



CMP 8.23

Installation Guide

Version 1.0

Classification: **Customer Confidential**



Copyright

© MDS Global 2026

THE CONTENTS OF THIS DOCUMENT ARE THE COPYRIGHT OF MDS GLOBAL LTD. ALL RIGHTS RESERVED. THIS DOCUMENT OR PARTS THEREOF MAY NOT BE REPRODUCED IN ANY FORM WITHOUT THE WRITTEN PERMISSION OF MDS GLOBAL.

Confidentiality

This document contains information that is proprietary to MDS Global and is confidential. The original recipient of this document may duplicate this document in whole or in part for internal distribution only, provided that this entire notice appears in all copies. This document and its contents may not otherwise be reproduced, distributed or disclosed. The recipient agrees to make every effort to prevent the unauthorised use, distribution or disclosure of the proprietary information contained in this document.

Disclaimer

No representation or warranty is contained in, made or given by this document or the information contained within it and no warranty or representation is made or to be implied that the information contained in this document is complete, up to date, accurate or fit for the purpose for which this document is supplied. In no event shall MDS Global be liable for incidental or consequential damages or loss in connection with, or arising from its use, whether MDS Global was made aware of the probability of such damages or loss arising or not.

Trademarks

The teal symbol above is an unregistered trademark of MDS Global Ltd. Other trademarks referred to within this document are the property of their respective trademark holders.

Contact Details

Please visit www.mdsglobal.com for further information on MDS Global products, solutions and services.

ISO 22301 standard is applicable to MDS Global Business Operations.



Table of Contents

Table of Contents	ii
Version Control	iv
Terms Used in this Document	v
1.0 Assumptions	1
2.0 About CMP Installation	2
2.1 RPM packages	2
2.1.1 Yum Repository	2
2.2 Ansible	2
2.3 CMP Automatic Deployment Process	4
Step 1	5
Step 2	6
Step 3	7
Step 4	8
2.4 Encrypting Sensitive Information	8
2.5 Memory Allocations	10
2.6 High Availability Stack Installation	10
2.7 Health Check Service	12
3.0 Prerequisites	13
3.1 High Availability Prerequisites	16
3.1.1 Load Balancer Virtual Services	18
3.1.2 High Availability for Reports	20
3.2 Internal TLS Certificates	20
3.2.1 External Client Access	21
3.2.2 Internal Trust Management	22
3.2.3 Java Truststores	22
3.2.4 Operating System Trust Store	22
3.2.5 Certificate Lifecycle	23
3.3 PKI Certificate Storage and Retention	23
3.3.1 PKI storage location	23
3.3.2 Why the PKI files must be retained	24
3.3.3 Requirements for upgrades and subsequent deployments	24
3.3.4 Backup and security	25
3.3.5 Multiple CMP system instances	25
3.4 SABRE Server Encryption	26
3.5 Network Communication	27
3.5.1 Intra Stack Communication	27
3.5.1.1 High Availability Intra Stack Communication	30
3.5.2 External Access	31
4.0 Pre-Installation Tasks	33

5.0 Installation Tasks	35
5. 1 Installation Configuration Tool	35
5. 1. 1 Preparing an Inventory File with the Installation Configuration Tool	35
5. 1. 2 Working with an Existing Inventory File	42
5. 2 Pre-Deploy CMP	43
5. 3 Install Cached version	44
5. 4 Deploy CMP	44
5. 4. 1 Common Ansible Command Line Parameters	45
5. 4. 2 Summary File	49
6.0 Upgrading an Existing Installation	50
6. 1 Certificate Updates	51
6. 2 Patch Updating	51
6.0 Copying Database-Based Configuration	53
7.0 Troubleshooting	55
8.0 Configure Azure Active Directory as a SAML Federated Identity Provider in WSO2	60
9.0 Post-Installation Checks and Tasks	70
9.0 High Availability Deployment Post-Installation Tasks	72
10.0 Uninstall CMP	73

Version Control

Version	Issue Date	Author	Comments
Version 1.0	21 August 2026	MDS	CMP 8.23 Release - Various updates have been made including updates to the certificates topics, load balancer High Availability Prerequisites moved to Prerequisites, and the latest Playbooks and WSO2 screens added.

Terms Used in this Document

For definitions and explanations of the terms, abbreviations and acronyms used in this document, please see the *CMP Glossary* document.

1.0 Assumptions

The installation and deployment instructions provided in this document are based on a number of assumptions:

- **IMPORTANT:** Those performing the installation have *at least* basic Linux administration knowledge, an appreciation of networking fundamentals and an understanding of Ansible automatic deployment.



Ensure you are familiar with the concepts in "About CMP Installation" on page 2.

- For production or production grade installations, that although the database installation is automated, a suitably qualified PostgreSQL Database Administrator is associated with the configuration, installation and ongoing operation of CMP.
- A person performing a High Availability installation of CMP requires a higher level of Linux, networking and database administration expertise because such installations are technically more complex.
- Servers are available with the capacity and configuration as defined in conjunction with MDS Global to meet processing requirements of the installation.
- A load balancer is deployed and configured as part of the installation.
- Ansible experience is beneficial.

2.0 About CMP Installation

CMP is delivered as a set of components that are RPM packages and it is installed and deployed by running an [Ansible playbook](#).

The installation repository for CMP is available on the CMP host in the cloud (<https://vault.mdsglobal.dev/>), accessible via HTTPS with the credentials provided by MDS Global, according to the customer's contract. It includes the following:

- YUM and Maven repositories, which store the RPM packages and database package, as well as the Ansible Playbook package used for deployments.

You need download only the Ansible playbook package to the control server. The RPM and database packages are deployed automatically when you run the playbook.

2.1 RPM packages

The RPM is a program for installing, uninstalling, and managing software packages in Linux. The *RPM* portion of this term comes from the fact that `.rpm` is the default extension for files used by the program.

Because CMP components are delivered as RPM packages, this dictates where in the operating system each file is deployed. It is not possible to deploy components anywhere else on the host file system.

In addition, the RPM-based deployment does not allow more than one instance of a component to a host. For example, you cannot install multiple instances of the AgentView Interfaces Layer on the same host.

 For more information, see <https://rpm.org/>.

2.1.1 Yum Repository

CMP RPM packages are stored in YUM and Maven repositories.

