

MTAS Internal and External Connectivity

MTAS

DESCRIPTION

Copyright

© Ericsson AB 2016. 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

1	Introduction	1
1.1	IMS Security Principles	1
1.2	Cloud Deployment Generic Principles	5
2	MTAS Internal Communication	7
2.1	TIPC	7
2.2	INET IPv4	8
2.3	INET IPv6 over MAC VLAN	11
3	MTAS External Communication	13
3.1	Operation and Maintenance	13
3.2	Virtual IP Access - OM_SC VPN	15
3.3	Virtual IP Access - Sig_SC VPN	19
3.4	External Connectivity Through ASIC Routers	20
3.5	Appendix - Configuration Templates	22





1 Introduction

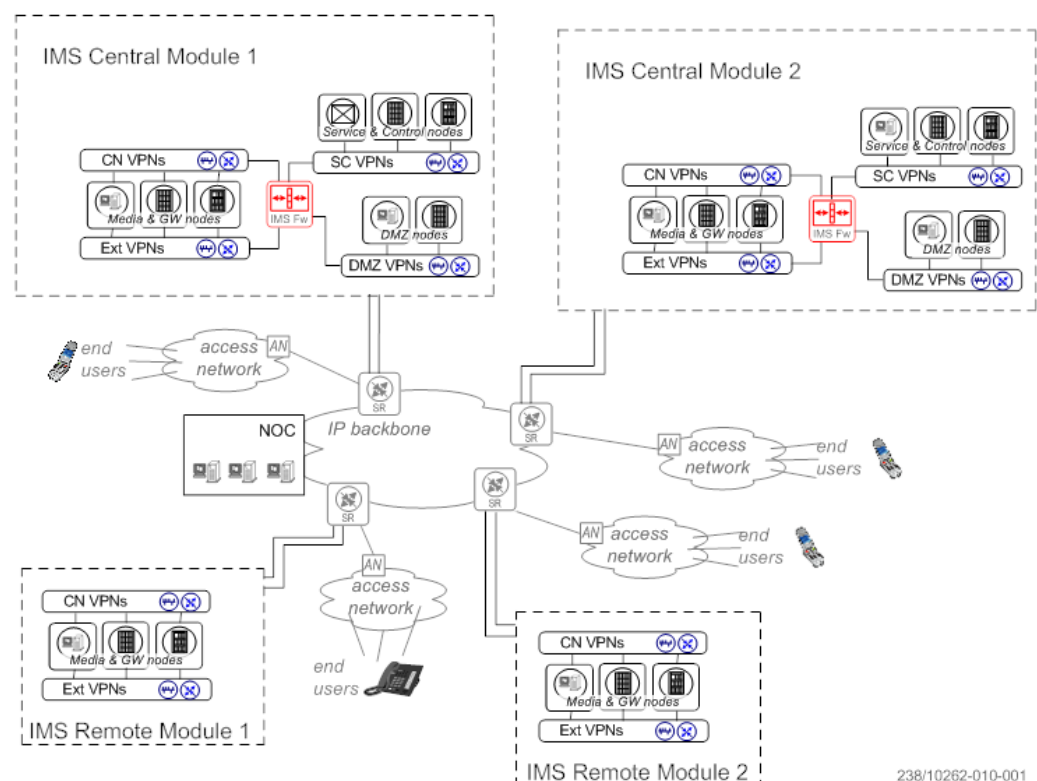
This document provides description about the internal and external connectivity of the MTAS product that is designed around the generic security principles and design rules of the Ericsson IMS System. The document also describes how these can be realized on cloud environments.

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.



238/10262-010-001

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.

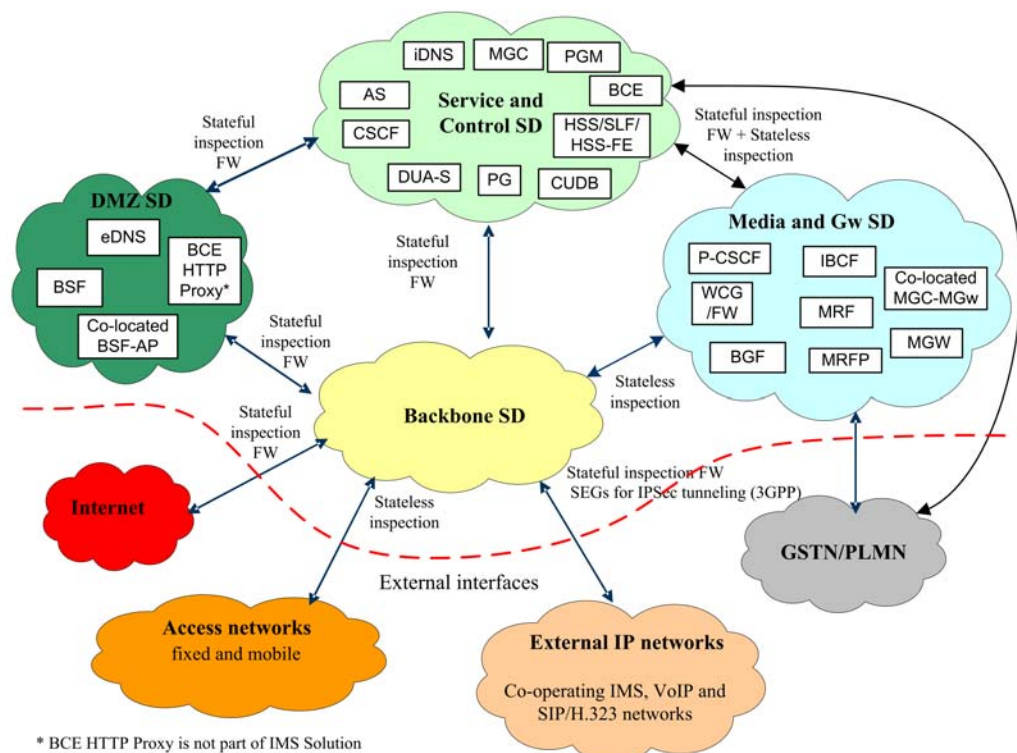


Figure 2 IMS Security Domains

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

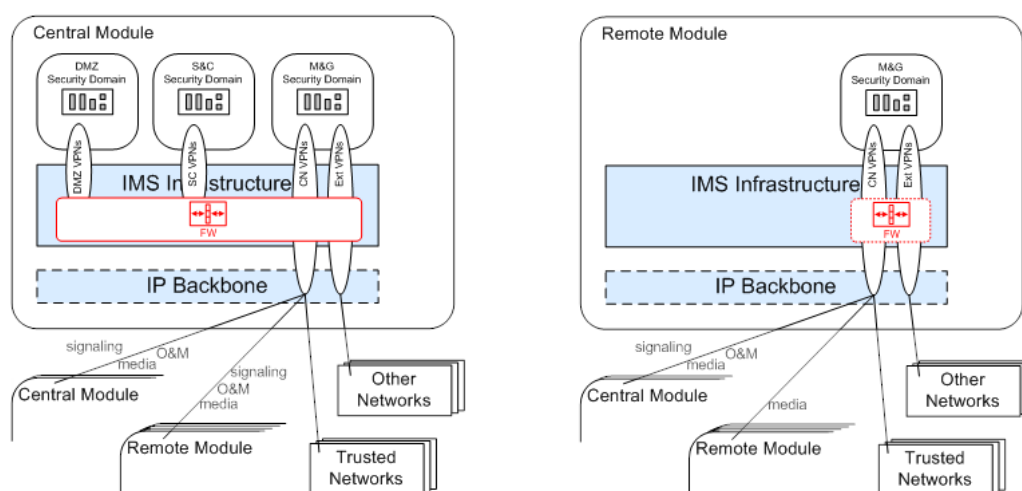
1.1.3 Traffic Separation

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 and Sig_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.

