

# MTAS Internal and External Connectivity

MTAS

DESCRIPTION

**Copyright**

© Ericsson AB 2016–2018. All rights reserved. No part of this document may be reproduced in any form without the written permission of the copyright owner.

**Disclaimer**

The contents of this document are subject to revision without notice due to continued progress in methodology, design and manufacturing. Ericsson shall have no liability for any error or damage of any kind resulting from the use of this document.

**Trademark List**

All trademarks mentioned herein are the property of their respective owners. These are shown in the document Trademark Information.



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	IMS Security Principles	1
1.2	Cloud Deployment Generic Principles	8
<b>2</b>	<b>MTAS Internal Communication</b>	<b>11</b>
2.1	TIPC	11
2.2	INET IPv4	12
2.3	INET IPv6 over IPVLAN	15
<b>3</b>	<b>MTAS External Communication</b>	<b>17</b>
3.1	Operation and Maintenance	17
3.2	Virtual IP Access	19
3.3	Appendix – Configuration Templates	28





# 1 Introduction

This document provides description about the internal and external connectivity of the MTAS product that is designed around the generic security principles and design rules of the Ericsson IMS System. The document also describes how these can be realized on cloud environments.

## 1.1 IMS Security Principles

This section describes the IMS security principles.

### 1.1.1 Network Architecture

Figure 1 shows the architecture of the IMS security principles in terms of modules. The IMS Central and Remote Modules are introduced to describe options for placing the IMS nodes in geographically dispersed sites in the network of an operator. A module is a grouping of nodes and infrastructure equipment that are placed in the same site.

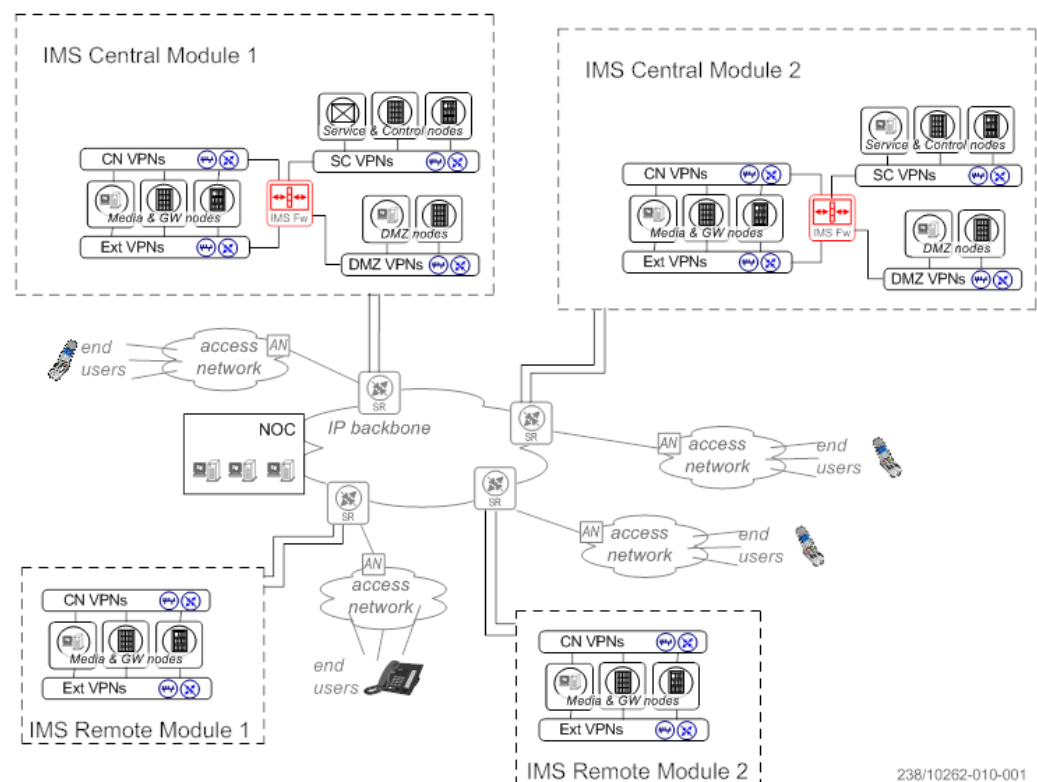


Figure 1 IMS Network Modules

**Principle 1 – MTAS is placed in an IMS Central Module**

## 1.1.2 Network Domain Security

Network Domain Security (NDS) provides access control to the network and in particular to the sites where the Network Elements (NE) and nodes reside. The whole network is partitioned into Security Domains (SD). The different NEs are placed in different SDs based on their functionality and level of trust. Different NDS security functions at the border of each Security Domain, in the backbone and also internally within each Security Domain are applied. Only legitimate traffic is allowed to pass from one Security Domain to another by enforcing control policies for all inter-Security Domains traffic.

Apart from the IP backbone, three security domains are defined:

- The Service and Control Security Domain (S&C SD)
- The Demilitarized Zone Security Domain (DMZ SD)
- The Media and Gateway Security Domain (M&G SD)

Figure 2 shows the IMS Security Domains.

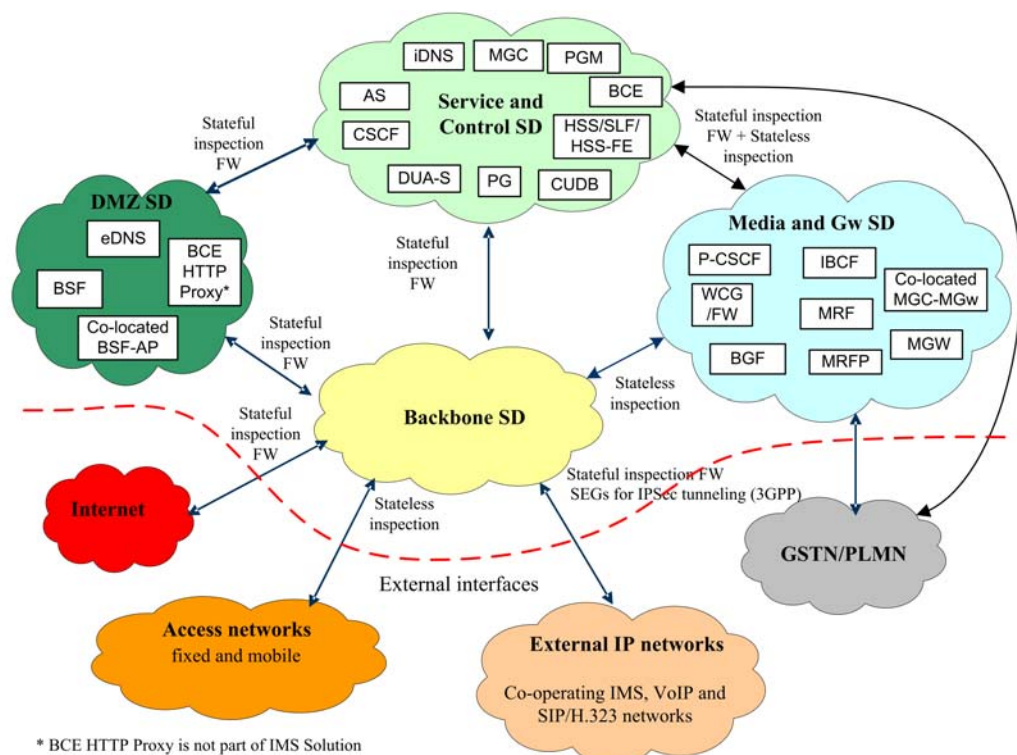


Figure 2 IMS Security Domains

**Principle 2 – MTAS belongs to the Service and Control IMS Security Domain**



### 1.1.3

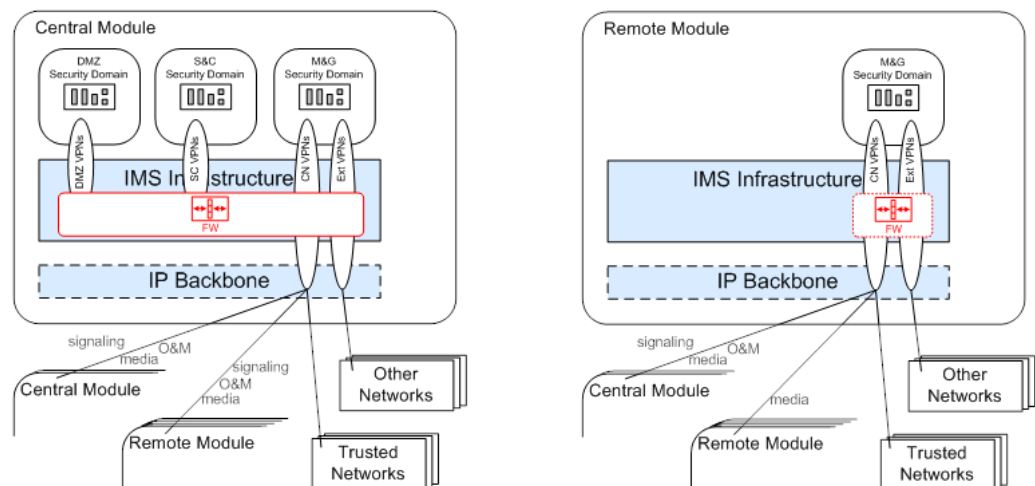
### Traffic Separation

The IMS traffic is divided into separate networks according to the traffic type and the level of security. Traffic in each category is transported over its own IP network (VPN) separated from traffic classified to other categories and IP networks.

The following is a list of VPN groups in the IMS network:

- SC VPNs
- DMZ VPNs
- CN VPNs
- Ext VPNs

Figure 3 shows the traffic separation and VPNs.



238/10262-010-003

Figure 3 Traffic Separation and VPNs

The DMZ VPNs and the SC VPNs are site-internal. That is, they are not extended over the backbone. The DMZ VPNs interconnect the nodes in the DMZ SD with the switches and the IMS firewall. The SC VPNs interconnect the nodes in the S&C SD with the switches and the IMS firewall.

The CN VPNs and the Ext VPNs, span the backbone and interconnect IMS modules in different sites. These VPNs interconnect the nodes in the M&G SD with the switches, IMS firewall, and the Site Routers. The CN VPNs traverse the backbone and interconnect nodes in different IMS Modules. The Ext VPNs interconnect the IMS modules with networks that are considered 'not trusted'.

**Principle 3** - MTAS is providing its services through the SC VPNs (OM\_SC, Sig\_SC and Bar\_SC)

### 1.1.4 MTAS External Interfaces

The MTAS included SCC, Conference, ST, and MMTel IMS Application Servers are providing various type of services over the logical interfaces, shown in Figure 4.

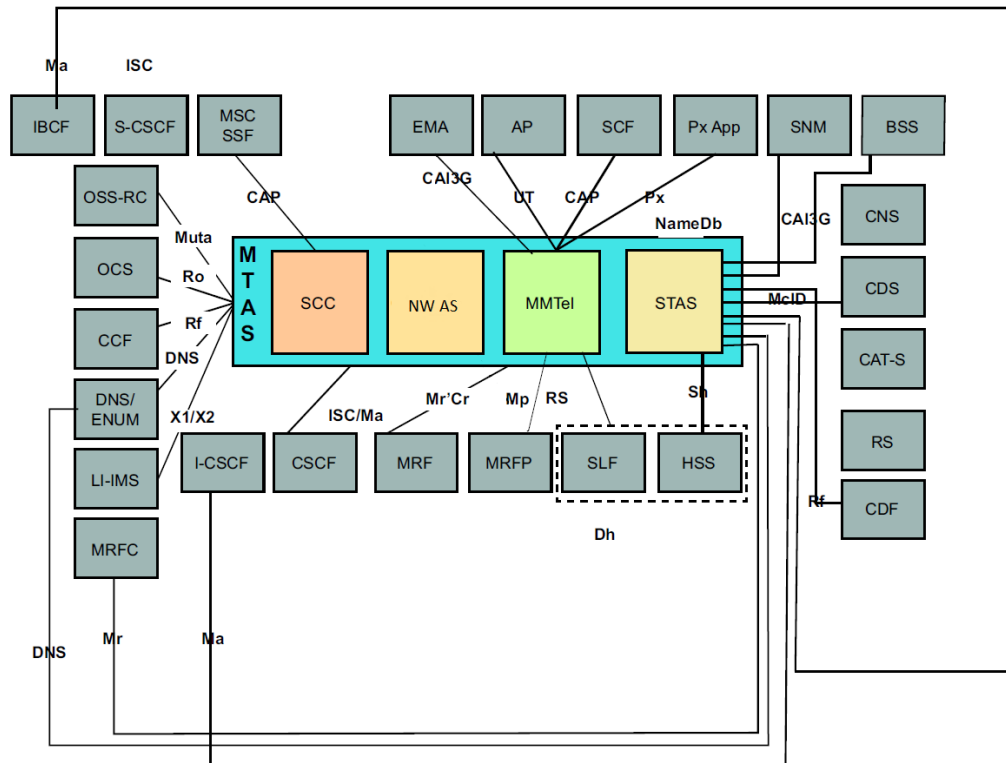


Figure 4 Interfaces and Protocols Supported by MTAS

The Service Access Points are single-homed IPv4/IPv6 UDP/TCP and single- or multihomed IPv4/IPv6 SCTP sockets that are bound to the OM\_SC, Sig\_SC, and Bar\_SC VPNs. Exposing the SAPs to the external systems within these VPNs is possible in two alternative ways (Ericsson vIMCN aligned, verified reference configurations / traffic separation profiles).