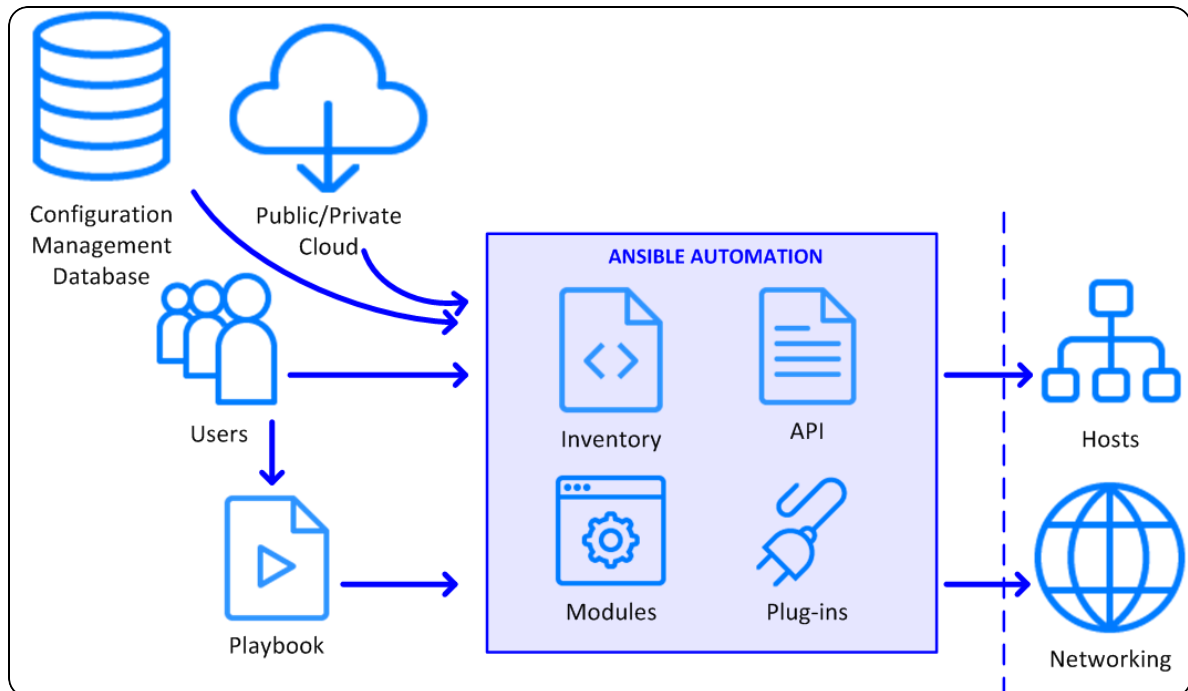
YUM (Yellowdog Updater, Modified) is an open-source command-line package-management utility for computers using the RPM Package Manager. YUM Repositories are warehouses of RPM package files that can hold RPM packages locally or remotely.

2.2 Ansible

Ansible is a configuration management tool. Although it can be run in *ad hoc* mode to run commands manually like any command line interface tool, it is usually used to automate the manual tasks of configuration and installation, such as shutting down and restarting hosts, installing packages and components, and deploying configurations.

Ansible is agent-less and communicates with hosts using SSH (Secure Shell) for Unix/Linux and WinRM for Windows installations. For CMP purposes, SSH is used.

 For more information see <https://docs.ansible.com/>.



Ansible Architecture

CMP deployment uses Ansible automatic installation, which involves the following concepts:

Control Server

For CMP purposes a control server is a server that runs Linux and has Ansible installed. You use it to run commands and *playbooks*.

Playbooks

A playbook is an ordered list of tasks, which are run sequentially. Playbooks can declare configurations, but they can also orchestrate steps of any manual ordered process, even as different steps must move back and forth between machines in particular sequence.

 For more information, see https://docs.ansible.com/ansible/latest/user_guide/playbooks_intro.html.

Playbooks are written in YAML (YAML Ain't Markup Language), which is a human-readable data serialisation language that is simple to use and commonly used in configuration files. CMP provides playbooks for the installation and deployment of CMP components.

 For more information, see <https://yaml.org/>.

Each *task* in the playbook is a unit of action and correlates to a single command that you would run in ad hoc mode. Tasks invoke one or more modules, which are the units of code executed by Ansible. Each module has a particular use, such as file management, package management and remote execution. Ansible provides an extensive list of modules for most common tasks and you can also write your own. CMP provides modules for the installation and deployment of CMP components.



For more information, see https://docs.ansible.com/ansible/latest/modules/modules_by_category.html.

Inventory File

An inventory file is a list of the target hosts (or *managed nodes* in Ansible terminology). That is, they are the machines on which you want to install the CMP components. The inventory file uses variables (properties) to describe the desired state of the target hosts. The hosts can be grouped and nested, usually according to their role or function, such as database or web server, or load balancer, for example.



The inventory file is created and modified using the Inventory Configuration Tool.

A number of host groups have been predefined for CMP; see the *Groups* table in [Prepare the Inventory File](#) for more information.

