The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB’s public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS” WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

All printed copies and duplicate soft copies are considered un-Controlled copies and the original on-line version should be referred to for latest version.

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco website at www.cisco.com/go/offices.
Contents

Preface .................................................................................................................. 3
  Document Objective and Scope ........................................................................... 3
  Cisco Managed Services Accelerator Documentation ........................................ 3

Cisco MSX Solution Overview ............................................................................... 5
  Managed Services Accelerator Platform ............................................................. 6
    Service Integration Framework (SIF) ................................................................ 6
  Platform Integration Framework .......................................................................... 7
  Managed Services Accelerator Microservices ..................................................... 7
  Data Platform ..................................................................................................... 8
  Service Orchestration Platform ............................................................................ 9
  Managed Services Accelerator Service Packs .................................................... 10

Cisco MSX Solution Architecture ......................................................................... 13
  Modular Architecture .......................................................................................... 13
  Architecture and ETSI Mano Standards ............................................................. 14
    Cisco Network Services Orchestrator ................................................................. 15
      Cisco NSO as the Plug-and-Play Server ......................................................... 16
    Cisco Elastic Service Controller (ESC) ............................................................ 17
    Virtual Infrastructure Manager—OpenStack .................................................... 18

Cisco MSX Service Instantiation ......................................................................... 19
  Service Blueprints .............................................................................................. 19
  Create, Read, Update, and Delete Configuration Optimizations ........................... 19
  Service Device Mapping ..................................................................................... 20
  Configuring Infrastructure Elements .................................................................. 21
  A Summary ......................................................................................................... 22

Cisco MSX Service Interface ................................................................................. 25

Cisco MSX Service Packs ..................................................................................... 31
  vBranch Service Pack ....................................................................................... 31
    Cisco Enterprise Network Compute System .................................................... 32
      Supported VMs. ............................................................................................... 32
  SD-WAN Service Pack ........................................................................................ 33
  Managed Device Service Pack ............................................................................ 34
Preface

Cisco Managed Services Accelerator (MSX) solution changes ‘how’ managed services are instantiated by shifting the focus from device configuration to capturing the service designer intent by using the service models. The decoupling of service design from service instantiation allows service providers the ability to deploy MSX services with the knowledge these services can be extended with future releases without the need of a redesign.

This chapter contains the following topics:

- Document Objective and Scope
- Cisco Managed Services Accelerator Documentation

Document Objective and Scope

This guide provides a comprehensive explanation and guide to the design of the Cisco Managed Services Accelerator solution that enables service providers to offer flexible and extensible services to their business customers. This guide covers details on:

- MSX Service Packs, Service Ordering, and Service Orchestration
- MSX Service Interface—Front-end and Back-end
- MSX Platform Orchestration—NSO, ESC, and so on.

Cisco Managed Services Accelerator Documentation

The Cisco Managed Services Accelerator documentation set consists of:

- Cisco Managed Services Accelerator (MSX) Release Notes
- Cisco Managed Services Accelerator (MSX) Installation Guide
- Cisco Managed Services Accelerator (MSX) Platform User Guide
- Cisco Managed Services Accelerator (MSX) Platform and Service Packs Permission Addendum
- Cisco Managed Services Accelerator (MSX) SD-WAN Service Pack Guide
- Cisco Managed Services Accelerator (MSX) vBranch Service Pack Guide
- Cisco Managed Services Accelerator (MSX) Managed Device Service Pack Guide

Cisco Managed Services Accelerator documentation is available here:

Cisco MSX Solution Overview

Cisco MSX is a service creation and delivery platform that enables fast deployment of cloud-based networking services for both Enterprises and Service Providers customers. 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 which allows them to rapidly and profitably deliver service offerings to market. The service turn-up time is reduced from 4 to 6 weeks to a matter of a few days.

Cisco MSX also integrates with existing customer premise equipment, allowing service providers to build upon and utilize existing infrastructure. Cisco MSX provides a complete self-service user experience that allows to select, create, customize, and activate services on-demand in minutes from a simple self-service portal.

Cisco Managed Services Accelerator solution shifts the deployment of managed services away from the manual configuration of the latest network devices to the creation of a software abstraction to represent the service definition. This approach allows the service intent of the user, to be realized through the use of service models to automate the creation and customization of cloud-based services. These services are instantiated through Network Services Orchestrator (NSO). Through the advanced service orchestration capabilities of the NSO, an abstraction of the service exists as a YANG based service model, which is then processed by specialized Fastmap code, resulting in the instantiation of the service atop a virtual infrastructure.

Through the combination of a new MSX platform and service packs, the MSX solution offers a complete platform that allows the service providers to offer the next-generation Managed Services.

The following are some key highlights of Cisco MSX:

- Automated end-to-end cloud-based services managed from the service provider cloud
- Secure multi-tenant cloud managed platform, simplified orchestration, and tenant self-service
- Cloud-based services created with Zero Touch Provisioning
- Rapidly create new monetized services, modify existing services instantly from the cloud
- Perfect solution for distributed customers looking for lower cost and self-managed services
- Open, multi vendor, services catalog
- Supports physical devices and virtual functions
- Supports Cisco and 3rd party devices and VNFs
- Bring your own service or adapt existing services
- Can be integrated with existing OSS and BSS systems

The chapter has the following sections:

- Managed Services Accelerator Platform
- Managed Services Accelerator Service Packs
Managed Services Accelerator Platform

Cisco MSX is a service creation platform, implementing the different functions that are required to instantiate and provision virtual and physical elements in order to construct end to end managed services for service provider customers. The MSX solution uses software-defined networking (SDN), Network Functions Virtualization (NFV), Open APIs, and advanced orchestration capabilities to deliver a suite of business services through the service providers’ cloud infrastructure and over their existing network infrastructure.

The MSX platform is comprised into layers—each of which abstract the layer below it—and each layer may be scaled horizontally. A layered abstraction approach, with well documented API contracts between the layers, allows for modularization, a key tenet of the MSX platform. The modularization allows for the separation of concerns, independent scaling, development velocity, and ultimately component replacement, if necessary. The following illustration depicts the overall framework of MSX architecture.