## Traffic Separation profile1

Table 1 Appearance of the SAPs in Traffic Separation profile1

VPN	(VIP Front End) Network	Distribut or Device (ALB)	IPsec	(V)IP	SAP
OM_SC	mtas_om_sp1	- (non-vip)	-	SC-1 System Management IP	<ul style="list-style-type: none"> <li>SC-1 maintenance</li> <li>NTP synchronization, client side (used when the OM IP is not active on SC-1)</li> </ul>
				SC-2 System Management IP	<ul style="list-style-type: none"> <li>SC-2 maintenance</li> <li>NTP synchronization, client side (used when the OM IP is not active on SC-2)</li> </ul>
				OM IP ("Muta MIP")	<ul style="list-style-type: none"> <li>Muta interfaces (COM CLI, COM Netconf, COM SNMP, COM FileM SFTP)</li> <li>NeLS server connectivity, client side</li> </ul>
	mtas_om_sp2	mtas_om	"mtas_om" IPsec ESP Tunnel EndPoint IP (optional)	OM VIP	<ul style="list-style-type: none"> <li>Provisioning LDAP</li> <li>Ro, Rf, and CDS interfaces (Diameter over TCP or single-homed SCTP)</li> </ul>
				CAI3G VIP	<ul style="list-style-type: none"> <li>CAI3G interface</li> </ul>
				Foo VIP	<ul style="list-style-type: none"> <li>X1</li> </ul>
Sig_SC	mtas_sig_sp	mtas_sig	-	Ut VIP	<ul style="list-style-type: none"> <li>Ut interface</li> </ul>
				SIGTRAN VIP SH1	<ul style="list-style-type: none"> <li>SIGTRAN interfaces, ASP/IPSP 1 (M3UA over single-homed SCTP)</li> </ul>
				SIGTRAN VIP SH2	<ul style="list-style-type: none"> <li>SIGTRAN interfaces, ASP/IPSP 2 (M3UA over single-homed SCTP)</li> </ul>
				Traffic VIP	<ul style="list-style-type: none"> <li>All the other logical interfaces, including ... <ul style="list-style-type: none"> <li>- H.248 (over single-homed SCTP)</li> <li>- Sh/Dh (Diameter over TCP or single-homed SCTP)</li> </ul> </li> </ul>
Bar_SC	mtas_bar_sp	mtas_bar	"mtas_bar" IPsec ESP Tunnel EndPoint IP (optional)	Bar VIP	<ul style="list-style-type: none"> <li>X2</li> </ul>

## Traffic Separation profile2

The following design objectives have been considered upon the development of this reference configuration:



- Diameter to be used over symmetric SCTP associations, in a diverse path fashion
- The limitation of some data center routers to be reflected, that is, policy based routing is not always possible. In turn, path diversity between the multihomed SCTP EndPoints to be achieved by bisecting the IP paths VNF internally.

Table 2 Appearance of the SAPs in Traffic Separation profile2

VPN	(VIP Front End) Network	Distribut or Device (ALB)	IPsec	(V)IP	SAP
OM_SC	mtas_om_sp1	- (non-vip)	-	SC-1 System Management IP	<ul style="list-style-type: none"> <li>SC-1 maintenance</li> <li>NTP synchronization, client side (used when the OM IP is not active on SC-1)</li> </ul>
				SC-2 System Management IP	<ul style="list-style-type: none"> <li>SC-2 maintenance</li> <li>NTP synchronization, client side (used when the OM IP is not active on SC-2)</li> </ul>
				OM IP ("Muta MIP")	<ul style="list-style-type: none"> <li>Muta interfaces (COM CLI, COM Netconf, COM SNMP, COM FileM SFTP)</li> <li>NeLS server connectivity, client side</li> </ul>
	mtas_om_sp2	mtas_om	"mtas_om" IPsec ESP Tunnel EndPoint IP (optional)	OM VIP	<ul style="list-style-type: none"> <li>Provisioning LDAP</li> </ul>
				CAI3G VIP	<ul style="list-style-type: none"> <li>CAI3G interface</li> </ul>
				Foo VIP	<ul style="list-style-type: none"> <li>X1</li> </ul>
	mtas_om_pdl	mtas_om_pdl	-	Ro/Rf/CDS VIP MH1	<ul style="list-style-type: none"> <li>Ro, Rf and CDS interfaces, mh1 (Diameter over multihomed SCTP)</li> </ul>
	mtas_om_pdr	mtas_om_pdr	-	Ro/Rf/CDS VIP MH2	<ul style="list-style-type: none"> <li>Ro, Rf and CDS interfaces, mh2 (Diameter over multihomed SCTP)</li> </ul>



VPN	(VIP Front End) Network	Distribut or Device (ALB)	IPsec	(V)IP	SAP
Sig-SC	mtas_sig_sp	mtas_sig	-	Ut VIP	<ul style="list-style-type: none"> <li>Ut interface</li> </ul>
				Traffic VIP	<ul style="list-style-type: none"> <li>All the other logical interfaces, including ...               <ul style="list-style-type: none"> <li>- H.248 (over single-homed SCTP)</li> </ul> </li> </ul>
	mtas_sig_pdl	mtas_sig_pdl	-	Traffic VIP MH1	<ul style="list-style-type: none"> <li>Sh/Dh, mh1 (Diameter over multihomed SCTP)</li> </ul>
				SIGTRAN VIP MH1	<ul style="list-style-type: none"> <li>SIGTRAN interfaces, mh1 (M3UA over multihomed SCTP) (ASP 1: port X; ASP 2: port Y)</li> </ul>
	mtas_sig_pdr	mtas_sig_pdr	-	Traffic VIP MH2	<ul style="list-style-type: none"> <li>Sh/Dh, mh2 (Diameter over multihomed SCTP)</li> </ul>
				SIGTRAN VIP MH2	<ul style="list-style-type: none"> <li>SIGTRAN interfaces, mh2 (M3UA over multihomed SCTP) (ASP 1: port X; ASP 2: port Y)</li> </ul>
Bar-SC	mtas_bar_sp	mtas_bar	"mtas_bar" IPsec ESP Tunnel EndPoint IP (optional)	Bar VIP	<ul style="list-style-type: none"> <li>X2</li> </ul>

The handshaked customization of either of the verified reference configurations can be done by considering the following constraints:

- SCTP multi-homing for H.248 is not supported
- The Ro-, Rf-, and CDS external logical interfaces can be bound to the same TCP (profile1) or SCTP (profile1 or profile2) sockets only
- The Sh- and Dh external logical interfaces can be bound to the same TCP (profile1) or SCTP (profile1 or profile2) sockets only
- The Sh/Dh and Ro/Rf/CDS external logical interfaces cannot be bound to the same set of TCP/SCTP sockets
- Maximum 9 access vNICs can be used for the external connectivity. Among these 9, 1 is reserved for non-VIP external, while the remaining 8 can be used for VIP external connectivity.
- VLAN trunking over the vNICs is not supported.
- The mtas\_om and mtas\_sig are mandatory ALBs and they cannot be left without a defined VIP address.



---

---

## Attention!

Risk of system malfunction or traffic disturbance.

---

---

Any customization that does not follow the constraints can result system malfunction and traffic disturbance.

## 1.2 Cloud Deployment Generic Principles

This section describes the generic principles related to cloud deployment.

### 1.2.1 Design Objectives

- The number of vNICs per VM must be less or equal to 10 – Fulfilled
- Does not use guest VLAN tagging – Fulfilled
- Does not use bonding on guest level – Fulfilled
- Guest OS to use the para-virtualization driver – Fulfilled
- Uses OSPFv2 (and v3) as a routing protocol – Fulfilled
- Uses only a single application VLAN for internal connectivity – Fulfilled
- “One VM-type” per VNF – Fulfilled  
(in the context of cloud management use cases; only one VM flavor needs to be defined with the same number of vNICs; profile1: 5x vNICs, profile2: 9x vNICs)

### 1.2.2 MTAS Virtual Network Function

The MTAS application system as Virtual Network Function (VNF) is built up from Virtual Machines in a 2+N-Way fashion. The supported minimum configuration is 2+2, meaning that 2 × SCs and 2 × PLs must exist in the system. The maximum configuration is 2+34 (2x SCs and 34x PLs). The VMs must be created on separate failure domains, on different Compute Nodes.

Figure 5 shows the MTAS VNF.

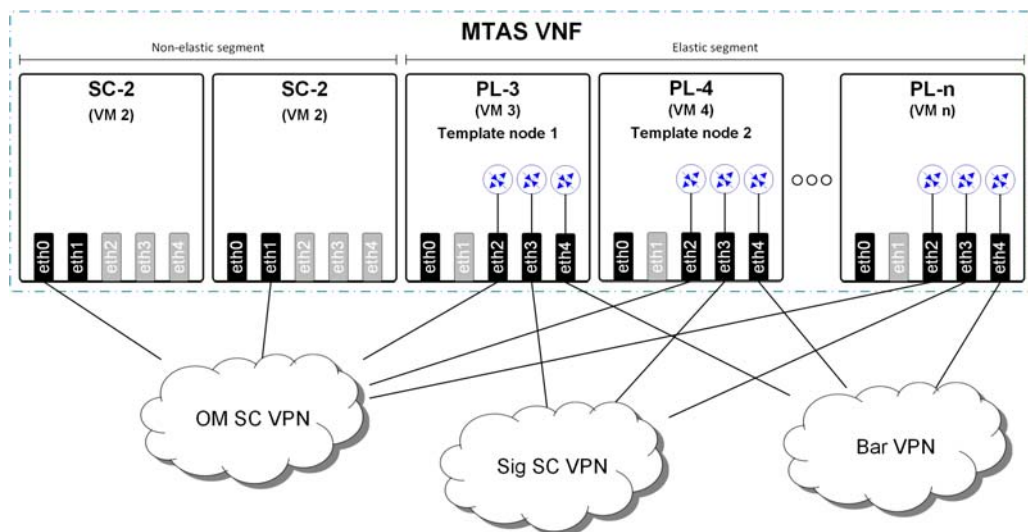


Figure 5 MTAS Virtual Network Function

The VMs are partitioned into a Non-elastic and an Elastic segment.

- The non-elastic part contains two Virtual Machines that are playing the role of redundant System Controllers (SC-1 and SC-2).
- The elastic part contains such Virtual Machines that are playing the role of Payload Nodes (PL-3 to PL-n). There are two VMs in this segment that are defined as redundant “Template nodes” (PL-3 and PL-4).
  - MTAS uses a concept of template nodes during the early discovery of VMs upon scale out use case to assign names appropriately to network interfaces. It also uses the template node as a source to clone the software used on the scaled node.
  - Although they are residing in the elastic segment, the template nodes cannot be scaled in.

The cloud infrastructure-provided virtual network ports (→ vNICs) are used by the VMs as follows:

- Used in traffic separation profile1 and profile2
  - eth0 is used on all the VMs for MTAS VNF internal communication
  - eth1 is used on the SCs for System Management and Operation and Maintenance external connectivity inside the OM\_SC VPN, single-homed
  - eth2 is used on the PLs for VIP external connectivity inside the OM\_SC VPN, single-homed
  - eth3 is used on the PLs for VIP external connectivity inside the Sig\_SC VPN, single-homed
  - eth4 is used on the PLs for VIP external connectivity inside the Bar VPN, single-homed
- Used in traffic separation profile2 only



- eth5 is used on the PLs for VIP external connectivity inside the OM\_SC VPN, attaching the multihomed PDL (Path Diversity Left) network infrastructure
- eth6 is used on the PLs for VIP external connectivity inside the OM\_SC VPN, attaching the multihomed PDR (Path Diversity Right) network infrastructure
- eth7 is used on the PLs for VIP external connectivity inside the Sig\_SC VPN, attaching the multihomed PDL (Path Diversity Left) network infrastructure
- eth8 is used on the PLs for VIP external connectivity inside the Sig\_SC VPN, attaching the multihomed PDR (Path Diversity Right) network infrastructure

The details about vNICs use are provided in the next sections. The Maximum Transmission Unit (MTU) size on the vNICs is configurable, and default value is 1500 bytes. The highest verified MTU value is 2140.



