Converged Application Server

Intelligent Media Connector

Release 8.3
G50166-01
February 2026

ORACLE

Converged Application Server Intelligent Media Connector, Release 8.3

G50166-01

Contents

About This Guide

1 Intelligent Media Connector

Transcoding Support 2

2 Media Flow Engine

Install the Media Gateway Service Domain 2
Install Patches 3
Start Your Server 3
Install the Media Flow Engine 3
Set Up MFE Connections 5
- Set Up the MFE Connection with the Remote Console 5
- Set Up MFE Connection with REST API 5
- Set Up MFE Connection with WLST Commands 5

3 Install the RTP Proxy

KVM Configuration 1
Install RTP Proxy 6
Configuring the RTP Proxy 8

4 Configure JWT Authentication

Configure TLS 2

5 Set System Properties

6 Configure Overload Protection

7 Deploy the Sample UA Application

About This Guide

Table 1 Revision History

Date	Revision
March 2026	• Initial release

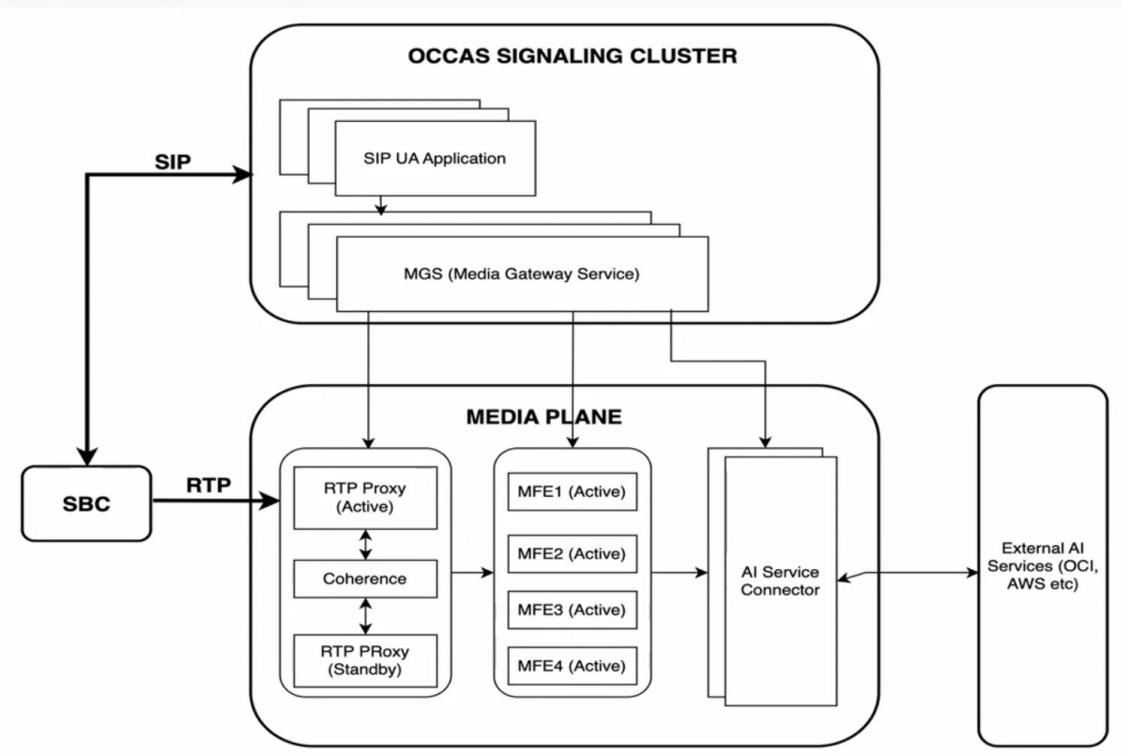
1

Intelligent Media Connector

The Intelligent Media Connector allows you to manage media sessions from the Converged Application Server application. The Intelligent Media Connector enables integration between SIP-based telephony environments, speech processing solutions and GenAI services. It facilitates communication between telephony infrastructure (such as Session Border Controllers and SIP trunks) and components like Automatic Speech Recognition (ASR), Speech-to-Text (STT), and Text-to-Speech (TTS). The Intelligent Media Connector provides real-time call media handling and codec transcoding to support AI-powered voice services such as personal assistant for consumers and employees, AI agents for call handling, call overflow and unlimited availability, and enterprise productivity tools like translation, summaries, sentiment analysis, compliance and security.

Enhancing Converged Application Server with media processing allows you to build innovate applications where call services and call control are influenced by AI. You can support complex call scenarios where you chain several services such as caller attestation, voice biometrics, call queue and call transfer with automated self-service and automated calling solutions. You can build personal assistants to acts on behalf of the consumer such as screen calls, transcribe voicemails and determine call-back or appointment scheduling, manage "Do Not Disturb," and much more.

The Converged Application Server architecture separates call signaling from audio-media stream processing, enabling seamless scaling, failover, and extensibility to AI-powered use cases (such as live transcription, call summary, and sentiment analysis).



Transcoding Support

Transcoding support addresses the interoperability challenges encountered when telephony equipment, speech processing services, and AI agents use different audio codecs. By providing real-time, configurable conversion between common telephony formats and the raw audio formats required by AI solutions, the Intelligent Media Connector ensures robust, end-to-end speech processing integration in enterprise environments.