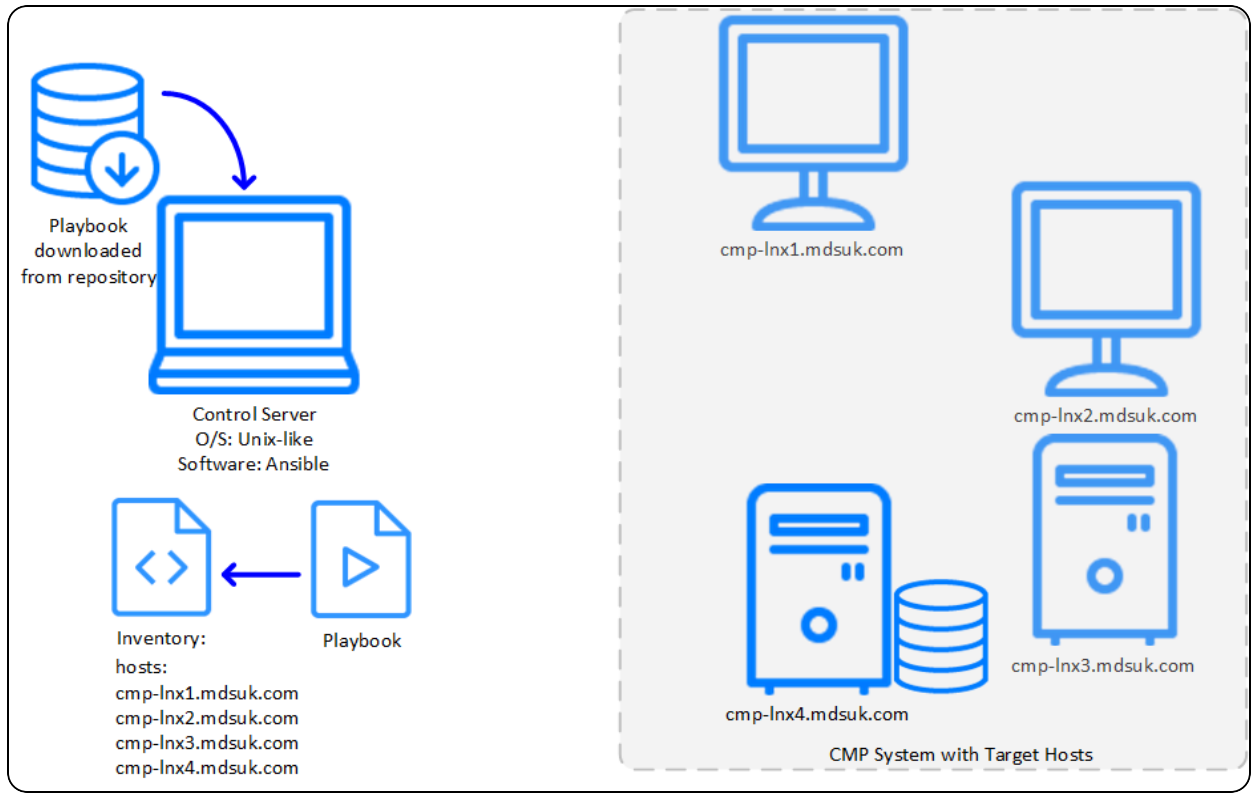
CMP provides a YAML inventory file template (`template.yaml`) that you must copy and edit to suit your deployment.

For more information, see [Prepare the Inventory File](#).

2.3 CMP Automatic Deployment Process

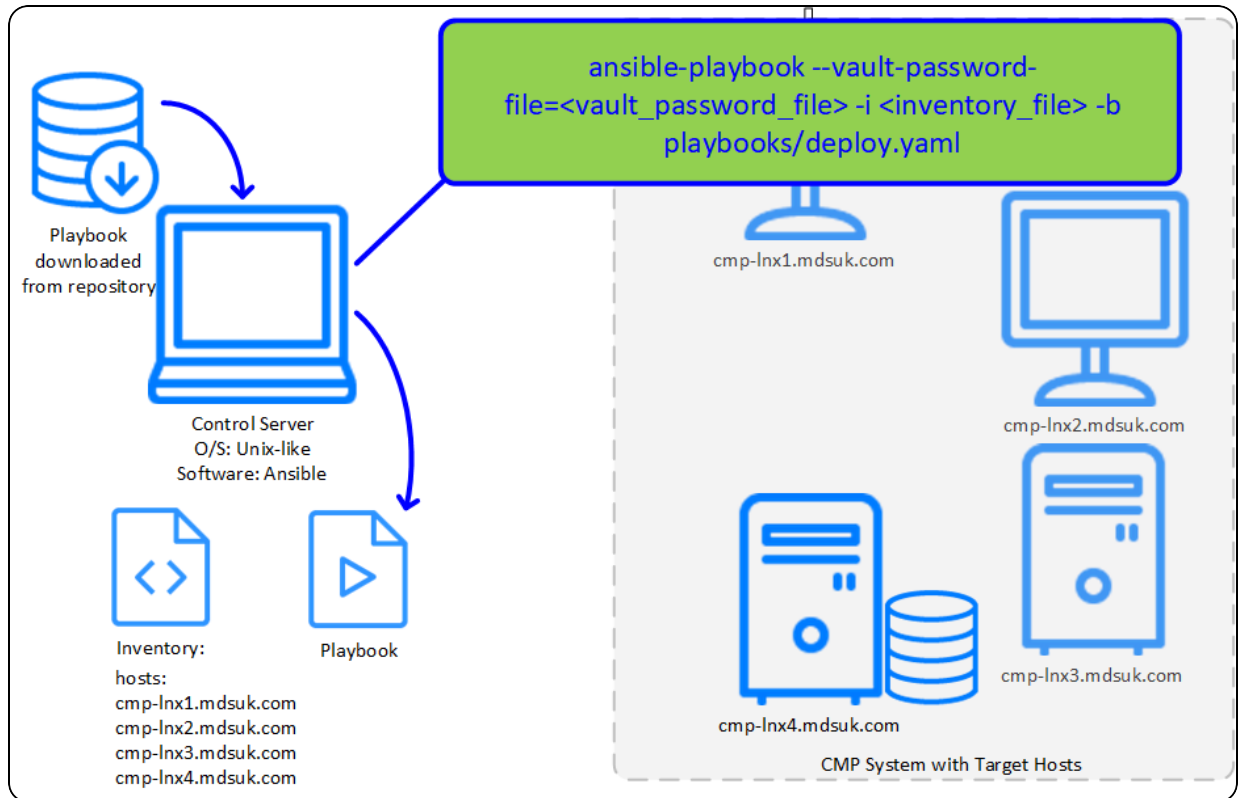
The Ansible deployment process used to deploy CMP is as follows:

