
3. Management IP address: This is the IP address allocated by NSO within the management tunnel. Once the Day 0 configuration has been deployed to the CPE, this IP address is statically configured on the CPE's Tunnel0 interface. This will be the IP address that the NSO uses to reach the CPE.

To obtain the IP addresses:

On the CPE:

- Execute s`how ip interface brief`. The IP address shown here for the WAN interface is the CPE's WAN IP address.

On NSO:

- Execute `show pnp list` and take note of the IP address shown for the CPE here. This IP address is the CPE's IP address as seen by NSO, i.e. the address referred to as "Outside IP address" above.
- Execute `show pnp-state device <serial number>`. The IP address seen as "mgmt-ip" is the CPE's Management IP address.

Specific troubleshooting steps on the CPE include:

- `show interface Tunnel0` on the CPE. If the Management IPsec tunnel established successfully, Tunnel0 state will show as "interface up, line protocol up". If the interface or line protocol is down or is flapping, there is an issue with establishing the tunnel. If Tunnel0 is not present at all, Day 0 config was not properly applied.
- `show log`. Even without any debug, these logs would show if Tunnel0 has been flapping recently, and may show any other relevant logs or errors.

Specific troubleshooting steps on the Management Hub include:

- `show interface`, `show ip route`. Verify there is a Virtual-Access interface for the CPE -- there should be a Virtual-Access interface with a host route towards the CPE's Management Address. In case there are multiple Management Hubs, check all of them, and attempt to ping the CPE from the specific Management Hub on which the Virtual-Access interface is present, if any. If the CPE is reachable from the Management Hub but not from NSO, there may be a routing issue between NSO and the Management Hub.

Troubleshooting steps on both CPE and Management Hub:

- `debug crypto ikev2`. (On the Management Hub, it may be necessary to restrict this output to a single CPE by entering `debug condition ip <outside IP address>`)
  - There should be four packets being exchanged between CPE and Management Hub:
    - IKE_SA_INIT Exchange REQUEST (from CPE to Management Hub)
    - IKE_SA_INIT Exchange RESPONSE (from Management Hub to CPE)
    - IKE_AUTH Exchange REQUEST (from CPE to Management Hub)
    - IKE_AUTH Exchange RESPONSE (from Management Hub to CPE)
  - If any of the above packets is sent by one side but not received by the other side, this will result in retransmission and may indicate packet loss, or lack of IPsec support, on any device along the network path.
- `debug crypto ipsec` may be useful for further IPsec troubleshooting if IKEv2 succeeds but the IPsec tunnel still fails to come up for any reason
- In case any IKEv2 packets or IPsec packets are lost along the network path, capture external packet traces at different points along the network path to narrow down the source of packet loss.

- If there are firewalls or NAT devices along the network path, verify that the UDP session timeout on these devices is longer than 60 seconds (minimum of 120 seconds suggested).
- If possible, move the CPE to a known working location (while keeping it attached to the same service chain and keeping the same Day 0 configuration) to narrow down the root cause to the CPE/Service Chain or to the location.

Troubleshooting the Service IPsec tunnel

The same troubleshooting steps that apply to the Management IPsec tunnel, also apply to the Service IPsec tunnel. If troubleshooting the Service IPsec tunnel, the Service Chain's CSR1000v replaces the Management Hub in all of the troubleshooting steps described above.

Two-Tunnel requirement

In rare occasions, there may be firewalls or NAT devices that can only support a single IPsec tunnel but not two. This would manifest itself in the Management IPsec tunnel coming up initially, and the Day 1/Day 2 configuration being deployed successfully, and then either:

- the Service IPsec tunnel failing to come up,
- the Managment IPsec tunnel going down as soon as the Service IPSEC tunnel comes up,
- Both tunnels starting to flap after the Day 1/Day 2 configuration is applied.

If any of these symptoms are observed, ensure that the network infrastructure between CPE and Management Hub (CSR1000v) supports multiple IPsec tunnels for this CPE, and narrow down the cause of packet loss by using debugs and/or packet traces.

Day 1/Day 2 configuration

Day 1/Day 2 configuration is pushed to the CPE from the NSO. The NSO establishes an SSH session to the CPE via the information displayed in `show pnp-state device` output and then sends the configuration. A record of this configuration and other interactions are stored in `ned*<serial_number>.trace` files in `/var/log/ncs/` directory. The logs show NSO logging into devices, each command that it executes, along with the CPE responses. Experience with IOS configuration makes reading these files very simple. If the log displays timeouts, it is likely that there are problems with NSO establishing an SSH session to the CPE. This is usually caused by the Tunnel0 interface state on CPE not being up or flapping.

Connecting to CPE console

This is a last resort for troubleshooting because it requires on-site presence and the use of a laptop with a console cable attached to the CPE. If PnP has completed Day 0 configuration as seen in the `PID*.trace` files, then it will also require the password to be shared with the person on the console. The password can be obtained from the output of `show pnp-state device <SERIAL_NUMBER>` as previously described.

8. vMS Operations and Administration

8.1 Introduction

This chapter describes common administrative and operational tasks needed to successfully operate the vMS solution. This chapter is divided into two logical parts:

- Infrastructure Administration Tasks - shows the most common vMS infrastructure-related administration tasks such as: orchestration components (NSO/ESC) log management, Service Chain data center migration, VNF image upgrade procedures and adding compute nodes to the existing environment.
- Security Considerations - describes security best practices for deploying the vMS solution.

8.2 Infrastructure Administration Tasks

Log management

NSO

Log rotation settings for NSO are located in `/etc/logrotate.d/cloudvpn.conf`. A daily cron job will rotate log files if they have reached file size limits specified in `cloudvpn.conf` file.

- Access logs and *.logs in NSO log directory if bigger than 150M in size.
- Trace logs in NSO log directory are rotated if bigger than 20M in size.

> **Note**: The log rotate cron job is executed once per day. As a result, log files can grow larger than the mentioned sizes if there is a lot of activity on the system.

ESC

- ESC manager logs in ESC log directory are rotated if bigger than 150M in size.
- YANG debug logs in ESC log directory are rotated if bigger than 50M in size.

ESC Service Suspension

The NSO will suspend the use of ESC if OpenStack is not available. When ESC detects Keystone identity management is not running or otherwise not reachable on OpenStack, ESC notifies a failure in VIM operational state to the NSO and logs the following lines to `yangesc.log`:

```
09:11:40,380 09-Dec-2015 INFO  ===== SEND NOTIFICATION STARTS =====
09:11:40,380 09-Dec-2015 INFO  Type: VIM_OPERATIONAL_STATE
09:11:40,380 09-Dec-2015 INFO  Status: FAILURE
09:11:40,381 09-Dec-2015 INFO  Status Code: 0
09:11:40,381 09-Dec-2015 INFO  Status Msg: VIM Operational State Down
09:11:40,381 09-Dec-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
```

Notification sent by ESC to NSO is logged in `netconf-<esc_name>.trace` file:

```
<<<<in 9-Dec-2015::14:11:40.130 device=esc session-id=14
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2015-12-09T14:11:40.358+00:00</eventTime>
  <escEvent xmlns="http://www.cisco.com/esc/esc">
    <status>FAILURE</status>
```

```
      <status_code>0</status_code>
      <status_message>VIM Operational State Down</status_message>
      <vm_source/>
      <vm_target/>
      <event>
        <type>VIM_OPERATIONAL_STATE</type>
      </event>
    </escEvent>
  </notification>
```

NSO will mark the infrastructure as not active once this notification is received from ESC. Subsequent service chains will not be deployed to this particular data center. The following command can be used to verify the state:

```
admin@ncs-vm> show configuration infrastructure esc is-active
esc esc {
    is-active false;
}
[ok][2015-12-09 14:11:45]
admin@ncs-vm>
```

If it is desired to not deploy service chains to a particular data center, the infrastructure has to be manually marked as not active in the NSO. The subsequent service chains will then not be deployed to this particular data center.

Reactivate ESC

Once Keystone identity management has recovered, the ESC will notify the NSO and log the following messages in `yangesc.log`:

```
09:15:26,487 09-Dec-2015 INFO  ===== SEND NOTIFICATION STARTS =====
09:15:26,487 09-Dec-2015 INFO  Type: VIM_OPERATIONAL_STATE
09:15:26,488 09-Dec-2015 INFO  Status: SUCCESS
09:15:26,488 09-Dec-2015 INFO  Status Code: 0
09:15:26,488 09-Dec-2015 INFO  Status Msg: VIM Operational State Up
09:15:26,488 09-Dec-2015 INFO  =====  SEND NOTIFICATION ENDS  =====
```

Notification sent by ESC to NSO is logged in `netconf-<esc_name>.trace` file:

```
<<<<in 9-Dec-2015::14:15:26.239 device=esc session-id=14
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2015-12-09T14:15:26.474+00:00</eventTime>
  <escEvent xmlns="http://www.cisco.com/esc/esc">
    <status>SUCCESS</status>
    <status_code>0</status_code>
    <status_message>VIM Operational State Up</status_message>
    <vm_source/>
    <vm_target/>
    <event>
      <type>VIM_OPERATIONAL_STATE</type>
    </event>
  </escEvent>
</notification>
```

NSO will mark the infrastructure as active once this notification is received from ESC, and will provision subsequent service chains to this Data Center.

```
admin@ncs-vm> show configuration infrastructure esc is-active
esc esc {
    is-active true;
}
[ok][2015-12-09 14:15:41]
```

Migration of Service chains between Data Centers

NSO provides an option to migrate all service chains from one virtual Data Center to another. In the case of extended, unscheduled downtime in the virtual infrastructure, the administrator should mark the infrastructure as not active before migrating all service chains to another. During scheduled maintenance, the administrator has the option to migrate all service chains to another infrastructure before marking the infrastructure as not active.

```
admin@ncs-vm> request cloudvpn-action undeploy-redeploy vnfm esc2
```

Individual service chains can be migrated from one virtual Data Center to another following the round robin ESC selection algorithm. If the objective is to move a particular service chain from one infrastructure to another, mark the current infrastructure as inactive before executing the command below.

```
admin@ncs-vm> request cloudvpn-action undeploy-redeploy cloudvpn <service-name>
```

Upgrading VNF images

Out-of-band Image and Flavor Management for Hosted Deployment model

In this vMS deployment model, all the OpenStack tenants, flavors and images are created by an OpenStack administrator using Horizon GUI or Keystone, Glance and Nova CLI. In this case, ESC only needs user privileges in OpenStack. The image upload is done in the following way:

- Upload new VNF images to Glance
- Add new version for VNF images in `escday0` configuration.

Subsequent service chains will use the new VNF images.

```
admin@ncs-vm> show configuration escday0
image-registration vFirewall 1.1 {
    image-id  <openstack image UUID>;
    flavor-id <openstack flavor UUID>;
    unmanaged;
    devices   [ esc ];
}
image-registration vRouter 1.1 {
    image-id  <openstack image UUID>;
    flavor-id <openstack flavor UUID>;
    unmanaged;
    devices   [ esc ];
}
image-registration vWSA 1.1 {
    image-id  <openstack image UUID>;
    flavor-id <openstack flavor UUID>;
    unmanaged;
    devices   [ esc ];
}
image-registration vFirewall 1.2 {
    image-id  <openstack image UUID>;
    flavor-id <openstack flavor UUID>;
    unmanaged;
    devices   [ esc ];
}
image-registration vRouter 1.2 {
    image-id  <openstack image UUID>;
    flavor-id <openstack flavor UUID>;
    unmanaged;
    devices   [ esc ];
}
image-registration vWSA 1.2 {
    image-id  <openstack image UUID>;
    flavor-id <openstack flavor UUID>;
    unmanaged;
    devices   [ esc ];
}
[ok][2015-12-09 15:29:04]
```

VNF images on existing service chains can be upgraded by undeploying and redeploying the service chain.

```
admin@ncs-vm> request cloudvpn-action undeploy-redeploy cloudvpn <service-name>
```

In-band Image and Flavor management for On-Premise Deployment model

This vMS deployment model mandates that all OpenStack objects are created by NSO through ESC. In this case, ESC will require administrative privileges for OpenStack. The image upload is done in the following way:

- The NSO hosts a web server from where ESC can fetch VNF images. ESC can fetch VNF images from an external server as well.
- On the NSO, configure `escday0` to point to an URL for new VNF images, where ESC can retrieve them.
- New VNF images are created in Glance on OpenStack.

Subsequent service chains will use the new VNF images.

```
admin@ncs-vm> show configuration escday0 registration
registration 1.1 {
    devices    [ esc1 esc2 ];
    csr-url    https://[IPv4 or IPv6 address]/images/csr11_latest.qcow2;
    asa-url    https://[IPv4 or IPv6 address]/images/asa11_latest.qcow2;
    wsa-url    https://[IPv4 or IPv6 address]/images/wsa11_latest.qcow2;
}
registration 1.2 {
    devices    [ esc1 esc2 ];
    csr-url    https://[IPv4 or IPv6 address]/images/csr12_latest.qcow2;
    asa-url    https://[IPv4 or IPv6 address]/images/asa12_latest.qcow2;
    wsa-url    https://[IPv4 or IPv6 address]/images/wsa12_latest.qcow2;
}
[ok][2015-12-09 15:59:39]
```

VNF images on existing service chains can be upgraded by undeploying and redeploying the service chain.

```
admin@ncs-vm> request cloudvpn-action undeploy-redeploy cloudvpn <service-name>
```

Adding compute nodes in an On-Premise Deployment model

NSO

NSO is prepped in the following way before compute nodes can be added:

- Update the resource pools by configuring the IP address pool associated with the new compute node.

- Associate the IP resource pool by configuring the same under occupancy.

```
admin@vMS-NSO-sm> show configuration resource-pools ip-address-pool compute2
    ip-address-pool compute2 {
        owner vMS;
        tags [ br-outside-compute2 ];
        subnet 192.168.100.0 24;
        exclude 192.168.100.0 32;
        exclude 192.168.100.51 32;
        exclude 192.168.100.75 32;
        exclude 192.168.100.185 32;
        exclude 192.168.100.255 32;
    }
}
  admin@vMS-NSO-sm> show configuration occupancy
provider vMS-lab {
    vnfm esc-device;
    ip-resource-pool CPE-Superpool;
    ip-resource-pool br-inside;
    ip-resource-pool csr-to-csr;
    ip-resource-pool loopback;
    ip-resource-pool compute1;
    ip-resource-pool compute2;
}
[ok][2015-12-08 16:19:40]
```

ESC

ESC makes use of a PostgreSQL database. If ESC was installed with compute-host properties, CIDR blocks for new compute-host are appended to compute-host properties in the database.

```
  psql -p 7878 -U esc
    select * from esc_schema.compute_host_properties;

   computehostid |     cidr_address      | cidr_prefix | compute_host |        gateway         |
ip_address_pool_end    |  ip_address_pool_start   | neutron_network_name | network_uuid
  ---------------+----------------------+-------------+--------------+-----------------------+-
-------------------------+--------------------------+----------------------+-------------------
-------------------
               1 | 172.32.100.0         | 24          | compute1     | 172.32.100.1          |
172.32.100.254        |     172.32.100.2         | vnf-management       | 0462a8ec-0798-4f8b-
ae69-99840568cb60
  (1 rows)
  insert into esc_schema.compute_host_properties (cidr_address, cidr_prefix, compute_host,
gateway, ip_address_pool_end, ip_address_pool_start, neutron_network_name, network_uuid)
   VALUES ('192.168.100.0', 24, 'compute2', '192.168.100.1', '192.168.100.254',
'192.168.100.2', 'vnf-management', '0462a8ec-0798-4f8b-ae69-99840568cb60');

  select * from esc_schema.compute_host_properties;

   computehostid |     cidr_address      | cidr_prefix | compute_host |        gateway        |
ip_address_pool_end    |  ip_address_pool_start    | neutron_network_name | network_uuid
  ---------------+----------------------+-------------+--------------+-----------------------+-
-------------------------+-------------------------+----------------------+-------------------
-------------------
               1 | 172.32.100.0         | 24          | compute1     | 172.32.100.1          |
172.32.100.254        |     172.32.100.2         | vnf-management       | 0462a8ec-0798-4f8b-
ae69-99840568cb60
               2 | 192.168.100.0        | 24          | compute2     | 192.168.100.1         |
192.168.100.254       |     192.168.100.2        | vnf-management       | 0462a8ec-0798-4f8b-
ae69-99840568cb60
```

Manual Service Clean Up

In the event that an ESC becomes unrecoverable, it may be necessary to manually clean the NSO and remove virtual resources from OpenStack once services have been recovered to a second Data Center.

In order to remove an ESC device from NSO, first set the administrative state of the ESC to southbound-locked before deleting the device configuration. This is shown in the example below.

```
  admin@ncs-vm% set devices device esc state admin-state southbound-locked
  [ok][2015-12-09 16:03:25]

  [edit]
  admin@ncs-vm% commit
  Commit complete.
  [ok][2015-12-09 16:03:50]

  [edit]
  admin@ncs-vm% delete devices device esc
  [ok][2015-12-09 16:04:01]

  [edit]
  admin@ncs-vm% commit
```

After removing the ESC device from the NSO configuration, it may be necessary to remove any remaining instances, Nova flavors and Neutron networks that have been created by ESC from the OpenStack infrastructure. The following OpenStack client commands are used to identify ESC create instances, flavors and networks.

```
nova list | grep BR-INSIDE
nova flavor-list | grep ESC_Day0
neutron net-list | grep BR-INSIDE
```

Depending on the deployment model (On-premises vs Hosted) it may also be necessary to remove the provider-tenant/project from OpenStack along with the VNF images in Glance. Essentially if ESC was used to create the project and upload VNF images during registration, then these will also need to be manually removed before ESC can be redeployed. The tenant name in OpenStack will be the same as that used for the provider name in NSO.

`keystone tenant-list` and `keystone tenant-delete` commands can be used to identify and remove any ESC created tenants.

```
keystone tenant-list
keystone tenant-delete <tenant-id>
```

`glance image-list` command can be used to identify and `glance image-delete` command can be used to remove any ESC created images from Glance:

```
glance image-list | grep ESC_Day0
glance image-delete <image-id>
```

Upgrade ESC

Upgrade ESC in HA mode

There are currently two models for running ESC in HA mode. The first model uses a shared Cinder volume to store a database which is remounted during failover to the new MASTER. The second model replicates a DRDB database directly between two ESC instances using TCP/IP.