Service Integration Framework (SIF)

The SIF layer enables external entities to consume the visibility and control that is provided by the platform functionality, optional capabilities, and use case-specific logic in the format they require.

---

**Figure 1** MSX Functional Architecture

**Figure 2** SIF Interactions
The SIF provides not only standard APIs required to operate the platform and use cases, but also supports the customization and extension of these APIs on per implementation or per customer basis. The ability of the SIF to be extended and customized to consume use case-specific functions is critical to the ability for MSX to support multiple use cases that are deployed in a service provider setup with reduced operationalization that is required beyond the initial use case.

The consumers of the SIF include:

- User Interface (UI)
- External Clients
- Service Packs

Platform Integration Framework

The MSX Platform Integration Framework (PIF) is a thin layer that sits north of the SOP and Data Platform. It provides a consistent interface to the SIF layer and external systems that need to interact with the MSX Platform via APIs.

The MSX PIF provides a set of REST end points for the MSX platform, shielding the consumer of the APIs from being aware of the actual platform components that provide the implementation.

For the consumers of these APIs, integration is made easier as only a single set of credentials are needed to use the APIs of multiple platform components. These platform APIs are for public consumption and are secured via authentication methods such as OAuth2.

Managed Services Accelerator Microservices

All of SIF functions and integrations are enabled as microservices that contain the logic to consume the functions that are exposed by the platform and service packs. There is a standard set of microservices that consumes the base platform functionality that is required to operate the platform.

Each use case can also provide one or more microservices for any additional use case-specific functionality that needs to be enabled.

The following core microservices exist in the SIF layer of the MSX platform:

- Identity Management—Provides user management capabilities. Upon login, fetches the user profile and stores it in the common cache so other micro-services can use it.
- Consume—Provides capability to create services, service offer, price plan, define terms and conditions of service, submit an order, upgrade, or downgrade a service.
- Manage—Provides the service management capabilities, such as managing customer subscriptions and device data.
- Monitor—Communicates with the Service Assurance components for device and service status and displays the statistics on the Portal.
- Orchestration—Communicates with NSO to process the request to create or upgrade a service chain, add or delete a device, register device serial numbers, and advanced device configurations.
- Notification—Sends e-mail notifications.
- Logger—Collects the log from all the microservices.
- Administration—Provides capability to create or update a tenant, update provider name, and contact information, set notification preferences, update terms and conditions for a service, and so on.
Data Platform

The data platform interface is the mechanism by which the data platform component interacts with the other platform components and applications. The major function that is performed by the data platform component is the collection of data. The collection interface allows a platform component or application the ability to instruct the data platform which data to collect, the frequency with which to collect, and how to store and correlate the data. This is to enable other platform components or external systems to augment the collection that is enabled by the SOP provisioning. For example, the instantiation of a typical service chain programs the devices participating in the chain to send data (Syslog/SNMP) to the data platform.

There are a number of actions that must be supported by the data platform component, and thus exposed via the service interface. The actions include, but are not limited to:

- Data Retrieval (Pull)
- Configurable Collection
- Data Subscription/Streaming (Triggers/Events)

The data platform component of the architecture is broken down into the following major sub-functions, as shown in the following illustration.
Service Orchestration Platform

The Service Orchestration Platform (SOP) orchestrates a service end to end across multiple domains and creates a service chain that is based on service intent. It contains the logic that brings virtual and physical devices under management and deploys the required service configurations necessary to bring service and service topologies online and under management.

The Orchestration platform contains of a multi-vendor model driven Cisco Network Service Orchestrator (NSO), and an automated Virtual Network Function Lifecycle Manager (VNFM) and represents the resource facing services (RFS).

The following illustration depicts the orchestration platform for MSX along with the major roles of the components. Also, the interactions with the platform interface, VIM, and data platform are highlighted to further illustrate their relationships.
Managed Services Accelerator Service Packs

MSX service packs contain all the elements necessary to orchestrate a service that is requested by the business customer or tenant operator. The operator initiates the service request to Cisco NSO through the MSX Portal or an open API call from the existing Provider OSS/BSS systems.

Figure 6 shows a detailed view of the service pack elements. The service definition model is written in Yang and describes the end-to-end service. A service definition model is referred to as a ‘Virto’. The Yang service model software has validation logic that validates the service requests to ensure that incorrect service requests are not completed.

The Service Model must be mapped to Device Models, which generate the service, and device configurations that are applied the physical and virtual infrastructure devices. Cisco NSO software uses innovative Fastmap functionality to handle this process. This mapping with Fastmap software could be accomplished through a template or using Java for more complex mapping applications.
Device models also use Yang modeling constructs but are intended to model the infrastructure rather than the service. This set of device models is what is referred to as the underlay and is an abstraction of only the physical and virtual infrastructure that must be configured to enable the requested service.

For more information on MSX service packs, see Cisco MSX Service Packs.
Cisco MSX Solution Architecture

This chapter provides an insight on how the orchestration engine processes a service request, using service definition models to create and instantiate cloud-based services.

This chapter contains the following sections:

- Modular Architecture
- Architecture and ETSI Mano Standards
- Virtual Infrastructure Manager—OpenStack

Modular Architecture

Service providers can automate the creation of cloud based business services by using YANG-based service models in the orchestration engine software modules, and service function chaining of the required virtual network functions (VNFs) to enable those services.

As MSX evolves the architecture continues to modularize components, creating a clear demarcation points between various layers in the solution allowing maximum flexibility in both commercial and deployment models.

MSX Modules

Service Interface (SI)—Customer Facing Level

The Service Interface consists of two modules, a customer facing Front-End and service intent to service request processing Back-End.

The SI Front-End is the Self-Service User Interface; from here the end-customer can select the Service Package that meets their VNF requirements. At that point, the end-customer can construct MSX services consistent with the Service Package, in this example, the Advance Service Package is selected.

The SI Back-End is separate module that constructs the service request based on the service intent that is determined by the end-customer choices in the From-End module.