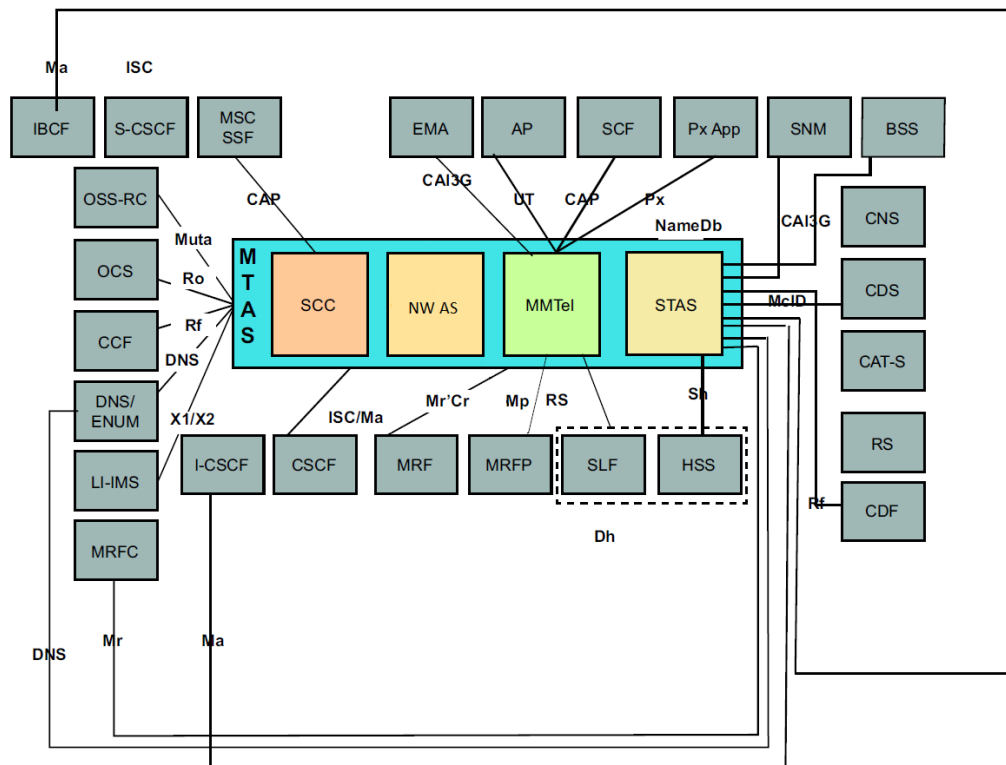


Figure 4 Interfaces and Protocols Supported by MTAS

The Service Access Points are single-homed IPv4/IPv6 UDP/TCP/SCTP sockets that are bound to the OM_SC and Sig_SC VPNs as follows (VPN → IP → logical interface).

- OM_SC
 - SC-1 System Management IP
 - SC-1 maintenance
 - NTP synchronization, client side (used when the OM IP is not active on SC-1)
 - SC-2 System Management IP
 - SC-2 maintenance
 - NTP synchronization, client side (used when the OM IP is not active on SC-2)
 - OM IP ("Muta MIP")
 - Muta interfaces (COM CLI, COM Netconf, COM SNMP, COM FileM SFTP)
 - Arwa server connectivity



- OM VIP
 - Provisioning LDAP
 - Ro- and Rf interfaces
- CAI3G VIP
 - CAI3G interface
- Sig_SC
 - Ut VIP
 - Ut interface
 - SIGTRAN VIP 1
 - CAP interface (ASP/IPSP 1)
 - SIGTRAN VIP 2
 - CAP interface (ASP/IPSP 2)
 - Traffic VIP
 - All the other logical interfaces

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
- Do not use guest VLAN tagging - Fulfilled
- Bonding must not be used on guest level - Fulfilled
- Guest OS to use the para-virtualization driver - Fulfilled
- Use OSPFv2 (and v3) as a routing protocol - Fulfilled
- Use 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 4 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 x SCs and 2 x PLs must exist in the system. The maximum configuration is 2+73 (2 x SCs and 73 x PLs). The VMs must be created on separate failure domains, on different Compute Nodes.

Figure 5 shows the MTAS VNF.

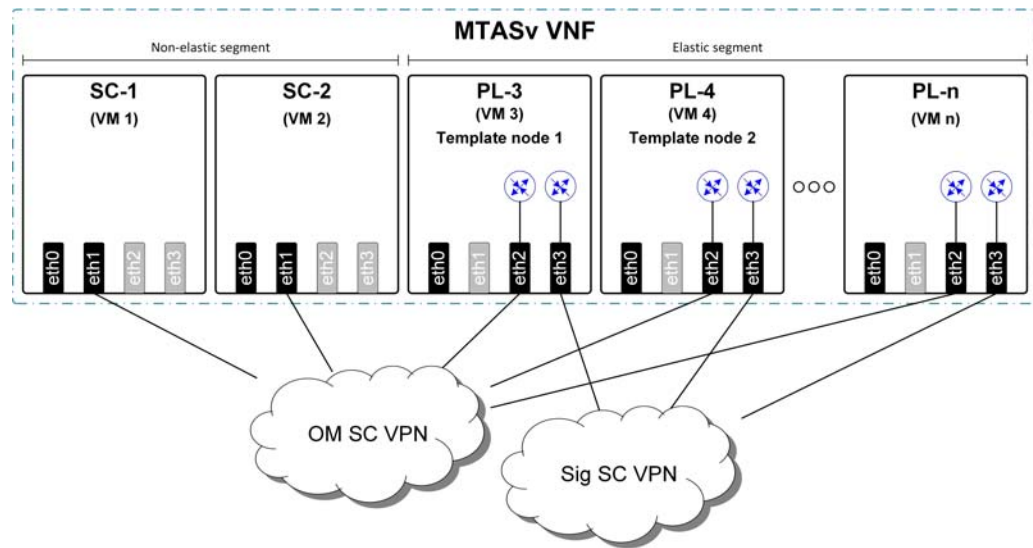


Figure 5 MTAS Virtual Network Function

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 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:

- eth0 is used for MTAS VNF internal communication on all the VMs
- eth1 is used for System Management and Operation and Maintenance connectivity inside the OM_SC VPN on the SCs
- eth2 is used for VIP connectivity inside the OM_SC VPN on the PLs.
- eth3 is used for VIP connectivity inside the Sig_SC VPN on the PLs.

The details about vNICs use are provided in the next chapters. The Maximum Transmission Unit (MTU) size on the vNICs is 1500 bytes.



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

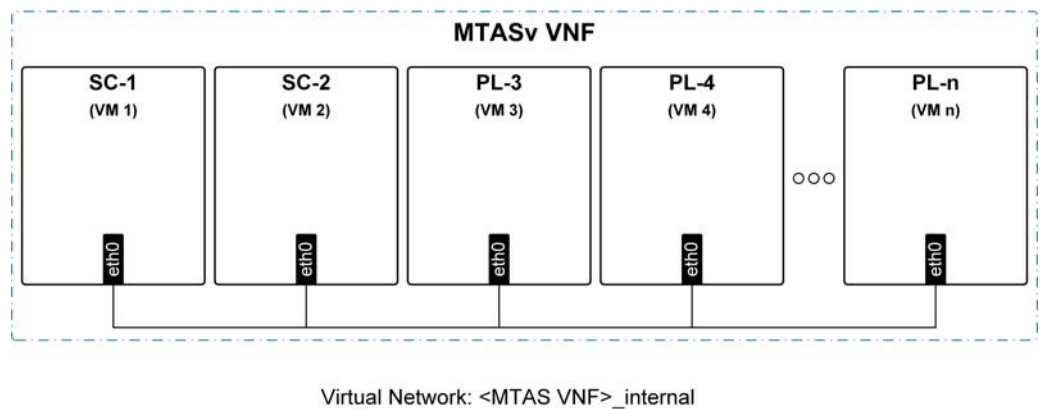


Figure 6 "<MTAS VNF>_internal" virtual network

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.

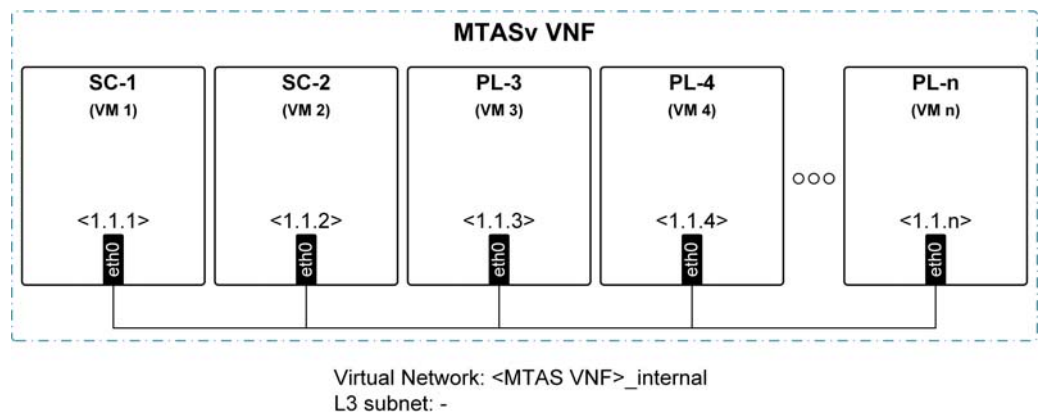


Figure 7 "<MTAS VNF>_internal" virtual network - TIPC

Upon the installation procedure, the rather simple TIPC logical network topology is established within the MTAS VNF automatically. The VMs, as TIPC nodes, are connected to each other in a full-meshed fashion through point-to-point links in Cluster = 1, Zone = 1.