## 2 MTAS Internal Communication

This section provides a description about how the MTAS VNF internal communication is accomplished, no matter of what product is providing the underlying cloud infrastructure.

All the VMs within the MTAS VNF are connected to the “<MTAS VNF>\_internal” virtual network through the eth0 interface.

Figure 6 shows the “<MTAS VNF>\_internal” virtual network.

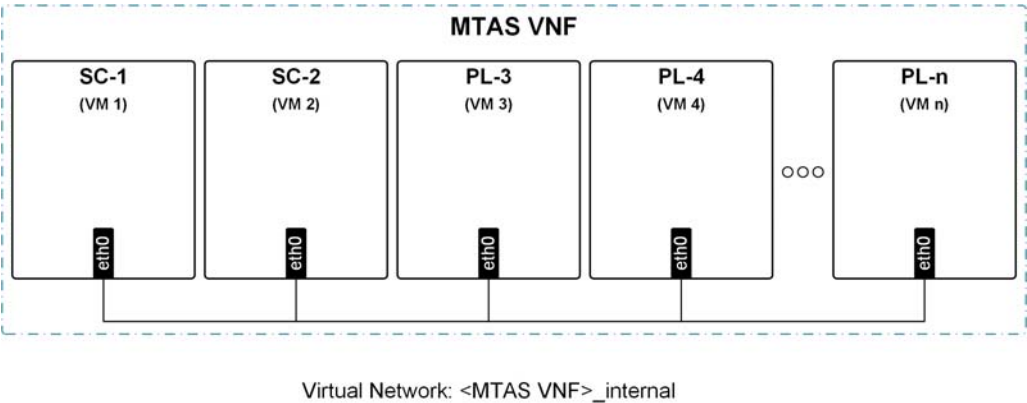


Figure 6 “<MTAS VNF>\_internal” Virtual Network

### 2.1 TIPC

TIPC is accommodated for Inter-Process Communication by a set of software components.

Figure 7 shows the “<MTAS VNF>\_internal” virtual network – TIPC.

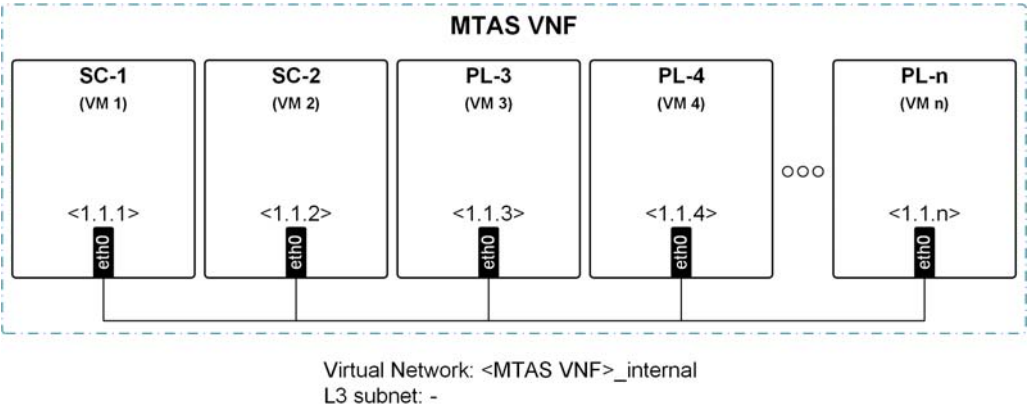


Figure 7 “<MTAS VNF>\_internal” Virtual Network – TIPC

A rather simple TIPC logical network topology is established within the MTAS VNF automatically. The VMs, as unique TIPC nodes, are organized into Cluster = 1, Zone = 1.



## Attention!

Risk of system malfunction or traffic disturbance.

- The virtual network provider must secure that this non-INET/INET6 based, pure L2 communication is enabled on the “<MTAS VNF>\_internal” virtual network.
- TIPC (ether proto 0x88ca) multicast is accomplished by using Ethernet broadcast (destination address: ff:ff:ff:ff:ff:ff) that must be enabled on the “<MTAS VNF>\_internal” virtual network.

## 2.2

### INET IPv4

The IPv4 based communication is another intra-cluster communication pattern that the system is heavily dependent on.

Figure 8 shows the “<MTAS VNF>\_internal” virtual network INET IPv4.

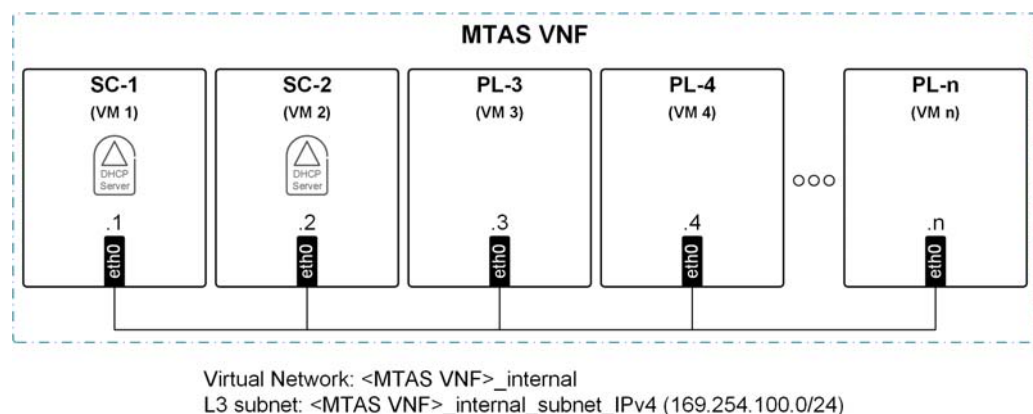


Figure 8 “<MTAS VNF>\_internal” Virtual Network – INET IPv4

The system is using the 169.254.100.0/24 subnet over the “<MTAS VNF>\_internal” virtual network infrastructure. Address allocation is an autonomous MTAS VNF internal service that is not tight to the cloud infrastructure provided address allocation mechanism by any means. On the “<MTAS VNF>\_internal” network, the DHCP service is given by the System Controllers to each other and for all PLs in a redundant fashion.





## Attention!

Risk of system malfunction or traffic disturbance.

- The virtual network provider must secure that the DHCP communication (BOOTP; UDP src/dst port: 67/68) is not filtered on the “<MTAS VNF>\_internal” virtual network.
- The virtual network provider must not connect cloud infrastructure DHCP service to the “<MTAS VNF>\_internal” virtual network.

### 2.2.1 INET IPv4 MIP Aliases for TFTP Purpose

As a completion of the MTAS VNF internal PXE bootstrap mechanism, the System Controllers are also presenting TFTP service to each other and for all PLs in a two-way active load shared manner. The service is bound to Movable IP (MIP) addresses on the “<MTAS VNF>\_internal” network.

Figure 9 shows the “<MTAS VNF>\_internal” virtual network -- TFTP over INET IPv4.

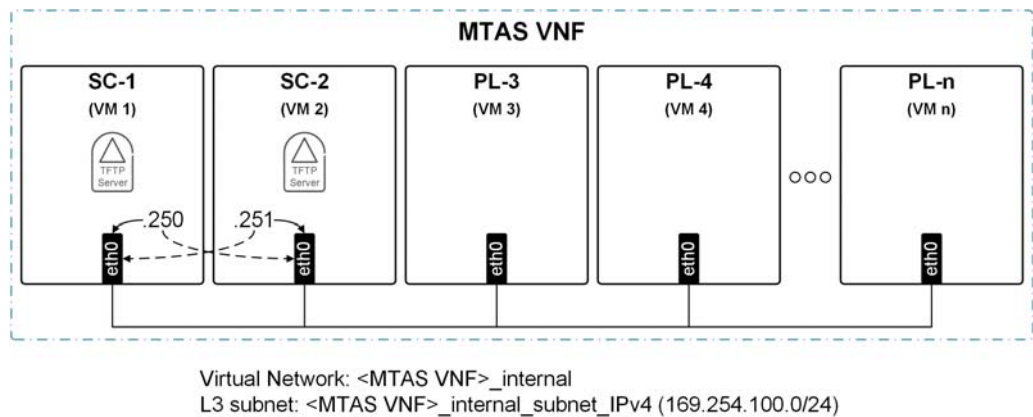


Figure 9 “<MTAS VNF>\_internal” Virtual Network – TFTP over INET IPv4

Load sharing is achieved through allocating a MIP address to each System Controller. If there is an SC failure, the system automatically assigns both MIP addresses to the remaining SC.

### 2.2.2 INET IPv4 MIP Alias for NFS Purpose

System Controllers are providing NFS service to each other and for all PLs in a 1+1 redundant manner. The service is bound to a Movable IP (MIP) address on the “<MTAS VNF>\_internal” network.



Figure 10 shows the “<MTAS VNF>\_internal” virtual network – NFS over INET IPv4.

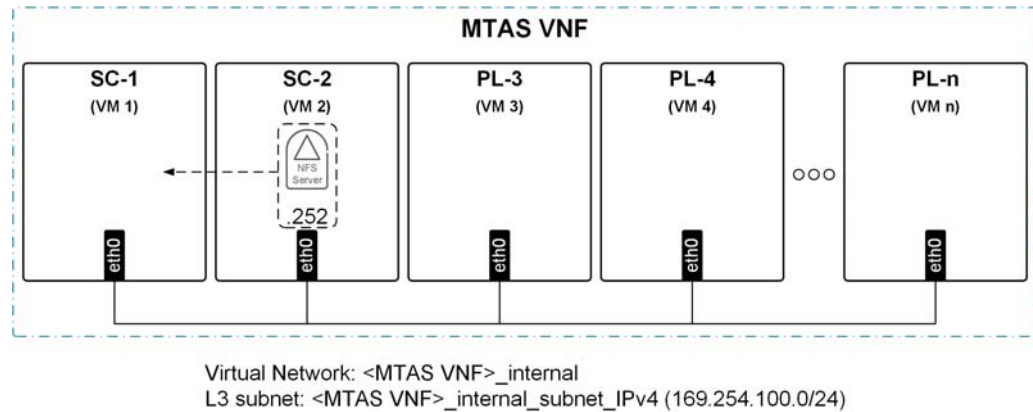


Figure 10 “<MTAS VNF>\_internal” Virtual Network – NFS over INET IPv4

If there is an SC failure, the system automatically reassigns the NFS MIP address and starts the NFS service on the redundant SC pair.

### 2.2.3

#### INET IPv4 MIP Alias for Common Parts IPC Purpose

Common Parts IPC is an INET IPv4-based communication method within the MTAS VNF. A piece of internal component, called Common Parts Manager (CP-M) is providing services for the clients (Common Parts Client; CP-C). The Service Access Point of CP-M is bound to a well-know IPv4 address on the “<MTAS VNF>\_internal” network. The CP-M is available on either of the available PLs.

Figure 11 shows the “<MTAS VNF>\_internal” virtual network – CP-M.

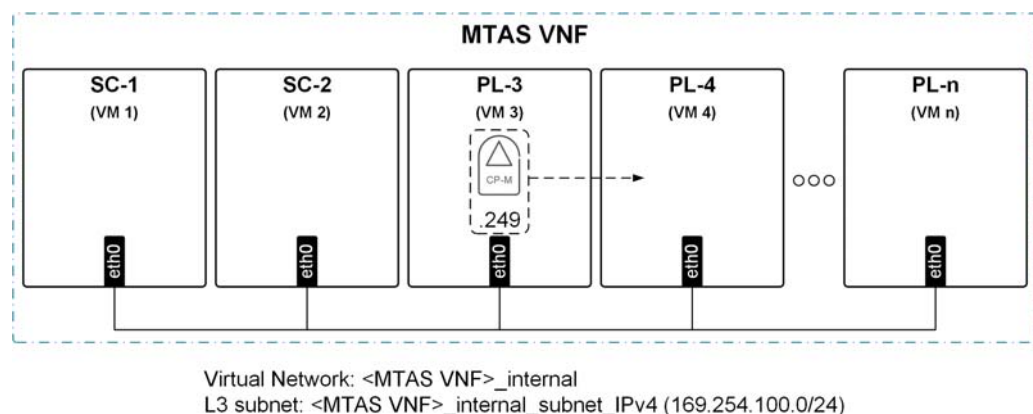


Figure 11 “<MTAS VNF>\_internal” Virtual Network – CP-M

When the PL that is hosting the CP-M goes down for any reason (failure, scale in), the system automatically reassigns the CP-M MIP address and starts the CP-M on another PL.



## 2.3 INET IPv6 over IPVLAN

Specific SW components that are running on the VMs of the MTAS VNF use Linux namespaces to separate their functionality from the rest.

Figure 12 shows the “<MTAS VNF>\_internal” virtual network – INET IPv6 over IPVLAN.