The Intelligent Media Connector supports transcoding between the following codecs and formats:

- G.711 (PCMA/PCMU) to PCM-16
- AMR-NB to PCM-16
- OPUS to PCM-16
- OPUS to OPUS (pass-through)
- G.711 to OPUS
- AMR-NB to OPUS

In supported scenarios, OPUS can be passed through without transcoding if downstream services accept it.

Media transcoded to OPUS is encapsulated in an Ogg container.

2 Media Flow Engine

To install the Intelligent Media Connector, first install the Media Flow Engine.

1. Verify your system meets the minimum requirements.

Table 2-1 Hardware Requirements

Resource	Requirement
Memory	8GB (16GB recommended)
Storage	150GB (300 GB recommended)
CPU	8 (16 recommended)
Network	1Gpbs Ethernet

Table 2-2 Software Requirements

Resource	Requirement
Operating System	Oracle Linux 9.6 or later
Java for Converged Application Server	JDK 17 or JDK 21
Java for Media Flow Engine (MFE)	JDK 25
Java for sample AI connector	JDK 25

Table 2-3 Network Requirements

Component	Requirement	Purpose
Converged Application Server Node	1 NIC (3 recommended)	With 3 NICs, use a dedicated NIC for each of the following: • control traffic • SIP Signalling • Coherence communication
MFE Node	1 NIC (2 recommended)	With 2 NICs, use a dedicate NIC for each of the following: • control traffic • RTP media traffic
RTP Proxy Node	2 NICs (3 recommended)	With 3 NICs, use a dedicated NIC for each of the following: • control traffic • inbound media • outbound media

2. Install the Media Gateway Service Domain.
3. Install Patches.
4. Start Your Server.
5. Install the Media Flow Engine.
6. Set Up the MFE Connection with the Remote Console.

7. Deploy the Sample UA Application.

Install the Media Gateway Service Domain

Before installing the Intelligent Media Connector, first install the Converged Application Server.

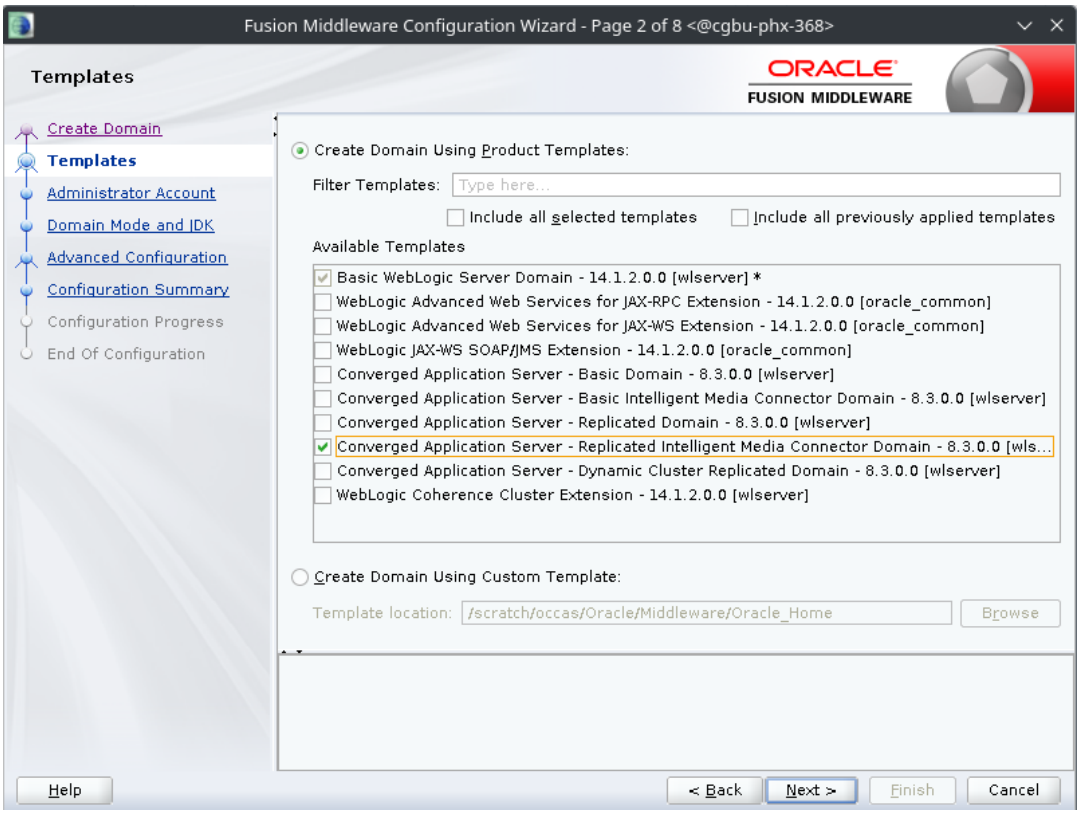
The hardware requirements:

- 8GB Memory (16GB recommended)
- 150GB Storage
- 8 CPUs (16 recommended)
- 1Gpbs Ethernet connection

1. Dowload the occas_generic.jar file.
2. Start the installation.

```
java -jar occas_generic.jar
```

   Follow the instructions in the *Installation Guide* to install the Converged Application Server.
3. Start the Configuration Wizard.
4. On the Create Domain page, select "Create a new domain" and click **Next**.
5. On the Templates page, select "Converged Application Server - Replicated Intelligent Media Connector Domain" and click **Next**.