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

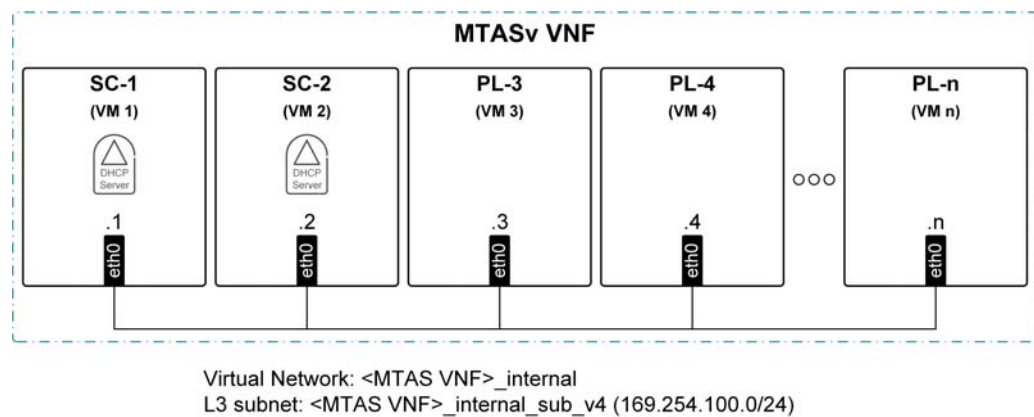


Figure 8 "<MTAS VNF>_internal" virtual network - INET IPv4

The system is using the 169.254.100.0/24 subnet over the "<MTAS VNF>_internal" virtual network infrastructure. Address allocation is secured by MTAS VNF internal mechanism, has no dependency on the DHCP service what the cloud infrastructure can provide. 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 cloud infrastructure must secure that the DHCP communication (BOOTP; UDP src/dst port: 67/68) is not filtered on 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 a Movable IP (MIP) address on the "<MTAS VNF>_internal" network.

Figure 9 shows the "<MTAS VNF>_internal" virtual network - TFTP over INET IPv4

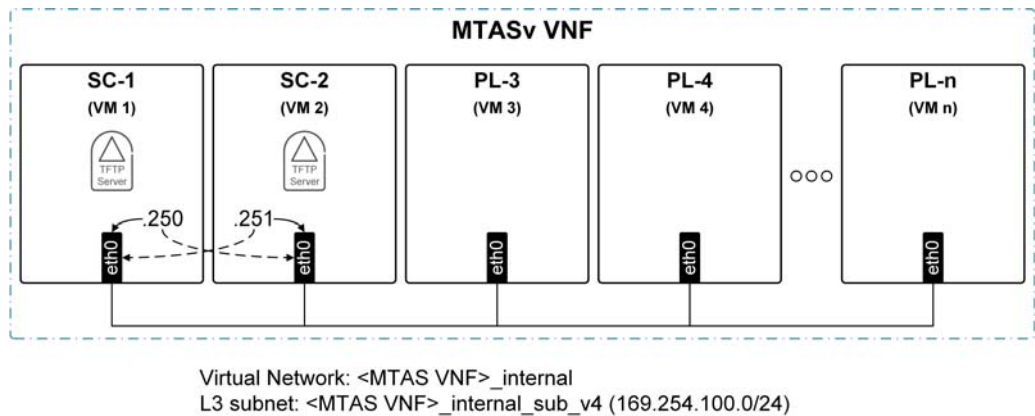


Figure 9 "<MTAS VNF>_internal" virtual network - TFTP over INET IPv4

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

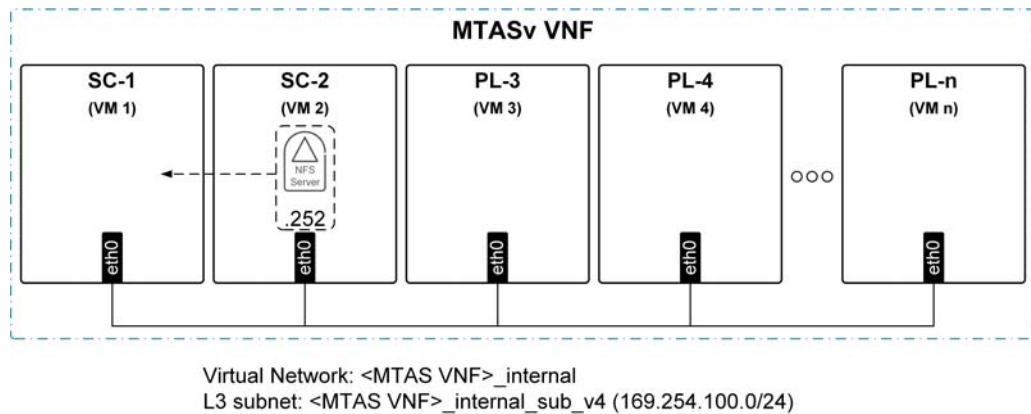


Figure 10 "**<MTAS VNF>_internal**" virtual network - NFS over INET IPv4

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. CP-M is available on either of the available PLs.

Figure 11 shows the "<MTAS VNF>_internal" virtual network - CP-M.

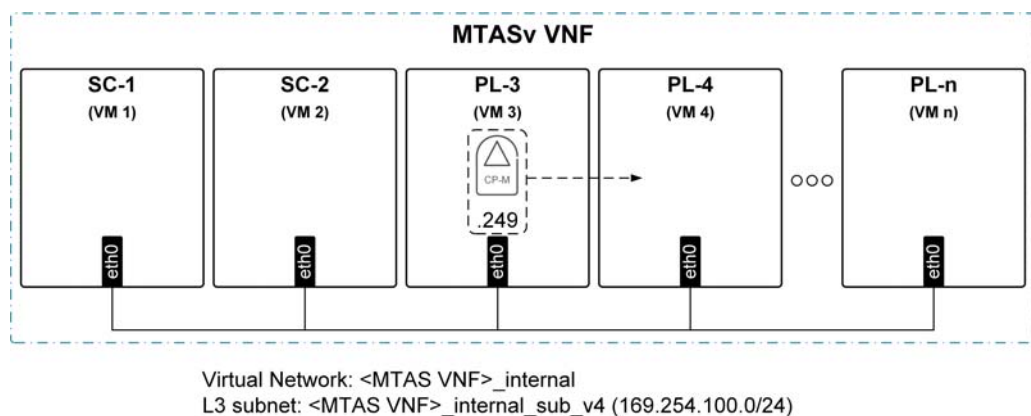


Figure 11 "**<MTAS VNF>_internal**" virtual network - CP-M

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 MAC VLAN

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

Figure 12 shows the "<MTAS VNF>_internal" virtual network - INET IPv6 over MAC VLANs.