MS Platform—Resource Facing Level

The MSX Platform mirrors the ETSI MANO Standard for a virtualization orchestration model. The NFV Manager or Network Service Orchestrator provisions services that is based on service packs that logically mirror CFS level Service Packages. Service packs are internal software modules that house the service models and Fastmap execution logic that define specific Service Packages. Based on these service models, the NSO uses the internal Fastmap code to provision the service components through the Elastic Service Controller (ESC).

The ESC is the VNF Lifecycle Manager, based in request from the NSO, this component will execute Create, Read, Update, and Destroy (CRUD) operations, resulting in the operation request to the Virtual Infrastructure Manager.
Architecture and ETSI Mano Standards

The ETSI NFV Management and Orchestration (MANO) standards define a method of orchestration whereby the components of orchestration of compartmentalized to specific functions. Figure 7 shows the ETSI MANO model. The layers are broken into Orchestrator (NFVO), Life Cycle Management (VNF-M), and virtual interface management (VIM). The NFV orchestrator is responsible for understanding the overall service being instantiated. The life cycle management is for the management and monitoring of virtual network functions (VNFs). Finally, the VIM handles interconnecting physically and virtual network and compute infrastructure.

**Figure 7**  MSX Uses ETSI Mano Architecture for Orchestration

Cisco Managed Services Accelerator solution follows the ETS MANO model as show in the Figure 8. The Cisco Network Services Orchestrator (NSO) orchestration engine software modules handle the NFVO functions. The Elastic Services Controller (ESC) software modules are responsible for VNF life cycle management (VNF-M). OpenStack networking software plug-ins modules provide virtual infrastructure management (VIM) functionality.

**Figure 8**  MSX Orchestration Component Mapping
Cisco Network Services Orchestrator

Cisco Network Services Orchestrator (NSO) is a model-driven orchestrator which uses YANG for modeling the services, and can use various methods such as NETCONF, SSH, REST, and APIs to provision the devices. NSO receives a service intent request through the open API interface presented northbound to the service interface (or customer OSS/BSS).

The Service Manager software subcomponent of NSO processes the service intent, corresponding to a particular service model. The purpose of the Service Manager is to analyze the service request, then interpret the request to a specific set of configuration asks. Similarly, the Device Manager is used to abstract out the infrastructure that is also based on a modeling language. A Mapping Controller is responsible for moving service models to device models. This is discussed in more detail in later sections.

A transaction database that is known as the “CDB” stores all elements related to configuration and devices. For all services intended to be instantiated, NSO has a Yang service definition model that is loaded into the transaction database to handle such a request.

Network Element Driver (NED) software modules are used to abstract out the different physical and network devices to which service configuration data may be pushed. As such the NED allows the service definition models to apply to equipment from different vendors.

NSO runs as a container in the MSX solution. The NSO container is deployed in one of the Kubernetes nodes and monitored by the Kubernetes master. Each Service pack has an NSO container.

Figure 9 shows the individual service intent modules.

Figure 9  MSX VNF-Orchestrator—NSO

Cisco NSO as the Plug-and-Play Server

The Plug-and-Play (PnP) Server software element is used to handle and achieve zero touch deployment (ZTD) for CPEs coming online and wanting to utilize the services configured orchestration engine. In the MSX solution, NSO also functions as the PnP server. The site CPE deployments are supported in MSX by a PnP service, which is accessible as an NSO PnP Resource Facing Service (RFS) API. Once a CPE device is connected to the Internet, a “call home” protocol communicates with the NSO PnP RFS API via a secure PnP Configuration Manager that is located in the Service Provider MSX Cloud. The CPE device is fully configure using a four-step process:
Cisco MSX Solution Architecture

- Day-(-1) — Initial config of CPE to find PnP Server
- Day-0 — Configuration of the device management interface
- Day-1 — System configuration, basic interface and system configuration
- Day-2 — Service configuration, service-specific configuration

Inclusion of the PnP service in the MSX Solution removes costly truck-rolls from the service deployment model, which are required to install and configure each CPE device. The removal of this activity greatly reduces the cost of service deployment, a cost that has long haunted traditional Managed Services.

The following illustration depicts the device registration process.

**Figure 10 Registering a Device or CPE**

- **1** New device is powered on and gets IP address and internet connectivity from ISP.
- **2** New device invokes web service API call to the PnP server and registers its UDI (serial number). Management channel is established.
- **3** User activates desired device (branch or hub router).
- **4** PnP server matches serial numbers and downloads the configuration.

---

**Cisco Elastic Service Controller (ESC)**

The elastic services controller software module provides the automation of life cycle management for any VNFs that a service needs to have instantiated—also known as the VNF-M in ETSI MANO terms. There are three main functions achieved through the ESC software module including service provisioning, a rules engine, and service monitoring (see Figure 11).
The service provisioning function takes its VNF instantiation actions from NSO requests which are pushed down via a NED software process into the ESC. The purpose of ESC software module is to bring VNF devices online and be network reachable and configurable from the NSO software modules. The ESC software module use calls to the OpenStack networking software modules to accomplish the tasks of launching virtual machines. Specifically, the ESC software will call the OpenStack Nova software process for compute launch and the OpenStack Neutron software process to establish network connectivity to the newly created virtual machines. The OpenStack VIM software has no knowledge of the overall service being instantiated. OpenStack software is merely acting on the requests pushed by ESC software. All external networks must be created in OpenStack prior to the NSO instantiating the service requests.

The rules engine drives a critical feature of the ESC software, elasticity. Elasticity refers to the ability to spin up additional VNFs that may be required from increasing service load. Conversely elasticity also includes removing VNFs from service when they are no longer needed. In this release, the elasticity configuration is locked at “1” in the ESC software schema. This will prevent the VNFs from scaling up or down. The configurability of the ESC software rules engine elasticity factor will be enabled in subsequent releases.