The procedures below assume that the new ESC image has already been added to the Glance image store and the corresponding `bootvm.py` script is available to use for deploying the new ESC image.

Upgrading ESC in Cinder HA model:

1. Connect to the ESC BACKUP instance. This can be checked using the `cat /cisco/keepalived_state`.

2. Using Nova, stop and rename the ESC BACKUP instance.

```
cisco@jungfrau:~$ nova stop 3da7a9d8-1c9b-4d1b-a8d0-905cd236d89b
```

```
  Request to stop server 3da7a9d8-1c9b-4d1b-a8d0-905cd236d89b has been accepted.

  cisco@jungfrau:~$ nova rename 3da7a9d8-1c9b-4d1b-a8d0-905cd236d89b old_esc
  cisco@jungfrau:~$ nova list | grep esc
  | 3da7a9d8-1c9b-4d1b-a8d0-905cd236d89b | old_esc    | SHUTOFF | -          | Shutdown    |
orch-mgmt=192.168.12.111 |
```

3. Deploy the ESC BACKUP instance using the `bootvm.py` deployment script with the same parameters as the current BACKUP instance (currently SHUTOFF) including the existing ESC Cinder volume. You should specify the target ESC image that you are upgrading to.

4. Once the new BACKUP instance has booted connect to the VM and verify that it has entered BACKUP state using `cat /cisco/keepalived_state`.

5. Copy the SSH server keys from `/etc/ssh/ssh_host_*` on the MASTER to the new BACKUP, overwriting the SSH server key files on the BACKUP instance. Be careful to preserve the file ownership and permissions on the SSH keys.

```
  [localadmin@esc-1 ~]$ ls -la /etc/ssh/ssh_host_*
  -rw-------. 1 root root  668 Oct 12 13:20 /etc/ssh/ssh_host_dsa_key
  -rw-r--r--. 1 root root  590 Oct 12 13:20 /etc/ssh/ssh_host_dsa_key.pub
  -rw-------. 1 root root  963 Oct 12 13:20 /etc/ssh/ssh_host_key
  -rw-r--r--. 1 root root  627 Oct 12 13:20 /etc/ssh/ssh_host_key.pub
  -rw-------. 1 root root 1675 Oct 12 13:20 /etc/ssh/ssh_host_rsa_key
  -rw-r--r--. 1 root root  382 Oct 12 13:20 /etc/ssh/ssh_host_rsa_key.pub
```

After this step, the original MASTER is still running as MASTER and the BACKUP ESC instance is running the new software image with a copy of the MASTER SSH keys.

6. Shutdown the MASTER instance from the shell using `sudo shutdown now`.

7. Use Nova to stop the MASTER instance and rename as instructed in step 2 for the BACKUP instance.

8. Connect to the BACKUP instance and verify that the HA status moves from SWITCHING --> MASTER with `tail -f /cisco/keepalived_state`. The switching status can be checked by tailing the log file `/var/log/esc/esc_monitor.log`.

```
  [localadmin@esc-2 ~]$ tail -f /var/log/esc/esc_monitor.log
          {
              "id": "ece585b0-9ca5-4683-9c57-06669a79715e",
              "message": "tomcat6 (pid 12847) is running...[  OK  ]",
              "type": "tomcat6"
          }
```

```
    ],
    "operational_period": 15000,
    "operational_state": "OPER_UP"
}
127.0.0.1 - - [11/Dec/2015 14:05:16] "POST /status HTTP/1.1" 200 0
```

Also the verify that the Cinder volume has been mounted using `df -h`.

```
[localadmin@esc-2 ~]$ df -h
Filesystem       Size  Used Avail Use% Mounted on
/dev/vda2         20G  1.4G   18G   8% /
tmpfs            7.8G  4.0K  7.8G   1% /dev/shm
/dev/vda1        477M   55M  397M  13% /boot
/dev/vda6        3.9G  140M  3.5G   4% /opt
/dev/vda3         20G  312M   19G   2% /var
/dev/sr0         424K  424K     0 100% /media/cdrom
/dev/vdb1        992M   85M  857M   9% /mnt/esc_database
```

9. Once the ESC reaches the MASTER status verify from NSO that a successful connection can be made with the command `request devices devices esc connect`.

```
admin@ncs-vm> request devices device esc connect
result true
info (admin) Connected to esc - 192.168.15.20:830
[ok][2015-12-11 14:12:37]
```

10. Deploy the second ESC instance with the same parameters as the original MASTER. Once the instance has booted, log in and verify that it enters the BACKUP status with the command `cat /cisco/keepalived_state`.

11. Finally, copy the SSH server keys from the current MASTER to the BACKUP, again being careful to maintain the file ownership and permissions. The goal here is maintain the SSH server identity of the original ESC across both instances. If this step is not completed, it will be necessary to resync SSH keys on NSO using the command `request devices device <esc-device> ssh fetch-host-keys`.

Following this upgrade procedure, it is advisable to test ESC HA failover by rebooting the MASTER ESC instance. The old ESC instances that were shutdown and renamed can be deleted from OpenStack.

ESC HA with Distributed Replicated Block Device (DRBD):

From ESC version 1.1 it is possible to deploy ESC HA using Distributed Replicated Block Device (DRBD) instead of Cinder volumes. In this case the upgrade procedure is almost identical to the above Cinder upgrade procedure, however there will be no Cinder volume mounted in either ESC instance.

Before failing over the MASTER in step 6, verify that DRBD has completed replication of the local block device using the command `esc_service status` and `drdb -overview`

```
  [admin@esc-primary ~]$ drbd-overview
   1:esc/0  Connected Secondary/Primary UpToDate/UpToDate
  [admin@esc-backup ~]$ drbd-overview
   1:esc/0  Connected Primary/Secondary UpToDate/UpToDate /mnt/esc_database ext4 2.7G 75M 2.5G
3%
```

Note: Review the release notes to ensure that upgrade from/to the specific versions are supported

8.3 Security Considerations

The security recommendations made here should be considered as a minimum requirement for the vMS, however, each deployment should evaluate the specific security risks and operational procedures required to safeguard the system. An organization might be required to take steps to harden individual operating systems in addition to what is described here.

SSH should be permitted to allow the administrator access to NSO. If possible, this should be restricted to specific management address ranges	2022 TCP	Required by NSO HA-Clustering

IPtables and Required ports

The orchestration infrastructure comprises multiple virtual machines running mainly RedHat, CentOS or Ubuntu distributions of Linux. It is recommended that IPtables are enabled on each of the vMS virtual machines.

Table 8.1 below lists each of the components of the solution along with the TCP/UDP ports that are required for correct operation.

Table 8.1 Open Ports for vMS Components

Component	TCP/UDP Ports	Description
NSO	22 TCP	
	4570 TCP	Required by NSO HA-Clustering
	2605 TCP	Required by NSO to access Quagga configuration for HA
	830 TCP	By default, the netconf runs on this port and is used, for example, by the portal
	179 TCP	Used by the quagga daemon to establish BGP with external router for HA anycast IP
	443 TCP	HTTPS is used by the PnP and Day 0 configuration services
	161/162 UDP	NSO uses SNMPv2c to collect operational data from the service devices (vRouter, vFirewall, CPE etc.)
ESC	22 TCP	SSH should be permitted to allow the administrator access to NSO. If possible, this should be restricted to specific management address ranges.
	830 TCP	By default, the netconf runs on this port and is used by NSO to communicate to the ESC.

	179 TCP	Required by exabgp to establish BGP session with external router for HA anycast IP
Portal	443 TCP	HA-Proxy will access the portal via HTTPS
HA-Proxy (Portal)	443 TCP	End customer clients will access the portal via HA-Proxy using HTTPS
	80 TCP	End customer clients accessing the portal via HTTP will be redirected to HTTPS

When setting up IPtables it is good practice to log any dropped packets with a rate limit. An example of IPtables filter configuration can be found below and can be used as base template before adding device specifics. Logs for dropped packets will be sent to the system log `/var/log/syslog.log` with the prefix "IPTables-Drop". Note that all return TCP and UDP traffic is allowed where the local device was the initiator.

```
#iptables Common rules for vMS

*filter

#Create new iptables chain for logging ACCEPT packets
-N LOG_ACCT
-A LOG_ACCT -m limit --limit 2/min -j LOG --log-prefix "IPTables-Accept: " --log-level 7
-A LOG_ACCT -j ACCEPT

#Create new iptables chain for logging DROP packets
-N LOG_DROP
-A LOG_DROP -m limit --limit 2/min -j LOG --log-prefix "IPTables-Drop: " --log-level 7
-A LOG_DROP -j DROP

#Allow outgoing traffic and related return treaffic for TCP and UDP:
-A INPUT -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT
-A INPUT -p udp -m state --state ESTABLISHED -j ACCEPT

#Allow ANY from: Loopback
-A INPUT -i lo -j ACCEPT

#Accept ICMP from ANY with rate-limit AVG 10PPS
-I INPUT -p icmp --icmp-type echo-request -m limit --limit 600/min -j ACCEPT
-I INPUT -p icmp --icmp-type echo-reply   -m limit --limit 600/min -j ACCEPT
-I INPUT -p icmp --icmp-type fragmentation-needed -m limit --limit 600/min -j ACCEPT

#Send UNMATCHED packets to LOG_DROP (log then drop)
-A INPUT -j LOG_DROP

#Set Default policy to DROP
-P INPUT DROP
```

```
COMMIT
```

Certificates for SSL Services

The following vMS solution elements will require SSL certificates:

NSO PnP/Day 0 Services

NSO will require a valid CA-signed certificate for the PnP and Day 0 configuration services, although actual SSL termination point for these services may be on a HA-Proxy instance. The installation for these certificates on the NSO is described in section "SSL Certificates" in Chapter 5.4 "NSO Operations".

Portal

The Portal will require a valid CA-signed certificate although the actual SSL termination point for these services may be on a HA-Proxy instance.

CPE

The CPE Day -1 configuration will require a CA certificate in order for the CPE to trust the NSO PnP service. Please refer to Chapter 7.4 "CPE Preparation and Verification".

ASAv VNFs for SSL VPN

The ASAv instance in the CloudVPN service will require a valid CA-signed SSL certificate, otherwise the AnyConnect clients will display a security warning when trying to connect. The SSL certificate is configured on the ASAv by NSO as part of the Day 0 configuration and should be a wildcard certificate matching only the domain portion of the SSL VPN domain-name (E.g. cvpn.cisco.com). The certificate is configured under the global provider variable VFW_PKCS12_CERT; please refer to section "Global Variables" in Chapter 5.3 "NSO Configurations".

Service Management Security

As part of the Day 0 configuration for each of the service devices (E.g. CPE, ASAv , CSR1000v, WSAv), the NSO will configure access lists to secure SSH and SNMP management access to the devices. The specific networks that will be allowed are defined in the provider-specific variable sections of the NSO global configuration; please refer section "Global Variables" in Chapter 5.3 "NSO Configuration". These entries should be adapted to only allow access from the specific NSO device nodes, SA components and other deployment-specific subnets such as those required by the network operations teams and tools.

The authentication credentials for the CSR1000v and ASAv are set under provider-specific global variables in NSO. The username and password variables set here are what NSO will use when generating the Day 0 configuration for the VNFs. The variables are shown here for reference (please refer to section "Global Variables" in Chapter 5.3 "NSO Configuration" for more information).

```
admin@ncs-vm> show configuration globals provider vms-lab-pod variable VFW_CUSTOMER_* | tab

                                                           ENCRYPTED
NAME                                     VALUE             VAL
----------------------------------------------------------------------
VFW_CUSTOMER_DNS_SERVER_1                192.168.0.10      -
VFW_CUSTOMER_DNS_SERVER_2                192.168.0.11      -
VFW_CUSTOMER_DOMAIN_NAME                 cisco.com         -
VFW_CUSTOMER_ENABLED_PASSWORD_ENCRYPTED  PVSASRJovmamnVkD  -
VFW_CUSTOMER_PASSWORD_ENCRYPTED          eY/fQXw7Ure8Qrz7  -
VFW_CUSTOMER_USERNAME                    admin             -

[ok][2015-12-09 13:56:06]
admin@ncs-vm> show configuration globals provider vms-lab-pod variable VR_CUSTOMER_* | tab

                                                                                   ENCRYPTED
NAME                                   VALUE                                       VAL
---------------------------------------------------------------------------------------------
-
VR_CUSTOMER_DNS_1                      10.0.100.1                                  -
VR_CUSTOMER_DNS_2                      10.0.101.1                                  -
VR_CUSTOMER_DOMAIN_NAME                cisco.com                                   -
VR_CUSTOMER_PASSWORD_ENCRYPTED         ${DS}1${DS}.GF9${DS}/b8EVjeqXFdBcezO0PgMa/  -
VR_CUSTOMER_SECRET_PASSWORD_ENCRYPTED  ${DS}1${DS}PpaX${DS}eq5B91f7zp.8gn9h2RJi20  -
VR_CUSTOMER_USERNAME                   admin                                       -

[ok][2015-12-09 13:56:24]
```

Note that the passwords entered are pre-encrypted by the VNF devices. If the pre-encrypted password contains the "$" character, this needs to be followed by the string "{DS}" to prevent the "$" being interpreted as a variable by the NSO substitution engine. For example, the IOS encrypted password string of **thi$i$encr.&** would become **thi${DS}i${DS}encr.&** before being set in NSO global variables.

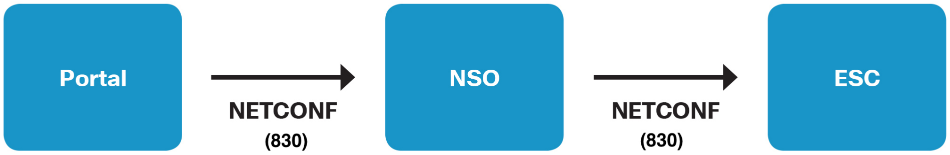
In order for the NSO to be able to successfully authenticate against the VNFs, the same username and password that is set in the global variables above need to be configured in the device authgroup. Each VNF type has its own authgroup, but these passwords should be entered in the non-encrypted format and the NSO will execute its own hash to obfuscate the password in the configuration.

```
admin@ncs-vm> show configuration devices authgroups group csr
default-map {
    remote-name               admin;
    remote-password           $4$drvlL0+wRQ+0Rs4icifw4A==;
    remote-secondary-password $4$drvlL0+wRQ+0Rs4icifw4A==;
}
[ok][2015-12-09 14:04:47]
admin@ncs-vm> show configuration devices authgroups group asa
default-map {
    remote-name               admin;
    remote-password           $4$drvlL0+wRQ+0Rs4icifw4A==;
    remote-secondary-password $4$drvlL0+wRQ+0Rs4icifw4A==;
}
[ok][2015-12-09 14:04:53]
admin@ncs-vm> show configuration devices authgroups group wsa
default-map {
    remote-name     admin;
    remote-password $4$e6n7A1Pkc17RLYhNGn2T0pk2oMA9yd7XaQ9+j7/Ctwo=;
}
[ok][2015-12-09 15:48:57]
```

NETCONF Authentication

Within the vMS solution, both NCS and ESC provide a NETCONF interface which must be authenticated. The NETCONF protocol itself leverages SSH to provide secure transport, and, therefore provides the same authentication mechanisms (password, publickey). Password authentication is used between the Portal and the NSO and the NSO and ESC.

Figure 8.1 NETCONF Communication



In the Portal, the NETCONF client is configured with a username and password as specified in the NSO AAA authentication configuration as highlighted below.

```
admin@ncs-vm> show configuration aaa authentication users
user admin {
    uid        1000;
    gid        1000;
    password   $1$6bNdL5Nf$Gt0ESHpWTOWcNadSyJTKR1;
    ssh_keydir /var/ncs/homes/admin/.ssh;
    homedir    /var/ncs/homes/admin;
}
user oper {
    uid        1000;
    gid        1000;
    password   $1$mssL.Xb8$cE1y6M9b2x34OodNGhEy4.;
    ssh_keydir /var/ncs/homes/oper/.ssh;
    homedir    /var/ncs/homes/oper;
}
user private {
    uid        1000;
    gid        1000;
    password   $1$1p26x7nK$UryplT/WAWRXm8pdoCQP91;
    ssh_keydir /var/ncs/homes/private/.ssh;
    homedir    /var/ncs/homes/private;
}
user public {
    uid        1000;
    gid        1000;
    password   $1$n0WhMG2P$vXYT.hIxfNCJSizKOYurB/;
    ssh_keydir /var/ncs/homes/public/.ssh;
    homedir    /var/ncs/homes/public;
}
[ok][2015-12-09 01:40:00]
```

This connection can be tested from the portal using SSH on port 830 and the configured username and password.

```
ssh -p 830 <portal-user>@<ncs-ip> -s netconf
```

The result should be that NSO returns a list of NETCONF capabilities which confirms that SSH session (and hence NETCONF) has been established.

It is best practice to restrict the source IP address range for port 830 on NSO using IPtables as described earlier in this chapter.

In order for NSO to make service action requests, the ESC also provides a NETCONF interface on port 830. In NSO, the credentials to be

```
  admin@ncs-vm> show configuration devices authgroups group esc
 used when connecting to the ESC are specified in a device authgroup that is referenced under
the ESC device declaration. This NSO configuration is illustrated below.  default-map {
      remote-name     admin;
      remote-password $4$dLb7JU6Bdzy0Rs4icifw4A==;
  }
```

```
[ok][2015-12-09 01:45:54]
admin@ncs-vm> show configuration devices device esc authgroup
authgroup esc;
[ok][2015-12-09 01:45:58]
```

On the ESC, there needs to be a matching username and password in ConfD AAA authentication, in the same way NSO configures authentication for he NETCONF interface. On the ESC, ConfD can be accessed and viewed as shown below.

```
[localadmin@esc_pod-1 ~]$ /opt/confd/bin/confd_cli -u admin

admin connected from 10.0.101.1 using ssh on esc_pod-1
admin@esc_pod-1> show configuration aaa authentication
users {
    user admin {
        uid         9000;
        gid         100;
        password    $1$gXCL.Cnf$HLNsIKXlUanYiDVmHheoQ.;
        ssh_keydir /var/confd/homes/admin/.ssh;
        homedir     /var/confd/homes/admin;
    }
    user oper {
        uid         9000;
        gid         100;
        password    $1$bzEEREx0$aMY2a3iC.28DErNw.XFDh1;
        ssh_keydir /var/confd/homes/oper/.ssh;
        homedir     /var/confd/homes/oper;
    }
    user private {
        uid         9000;
        gid         100;
        password    $1$.K3NeO7A$GWip6YR.T0/jqdsdU.gQM1;
        ssh_keydir /var/confd/homes/private/.ssh;
        homedir     /var/confd/homes/private;
    }
    user public {
        uid         9000;
        gid         100;
        password    $1$a5/VgENL$n.JIirB8ghDS3HuZ6JDF90;
        ssh_keydir /var/confd/homes/public/.ssh;
        homedir     /var/confd/homes/public;
    }
}
[ok][2015-12-08 20:53:32]
```

This NETCONF interface can be verified from NSO in the same way as before.

```
ssh -p 830 <esc-user>@<esc-ip> -s netconf
```

The authentication credentials can be updated from NSO ncs_cli and ESC confd_cli configuration mode with the following command:

```
set aaa authentication users user admin password <new-password>
```

It is worth noting that NSO Device nodes provide and authenticate the NETCONF interface in the same way as the NSO Service node and ESC ConfD. Again, this will be specified on the NSO Service node as a cluster authgroup, along with the Device node IP address with the username and password set at the NSO Device node under the 'aaa authentication' configuration.