6. On the Administrator Account page, set the passphrase for your Converged Application Server.
7. On the Domain Mode and JDK page, select the appropriate startup mode.

   If you are deploying the Converged Application Server in a production environment, select Production; otherwise, select Development.
   In the **Enable or Disable Default Ports for Your Domains**, select whether or not your instance listens on SSL ports. By default, Development mode listens on non-SSL ports and Production listens on SSL ports.
8. On the Advanced Configuration page, select **Next**.
9. On the Configuration Summary page, select **Create**.
10. After the domain has been installed, click **Finish** to close the Configuration Wizard.

Install Patches

Before starting your Converged Application Server, you must apply the latest WebLogic Server patches. See the "Patching Converged Application Server" chapter in the *Installation Guide* for instructions.

Start Your Server

After the latest patches are installed, start the Converged Application Server. First set your `DOMAIN_HOME` variable.

```
export DOMAIN_HOME=<path to your domain home>
```

- To start the Admin Server:

  ```
  sh $DOMAIN_HOME/bin/startWebLogic.sh
  ```

- To start the Managed Servers:

  ```
  sh $DOMAIN_HOME/bin/startManagedWebLogic.sh engine1 http://<Admin
  IP>:<Admin Port>
  ```

See "Configure JWT Authentication" for details on securing communications between the Converged Application Server and infrastructure components like Media Flow Engines and RTP Proxies.

To verify your servers are running, from the Remote Console select **Monitoring Tree**, and then **Environment**, and then **Servers**.

Install the Media Flow Engine

Install the Media Flow Engine on a separate machine from your Converged Application Server.

The IP address must be reachable by both the RTP Proxy and the Converged Application Server.

> **Note**
>
> Oracle recommends using two separate IP addresses: one for control traffic from the Converged Application Server, the other for receiving RTP packets from the RTP Proxy.

1. Download the Media Flow Engine installer tarball.
2. Set the JAVA_HOME variable to the path of your JDK 25. Then update your PATH variable to include the bin directory of your new JAVA_HOME.

   ```
   export JAVA_HOME=/usr/lib/jvm/jdk-25.0.2-oracle-x64
   export PATH="$JAVA_HOME/bin:$PATH"
   ```

3. Extract the Media Flow Engine file.

   ```
   tar xf media-flow-engine-8.3.0.0.0.tar.gz
   ```

4. Navigate to the `media-flow-engine-8.3.0.0.0-MFE/MFE_HOME/config` directory and update the two IP address parameters in the `application.yaml` file to match the IP address of the server where you will deploy the MFE.
   - application.ip-address – The IP address which Converged Application Server uses to communicate with the Media Flow Engine over HTTP or HTTPS.
     When configuring **Media Flow Engines** in the Remote Console, use this IP address for the **Management IP** field.
   - Gst.Ip-address – The internal IP address used for receiving RTP packets from the RTP Proxy.

   ```
   application:
     media-session:
       base-path: /media/v1/session
     ip-address: "127.0.0.1"
     custom-metrics:
       endpoint-path: /metrics

   Gst:
    Ip-address: "127.0.0.1"
   ```

5. Navigate to the `media-flow-engine-8.3.0.0.0-MFE/MFE_HOME/bin` directory.

   ```
   cd ../bin
   ```

6. Start the MFE.

   ```
   sh startMFE.sh
   ```

To verify the MFE has started, open a browser and navigate to http://<IP address>:9090/metrics. The Converged Application Server uses port 9090 for health checks.

Set Up MFE Connections

A typical deployment includes multiple MFEs. After starting your MFEs, you need to connect them to the Converged Application Server.

To connect an MFE, you can use the Remote Console, the REST APIs, or WLST.

Set Up the MFE Connection with the Remote Console

1. From the Edit Tree in the Remote Console, navigate to **Custom Resources**, and then **IntelligentMediaConnector**, and then **Media Connector Configurations**, and then **Media Flow Engines**.
2. Click **New**.
3. Enter a **Server Name**.
4. Enter a **Management IP**.
5. Set a port for the **Management TLS Port**.
6. Set a port for **Management Non-TLS Port**.
7. If you are using TLS, enable the **SSL/TLS Enabled** slider.
8. Click **Create**.

Set Up MFE Connection with REST API

- Use the following command, with appropriate modifications, to create the MFE connection for Converged Application Server.

```
curl -v -X POST \
    --user <username>:<password> \
    -H X-Requested-By:MyClient \
    -H Accept:application/json \
    -H Content-Type:application/json \
    -d '{
  "serverName": "<mfe_name>",
  "managementIP": "<mfe_ip>",
  "managementPort": <management_port>,
  "secondaryPort": <secondary_port>,
  "sslEnabled": true
}' \
    "http://<adminserver_host>:<port>/management/weblogic/latest/edit/
customResources/IntelligentMediaConnector/customResource/mediaFlowEngines/"
```

Do not put quotation marks around the value of the `sslEnabled` parameter.