Service monitoring is essential for detecting whether a VNF has stopped operating or is no longer functioning as expected per the service level agreement (SLA). The ESC software module uses various mechanisms such as SNMP and Ganglia, to detect that a VNF is acting out of profile. When this occurs, the ESC software restarts the VNF, maintaining the user license, and potentially being moved to a new compute resource if necessary. The VNF service monitoring information may also be sent as traps northbound via Open APIs to service assurance tools.

**Virtual Infrastructure Manager—OpenStack**

The MSX platform assumes the Mitaka version of OpenStack as a Virtual Infrastructure Manager (VIM). The Elastic Service Controller will communicate with OpenStack Nova compute services and Neutron network service to provision Cloud VPN VNF types and create the network connectivity. OpenStack is implemented as a series of controller services, these services are installed independent of MSX and requires dedicated servers.

For provisioning of VNF, the ESC communicates through a Nova REST Interface to provision a VM based on a particular VNF Type resources and image requirements. Provisioning of the virtual interfaces, Linux Bridges and OVS segments, the ESC communicates through a Neutron REST Interface. Lastly, VNF lifecycle operations utilize the same REST interfaces.
Cisco MSX Service Instantiation

The key differentiator of Cisco Managed Services Accelerator (MSX) is the orchestration and management of services using a deterministic and repeatable method, resulting in the consistent instantiation of a service. MSX, by using the service packs creates a consistent and well-formed service request, is able to instantiate a service based on a well-defined service model and associated execution code. Each instantiated service will share common feature configuration and service topology.

This chapter contains the following sections:

- Service Blueprints
- Create, Read, Update, and Delete Configuration Optimizations
- Service Device Mapping
- Configuring Infrastructure Elements
- A Summary

Service Blueprints

Service blueprints are essentially the set of predefined service packages developed to render custom services over physical and virtual infrastructure. These service blueprints are made available as service packs on the MSX Portal. These packages live in NSO software modules and have subcomponents. These subcomponents are the Service Model, Mapping Code, Device Model, and Network Element Drivers (NEDs). All of these components are required to instantiate service intention. When the elements of a service pack are compiled by the service developer, Cisco MSX solution automatically creates the APIs necessary for a northbound application (such as a portal) to request a given service definition model ‘blueprint’. For example, the service interface portal may offer a blueprint for a Virtual Router service connected to a CPE device.

Create, Read, Update, and Delete Configuration Optimizations

Create, Read, Update, Delete (CRUD) operations are at the heart of any services orchestration system. Typically these are achieved through complex workflow design. The Cisco NSO orchestrator software models provide a unique approach to solving this complexity issue. Hardened software processes enabled by Tail-F, service developers are only required to write the service create functions. Cisco NSO software automatically calculates all functions necessary to carry out the Read, Update, and Delete functions.

In Figure 12, a Venn diagram shows this operations concept. When a service request is made, Cisco NSO software will examine the requested service against any existing service currently deployed. A change set is then determined that represents the delta between the two service model definitions in the transaction database (CDB). Cisco NSO is capable of deriving all the actions necessary to move that new change of the service or device model set into operation. The transactional database (CDB) software allows the change sets to be unrolled either automatically if a service fails, or through the operator request back to a previous stable state.
Service Device Mapping

The purpose of the Fastmap software in the service pack of a service blueprint is to map service intent to the device infrastructure. The Fastmap software process uses Java logic resulting in the creation of a service template that is mapped to a device model. This process works well for physical infrastructure. But what if the infrastructure does not yet exist, as is the case with virtual machines (MSX) running virtual network functions (VNFs). In these conditions, Cisco NSO can make use of the Reactive Fastmap software process.

The Reactive Fastmap software is capable of detecting when a service model requires a virtual network function (VNF) in the requested service model. Cisco NSO cannot complete service model mapping until all devices, both physical and virtual, are active. However, the Transactional Database (CDB) software cannot remain locked while the virtual devices are started.

In the case of the required VNF, NSO calls the Elastic Service Controller (ESC) software modules to handle the VNF life-cycle management. The Fastmap software process subscribes the service to an event in the transaction database indicating a VNF has been started and brought under management. When this occurs, Cisco NSO software attempts to redeploy the service model requests. Here the entire service request process begins again with Cisco NSO software checking that all devices are in a ready state. If all physical and virtual devices are not ready, the Cisco NSO software modules defer the service request again.

Eventually either the service deployment fails because all devices cannot be brought into a ready state or all required components become fully available. In the case where the devices do not all appear in the ready state, the service request fails and all potential configurations are rolled back. When Cisco NSO detects that all devices are in a ready state, the service request process proceeds to map the service model to the appropriate device model. This entire process is shown in Figure 13.
Cisco MSX Service Instantiation

Configuring Infrastructure Elements

An underlay represents the physical and virtual infrastructure to which cloud-based services will be orchestrated. An underlay consists of physical and virtual devices, links, network bridges, and resource pools. Cisco NSO software is capable of supporting the loading of multiple infrastructure topologies into the transactional database (CDB).

The entire infrastructure is typically not necessary to instantiate every service. A service typically requires a subset of a given infrastructure. Cisco NSO software implements a mechanism known as the GraphDB, which is a tree representation of all the elements in the infrastructure. GraphDB software in Cisco NSO allows the Fastmap software processes to use queries based on the service model to find the applicable infrastructure required for the service.

To influence the result of the GraphDB query, Tags and Capabilities service requirements are programmed into the service request. Tags provide a description in the service model to which the Fastmap can parse that provides data to the mapping logic. Similarly, capabilities in the service definition model describe what is required of the infrastructure components for that particular service. The process of querying the GraphDB software modules is illustrated in Figure 14.
A Summary

Figure 15 is a complete representation of the advanced orchestration processes required to on-board a service. The first step is to ensure the Service Models required have been loaded into the Cisco NSO transaction database (CDB). The service model configurations that make up the service pack are written specifically to offer the required service. A service request received by Cisco NSO software results in configuration checks against the service model which is handed off to the Fastmap process in Step 2. At this point GraphDB software is queried to retrieve only the infrastructure necessary to realize the specific service. Tags and Capabilities parameters in the service request will influence the service query. The output of the query request is the Service Overlay illustrated in Step 3. Based on the completion of these steps, Cisco NSO software attempts to instantiate and create a cloud-based service comprising of both physical and virtual components.
The reactive Fastmap software in Cisco NSO makes an external call to the Elastic Service Controller (ESC) software to spin up the required MSX in Step 4. ESC understands the affinity rules for launching the service, working with OpenStack VIM software (Nova, Neutron) to launch the necessary compute resources and bring up networking interfaces that enable the VMs to be manageable (Step 5). Cisco NSO software enables the initial minimal configurations of the newly launched VMs that are required to connect the VNFs to the management channel of Cisco NSO software.