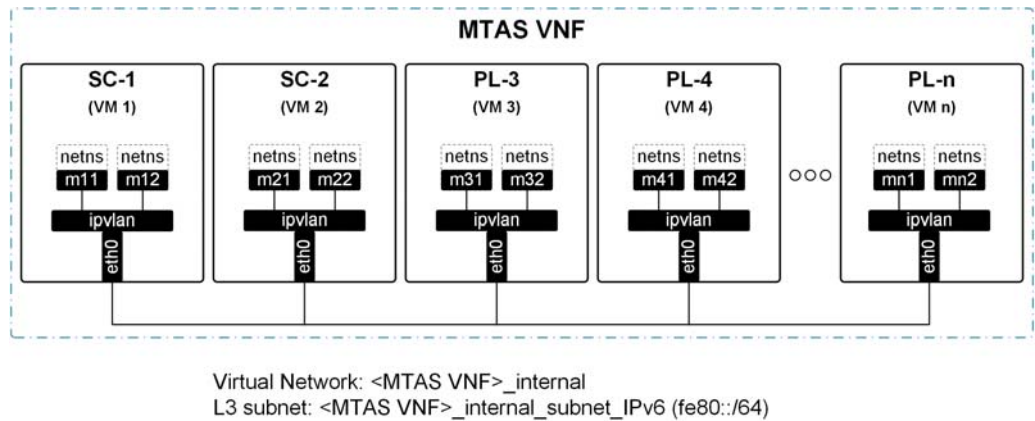


Figure 12 “<MTAS VNF>\_internal” Virtual Network – INET IPv6 over MAC VLANs

Inter-Process Communication between these entities is facilitated over IPVLAN. The IPVLAN slave interfaces are created with eth0 as master device. On L3 level, the communication is performed on the IPv6 link-local network. The allocation of these IPVLAN slave communication endpoints (m11 ... mn2) is dynamic within the cluster, meaning that an assigned IPv6 link-local address can pop up behind different MAC addresses.





## 3 MTAS External Communication

This section provides a description about how MTAS VNF external communication can be accomplished. In such a scenario, when the external connectivity is established through soft routers, the deployment of the Software router VMs must take place on dedicated compute nodes, not to cause background load for any of the MTAS VNF VMs.

### 3.1 Operation and Maintenance

#### 3.1.1 Overview

The purpose of having this infrastructure in place is as follows:

- Beside the virtual console connections provided by the cloud infrastructure, IP-based direct remote accessibility (over SSH) is also supplied to the Linux shell of the System Controllers; intended to be used in emergency scenarios only.
- Grant the availability of remote NTP time references for the System Controllers.  
(Payload Nodes are retrieving NTP time reference from either of the System Controllers.)
- Secure the availability of remote NeLS server for License Management purpose.
- The operation and maintenance services those are provided over the COM CLI/Netconf/SNMP interfaces are made available through this infrastructure. The single-point of COM NBI connectivity is secured by providing MIPs for both IPv4 and IPv6 that are owned by either of the SCs. The COM NBI services are always reachable through these MIPs.

Figure 13 shows the MTAS VNF System Management Infrastructure.

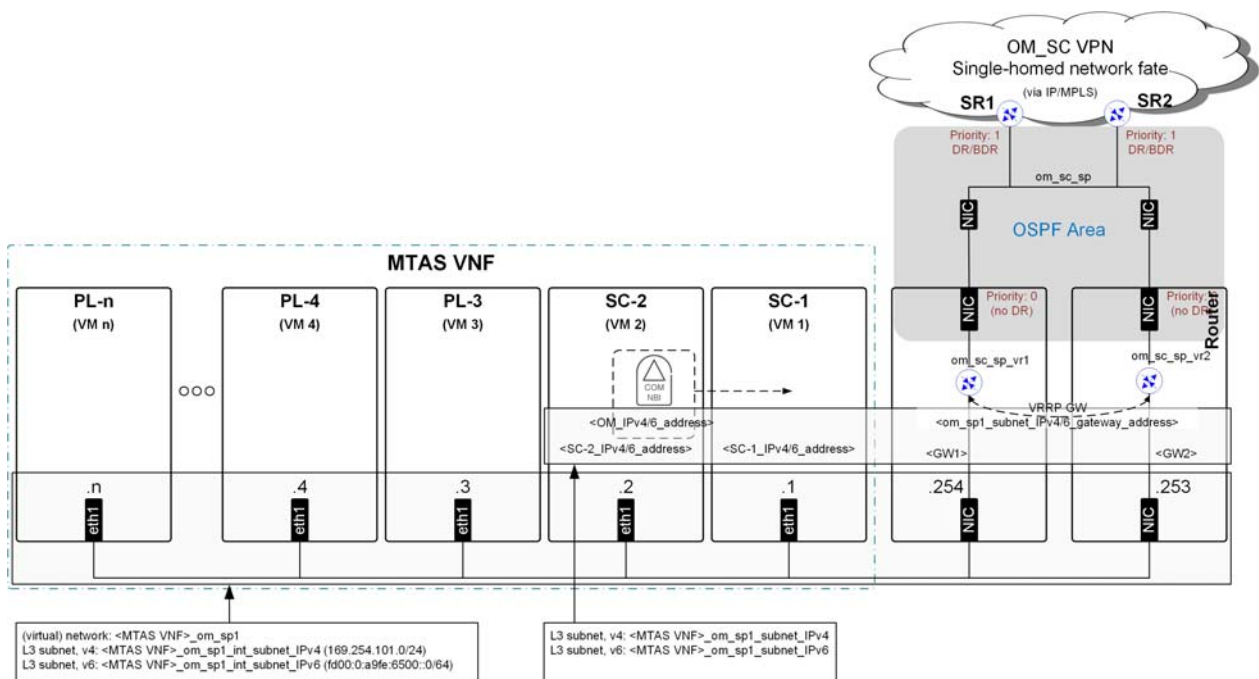


Figure 13 MTAS VNF System Management Infrastructure

### 3.1.2 Routing Design

All the VMs within the MTAS VNF are connected to the “<MTAS VNF>\_om\_sp1” network infrastructure through their eth1 interfaces. This network infrastructure is shared with the router instances that can either be located in- or outside of the cloud infrastructure. A fixed primary IP address is assigned to all the connected virtual network ports/vNICs from the “<MTAS VNF>\_om\_sp1\_int\_subnet\_IPv4/6” link-local subnet.

Additional IPv4/6 addresses are assigned to the connected eth1 of the SC (SC-1/2\_IPv4/6\_address) and the router (GW1, GW2) interfaces on the created broadcast domain. These addresses are taken from the “<MTAS VNF>\_om\_sp1\_subnet\_IPv4/6” routable subnet whereby transit is enabled by a VRRP protected gateway (“om\_sp1\_subnet\_IPv4/6\_gateway\_address”). By setting the default route against this VRRP IP on the SCs, the system management connectivity of them is established at once.

Beside the SC and GW addresses, there is another address reserved and shared by the System Controllers, the “OM\_IPv4/6\_address” (also referred as “Muta MIP”) that is active on SC where the VNF is actually providing its operation and maintenance services over the COM CLI/Netconf/SNMP interfaces.

The single point of operation and maintenance access is guaranteed for the COM NBI-related in- and outbound streams. On the SC where the “OM\_IPv4/6\_address” is active, this movable IP is set as the preferred source address in the default route setting for all the outbound streams (UDP: SNMP Trap; TCP: SFTP). This route preference is only used on the SC where the MIP is active.



## Attention!

Risk of system malfunction or traffic disturbance.

The “<MTAS VNF>\_om\_sp1\_subnet\_IPv4/6” subnet is such an overlay what the virtual networking provider of the cloud environment is not aware of. Therefore, the virtual networking service must guarantee that the traffic on the subnet other than “<MTAS VNF>\_om\_sp1\_int\_subnet\_IPv4/6” is also passing through the connected virtual ports.

As long as the virtual networking firewall service is globally inactivated for other reasons, it is not an issue at all. However, if it is turned on, firewall exceptional rules must be defined for every connected port on the “<MTAS VNF>\_om\_sp1” virtual network infrastructure (in the OpenStack case, use “allowed\_address\_pairs” setting on Neutron ports; the “<MTAS VNF>\_om\_sp1\_subnet\_IPv4/6” subnet to be defined in a CIDR notation).

**Note:** The infrastructure for operation and maintenance is designed by taking an OpenStack specific issue into consideration: Neutron does not allow creating more ports on a virtual network infrastructure than the available number of IP hosts on the connected IP subnet. However, for the benefit of unified product follow-up activities (cloud infrastructure agnostic unified network design), the approach must be adopted in VMware environment too.

The “<MTAS VNF>\_om\_sp1\_subnet\_IPv4/6” to be advertised through OSPF in the same pair of routing process instances that are used to handle VIP external connectivity in the OM\_SC VPN.

## 3.2 Virtual IP Access

### 3.2.1 Overview

Shared IP addresses are used to address distributed application functions in a multi-processor cluster. The shared IP addresses are called VIP (Virtual IP) addresses that can be grouped into a logical container, called Abstract Load Balancers (ALB). The ALBs are represented by their associated network (distributor) device on every SC/PL instance. Service instances that are running on different VMs can all make their provided functions externally available by binding them to UDP/TCP/SCTP sockets (Service Access Points). In these sockets, any of the VIP addresses provided by any of the distributor device can be used. Table 1 and Table 2 are summarizing the two alternative ways regarding what VIP addresses provided by which distributor device are used to bind the distributed application functions.

The connectivity between the set of ALB provided VIP addresses and the external network entities is established through the VNF embedded redundant routing instances, referred to as FrontEnds. For an ALB that is grouping single-homed VIP addresses, 4x FrontEnds are defined. The ALBs that are grouping multihomed VIP addresses are equipped with 2x FrontEnds. Because of the nature of the VNF internal life cycle management of the FrontEnds, their position is not fixed, they can pop up on any of the available PLs. One PL can host one FrontEnd from an ALB, but can host multiple FrontEnds from different ALBs. To maximize the availability of the established SCTP association, the FrontEnds of the paired multihomed ALBs (<mtas\_om|mtas\_sig>\_pdl and <mtas\_om|mtas\_sig>\_pdr) are never co-located on a single PL.

All the FrontEnds that are belonging to an ALB, together with the adjacent VIP GW routers, are connected to the same VIP FrontEnd network. Routing over the VIP FrontEnd networks can be accommodated in three different ways. These are (in preference order):

- Static routing (without single-hop BFD supervision)
- Static routing with single-hop BFD supervision
- Dynamic routing, with or without single-hop BFD supervision

Mixed routing fashion is not supported in the IPv4/IPv6 dual-stack scenario. That is, either of the alternatives can be used for both IPv4 and IPv6. The following chapters are providing an overview about these routing options.

## 3.2.2 Static Routing (without Single-Hop BFD Supervision)

The static routing reference design is considering the constraint, that is, a FrontEnd can be configured statically with a single piece of default gateway, per INET flavor.

### 3.2.2.1 Static Routing (without Single-Hop BFD Supervision) for Single-Homed VIP Addresses

The following figure shows the MTAS VNF Virtual IP Access, static routing without BFD routing design for single-homed VIP addresses.





The static routes towards the VIP addresses are imported into the OSPF routing domain by the `<om|sig|bar>_sc_sp_vr1` and `<om|sig|bar>_sc_sp_vr2` OSPF processes.

### 3.2.2.2 Static Routing (without Single-Hop BFD Supervision) for Multihomed VIP Addresses

The following figure shows the MTAS VNF Virtual IP Access, static routing without BFD design for multihomed VIP addresses.

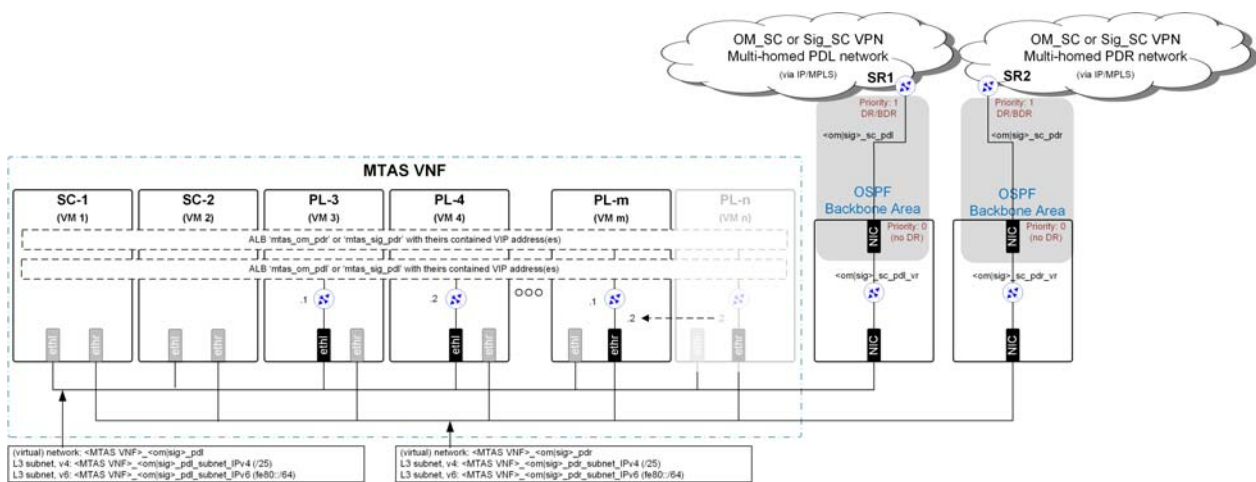


Figure 15 MTAS VNF Virtual IP Access – Static Routing Design, Multihomed VIP Addresses

The fundamental behavior is the same as in the network design for single-homed VIP addresses. However, there are some differences:

— Receive direction

- All the FrontEnds are statically configured as possible equal cost next hops towards the mtas\_<om|sig>\_pd1 ALB provided VIPs in the <om|sig>\_sc\_pd1 VIP GW router.