9. OpenStack Troubleshooting

9.1 Introduction

This chapter describes troubleshooting OpenStack in a vMS deployment. It is assumed that the operators of the vMS solution do not have administrator access to the OpenStack environment. This chapter includes the following topics:

- OpenStack Health Check - commands to verify the OpenStack project is healthy
- Common OpenStack Errors - common OpenStack issues that could impact service chain creation/operation

> **Note**: ESC is the only component of the vMS Solution that directly interacts with OpenStack. Please note that troubleshooting of OpenStack from ESC perspective is shown in the section "Troubleshooting ESC OpenStack related issues" in Chapter 6.7 " ESC Troubleshooting" . ESC logs contain all of the OpenStack API calls made and the result codes.

9.2 Health Check

In case there are failures between ESC and OpenStack, it is a good practice to manually check OpenStack health and resource availability for the OpenStack tenant hosting vMS. ESC makes use of the following OpenStack services:

- Keystone (Identity)
- Nova (Compute)
- Glance (Image)
- Cinder (Storage)
- Neutron (Network)

Manual health check

To verify that ESC is able to communicate with OpenStack, make sure that an OpenStack Neutron network and a subnet can be created, and that a VM can be booted and attached to the network from the ESC. When ESC is deployed, OpenStack credentials are installed in `/cisco/esc-config/openrc.configs/vim_openrc`. This file is linked to `/etc/profile.d/openrc.sh` which makes it automatically sourced upon login to ESC and available to all users upon login. If the OpenStack clients are able to use any of the commands to list data or create items, then authentication (keystone) is working. The other commands need to be verified by actually using the OpenStack python clients to create a network, subnet, and boot a VM. The Nova and Neutron OpenStack python clients are installed on ESC by default. Examples of OpenStack health check command lines are shown below.

1. Neutron - Create Network Test

Create test network using OpenStack neutron, as follows.

```
$ neutron net-create test_net
Created a new network:
+----------------+--------------------------------------+
| Field          | Value                                |
+----------------+--------------------------------------+
| admin_state_up | True                                 |
| id             | 80db94ad-5a77-4ac9-a8ec-8d6d99221826 |
| name           | test_net                             |
| shared         | False                                |
| status         | ACTIVE                               |
| subnets        |                                      |
| tenant_id      | 3098b55808e84484a4f8bab2160a41a7     |
+----------------+--------------------------------------+
```

2. Neutron - Create Subnet Test

Next step is to try to create a subnet using OpenStack Neutron, as shown below.

```
$ neutron subnet-create test_net 1.1.1.0/24
Created a new subnet:
+-------------------+------------------------------------------+
| Field             | Value                                    |
+-------------------+------------------------------------------+
| allocation_pools  | {"start": "1.1.1.2", "end": "1.1.1.254"} |
| cidr              | 1.1.1.0/24                               |
| dns_nameservers   |                                          |
| enable_dhcp       | True                                     |
| gateway_ip        | 1.1.1.1                                  |
| host_routes       |                                          |
| id                | d8127349-7c0a-4bb7-bfeb-9d9d0f35cc14     |
| ip_version        | 4                                        |
| ipv6_address_mode |                                          |
| ipv6_ra_mode      |                                          |
| name              |                                          |
| network_id        | 80db94ad-5a77-4ac9-a8ec-8d6d99221826     |
| tenant_id         | 3098b55808e84484a4f8bab2160a41a7         |
+-------------------+------------------------------------------+
```

3. Nova Boot Command Test and Verification

If previous Neutron commands are successful, the next step in troubleshooting potential OpenStack issues is to try to boot a VM using Nova boot command, as shown in the output below.

```
  $ nova boot --image 51c86b9d-b477-4d17-b05e-c247499103df --flavor 436576f6-4910-43fd-ace5-
83e207ba6620 --nic net-id=80db94ad-5a77-4ac9-a8ec-8d6d99221826 test_vm
  +--------------------------------------+-----------------------------------------------------+
  | Property                             | Value                                               |
  +--------------------------------------+-----------------------------------------------------+
  | OS-DCF:diskConfig                    | MANUAL                                              |
  | OS-EXT-AZ:availability_zone          | nova                                                |
  | OS-EXT-STS:power_state               | 0                                                   |
  | OS-EXT-STS:task_state                | scheduling                                          |
  | OS-EXT-STS:vm_state                  | building                                            |
  | OS-SRV-USG:launched_at               | -                                                   |
  | OS-SRV-USG:terminated_at             | -                                                   |
  | accessIPv4                           |                                                     |
  | accessIPv6                           |                                                     |
  | adminPass                            | xzG3uedffcqBx                                       |
  | config_drive                         |                                                     |
  | created                              | 2015-12-07T23:31:45Z                                 |
  | flavor                               | Micro-Small (436576f6-4910-43fd-ace5-83e207ba6620) |
  | hostId                               |                                                      |
  | id                                   | c2f54ac9-a485-4390-824e-65d93d996266                 |
  | image                                | RHEL-6 (51c86b9d-b477-4d17-b05e-c247499103df)       |
  | key_name                             | -                                                    |
  | metadata                             | {}                                                   |
  | name                                 | test_vm                                              |
  | os-extended-volumes:volumes_attached | []                                                   |
  | progress                             | 0                                                    |
  | security_groups                      | default                                              |
  | status                               | BUILD                                                |
  | tenant_id                            | 3098b55808e84484a4f8bab2160a41a7                     |
  | updated                              | 2015-12-07T23:31:45Z                                 |
  | user_id                              | openstack@cisco.com                                  |
  +--------------------------------------+------------------------------------------------------+

  $ nova list --name test_vm
  +--------------------------------------+---------+--------+------------+-------------+-------
-----------+
  | ID                                   | Name    | Status | Task State | Power State |
Networks         |
  +--------------------------------------+---------+--------+------------+-------------+-------
-----------+
  | c2f54ac9-a485-4390-824e-65d93d996266 | test_vm | ACTIVE | -          | Running     |
test_net=1.1.1.2 |
  +--------------------------------------+---------+--------+------------+-------------+-------
-----------+
```

When booting a VM using `nova boot`, it will cycle through various states until reaching "ACTIVE" status, so it may be necessary to repeat the `nova list` command several times. VMs typically reach "ACTIVE" status within less than one minute of being created but may occasionally take longer depending on system load and image size.

These 3 operations verify that:

- OpenStack APIs are reachable and working correctly
- ESC has the correct credentials for sending API calls to OpenStack
- OpenStack internal components operate and communicate correctly
- There are sufficient resources on the target tenant to create all the objects

> **Note**: Please note that if the current usage of certain resource is close to the limit, the manual test shown above might work fine but actual ESC deployment still might fail because it needs to create more objects (ex. Neutron ports) for the whole deployment

Nova quotas and current usage

Command for checking Nova limits and usage is `nova absolute-limits` which shows the maximum value and current usage for each resource.

```
$ nova absolute-limits
+--------------------+-------+--------+
| Name               | Used  | Max    |
+--------------------+-------+--------+
| Cores              | 59    | 400    |
| FloatingIps        | 0     | 10     |
| ImageMeta          | -     | 128    |
| Instances          | 22    | 200    |
| Keypairs           | -     | 100    |
| Personality        | -     | 5      |
| Personality Size   | -     | 65535  |
| RAM                | 98304 | 819200 |
| SecurityGroupRules | -     | 20     |
| SecurityGroups     | 1     | 10     |
| Server Meta        | -     | 128    |
+--------------------+-------+--------+
```

If the ESC has an older Nova python client version, the output might be different and might need additional interpretation. For example, comparing `maxTotalCores` and `totalCoresUsed` will show the limit and current usage for vCPU resources. Here is a command output example:

```
$ nova absolute-limits
+-------------------------+-------+
| Name                    | Value |
+-------------------------+-------+
| maxServerMeta           | 128   |
| maxPersonality          | 5     |
| maxImageMeta            | 128   |
| maxPersonalitySize      | 10240 |
| maxTotalRAMSize         | 51200 |
| maxSecurityGroupRules   | 20    |
| maxTotalKeypairs        | 100   |
| totalRAMUsed            | 512   |
| maxSecurityGroups        | 10    |
| totalFloatingIpsUsed     | 0     |
| totalInstancesUsed       | 1     |
| totalSecurityGroupsUsed  | 1     |
| maxTotalFloatingIps      | 10    |
| maxTotalInstances        | 10    |
| totalCoresUsed           | 1     |
| maxTotalCores            | 20    |
+--------------------------+-------+
```

Neutron quotas and current usage

Command for checking resource quota on Neutron is `neutron quota-show`.

```
$ neutron quota-show
+---------------------+-------+
| Field               | Value |
+---------------------+-------+
| floatingip          | 12    |
| health_monitor      | -1    |
| member              | -1    |
| network             | 30    |
| pool                | 10    |
| port                | -1    |
| router              | 2     |
| security_group      | 20    |
| security_group_rule | 100   |
| subnet              | 30    |
| vip                 | 10    |
+---------------------+-------+
```

Checking current Neutron resources usage against the quotas will be more challenging. The easiest would be to run a `-list` command with some filtering to display the current count of specific resources in the tenant. For example, to display a current count of Neutron networks inside a tenant, the command would be:

```
$ neutron net-list | egrep -v "+---| id" -A 0 | wc -l
19
```

The displayed number (19 in this example) should be compared with the corresponding resource limit from the previously shown `neutron quota-show` to make sure we are not exhausting the resources assigned to the tenant. Other resources (subnets, ports…) can be checked in a similar way.

9.3 Common Errors and Resolutions

These are some of the most common OpenStack failure scenarios that are relevant for ESC:

- API connection issue or API failure
- Network or Subnet create failure
- VNF boot failure

> ! **Note**: ESC is the only component of the vMS Solution that directly interacts with OpenStack. Please note that troubleshooting of OpenStack from ESC perspective is shown in section "Troubleshooting ESC OpenStack related issues" in Chapter 6.7 "ESC Troubleshooting". ESC logs contain all of the OpenStack API calls and result codes that are made by vMS.

API connection issue or API failure

The following are the most common reasons why API might not be available to ESC:

Network connectivity

This relates to connectivity issues between ESC VM and host/VM/LXC where a particular API is running. These can be detected with standard networking troubleshooting tools and steps. Resolution depends on the particular issue.

Incorrect credentials

Incorrect credentials will prevent ESC from making any API calls. The credentials can be verified by executing `printenv | grep OS_` command on ESC shell. If some of the parameters are wrong they can be corrected by editing `/cisco/openrc` file. Restart `esc_service` for the changes to take effect.

API not responding

If there is an internal OpenStack issue, the ESC API requests may not get answered at all, or might get an answer with 5xx status code. Troubleshooting and resolution need to be done by the OpenStack support team.

Network or Subnet create failure

The following are most common reasons why ESC might not be able to create a network or a subnet on OpenStack:

Internal OpenStack issues

If there is an internal OpenStack issue, the ESC will not be able to create the object and instead it will get an answer with 5xx status code. Troubleshooting and resolution need to be done by the OpenStack support team.

Duplicate network-CIDR pair

As a result of incomplete ESC un-deployment of service chains, OpenStack might retain network and CIDR definition pairs. If ESC tries to re-deploy the same service chain with the same network and CIDR pair, the OpenStack will error out in the following way:

```
  Invalid input for operation: Requested subnet with cidr: 10.20.0.0/24 for network: c2cdbeb8-
60f6-4740-af9b-7aa95463c880 overlaps with another subnet.
```

Troubleshooting and resolution steps consist of manually going through OpenStack outputs and manually deleting overlapping networks.

Neutron resource exhaustion

In the event that the OpenStack tenant runs out of quota for the network, subnet and port resources, it will be impossible for ESC to create those resources via the Neutron API and it will return status code 403. Troubleshooting steps are to verify the existing quota and current resource usage as explained in the previous Chapter 9.2 "OpenStack Health Check".

To resolve the issue, release unused resources or ask the OpenStack administrator to increase quota.

VNF boot failure

Internal OpenStack issues

If there is an internal OpenStack issue, ESC will not be able to boot the VNF instance. The VNF instance status can be checked with `nova list [--name]`, like shown below.

```
  $ nova list
  +--------------------------------------+--------------+--------+------------+-------------+--
-------------------------------+
  | ID                                   | Name         | Status | Task State | Power State |
Networks                         |
  +--------------------------------------+--------------+--------+------------+-------------+--
-------------------------------+
  | 913145d9-c8b8-4f6e-8d13-9f469a788e0e | Test123      | ACTIVE | -          | Running     |
BR-INSIDE-02-cloudvpn=10.20.0.4  |
  | b69bed4a-d342-45fc-8478-a05077d6ef4d | Test124      | ACTIVE | -          | Running     |
BR-INSIDE-01-cloudvpn=10.20.5.5 |
  | 023522f1-16a2-42ec-adf3-4cfba9621fac | test-cirros1 | ERROR  | -          | NOSTATE     |
private-direct=10.35.120.31      |
  | b270a57a-de6f-46ca-b44d-f6449b33e4d0 | test-cirros2 | ACTIVE | -          | Running     |
private-direct=10.35.120.32      |
  | c2f54ac9-a485-4390-824e-65d93d996266 | test_vm      | ACTIVE | -          | Running     |
testing=1.1.1.2                  |
  +--------------------------------------+--------------+--------+------------+-------------+--
--------------------------------+
```

If the status is "ERROR" or Task State is "SPAWNING" for very long time (depending on the system load), further debugging needs to be done to determine the reason.

The next step is usually doing a `nova show <instance_id>` command.

```
  $ nova show 023522f1-16a2-42ec-adf3-4cfba9621fac
  +--------------------------------------+----------------------------------------------------
------------------------------------+
  | Property                             | Value
|
  +--------------------------------------+----------------------------------------------------
------------------------------------+
  | private-direct network               | 10.35.120.31
|
  | OS-DCF:diskConfig                    | MANUAL
|
  | OS-EXT-AZ:availability_zone          | nova
|
  | OS-EXT-STS:power_state               | 0
|
  | OS-EXT-STS:task_state                | -
|
  | OS-EXT-STS:vm_state                  | error
|
  | OS-SRV-USG:launched_at               | 2015-12-02T04:38:52.000000
|
  | OS-SRV-USG:terminated_at             | -
|
  | VNF_Management_Prod network          | 10.60.79.85
|
  | accessIPv4                           |
|
  | accessIPv6                           |
|
  | config_drive                         | True
|
  | created                              | 2015-12-02T04:30:21Z
|
  | fault                                | {"message": "No valid host was found", "code": 500,
"created": "2015-12-04T16:44:52Z"} |
  | flavor                               | m1.tiny (c6f5f979-7a4f-4878-aad5-cbd9c64a9644)
|
  | hostId                               |
8d00f8bf665906e0b832ed52432d8873b825d34a3ba6d90b0c2cda4d                              |
  | id                                   | 023522f1-16a2-42ec-adf3-4cfba9621fac
|
  | image                                | cirros (1c6ff88e-0d19-4dfd-ae8b-c0af4aa325cc)
|
  | key_name                             | -
|
  | metadata                             | {}
|
  | name                                 | test-cirros1
|
  | os-extended-volumes:volumes_attached | []
|
  | status                               | ERROR
|
  | tenant_id                            | 3098b55808e84484a4f8bab2160a41a7
|
  | updated                              | 2015-12-04T16:48:15Z
|
  | user_id                              | openstack@cisco.com
|
  +--------------------------------------+----------------------------------------------------
-----------------------------------+
```

The fault field should show more information about why the VNF instance failed to boot. Troubleshooting and resolution usually need to be done by the OpenStack support team/department.

Corrupted Glance image upload

Another potential reason for the "No valid host was found" error message shown above, could be that the image uploaded to OpenStack Glance service is corrupted. In that case, next troubleshooting step is to get the checksum from Glance metadata and compare it to the MD5 hash of the image. ESC at this time doesn't have the Glance python client pre-installed, so this operation needs to be done from another host where the Glance python client is available. Sample output is provided below.

```
  $ glance image-show 693ad7e3-5eac-4b65-b2a9-b2a5394cb49c
  +------------------+-------------------------------------------------------------------------
---------+
  | Property         | Value
|
  +------------------+-------------------------------------------------------------------------
---------+
  | checksum         | 9fac98bc4f10daabf36273fdd5b7452b
|
  | container_format | bare
|
  | created_at       | 2015-11-12T21:30:28Z
|
  | direct_url       | rbd://51aa0523-408e-4386-96b3-734500906e80/csx-a-aio-glance-image-
1/693ad7e3     |
  |                  | -5eac-4b65-b2a9-b2a5394cb49c/snap
|
  | disk_format      | qcow2
|
  | id               | 693ad7e3-5eac-4b65-b2a9-b2a5394cb49c
|
  | min_disk         | 0
|
  | min_ram          | 0
|
  | name             | asav951-201.qcow2
|
  | os_distro        | Unknown
|
  | os_name          | Unknown
|
  | os_version       | Unknown
|
  | owner            | 3b751f71ea7342f2a7e791f6531e0311
|
  | protected        | False
|
  | size             | 160038912
|
  | status           | active
|
  | tags             | []
|
  | updated_at       | 2015-11-13T07:03:35Z
|
  | visibility       | private
|
  +------------------+-----------------------------------------------------------------------
---------+
```

IP address pool exhaustion

In the event that the subnet IP address pool is exhausted, ESC will not be able to create a Neutron port for the VNF VM. OpenStack will return the following error:

```
  No more IP addresses available on network a43c77be-4c02-4eef-ad0a-953d102b5bb1.
```

To resolve this issue, we should release unused resources or ask the OpenStack admin to add another subnet or increase the allocation pool.