During the process of launching the VMs, Cisco NSO software will release control of the database for other functions and wait (Step 6) until the VMs appear to be in the ready state. As with a CPE, a VM is online and ready once it completes an online service call-back process (Step 7) to put the newly started device into the Cisco NSO transactional database. Cisco NSO software will attempt to redeploy the service each time a service callback is made. However, no changes will be committed until all devices in the model register as ready in the transaction database (CDB).

Cisco NSO attempts to reach the physical and virtual components in the service overlay via SSH connectivity over the management network interface (Step 8). The service model configuration applied by Cisco NSO is referred to as the Day 1 configuration. When all configurations to all physical and virtual devices are complete then the specific service is considered deployed. If any of the configurations fail, then the service and device model changes applied in the Fastmap or Reactive Map processes are rolled back to the last known working configurations in the transaction database (CDB).
Cisco MSX Service Interface

The Cisco Managed Services Accelerator (MSX) service interface captures the service intent of the customer and requests service to Cisco Network Service Orchestrator (NSO). The Service Interface is composed of two subsystems—the web portal (front-end) and the back-end.

The web portal is designed as an all-in-one web-based solution GUI. Based on the user login type, the user will be presented with one of three service interfaces: administer, operator, or user. With a successful authentication, the user is directed to the web interface, based on the predefined role of the username.

The Service Interface, based on the user role, will allow the Service Provider administrator to provision tenant space for end users, while the Service Provider operator can view the status of all services running, and lastly the user role is for the Service Provider end-customer to order the service based on their requirements.

Figure 16  MSX Service Interface: Login Screen

Using this portal provider administrator can create and manage tenants (end customers).
Through the Service catalogue in the portal, end customers or tenants can order, manage, and monitor their services. Using this portal an operator or an administrator can also view service status and statistics for the deployed services.
The back-end is the composition of micro-services that together communicate with various components in the MSX Solution. The back-end processes the service intents that you input via the web interface or self-service portal, and creates parametrized service requests to be sent to the MSX platform NSO.

The MSX Services that are available through the portals are dependent on the Service Packages made available by the Service Provider; currently MSX provides the vBranch, SDWAN, and Managed Device service packages. The MSX Service Interface utilizes an OpenAM ID system to determine customer identity.
The Service Interface back-end communicates with the web portal through a REST API Gateway. The interface portals of the web portal rely on back-end micro-services to process user data entered in the various interface portal screens. Dependent on the type of interface portal and data entered, the information will be sent to/from the back-end API and delivered to/sent from the micro-service responsible for processing the incoming data. The back-end micro-services are responsible for multiple functions listed below; individual micro-services communicate with MSX modules or other OSS modules to fulfill their functions.

The portal subsystem is built on a micro services framework. Different types of microservices are listed below:

- **Business functions microservices**
  - Identity Management provides authentication and authorization. (Authorization is the component that defines what you have access to. Federation with other identity engines is possible, but requires integration.)
  - Consume (Catalog)
  - Manage (Service Provisioning and Life-Cycle)
  - Monitor (Service Status and Metrics)
  - Orchestration (NSO RFS Integrations)

- **Technical Microservices**
  - Notification (Email Templates)
  - Logger

- **Microservices-based**
  - Scalable, swappable and extensible (DevOps Enablement)
  - Each service can be deployed independently of other services - easier to deploy new versions of services frequently
  - Each microservice is relatively small (Easier for a developer to understand)
  - Improved fault isolation. For example, if there is a memory leak in one service then only that service will be affected
  - Each service can be developed and deployed independently
  - Eliminates any long-term commitment to a technology stack

- **Robust and well documented REST APIs to enable custom app development**
  - Use our out-of-box UI or build your own
- Supports heterogeneous identity providers (IDM)
Cisco MSX Service Packs

A central theme behind the Managed Services Accelerator (MSX) solution is reducing the operational cost of deploying and maintaining service provider-based managed services. The solution shifts the deployment of managed services away from the manual configuration of the latest network devices to the creation of a software abstraction to represent the service definition. This approach allows the service intent of the user, to be realized by using the service models to automate the creation and customization of cloud-based services.

MSX is a next generation managed service solution for service providers who are interested in hosting enterprise connectivity and security features in their cloud infrastructure. The keys to the MSX solution are virtualization, plug-n-play CPE devices, and a flexible orchestration engine capable of centralizing the configurations of all the devices involved in the delivery of a service. With flexibility inherent in the orchestration engine, service providers can offer end customers the ability to order the service that best meets their technical and TCO requirements.

The MSX architecture is based on a service package infrastructure. MSX service packages are bundled Virtual Network Function (VNF) types, the type of services available that are tightly coupled to the VNF types included in a specific service package. The end customer, based on a service provider deployment, has a choice based on services that can be orchestrated given the VNF types, which are included in the service package bundle.

This chapter contains the following sections:

- vBranch Service Pack
- SD-WAN Service Pack
- Managed Device Service Pack

vBranch Service Pack

Cisco MSX vBranch service pack enables unified routing, switching, storage, processing, and a host of other computing and networking activities into a single box. The vBranch service pack provides a way to collapse the services that a branch requires into a single box, which results in easier management of services, and smaller device footprint on a branch site.