Set Up MFE Connection with WLST Commands

1. Log in to the Admin Node and navigate to the <Domain_Home>/bin directory.

2. Load the domain environmental variables.

```
. ./setDomainEnv.sh
```

3. Save the following script to your system and modify the parameters to be appropriate to your environment.

```
try:
    connect("<username>", "<password>", "t3://<adminserver_ip>:<port>")
    edit()
    startEdit()
    cd('/CustomResources/IntelligentMediaConnector')
    ms = cmo.getResource()
    name = '<name_of_mfe>'
    mfe = ms.lookupMediaFlowEngine(name)
    if mfe is None:
        print 'Creating MediaFlowEngine:', name
        mfe = ms.createMediaFlowEngine(name)
        mfe.setManagementIP('<mfe_ip>')
        mfe.setManagementPort(<management_port>)
        mfe.setSecondaryPort(<secondary_port>)
        mfe.setSslEnabled(<true_or_false>)
    else:
        print 'MediaFlowEngine already exists:', name
    save()
    activate(block="true")
    print 'res:1'
except Exception, e:
    print e
    dumpStack()
    print 'res:0'
print 'End of script...'
disconnect('true')
exit('true')
```

4. Run the WLST script.

```
java weblogic.WLST /path/to/file.py
```

3

Install the RTP Proxy

After installing the Media Flow Engine, next install the RTP Proxy.

Before installing the RTP Proxy, verify your system matches the requirements.

Hypervisor	KVM/QEMU
Host OS	Oracle Linux 9 (UEK kernel recommended)
Guest OS	Oracle Linux 9 (x86_64)
Host NICs	Intel (E810/X710/XL710) or Mellanox/NVIDIA (ConnectX-5/6) with SR-IOV
Guest NICs	1 (virtio)
Access	Root access
vCPUs per VM	12 (16 recommended)
RAM per VM	16 - 48GB
Storage per VM	40GB
SR-IOV VFs per VM	2
Java	JDK 25

- NTP or chrony must be enabled on both the hosts and guests VMs.
- There must not be any layer 3 device between the active and standby RTP proxy nodes; L2 adjacency is required for VIP failover and gratuitous ARP propagation.

In failover scenarios, the standby node takes over the virtual IP and sends GARP (Gratuitous ARP) announcements. These GARP messages update the MAC/IP association of the VIP on the network, ensuring continuity of RTP stream delivery. Switches and connected infrastructure must not block or filter GARP messages, and should support fast MAC address table updates. Port security mechanisms (such as sticky MAC or dynamic ARP inspection) must allow legitimate MAC moves and GARP updates to prevent disruption during failover.

Receiver Side Scaling (RSS) must be enabled on every NIC to ensure performance.

KVM Configuration

Complete the following steps to configure KVM for your RTP Proxies.

BIOS/UEFI Settings

Depending on your hardware, some of these settings may not be available.

1. In the system's BIOS, set the CPU profile to "Performance".
2. Disable C3/C6 power states.
3. Enable Turbo.

Tune the Kernel

The values for each setting must be customized for your CPU.

1. Disable IRQ balance.

```
sudo systemctl disable --now irqbalance
```

2. Add the following CPU parameters to grub.

Table 3-1 Kernel Arguments

Parameter	Purpose
isolcpus	Marks these CPUs as isolated from the scheduler's general load balancing
nohz_full	Puts those CPUs into full tickless mode to minimize scheduler timer interrupts
rcu_nocbs	Offloads RCU callbacks from those CPUs to housekeeping CPUs
intel_iommu	Enables Intel VT-d IOMMU for DMA isolation and for vfio-pci use
iommu	Enables pass-through mode for IOMMU on non-vfio devices to reduce overhead
tuned.non_isolcpus	Declares which CPUs are housekeeping (NOT isolated)
intel_pstate	Enables or disables the Intel P-state driver

The following command isolates Intel CPUs 2 through 47 and 50 through 97.

```
sudo grubby --update-kernel=ALL \
  --args="isolcpus=2-47,50-95 nohz_full=2-47,50-95 rcu_nocbs=2-47,50-95
intel_iommu=on iommu=pt tuned.non_isolcpus=00030000,00000003
intel_pstate=disable"
```

The following command isolates AMD CPUs 2 through 47 and 50 through 97

```
sudo grubby --update-kernel=ALL \
  --args="isolcpus=2-47,50-95 nohz_full=2-47,50-95 rcu_nocbs=2-47,50-95
amd_iommu=on iommu=pt tuned.non_isolcpus=00030000,00000003"
```

3. Once the appropriate kernel parameters have been set for your hardware, reboot the system.
Use this command to verify your kernel arguments:

```
cat /proc/cmdline
```

Install KVM

1. Verify your CPU supports virtualization.

```
egrep -i 'vmx|svm' /proc/cpuinfo
```

2. Install virtualization software.

```
sudo dnf groupinstall "Virtualization Host"
```

3. Start the libvirtd service.

```
sudo systemctl enable --now libvirtd
```

Manage Virtual Networks

1. Block the kernel from automatically loading the i40evf and iavf drivers.

```
echo "blacklist i40evf" | sudo tee -a /etc/modprobe.d/blacklist.conf
echo "blacklist iavf" | sudo tee -a /etc/modprobe.d/blacklist.conf
```

   This allows the build-in driver mlx5_core to be used instead.