Tenant resource exhaustion

In the event that the OpenStack tenant runs out of quota for network and compute resources, the ESC will not be able to create instances. Troubleshooting steps are to verify the existing quota and current resource usage for network and compute resources as explained in the previous Chapter 9.2 "OpenStack Health Check". To resolve the issue, we should release unused resources or ask the OpenStack admin to increase quota.

10. Service Chain Troubleshooting

10.1 Introduction

This chapter provides details on vMS CloudVPN Service Chain Troubleshooting, and consists of the following sections:

- General Troubleshooting - this section describes steps that can be used for verifying Service Chain operation and end-to-end service verification procedures.
- CSR1000v - this section describes features, basic configuration, and troubleshooting of the Cisco Cloud Services Router (CSR1000v) in the context of the CloudVPNService Chain
- ASAv - this section describes features, basic configuration, and troubleshooting of the Cisco Adaptive Security Appliance (ASAv) Firewall in the context of the CloudVPN Service Chain
- WSAv - this section describes Cisco Web Security Appliance (WSAv) features, basic configuration, and troubleshooting of WSAv Web Proxy in the context of the CloudVPN Service Chain
- Common Issues - this section describes solutions and troubleshooting steps for some common issues that can be reported by the end customers.

10.2 General Troubleshooting

This chapter covers troubleshooting aspects of the end-end Service Chain. Please refer to the individual component sections for more details.

Service Chain Details

There are three service chain offerings as described in Chapter 3.2 "CloudVPN Service". The service chain offerings are: Basic, Medium and Full.

Basic Service

The Basic service offering consists of a CSR1000v and two or more CPEs. Site-to-site connectivity is established through the hub CSR1000v. Public Internet access is routed locally via each CPE. The output below shows how a fully provisioned Basic Service Chain with two CPEs appears in the NSO when it is ready for use.

```
admin@ncs-vm> show cloudvpn-data basic-cvpn device-list
DEVICE               REMOTE  READY           PROVISIONED
----------------------------------------------------------
basic-cvpn-CSR-esc   local   ready           true
cpe-9N357LLRDSG      local   ready           true
cpe-9N260SLJSKO      local   ready           true
```

Medium Service

The Medium service offering consists of a CSR1000v, ASAv and one or more CPEs. Site-to-site and public Internet traffic is directed from each CPE to the CSR1000v, and Internet-bound traffic is then sent through the ASAv which provides Firewall service. The Medium profile allows for SSL-VPN remote access users to connect to the service chain (ASAv) by use of an Anyconnect VPN client. The output below shows how a fully provisioned Medium Service Chain with two CPEs appears in the NSO when it is ready for use.

```
admin@ncs-vm> show cloudvpn-data medium-cvpn device-list
DEVICE               REMOTE  READY           PROVISIONED
----------------------------------------------------------
medium-cvpn-ASA-esc   local   ready           true
medium-cvpn-CSR-esc   local   ready           true
cpe-9N357LLRDSG       local   ready           true
cpe-9N260SLJSKO       local   ready           true
```

Full Service

The Full service offering adds a WSAv to the Medium service offering. The output below shows how a fully provisioned Full Service Chain with two CPEs appears in the NSO when ready for use.

```
admin@ncs-vm> show cloudvpn-data full-cvpn device-list
DEVICE               REMOTE  READY           PROVISIONED
----------------------------------------------------------
full-cvpn-ASA-esc     local   ready           true
full-cvpn-CSR-esc     local   ready           true
full-cvpn-WSA-esc     local   ready           true
cpe-9N357LLRDSG       local   ready           true
cpe-9N260SLJSKO       local   ready           true
```

For further details about Service Chain topologies and functionalities, please refer to Chapter 3.2 "CloudVPN Service".

Orchestration Verification

Once a Service Chain is provisioned, NSO can be used to verify if it has been successfully deployed. The output of the command below can be used to confirm if all of the components of the service are created, configured and ready.

Note: the example below shows a service chain with profile Full, with a CSR1000v, an ASAv, a WSAv and a single CPE. A Basic profile will only have a CSR1000v and one or more CPEs, while a Medium profile will have a CSR1000v, an ASAv and one or more CPEs

```
admin@ncs-vm> show cloudvpn-data my-new-cvpn device-list
DEVICE               REMOTE  READY           PROVISIONED
----------------------------------------------------------
my-new-cvpn-ASA-esc  local   ready           true
my-new-cvpn-CSR-esc  local   ready           true
my-new-cvpn-WSA-esc  local   ready           true
cpe-9N989LLRWSN      local   ready           true
```

For a VNF, the READY column displays "ready" once the NSO has received a VM_ALIVE message from ESC. If a VNF continues to display a status of "not-existing" for more than 20 minutes, refer to Chapters 5.8 "NSO Troubleshooting" and 6.7 "ESC Troubleshooting" and specifically look at ESC log `/var/log/esc/yangesc.log` for additional details.

For CPE on-boarding, "ready" will be displayed once the PnP server has successfully pushed the Day 0 configuration. If "not-existing" is displayed for a CPE, this could simply indicate that the CPE has been registered with the a service through the Portal but the physical device has not yet been connected to the Internet or contacted the NSO PnP service. The output of `show pnp-state device <serial number>` can be helpful. It should show "configured", "added", and "synced" each as "true." Refer to Chapter 7.8 "CPE Troubleshooting" for additional details.

The output above the PROVISIONED column will show "true" when the Day 1/Day 2 configuration has been successfully added to the devices by NSO. It is worth noting that the CSR1000v will only move to the "true" status after a CPE has been successfully provisioned in the Service Chain.

In the event that a device does not reach the provisioned true status, the first step would be to ensure that NSO is able to connect and login to the device via SSH. If NSO can successfully connect to the device, the individual device NED traces located under `/var/log/ncs`, or the Java VM log in `/var/log/ncs/ncs-java-vm.log`, may provide more information indicating why the device could not be configured. The same log will also show connection attempts with information about success and failure.

It is possible to request NSO to connect to a device with the command below:

```
admin@ncs-vm> request devices device my-first-cvpn-CSR-esc connect
result true
info (admin) Connected to my-first-cvpn-CSR-esc - 192.168.14.4:22
[ok][2015-12-10 01:06:52]
```

Once a service chain is fully provisioned, operational status information and traffic statistics can be displayed by using the following command:

```
admin@ncs-vm> show cloudvpn-data my-new-cvpn oper-data
oper-data oper-state down
oper-data internet_traffic 69209
oper-data onnet_traffic 14519
oper-data connected_users 0
oper-data S2S oper-state up
oper-data S2S inBytes 7068
oper-data S2S outBytes 7451
                   DEVICE  CONNECTION  IN     OUT    IN       OUT
ID                 STATE   STATE       BYTES  BYTES  PACKETS  PACKETS
-----------------------------------------------------------------------
cpe_1444911212048  up      up          2562   2497   34       33

oper-data RA oper-state up
oper-data RA inBytes 0
oper-data RA outBytes 0
oper-data FW-INET oper-state up
oper-data FW-INET inBytes 33189
oper-data FW-INET outBytes 36020
```

```
  oper-data WSEC
                              ACTIVE  AVG         PEAK
                       OPER   IPV4    THROUGHPUT  THROUGHPUT                    OPER   IN
OUT      IN       OUT      TUNNEL  OPER    IN      OUT     IN       OUT
  ID                   STATE  NATS    1HR         1D          IFINDEX  IFDESCR  STATE  BYTES
BYTES    PACKETS  PACKETS  INDEX   STATE   BYTES   BYTES   PACKETS  PACKETS
  ---------------------------------------------------------------------------------------------
----------------------------------------------------------------------------
  my-new-cvpn-CSR-esc  up     -       -           -           1        Gi1      up     430231
705157   5254     6890
                                                              2        Gi2      up     7336
7719     919      51
                                                              3        Gi3      up     524767
12267    8873     157
                                                              4        Vo0      up     0
0        0        0
                                                              5        Nu0      up     0
0        0        0
                                                              6        Lo1      up     0
0        0        0
                                                              7        Vt1      down   0
0        0        0
                                                              8        Vi1      up     -
-        35       36       1       up      1944    1989    35       36
  my-new-cvpn-ASA-esc  up     -       -           -
  my-new-cvpn-WSA-esc  up     -       -           -
  cpe-9N989LLRWSN      up     -       -           -           1        Gi1      up     658757
2445419  5795     28930
                                                              2        Gi2      up     191214
1526391  1375     25875
                                                              3        Vo0      up     0
0        0        0
                                                              4        Nu0      up     0
0        0        0
                                                              5        Tu0      up     192461
563361   2743     5074
                                                              6        Tu1      up     2709
2644     36       35       1       up      249972  462181  2757     5078

2       up     3864    2051    38       37

  [ok][2015-12-08 23:25:13]
```

The above output provides useful information on data-plane traffic showing increasing packet counts on various interfaces. The following sections will show how to check the service chain data plane further.

Finding VNF or CPE Management IP addresses

During the course of troubleshooting, it is sometimes helpful to log directly into the device. The command below can be used to discover the management IP address assigned to a VNF or a CPE. These are the same IP addresses used by NSO to SSH to the devices when making configuration updates. Authentication credentials that the NSO uses to connect to the VNFs are set in device authgroups and determined by the setting in the per provider global variables (see vMS Operations and Administration - Security Considerations). For CPEs, it is possible to enable unique usernames and passwords. Please refer to the Chapter 7 "Customer Premise Equipment (CPE)".

```
admin@ncs-vm> show devices list
NAME                    ADDRESS        DESCRIPTION  NED ID     ADMIN STATE
--------------------------------------------------------------------------
cpe-9N989LLRWSN         10.254.0.8     -            cisco-ios  unlocked
cpe-9YLMGXIQ0ON         10.254.0.9     -            cisco-ios  unlocked
esc                     192.168.15.20  -            netconf    unlocked
my-first-cvpn-ASA-esc   192.168.14.41  -            cisco-asa  unlocked
my-first-cvpn-CSR-esc   192.168.14.4   -            cisco-ios  unlocked
my-new-cvpn-ASA-esc     192.168.14.44  -            cisco-asa  unlocked
my-new-cvpn-CSR-esc     192.168.14.42  -            cisco-ios  unlocked
my-new-cvpn-WSA-esc     192.168.14.43  -            ciscowsa   unlocked
[ok][2015-12-09 00:03:08]
```

When the NSO is configured for unique CPE passwords, use the command `show pnp-state device <serial>` to find the username and password for an individual CPE device.

```
admin@ncs-vm> show pnp-state device 9N989LLRWSN
pnp-state device 9N989LLRWSN
 udi            PID:ISR881,VID:V00,SN:9N989LLRWSN
 device-info    15.6(20150712:154917)
 ip-address     192.168.11.14
 mgmt-ip        10.254.0.8
 port           22
 name           cpe-9N989LLRWSN
 username       admin
 password       cisco123
 sec-password   cisco123
 salt           ABCD
 remote-node    ""
 wan-interface GigabitEthernet1
 lan-interface GigabitEthernet2
 configured     true
 request        backoff
 added          true
 synced         true
```

```
    is-netsim     false
    need-clean    false
    pending-exec  ""
    last-contact  2015-12-11 02:47:20
    last-clean    0
  [ok][2015-12-11 02:50:29]
  admin@ncs-vm>
```

Service Chain Verification

One approach to Service Chain verification is to verify each step of the service orchestration in Figure 2.6 "Service Chain Creation Detailed Call Flow" in Chapter 2.4 "vMS Call Flows".

Service Configuration Request

A new configuration request for a cloudvpn service is accepted by NSO. This can be seen in the NSO configuration using the command `show configuration cloudvpn <service-chain-name>`. This service configuration may be sent from an external system (e.g. an end customer portal) and received via the northbound NETCONF interface of NSO. The log for the northbound NETCONF interface can be found in `/var/log/ncs/netconf-north.log`.

This milestone is successfully achieved if the service chain configuration as seen in NSO `show configuration cloudvpn <service-chain-name>` is complete and correct.

If the inbound requests are not seen in this log, then continue troubleshooting on the Portal components. If the inbound requests are found but actual NSO configuration differs from the requested configuration, then continue investigation on NSO.

VNF Deployment

Once NSO has committed the cloudvpn configuration, a request will be made to ESC to spin up the VNFs required by the service chain. As the VNFs are created, they will be visible in the OpenStack virtualization infrastructure and can be seen using `nova list` command. Deployment progress can be monitored as ESC will set various lifecycle status flags, initially VM_DEPLOYED, sending notifications upstream to NSO. These flags and notifications can be correlated in the log files on the ESC (`/var/log/esc/yangesc.log`) and NSO Java VM log (`/var/log/ncs/ncs-java-vm.log`).

This milestone is successfully achieved if NSO has received a VM_DEPLOYED-SUCCESS message for each VNF.

If ESC fails to deploy any of the VNFs, this will be indicated in the `yangesc.log` file along with a summary description of the failure. More detailed logs of VNF deployment can be found in

`/var/log/ecs/escmanager.log`. Common reasons why ESC might fail to deploy a VNF can be found in Chapter 10.6 "Service Chain Common Issues".

If ESC has successfully sent a notification but the NSO didn't receive the notification, refer to section "Re-establish connection with ESC" in Chapter 5.8 "NSO Troubleshooting".

VNF reachability

When each VNF is reachable via its management interface, ESC will set VM_ALIVE. When all VNFs reach VM_ALIVE, ESC will set SERVICE_ALIVE state. As each VNF reaches the VM_ALIVE and the notification is received, NSO will attempt to connect via the device's management interface.

This milestone is successfully achieved if NSO has received a VM_ALIVE-SUCCESS message for each VNF, and a SERVICE_ALIVE-SUCCESS message. Some VNF types may take a considerable amount of time to become VM_ALIVE: Depending on VNF type, this may take between five and 20 minutes.

ESC will continue to monitor VNFs and attempt to recover them as needed, but if this milestone is achieved, and assuming VNFs remain up and reachable, then this concludes ESC's involvement in the provisioning process. All remaining device configuration is done by NSO.

VNF provisioning

After a VNF has reached VM_ALIVE state, NSO will provision the VNF with a Day 1/Day 2 configuration. **The one exception is the CSR1000v, which will not be provisioned until after at least one CPE is alive.**

This milestone is successfully achieved if the "PROVISIONED" column in `show cloudvpn-data <> device-list` shows "true" for each VNF.

Upon reaching this milestone, a "Provisioned" notification will be sent upstream to the portal, and the Service Chain status should change from "Provisioning" to "Provisioned" in vMS Portal.

If one or more VNFs are not being provisioned:

- In case the CSR1000v is not being provisioned, ensure that there is at least one CPE on-boarded.
- Verify end-to-end IP connectivity from NCS by attempting both ping and SSH from NCS to the VNF.
- Verify that the VNF has remained alive (i.e. that there have not been any VM_UNDEPLOYED or RECOVERY events for the VNF).
- Troubleshoot as per Chapter 5.8 "NSO Troubleshooting", specifically using ncs-java-vm and NED logs.

CPE on-boarding

For a service to be fully operational, each service chain requires at least one CPE. CPE on-boarding starts as soon as the CPE Serial Number is added to the service chain configuration in NSO (seee "Service Configuration Request" above) and may happen in parallel with VNF provisioning.

This milestone is successfully achieved if the "provisioned" column in `show cloudvpn-data <> device-list` shows "true" for each CPE.

If one or more CPEs are not being provisioned, refer to Chapter 7.8 "CPE Troubleshooting".

End-to-End Connectivity

After all VNFs and CPEs have been provisioned with their intended Day 1/Day 2 configuration, end-users should have site-to-site and site-to-Internet connectivity through their service chain.

This milestone is successfully achieved if at least one device behind each CPE's LAN interface has reachability to at least one other site, and to the public Internet, and configured features such as SSL-VPN remote access users are verified as working.

If end-to-end connectivity is not functioning or partially functioning, refer to the below section "End-to-end Connectivity Verification".

[end-to-end-connectivity-verification]End-to-end Connectivity Verification

After having verified that from the orchestration point of view the service chain is correctly provisioned, the end-to-end connectivity of the service data plane can be verified by connecting to the various components and performing the following checks.

CPE Checks

From the CPE verify that the FlexVPN tunnel interface is up:

```
router#show crypto session interface tunnel1
Crypto session current status

Interface: Tunnel1
Profile: FlexVpnToCSR1
Session status: UP-ACTIVE
Peer: 20.0.0.10 port 500
  Session ID: 2
  IKEv2 SA: local 192.168.11.14/500 remote 20.0.0.10/500 Active
  IPSEC FLOW: permit 47 host 192.168.11.14 host 20.0.0.10
        Active SAs: 2, origin: crypto map
```

Verify that traffic counters on the CPE are incrementing for Tunnel1:

```
router#show int Tunnel1
Tunnel1 is up, line protocol is up
  Hardware is Tunnel
```

```
    Description: tunnel to CSR active
    Internet address is 10.0.0.2/24
    MTU 9934 bytes, BW 100 Kbit/sec, DLY 50000 usec,
       reliability 255/255, txload 1/255, rxload 1/255
    Encapsulation TUNNEL, loopback not set
    Keepalive not set
    Tunnel linestate evaluation up
    Tunnel source 192.168.11.14 (GigabitEthernet1), destination 20.0.0.10
     Tunnel Subblocks:
        src-track:
           Tunnel1 source tracking subblock associated with GigabitEthernet1
            Set of tunnels with source GigabitEthernet1, 2 members (includes iterators), on
interface <OK>
    Tunnel protocol/transport GRE/IP
      Key disabled, sequencing disabled
      Checksumming of packets disabled
    Tunnel TTL 255, Fast tunneling enabled
    Tunnel transport MTU 1434 bytes
    Tunnel transmit bandwidth 8000 (kbps)
    Tunnel receive bandwidth 8000 (kbps)
    Tunnel protection via IPSec (profile "IpsecToCSR1")
    Last input never, output never, output hang never
    Last clearing of "show interface" counters 02:02:50
    Input queue: 0/375/0/0 (size/max/drops/flushes); Total output drops: 0
    Queueing strategy: fifo
    Output queue: 0/0 (size/max)
    5 minute input rate 0 bits/sec, 0 packets/sec
    5 minute output rate 0 bits/sec, 0 packets/sec
       1556 packets input, 114723 bytes, 0 no buffer
       Received 0 broadcasts (0 IP multicasts)
       0 runts, 0 giants, 0 throttles
       0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
       1559 packets output, 115163 bytes, 0 underruns
       0 output errors, 0 collisions, 0 interface resets
       0 unknown protocol drops
       0 output buffer failures, 0 output buffers swapped out
```

Note: Refer to Chapters 10.3 "CSR1000v Troubleshooting" and 7.8 "CPE Troubleshooting" for further information and troubleshooting steps for FlexVPN tunnels.