The MSX vBranch service pack includes the following:

- An orchestration environment to allow automation of virtualized network service deployment, consisting of multiple Virtualized Network Functions (VNF).
- VNFs, which provide the desired network functionality, or even non-networking software applications, required at a deployment location.
- The NFV Infrastructure Software platform to facilitate the deployment and operation of VNFs and hardware components.
Cisco MSX Service Packs

The figure below illustrates the functional architecture of a vBranch site.

Some of the advantages of the MSX vBranch service pack are:

- Zero touch provisioning for initial device connectivity through PnP server processes.
- Service provisioning of on-premise CPEs through orchestration.
- User interface portal for ordering service, network visualization, and performance or fault monitoring.
- Lifecycle Management.

vBranch supports the branch site on Cisco 5000 Enterprise Network Compute System (ENCS) platform.

Cisco Enterprise Network Compute System

The Cisco 5000 Enterprise Network Compute System (ENCS) is a line of compute appliances designed for the Cisco Enterprise Network Functions Virtualization (ENFV) solution. It delivers a new standard of software-defined flexibility and performance, and offers a low Total Cost of Ownership (TCO). The 5000 ENCS is a hybrid platform that combines the best attributes of a traditional router and a traditional server, and offers the same functionality with a smaller infrastructure footprint. Offered with the Cisco Integrated Services Virtual Router (ISRv) and NFV Infrastructure Software (NFVIS) as the hosting layer, the platform offers a complete solution for a simplified deployment.

Supported VMs

Currently, the following Cisco supplied VMs and third-party VMs are supported on Cisco ENCS:

- **Cisco Integrated Services Virtual Router (ISRv)**—A virtual form-factor of the Cisco IOS XE software router that delivers WAN gateway and network services functions into virtual environments.

- **Cisco Adaptive Security Virtual Appliance (ASAv)**—Enables ASA firewall and VPN capabilities on virtualized environments to safeguard traffic and multitenant architectures. Optimized for data center deployments, it is designed to work in multiple hypervisor environments, reduce administrative overhead, and increase operational efficiency.
Cisco Virtual Wide Area Application Services (vWAAS)—A virtual appliance that accelerates business applications delivered from private and virtual private cloud infrastructure. Cisco vWAAS enables you to rapidly create WAN optimization services with minimal network configuration or disruption.

Virtual Wireless LAN Controller (vWLC)—Virtual form-factor controller for any x86 server with VMware Hypervisor ESXi.

3rd Party VNFs—Third-Party VNFs.

---

SD-WAN Service Pack

Cisco SD-WAN service pack enables service providers to deploy and manage SD-WAN service for their customers. The deployment of an SD-WAN service in the context of a managed service requires deployment per customer and includes the SD-WAN management control plane (vManage, vBond and vSmart), and the corresponding data plane (vEdge).

The SD-WAN service pack management control plane and data plane consists of:

- **vManage**—The vManage is a centralized dashboard that enables automatic configuration, management, and monitoring of the overlay network. Users log in to vManage to centrally manage all aspects of the network life cycle—from initial deployment, on-going monitoring and troubleshooting, to change control and software upgrades.

- **vBond**—The vBond facilitates the initial bring-up by performing initial authentication and authorization of all elements into the network. vBond provides the information on how each of the components connects to other components. It plays an important role in enabling devices that sit behind the NAT to communicate with the network.

- **vSmart Controller**—The vSmart controllers establish the secure SSL connections to all other components in the network, and run an Overlay Management Protocol (OMP) to exchange routing, security, and policy information. The centralized policy engine in vSmart provides policy constructs to manipulate routing information, access control, segmentation, extranets, and service chaining.

- **vEdge Router**—The vEdge router establishes secure connectivity to all of the control components and also establishes IPSec sessions with other vEdge routers in the WAN network. In the MSX SD-WAN 3.1.1, you can deploy a customer site on Cisco 5000 Enterprise Network Compute System (ENCS) platform.

The Cisco 5000 Enterprise Network Compute System (ENCS) is a line of compute appliances designed for the Cisco Enterprise Network Functions Virtualization (ENFV) solution. Cisco 5000 ENCS is a hybrid platform that combines the best attributes of a traditional router and a traditional server, and offers the same functionality with a smaller infrastructure footprint.

Some of the advantages of the MSX SD-WAN service pack are:
Provides the interface to associate the tenant (customer) with the Control Plane and Data Plane.

User interface portal for ordering service (Control Plane and Data Plane Connectivity) and network visualization.

Lifecycle management of services.

Managed Device Service Pack

Cisco MSX Managed Device service pack enables service providers to provide their customers manage devices services through a self-service portal. With Managed Device service pack, IT organizations can bring into its network (on-board) devices located at the customer premise (CPEs) and apply or manage configuration settings remotely from its Network Operations Center (NOC). The service provider can configure parameterized configuration template that needs to be deployed on these CPEs.

MSX Managed Device service pack makes device deployment fast and easy. Using this service pack user interface you can configure and deploy MSX CPEs.

Some of the advantages are as follows:

Zero touch provisioning for initial device connectivity through PnP server processes.

Service provisioning of on-premise routers through NSO orchestration.

User Interface portal for configuration templates, ordering service, and performance or fault monitoring.

Cisco MSX Managed Device supports different devices. The list of supported device models are:

- Cisco Cloud Services Router (CSR) 1000V
- Cisco 829 Industrial Integrated Services Routers (IR829)
- Cisco 1100 Integrated Services Routers (ISR)
- Cisco 4000 Series Integrated Services Routers (ISR):
  - ISR 4221
  - ISR 4321
  - ISR 4331
  - ISR 4351
  - ISR 4431
  - ISR 4451
Cisco MSX Service Extensions

This chapter provides an overview of Managed Services Accelerator (MSX) service extensions and contains the following topics:

- Understanding How Service Extensions Work, page 35
- Creating a Service Extension Template XML File, page 36

Understanding How Service Extensions Work

Cisco MSX service extensions simplify how configuration snippets can be applied to a service or a device. MSX leverages the underlying capability of Cisco Network Services Orchestrator (NSO) custom templates, which get pushed along with the derived configuration templates. Service extensions can be used, in most cases, to map services to device configurations, without the need for any additional programming.