Similarly, all the FrontEnds are statically configured as possible equal cost next hops towards the `mtas_om_sig_pdr` ALB provided VIPs in the `om_sig_sc_pdr` VIP GW router.

- If a PL that is hosting a FrontEnd is scaled in and there is no free PL exist in the VNF that could host the configured but not instantiated FrontEnd, its external interface address is taken over by a still running FrontEnd element.

In the figure, consider the case when there are 4 PLs available, and PL-n is scaled in. In this case, the external interface address of the hosted FrontEnd is automatically taken over by a still running FrontEnd instance that is belonging to the same ALB; in the example, by the FrontEnd that is hosted on PL-m.



- With the proper configuration, the VNF guarantees that at least one FrontEnd remains for both the `mtas_<om|sig>_pdl` and the `mtas_<om|sig>_pdr` ALBs, as a last resort.
  - Because of this mechanism, the configured next-hop addresses of the VIP routes cannot disappear. Therefore, their availability does not need to be supervised with BFD; all the statically configured next-hop addresses towards a VIP can permanently remain in the ECMP group of the VIP GW.
- Sending direction
- The single piece of default gateway address (per INET flavor) that can be configured in the FrontEnds is provided by the adjacent VIP GW router.
    - For the FrontEnds in the `mtas_<om|sig>_pdl` ALB, the default route is provided through the `<om|sig>_sc_pdl` VIP GW router
    - For the FrontEnds in the `mtas_<om|sig>_pdr` ALB, the default route is provided via the `<om|sig>_sc_pdr` VIP GW router

The static routes towards the VIP addresses are imported into the separate OSPF routing domains by the `<om|sig>_sc_pdl` and `<om|sig>_sc_pdr` OSPF processes.

### SCTP Multi-Homing, Double-Failure Scenario

A double-failure scenario needs to be considered when there is only one PL remaining in the system. See the use case depicted in the following figure:

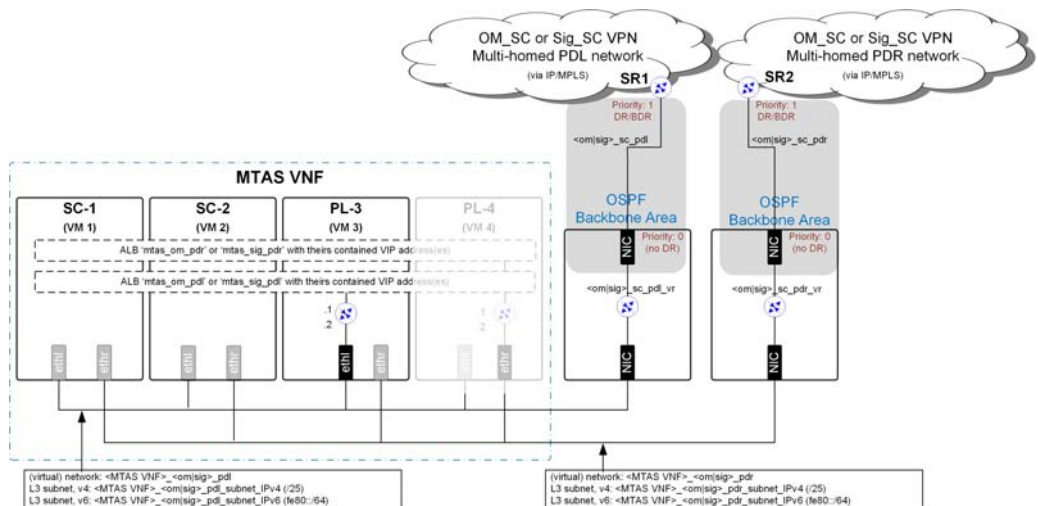


Figure 16 Double-failure Scenario Use Case

Let us assume that the VNF contains two PLs only, and these are hosting two FrontEnds that are belonging to the `mtas_<om|sig>_pdl` and the `mtas_<om|sig>_pdr` ALBs. The external interface addresses of the not instantiated FrontEnd entities are stacked, because of the limited number of available PLs. Assume that PL-4 becomes unavailable. In that situation, the VIP external connectivity over the PDR network is broken. The higher-level SCTP

protocol embedded mechanism detects the path failure, resulting in an active path reselection over the still established SCTP associations.

### 3.2.3 Static Routing with Single-Hop BFD Supervision

The static routing reference design is considering the constraint, that is, a FrontEnd can be configured statically with a single piece of default gateway, per INET flavor.

The VIP GW routers are not always BFD enabled, and even if so, the number of active BFD sessions can be limited. In turn, use static routing without BFD supervision whenever it is possible.

#### 3.2.3.1 Static Routing with Single-Hop BFD Supervision for Single-Homed VIP Addresses

The following figure shows the MTAS VNF Virtual IP Access, static routing design for single-homed VIP addresses with BFD enabled.

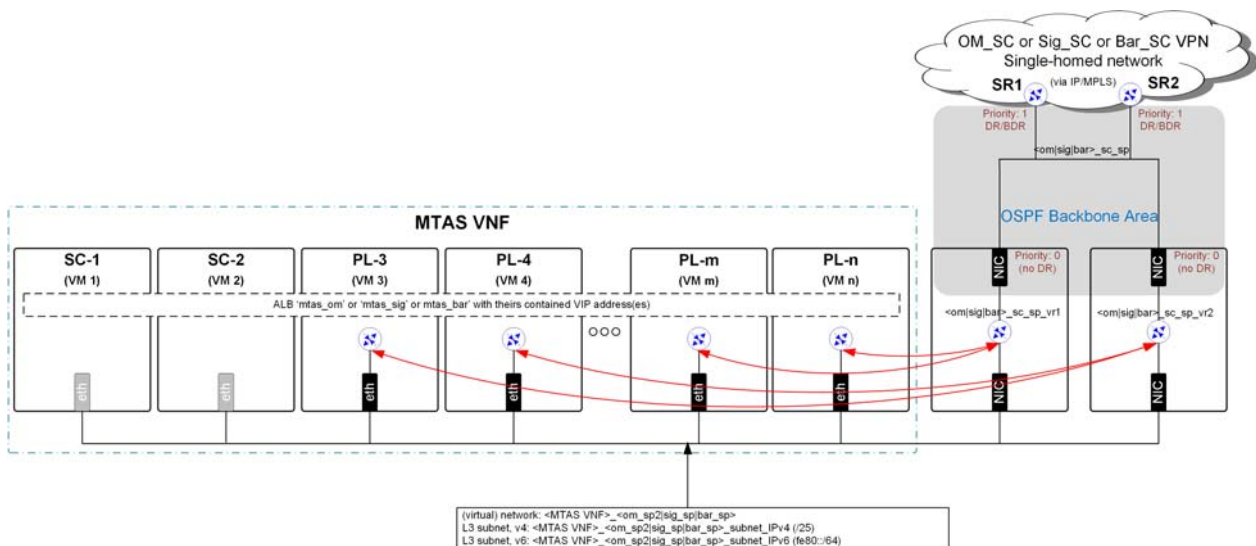


Figure 17 MTAS VNF Virtual IP Access – BFD-Enabled Static Routing Design, Single-Homed

The FrontEnds are split into two groups; two routing instances in each group:

#### — Group 1

- The <om|sig|bar>\_sc\_sp\_vr1 router is statically configured as default gateway in the FrontEnds in this group (PL-m and PL-n in the figure).
- The FrontEnds in this group are statically configured as possible equal cost next hops towards the mtas\_<om|sig|bar> ALB provided VIPs in the <om|sig|bar>\_sc\_sp\_vr1 router.
- Every statically configured route is protected with single-hop BFD session.

#### — Group 2



- The `<om|sig|bar>_sc_sp_vr2` router is statically configured as default gateway in the FrontEnds in this group (PL-3 and PL-4 in the figure).
- The FrontEnds in this group are statically configured as possible equal cost next hops towards the `mtas_<om|sig|bar>` ALB provided VIPs in the `<om|sig|bar>_sc_sp_vr2` router.
- Every statically configured route is protected with single-hop BFD session.

The static routes towards the VIP addresses are imported into the OSPF routing domain by the `<om|sig|bar>_sc_sp_vr1` and `<om|sig|bar>_sc_sp_vr2` OSPF processes.

### 3.2.3.2 Static Routing with Single-Hop BFD Supervision for Multihomed VIP Addresses

The following figure shows the MTAS VNF Virtual IP Access, static routing design for multihomed VIP addresses with BFD enabled.

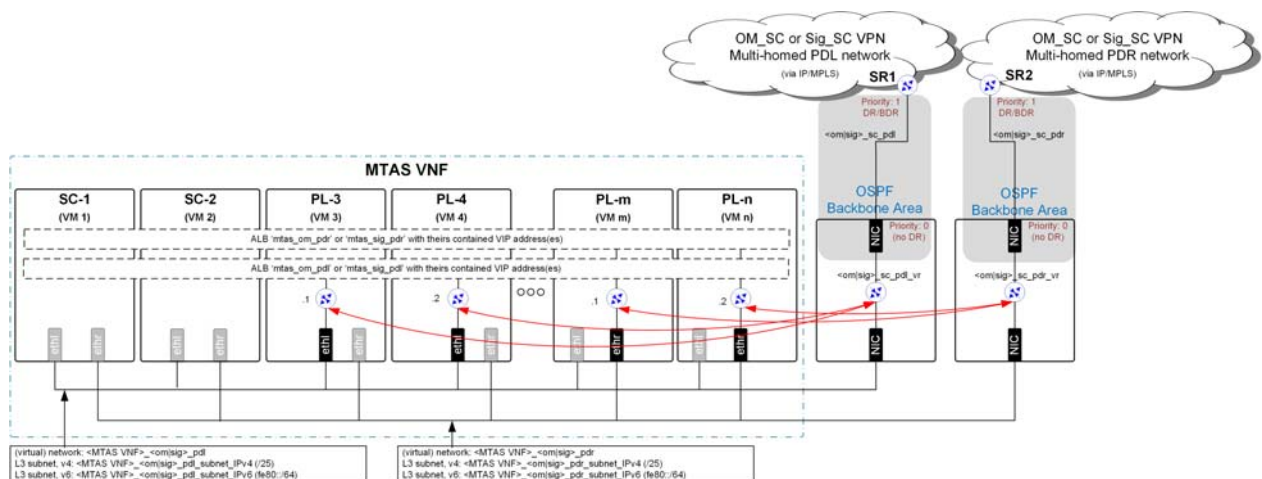


Figure 18 MTAS VNF Virtual IP Access – BFD-Enabled Static Routing Design, Multihomed

The connected PLs embedded routing instances, the FrontEnds are split into two groups; two routing instances in each group.

#### — Group 1, PDL

- The `<om|sig>_sc_pdl_vr` router is statically configured as default gateway in the FrontEnds of the `mtas_<om|sig>_pdl` ALB
- The FrontEnds of the `mtas_<om|sig>_pdl` ALB are statically configured as possible equal cost next hops towards the provided VIPs in the `<om|sig>_sc_pdl_vr` router
- Every statically configured route is protected with single-hop BFD session

#### — Group 2, PDR

- The `<om|sig>_sc_pdr_vr` router is statically configured as default gateway in the FrontEnds of the `mtas_<om|sig>_pdr` ALB
- The FrontEnds of the `mtas_<om|sig>_pdr` ALB are statically configured as possible equal cost next hops towards the provided VIPs in the `<om|sig>_sc_pdr_vr` router
- Every statically configured route is protected with single-hop BFD session

The static routes towards the VIP addresses are imported into separate OSPF routing domains by the `<om|sig>_sc_pdr_vr` and the `<om|sig>_sc_pdr_vr` OSPF processes.

### 3.2.4 Dynamic Routing Design

The dynamic routing reference design is considering the constraint, that is, virtual OSPF links cannot be configured in the FrontEnds. Hence, and because every OSPF area must be connected to the backbone, the VIP GW routers must be connected to the OSPF backbone.