2. Customize the following script as appropriate to your network interfaces and requirements.

```
sudo bash -c 'cat > /etc/rc.d/rc.local << "EOF"
#!/usr/bin/env bash
# Create VFs
echo 2 > /sys/bus/pci/devices/0000:af:00.0/sriov_numvfs
echo 2 > /sys/bus/pci/devices/0000:af:00.1/sriov_numvfs
# Assign MACs
ip link set enp175s0f0 vf 0 mac ee:07:94:7a:a7:81
ip link set enp175s0f0 vf 1 mac ee:07:94:7a:a7:82
ip link set enp175s0f1 vf 0 mac 5a:b0:f8:ce:04:51
ip link set enp175s0f1 vf 1 mac 5a:b0:f8:ce:04:52
# Set trust and link state
ip link set enp175s0f0 vf 0 trust on
ip link set enp175s0f0 vf 1 trust on
ip link set enp175s0f1 vf 0 trust on
ip link set enp175s0f1 vf 1 trust on
ip link set enp175s0f0 vf 0 state auto
ip link set enp175s0f0 vf 1 state auto
ip link set enp175s0f1 vf 0 state auto
ip link set enp175s0f1 vf 1 state auto
# Optional: Disable spoof check for multi-mac
ip link set enp175s0f0 vf 0 spoof off
ip link set enp175s0f0 vf 1 spoof off
ip link set enp175s0f1 vf 0 spoof off
ip link set enp175s0f1 vf 1 spoof off
EOF
chmod +x /etc/rc.d/rc.local'
```

3. Reboot to apply the changes.

Edit the Guest's Domain XML

The edits shown below are samples only. Do not reuse them without careful evaluation.

1. Use the command `lspci -D` to discover the Bus:Device:Function (BDF) values for your server.
2. Attach the VFs with VFIO.

Edit the actual BDF values to match those of your server.

```
<hostdev mode='subsystem' type='pci' managed='yes'>
  <driver name='vfio'/>
  <source>
    <address domain='0x0000' bus='0xaf' slot='0x02' function='0x0'/>
  </source>
</hostdev>
<hostdev mode='subsystem' type='pci' managed='yes'>
  <driver name='vfio'/>
  <source>
    <address domain='0x0000' bus='0xaf' slot='0x06' function='0x0'/>
  </source>
</hostdev>
```

3. Add a management interface.

```
<interface type='bridge'>
  <mac address='52:54:00:22:9a:59'/>
  <source bridge='br-mgmt0'/>
  <model type='virtio'/>
</interface>
```

4. Define and pin the virtual CPUs to isolated NUMA-local CPUs of the host.

```
<vcpu placement='static'>16</vcpu>
<cputune>
  <vcpupin vcpu='0' cpuset='2'/>
  <vcpupin vcpu='1' cpuset='3'/>
  ... (Repeat for all vCPUs) ...
  <emulatorpin cpuset='2-17'/>
</cputune>
```

5. Enable NUMA memory pinning.

```
<numatune>
  <memory mode='strict' nodeset='0'/>
</numatune>
```

6. Enable hugepages-backed memory.

```
<memoryBacking>
  <hugepages/>
</memoryBacking>
```

Start the Virtual Machine

1. Define the virtual machine.

```
virsh define vm.xml
```

2. Start the virtual machine.

```
virsh start <name>
```

3. On the guest VM, enable console access from the KVM host.

```
sudo systemctl enable --now serial-getty@ttyS0
```

4. From the KVM host, connect to the virtual machine.

```
virsh console <name>
```

Prepare the Guest OS

These steps should be taken after you have connected to the guest VM.

1. Enable IOMMU.

```
sudo grubby --update-kernel=ALL --args="intel_iommu=on iommu=pt"
sudo reboot
```

2. Select the appropriate driver for your CPU.

Table 3-2 List of Supported NIC and Their Drivers

Vendor	Model/ Series	PF Driver	VF Driver	PF Usage	VF Usage	Notes
Intel	E810 (800 series)	ice	iavf	Keep on kernel (ice).	Recommended: vfio-pci	Enable IOMMU for vfio-pci. Bind VFs to vfio-pci for DPDK iavf PMD.
Intel	X710/XL710 (700 series)	i40e	iavf (i40evf deprecated)	Keep on kernel (i40e).	Recommended: vfio-pci	Avoid i40evf (deprecated). Use iavf + vfio-pci for VFs.
NVIDIA/ Mellanox	ConnectX-5 / ConnectX-6	mlx5_core (eth: mlx5e)	mlx5_core (eth: mlx5e)	Do NOT use vfio-pci (keep on kernel)	Do NOT use vfio-pci (keep on kernel)	mlx5 PMD works with kernel mlx5_core + rdma-core/ DevX. Binding to vfio-pci breaks mlx5 PMD.

3. Set up hugepages.
The example below creates a 2GB hugepage that persists across reboots.

```
echo 2048 | sudo tee /proc/sys/vm/nr_hugepages
echo "vm.nr_hugepages = 2048" | sudo tee /etc/sysctl.d/99-hugepages.conf
sudo sysctl --system

sudo mkdir -p /dev/hugepages
echo "nodev /dev/hugepages hugetlbfs defaults 0 0" | sudo tee -a /etc/fstab
sudo mount -a
```

4. Apply the network-latency tuned profile.

```
sudo dnf install -y tuned-profiles-cpu-partitioning
sudo tuned-adm profile network-latency
```

5. Disable irqbalance.

```
sudo systemctl disable --now irqbalance
```

6. Install the following dependencies.

```
sudo dnf install libibverbs rdma-core librdmacm pciutils
```

7. Install the JDK 25.

```
sudo dnf install jdk-25-headless
```

Install RTP Proxy

Install the RTP proxy.