MSX service extension templates use variables to map service attributes to the corresponding device configurations and are applied. The service extensions allow a declarative way to describe such manipulations. The MSX operator can apply a service extension template to an existing service chain in MSX or to a device, without having to manually go into the NSO CLI. This service extension template is used by NSO to add, modify, or delete service configuration snippets before NSO pushes the configuration to the devices.

You can apply MSX service extension templates to a service ordering workflow or a single device. When you import a service extension template into MSX, you can specify if the template is to be applied to a service workflow or a device.

When a service extension template is applied to service ordering workflow, MSX service workflow gathers the parameter values the tenant users enter during service ordering process. These values are passed to NSO, which further uses these values in the device configurations.

When a service extension template is applied to a device after the service is ordered, the service extension template is applied to the device outside the workflow, and is available for use for individual devices.

The following illustration depicts the end-to-end workflow that needs to be followed to work with service extension templates in MSX.
Creating a Service Extension Template XML File

The MSX service extension template is an XML file. The structure of that file is defined by the YANG model.

The basic principles of defining a template are as following:

1. A template is an XML file (for example mytemplate.xml) that corresponds to a node in the device tree.
2. Each value in a template is stored as a string. This string value is converted to the actual value type of the YANG model when the template is applied.
3. The value part of the XML tag that needs to be configured must be represented with a variable name prefixed with ‘$’ literal.
4. The templates allow for defining different behavior while applying the template. This is accomplished by setting tags such as merge, replace, delete, create or nocreate on the relevant nodes in the template.

For example, to create a template to set the NTP server on a device, the sample template XML file should be:

```xml
<config
   xmlns="http://tail-f.com/ns/config/1.0">
  <device
     xmlns="http://tail-f.com/ns/ncs">
    <name>ntp</name>
    <config tags="merge">
      <system xmlns="http://pica8.org/yang">
        <ntp-server-ip>{$NTP}</ntp-server-ip>
      </system>
    </config>
  </device>
</config>
```
Creating a Service Extension Template XML File

```xml
<server>
  <server-list>
    <ip-address>{$NTP}</ip-address>
  </server-list>
</server>
</ntp>
</config>
</devices>
</config>
```

After you create a service extension template, you need to do the following:

- Import the template XML file into NSO. For details see, the service pack guides.
- Import the template XML file into MSX. For details see, the service pack guides.
Creating a Service Extension Template XML File
# Terminology

The table below provides an alphabetical listing of MSX acronyms.

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAA</td>
<td>Authentication, Authorization, and Accounting</td>
</tr>
<tr>
<td>ACL</td>
<td>Access List</td>
</tr>
<tr>
<td>API</td>
<td>Application Programmable Interface</td>
</tr>
<tr>
<td>ARPU</td>
<td>Average Revenue Per User</td>
</tr>
<tr>
<td>ASAv</td>
<td>Adaptive Security Appliance - vFW VNF</td>
</tr>
<tr>
<td>BGP</td>
<td>Border Gateway Protocol</td>
</tr>
<tr>
<td>BSS</td>
<td>Business Support System</td>
</tr>
<tr>
<td>CDB</td>
<td>Configuration Database</td>
</tr>
<tr>
<td>CFS</td>
<td>Consumer Facing Service</td>
</tr>
<tr>
<td>CLI</td>
<td>Configuration Line Interface</td>
</tr>
<tr>
<td>CMSP</td>
<td>Cisco Cloud and Managed Service Program</td>
</tr>
<tr>
<td>CPE</td>
<td>Customer Premise Equipment (ISR-G2)</td>
</tr>
<tr>
<td>CRUD</td>
<td>Create, Read, Update, and Delete orchestration operations</td>
</tr>
<tr>
<td>CSR</td>
<td>Cloud Service Router - vRouter VNF</td>
</tr>
<tr>
<td>cURL</td>
<td>Client URL</td>
</tr>
<tr>
<td>DC</td>
<td>Data Center</td>
</tr>
<tr>
<td>DCI</td>
<td>Data Center Interconnect</td>
</tr>
<tr>
<td>DEST</td>
<td>Destination IP Address</td>
</tr>
<tr>
<td>DHCP</td>
<td>Dynamic Host Configuration Protocol</td>
</tr>
<tr>
<td>DMVPN</td>
<td>Dynamic Multi-Point Virtual Private Network</td>
</tr>
<tr>
<td>DMZ</td>
<td>Demilitarized Zone (Networking private to public)</td>
</tr>
<tr>
<td>DNS</td>
<td>Domain Name System</td>
</tr>
<tr>
<td>DST</td>
<td>Destination IP Address</td>
</tr>
<tr>
<td>EIGRP</td>
<td>Enhanced Interior Gateway Routing Protocol</td>
</tr>
<tr>
<td>ESC</td>
<td>Elastic Services Controller</td>
</tr>
<tr>
<td>ETS</td>
<td>European Telecommunications Standards</td>
</tr>
<tr>
<td>ETSI</td>
<td>European Telecommunications Standards Institute</td>
</tr>
<tr>
<td>EzVPN</td>
<td>Easy Virtual Private Network</td>
</tr>
<tr>
<td>FW</td>
<td>Firewall</td>
</tr>
<tr>
<td>GraphDB</td>
<td>Graph Data Base</td>
</tr>
<tr>
<td>GuestOS</td>
<td>Guest Operating System</td>
</tr>
<tr>
<td>Acronym</td>
<td>Definition</td>
</tr>
<tr>
<td>---------</td>
<td>------------</td>
</tr>
<tr>
<td>HA</td>
<td>High Availability is</td>
</tr>
<tr>
<td>HDR</td>
<td>Header</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol</td>
</tr>
<tr>
<td>I2RS</td>
<td>Interface to Routing System (diagram format)</td>
</tr>
<tr>
<td>IKEv2</td>
<td>Internet Key Exchange version 2</td>
</tr>
<tr>
<td>IO</td>
<td>Input/Output</td>
</tr>
<tr>
<td>Intrusion Detection</td>
<td>Generally refers to the process of passively analyzing network traffic for potential intrusions and storing attack data for security analysis.</td>
</tr>
<tr>
<td>Intrusion Prevention</td>
<td>Includes the concept of intrusion detection, but adds the ability to block or alter malicious traffic as it travels across your network</td>
</tr>
<tr>
<td>iOS</td>
<td>Cisco Operating System</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>IPSec</td>
<td>Internet Protocol Secure</td>
</tr>
<tr>
<td>ISR</td>
<td>Integrated Service Router</td>
</tr>
<tr>
<td>IT</td>
<td>Information Technology</td>
</tr>
<tr>
<td>KVM</td>
<td>Kernel-based Virtual Machine</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>ML2</td>
<td>Modular Layer 2</td>
</tr>
<tr>
<td>MPLS</td>
<td>Multiprotocol Label Switching</td>
</tr>
<tr>
<td>MSX</td>
<td>Managed Services Accelerator</td>
</tr>
<tr>
<td>NAT</td>
<td>Network Address Translation</td>
</tr>
<tr>
<td>NAT44</td>
<td>Network Address Translation IPv4-to-IPv4</td>
</tr>
<tr>
<td>NED</td>
<td>Network Element Driver</td>
</tr>
<tr>
<td>NEDs</td>
<td>Network Element Drivers</td>
</tr>
<tr>
<td>NFCONF</td>
<td>Network Configuration protocol</td>
</tr>
<tr>
<td>NFV</td>
<td>Network Functions Virtualization</td>
</tr>
<tr>
<td>NH</td>
<td>Next Hope</td>
</tr>
<tr>
<td>NIC</td>
<td>Network Interface Card</td>
</tr>
<tr>
<td>NICs</td>
<td>Network Interface Cards</td>
</tr>
<tr>
<td>NSO</td>
<td>Network Service Orchestrator</td>
</tr>
<tr>
<td>ODL</td>
<td>Open Daylight</td>
</tr>
<tr>
<td>OS</td>
<td>Operating System</td>
</tr>
<tr>
<td>OSPF</td>
<td>Open Shortest Path First</td>
</tr>
<tr>
<td>OSS</td>
<td>Operations Support System</td>
</tr>
<tr>
<td>OVS</td>
<td>Open Virtual Switch</td>
</tr>
<tr>
<td>PE</td>
<td>Provider Edge</td>
</tr>
<tr>
<td>PnP</td>
<td>Plug-N-Play</td>
</tr>
<tr>
<td>QEMU</td>
<td>Quick Emulator</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>RA</td>
<td>Remote Access</td>
</tr>
</tbody>
</table>
### Table 1  Terminology