Step 1



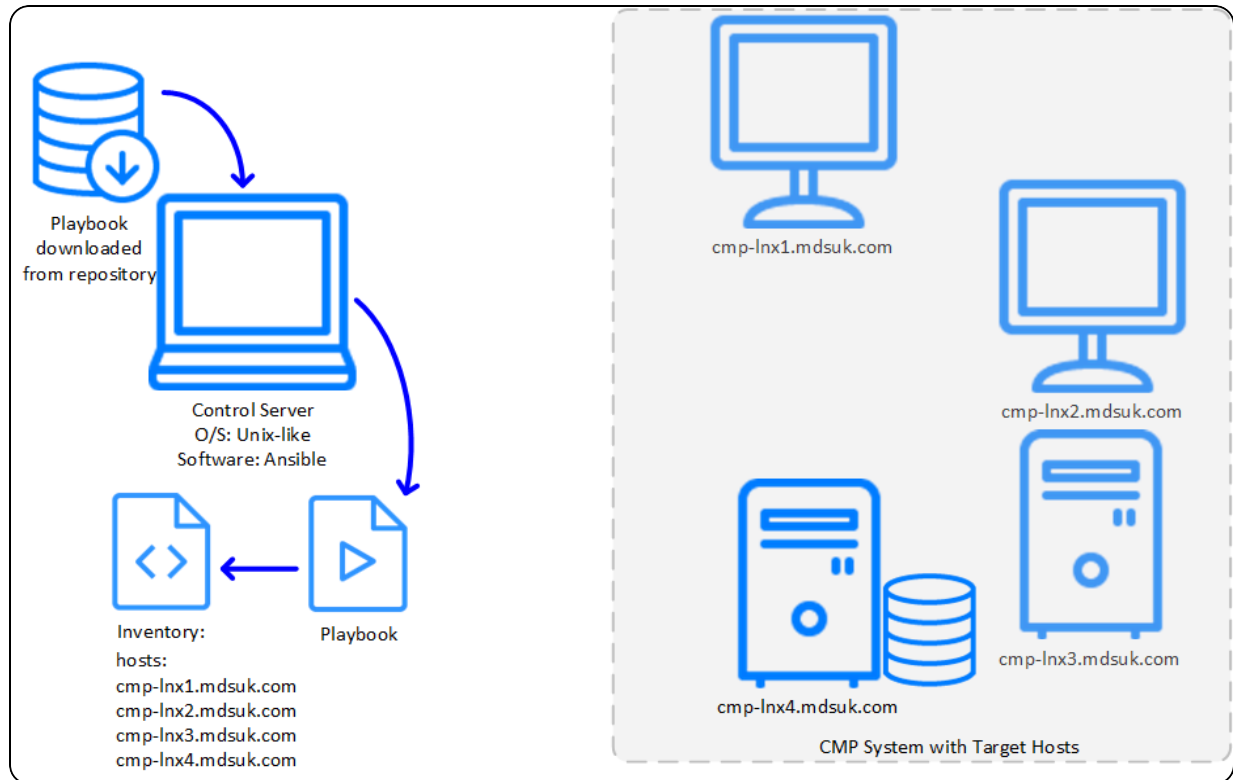
Download and unzip the Ansible Playbook package to the control server (controller node) and prepare the inventory file using the Inventory Configuration Tool.

Step 2



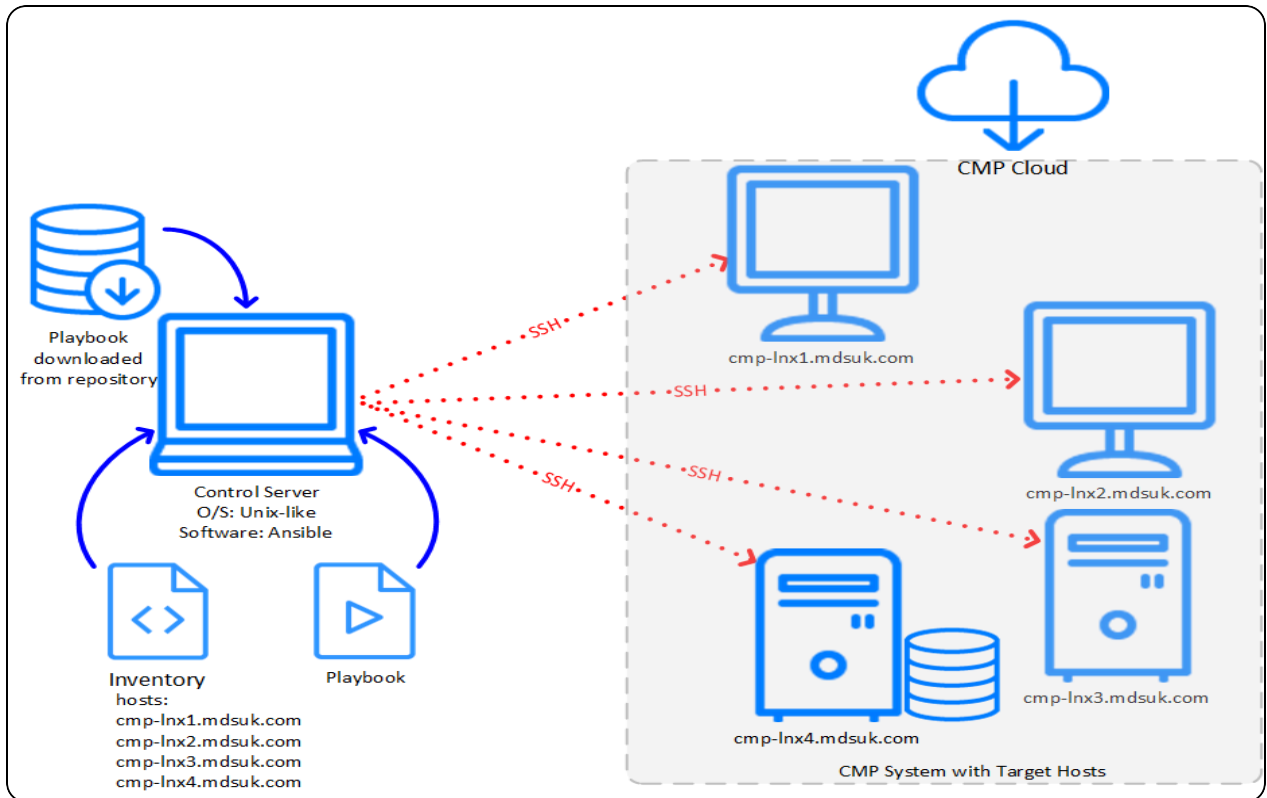
Run the command to execute the deployment playbook.

Step 3



As Ansible runs step by step through each task in the playbook, it encounters deployment instructions for groups of hosts. It consults the inventory to determine the list of specific hosts and what configuration is required for each.

Step 4



Having identified the hosts, the control server communicates the configuration and deployment instructions to the hosts via SSH. The hosts access the CMP installation repository in the cloud and deploy the required packages and configuration.

2.4 Encrypting Sensitive Information

Ansible Vault is an Ansible feature that enables you to keep sensitive data such as passwords or keys in encrypted form.

Although Ansible supports both encrypted vault files to specifically hold sensitive data or encryption of individual fields directly in the inventory file, the CMP installation uses the encryption of individual fields method. If a vault password is specified when saving the inventory file from the Inventory Configuration Tool then all data sensitive fields will be encrypted.

When sensitive data is encrypted, one of the following command line flags should be used when running the `ansible-playbook` program: `--ask-vault-pass` or `--vault-password-file`.