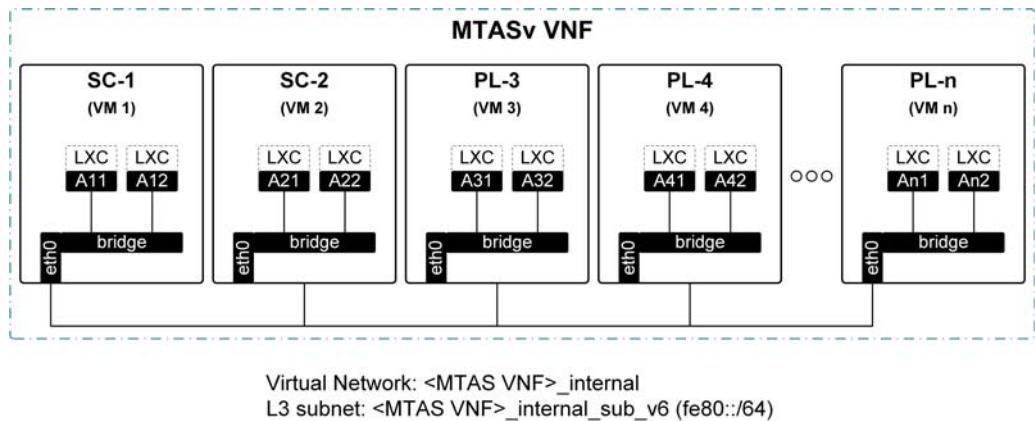


Figure 12 "<MTAS VNF>_internal" virtual network - INET IPv6 over MAC VLANs

The LXC networking is facilitated over MAC VLANs in bridge mode. The bridges are created on the vNIC (eth0) that is bound to the "<MTAS VNF>_internal" network. On L3 level, the communication is performed on the IPv6 link-local network. The MAC aliases (in Figure 12, A11 ... An2) for the LXC containers are defined through IPv6 link-local to MAC address conversion.

Because of bridge mode, the communication between two LXC containers that are on the same VM remains within the VM. But this is not the case between LXC containers that are residing on different VMs.

Attention!

Risk of system malfunction or traffic disturbance.

If the underlying cloud infrastructure is performing traffic policing based on Cloud Infrastructure Controller aware, virtual network port assigned vNIC MAC addresses, the policy rule set is to be loosened/extended. It must be secured that the communication between the VNF internal MAC aliases is not filtered by any means.





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 secured by soft routers, the deployment of the Software router VMs must take place on dedicated compute nodes, referred as "Network Nodes", to not 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 secured to the Linux shell of the System Controllers; intended to be used in emergency scenarios only.
- Secure 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 Arwa 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. However, it is secured that the COM NBI services are always reachable through these MIPs.

Figure 13 shows the MTAS VNF System Management Infrastructure.

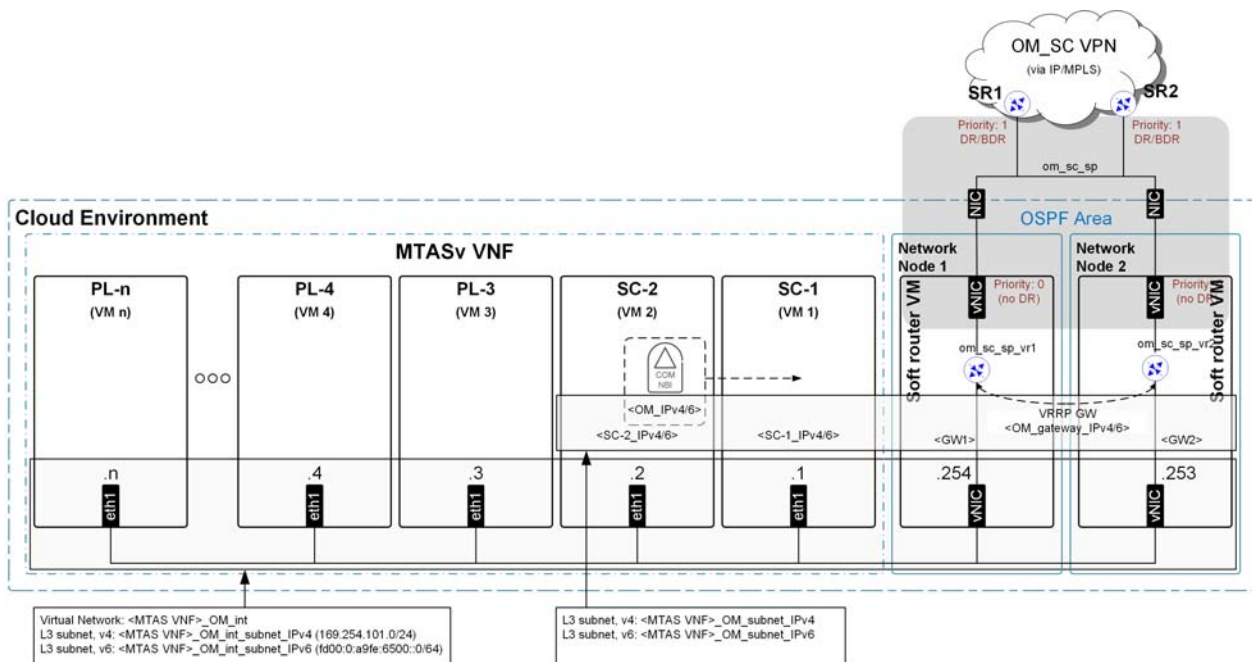


Figure 13 MTAS VNF System Management Infrastructure

3.1.2 Routing Design

All the VMs within the MTAS VNF are connected to the "<MTAS VNF>_OM_int" virtual network infrastructure through their eth1 interfaces. This infrastructure is internal and shared with the Soft Router instances. A fixed primary IP address is assigned to all the connected virtual network ports/vNICs from the "<MTAS VNF>_OM_int_subnet_IPv4/6" link-local subnet.

Additional IPv4/6 addresses are assigned to the connected eth1 of the SC (SC-1_IPv4/6, SC-2_IPv4/6) and the soft router interfaces (GW1, GW2) on the created broadcast domain. These addresses are taken from the "<MTAS VNF>_OM_subnet_IPv4/6" routable subnet whereby transit is enabled by a VRRP protected gateway (OM_gateway_IPv4/6). By setting the default route against this VRRP IP on the SCs, the system management connectivity of them can be secured at once.

Beside the SC and GW addresses, there is another address reserved and shared by the System Controllers, the OM_IPv4/6 (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 inbound and outbound streams. For the outbound streams (UDP: SNMP Trap; TCP: SFTP) it is secured by indicating the OM_IPv4/6 Movable IP as the preferred source address in the default route setting. This preference is only used on the SC where the MIP is active.



Attention!

Risk of system malfunction or traffic disturbance.

The "<MTAS VNF>_OM_subnet_IPv4/6" subnet is such an overlay what the virtual networking service 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_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_int" virtual network infrastructure (in the OpenStack case, use "allowed_address_pairs" setting on Neutron ports; the "<MTAS VNF>_OM_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_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. For additional details, see the next section.

3.2 Virtual IP Access - OM_SC VPN

3.2.1 Overview

The purpose of having this infrastructure in place is to secure connectivity to the MTAS services that are bound to Virtual IPs available through the OM_SC VPN. These services are:

- OM VIP
 - Services over the Provisioning LDAP interface
 - Services over the Ro- and Rf interfaces
- CAI3G VIP
 - Services over the CAI3G interface

The VMs that are playing the role of PL, but maximum 4 at any given moment, are connected to the "<MTAS VNF>_om_sc_sp" virtual network through the eth2 interface. Routing can either be dynamic or static on this infrastructure; dynamic is preferred. Mixed routing fashion is not supported in the IPv4/IPv6 dual-stack scenario. That is: either use dynamic for both IPv4 and IPv6, or use static for both of IPv4 and IPv6.

3.2.2 Dynamic Routing Design

The dynamic routing reference design is considering the following constraints:

- Not all soft router type supports multiple OSPF process instances. In turn, the om_sc_sp_vr1/2 and sig_sc_sp_vr1/2 routing instances to be secured by a different pair of soft routers.
- Virtual OSPF links cannot be configured in the routing instances that are residing in the MTAS VNF. Hence, and because every OSPF area must be connected to the backbone, the soft routers must be connected to the OSPF backbone.

Figure 14 shows the MTAS VNF Virtual IP Access - OM_SC VPN, Dynamic Routing Design.