Preferably, OSPF must be combined with Bidirectional Forwarding Detection (BFD) to achieve faster forwarding path failure detection.

#### 3.2.4.1 Dynamic Routing for Single-Homed VIP Addresses

Figure 19 shows the MTAS VNF Virtual IP Access, dynamic routing design for single-homed VIP addresses.

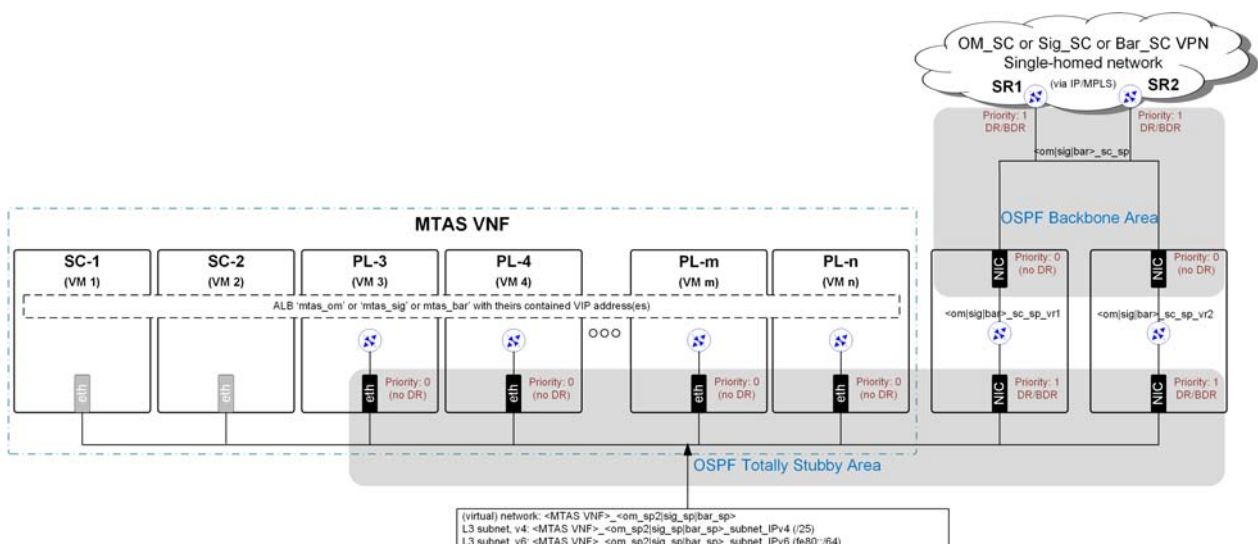


Figure 19 MTAS VNF Virtual IP Access – Dynamic Routing Design, Single-Homed

The `<om|sig|bar>_sc_sp_vr1` and the `<om|sig|bar>_sc_sp_vr2` routing instances are ABRs in two areas:



#### — OSPF Totally Stubby Area

- To hide the OSPF topological changes from the MTAS VNF routing instances as much as possible, hence to minimize the load on them, ensure the following:
  - The priority of the OSPF interfaces of the PLs embedded routing instances is set to 0. This way they are never elected as DR or BDR inside the area.
  - The OSPF area, between the PLs embedded and the soft router instances, is defined as a Totally Stubby Area (area <X.Y.Z.W> stub no-summary). In turn:
    - Only type 1 and type 2 LSAs are exchanged inside the area. The size of the link state database of the PLs embedded routing instances can be minimized.
    - All routing out of the area relies on the redundant default routes that injected by the ABRs.
- The VIP addresses that are collected in the `mtas_<sig|om|bar>` ALB, as connected /32 (IPv4) and /128 (IPv6) stubs, are injected into OSPF by the MTAS VNF routing instances with LSA Type 1 (Router LSA).

#### — Backbone

- To minimize the load on the soft routers upon OSPF topological changes in the backbone area, the priority of the backbone interfaces is set zero. This way they are never elected as DR or BDR inside the backbone area.
- All routing out of the IMS Central Module (Site) relies on the default routes that are injected into OSPF by the Site Routers (default-information originate).

### 3.2.4.2 Dynamic Routing for Multihomed VIP Addresses

Figure 20 shows the MTAS VNF Virtual IP Access, dynamic routing design for multihomed VIP addresses.



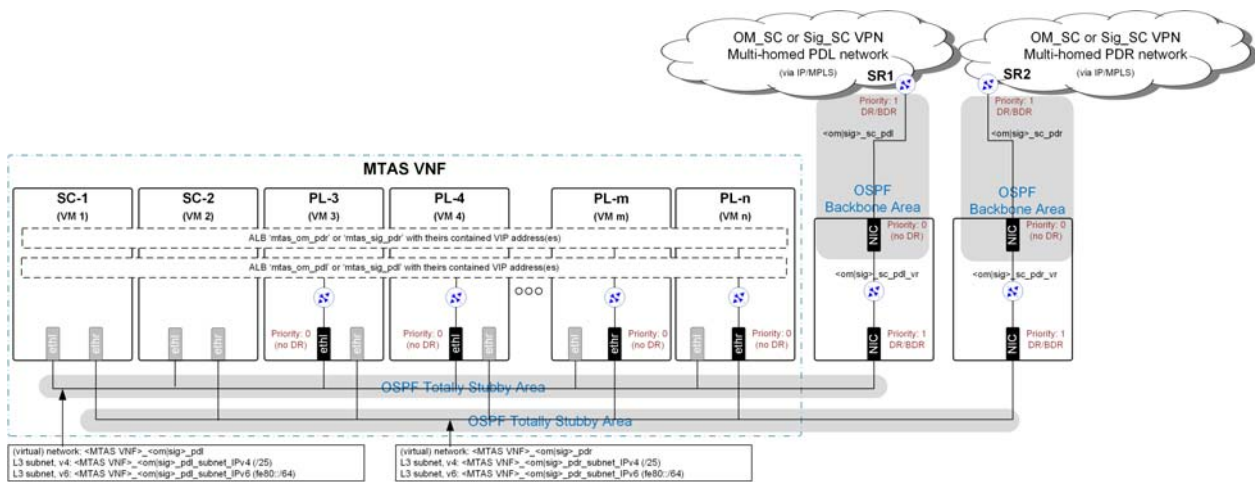


Figure 20 MTAS VNF Virtual IP Access – Dynamic Routing Design, Multihomed

The routing does not differ fundamentally from the single-homed scenario. Here, the only difference is that the VIP addresses that are collected in the `mtas_<om|sig>_pdl` and the `mtas_<om|sig>_pdr` ALBs, as connected /32 (IPv4) and /128 (IPv6) stubs, are injected into different routing domains. There is no network layer cross connect between these two routing domains.

### 3.2.5 MTU

The Maximum Transmission Unit (MTU) size within the ALBs is 1452 bytes default. If Path MTU Discovery (PMTUD) is not used, so that the size of a received IP packet can fall into the range of 1453–1500 bytes, the sender must ensure that the Do not Fragment (DF) bit is not set in the IPv4 header. In turn, the embedded routing instances can split such received IPv4 packet into smaller fragments.



### Attention!

Risk of system malfunction or traffic disturbance.

If the DF flag is set for an incoming IP packet, and the size of the IPv4 packet is in between 1453–1500 bytes, the packet is dropped in the FrontEnds.

## 3.3 Appendix – Configuration Templates

The VIP-independent system management and Operation and Maintenance infrastructure is established upon the orchestration of the MTAS VNF. For that, the site-specific details to be provided through the orchestration engine of the given cloud infrastructure. For further details, refer to MTAS SW Installation.





Once MTAS is deployed, the Virtual IP access specifics can be configured over the COM NBI interface.



## Attention!

Risk of system malfunction or traffic disturbance.

Any deviation from the recommended configuration can cause unwanted behavior of the system.

### 3.3.1 Inactive the Predefined ALBs

Two predefined ALBs can be found in the image of the MTAS VNF. These are: `mtas_om` and `mtas_sig`. To start with the configuration, inactive these ALBs:

```
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<mtas_om|mtas_sig>,inactivate
```

### 3.3.2 Create Floating EvipNodes

Floating EvipNodes cannot be predefined at design time, since their priority cannot be changed once they exist. However, their priority must be adjusted in-line with the chosen VIP routing strategy. Set the priorities as follows ({set of EvipNodes} / priority):

- Use these priority values only in that case if traffic separation profile1 is chosen (no VNF internal path bisection) and the “static without BFD” or the dynamic routing pattern is chosen on the FrontEnd networks.

```
{3,4,5,6} / 5.000000 ; {7,8,9,10,11,12,13,14,15} / 10.000000
```

- Use these values in every other case, that is, ALBs for VNF internal path bisection are defined or the “static with BFD” routing pattern is applied over any of the existing FrontEnd networks.

```
{3,5} / 5.000000 ; {4,6} / 6.000000 ; {7,8,9,10,11,12,13,14,15} / 10.000000
```

By substituting the priorities into the following template, issue the commands over the COM NBI.



```
ManagedElement=1,Transport=1,Evip=1,EvipDeclarations=1,EvipCluster=1
EvipNode=3
distribution="floating"
floatPriority="<priority>"
up
EvipNode=4
distribution="floating"
floatPriority="<priority>"
up
EvipNode=5
distribution="floating"
floatPriority="<priority>"
up
EvipNode=6
distribution="floating"
floatPriority="<priority>"
up
EvipNode=7
distribution="floating"
floatPriority="<priority>"
up
EvipNode=8
distribution="floating"
floatPriority="<priority>"
up
EvipNode=9
distribution="floating"
floatPriority="<priority>"
up
EvipNode=10
distribution="floating"
floatPriority="<priority>"
up
EvipNode=11
distribution="floating"
floatPriority="<priority>"
up
EvipNode=12
distribution="floating"
floatPriority="<priority>"
up
EvipNode=13
distribution="floating"
floatPriority="<priority>"
up
EvipNode=14
distribution="floating"
floatPriority="<priority>"
up
EvipNode=15
distribution="floating"
floatPriority="<priority>"
top
commit -s
```

**Note:** Modifying the `floatPriority` attribute of the predefined `EvipNode` Managed Object Instances can be possible in a later MTAS release.

### 3.3.3

#### Create Additional ALBs

Create the additional ALBs – in-line with the selected traffic separation profile – with the following template. Use the respective values for the `serviceIp` and the `ipsecServiceIp` parameters.



```

ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<mtas_bar|⇒
    mtas_om_pdl|mtas_om_pdr|mtas_sig_pdl|mtas_sig_pdr>
serviceIp=<fc00::202|fc00::203|fc00::204|fc00::205|fc00::206>
ipsecServiceIp=<fc00::302|fc00::303|fc00::304|fc00::305|fc00::306>
EvipErsipParams=1
EvipParam=hi_watermark
value="80"
up
EvipParam=lo_watermark
value="20"
up
EvipParam=parse_ss_interval
value="5"
up
EvipParam=timeout_connection_session
value="30"
up
EvipParam=timeout_requested_resources
value="10"
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<mtas_bar|⇒
    mtas_om_pdl|mtas_om_pdr|mtas_sig_pdl|mtas_sig_pdr>,EvipLbes=1
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<mtas_bar|⇒
    mtas_om_pdl|mtas_om_pdr|mtas_sig_pdl|mtas_sig_pdr>,EvipSes=1
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<mtas_bar|⇒
    mtas_om_pdl|mtas_om_pdr|mtas_sig_pdl|mtas_sig_pdr>,EvipTargetPools=1
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<mtas_bar|⇒
    mtas_om_pdl|mtas_om_pdr|mtas_sig_pdl|mtas_sig_pdr>,EvipVips=1
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<mtas_bar|⇒
    mtas_om_pdl|mtas_om_pdr|mtas_sig_pdl|mtas_sig_pdr>,EvipFees=1
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<mtas_bar|⇒
    mtas_om_pdl|mtas_om_pdr|mtas_sig_pdl|mtas_sig_pdr>,EvipFlowPolicies=1
top
commit -s
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<mtas_bar|⇒
    mtas_om_pdl|mtas_om_pdr|mtas_sig_pdl|mtas_sig_pdr>
EvipTargetPool=Pls_rr
allUndesignated="yes"
stickyGroup="no"
udpStateless="yes"
top
commit -s

```

### 3.3.4 Create LBEs and SEs

Load Balancer and Security Elements cannot be created within an ALB until the floating EvipNodes and the ALB itself do not exist. Once they are defined, the LBE and SE entities can be created:

```

ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<mtas_om|mtas_sig|mtas_bar|⇒
    mtas_om_pdl|mtas_om_pdr|mtas_sig_pdl|mtas_sig_pdr>,EvipLbes=1
EvipLbe=lbe_1
node="3"
up
EvipLbe=lbe_2
node="4"
up
EvipLbe=lbe_3
node="5"
up
EvipLbe=lbe_4
node="6"
up
EvipLbe=lbe_5
node="7"
up
EvipLbe=lbe_6
node="8"
up
EvipLbe=lbe_7
node="9"
up
EvipLbe=lbe_8

```



```
node="10"
up
EvipLbe=lbe_9
node="11"
up
EvipLbe=lbe_10
node="12"
up
EvipLbe=lbe_11
node="13"
up
EvipLbe=lbe_12
node="14"
up
EvipLbe=lbe_13
node="15"
top
commit -s
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<mtas_om|mtas_sig|mtas_bar|⇒
mtas_om_pdl|mtas_om_pdr|mtas_sig_pdl|mtas_sig_pdr>,EvipSe=1
EvipSe=se_1
node="3"
up
EvipSe=se_2
node="4"
up
EvipSe=se_3
node="5"
up
EvipSe=se_4
node="6"
up
EvipSe=se_5
node="7"
up
EvipSe=se_6
node="8"
up
EvipSe=se_7
node="9"
up
EvipSe=se_8
node="10"
up
EvipSe=se_9
node="11"
up
EvipSe=se_10
node="12"
up
EvipSe=se_11
node="13"
up
EvipSe=se_12
node="14"
up
EvipSe=se_13
node="15"
top
commit -s
```

### 3.3.5 Adjust the lbeHash Parameter Value

If the IPsec feature is not used, then the recommended setting of the parameter for traffic distribution lbeHash is 5-tuple.

```
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<mtas_om|mtas_sig|mtas_bar|⇒
mtas_om_pdl|mtas_om_pdr|mtas_sig_pdl|mtas_sig_pdr>
lbeHash=5-tuple
top
commit -s
```



### 3.3.6 Create the FEE Elements

The Front-End Element (FEE) elements are carrying the MTAS VNF embedded routing instances, as they are referred to in the document. The next sections are providing instructions how to define them, depending on the chosen VIP routing strategy.