The `--ask-vault-password` parameter forces the `ansible-playbook` program to prompt the user to enter the vault password and like `ansible-vault` above, the `--`

`vault-password-file` property points to the plain text file where the vault password is stored.

2.5 Memory Allocations

It is possible to adjust the allocated JVM heap size for individual CMP applications in some cases, in the expandable *Advanced Properties* section in the Installation Configuration Tool. These values should only be changed from the defaults after discussion with MDS Global or based on server sizing details provided by MDS Global.



However, when choosing the minimum and maximum heap sizes for a CMP module, it is essential not to oversubscribe the available physical memory on any particular host.

You must ensure that the sum total of the maximum heap sizes for all of the CMP modules on a particular host does not exceed the memory available to CMP on the host. You can calculate available memory by subtracting the memory required for all non-CMP processes (for example, the memory requirement for the OS) from the total available memory on the host itself.

2.6 High Availability Stack Installation

Most production-type deployments require that all CMP components are [highly available](#)¹.

This means that:

- More than one instance of each component is configured.
- Multiple instances are clustered; that is, they can transfer state in the real time between instances, or the instances don't hold any state so can be used indiscriminately.
- The connectivity into each component is load balanced over all instances of that component, with the load balancer using the ["Health Check Service" on page 12](#) to determine if each instance should receive traffic.
- The connectivity of each of the components should be via an external load balancer except for those that require only internal network interaction. This is the case for both Shared Services and SAM Services components which should be configured to use an internal load balancer instead. This is to improve the performance for these micro services by preventing unnecessary external traffic routing and therefore resulting in reduced network latency.

Important:

¹High availability is a characteristic of a system, which aims to ensure an agreed level of operational performance, usually uptime, for a higher than normal period.

Before you install a high availability deployment, read the section [High Availability](#) in the *CMP Technical Architecture Overview*, which describes the requirements and responsibilities of those who will install and maintain a high availability system. These requirements and responsibilities must be met for a successful deployment.

In addition, note that configuring the replication and the load balancing is not part of the CMP deployment and must be configured using relevant third party software and hardware as a separate step to the installation itself.

The CMP stack can be installed in different configurations:

- Simple installation, when each application component has only one instance.
- Highly available, when all the components of the stack are highly available.
- Mixed, when some application components are highly available and some are represented by the single instance.

2.7 Health Check Service

CMP provides a global health check service which it is mandatory for any external load balancer to use to determine the health of instances of CMP modules. The service checks the `cmpModuleMaintenance` table to determine whether the module is configured to be online, and then checks whether the module is running correctly and able to handle traffic. This allows for modules to be placed into maintenance mode. This can be used by the load balancer and allows any HA deployment to place modules into maintenance mode so that they appear unavailable on the load balancer when upgrading.

It is deployed on the same port (default of 21215) across all of the hosts on which the CMP application is installed.

The global health service can be accessed through the following URI:

```
https://{hostName}:21215/healthCheck/{moduleName},
```

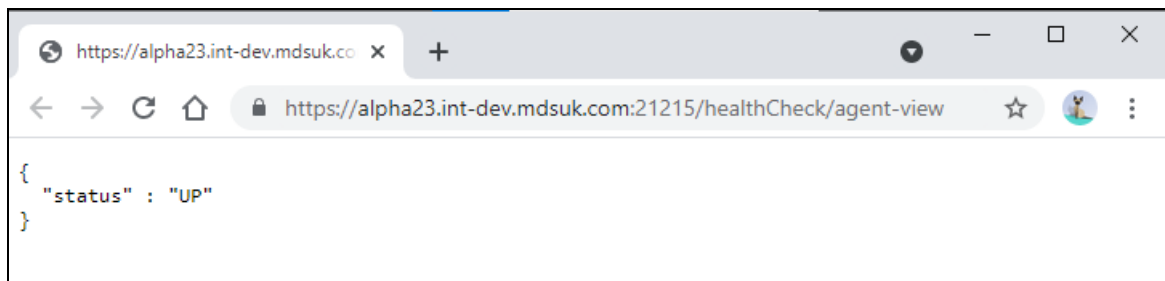
where `{hostName}` is the host on which the module is deployed and `{moduleName}` is the name of the specific module. For example:

```
https://alpha23.int-dev.mdsuk.com:21215/healthCheck/agent-view
```

The service returns a response such as:

```
{ "status" : "UP" },
```

where `"status"` can be one of `"UP"`, `"DOWN"`, `"MAINTENANCE"`, or `"UNKNOWN"`, for example:



The status of `"UNKNOWN"` will only be returned for a module that does not have an application specific health check defined.

The global health service is configured in the Global Properties in the inventory. See ["Installation Configuration Tool" on page 35](#) for more information.

3.0 Prerequisites

Before starting deployment ensure that the following are in place:

- A **control server**¹ is available that runs Unix, is connected to the network and has Ansible 9.2.x with Ansible Core 2.16.x, and python3.12-jmespath, and a version of the unzip utility installed.

Note:

If different Ansible and Ansible Core versions (higher or lower) are already installed on the control server they should be removed before installing the version above. If not readily available from the systems standard repositories, the correct Ansible version can be installed from an MDS Global hosted repository as follows:

1. Remove any Ansible package exclusion in place by editing `/etc/dnf/dnf.conf` and removing any line stating `exclude=ansible*`
 2. Add the relevant MDS Global repository: `sudo dnf config-manager --add-repo https://<username>:<password>@vault.mdsglobal.dev/repository/repos/cmp-pre-req.repo` replacing username and password with the repository credentials supplied by MDS Global for CMP installation
 3. Edit `/etc/yum/repos.d/cmp-prereq.repo` to replace `<username>` and `<password>` with the repository credentials supplied by MDS Global for CMP installation
 4. Install Ansible with Ansible Core and the Python jmespath: **`sudo dnf install ansible python3.12-jmespath`**
 5. Exclude updates for ansible packages by editing `/etc/dnf/dnf.conf` and adding the following line `exclude=ansible*`
- The control server is locked to the above Ansible version so that is not inadvertently updated by Linux package updates (i.e. add a line with `exclude=ansible*` at the end of the `/etc/dnf/dnf.conf` file on the server).
 - Red Hat Enterprise Linux is installed and updated on each target host.
-
-  For more information, see *Third Party Software Versions* in the *CMP Technical Architecture Guide*.
-
- The time on all servers being used is synchronised and set up using Network Time Protocol.
 - The timezone on all the servers is set to the timezone in which users expect to schedule activities and to be used to trigger processing of events. This is usually the local timezone of the location where CMP is being used.