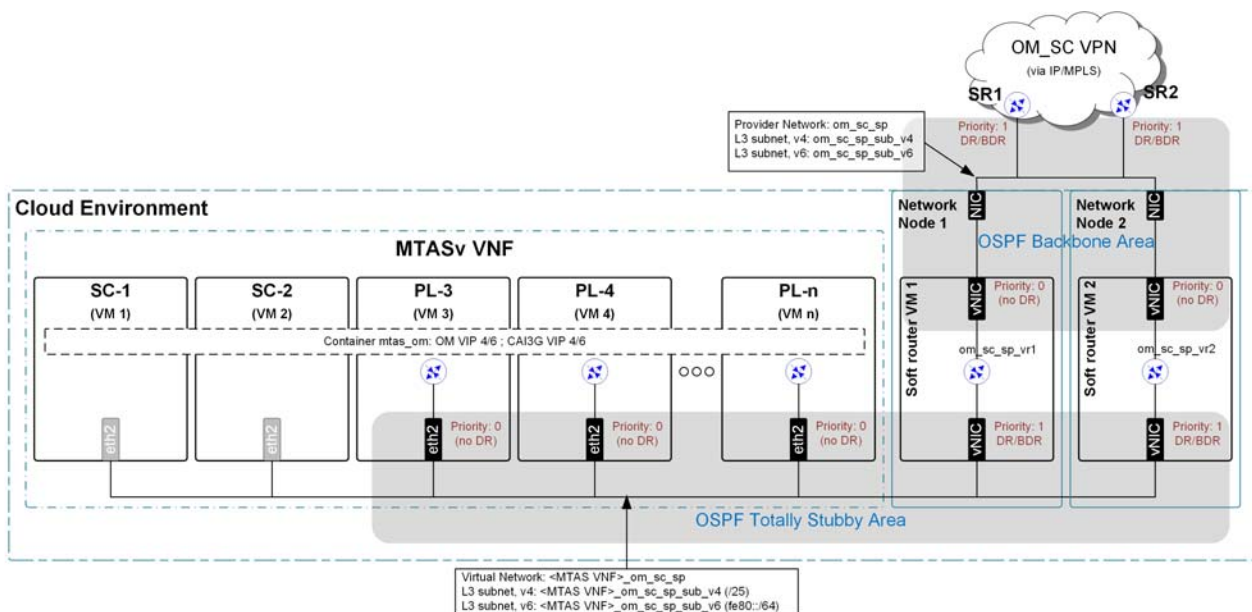


Figure 14 MTAS VNF Virtual IP Access - OM_SC VPN, Dynamic Routing Design

The 'om_sc_sp_vr1' and 'om_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 zero. 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, 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`).

Preferably, OSPF must be combined with BFD to achieve faster forwarding path failure detection.

3.2.3 Static Routing Design

Using static routing over the `<MTAS_VNF>_om_sc_sp` infrastructure is an option if the soft routers are supporting BFD. The following constraints are considered:

- A PL embedded routing instance can be configured statically with a single piece of default gateway, per INET flavor

Figure 15 shows the MTAS VNF Virtual IP Access - OM_SC VPN, static routing design.

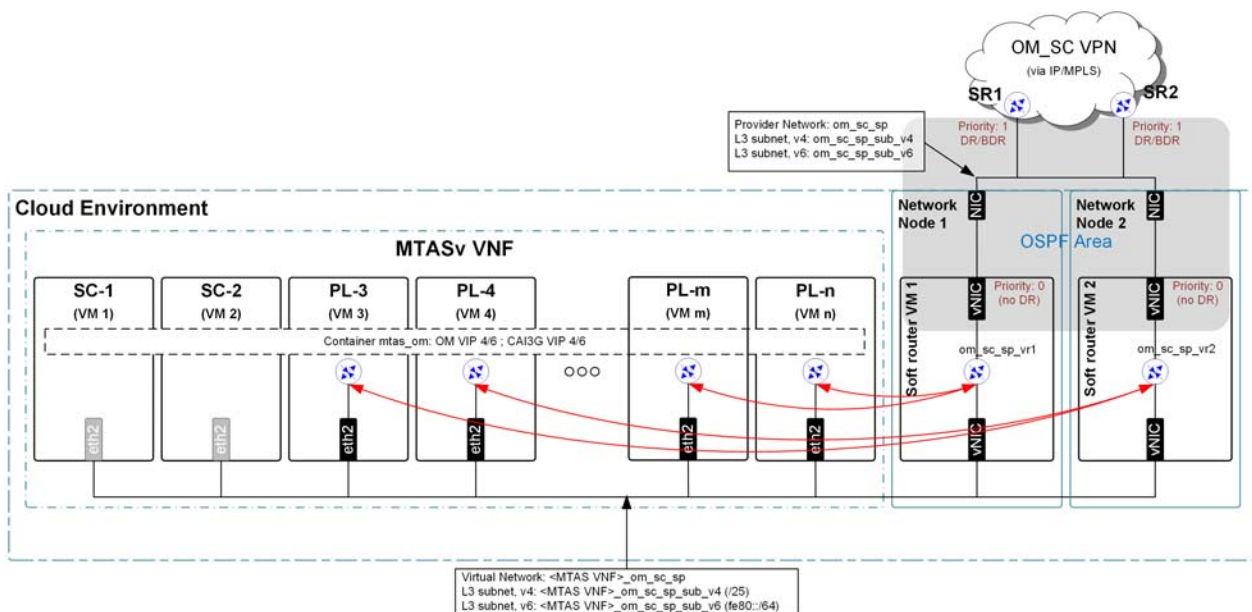


Figure 15 MTAS VNF Virtual IP Access - OM_SC VPN, static routing design

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

- Group 1
 - The 'om_sc_sp_vr1' soft router is statically configured as default gateway on the PLs in this group (PL-m and PL-n in Figure 15).
 - The PLs in this group are statically configured as possible next hops (with the same distance) towards the OM- and CAI3G VIPs in the 'om_sc_sp_vr1' soft router
 - Every statically configured route is protected with single-hop BFD session
- Group 2
 - The 'om_sc_sp_vr2' soft router is statically configured as default gateway on the PLs in this group (PL-3 and PL-4 in Figure 15).
 - The PLs in this group are statically configured as possible next hops (with the same distance) towards the OM- and CAI3G VIPs in the 'om_sc_sp_vr2' soft router
 - Every statically configured route is protected with single-hop BFD session

The routes towards the VIP addresses are injected into OSPF by the Soft Routers.



3.2.4 MTU

The Maximum Transmission Unit (MTU) size within the `mtas_om` container is 1452 bytes. 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 instance(s) can split such received IPv4 packet into smaller chunks (fragments).

Attention!

Risk of traffic disturbance. If PMTUD is not used, DF flag is set, and the size of the IPv4 packet is in between 1453 and 1500 bytes, the packet is dropped in the embedded routing instance(s).

The size of the `mtas_om` container emitted IP packets will never be bigger than 1452 bytes.

3.3 Virtual IP Access - Sig_SC VPN

3.3.1 Overview

The purpose of having this infrastructure in place is to secure connectivity to the MTAS services that are bound to Virtual IPs available through the Sig_SC VPN. These services are:

- Ut VIP
 - Services over the Ut interface
- SIGTRAN VIP 1 and SIGTRAN VIP 2
 - Services over the CAP interface
- Traffic VIP
 - All the other services (not Muta, Ro/Rf, CAI3G, Ut, and CAP)

The VMs that are playing the role of PL, but maximum 4 at any given moment, are connected to the “<MTAS VNF>_sig_sc_sp” virtual network through the `eth3` interface. Routing can either be dynamic or static on this infrastructure; dynamic is preferred. Mixed routing fashion is not supported in the IPv4/IPv6 dual-stack scenario. That is: either use dynamic for both IPv4 and IPv6, or use static for both of IPv4 and IPv6.

3.3.2 Routing Design

The routing can be designed in a similar fashion as in case of the OM_SC VPN, along with the Sig_SC VPN-related entities (VIP addresses, networks) as they are depicted in Figure 16:

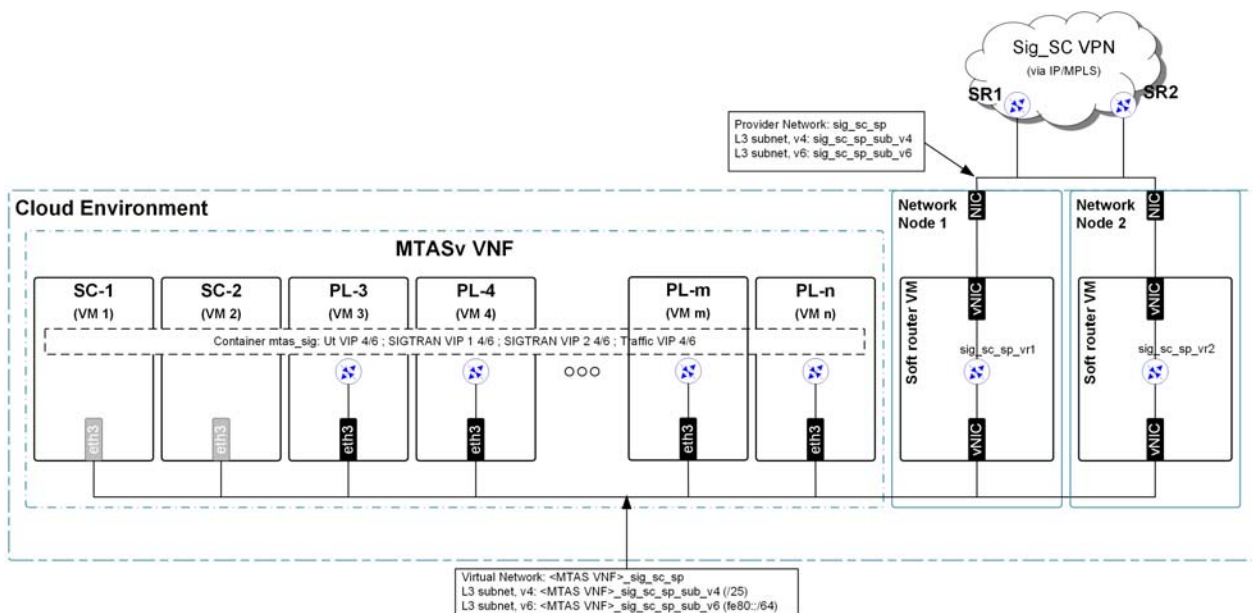


Figure 16 MTAS VNF Virtual IP Access - Sig_SC VPN

The dynamic/static VIP routing aspects are not detailed nor repeated in Figure 16, see Section 3.2.2 Dynamic Routing Design on page 16 and Section 3.2.3 Static Routing Design on page 17. The MTU aspects are also applicable here, see Section 3.2.4 MTU on page 19.

3.4 External Connectivity Through ASIC Routers

In specific environments, there be legacy ASIC routers available and the intention is, for performance and cost saving reasons, or both, to (re)use them for providing redundant L3 functions. This chapter describes on a high level how such switching/routing environment can be established.

3.4.1 Operation and Maintenance

This chapter describes how the operation and maintenance external connectivity looks like in such an L2/L3 eco-system whereby the redundant L3 functions are provided by ASIC routers.

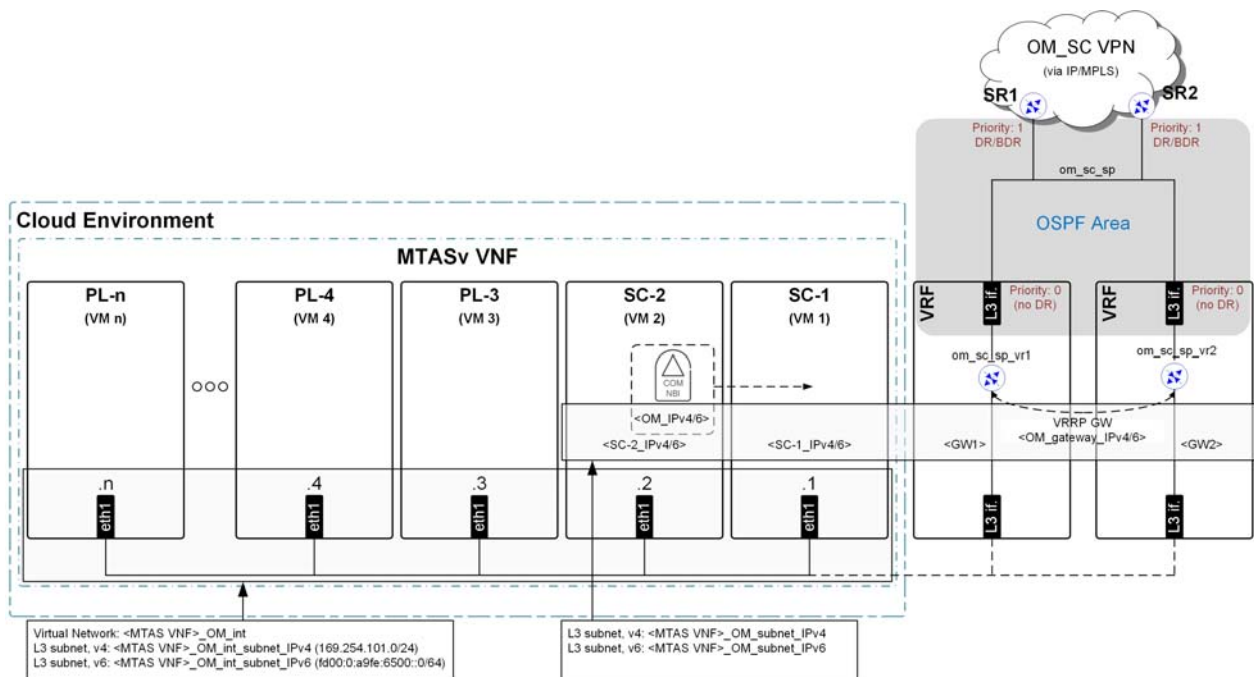


Figure 17 MTAS VNF System Management Infrastructure Through ASIC L3

The L3 functions are placed into Virtual Router Functions (VRFs) those are created/managed in physically separated, redundant ASIC routers. The segmentation ID and the VLAN ID of the "<MTAS VNF>_OM_int" infrastructure is agreed between the administrators of the Cloud Infrastructure and the ASIC routers.

The routing principles are the same as depicted in Section 3.1 on page 13. The only difference is: while in case of soft routers the "<MTAS VNF>_OM_int" infrastructure is shared by the MTAS- and the soft router VNFs, in the connected ASIC router scenario, the "<MTAS VNF>_OM_int" network remains private for the MTAS VNF. The VRFs are connected to the network segment (-> VLAN) in a "hidden" manner; those are not appearing as managed cloud resources for any of the cloud services (including Nova, Neutron, ...).

3.4.2 Virtual IP Access

On OM_SC VPN, this chapter describes how the Virtual IP external connectivity looks like in such an L2/L3 eco-system whereby the redundant L3 functions are provided by ASIC routers. The setup is the same in the Sig_SC VPN.

Figure 18 shows the MTAS VNF Virtual IP Access Through ASIC L3 - OM_SC VPN.

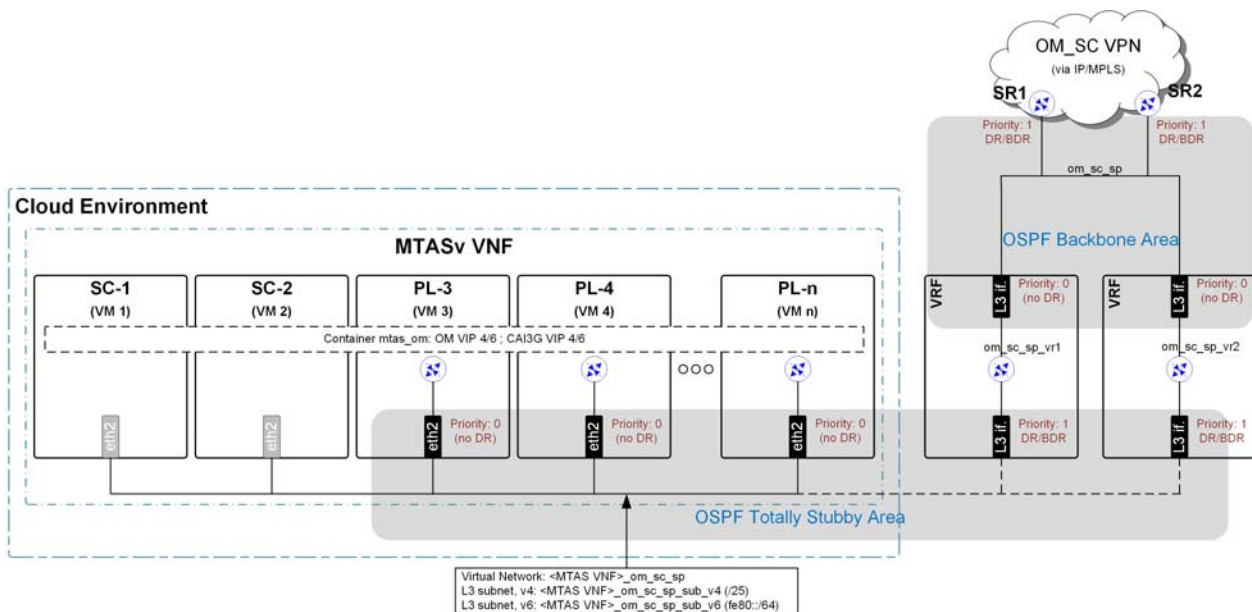


Figure 18 MTAS VNF Virtual IP Access Through ASIC L3 - OM_SC VPN

The L3 functions are placed into Virtual Router Functions (VRFs) those are created/managed in physically separated, redundant ASIC routers. The segmentation ID and the VLAN ID of the "<MTAS VNF>_om_sc_sp" infrastructure is agreed between the administrators of the Cloud Infrastructure and the ASIC routers.