Active Node

Follow these steps to install the RTP Proxy on the primary node.

1. Download and extract the rtp-proxy-8.3.0.0.0.tar.gz file.

```
tar xvf rtp-proxy-8.3.0.0.0.tar.gz
```

   This creates a directory called `ORACLE_HOME`.

2. Copy the contents to the `ORACLE_HOME` directory to your actual Oracle home directory.

```
cp -r ORACLE_HOME/* /scratch/occas/Oracle/Middleware/Oracle_Home/
```

3. Set the RP_CONFIG_PATH and RP_LOG_PATH variables.

```
export RP_CONFIG_PATH=/scratch/occas/Oracle/Middleware/Oracle_Home/config
export RP_LOG_PATH=/scratch/occas/Oracle/Middleware/Oracle_Home/log
```

4. Create the config.json file, the log4j2.xml file, and the tangosol-coherence-override.xml file.

```
cd $RP_CONFIG_PATH
mv config.json.sample config.json
mv log4j2.xml.sample log4j2.xml
mv tangosol-coherence-override.xml.sample tangosol-coherence-override.xml
```

5. Modify the `config.json` file to match the details of your environment. Pay attention to the following parameters:
   - Controller.main.server – The IP address and port where you are running the RTP Proxy.
   - Controller.metrics.server – The IP address and port that will serve up metrics.
   - Controller.vips – The virtual IP addresses used by the RTP Proxy

- Controller.tls.enabled – Whether TLS is enabled (true) or disabled (false)
- Datapath.spec.portConfig.ports – Configure port information
- Datapath.spec.domains – Configure domains
- Datapath.spec.vipEndpoints – Set the IP address and port for the virtual IP endpoints

6. Add the active and standby RTP Proxy IP addresses to the `tangosol-coherence-override.xml` file.

```
<member-name system-property="coherence.member">RtpProxy1</member-name>
  <well-known-addresses>
    <address id="1"><ACTIVE_RTP_PROXY_IP></address>
    <address id="2"><STANDBY_RTP_PROXY_IP></address>
    <!-- Add as many WKA addresses as needed for your cluster -->
  </well-known-addresses>
<address system-property="coherence.localhost"><ACTIVE_RTP_PROXY_IP></
address>
```

   If your active node has the IP address of 10.1.1.1 and your standby the IP address of 10.2.2.2, your `tangosol-coherence-override.xml` file would contain the following:

```
<member-name system-property="coherence.member">RtpProxy1</member-name>
  <well-known-addresses>
    <address id="1">10.1.1.1</address>
    <address id="2">10.2.2.2</address>
    <!-- Add as many WKA addresses as needed for your cluster -->
  </well-known-addresses>
<address system-property="coherence.localhost">10.1.1.1</address>
```

7. Install the RTP proxy service.

```
cd $ORACLE_HOME/systemd
./install-systemd.sh
```

8. Start the service.

```
sudo systemctl enable --now rtp-proxy-datapath
sudo systemctl enable --now rtp-proxy-controller
```

See the `README.md` file for comprehensive details.

Standby Node

When installing the RTP proxy service on the standby node, follow the steps for the primary node with these differences:

1. In the `config.json` file, set the IP address of the following parameters to the standby IP address:
   - Controller.main.server.host
   - Controller.metrics.server.host
2. In the `tangosol-coherence-override.xml` file, flip the <address> nodes so that the secondary node uses the ID attribute of 1 and the primary of 2. For example, if your active

node has the IP address of 10.1.1.1 and your standby the IP address of 10.2.2.2, your `tangosol-coherence-override.xml` file would contain the following:

```
<member-name system-property="coherence.member">RtpProxy1</member-name>
  <well-known-addresses>
    <address id="1">10.2.2.2</address>
    <address id="2">10.1.1.1</address>
    <!-- Add as many WKA addresses as needed for your cluster -->
  </well-known-addresses>
<address system-property="coherence.localhost">10.2.2.2</address>
```

Configuring the RTP Proxy

After installing the RTP Proxy, follow one of these procedures to identify the RTP Proxy to your Converged Application Server.

Remote Console

You may use the Remote Console to identify the RTP Proxy to your Converged Application Server instance.

1. From the Edit Tree, select **Custom Resources**, and then **IntelligentMediaConnector**, and then **Media Connector Configurations**, and then **RTP Proxies**.
2. Click **New**.
3. Enter the following required fields:
   - **Server Name** – Enter a name for this proxy, such as RTP-1.
   - **Management IP Address** – Enter the management IP address for this proxy.
     This IP address is the same identified in the `Controller.main.server.host` parameter of the `config.yaml` file.
   - **Management Port** – Enter the port that the RTP Proxy will use for HTTPS management traffic.
     This value must match the value of `Controller.main.server.httpsPort` in the `config.yaml` file.
   - **Secondary Port (Non-TLS)** – Enter the port that the RTP Proxy will use for HTTP management traffic.
     This value must match the value of `Controller.main.server.port` in the `config.yaml` file.
4. To enable TLS, move the **SSL/TLS Enabled** slider to the right.

> **Note**
>
> If you enable TLS, you must create a keystore and update the `JAVA_OPTIONS` variable to point to your keystore.
>
> ```
> JAVA_OPTIONS="${SAVE_JAVA_OPTIONS} \
>   -Djavax.net.ssl.trustStore=${ORACLE_HOME}/client-truststore.jks \
>   -Djavax.net.ssl.trustStorePassword=changeit \
>   -Djavax.net.ssl.trustStoreType=JKS"
> ```
>
> See the keytool documentation for instructions on creating a keystore.