Verify that the BGP session is up over Tunnel1 toward the CSR1000v:

```
router#show ip bgp vpnv4 vrf IVRF summary
BGP router identifier 192.168.11.14, local AS number 65000
BGP table version is 4, main routing table version 4
3 network entries using 768 bytes of memory
3 path entries using 360 bytes of memory
5/3 BGP path/bestpath attribute entries using 1360 bytes of memory
2 BGP AS-PATH entries using 64 bytes of memory
1 BGP extended community entries using 24 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 2576 total bytes of memory
BGP activity 3/0 prefixes, 3/0 paths, scan interval 60 secs

Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
10.0.0.254      4        65009     705     705        4    0    0 01:51:30        2
```

The above outputs show that the BGP session towards the CSR is up and 2 prefixes (2 other remote VPN sites) are received.
General BGP troubleshooting related information can be found in "Troubleshooting BGP", http://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/22166-bgp-trouble-main.html.

Verify that the CPE can reach the Internet by pinging a known public IP address. It is important here to use data VRF and source traffic from the LAN interface:

```
router#ping vrf IVRF 8.8.8.8 source Gi2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 8.8.8.8, timeout is 2 seconds:
!!!!!
Success rate is 0 percent (5/5)
```

Verify that WCCP is actively redirecting traffic to the WSAv (cloudvpn Full profile only)

On ASAv check if the number of cache engines and routers is 1, and WCCP status is Usable, as shown in following ASAv outputs.

```
my-new-cvpn-asa#show wccp 90

Global WCCP information:
    Router information:
        Router Identifier:                   192.168.14.44
        Protocol Version:                    2.0
    Service Identifier: 90
        Number of Cache Engines:             1
        Number of routers:                   1
        Total Packets Redirected:            0
        Redirect access-list:                wccp-redirect
        Total Connections Denied Redirect:   0
        Total Packets Unassigned:            0
        Group access-list:                   -none-
        Total Messages Denied to Group:      0
        Total Authentication failures:       0
        Total Bypassed Packets Received:     0
```

```
my-new-cvpn-asa# show wccp 90 detail

WCCP Cache-Engine information:
 Web Cache ID:          10.192.1.3
    Protocol Version:      2.0
   State:                 Usable
   Initial Hash Info:     00000000000000000000000000000000
                         00000000000000000000000000000000
  Assigned Hash Info:    FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
                         FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
  Hash Allotment:        256 (100.00%)
 Packets Redirected:    0
 Connect Time:          2d09h
```

10.3 CSR1000v

Overview

CSR1000v is a virtualized router. The vMS solution makes use of the following core features on CSR1000v:

- FlexVPN
- Routing
- QoS

Configuration of these features is deployed in three stages:

- Day 0 configuration - Contains basic device setting and IP connectivity that is added at device boot time via ESC. The Day 0 template contains minimal configuration to configure the device and bring it into management (SSH login). The template for Day 0 configuration is located on NSO at `/var/opt/ncs/state/packages-in-use/1/day0/cfg/`
- Day 1 and Day 2 configurations - NSO uses these templates to configure features directly on the device via SSH. XML templates are located on NSO at `/var/opt/ncs/state/packages-in-use/1/cloudvpn/templates/`

FlexVPN

FlexVPN is Cisco's implementation of the IKEv2 standard featuring a unified paradigm and CLI that combines site-to-site, remote access, hub and spoke topologies and partial meshes (spoke-to-spoke direct). FlexVPN offers a simple but modular framework that extensively uses the tunnel interface paradigm while remaining compatible with legacy VPN implementations using crypto maps.

Inside the vMS solution, FlexVPN is used to build IPsec tunnels between the CPEs (spokes) and CSR1000v (hub).

FlexVPN configuration on CSR1000v consists of several constructs.

```
crypto ikev2 authorization policy CPE-9N989LLRWSN.my-new-cvpn.cisco.com
 route set interface
 route set access-list InjectRouteToCPE
 route accept any distance 220
```

An IKEv2 authorization policy defines the local authorization policy and contains local and/or remote attributes.

```
crypto ikev2 keyring FlexVpnKeyring
 peer CPE-9N989LLRWSN.my-new-cvpn.cisco.com
  identity fqdn CPE-9N989LLRWSN.my-new-cvpn.cisco.com
  pre-shared-key local QUFNDfDc6Xv3zzj
  pre-shared-key remote JeH8aLhOK1U5ILh
```

An IKEv2 keyring contains sub-blocks for each peer that defines pre-shared keys and the identity that should be matched.

```
crypto ikev2 profile FlexVpnToCPE
 match fvrf any
 match identity remote fqdn domain my-new-cvpn.cisco.com
 identity local fqdn my-new-cvpn-CSR-esc.my-new-cvpn.cisco.com
 authentication remote pre-share
 authentication local pre-share
 keyring local FlexVpnKeyring
 aaa authorization group psk list default
 virtual-template 1
```

An IKEv2 profile is a repository of non-negotiable parameters of the IKE SA, such as local or remote identities, authentication methods and services that are available to authenticated peers that match the profile.

```
crypto ipsec transform-set ENCR esp-aes esp-sha-hmac
 mode transpor
```

IPsec transform set defines ciphers and encapsulation method.

```
crypto ipsec profile IpsecToCPE
 set transform-set ENCR
 set ikev2-profile FlexVpnToCPE
```

IPsec profile binds the transform-set and IKEv2 profile.

```
interface Virtual-Template1 type tunnel
 ip vrf forwarding IVRF
 ip unnumbered Loopback1
 tunnel protection ipsec profile IpsecToCPE
```

IPsec profile is applied to the Virtual-Template from which the Virtual-Access tunnels for CPEs are generated.

Routing

vMS solution uses BGP and OSPF routing protocols. BGP is used between the CPEs and CSR1000v to distribute routes to CPEs. OSPF is used to export all CPE local LAN routes to ASAv.

Routing configuration for CSR1000v is shown below.

```
router ospf 9 vrf IVRF
 router-id 10.0.0.254
 no log-adjacency-changes
 redistribute bgp 65009 subnets route-map BGPintoOSPF
 network 10.192.1.2 0.0.0.0 area 0
!
router bgp 65009
 bgp log-neighbor-changes
 timers bgp 10 30
 !
 address-family ipv4 vrf IVRF
  bgp router-id 10.0.0.254
  redistribute ospf 9 route-map OSPFintoBGP
  neighbor CPE peer-group
  neighbor CPE remote-as 65000
  neighbor CPE update-source Loopback1
  neighbor CPE next-hop-self
  neighbor CPE as-override
  neighbor 10.0.0.2 peer-group CPE
  neighbor 10.0.0.2 activate
  neighbor 10.0.0.3 peer-group CPE
  neighbor 10.0.0.3 activate
  default-information originate
 exit-address-family
```

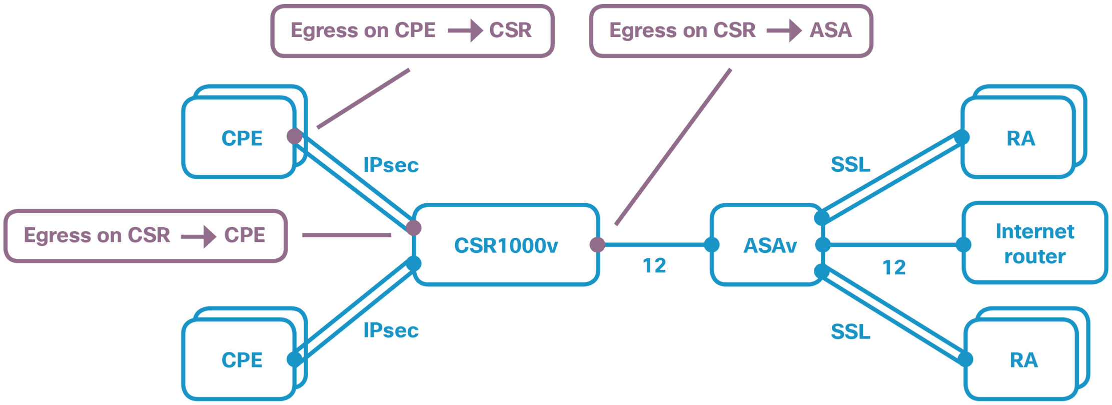
QoS

QoS in the vMS solution is used to achieve the following:

- Bandwidth prioritization inside IPSec tunnels between CPEs and CSR1000v
- Per CPE bandwidth limitation
- WAN Download Bandwidth is implemented via QoS shaping on the CSR1000v egress interface (Virtual-Template / Virtual-Access)
- WAN Upload Bandwidth is implemented via QoS shaping on the CPE egress interface (Tunnel 1)

These are end-customer customizable through the vMS End Customer Portal.

Figure 10.1 QoS Settings on CSR1000v and CPE.



The following is an example of QoS configuration applied to CSR1000v related to CPEs:

```
policy-map QoS-Profile1
 class CVPN-CRITICAL
  priority level 1 percent 40
 class CVPN-IMPORTANT
  priority level 2 percent 30
 class CVPN-SCAVENGER
  bandwidth remaining percent 10
 class class-default
  bandwidth remaining percent 20
policy-map QOS-POLICY-PARENT-DOWNSTREAM-CPE-9N989LLRWSN
 class class-default
  shape average 2097152
   service-policy QoS-Profile1
policy-map QoS-1Gbps
 class class-default
  shape average 1000000000
   service-policy QoS-Profile1


interface GigabitEthernet3
 description to-ASAv
 ip vrf forwarding IVRF
 ip address 10.192.1.2 255.255.255.0
 negotiation auto
 service-policy output QoS-1Gbps
```

The "QoS-Profile" policy-map defines traffic classes for the traffic prioritization. This policy-map is then used by the overall traffic-shaping policy-maps ("QOS-POLICY-PARENT-DOWNSTREAM-CPE..." and "QoS-1Gbps") for the directions to particular CPEs and towards the ASAv, respectively.

The CPE traffic-shaping policy-maps then get applied to the Virtual-Access interfaces on the CSR1000v via a combination of crypto maps and aaa attributes as shown below.

```
aaa attribute list CPE-9N989LLRWSN.my-new-cvpn.cisco.com
 attribute type interface-config "service-policy output QOS-POLICY-PARENT-DOWNSTREAM-CPE-9N989LLRWSN"
 !
crypto ikev2 authorization policy CPE-9N989LLRWSN.my-new-cvpn.cisco.com
 aaa attribute list CPE-9N989LLRWSN.my-new-cvpn.cisco.com
 route set interface
 route set access-list InjectRouteToCPE
 route accept any distance 220
```

End customers are allowed to modify QoS traffic classes and bandwidth ratios via the vMS Portal.

Troubleshooting

FlexVPN

Since FlexVPN is a configuration construct and its end-product is an IPSec/IKEv2 tunnel between two routers, the troubleshooting process is the same as for a Site-to-Site VPN. The first step is to check tunnel status and bi-directional traffic flow.

```
my-new-cvpn-csr#show crypto session
Crypto session current status

Interface: Virtual-Access1
Profile: FlexVpnToCPE
Session status: UP-ACTIVE
Peer: 192.168.11.14 port 500
  Session ID: 1
  IKEv2 SA: local 20.0.0.10/500 remote 192.168.11.14/500 Active
  IPSEC FLOW: permit 47 host 20.0.0.10 host 192.168.11.14
        Active SAs: 2, origin: crypto map
```

```
my-new-cvpn-csr#show crypto ipsec sa

interface: Virtual-Access1
    Crypto map tag: Virtual-Access1-head-0, local addr 20.0.0.10
```

```
   protected vrf: IVRF
   local  ident (addr/mask/prot/port): (20.0.0.10/255.255.255.255/47/0)
   remote ident (addr/mask/prot/port): (192.168.11.14/255.255.255.255/47/0)
   current_peer 192.168.11.14 port 500
     PERMIT, flags={origin_is_acl,}
    #pkts encaps: 12802, #pkts encrypt: 12802, #pkts digest: 12802
    #pkts decaps: 12676, #pkts decrypt: 12676, #pkts verify: 12676
    #pkts compressed: 0, #pkts decompressed: 0
    #pkts not compressed: 0, #pkts compr. failed: 0
    #pkts not decompressed: 0, #pkts decompress failed: 0
    #send errors 0, #recv errors 0
...
```

If further troubleshooting is required please refer to "IOS IKEv2 Debugs for Site-to-Site VPN with PSKs Troubleshooting TechNote", http://www.cisco.com/c/en/us/support/docs/ip/internet-key-exchange-ike/115934-technote-ikev2-00.html for more info.

Routing

BGP sessions on CSR1000v towards CPEs can be checked with the following command:

```
my-new-cvpn-csr#show ip bgp vpnv4 vrf IVRF summary
BGP router identifier 10.0.0.254, local AS number 65009
BGP table version is 4, main routing table version 4
3 network entries using 768 bytes of memory
3 path entries using 360 bytes of memory
3/2 BGP path/bestpath attribute entries using 792 bytes of memory
1 BGP AS-PATH entries using 24 bytes of memory
2 BGP extended community entries using 84 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 2028 total bytes of memory
BGP activity 3/0 prefixes, 3/0 paths, scan interval 60 secs

Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
10.0.0.2        4        65000    6455    6454        4    0    0 17:03:36        1
10.0.0.3        4        65000    6437    6427        4    0    0 16:59:43        1
```

For more information please refer to "Troubleshooting BGP", http://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/22166-bgp-trouble-main.html.

QoS

The first troubleshooting step is to verify that the policy-maps are applied to the correct interface and to verify traffic statistics are incrementing. Notice that the LAN interface GigabitEthernet3 and Virtual-Access1 both have QoS applied.

```
my-new-cvpn-csr#show policy-map interface
 GigabitEthernet3

  Service-policy output: QoS-1Gbps

    Class-map: class-default (match-any)
      2609 packets, 280281 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any
      Queueing
      queue limit 208 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 2296/250859
      shape (average) cir 1000000000, bc 4000000, be 4000000
      target shape rate 1000000000

      Service-policy : QoS-Profile1

        queue stats for all priority classes:
          Queueing
          priority level 1
          queue limit 512 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 0/0

        queue stats for all priority classes:
          Queueing
          priority level 2
          queue limit 512 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 0/0

        Class-map: CVPN-CRITICAL (match-any)
          0 packets, 0 bytes
          5 minute offered rate 0000 bps, drop rate 0000 bps
          Match: protocol attribute category internet-security
          Priority: 40% (400000 kbps), burst bytes 10000000, b/w exceed drops: 0

          Priority Level: 1
```

```
       Class-map: CVPN-IMPORTANT (match-any)
         0 packets, 0 bytes
         5 minute offered rate 0000 bps, drop rate 0000 bps
         Match: protocol attribute category business-and-productivity-tools
         Priority: 30% (300000 kbps), burst bytes 7500000, b/w exceed drops: 0

         Priority Level: 2

       Class-map: CVPN-SCAVENGER (match-any)
         0 packets, 0 bytes
         5 minute offered rate 0000 bps, drop rate 0000 bps
         Match: protocol attribute category trojan
         Match: protocol attribute category gaming
         Queueing
         queue limit 208 packets
         (queue depth/total drops/no-buffer drops) 0/0/0
         (pkts output/bytes output) 0/0
         bandwidth remaining 10%

       Class-map: class-default (match-any)
         2609 packets, 280281 bytes
         5 minute offered rate 0000 bps, drop rate 0000 bps
         Match: any
         Queueing
         queue limit 208 packets
         (queue depth/total drops/no-buffer drops) 0/0/0
         (pkts output/bytes output) 2296/250859
         bandwidth remaining 20%
Virtual-Access1

 Service-policy output: QOS-POLICY-PARENT-DOWNSTREAM-CPE-9N989LLRWSN

   Class-map: class-default (match-any)
     615 packets, 45457 bytes
     5 minute offered rate 0000 bps, drop rate 0000 bps
     Match: any
     Queueing
     queue limit 64 packets
     (queue depth/total drops/no-buffer drops) 0/0/0
     (pkts output/bytes output) 615/82698
     shape (average) cir 2097152, bc 8389, be 8389
     target shape rate 2097152

     Service-policy : QoS-Profile1
```

```
queue stats for all priority classes:
  Queueing
  priority level 1
  queue limit 512 packets
  (queue depth/total drops/no-buffer drops) 0/0/0
  (pkts output/bytes output) 0/0

queue stats for all priority classes:
  Queueing
  priority level 2
  queue limit 512 packets
  (queue depth/total drops/no-buffer drops) 0/0/0
  (pkts output/bytes output) 0/0

Class-map: CVPN-CRITICAL (match-any)
  0 packets, 0 bytes
  5 minute offered rate 0000 bps, drop rate 0000 bps
  Match: protocol attribute category internet-security
  Priority: 40% (838 kbps), burst bytes 20950, b/w exceed drops: 0

  Priority Level: 1

Class-map: CVPN-IMPORTANT (match-any)
  0 packets, 0 bytes
  5 minute offered rate 0000 bps, drop rate 0000 bps
  Match: protocol attribute category business-and-productivity-tools
  Priority: 30% (629 kbps), burst bytes 15700, b/w exceed drops: 0

  Priority Level: 2

Class-map: CVPN-SCAVENGER (match-any)
  0 packets, 0 bytes
  5 minute offered rate 0000 bps, drop rate 0000 bps
  Match: protocol attribute category trojan
  Match: protocol attribute category gaming
  Queueing
  queue limit 64 packets
  (queue depth/total drops/no-buffer drops) 0/0/0
  (pkts output/bytes output) 0/0
  bandwidth remaining 10%

Class-map: class-default (match-any)
  615 packets, 45457 bytes
  5 minute offered rate 0000 bps, drop rate 0000 bps
```

```
          Match: any
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 615/82698
          bandwidth remaining 20%
```

Troubleshooting Licensing

In the vMS solution, CSR1000v is licensed through Cisco Smart Software Licensing. This is an automated licensing model where the device (hardware or virtual) acquires the license from the Service Provider's Virtual Account which is hosted on Cisco Licensing Servers. The only information that is relayed to the device is the "ID token" which gets generated on the Virtual Account GUI.