The routing principles are the same as depicted in Section 3.2 on page 15 (and in Section 3.3 on page 19). The only difference is: while in case of soft routers the "<MTAS VNF>_om_sc_sp" infrastructure is shared by the MTAS- and the soft router VNFs, in the connected ASIC router scenario, the "<MTAS VNF>_om_sc_sp" network remains private for the MTAS VNF. The VRFs are connected to the network segment (-> VLAN) in a "hidden" manner; those are not appearing as managed cloud resources for any of the cloud services (including Nova, Neutron, and so on). The MTU aspects are also applicable here, see Section 3.2.4 MTU on page 19.

3.5 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. For that, follow the below steps.



3.5.1 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 (EvipNode(s) / priority):

- Dynamic routing: 3-6 / 5.000000 ; 7-12 / 10.000000
- Static routing: 3-4 / 5.000000 ; 5-6 / 6.000000 ; 7-12 / 10.000000

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

```
ManagedElement=1,Transport=1,Evip=1,EvipDeclarations=1,E
vipCluster=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>"
top
commit -s
```

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

3.5.2 Create LBEs and SEs

Load Balancer and Security Elements can not be created until floating EvipNodes exists. Create these entities by issuing the following commands

```
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=mt
as_om,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"
top
commit -s
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=m
tas_om,EvipSes=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"
top
commit -s
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=mta
s_sig,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"
top
commit -s
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=mt
as_sig,EvipSes=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"
top
commit -s
```

3.5.3 Define the VIP Addresses

```
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=mt
as_om,EvipVips=1
EvipVip=<OM VIP IPv4 Address>
```



```

deflt="no"
equivSrcAddr="no"
up
EvipVip=<OM VIP IPv6 Address>
deflt="no"
equivSrcAddr="no"
up
EvipVip=<CAI3G VIP IPv4 Address>
deflt="no"
equivSrcAddr="no"
up
EvipVip=<CAI3G VIP IPv6 Address>
deflt="no"
equivSrcAddr="no"
top
commit -s
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=mta
s_sig,EvipVips=1
EvipVip=<Traffic VIP IPv4 Address>
deflt="no"
equivSrcAddr="no"
up
EvipVip=<Traffic VIP IPv6 Address>
deflt="no"
equivSrcAddr="no"
up
EvipVip=<Ut VIP IPv4 Address>
deflt="no"
equivSrcAddr="no"
up
EvipVip=<Ut VIP IPv6 Address>
deflt="no"
equivSrcAddr="no"
up
EvipVip=<SIGTRAN1 VIP IPv4 Address>
deflt="no"
equivSrcAddr="no"
up
EvipVip=<SIGTRAN1 VIP IPv6 Address>
deflt="no"
equivSrcAddr="no"
up
EvipVip=<SIGTRAN2 VIP IPv4 Address>
deflt="no"
equivSrcAddr="no"
up
EvipVip=<SIGTRAN2 VIP IPv6 Address>
deflt="no"
equivSrcAddr="no"
top
commit -s

```



3.5.4 Create the FEE Elements

The FEE elements are carrying the MTAS VNF embedded routing instances (four FEEs in both VPNs -> ALBs), as they are referred to in the document. The following chapters are providing instructions how to define them, depending on the chosen VIP routing strategy.

3.5.4.1 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 Table 1.

Table 1 Summary of FEE elements within the MTAS VNF, dynamic routing

ALB Name	FEE ID	Interface ID	Evip Node ID	Local IPv4 Address	OSPFv2 Area ID	OSPFv2 Router ID	OSPFv3 Area ID	OSPFv3 Router ID
mtas_o_m	fee1	eth2	3	<for example 10.0.12.1/25>	<for example 0.1.1.1>	<for example 1.1.1.1>	<for example 0.1.1.1>	<for example 1.1.1.1>
mtas_o_m	fee2	eth2	4	<for example 10.0.12.2/25>	<for example 0.1.1.1>	<for example 1.1.1.2>	<for example 0.1.1.1>	<for example 1.1.1.2>
mtas_o_m	fee3	eth2	5	<for example 10.0.12.3/25>	<for example 0.1.1.1>	<for example 1.1.1.3>	<for example 0.1.1.1>	<for example 1.1.1.3>



ALB Name	FEE ID	Interface ID	Evip Node ID	Local IPv4 Address	OSPFv2 Area ID	OSPFv2 Router ID	OSPFv3 Area ID	OSPFv3 Router ID
mtas_o_m	fee_4	eth2	6	<for example 10.0.12.4/25>	<for example 0.1.1.1>	<for example 1.1.1.4>	<for example 0.1.1.1>	<for example 1.1.1.4>
mtas_sing	fee_1	eth3	3	<for example 10.1.12.1/25>	<for example 0.0.1.1>	<for example 1.1.1.1>	<for example 0.0.1.1>	<for example 1.1.1.1>
mtas_sing	fee_2	eth3	4	<for example 10.1.12.2/25>	<for example 0.0.1.1>	<for example 1.1.1.2>	<for example 0.0.1.1>	<for example 1.1.1.2>
mtas_sing	fee_3	eth3	5	<for example 10.1.12.3/25>	<for example 0.0.1.1>	<for example 1.1.1.3>	<for example 0.0.1.1>	<for example 1.1.1.3>
mtas_sing	fee_4	eth3	6	<for example 10.1.12.4/25>	<for example 0.0.1.1>	<for example 1.1.1.4>	<for example 0.0.1.1>	<for example 1.1.1.4>

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.

3.5.4.2

Static Routing

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
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 Table 2.

Table 2 Summary of FEE elements within the MTAS VNF, static routing

ALB Name	FEE ID	Interface ID	Evip Node ID	Local IPv4 Address	GW IPv4 Address	Local IPv6 Address	GW IPv6 Address
mtas_o_m	1	fee_eth2	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>
mtas_o_m	2	fee_eth2	4	<for example 10.0.12.2/25>	<for example 10.0.12.126>	<for example fd00:0:a00:c00::2/64>	<for example fd00:0:a00:c00::fffe/64>



ALB Name	FEE ID	Interface ID	Evip Node ID	Local IPv4 Address	GW IPv4 Address	Local IPv6 Address	GW IPv6 Address
mtas_om	3	fee_eth2	5	<for example 10.0.12.3/25>	<for example 10.0.12.125>	<for example fd00:0:a00:c00::3/64>	<for example fd00:0:a00:c00::fffd/64>
mtas_om	4	fee_eth2	6	<for example 10.0.12.4/25>	<for example 10.0.12.126>	<for example fd00:0:a00:c00::4/64>	<for example fd00:0:a00:c00::fffe/64>
mtas_sig	1	fee_eth3	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>
mtas_sig	2	fee_eth3	4	<for example 10.1.12.2/25>	<for example 10.1.12.126>	<for example fd00:0:a01:c00::2/64>	<for example fd00:0:a01:c00::fffe/64>
mtas_sig	3	fee_eth3	5	<for example 10.1.12.3/25>	<for example 10.1.12.125>	<for example fd00:0:a01:c00::3/64>	<for example fd00:0:a01:c00::fffd/64>
mtas_sig	4	fee_eth3	6	<for example 10.1.12.4/25>	<for example 10.1.12.126>	<for example fd00:0:a01:c00::4/64>	<for example fd00:0:a01:c00::fffe/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.

3.5.5 Create Flow Policies

After performing all of 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.