5. Select **Create**.
6. Select the shopping cart and then **Commit Changes**.

REST APIs

You may use the REST API to identify the RTP Proxy to your Converged Application Server instance.

Send a POST request to the endpoint `/management/weblogic/latest/edit/customResources/IntelligentMediaConnector/customResource/mediaFlowEngines/`.

```
curl -X POST \
  --user <ADMIN_USERNAME>:<ADMIN_PASSWORD> \
  -H 'X-Requested-By:MyClient' \
  -H 'Accept:application/json' \
  -H 'Content-Type:application/json' \
  -d @rtp.json \
  "http://10.0.0.1:7001/management/weblogic/latest/edit/customResources/
IntelligentMediaConnector/customResource/RTPProxys/"
```

The contents of the `rtp.json` file sent in the request must contain the parameters shown below. Modify the values to match with your deployment. Note that 8443 is the default TLS port for the RTP Proxy and 8080 is the default non-TLS port for the RTP Proxy.

```
{
  "managementIP": "10.0.0.7",
  "managementPort": 8443,
  "rtpProxyName": "example.net",
  "sslEnabled": false,
  "secondaryPort": 8080
}
```

4

Configure JWT Authentication

You can optionally configure JWT-based authentication between the Intelligent Media Connector and the Media Flow Engines and RTP Proxies.

JWT-based authentication must be enabled globally within the environment. Either it is not enabled at all or it is enabled:

- between the Intelligent Media Connector and every MFE
- between the Intelligent Media Connector and every RTP Proxy

Mixed modes – with authentication enabled between some nodes and not others – is not supported.

Enable JWT Authentication for the MFE

1. Connect to the VM where you installed the MFE.
2. Navigate to the MFE's `config` directory.

   ```
   cd media-flow-engine-8.3.0.0.0-MFE/MFE_HOME/config/
   ```

3. In the `application.yaml` file, set the `security.jwt.enabled` property to true.

   ```
   security:
     jwt:
       enabled: true
       public-key:
         path: ../security/public.pem
   ```

Enable JWT Authentication for the RTP Proxy

4. Connect to the VM where you installed the RTP Proxy.
5. Navigate to the RTP Proxy's `config` directory.

   ```
   cd ORACLE_HOME/config/
   ```

6. Set the `jwt.enabled` parameter to true.

   ```
       "jwt": {
         "_comment": "Enable JWT verification for protected endpoints. Public
   key is loaded from $RP_CONFIG_PATH/security/public.pem.",
         "enabled": true
       },
   ```

Generate a Private Key

7. On the VM where Converged Application Server is installed, navigate to the Oracle Home directory.

8. In the Oracle Home directory, generate a new private key to encrypt node-to-node communications.

   ```
   openssl genrsa -out private.pem 2048
   ```

Load the Private Key at Startup

9. Open the domain environment file: `<Domain_Home>/bin/setDomainEnv.sh`.
10. Add the following options to the JAVA_OPTIONS parameter.

    ```
    -Doracle.sdp.mgs.jwt.pem.path=$ORACLE_HOME/private.pem
    ```

11. Start the Converged Application Server.

Configure TLS

1. Generate the public key for the previously created private key.

   ```
   openssl pkey -in private.pem -pubout -out public.pem
   ```

2. Copy the public key to each VM that runs either a Media Flow Engine and an RTP Proxy.
3. If your Managed Servers run on separate VMs, copy the private key to each Managed Server.
4. Use the `keytool` command to create a keystore (for example, `client-truststore.jks`) and import the public key into the keystore.

   For instructions on completing this step, review the keytool documentation.
5. Append the following options in the `JAVA_OPTIONS` parameter of both the Admin Server and Managed Servers start-up script:

   ```
   -Djavax.net.ssl.trustStore={ORACLE_HOME}/client-truststore.jks \
   -Djavax.net.ssl.trustStorePassword=changeit \
   -Djavax.net.ssl.trustStoreType=JKS"
   ```

6. Restart the Admin Server and the Managed Servers.

Enable TLS in the MFE

7. Connect to the VM where you installed the MFE.
8. Navigate to the MFE's `config` directory.

   ```
   cd media-flow-engine-8.3.0.0.0-MFE/MFE_HOME/config/
   ```

9. In the `application.yaml` file, set the `server.tls.enabled` property to true.

   ```
   server:
     http-port: 8080
     https-port: 8443
     tls:
       enabled: true
       client-auth: "NONE"
       protocols: [TLSv1.3, TLSv1.2]
   ```

5

Set System Properties

When starting the Converged Application Server, you may optionally pass in any of the following arguments that control system properties.