In vMS, the ID token is stored in NSO global variables `VR_LICENSE_TOKEN` and is part of Day 0 configuration. As soon as the CSR1000v gets network connectivity, it registers with Cisco Licensing Server. CSR1000v also requires the `platform hardware throughput level` configuration line (also present in Day 1 configuration) in order to consume the license.

Having a registered license is critical for the proper operation of vMS solution. If a license is not present, bandwidth is limited to 100kbs. The first step in verifying the licensing status is:

```
my-new-cvpn-csr#show license all
Smart Licensing Status
======================

Smart Licensing is ENABLED

Registration:
  Status: REGISTERED
  Smart Account: Cisco Systems, Inc.
  Virtual Account: Default
  Initial Registration: SUCCEEDED on Dec  8 15:14:32 2015 UTC
  Last Renewal Attempt: None
  Next Renewal Attempt: Jun  5 15:14:32 2016 UTC
  Registration Expires: Never

License Authorization:
  Status: AUTHORIZED on Dec  8 23:22:08 2015 UTC
  Last Communication Attempt: SUCCEEDED on Dec  8 23:22:08 2015 UTC
  Next Communication Attempt: Jan  7 23:22:08 2016 UTC
  Communication Deadline: Mar  7 23:16:03 2016 UTC

License Usage
```

```
  ==============

  CSR 1KV AX 50M (ax_50M):
    Description: CSR 1KV AX 50M
    Count: 1
    Version: 1.0
    Status: AUTHORIZED

  Product Information
  ===================
  UDI: PID:CSR1000V,SN:9SEY1U5MOFC

  Agent Version
  =============
  Smart Agent for Licensing: 1.2.1_throttle/5
  Component Versions: SA:(1_2_1_throttle)1.1.0, SI:(rel20)1.0.1, CH:(rel4)1.0.19, PK:
(rel16)1.0.7
```

This output shows that the virtual device is registered with the Licensing Server and that the license is authorized and used.

It is important to verify the current platform throughput level:

```
  my-new-cvpn-csr#show platform hardware throughput level
  The current throughput level is 50000 kb/s
```

10.4 ASAv

Overview

The Adaptive Security Virtual Appliance (ASAv) is a virtualized network security solution based on the hardware Cisco ASA 5500-X Series firewalls. The vMS solution makes use of the following core features on ASAv:

- SSLVPN Remote Access
- Firewall, NAT, and Threat-Detection
- WCCPv2

Configuration of these features is deployed in three stages:

- **Day 0**: Contains basic device setting and IP connectivity that is added at device boot time via ESC. The Day 0 template contains minimal configuration to configure the device and bring it into management (SSH login). It is located on NSO at `/var/opt/ncs/state/packages-in-use/1/day0/cfg/`
- **Day 1 / Day 2**: NSO configures features directly on the device via SSH. XML templates are located on NSO at `/var/opt/ncs/state/packages-in-use/1/cloudvpn/templates/`

SSLVPN Remote Access

Cisco SSLVPN Remote Access is a comprehensive and versatile remote-access solution that supports the widest range of connectivity options, endpoints, and platforms. The main advantage over legacy VPN solutions is that SSL and DTLS protocols run over TCP avoiding NAT-related issues.

The vMS solution utilizes the SSLVPN feature on ASAv to enable remote access into a CloudVPN service. End customers are able to download the AnyConnect VPN client through the ASAv Web interface and install it on their desktop, laptop or hand-held devices.

ASAv SSLVPN related configuration is shown below.

```
webvpn
 enable outside
 anyconnect image disk0:/anyconnect/anyconnect-win-4.1.04011-k9.pkg 1
 anyconnect image disk0:/anyconnect/anyconnect-macosx-i386-4.1.04011-k9.pkg 2
 anyconnect image disk0:/anyconnect/anyconnect-linux-64-4.1.04011-k9.pkg 3
 anyconnect enable
 tunnel-group-list enable
 error-recovery disable
group-policy vpn internal
group-policy vpn attributes
```

```
 dns-server value 8.8.8.8 8.8.4.4
 vpn-tunnel-protocol ikev2 ssl-client ssl-clientless
dynamic-access-policy-record DfltAccessPolicy
username admin password cisco123 encrypted privilege 15
username admin attributes
 vpn-simultaneous-logins 0
username user1@cisco.com password ciscocisco encrypted
username user1@cisco.com attributes
 vpn-simultaneous-logins 1
tunnel-group vpn type remote-access
tunnel-group vpn general-attributes
 address-pool vpn_users
 authorization-server-group LOCAL
 default-group-policy vpn
tunnel-group vpn webvpn-attributes
 group-alias vpn enable
```

User accounts are configured through the vMS Portal and stored on the ASAv. User account suspension is available on the vMS Portal and is implemented via `vpn-simultaneous-logins 0` configuration line.

Firewall, NAT and Threat-Detection

The vMS solution uses these features for:

- Filtering unwanted traffic and blocking inbound connections.
- Inspecting TCP/UDP and Layer7 protocols.
- Providing NAT gateway for Internet access.
- Protecting against attacks and protocol exploits.

The complete ASAv configuration has been omitted for brevity.

WCCPv2

The Web Cache Communication Protocol v2 (WCCPv2) is a Cisco proprietary protocol that allows transparent traffic redirection and is often used for redirection and/or load-balancing of traffic towards cache/content engines. The main benefit of transparent redirection is that no other changes are required except configuring WCCPv2 on the ASAv and WSAv.

WCCP in vMS solution is used to transparently redirect traffic between ASAv and WSAv in order to perform URL Filtering, Anti-Malware and Anti-Virus scanning.

The WCCP configuration on ASAv is shown below.

```
access-list wccp-redirect extended deny ip host 10.192.2.3 any
access-list wccp-redirect extended permit tcp any any eq www
wccp 90 redirect-list wccp-redirect
wccp interface inside 90 redirect in
```

Troubleshooting

SSLVPN Remote Access

There are two typical scenarios regarding troubleshooting SSLVPN:

- Users cannot connect via their AnyConnect SSLVPN client. Relevant outputs are `show vpn-sessiondb anyconnect` and `debug webvpn anyconnect`
- Users can connect, but are not able to access resources. This requires checking firewall rules on ASAv and general connectivity verification across the Service Chain.

Firewall, NAT, and Threat-Detection

These features are usually troubleshot by inspecting logs, checking `show conn` and `show xlate` outputs on ASAv. Sample outputs are provided below.

```
my-new-asa# show conn
8 in use, 100 most used

UDP outside  129.131.120.18:53 inside  10.192.2.3:6393, idle 0:00:04, bytes 35, flags h
ICMP outside 8.8.8.8:0 inside  10.192.2.1:45020, idle 0:00:00, bytes 36, flags
ICMP outside 8.8.8.8:0 inside  10.192.2.1:45020, idle 0:00:00, bytes 36, flags
```

```
my-new-asa# show xlate
28 in use, 118 most used
Flags: D - DNS, e - extended, I - identity, i - dynamic, r - portmap,
       s - static, T - twice, N - net-to-net
NAT from outside:172.31.248.0/21 to outside:172.31.248.0/21
    flags sIT idle 125:08:38 timeout 0:00:00
NAT from outside:172.31.248.0/21 to outside:172.31.248.0/21
    flags sIT idle 125:08:38 timeout 0:00:00
NAT from outside:0.0.0.0/0 to outside:0.0.0.0/0
    flags sIT idle 125:08:38 timeout 0:00:00
NAT from inside:10.0.0.0/8 to outside:10.0.0.0/8
    flags sIT idle 125:08:38 timeout 0:00:00

ICMP PAT from inside:10.192.2.1/45032 to outside:23.47.112.66/45032 flags ri idle 0:00:05
```

```
timeout 0:00:30
  ICMP PAT from inside:10.192.2.1/45031 to outside:23.47.112.66/45031 flags ri idle 0:00:15
timeout 0:00:30
  ICMP PAT from inside:10.192.2.1/45030 to outside:23.47.112.66/45030 flags ri idle 0:00:25
timeout 0:00:30
```

WCCPv2

Basic troubleshooting steps include verifying if WCCPv is negotiated and redirection is happening. The following outputs are useful for this:

```
my-new-asa# show wccp 90

Global WCCP information:
    Router information:
        Router Identifier:                     23.47.112.66
        Protocol Version:                      2.0

    Service Identifier: 90
        Number of Cache Engines:               1
        Number of routers:                     1
        Total Packets Redirected:              2346
        Redirect access-list:                  wccp-redirect
        Total Connections Denied Redirect:     0
        Total Packets Unassigned:              0
        Group access-list:                     -none-
        Total Messages Denied to Group:        0
        Total Authentication failures:         0
        Total Bypassed Packets Received:       0
```

```
my-new-asa# show wccp 90 view

    WCCP Routers Informed of:
        23.47.112.66

    WCCP Cache Engines Visible:
        10.192.2.3

    WCCP Cache Engines NOT Visible:
        -none-
```

For more in depth WCCP troubleshooting, please refer to "WCCP Troubleshooting for Transparent Caching", http://www.cisco.com/c/en/us/support/docs/application-networking-services/500-series-cache-engines/9250-tshoot-wccp-9250.html. Even though this document is written for IOS devices, ASAv troubleshooting is the same and commands are very similar (typically without the"IP" option in the command).

Troubleshooting Licensing

In the vMS solution, ASAv is licensed through Cisco Smart Software Licensing. This is an automated licensing model where the device (hardware or virtual) acquires the license from the Service Provider's Virtual Account which is hosted on Cisco Licensing Servers https://software.cisco.com/#module/SmartLicensing. The only information that is relayed to the device is the ID token which gets generated on the Virtual Account GUI.

In vMS, the ID token is provided as a part of Day 0 configuration at boot. As soon as the device gets network connectivity, it tries to register to Cisco Licensing Server and acquire the license.

Having a proper license in place is critical for the proper operation of vMS solution. If a license is not present, the ASA is limited to 100 kbps. The first step in verifying the licensing status is:

```
  my-new-asa# show license all

  Cisco Smart Licensing Agent, Version 1.1.4_throttle/13

  Smart Licensing Enabled: Yes

  UDI:
  PID:ASAv,SN:9ABC0G2P35F

  Compliance Status: In Compliance

  Assigned License Pool: cVPN

  Grace period: Not in use

  Entitlement:
      Tag: regid.2014-08.com.cisco.ASAv-STD-2G,1.0_43ef7036-d89f-472e-ade2-ae8609af7365,
Version: 1.0, Enforce Mode: Authorized
      Requested Time: Dec  4 17:33:15 2015 UTC,  Requested Count: 1
      Vendor String: (null)

  Smart Licensing State: authorized (4)

  Licensing Certificates:
      ID Cert Info:
          Start Date: Dec  4 17:27:55 2015 UTC. Expiry Date: Dec  3 17:27:55 2016 UTC
          Serial Number: 369484
          Version: 3
          Subject/SN: b3161236-9763-4595-9890-c3246ed9c1f7
          Common Name: 9c267d7111296bfdfd6910277512f2aa3e60931d::1,2
      Signing Cert Info:
          Start Date: Sep 11 19:05:34 2013 UTC. Expiry Date: May 30 19:48:46 2038 UTC
          Serial Number: 3
          Version: 3

  Upcoming Scheduled Jobs:
      Certificate Renewal: Jun  1 17:33:56 2016 UTC (174 days, 18 hours, 24 minutes, 17 seconds
remaining)
      Certificate Expiration: Dec  3 17:27:53 2016 UTC (359 days, 18 hours, 18 minutes, 14
seconds remaining)
      Authorization Renewal: Jan  3 17:34:07 2016 UTC (24 days, 18 hours, 24 minutes, 28
seconds remaining)
      Authorization Expiration: Mar  3 17:28:05 2016 UTC (84 days, 18 hours, 18 minutes, 26
seconds remaining)
      Daily Job: Dec 10 17:33:13 2015 UTC (18 hours, 23 minutes, 34 seconds remaining)

  HA Info:    HA not available
      HA Sudi: Not Available
```

This output shows that the virtual device is registered with the Licensing Server and that the license is authorized and used.

Alternatively, the `show version` command will also indicate licensing status:

```
  my-first-cvpn-asa# show ver
  [...]
  License mode: Smart Licensing
  ASAv Platform License State: Unlicensed
  Active entitlement: ASAv-STD-2G, enforce mode: Eval period
```

This output shows a virtual device that is not (yet) successfully registered with the Licensing Server, and will be limited in bandwidth.

10.5 WSAv

Introduction

This section describes Web Security Appliance (WSA) VNF including its features used in the vMS solution, configuration, and basic troubleshooting steps. The WSAv is present in a Service Chain only for the CloudVPN Full Service.

vMS WSA Functionalities

In vMS solution the following subset of WSA features and functionalities are used:

- Cisco Web Usage Controls
- Application Visibility and Control (AVC)
- Web Reputation Filtering (WBRS)
- Anti-Malware/Anti-Spyware engines:
    - Sophos
    - Webroot
- Cisco Anti-Malware Protection (AMP)

For more details on WSAv features and capabilities, please refer to the "Cisco Web Security Appliance Data Sheet", http://www.cisco.com/c/en/us/products/collateral/security/content-security-management-appliance/data_sheet_c78-729630.html

vMS WSAv Configurations

Traditionally, the WSA appliance is configured using Web GUI (Graphical User Interface) interface, or some of the parts may be configured using CLI (Command Line Interface). In vMS Solution, all WSAv configuration is controlled by NSO. WSAv Web GUI access is not available. Cisco WSA uses XML configuration format instead of the traditional Cisco CLI.

WSA configuration is deployed in three stages. The three stages of deployment are known as Day 0, Day 1, and Day 2 and are described below.

Note: Please refer to Figure 2.6 "Service Chain Creation Detailed Call Flow" in Chapter 2.4 "vMS Call Flows" for a complete call flow diagram.

1. Day 0 WSAv Configuration

Day 0 WSAv configuration is deployed at WSA boot time to establish basic management network connectivity. Day 0 configuration contains basic networking configuration that enables other orchestration components to reach WSAv VNF. The Day 0 config is downloaded by ESC from NSO and used when booting the VM in OpenStack.

Day 0 WSAv configuration contains:

- Hostname, physical interface settings
- Management IPv4/IPv6 interface configuration / and hostnames with ports where Management port is listening on.
- Management Routing Table Configuration
- Administrator user credentials
- DNS Settings, and preferred DNS source interfaces
- Security Features used in the solution
- Log configuration and log level settings (all the relevant logs are configured, including level of internal logs, and in some cases locations where the logs need to be stored (pushed by means of Syslog / or SCP / FTP protocols)

2.Day 1 WSAv Configuration

Day 1 WSA configuration is deployed after WSAv is successfully booted.

Day 1 configuration continues configuration of the WSAv appliance and adds additional networking and logging/supportability configurations to Web Security Appliance. NSO establishes an SSH connection to WSAv and configures WSAv.

In general, inside Day 1 configuration WSAv receives the following:

- WSA receives configuration of Data (proxy) interfaces (IPv4 and/or IPv6 address)
- Configuration of Data routing Table
- WCCPv2 (Web Cache Communication Protocol) configuration – that will enable transparent redirection of end customer's traffic from ASAv to WSAv
- Additional routes for Data traffic
- NTP Server and TimeZone configuration
- SNMP configuration

3. Day 2 WSAv Configurations

After NSO successfully deploys Day 1 configuration, it deploys final Day 2 WSAv configuration. Day 2 WSA configuration contains relevant security policy configurations that reflect the end customer's Web Security URL filtering level selected in vMS Portal. The Day 2 configuration is sent to WSAv from the NSO.

Three different flavors of WSAv's Day 2 HTTP security policy configurations are referred to as:

- **Basic**
- **Enhanced**
- **Full**

The difference between these WSA Security Policy Configurations are configurations of URL Filtering Services, and decisions to either:

- ○ Block access to certain URL Categories
- ○ Monitor (further scan) traffic towards certain URL Categories, or
- ○ Warn end customers about certain sites

The following sections will describe URL Filtering and Anti-Malware settings for each of the three flavors of WSA's Web Security Policy configurations.

Anti-Malware and Anti-Virus Settings

In all three configuration flavors, Sophos, Webroot and AMP Scanners will be set in the following manner:

- ○ WSAv **blocks** all known Malware/Virus traffic discovered by Sophos, Webroot or AMP Scanners.
- ○ WSAv applies "**monitor**" action for all Unscannable files and Suspect User Agents.

> **Note:** Monitor action means that WSA will continue to monitor the request, through WSA's request pipeline.

Basic WSAv Day 2 Security Policy Configuration Overview

In basic WSA Policy Security flavor, Security policies are configured in such a way that:

- URL categories containing: Illegal activities (such as Illegal activities, Downloads, Hacking, Child Abuse) are being **blocked**.
  - This means that if the end customer tries to access a site that falls into one of the categories listed in the following table, the request will be blocked by WSA, and the end customer would receive an End User Notification page from WSA about the reason of site being blocked.
- For URL categories that can be considered as "business inappropriate content", WSA will apply a security policy action "**warning**".
  - This means that if the end customer tries to access such a URL, WSA will first display a warning page notifying the customer that the content is not recommended to be visited, according to corporate policies, and the customer must acknowledge that he or she is aware of corporate policies. After clicking on an acknowledgment link, the customer will be able to access the URL.
- All other URL categories are configured with the default action of the Monitor
  - Monitor action means that request will not be terminated, but will be further processed in the WSA pipeline, and further scanned by means of Anti-Virus and Anti-Malware software (Sophos, Webroot, AMP).

The following table shows Web Security Policy configuration applicable in Basic WSA service.