3.5.5.1 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>
```



```

dest=<VIP address>
addressFamily=<Address family>
protocol=<Protocol family>
destPort=<DST port>
targetPool=<Target pool>
top
commit -s

```

Create all the required flow policies, in-line with Table 3. Refer to *MTAS Hardening Guide* for further details regarding MTAS supported TCP/UDP SAPs.

Table 3 Summary of MTAS flow policies for incoming TCP/UDP streams, IPv4

ALB Name	FlowPolicy ID	Protocol Family	DST Port	Target Pool	Address Family	VIP Address
mtas_om	ProvisioningL DAP	tcp	17323	PLs_r r	ipv4	<OM VIP IPv4 Address>
mtas_om	ProvisioningL DAP_secure	tcp	17423	PLs_r r	ipv4	<OM VIP IPv4 Address>
mtas_om	RoRf_1	tcp	3868	PLs_r r	ipv4	<OM VIP IPv4 Address>
mtas_om	RoRf_2	tcp	3869	PLs_r r	ipv4	<OM VIP IPv4 Address>
mtas_om	RoRf_3	tcp	3870	PLs_r r	ipv4	<OM VIP IPv4 Address>
mtas_om	RoRf_4	tcp	3871	PLs_r r	ipv4	<OM VIP IPv4 Address>
mtas_om	RoRf_5	tcp	3872	PLs_r r	ipv4	<OM VIP IPv4 Address>
mtas_om	CAI3G	tcp	8095	PLs_r r	ipv4	<CAI3G VIP IPv4 Address>
mtas_om	CAI3G-HTTP S	tcp	8443	PLs_r r	ipv4	<CAI3G VIP IPv4 Address>
mtas_sig	XCAP	tcp	8090	PLs_r r	ipv4	<Ut VIP IPv4 Address>



ALB Name	FlowPolicy ID	Protocol Family	DST Port	Target Pool	Address Family	VIP Address
mtas_sig	ShDh_1	tcp	3868	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	ShDh_2	tcp	3869	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	ShDh_3	tcp	3870	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	ShDh_4	tcp	3871	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	ShDh_5	tcp	3872	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_01t	tcp	5060	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_01u	udp	5060	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_02t	tcp	5082	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_02u	udp	5082	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_03t	tcp	5083	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_03u	udp	5083	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_04t	tcp	5084	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_04u	udp	5084	PLs_r	ipv4	<Traffic VIP IPv4 Address>



ALB Name	FlowPolicy ID	Protocol Family	DST Port	Target Pool	Address Family	VIP Address
mtas_sig	SIP_05t	tcp	5085	PLs_rr	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_05u	udp	5085	PLs_rr	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_06t	tcp	5086	PLs_rr	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_06u	udp	5086	PLs_rr	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_07t	tcp	5087	PLs_rr	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_07u	udp	5087	PLs_rr	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_08t	tcp	5088	PLs_rr	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_08u	udp	5088	PLs_rr	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_09t	tcp	5089	PLs_rr	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_09u	udp	5089	PLs_rr	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_10t	tcp	5090	PLs_rr	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_10u	udp	5090	PLs_rr	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_11t	tcp	5094	PLs_rr	ipv4	<Traffic VIP IPv4 Address>



ALB Name	FlowPolicy ID	Protocol Family	DST Port	Target Pool	Address Family	VIP Address
mtas_sig	SIP_11u	udp	5094	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_12t	tcp	5160	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_12u	udp	5160	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_13t	tcp	5161	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_13u	udp	5161	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_14t	tcp	5162	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_14u	udp	5162	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_15t	tcp	5163	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SIP_15u	udp	5163	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	soap_http	tcp	9080	PLs_r	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	CCMP	tcp	8096	PLs_r	ipv4	<Traffic VIP IPv4 Address>

Table 4 Summary of MTAS flow policies for incoming TCP/UDP streams, IPv6

ALB Name	FlowPolicy ID	Protocol Family	DST Port	Target Pool	Address Family	VIP Address
mtas_om	6Provisioning LDAP	tcp	17323	PLs_r	ipv6	<OM VIP IPv6 Address>



ALB Name	FlowPolicy ID	Protocol Family	DST Port	Target Pool	Address Family	VIP Address
mtas_om	6Provisioning LDAP_secure	tcp	17423	PLs_r	ipv6	<OM VIP IPv6 Address>
mtas_om	6RoRf_1	tcp	3868	PLs_r	ipv6	<OM VIP IPv6 Address>
mtas_om	6RoRf_2	tcp	3869	PLs_r	ipv6	<OM VIP IPv6 Address>
mtas_om	6RoRf_3	tcp	3870	PLs_r	ipv6	<OM VIP IPv6 Address>
mtas_om	6RoRf_4	tcp	3871	PLs_r	ipv6	<OM VIP IPv6 Address>
mtas_sig	6ShDh_5	tcp	3872	PLS_r	ipv6	<Traffic VIP IPv6 Address>
mtas_om	6RoRf_5	tcp	3872	PLs_r	ipv6	<OM VIP IPv6 Address>
mtas_om	6CAI3G	tcp	8095	PLs_r	ipv6	<CAI3G VIP IPv6 Address>
mtas_om	6CAI3G-HTTPS	tcp	8443	PLs_r	ipv6	<CAI3G VIP IPv6 Address>
mtas_sig	6XCAP	tcp	8090	PLs_r	ipv6	<Ut VIP IPv6 Address>
mtas_sig	6ShDh_1	tcp	3868	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6ShDh_2	tcp	3869	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6ShDh_3	tcp	3870	PLs_r	ipv6	<Traffic VIP IPv6 Address>



ALB Name	FlowPolicy ID	Protocol Family	DST Port	Target Pool	Address Family	VIP Address
mtas_sig	6ShDh_4	tcp	3871	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6ShDh_5	tcp	3872	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_01t	tcp	5060	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_01u	udp	5060	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_02t	tcp	5082	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_02u	udp	5082	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_03t	tcp	5083	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_03u	udp	5083	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_04t	tcp	5084	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_04u	udp	5084	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_05t	tcp	5085	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_05u	udp	5085	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_06t	tcp	5086	PLs_r	ipv6	<Traffic VIP IPv6 Address>



ALB Name	FlowPolicy ID	Protocol Family	DST Port	Target Pool	Address Family	VIP Address
mtas_sig	6SIP_06u	udp	5086	PLs_rr	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_07t	tcp	5087	PLs_rr	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_07u	udp	5087	PLs_rr	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_08t	tcp	5088	PLs_rr	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_08u	udp	5088	PLs_rr	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_09t	tcp	5089	PLs_rr	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_09u	udp	5089	PLs_rr	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_10t	tcp	5090	PLs_rr	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_10u	udp	5090	PLs_rr	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_11t	tcp	5094	PLs_rr	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_11u	udp	5094	PLs_rr	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_12t	tcp	5160	PLs_rr	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_12u	udp	5160	PLs_rr	ipv6	<Traffic VIP IPv6 Address>



ALB Name	FlowPolicy ID	Protocol Family	DST Port	Target Pool	Address Family	VIP Address
mtas_sig	6SIP_13t	tcp	5161	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_13u	udp	5161	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_14t	tcp	5162	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_14u	udp	5162	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_15t	tcp	5163	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SIP_15u	udp	5163	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6soap_http	tcp	9080	PLs_r	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6CCMP	tcp	8096	PLs_r	ipv6	<Traffic VIP IPv6 Address>

Note: Opening ports of unused services must be avoided.

3.5.5.2

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>
dest=<VIP address>
addressFamily=<Address family>
protocol=sctp
soGrp=1011250
top
commit -s
```

Create all the required flow policies in-line with Table 5. Refer to *MTAS Hardening Guide* for further details regarding MTAS supported SCTP SAPs.



Table 5 Summary of MTAS flow policies for incoming SCTP streams, IPv4

ALB Name	FlowPolicy ID	Address Family	VIP Address
mtas_sig	SCTP_H248	ipv4	<Traffic VIP IPv4 Address>
mtas_sig	SCTP_SIGTRAN_1	ipv4	<SIGTRAN1 VIP IPv4 Address>
mtas_sig	SCTP_SIGTRAN_2	ipv4	<SIGTRAN2 VIP IPv4 Address>

Table 6 Summary of MTAS flow policies for incoming SCTP streams, IPv6

ALB Name	FlowPolicy ID	Address Family	VIP Address
mtas_sig	6SCTP_H248	ipv6	<Traffic VIP IPv6 Address>
mtas_sig	6SCTP_SIGTRAN_1	ipv6	<SIGTRAN1 VIP IPv6 Address>
mtas_sig	6SCTP_SIGTRAN_2	ipv6	<SIGTRAN2 VIP IPv6 Address>

Note:

- Opening ports of unused services must be avoided.
- SCTP transport is supported in one container, in the `mtas_sig` ALB only. This can be improved in a later MTAS release.