¹A computer on a network used to access and manage devices. For Ansible purposes, a control server runs Linux/Unix and has Ansible installed.

- The intended target hosts and the control server are connected to a network.
- Load balancing must be configured for each software component of the CMP stack (except Database).
 - The load balancer URLs must be predefined because this information needs to be added to the inventory file via the Installation Configuration Tool.
 - When using a load balancer with CMP, the chosen make and model must support the following:
 - WebSockets - required for the correct functioning of AgentView/Web-swing.
 - HTTP Health Checks - to determine the status of a service.
 - Varying Timeouts - timeouts can vary from short (seconds) to long (hours), needed to support long session timeouts with AgentView.
 - Session Affinity - for services such as AgentView and Administration Console that have interactive user interfaces requiring a connection to a unique instance for the session.
- Each intended target host can be accessed by their external hostname which is also known to the load balancer; that is, the name that will be used in the Inventory Configuration Tool for the installation, by the end users and support staff to access the target hosts from outside of the CMP stack. You can achieve this by:
 - Adding the respective hostname/IP address pairs to the DNS server.
 - Adding the respective hostname/IP address pairs to the `local hosts` file on the control server and all target hosts.
- The target servers can communicate with one another over the ports required for the intra stack communications (see the table in "[Network Communication](#)" on page 27). If there are firewalls between the target servers other than the default Linux firewall (firewalld), the communications described in the intra stack communications table must be allowed.



It is essential that on each host the external hostname resolves to the local IP address of the server that the server itself understands is its own address, and not to an external address which is routed to the server.

- The control server can communicate with the target host via SSH.
- The user account that will be used by Ansible is created on each target host. This account must have:
 - Either password access or the client public key added to list of the authorised keys for the user in question, allowing key-based authentication for the user.

Ensure that you can log into the target hosts over SSH from the control server with the password or the private key prior to starting deployment.

Ensure that the corresponding server keys are added to the `known_hosts`

file on the control server.

- `sudo`¹ access without a password.
- The customer has requested access to the MDS Global repository. The request must include the public IP addresses (IP address range) of the control server and the target hosts. MDS Global will whitelist the provided address(-es), allowing connectivity to the repository and will provide set of credentials that must be used both to download the installation scripts and to enter in the inventory file.

For more information on the inventory file, see [Prepare the Inventory File](#).

- For Production systems it is recommended to create separate filesystems for the database instance otherwise the database tablespace files will be stored in the “root” Linux filesystem. By default, the following locations are used to store the database data:
 - /pg_wal
 - /pg_logs
 - /pg_temp
 - /var/lib/pgsql
 - /pg_data/mdscmp/pg_index_data
 - /pg_data/mdscmp/pg_small_data
 - /pg_data/mdscmp/pg_large_data
 - /pg_data/mdscmp/pg_archive_data
 - /pg_archivelogs

The last two filesystems above are ideally suited to slower storage devices and do not need the cost associated with fast disk / SSD.



RedHat administration knowledge is needed to perform the creation of filesystems.

PostgreSQL Database Administration knowledge is needed to correctly configure, manage and maintain the database for a production system.

- The required public SSL certificates have been generated and installed on the load balancer.

For more information, see ["Internal TLS Certificates" on page 20](#).

- Any security certificates required to explicitly trust a system that will be connected to from the CMP product itself, a customer specific adapter deployed with CMP, or any other process running on the server are installed in a subdirectory of the default systemwide trust store and not in the Java trust store, as the content of the Java

¹A program for Unix-like computer operating systems that allows users to run programs with the security privileges of another user, by default the superuser.

trust store can be deleted as part of the CMP installation process.

See [Shared System Certificate Storage](#) for more.

- For Red Hat Linux installations only:
 - The target hosts are either registered with Red Hat, which allows management of the YUM repositories via Subscription Manager, with an active attached subscription for the YUM repositories required for each host.
 - OR, the target system has access to the hosted YUM repositories with JBoss and JBoss Web Server RPM packages, in the supported versions as per *Third Party Software Versions* in the *CMP Technical Architecture Guide*:
 - All target hosts require access to the Red Hat Enterprise Linux Server product subscription.
 - JBoss target hosts also require access to the Red Hat Enterprise Application Platform product subscription.
 - JBoss Web Server target hosts also require access to a product subscription for JBoss Web Server.

See the note below:

Important

The deployment process relies on access to the RedHat YUM repositories (for example, `jb-eap-7.2-for-rhel-7-server-rpms/jws-5-for-rhel-7-server-rpms`) or their equivalents.

Access to the JBoss Enterprise Application Platform repository is only required on the server that hosts JBoss and related application components (jboss Ansible inventory group of hosts). Similarly, access to the jws repository is only required on the server that hosts JBoss Web Server and related application components (jws Ansible inventory group of hosts). Corresponding subscriptions must be active on the servers belonging to the respective Ansible inventory group.

All target hosts require access to the 'Red Hat Enterprise Linux Server' product subscription or equivalent hosted YUM repository.

3.1 High Availability Prerequisites

By default, Ansible expects the shared storage to be mounted as follows:

Required

- Shared storage for the Artemis messaging server (ActiveMQ server) and its broker persistence directory, should be mounted to `/var/mdsglobal/amq-`

`broker/data` directory on all instances that host Artemis Server.

- Shared storage for the SABRE server (batch server) should be mounted to `/var/mdsglobal/sabre-server` directory on all instances that host SABRE Server.
- Shared storage for the Bulk Action Console server (bulk action server) should be mounted to `/var/mdsglobal/bulk-action-console` directory on all instances that host Bulk Action Console.
- Shared storage for the Report Server (Pentaho server) should be mounted to `/var/mdsglobal/pentaho-solutions` directory on all instances that host Report Server. (See ["High Availability for Reports" on page 20](#)).

Optional

- Shared storage for the Grafana Mimir (Grafana monitoring server) should be mounted to `/var/mdsglobal/mimir` directory on all instances that host Grafana Server.
- Shared storage for the Aurora Console should be mounted to `/var/mdsglobal/aurora-console` directory on all instances that host Aurora Console.