Table 10.1 WSA Web Security Policy - URL Filtering configuration in Basic Setting

No	Category based URL filtering - Basic WSA Flavor
WSA Policy Action Block	
1	Hacking
2	Child Abuse Content
3	Hate Speech
4	Illegal Activities
5	Illegal Downloads
WSA Policy Action Warn/Continue	
1	Adult
2	Cheating and Plagiarism
3	Illegal Drugs
4	Gambling
5	Non-Sexual Nudity
PA	**WSA Policy Action Scan/Monitor:**
1	Everything else

Enhanced WSAv Day 2 Security Policy Configuration Overview

In Enhanced WSA Policy flavor, a quite different distribution of URL categories is being blocked/warned or monitored. Access to illegal content and Adult content is blocked. If customers access inappropriate content, they are warned by WSA.

The following table shows Web Security Policy configuration applicable in Enhanced WSA service.

Table 10.2 WSA Web Security Policy - URL Filtering configuration in Enhanced setting

No	Category based URL filtering - Enhanced Setting
WSA Policy Action Block	
1	Hacking
2	Child Abuse Content
3	Hate Speech
4	Illegal Activities

5	Illegal Downloads
6	Adult
7	Cheating and Plagiarism
8	Illegal Drugs
WSA Policy Action Warn/Continue	
1	Alcohol
2	Entertainment
3	Sports and Recreation
4	Pornography
5	Social Networking
6	Sex Education
7	Dating
8	Extreme
9	Gambling
10	Non-Sexual Nudity
WSA Policy Action Scan/Monitor	
1	Everything else

Full WSAv Day 2 Security Policy Configuration Overview

Full WSA Security policy flavor is the most restrictive, where WSA blocks access to all the sites that are either considered as illegal, or hold business-inappropriate content.

The following table shows Web Security Policy configuration applicable in Enhanced WSA service.

Table 10.3 WSA Web Security Policy - URL Filtering configuration in Full Setting

No	**Category based URL filtering - Full Setting**
WSA Policy Action Block	
1	Hacking
2	Child Abuse Content
3	Hate Speech
4	Illegal Activities
5	Illegal Downloads
6	Adult
7	Cheating and Plagiarism

8	Illegal Drugs
9	Alcohol
10	Entertainment
11	Sports and Recreation
12	Pornography
13	Social Networking
14	Sex Education
15	Dating
16	Extreme
17	Gambling
18	Non-Sexual Nudity
WSA Policy Action Scan/Monitor	
1	Everything else

WSAv Troubleshooting

This section describes basic troubleshooting steps related to WSAv implementation inside the vMS solution. The Web Security Appliance is a highly complex security product on its own and deeper troubleshooting of WSA is out-of-scope for this document.

Basic WSA Sanity Check Commands

This section will list common commands that can be used in WSA CLI in order to check if WSA is operating as expected.

Note: It is assumed that the Service Provider's vMS Operations team has access to the NSO, and knows how to connect to each VNF from NSO. Also, it is assumed that the Service Provider knows all the passwords that are set in each VNF for administrative access.

Note: The IPv4/IPv6 address of each VNF belonging to each end customer (tenant) can be easily obtained from **NSO ncs_cli** command `show devices brief` command. Accordingly, the NSO server Service Provider vMS Operator / or Cisco representative can log in into each VNF's CLI interface by means of SSHv2 protocol.

For full description of all the commands that can be run in WSA CLI, please consult "Cisco Web Security Appliance End-User Guides", [http://www.cisco.com/c/en/us/support/security/web-security-appliance/products-user-guide-list.html.](http://www.cisco.com/c/en/us/support/security/web-security-appliance/products-user-guide-list.html)

Checking of WSA Version and Versions of Security Scanners

The WSA appliance runs its own Operating System called AsyncOS. vMS solution supports only specific WSA AsyncOS versions. A Service Provider or Cisco representative might want to check the version of AsyncOS running on WSA. It is important to check whether Security Scanners are running properly, and being updated properly.

From WSA CLI, the `version` command can be issued, as shown in the following output. The version shown below is 8.6.904 and it shows that Security Scanners are not being updated.

```
  my-new-cvpn-wsa.cisco.com> version

  Current Version
  ===============
  Product: Cisco S100V Web Security Virtual Appliance
  Model: S100V
  Version: 8.6.0-904
  Build Date: 2015-06-04
  Install Date: 2015-12-08 21:13:09
  Serial #: BD3697E597F20B4E9073-ED1385A02E52
  BIOS: 1.7.5-20150310_111955-batsu
  CPUs: 2 expected, 2 allocated
  Memory: 6144 MB expected, 6144 MB allocated
  RAID: NA
  RAID Status: Unknown
  RAID Type: NA
  BMC: NA
  Cisco DVS Engine: 1.0 (Never Updated)
  Cisco DVS Malware User Agent Rules: 0.554 (Never Updated)
  Cisco DVS Object Type Rules: 0.554 (Never Updated)
  Cisco Trusted Root Certificate Bundle: 1.2 (Never Updated)
  Cisco Certificate Blacklist: 1.2 (Never Updated)
  L4 Traffic Monitor Anti-Malware Rules: 1.0 (Never Updated)
  Cisco Web Usage Controls - Web Categorization Engine: 3.0.0.044 (Never Updated)
  Cisco Web Usage Controls - Web Categorization URL Keyword Filters: 1304226000 (Never Updated)
  Cisco Web Usage Controls - Web Categorization Prefix Filters: 1386668967 (Never Updated)
  Cisco Web Usage Controls - Web Categorization Categories List: 1337979188 (Never Updated)
  Cisco Web Usage Controls - Dynamic Content Analysis Engine: 2.1.0-016 (Never Updated)
  Cisco Web Usage Controls - Dynamic Content Analysis Engine Data: 3.1.0001 (Never Updated)
  Cisco Web Usage Controls - Application Visibility and Control Engine: 1.1.0-076 (Never
Updated)
  Cisco Web Usage Controls - Application Visibility and Control Data: 1.1.0.17-002 (Never
Updated)
  Web Reputation Engine: 3.0.0.044 (Never Updated)
  Web Reputation IP Filters: 1386637514 (Never Updated)
  Web Reputation Rules: 1358979215 (Never Updated)
  Web Reputation Prefix Filters: 1386668967 (Never Updated)
  Webroot Anti-Malware Engine: 2.1.5.8 (Never Updated)
  Webroot Engine Definition: 2.1.5.8 (Never Updated)
  Webroot Malware Categories DATs: 1353 (Never Updated)
  McAfee Anti-Malware Engine: 5600 (Never Updated)
  McAfee Engine Definition: 5200 (Never Updated)
  McAfee DATs: 5688 (Never Updated)
  Sophos Engine: 3.2.07.389_4.95 (Never Updated)
  Sophos IDE:  (Never Updated)
```

As seen above, this command displays AsyncOS running 8.6.904, version/model of WSAv (WSA S100v in our example), Serial Number. It is important to note the assigned number of CPUs and Memory. It is expected that WSAv has enough resources assigned, and that the "expected" value matches the "allocated" output value.

Please note that in the example output provided, it is evident that some Security components never received updates.

In normal WSA operations, all the security components like Anti-Virus / Anti-Malware scanners (Sophos/McAfee/Webroot/AMP), URL Filtering and WBRS (Web Based Reputation) should be receiving updates on a regular basis. If this is not the case (as shown in the above command output), WSA probably has an issue communicating with the update server. Further details about update services can be checked in the WSA's updater_logs.

Checking Overall WSA Status

WSA CLI Command: `status detail` gives an overview of basic WSA status, by providing information about: overall CPU usage, Memory and Disk Usage, Number of transactions (requests) per second, Bandwidth used, Response time, Cache Hit ratios and number of connections.

When troubleshooting potential WSA issues, one should pay attention to the following aspects of the status detail command:

- **CPU usage**: Higher CPU usage is naturally expected if WSAv is processing more traffic. Ideally, CPU usage will be below 85%-90%.
- **Memory usage:** his command displays overall memory usage. Please note that it is expected that WSA is always having higher memory usage. High usage of memory is not a reason to be alarmed.
- **Transactions per second:** represent the average number of requests per second that are redirected to WSA appliance. This number should be in relation to the end customer's reported number of users. Roughly, if an end customer reported a specific number of users using vMS service, it is expected to see 10 to 15% of those users to be active concurrently in the same second. If the WSA appliance handles more transactions per second then the model is designed for, the end customer might experience slowness / performance issues while accessing the Internet.
- **Response time** section of the output is important when talking about performance. Ideally, average response time in the last hour to be below 1000ms to 2000ms.

- **Connections** section of the output shows client and server connections, and can be used as a sanity check to see if WSA is properly releasing client/or server connections.

The following output shows an example of `status detail` WSA CLI command.

```
my-new-cvpn-wsa.cisco.com> status detail

Status as of:                  Wed Dec 09 20:49:42 2015 GMT-6
Up since:                      Tue Dec 08 21:12:50 2015 GMT-6 (23h 36m 52s)
System Resource Utilization:
  CPU                                     2.6%
  RAM                                    43.0%
  Reporting/Logging Disk                 10.3%
Transactions per Second:
  Average in last minute                     0
  Maximum in last hour                       0
  Average in last hour                       0
  Maximum since proxy restart                0
  Average since proxy restart                0
Bandwidth (Mbps):
  Average in last minute                 0.000
  Maximum in last hour                   0.000
  Average in last hour                   0.000
  Maximum since proxy restart            0.000
  Average since proxy restart            0.000
Response Time (ms):
  Average in last minute                     0
  Maximum in last hour                       0
  Average in last hour                       0
  Maximum since proxy restart                0
  Average since proxy restart                0
Cache Hit Rate:
  Average in last minute                     0
  Maximum in last hour                       0
  Average in last hour                       0
  Maximum since proxy restart                0
  Average since proxy restart                0
Connections:
  Idle client connections                    0
  Idle server connections                    0
  Total client connections                   0
  Total server connections                   0
```

Checking WSA Configuration File and WSA Configuration related-vMS Issues

Sometimes, in the process of vMS / WSAv troubleshooting, there is a need to check WSAv's configuration. This can be done using CLI command `showconfig`. As the output of this command is the entire WSA XML configuration, it will not be shown. The next sections describe the common vMS related WSA issues that can happen in the process of deploying WSA's Day 0, Day 1 and Day 2 configuration.

1. WSA Day 0 vMS Configuration Issues

As explained in the previous sections, each WSAv will boot with its Day 0 configuration and with loaded WSA license file.

In some rare cases, it can happen that Day 0 configuration is not loaded during WSA's first boot. In this case, the error will be obvious, as NSO will not be able to access WSA to deploy further Day 1 and Day 2 configurations, and WSA will have default administrative user credentials.

If WSA Day 0 configuration is not loaded in Day 0 stage, Day 1 and Day 2 configurations will also be missing, and this will be reflected in specific WSAv status shown in `ncs_cli`, as described in section "Service Chain Verification" in Chapter 10.2 "Service Chain General Troubleshooting".

2. WSA Day 1 vMS Configuration Issues

From the output of `showconfig` command from WSA CLI,it can be determined if WSA has Day 1 configuration deployed by searching showconfig output for XML tag for configuration of P1 data interface. If the WSAv configuration has the tag `<interface_name>P1</interface_name>` it is certain that NSO has successfully deployed Day 1 WSA configuration. Checking NED trace files in `/var/log/ncs` is a good source of troubleshooting information for these types of issues.

3. WSA Day 2 vMS Configuration Issues

In rare cases NSO will not deploy the expected Day 2 configuration and (Basic/Enhanced/Full) will not be loaded successfully. The easiest way to troubleshoot such issues is actually by checking the status of particular WSA VNF in NSO's `ncs_cli`, as described in section "Service Chain Verification" in Chapter 10.2 "Service Chain General Troubleshooting". In order to understand why the NSO failed to deploy configuration in specific cases, the best troubleshooting option are NED trace logs located on the NSO server. Logs of communication between NSO and each VNF will be located in: /`var/log/ncs` directory. WSA NED trace logs should be named: `ned-ciscowsa-<tenant_name>-WSA-esc.trace`.

Relevant WSA Logs

Web Security Appliance has lots of predefined logs that help monitor the appliance and troubleshoot issues related to traffic that is scanned/inspected by WSA appliance.

The most relevant logs for WSA vMS implementation consist of:

1. Accesslogs - used to log each end-user request and to log the action for the specific URL end-user requested.

Some of the examples of accesslog entries for the specific request are shown below:

- **Request permitted by WSA Security Policy**

The following output represents the response of WSA in the case where WSA by policy is permitting access to the particular URL. The response code of the request logged is "**200 OK**", which means that WSA inspected the traffic, forwarded it to the destination URL, and forwarded back the response to the requestor (end-user).

It is expected that all non-malicious requests (requests towards safe/permitted URLs), and all the requests that are not blocked by the policy, are logged with 20x or 30x HTTP response code.

> **Note:** Accesslogs will not be discussed in detail, as each field is described in Web Security Appliance User Guide / Section **Access Log Files**

Example output:

```
  1447290118.877 5329 10.1.2.3 TCP_MISS/200 9534 GET http://www.test.com "-"
DIRECT/www.test.com text/html DEFAULT_CASE_12-DefaultGroup-DefaultGroup-NONE-NONE-NONE-
DefaultGroup <IW_busi,0.0,0,"-",0,0,0,1,"-",-,-,-,"-",1,-,"-","-",-,-,IW_busi,-,"Unknown","-
","Unknown","Unknown","-","-",14.31,0,-,"Unknown","-",-,"-",-,-,"-","-"> -
```

> **Tip:** All WSA logs can be searched from WSA CLI, by issuing command " grep" and then selecting the appropriate log file, and entering the search criteria. Please note that search criteria for all the logs can also be a regular expression.WSA logs can always be searched by providing grep the command as a one line in CLI. For the example, to search for all the requests from end-users towards site www.cisco.com, the one line grep command `grep www.cisco.com accesslogs` could be used.

- **Request denied (blocked) by WSA Security Policy - URL Filtering**

The following accesslog line describes the request being blocked due to WSA Security policy configuration by URL filtering component. Of course, each WSA security flavor would block access to URL categories, as defined in the previous section.

Request denied (blocked) by WSA Security Policy - by URL filtering configuration would have **403 HTTP response code** in the access logs, with **BLOCK_WEBCAT** decision WSA tag in accesslogs as shown below.

One example of accesslog line is below:

```
  1441268435.380 2 10.1.2.3 TCP_DENIED/403 0 GET http://whiskey.com - NONE/- - BLOCK_WEBCAT-
DefaultGroup-DefaultGroup-NONE-NONE-NONE-NONE <IW_infr,-9.0,-,"-",-,-,-,-,"-",-,-,-,"-",-,-,"-
","-",-,-,IW_infr,-,"-","adware","Unknown","Unknown","-","-",0.00,0,-,"-","-",-,"-",-,-,"-","-
"> -
```

- **Request denied (blocked) by WSA Security Policy - WBRS - Site has a low reputation score**

The following accesslog line describes the request being blocked due to WSA Security policy configuration and WBRS component. As explained earlier, if a customer requests an URL whose domain has a low Web Reputation Score, WSA will block such a request, and log BLOCK_WBRS as a WSA policy decision code. The customer would receive **403 HTTP response code** for that particular request made from the Internet.

One example of accesslog line is shown below:

```
  1441268435.380 2 10.1.2.3 TCP_DENIED/403 0 GET http://mnet-ad.net/px.gif?
ch=2&rn=8.682410769157897 - NONE/- - BLOCK_WBRS_12-DefaultGroup-DefaultGroup-NONE-NONE-NONE-
NONE <IW_infr,-9.0,-,"-",-,-,-,-,"-",-,-,-,"-",-,-,"-","-",-,-,IW_infr,-,"-
","adware","Unknown","Unknown","-","-",0.00,0,-,"-","-",-,"-",-,-,"-","-"> -
```

2. SHD_logs

SHD_logs on WSA are useful when talking about checking the overall health status of the WSA appliance. By default WSA will log parameters such as CPU usage / Memory usage / Number of requests per second / Number of client/server connection / Response time etc.

It can be useful when the Service Provider or operator needs to check the overall amount of traffic going through the appliance, or during troubleshooting of potential performance issues.

Without going into too much detail, a SHD log example is shown in the output below.

```
  Sun Aug 30 16:47:32 2015 Info: Status: CPULd 25.9 DskUtil 12.5 RAMUtil 74.4 Reqs 0 Band 0
Latency 0 CacheHit 0 CliConn 0 SrvConn 0 MemBuf 0 SwpPgOut 234038 ProxLd 0 Wbrs_WucLd 0.0 LogLd
0.0 RptLd 0.0 WebrootLd 0.0 SophosLd 0.0 McafeeLd 0.0
```

3. Proxy logs

Proxy logs log errors related to general Web Proxy operations, and are in most cases meant to be used by Cisco Technical Assistance Center (TAC) Support Engineers in resolving more complex WSA issues.

Transparent Traffic Redirection Troubleshooting (WCCP)

End-customer traffic going through the vMS service chain is transparently redirected from ASAv to WSAv by means of WCCPv2 (Web Cache Communication Protocol).

The best indication of whether traffic is indeed being redirected can be found in WSA's accesslogs, as discussed in the previous section.

- If WSA CLI command `tail accesslogs` keeps giving output, it is certain that end-user traffic is redirected to the WSA appliance.
- WCCP service has to be negotiated and established properly, and this can be easily seen in ASAv by executing `show wccp` and `show wccp 90` commands, as shown in section "End-to-end Connectivity Verification" in Chapter 10.2 "General Troubleshooting".

WSAv Licensing Considerations and Troubleshooting

Web Security Appliance, in the version included in vMS release 2.x, does not support Smart Licensing. WSA is instead licensed using a specific license file containing feature keys for each of the Security Features used in the vMS WSA solution.

As shown in Chapter 2.4 "vMS Call Flows", each WSAv deployed gets the WSA license file at the time of first bootup. If the license is properly loaded by ESC, WSAv will have all the needed licenses upon booting.

Nevertheless, in rare cases it might happen that WSAv either did not get or did not install a license. In that case, none of the security features would be working, and as a result, in most of the cases, deployment of Day 2 configuration would not be successful.

To check if WSAv has the proper license, execute `showlicense` command from WSA CLI:

```
  my-new-cvpn-wsa.cisco.com> showlicense

  Virtual License
  ===============


  vln                      VLNWSAxxxxxxxxxx
  begin_date               Wed May 1 19:26:24 2014 GMT
  end_date                 Sun May 21 19:26:24 2016 GMT
  company                  Company Name that purchased the license;  seats                    1
  serial                   B9D4
  email                    contactemail@company.com#10;  issue
8d32d16d348d4c6a88276aae8f59377f
  license_version          1.1
```

If the license is present, output will look similar to the above. In the case where the WSA license was not applied, the message that the license is not present would be received.