Table 5-1 System Parameters

Argument	Description	Default Value
mediagw.lb.healthCheckTimeoutSeconds	Timeout (in seconds) for an MFE health check request.	1
mediagw.lb.healthCheckStartDelaySeconds	Delay (in seconds) before starting health checks for a newly added MFE.	1
mediagw.lb.healthCheckIntervalSeconds	Interval (in seconds) between consecutive MFE health check requests.	1
mediagw.lb.consecutiveHealthCheckFailureCount	Number of consecutive failed health check requests before an MFE is marked unhealthy.	1
mediagw.lb.scheduler.corePoolSize	Thread pool size used to run MFE and RTP health checks.	10
mediagw.lb.connectionTimeout	MFE and RTP Health Check HTTP Connection Timeout(in seconds).	1
rtpproxy.lb.healthCheckStartDelaySeconds	Delay (in seconds) before starting health checks for a newly added RTP proxy.	1
rtpproxy.lb.healthCheckIntervalSeconds	Interval (in seconds) between consecutive RTP proxy health check requests.	1
rtpproxy.lb.healthCheckTimeoutSeconds	Timeout (in seconds) for an RTP proxy health check request.	1
rtpproxy.lb.consecutiveHealthCheckFailureCount	Number of consecutive failed health check requests before an RTP proxy is marked unhealthy.	1

6
Configure Overload Protection

Overload Protection (OLP) is a critical feature in Converged Application Server designed to safeguard the Media Feature Engine (MFE) cluster against service degradation or crashes due to unexpected spikes in traffic. Effective OLP configuration helps ensure overall system stability and maintains service availability, even under heavy load conditions.

OLP applies uniformly to all MFEs in the cluster. Oracle strongly recommends enabling Overload Protection to prevent individual MFEs and the collective cluster from being overwhelmed during traffic surges. Proper configuration of OLP thresholds enables intelligent traffic rejection or throttling before system resources are exhausted, enabling graceful degradation instead of abrupt failure. You can configure, add, update, remove, and retrieve Media Feature Engine Overload Protection settings in Converged Application Server through the Remote Console, the REST API, or the WLST.

Remote Console

To add OLP using the Remote Console:

1. In the Edit Tree, navigate to **Custom Resources**, and then **IntelligentMediaConnector**, and then **Media Connector Configurations**, and then **MFE Overload Protection**, and then **Resource Threshold**.
2. Click **New**.
3. Select the resource (CPU, Memory, StreamCount) to protect.
4. Set the upper limit and the lower limit for this resource.
5. Click Create.
6. Click the shopping cart and then **Commit Changes**.

REST APIS

To add OLP using the REST APIs:

```
curl -v -X POST \
  -u <username>:<password> \
  -H X-Requested-By:MyClient \
  -H Accept:application/json \
  -H Content-Type:application/json \
  -d '{
  "resourceName": "<CPU|Memory|StreamCount>",
  "min": <lower_limit_integer>,
  "max": <upper_limit_integer>
}' \
  "http://<adminserver_host>:<port>/management/weblogic/latest/edit/
customResources/IntelligentMediaConnector/customResource/
mediaOverLoadProtection/resourceThresholds/resourceThreshold"
```

See the *Developer Guide* for more information about the REST API.

WLST

To add OLP using the WLST:

1. Navigate to the <Domain_Home>/bin directory.
2. Set the environmental variables.

```
. ./setDomainEnv.sh
```

3. Save the following file to a script.

```
try:
    connect("<username>", "<password>", "t3://<adminserver_ip>:<port>")
    edit()
    startEdit()

    cd('/CustomResources/IntelligentMediaConnector')
    ms = cmo.getResource()
    mop = ms.getMediaOverLoadProtection()
    rtb = mop.getResourceThresholds()

    name = '<threshold_name>'  # e.g. CPU / Memory / StreamCount

    rt = rtb.lookupResourceThreshold(name)

    if rt is not None:
        print "ResourceThreshold already exists:", name
        cancelEdit('y')
        print 'res:0'
        disconnect('true')
        exit('true')

    rt = rtb.createResourceThreshold(name)
    rt.setMin(<min_value>)
    rt.setMax(<max_value>)

    save()
    activate(block="true")
    print 'res:1'

except Exception, e:
    print e
    dumpStack()
    print 'res:0'

disconnect('true')
exit('true')
```

4. Run the script.

```
java weblogic.WLST /path/to/file.py
```

7

Deploy the Sample UA Application

Before starting the UA-APP deployment, make sure to create a configuration file named `mfe-app.properties`.

You can find a sample `mfe-app.properties` file in the UA-APP sample project.

1. Locate the file `IMCSampleApp-8.3.0.0.0.zip` in your download of the Converged Application Server Release 8.3.
2. Unzip `IMCSampleApp-8.3.0.0.0.zip`.
3. Locate the `mfe-app.properties` file.
4. Copy the properties file to the Oracle Home folder.
5. In the properties file, set `egress.destinationIp` to the IP address of the sample AI connector service. Set `egress.destinationPort` to the port of the AI connector service.

   ```
   egress.destinationIp=<IP of AI Connector Service>
   egress.destinationPort=<Port of AI Connector Service>

   egress.codec.name=PCM-16
   egress.codec.samplingRate=8000
   egress.codec.channelCount=1
   egress.codec.targetBitrate=64000
   egress.codec.format=S16LE
   egress.headerSize=146

   egress.languageCode=en-US
   egress.partialSilenceThreshold=50
   egress.finalSilenceThreshold=500
   egress.streamDirection=incoming
   ```

6. You need to tell your managed server where to find `mfe-app.properties`. You have two ways to do this.
   - Add `-Dmfe.config.file=<absolute path to mfe-app.properties>` to the server start command.
   - Update the `JAVA_OPTIONS` variable in the `startWebLogic.sh` file and then restart your Admin and Managed servers