#### 3.3.6.1 Static Routing without BFD

A FEE element using static routing without BFD can be created by using the following commands:

```
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<ALB name>,EvipFees=1,⇒
  EvipFee=<FEE ID>
  externalInterface="<Interface ID>"
  extIfBridging="0"
  node="<EvipNode ID>"
  EvipRoutingSetup=static
  EvipParam=local_address
  value="<Local IPv4 address (in CIDR notation)>"
  up
  EvipParam=gateway
  value="<GW IPv4 address>"
  up
  up
  EvipRoutingSetup=static6
  EvipParam=local_address
  value="<Local IPv6 address (in CIDR notation)>"
  up
  EvipParam=gateway
  value="<GW IPv6 address>"
  top
  commit -s
```

Create all the FEE elements by following the template and substituting the parameter values that are taken from the following table:

Table 3 Summary of FEE Elements Within the MTAS VNF, Static Routing without BFD

ALB Name	Interface ID	FEE ID	EvipNode ID	Local IPv4 Address	GW IPv4 Address	Local IPv6 Address	GW IPv6 Address
mtas_om	eth2	fee_1	3	<for example 10.0.12.1/25>	<for example 10.0.12.124> (VRRP protected GW address)	<for example fd00:0:a00:c00::1/64>	<for example fd00:0:a00:c00::fffc/64> (VRRP protected GW address)
		fee_2	4	<for example 10.0.12.2/25>		<for example fd00:0:a00:c00::2/64>	
		fee_3	5	<for example 10.0.12.3/25>		<for example fd00:0:a00:c00::3/64>	
		fee_4	6	<for example 10.0.12.4/25>		<for example fd00:0:a00:c00::4/64>	
mtas_sig	eth3	fee_1	3	<for example 10.1.12.1/25>	<for example 10.1.12.124> (VRRP protected GW address)	<for example fd00:0:a01:c00::1/64>	<for example fd00:0:a01:c00::fffc/64> (VRRP protected GW address)
		fee_2	4	<for example 10.1.12.2/25>		<for example fd00:0:a01:c00::2/64>	
		fee_3	5	<for example 10.1.12.3/25>		<for example fd00:0:a01:c00::3/64>	
		fee_4	6	<for example 10.1.12.4/25>		<for example fd00:0:a01:c00::4/64>	



Table 3 Summary of FEE Elements Within the MTAS VNF, Static Routing without BFD

ALB Name	Interface ID	FEE ID	EvipNode ID	Local IPv4 Address	GW IPv4 Address	Local IPv6 Address	GW IPv6 Address
mtas_bar	eth4	fee_1	3	<for example 10.2.12.1/25>	<for example 10.2.12.124> (VRRP protected GW address)	<for example fd00:0:a02:c00::1/64>	<for example fd00:0:a02:c00::fffc/64> (VRRP protected GW address)
		fee_2	4	<for example 10.2.12.2/25>		<for example fd00:0:a02:c00::2/64>	
		fee_3	5	<for example 10.2.12.3/25>		<for example fd00:0:a02:c00::3/64>	
		fee_4	6	<for example 10.2.12.4/25>		<for example fd00:0:a02:c00::4/64>	
mtas_om_pdl	eth5	fee_1	3	<for example 10.3.12.1/25>	<for example 10.3.12.126>	<for example fd00:0:a03:c00::1/64>	<for example fd00:0:a03:c00::fffe/64>
		fee_2	4	<for example 10.3.12.2/25>		<for example fd00:0:a03:c00::2/64>	
mtas_om_pdr	eth6	fee_1	5	<for example 10.4.12.1/25>	<for example 10.4.12.126>	<for example fd00:0:a04:c00::1/64>	<for example fd00:0:a04:c00::fffe/64>
		fee_2	6	<for example 10.4.12.2/25>		<for example fd00:0:a04:c00::2/64>	
mtas_sig_pdl	eth7	fee_1	3	<for example 10.5.12.1/25>	<for example 10.5.12.126>	<for example fd00:0:a05:c00::1/64>	<for example fd00:0:a05:c00::fffe/64>
		fee_2	4	<for example 10.5.12.2/25>		<for example fd00:0:a05:c00::2/64>	
mtas_sig_pdr	eth8	fee_1	5	<for example 10.6.12.1/25>	<for example 10.6.12.126>	<for example fd00:0:a06:c00::1/64>	<for example fd00:0:a06:c00::fffe/64>
		fee_2	6	<for example 10.6.12.2/25>		<for example fd00:0:a06:c00::2/64>	

**Note:** The IP addresses are site-specific, an alignment is required. The other parameters (ALB name, FEE ID, Interface ID, and EvipNode ID) must not be changed.

### 3.3.6.2 Static Routing with BFD

A FEE element using static routing can be created by using the following template:



```

ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<ALB name>,EvipFees=1,⇒
  EvipFee=<FEE ID>
externalInterface="<Interface ID>"
extIfBridging="0"
node="<EvipNode ID>"
EvipRoutingSetup=bfd_static
EvipParam=multiplier
value="3"
up
EvipParam=minrx
value="300"
up
EvipParam=local_address
value="<Local IPv4 address (in CIDR notation)>"
up
EvipParam=gateway
value="<GW IPv4 address>"
up
EvipParam=echo
value="no"
up
EvipParam=bfd_interval
value="300"
up
EvipRoutingSetup=bfd_static6
EvipParam=multiplier
value="3"
up
EvipParam=minrx
value="300"
up
EvipParam=local_address
value="<Local IPv6 address (in CIDR notation)>"
up
EvipParam=gateway
value="<GW IPv6 address>"
up
EvipParam=echo
value="no"
up
EvipParam=bfd_interval
value="300"
top
commit -s

```

Create all the FEE elements by following the template and substituting the parameter values that are taken from the following table:

Table 4 Summary of FEE Elements Within the MTAS VNF, Static Routing with BFD

ALB Name	Interface ID	FEE ID	EvipNode ID	Local IPv4 Address	GW IPv4 Address	Local IPv6 Address	GW IPv6 Address
mtas_o m	eth2	fee_1	3	<for example 10.0.12.1/25>	<for example 10.0.12.125>	<for example fd00:0:a00:c00::1/64>	<for example fd00:0:a00:c00::fffd/64>
		fee_2	4	<for example 10.0.12.2/25>		<for example fd00:0:a00:c00::2/64>	
		fee_3	5	<for example 10.0.12.3/25>	<for example 10.0.12.126>	<for example fd00:0:a00:c00::3/64>	<for example fd00:0:a00:c00::fffe/64>
		fee_4	6	<for example 10.0.12.4/25>		<for example fd00:0:a00:c00::4/64>	



Table 4 Summary of FEE Elements Within the MTAS VNF, Static Routing with BFD

ALB Name	Interface ID	FEE ID	EvipNode ID	Local IPv4 Address	GW IPv4 Address	Local IPv6 Address	GW IPv6 Address
mtas_sing	eth3	fee_1	3	<for example 10.1.12.1/25>	<for example 10.1.12.125>	<for example fd00:0:a01:c00::1/64>	<for example fd00:0:a01:c00::fffd/64>
		fee_2	4	<for example 10.1.12.2/25>		<for example fd00:0:a01:c00::2/64>	
		fee_3	5	<for example 10.1.12.3/25>	<for example 10.1.12.126>	<for example fd00:0:a01:c00::3/64>	<for example fd00:0:a01:c00::fffe/64>
		fee_4	6	<for example 10.1.12.4/25>		<for example fd00:0:a01:c00::4/64>	
mtas_bar	eth4	fee_1	3	<for example 10.2.12.1/25>	<for example 10.2.12.125>	<for example fd00:0:a02:c00::1/64>	<for example fd00:0:a02:c00::fffd/64>
		fee_2	4	<for example 10.2.12.2/25>		<for example fd00:0:a02:c00::2/64>	
		fee_3	5	<for example 10.2.12.3/25>	<for example 10.2.12.126>	<for example fd00:0:a02:c00::3/64>	<for example fd00:0:a02:c00::fffe/64>
		fee_4	6	<for example 10.2.12.4/25>		<for example fd00:0:a02:c00::4/64>	
mtas_om_pdl	eth5	fee_1	3	<for example 10.3.12.1/25>	<for example 10.3.12.126>	<for example fd00:0:a03:c00::1/64>	<for example fd00:0:a03:c00::fffe/64>
		fee_2	4	<for example 10.3.12.2/25>		<for example fd00:0:a03:c00::2/64>	
mtas_om_pdr	eth6	fee_1	5	<for example 10.4.12.1/25>	<for example 10.4.12.126>	<for example fd00:0:a04:c00::1/64>	<for example fd00:0:a04:c00::fffe/64>
		fee_2	6	<for example 10.4.12.2/25>		<for example fd00:0:a04:c00::2/64>	
mtas_sing_pdl	eth7	fee_1	3	<for example 10.5.12.1/25>	<for example 10.5.12.126>	<for example fd00:0:a05:c00::1/64>	<for example fd00:0:a05:c00::fffe/64>
		fee_2	4	<for example 10.5.12.2/25>		<for example fd00:0:a05:c00::2/64>	
mtas_sing_pdr	eth8	fee_1	5	<for example 10.6.12.1/25>	<for example 10.6.12.126>	<for example fd00:0:a06:c00::1/64>	<for example fd00:0:a06:c00::fffe/64>
		fee_2	6	<for example 10.6.12.2/25>		<for example fd00:0:a06:c00::2/64>	

**Note:** The IP addresses are site-specific, an alignment is required. The other parameters (ALB name, FEE ID, Interface ID, and EvipNode ID) must not be changed.