The table below is an overview of the required base folder configuration:

Shared Base Directory	Owner / Group UID / GID	Permissions
<code>/var/mdsglobal/</code>	<code>root:root</code>	<code>755</code>
<code>/var/mdsglobal/amq-broker/data</code>	<code>artemis 210:210</code>	<code>770</code>
<code>/var/mdsglobal/sabre-server</code>	<code>sabre 201:201</code>	<code>770</code>
<code>/var/mdsglobal/bulk-action-console</code>	<code>bulkaction 205:205</code>	<code>755</code>
<code>/var/mdsglobal/pentaho-solutions</code>	<code>tomcat 212:212</code>	<code>750</code>
<code>/var/mdsglobal/aurora-console</code>	<code>aurora 216:216</code>	<code>755</code>
<code>/var/mdsglobal/mimir</code>	<code>mimir 215:215</code>	<code>750</code>

For the Ansible installer to create the required sub directories and setup permissions for the above data shares during the deployment, the base folder must be configured with the correct permissions and configuration.

As there are many options available to create a shared storage solution, it is beyond the scope of this document to specify which solution to use and how that should be setup. However, as a guide when using the common and simple Linux NFS Share Solution, below is an example of the `/etc/exports` file configuration that should be used:

```
/var/mdsglobal/amq-broker/data <ip address> (rw, sync, no_subtree_
check, no_root_squash)
```

```
/var/mdsglobal/bulk-action-console <ip address> (rw, sync, no_sub-
tree_check, no_root_squash)
```

```
/var/mdsglobal/sabre-server <ip address>(rw, sync, no_subtree_
check, no_root_squash)

/var/mdsglobal/pentaho-solutions <ip address>(rw, sync, no_subtree_
check, no_root_squash)
```

3. 1. 1 Load Balancer Virtual Services

For the load balancer it is necessary to create the virtual services before an HA CMP installation takes place because URL access is checked during the installation process. The following table contains an example set of load balancer services:



The ports used are the standard ones in the Inventory Configuration. If they are changed as part of the installation then the ports accessed by the load balancer would need to be updated accordingly.

Service Name	Protocol	Default Port	Health Check Name	Timeouts	Affinity
AgentView / Pentaho	HTTPS	Port 7443	/healthCheck/agent-view	28,800 seconds (8 hours)	Auto generated session Cookie
WSO2 Identity Server	HTTPS	Port 9443	/healthCheck/wso2is	30 seconds	Client IP
Role Extender	HTTPS	Port 8081	/healthCheck/role-extender	30 seconds	None
SOAP Web Services	HTTPS	Port 8443		30 seconds	None
REST Web Services	HTTPS	Port 9000	/healthCheck/rest-ws	30 seconds	None
Business Configuration	HTTPS	Port 8443	/healthCheck/configuration-centre	30 seconds	Client IP
Bulk Action Console	HTTPS	Port 9009	/healthCheck/bulk-action-console	30 seconds	Client IP
CMP Administration Console	HTTPS	Port 31212	/healthCheck/sabre-console	30 seconds	Client IP
Artemis / ActiveMQ Broker	TCP/SSL	Port 61616	/healthCheck/apache-active-mq-broker	30 seconds	None
Artemis / ActiveMQ Console	HTTPS	Port 8161	/healthCheck/apache-active-mq-console	30 seconds	Client IP
Shared Services	HTTPS	Port 21218	/healthCheck/shared-services-rest	30 seconds	None
SAM Services	HTTPS	Port 21220	/healthCheck/sam-services	30 seconds	None
Voucher Management Services	HTTPS	Port 21223	/healthCheck/voucher-management-services	30 seconds	None
Context Sensitive Help	HTTPS	Port 21221	/healthCheck/context-sensitive-help	30 seconds	None
SPaRC Engine	HTTPS	Port 21222	/healthCheck/sparc-engine	30 seconds	None
Aurora Console	HTTPS	Port 9010	/healthCheck/aurora-console	30 seconds	Client IP
Grafana	HTTPS	Port 3000	healthCheck/grafana	30 seconds	Client IP
Prometheus	HTTPS	Port 9090	healthCheck/prometheus2	30 seconds	Client IP

For the load balancer services DNS names must be configured in the CMP inventory. As described above these are needed BEFORE the installation of CMP is started.

Note:

If host and path rules are used in the load balancer to direct traffic to the correct services, only DNS subdomain configurations are supported and not path rules.

For example the following are valid:

`wso2is.cmp8.com`

`soap-services.cmp8.com`

However, the following are invalid:

`www.cmp8.com/wso2is`

`www.cmp8.com/soap-services`

This is because certain services such as WSO2IS have in-built redirects from the root context(/) that override any paths that have been configured as part of the load balancer rules.

3. 1. 2 High Availability for Reports

Recommendations:

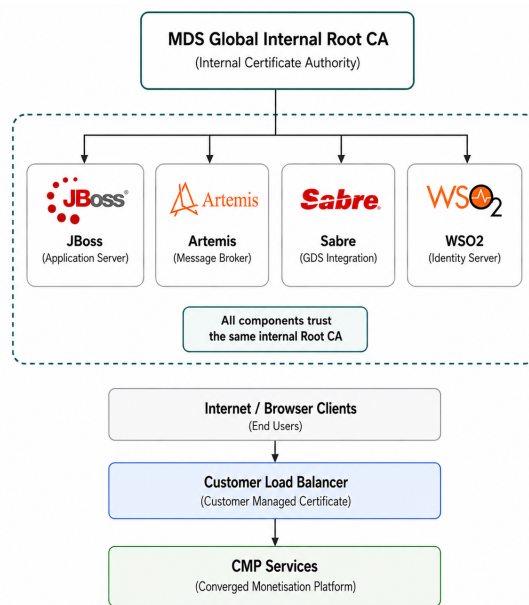
- Report Server (the Pentaho server) should be deployed to all AgentView hosts. This must be manually configured in the Installation Configuration Tool.
- Report Server should make use of shared storage as described above. This reduces the confusion/complexity of report output being stored on multiple hosts
- Run Report Server in an Active/Active state so that end users do not need to consider switching on/off schedules between active/passive nodes as needed.

3. 2 Internal TLS Certificates

All communication between CMP components is secured using Transport Layer Security (TLS).

During deployment, the CMP Ansible installer automatically provisions a complete internal Public Key Infrastructure (PKI) for the environment. This includes:

- An internal Root Certificate Authority (CA)
- Service certificates for all CMP components
- Java truststores
- Operating system trust stores
- Component-specific keystores and truststores.