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>REST</td>
<td>Representation State Transfer</td>
</tr>
<tr>
<td>RFC</td>
<td>Request For Comments</td>
</tr>
<tr>
<td>RFS</td>
<td>Resource Facing Service</td>
</tr>
<tr>
<td>ROI</td>
<td>Return on Investment</td>
</tr>
<tr>
<td>SA</td>
<td>Source Address</td>
</tr>
<tr>
<td>SDN</td>
<td>Solution uses software-defined networking</td>
</tr>
<tr>
<td>SLA</td>
<td>Service Level Agreement</td>
</tr>
<tr>
<td>SMB</td>
<td>Small Medium Business</td>
</tr>
<tr>
<td>SNMP</td>
<td>Simple Network Management Protocol</td>
</tr>
<tr>
<td>SP</td>
<td>Service Provider</td>
</tr>
<tr>
<td>SRC</td>
<td>Source IP Address</td>
</tr>
<tr>
<td>SSH</td>
<td>Secure Shell</td>
</tr>
<tr>
<td>SSL</td>
<td>Secure Sockets layer</td>
</tr>
<tr>
<td>SUDI</td>
<td>Secure Unique Device Identifier</td>
</tr>
<tr>
<td>TCO</td>
<td>Total Cost of Ownership</td>
</tr>
<tr>
<td>TP</td>
<td>Termination Point</td>
</tr>
<tr>
<td>UCS</td>
<td>Unified Computing System</td>
</tr>
<tr>
<td>vFW</td>
<td>Virtual Firewall</td>
</tr>
<tr>
<td>VIM</td>
<td>Virtual Interface Manager</td>
</tr>
<tr>
<td>VirtIO</td>
<td>Virtual Input/Output</td>
</tr>
<tr>
<td>VLAN</td>
<td>Virtual Local Area Network</td>
</tr>
<tr>
<td>VM</td>
<td>Virtual Machine</td>
</tr>
<tr>
<td>VMs</td>
<td>Virtual Machines</td>
</tr>
<tr>
<td>VNF</td>
<td>Virtual Network Function</td>
</tr>
<tr>
<td>VNFs</td>
<td>Virtual Network Functions</td>
</tr>
<tr>
<td>VNIC</td>
<td>Virtual Network Interface</td>
</tr>
<tr>
<td>VPN</td>
<td>Virtual Private Network</td>
</tr>
<tr>
<td>VPP</td>
<td>Vector Packet Parsing</td>
</tr>
<tr>
<td>VR</td>
<td>Virtual Router</td>
</tr>
<tr>
<td>VRF</td>
<td>Virtual Route Forwarding</td>
</tr>
<tr>
<td>VRRP</td>
<td>Virtual Registry Registrar Protocol</td>
</tr>
<tr>
<td>WAN</td>
<td>Wide Area Network</td>
</tr>
<tr>
<td>WCCPv2</td>
<td>Web Cache Communications Protocol</td>
</tr>
<tr>
<td>WSA</td>
<td>Web Security Appliance</td>
</tr>
<tr>
<td>WSAv</td>
<td>Web Security Appliance virtualized</td>
</tr>
<tr>
<td>XML</td>
<td>eXtensible Marking Language</td>
</tr>
<tr>
<td>YANG</td>
<td>Yet Another Next Generation (data modeling)</td>
</tr>
<tr>
<td>ZTD</td>
<td>Zero Touch Deployment</td>
</tr>
</tbody>
</table>