### 3.3.6.3 Dynamic Routing

A FEE element using dynamic routing can be created by using the following template:





```

ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<ALB name>,EvipFees=1,⇒
  EvipFee=<FEE ID>
externalInterface="<Interface ID>"
extIfBridging="0"
node="<EvipNode ID>"
EvipRoutingSetup=ospfv2
EvipParam=local_address
value="<Local IPv4 address (in CIDR notation)>"
up
EvipParam=area
value="<Area ID>"
up
EvipParam=area_type
value="stub"
up
EvipParam=router_id
value="<Router ID>"
up
EvipParam=router_priority
value="0"
up
up
EvipRoutingSetup=bfd_ospfv2
EvipParam=bfd_interval
value="300"
up
EvipParam=echo
value="no"
up
EvipParam=minrx
value="300"
up
EvipParam=multiplier
value="3"
up
up
EvipRoutingSetup=ospfv3
EvipParam=area
value="<Area ID>"
up
EvipParam=area_type
value="stub"
up
EvipParam=router_id
value="<Router ID>"
up
EvipParam=router_priority
value="0"
up
up
EvipRoutingSetup=bfd_ospfv3
EvipParam=bfd_interval
value="300"
up
EvipParam=echo
value="no"
up
EvipParam=minrx
value="300"
up
EvipParam=multiplier
value="3"
top
commit -s

```

Create all the FEE elements by following the template and substituting the parameter values that are taken from the following table:



Table 5 Summary of FEE Elements Within the MTAS VNF, Dynamic Routing

ALB Name	Interface ID	FEE ID	EvipNode ID	Local IPv4 Address	OSPFv2 Area ID	OSPFv2 Router ID	OSPFv3 Area ID	OSPFv3 Router ID
mtas_o m	eth2	fee_1	3	<for example 10.0.12.1/25>	<for example 0.0.0.1>	<for example 1.1.1.1>	<for example 0.0.0.1>	<for example 1.1.1.1>
		fee_2	4	<for example 10.0.12.2/25>		<for example 1.1.1.2>		<for example 1.1.1.2>
		fee_3	5	<for example 10.0.12.3/25>		<for example 1.1.1.3>		<for example 1.1.1.3>
		fee_4	6	<for example 10.0.12.4/25>		<for example 1.1.1.4>		<for example 1.1.1.4>
mtas_si g	eth3	fee_1	3	<for example 10.1.12.1/25>	<for example 0.0.0.2>	<for example 1.1.1.1>	<for example 0.0.0.2>	<for example 1.1.1.1>
		fee_2	4	<for example 10.1.12.2/25>		<for example 1.1.1.2>		<for example 1.1.1.2>
		fee_3	5	<for example 10.1.12.3/25>		<for example 1.1.1.3>		<for example 1.1.1.3>
		fee_4	6	<for example 10.1.12.4/25>		<for example 1.1.1.4>		<for example 1.1.1.4>
mtas_b ar	eth4	fee_1	3	<for example 10.2.12.1/25>	<for example 0.0.0.3>	<for example 1.1.1.1>	<for example 0.0.0.3>	<for example 1.1.1.1>
		fee_2	4	<for example 10.2.12.2/25>		<for example 1.1.1.2>		<for example 1.1.1.2>
		fee_3	5	<for example 10.2.12.3/25>		<for example 1.1.1.3>		<for example 1.1.1.3>
		fee_4	6	<for example 10.2.12.4/25>		<for example 1.1.1.4>		<for example 1.1.1.4>
mtas_o m_pdl	eth5	fee_1	3	<for example 10.3.12.1/25>	<for example 0.0.0.4>	<for example 1.1.1.1>	<for example 0.0.0.4>	<for example 1.1.1.1>
		fee_2	4	<for example 10.3.12.2/25>		<for example 1.1.1.2>		<for example 1.1.1.2>
mtas_o m_pdr	eth6	fee_1	5	<for example 10.4.12.1/25>	<for example 0.0.0.5>	<for example 1.1.1.1>	<for example 0.0.0.5>	<for example 1.1.1.1>
		fee_2	6	<for example 10.4.12.2/25>		<for example 1.1.1.2>		<for example 1.1.1.2>
mtas_si g_pdl	eth7	fee_1	3	<for example 10.5.12.1/25>	<for example 0.0.0.6>	<for example 1.1.1.1>	<for example 0.0.0.6>	<for example 1.1.1.1>
		fee_2	4	<for example 10.5.12.2/25>		<for example 1.1.1.2>		<for example 1.1.1.2>
mtas_si g_pdr	eth8	fee_1	5	<for example 10.6.12.1/25>	<for example 0.0.0.7>	<for example 1.1.1.1>	<for example 0.0.0.7>	<for example 1.1.1.1>
		fee_2	6	<for example 10.6.12.2/25>		<for example 1.1.1.2>		<for example 1.1.1.2>

**Note:** The IP addresses, Area IDs, and Router IDs are site-specific, an alignment is required. The other parameters (ALB name, FEE ID, Interface ID, and EvipNode ID) must not be changed.



### 3.3.7 Define the VIP Addresses

Either from Table 1 or Table 2, take the ALB/VIP pairs, and substitute their names into the following template:

```
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<mtas_om|mtas_sig|mtas_bar|⇒
    mtas_om_pdl|mtas_om_pdr|mtas_sig_pdl|mtas_sig_pdr>,EvipVips=1
EvipVip=<VIP Address>
deflt="no"
equivSrcAddr="no"
top
commit -s
```

### 3.3.8 Create Flow Policies

After performing all the configurations, the Service Access Points (representations of the MTAS Logical Interfaces) are still closed. To enable incoming streams passing through the `mtas_om` and `mtas_sig` containers (ALBs), flow policies must be defined.

#### 3.3.8.1 Flow Policies for TCP/UDP

Flow policies for incoming TCP/UDP streams can be created by using the following template:

```
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<ALB name>,EvipFlowPolicies=1,⇒
    EvipFlowPolicy=<FlowPolicy_ID|6FlowPolicy_ID>
dest=<VIPv4 or VIPv6 address>
addressFamily=<ipv4|ipv6>
protocol=<Protocol family>
destPort=<DST port>
targetPool=<Target pool>
top
commit -s
```

Create all the required flow policies, in-line with the next tables. Refer to MTAS Hardening Guide for further details regarding MTAS supported TCP/UDP SAPs.

**TCP/UDP Flow Policies in Traffic Separation profile1****Table 6** Summary of MTAS Flow Policies for Incoming TCP/UDP Streams, Traffic Separation profile1

ALB Name	FlowPolicy_ID	Protocol Family	DST Port	Target Pool	VIP Address
mtas_om	(6)ProvisioningLDAP	tcp	17323	PLs_rr	<OM VIPv4/VIPv6 Address>
	(6)ProvisioningLDAP_secure	tcp	17423	PLs_rr	<OM VIPv4/VIPv6 Address>
	(6)RoRf_1	tcp	3868	PLs_rr	<OM VIPv4/VIPv6 Address>
	(6)RoRf_2	tcp	3869	PLs_rr	<OM VIPv4/VIPv6 Address>
	(6)RoRf_3	tcp	3870	PLs_rr	<OM VIPv4/VIPv6 Address>
	(6)RoRf_4	tcp	3871	PLs_rr	<OM VIPv4/VIPv6 Address>
	(6)RoRf_5	tcp	3872	PLs_rr	<OM VIPv4/VIPv6 Address>
	(6)CAI3G	tcp	8095	PLs_rr	<CAI3G VIPv4/VIPv6 Address>
	(6)CAI3G-HTTPS	tcp	8443	PLs_rr	<CAI3G VIPv4/VIPv6 Address>



**Table 6** Summary of MTAS Flow Policies for Incoming TCP/UDP Streams, Traffic Separation profile1

ALB Name	FlowPolicy_ID	Protocol Family	DST Port	Target Pool	VIP Address
mtas_sig	(6)XCAP	tcp	8090	PLs_rr	<Ut VIPv4/VIPv6 Address>
	(6)ShDh_1	tcp	3868	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)ShDh_2	tcp	3869	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)ShDh_3	tcp	3870	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)ShDh_4	tcp	3871	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)ShDh_5	tcp	3872	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_01t	tcp	5060	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_01u	udp	5060	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_02t	tcp	5082	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_02u	udp	5082	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_03t	tcp	5083	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_03u	udp	5083	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_04t	tcp	5084	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_04u	udp	5084	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_05t	tcp	5085	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_05u	udp	5085	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_06t	tcp	5086	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_06u	udp	5086	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_07t	tcp	5087	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_07u	udp	5087	PLs_rr	<Traffic VIPv4/VIPv6 Address>



Table 6 Summary of MTAS Flow Policies for Incoming TCP/UDP Streams, Traffic Separation profile1

ALB Name	FlowPolicy_ID	Protocol Family	DST Port	Target Pool	VIP Address
mtas_sig (cont.)	(6)SIP_08t	tcp	5088	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_08u	udp	5088	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_09t	tcp	5089	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_09u	udp	5089	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_10t	tcp	5090	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_10u	udp	5090	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_11t	tcp	5094	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_11u	udp	5094	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_12t	tcp	5160	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_12u	udp	5160	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_13t	tcp	5161	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_13u	udp	5161	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_14t	tcp	5162	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_14u	udp	5162	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_15t	tcp	5163	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_15u	udp	5163	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)soap_http	tcp	9080	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)CCMP	tcp	8096	PLs_rr	<Traffic VIPv4/VIPv6 Address>



### TCP/UDP Flow Policies in Traffic Separation profile2

Table 7 Summary of MTAS Flow Policies for Incoming TCP/UDP Streams, Traffic Separation profile2

ALB Name	FlowPolicy_ID	Protocol Family	DST Port	Target Pool	VIP Address
mtas_om	(6)ProvisioningLDAP	tcp	17323	PLs_rr	<OM VIPv4/VIPv6 Address>
	(6)ProvisioningLDAP_secure	tcp	17423	PLs_rr	<OM VIPv4/VIPv6 Address>
	(6)CAI3G	tcp	8095	PLs_rr	<CAI3G VIPv4/VIPv6 Address>
	(6)CAI3G-HTTPS	tcp	8443	PLs_rr	<CAI3G VIPv4/VIPv6 Address>
mtas_sig	(6)XCAP	tcp	8090	PLs_rr	<Ut VIPv4/VIPv6 Address>
	(6)SIP_01t	tcp	5060	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_01u	udp	5060	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_02t	tcp	5082	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_02u	udp	5082	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_03t	tcp	5083	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_03u	udp	5083	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_04t	tcp	5084	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_04u	udp	5084	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_05t	tcp	5085	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_05u	udp	5085	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_06t	tcp	5086	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_06u	udp	5086	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_07t	tcp	5087	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_07u	udp	5087	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_08t	tcp	5088	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_08u	udp	5088	PLs_rr	<Traffic VIPv4/VIPv6 Address>



Table 7 Summary of MTAS Flow Policies for Incoming TCP/UDP Streams, Traffic Separation profile2

ALB Name	FlowPolicy_ID	Protocol Family	DST Port	Target Pool	VIP Address
mtas_sig (cont.)	(6)SIP_09t	tcp	5089	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_09u	udp	5089	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_10t	tcp	5090	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_10u	udp	5090	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_11t	tcp	5094	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_11u	udp	5094	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_12t	tcp	5160	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_12u	udp	5160	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_13t	tcp	5161	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_13u	udp	5161	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_14t	tcp	5162	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_14u	udp	5162	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_15t	tcp	5163	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)SIP_15u	udp	5163	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)soap_http	tcp	9080	PLs_rr	<Traffic VIPv4/VIPv6 Address>
	(6)CCMP	tcp	8096	PLs_rr	<Traffic VIPv4/VIPv6 Address>

**Note:** Opening ports of unused services must be avoided.

### 3.3.8.2 Flow Policies for SCTP

Flow policies for incoming SCTP streams can be created by using the following template:

```
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<ALB name>,EvipFlowPolicies=1,⇒
    EvipFlowPolicy=<FlowPolicy_ID|6FlowPolicy_ID>
dest=<VIPv4 or VIPv6 address>
addressFamily=<ipv4|ipv6>
protocol=sctp
soGrp=1011250
top
commit -s
```





Create all the required flow policies in-line with the next tables. Refer to MTAS Hardening Guide for further details regarding MTAS supported SCTP SAPs.

### SCTP Flow Policies in Traffic Separation profile1

Table 8 Summary of MTAS Flow Policies for Incoming SCTP Streams, Traffic Separation profile1

ALB Name	FlowPolicy_ID	VIP Address
mtas_sig	(6)SCTP_H248_SH	<Traffic VIPv4/VIPv6 Address>
	(6)SCTP_SIGTRAN_SH1	<SIGTRAN VIPv4/VIPv6 SH1 Address>
	(6)SCTP_SIGTRAN_SH2	<SIGTRAN VIPv4/VIPv6 SH2 Address>

### SCTP Flow Policies in Traffic Separation profile2

Table 9 Summary of MTAS Flow Policies for Incoming SCTP Streams, Traffic Separation profile2

ALB Name	FlowPolicy_ID	VIP Address
mtas_om_pdl	(6)SCTP_RoRf_MH1	<Ro/Rf/CDS VIPv4/VIPv6 MH1 Address>
mtas_om_pdr	(6)SCTP_RoRf_MH2	<Ro/Rf/CDS VIPv4/VIPv6 MH2 Address>
mtas_sig	(6)SCTP_H248_SH	<Traffic VIPv4/VIPv6 Address>
mtas_sig_pdl	(6)SCTP_SIGTRAN_MH1	<SIGTRAN VIPv4/VIPv6 MH1 Address>
	(6)SCTP_Traffic_MH1	<Traffic VIPv4/VIPv6 MH1 Address>
mtas_sig_pdr	(6)SCTP_SIGTRAN_MH2	<SIGTRAN VIPv4/VIPv6 MH2 Address>
	(6)SCTP_Traffic_MH2	<Traffic VIPv4/VIPv6 MH2 Address>

**Note:** Opening ports of unused services must be avoided.

### 3.3.9 Activate the ALBs

Once the changes are done, the configured ALBs must be activated.

```
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=<mtas_om|mtas_sig|mtas_bar|⇒
mtas_om_pdl|mtas_om_pdr|mtas_sig_pdl|mtas_sig_pdr>,activate
```