In order to check if all the features are used, after WSA has Day 2 configuration successfully deployed, please use featurekey WSA CLI command.

If the license is applied, and WSA has a valid Day 2 configuration, output of this command should be as shown below:

```
  my-new-cvpn-wsa.cisco.com> featurekey

  Module                                   Quantity   Status     Remaining    Expiration Date
  Cisco L4 Traffic Monitor                 1          Active     171 days     Sun May 29 05:59:59
2016
  Cisco HTTPS Proxy                        1          Dormant    171 days     Sun May 29 05:59:59
2016
  Cisco Web Usage Controls                 1          Active     171 days     Sun May 29 05:59:59
2016
  Sophos                                   1          Active    171 days     Sun May 29 05:59:59
2016
  McAfee                                   1          Dormant    171 days     Sun May 29 05:59:59
2016
  Webroot                                  1          Active    171 days     Sun May 29 05:59:59
2016
  Cisco Web Proxy & DVS Engine             1          Active     172 days     Mon May 30 01:26:24
2016
  Cisco AnyConnect Secure Mobility         1          Dormant    171 days     Sun May 29 05:59:59
2016
  Cisco Web Reputation Filters             1          Active     171 days     Sun May 29 05:59:59
2016
```

10.6 Common Issues

This section covers troubleshooting of the following common issues:

- VNFs not being created
- Unable to establish VPN Tunnels
- CloudVPN network reachability issues
- CloudVPN slowness or throughput reduction issues
- HTTP/HTTPS Destination reachability issues
- Remote Access User connection issues

VNFs not being Created

If the REMOTE field in output of the command `show cloudvpn-data <service-name> devices-list` does not move to "ready" from "not-existing" there may be a problem creating the VNFs.

```
admin@ncs-vm> show cloudvpn-data device-list
NAME           DEVICE                 REMOTE  READY           PROVISIONED
---------------------------------------------------------------------------
my-new-cvpn    my-new-cvpn-ASA-esc    -       not-existing    false
               my-new-cvpn-CSR-esc    -       not-existing    false
               my-new-cvpn-WSA-esc    -       not-existing    false
```

This could indicate an issue between ESC and OpenStack, or a problem with communications between ESC and NSO. The most relevant log files, in this case , would be the ESC `/var/log/esc/yangesc.log` and the NSO `/var/log/ncs/ncs-java-vm.log`. Please see Chapters 5.8 "NSO Troubleshooting" and 6.7 "ESC Troubleshooting".

Unable to Establish VPN Tunnels

After the service orchestration workflow has completed, NSO should be able to connect to each of the devices and deliver Day 1/ Day 2 configurations. If both Tunnel0 and Tunnel1 are configured on the CPE device but the FlexVPN tunnels are unable to establish, please refer Chapters 10.3 "CSR1000v Troubleshooting" and 7.8 "CPE Troubleshooting".

> **Note**: A common root cause of this scenario is related to improper Network Address Translation (NAT) of the IPsec tunnels.

CloudVPN network reachability issues

This section will describe the common issues and troubleshooting steps when customers report that some network destinations are not reachable.

Assumptions

- Tunnel1 towards the CSR1000v is UP on the CPE router(s).
- PCs behind the CPE router(s) can ping Internet destinations.

If Tunnel1 to the CSR1000v is not established, follow troubleshooting as shown in section "End-to-end Connectivity Verification" in Chapter 10.2 "General Troubleshooting".

Troubleshooting

1. The following data will be useful in diagnosing the issue:

- Which protocol(s) is causing the problem:
    - If end customers complain about reachability of HTTP or HTTPS destinations, please refer to section "HTTP/HTTPS Destination reachability Issues" in this chapter.
    - If end customers report problems with protocols other than HTTP or HTTPS, continue to step 2.

2. Obtain the following details regarding the specific reachability issues:

- Is it present for all the end customers in all locations or localized to one location?
    - If the issue is seen in all the locations, it is not local to CPE, but further troubleshooting should be done on CSRv and ASAv. Please refer to device-specific troubleshooting sections for the next steps.
    - If the issue is localized to only one location (one CPE), and not seen in other locations, please refer to section "End-to-end Connectivity Verification" in Chapter 10.2 "General Troubleshooting" to troubleshoot the issue further.

3. Collect packet captures from either end point (e.g. PC or server) in order to analyse the protocol flow.

> **Note**: A common root cause of such connectivity issues can be related to Path MTU discovery (PMTUD) due to IPsec overhead and blocked ICMP.

CloudVPN slowness or throughput reduction issues

There are multiple issues that could cause network slowness or throughput reduction in the vMS solution. The CPE's Internet connection speed should be verified independently as a benchmark for expected performance.

1. Low performance affecting all types of traffic (site-to-site traffic and site-to-Internet)

The first check would be verifying if the CSR1000v has correctly installed the Cisco Smart License. Without a valid license, the CSR1000v will be limited to 100 kb/s throughput. The following two commands can be used to check the current throughput level and the license installation:

```
my-new-cvpn-csr#show platform hardware throughput level
The current throughput level is 50000 kb/s

my-new-cvpn-csr#show license usage
License Authorization:
  Status: AUTHORIZED on Dec  8 23:22:08 2015 UTC

CSR 1KV AX 50M (ax_50M):
  Description: CSR 1KV AX 50M
  Count: 1
  Version: 1.0
  Status: AUTHORIZED
```

If no license is installed, the CSRv will default to 100kb/s bandwidth.

```
router#show platform hardware throughput level
The current throughput level is 100 kb/s
```

Refer to Chapter 10.3 "CSR1000v" for more information on licensing issues.

2. Low Performance affecting only Internet traffic.

Without a valid license, the ASAv will be limited to 100 kb/s throughput. The following license command can be used to verify that the platform is fully licensed:

```
mycvpn-ASA-esc# show license all

Cisco Smart Licensing Agent, Version 1.1.4_throttle/13

Smart Licensing Enabled: Yes

UDI:
PID:ASAv,SN:9ASAEDSNR8L

Compliance Status: In Compliance

Assigned License Pool: Cisco Systems
```

```
  Grace period: Not in use

  Entitlement:
      Tag: regid.2014-08.com.cisco.ASAv-STD-2G,1.0_43ef7036-d89f-472e-ade2-ae8609af7365,
Version: 1.0, Enforce Mode: Authorized
      Requested Time: Dec  4 17:20:05 2015 UTC,  Requested Count: 1
      Vendor String: (null)

  Smart Licensing State: authorized (4)

  Licensing Certificates:
      ID Cert Info:
          Start Date: Dec  4 17:14:46 2015 UTC. Expiry Date: Dec  3 17:14:46 2016 UTC
          Serial Number: 369278
          Version: 3
          Subject/SN: 54151b48-1f2b-4e9f-a72c-b0d7745456550029
          Common Name: f43e7474ad37aa10a88d7sgfdg83cfd87dca6f8::1,2
      Signing Cert Info:
          Start Date: Sep 11 19:05:34 2013 UTC. Expiry Date: May 30 19:48:46 2038 UTC
          Serial Number: 3
          Version: 3

  Upcoming Scheduled Jobs:
      Certificate Renewal: Jun  1 17:20:47 2016 UTC (174 days, 19 hours, 59 minutes, 46 seconds
remaining)
      Certificate Expiration: Dec  3 17:14:45 2016 UTC (359 days, 19 hours, 53 minutes, 44
seconds remaining)
      Authorization Renewal: Jan  3 17:20:58 2016 UTC (24 days, 19 hours, 59 minutes, 57
seconds remaining)
      Authorization Expiration: Mar  3 17:14:57 2016 UTC (84 days, 19 hours, 53 minutes, 56
seconds remaining)
      Daily Job: Dec 10 17:20:04 2015 UTC (19 hours, 59 minutes, 3 seconds remaining)

  HA Info:    HA not available
      HA Sudi: Not Available
```

Refer to Chapter 10.4 "ASAv" for further information on ASAv licensing.

3. Low performance affecting HTTP traffic.

It is important to note that the CloudVPN Full service includes web content filtering via the WSAv. The ASAv is responsible for redirecting HTTP traffic using WCCPv2. For issues involving web content filtering see Chapter 10.5 "WSAv".

HTTP/HTTPS Destination reachability Issues

In general, these types of issues fall into the following three categories:

1. End customers report that they can not reach external HTTP URL

Assumptions

- Tunnel1 towards the CSR1000v is UP on the CPE router(s).
- PCs behind the CPE router(s) can ping Internet destinations.
- CloudVPN service level is Full
- End customers are reporting intermittent issues where some of their HTTP requests are either blocked, or not being delivered.

If end customer has problems reaching any destination from behind the CPE, continue troubleshooting as shown in section "End-to-end Connectivity Verification" in Chapter 10.2 "General Troubleshooting".

Troubleshooting Steps

- Obtain the URL that is not reachable from the end customer's browser, and check end customer's IP address
- If WSAv is used, check if the request is present in WSAv's accesslogs as described in Chapter 10.5 "WSAv".

In order to check WSAv accesslogs, type the following command inside WSAv's CLI:

```
grep "<end_user_ip_address>.*<URL>" accesslogs
```

Where `<end_user_ip_address>` represents the IP address of the end user reporting the issue, `<URL>` represents the URL that the end customer reported to be inaccessible.

Depending on the output of the command:

- If the WSAv shows the traffic in the logs, check the access log decision tag and the HTTP response code to see if the content is intentionally blocked by content filters. Refer to WSAv User Guide or Chapter 10.6 "WSAv" for further details.
- If the WSAv does not have such a request in the accesslogs, this would indicate that the original request did not reach the WSAv appliance.

  Next steps:

  - Please see section "End-to-end Connectivity Verification" in Chapter 10.2 "General Troubleshooting" for details on how to verify end-end connectivity.

- Ensure that WCCP is up and running between WSAv and ASAv (see ASAv Troubleshooting)

> **Note**: It is worth noting that internal site-to-site traffic will only traverse the CSR1000v and will not pass through the ASAv or WSAv.

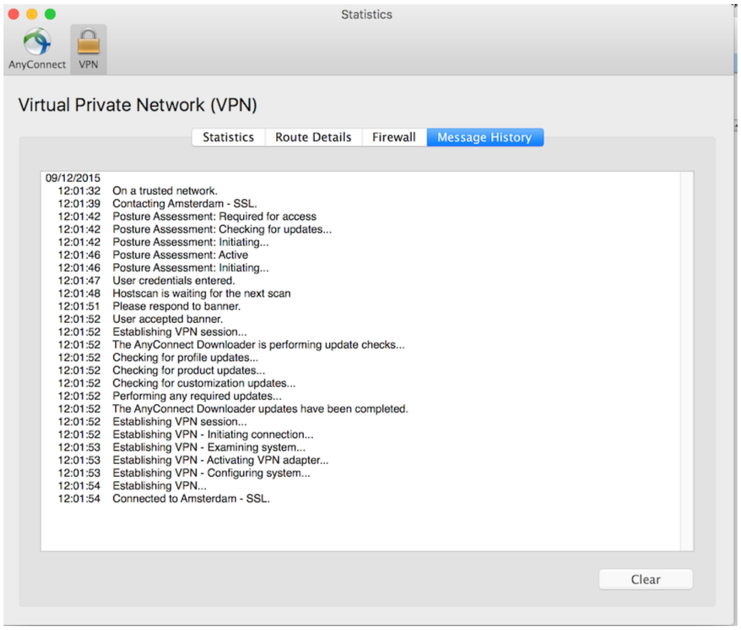
Remote Access User Connection Issues

SSL VPN connectivity issues usually fall into one of the following four categories:

- SSL VPN URL domain-name resolution issues (DNS)
- IP reachability issues
- SSL Certificate Issues
- SSL VPN user credentials

If Remote Access Users have problems connecting to the ASAv using AnyConnect client, the first step should be to capture the AnyConnect Message History. From Cisco AnyConnect menu go to the Statistics Window and check the Message History tab for error messages. The image below shows an example of a successful connection to SSL VPN.

Figure 10.2 Anyconnect Message History



```
09/12/2015
   12:01:32   On a trusted network.
   12:01:39   Contacting Amsterdam - SSL.
   12:01:42   Posture Assessment: Required for access
   12:01:42   Posture Assessment: Checking for updates...
   12:01:42   Posture Assessment: Initiating...
   12:01:46   Posture Assessment: Active
   12:01:46   Posture Assessment: Initiating...
   12:01:47   User credentials entered.
   12:01:48   Hostscan is waiting for the next scan
   12:01:51   Please respond to banner.
   12:01:52   User accepted banner.
   12:01:52   Establishing VPN session...
   12:01:52   The AnyConnect Downloader is performing update checks...
   12:01:52   Checking for profile updates...
   12:01:52   Checking for product updates...
   12:01:52   Checking for customization updates...
   12:01:52   Performing any required updates...
   12:01:52   The AnyConnect Downloader updates have been completed.
   12:01:52   Establishing VPN session...
   12:01:52   Establishing VPN - Initiating connection...
   12:01:53   Establishing VPN - Examining system...
   12:01:53   Establishing VPN - Activating VPN adapter...
   12:01:53   Establishing VPN - Configuring system...
   12:01:54   Establishing VPN...
   12:01:54   Connected to Amsterdam - SSL.
```

The log should help to diagnose the one of the issues below.

SSL VPN URL domain-name resolution issues (DNS)

The SSL VPN URL as displayed in the portal must resolve to a valid IP address. This can be validated using tools such as nslookup.

During service instantiation, NSO will register the ASAv outside IP address and hostname in DNS using the DNS Updater package (see NSO DNS Updater). If this step does not happen during service creation or the hostname registered by NSO to DNS does not match that URL displayed in the portal name,resolution will fail.

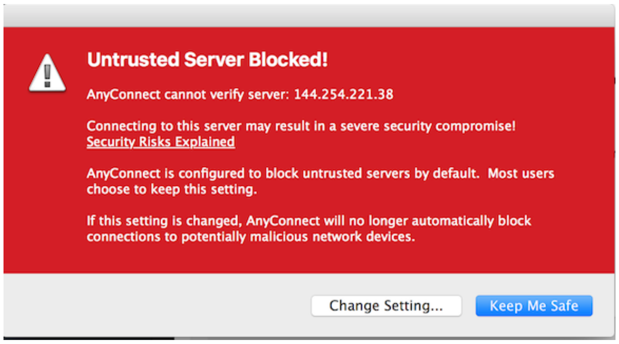
IP reachability issues

If DNS name resolution is successful and the ASAv is up but the resolved IP address is not reachable, then standard network reachability troubleshooting techniques should be applied.

SSL Certificate Issues

If there are problems with the SSL certificate installed on the ASAv the client will present the user with a security warning as shown below.

Figure 10.3 Anyconnect SSL Certificate Security Warning



Untrusted Server Blocked!

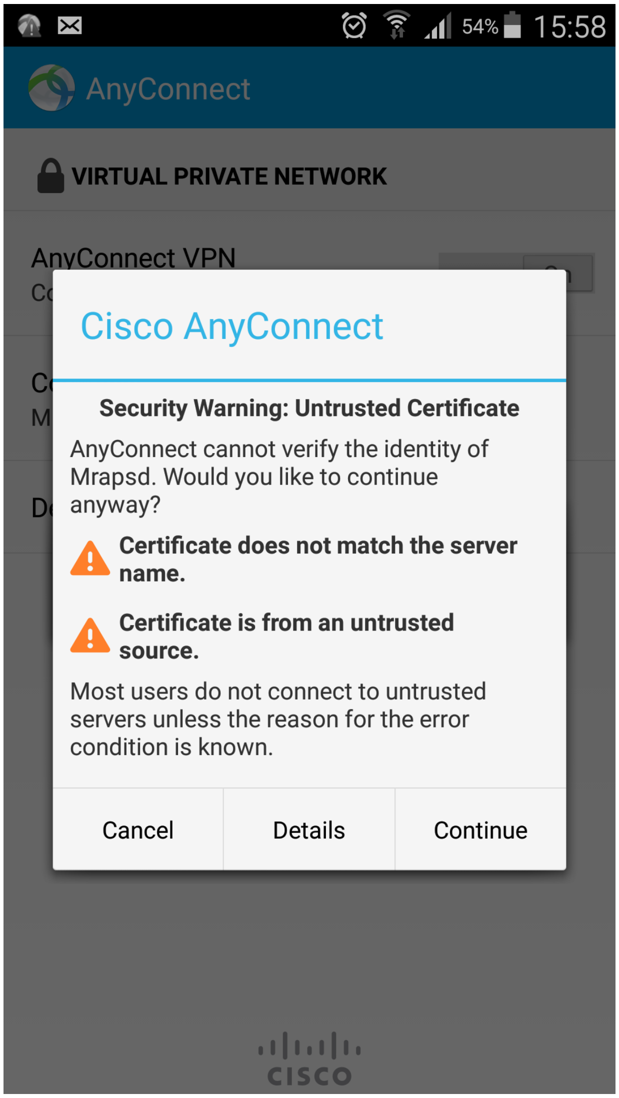
AnyConnect cannot verify server: 144.254.221.38

Connecting to this server may result in a severe security compromise!
Security Risks Explained

AnyConnect is configured to block untrusted servers by default. Most users choose to keep this setting.

If this setting is changed, AnyConnect will no longer automatically block connections to potentially malicious network devices.

Figure 10.4 Anyconnect SSL Certificate Security Warning on Android



Security Warning: Untrusted Certificate

AnyConnect cannot verify the identity of Mrapsd. Would you like to continue anyway?

Certificate does not match the server name.

Certificate is from an untrusted source.

Most users do not connect to untrusted servers unless the reason for the error condition is known.

The user can choose to ignore the security warning and attempt to continue the connection.

The ASAv SSL certificate is installed as part of the service Day 0 configuration by NSO (please see section "Global Variables" in Chapter 5.3 "NSO Configuration").

SSL VPN user credentials

When the AnyConnect client presents the user with a username and login prompt the user should enter the credentials as specified in the vMS Portal. If the user receives an authentication error check whether the user exists. User password can also be reset via the vMS portal.

Other AnyConnect Issues

For any other AnyConnect issues (including problems with the client itself) please consult the "AnyConnect VPN Client Troubleshooting Guide - Common Problems" at http://www.cisco.com/c/en/us/support/docs/security/asa-5500-x-series-next-generation-firewalls/100597-technote-anyconnect-00.html.