CMP Trust Relationship

The generated certificates are used exclusively for securing communication between CMP services and require no manual certificate management by the customer.

No customer supplied certificates are required for internal CMP communication.

Each service receives its own certificate containing the appropriate Subject Alternative Names (SANs) for the host on which it is deployed. All certificates are signed by the deployment-specific internal Root CA, allowing every CMP component to establish mutual trust automatically.

3. 2. 1 External Client Access

Customer-facing endpoints, such as web applications exposed through a load balancer or reverse proxy, are outside the scope of the CMP internal PKI.

Customers are responsible for provisioning and managing certificates presented by external endpoints, for example:

- HTTPS Load Balancers
- Reverse Proxies
- Web Application Firewalls
- Enterprise Ingress Controllers

These certificates should be issued by a publicly trusted Certificate Authority, or by the customer's own enterprise PKI, according to their security policies.

Typical examples include:

- Google Cloud Load Balancer certificates
- AWS ACM certificates
- Microsoft AD Certificate Services
- DigiCert
- Sectigo
- Let's Encrypt

These certificates are used only for browser and external client connections and are independent of the certificates generated by the CMP installer.

3. 2. 2 Internal Trust Management

During deployment, the installer automatically configures trust between all CMP components.

This includes:

- importing the CMP internal Root CA into component truststores
- creating application-specific Java truststores
- updating operating system trust stores where required
- configuring each component to use its assigned keystore and truststore

This process is fully automated and normally requires no manual intervention.

3. 2. 3 Java Truststores

Many CMP components are Java-based applications.

The installer automatically creates and configures Java truststores containing:

- the standard Java CA certificates (copied from the default JVM cacerts truststore)
- the CMP internal Root CA certificate

Retaining the standard Java trusted certificates ensures applications continue to trust public Certificate Authorities when accessing external services, while also trusting certificates issued by the CMP internal PKI.

3. 2. 4 Operating System Trust Store

Where required, the installer also updates the operating system trust store.

This enables operating system utilities such as:

- curl
- OpenSSL
- DNF/YUM
- Apache HTTP Server

to trust certificates issued by the CMP internal Root CA.

On Enterprise Linux systems this is achieved using the system certificate trust infrastructure and the update-ca-trust utility.

3.2.5 Certificate Lifecycle

The CMP internal PKI is generated automatically during deployment.

The installer manages:

- Root CA generation
- Service certificate generation
- Truststore creation
- Keystore creation
- Certificate distribution
- Trust configuration

No manual certificate creation is normally required.

If certificates are regenerated, the installer updates the relevant keystores and truststores accordingly.

3.3 PKI Certificate Storage and Retention

The CMP Ansible installer uses the MDS Global PKI infrastructure to generate and manage the TLS certificates required by the CMP components.

PKI files are generated on the Ansible control server during the pre-deployment phase. The generated PKI material is subsequently used by the Ansible deployment to install certificates, private keys and truststores on the target CMP hosts.

3.3.1 PKI storage location

The PKI files are generated and stored in the specified PKI TLS Certificate Generation Path as entered into the inventory on the Ansible control server: `<pk_i_tls_src_files>/mdsglobal/pki/<jdbc_server>/`

The exact location is determined by the PKI configuration used by the deployment.

The directory contains the PKI material associated with the deployed CMP system, including:

- The Application Root CA certificate.
- The Application Root CA private key.
- Root CA metadata.
- Service certificates and private keys.
- Service certificate metadata.
- Generated Java truststores containing the appropriate trusted CA certificates.
- Other PKI files required by the deployment.

The PKI directory is part of the persistent deployment state and must not be treated as temporary deployment data.

3.3.2 Why the PKI files must be retained

The Application Root CA is used to issue the TLS certificates for the CMP services. Subsequent deployments and upgrades must continue to use the same Root CA and existing PKI state wherever the certificates remain valid.

The PKI files must therefore be retained on the Ansible control server after the initial deployment.

In particular, the Root CA private key must be retained securely. If it is lost and a new Root CA is generated, the new CA will have a different identity and certificates issued by the original CA will no longer be trusted by components configured to trust the new CA.

Retaining the PKI state also allows subsequent deployments to:

- Detect existing certificates.
- Determine whether certificates require regeneration.
- Preserve certificates that are still valid.
- Renew certificates when they approach expiry.
- Detect changes to certificate configuration such as SANs or certificate issuer.
- Rebuild or update truststores consistently.
- Maintain trust between CMP components during upgrades.

3.3.3 Requirements for upgrades and subsequent deployments

The same Ansible control server PKI directory should be made available to all subsequent deployments and upgrades of the CMP system.

Before performing an upgrade or redeployment, ensure that the PKI directory for the target system is present and accessible to the user running Ansible.

For example:

```
<pki_tls_src_files>/mdsglobal/pki/  
└─ <jdbc-server-hostname>/  
    └─ application-root-ca/  
        ├── application-root-ca.crt  
        ├── application-root-ca.key  
        ├── application-root-ca-truststore.jks  
        └─ application-root-ca-metadata.yml  
    └─ <services>/  
        ├── service.crt  
        ├── service.key  
        ├── keystore.p12  
        └─ ...
```

The exact files and directory structure may vary between CMP versions.

3.3.4 Backup and security

The PKI directory should be included in the organisation's normal backup strategy for the Ansible control server.

The Root CA private key and service private keys are sensitive security material and must be protected against unauthorised access.

Access to the PKI directory should therefore be restricted to the deployment user and authorised administrators. The private key files must not be copied to source control systems, shared repositories, or other locations where they could be accessed by unauthorised users.

When the Ansible control server is replaced or rebuilt, the existing PKI directory must be restored before performing subsequent CMP upgrades or deployments.



Do not delete or regenerate the existing Application Root CA simply to refresh or renew service certificates. Service certificate renewal and Root CA regeneration are separate operations. Root CA regeneration should only be performed as an intentional PKI lifecycle operation, as it can require all certificates and truststores issued from the previous CA to be replaced.

3.3.5 Multiple CMP system instances

Where multiple CMP system instances are deployed, each system maintains its own PKI state unless the deployment has been specifically configured to use shared PKI material.

The PKI directory associated with each system must therefore be retained and made available to the Ansible control server used for future deployments and upgrades of that system.

The PKI files are generated and consumed by the Ansible deployment process and do not need to be manually copied to the target CMP servers unless specifically required by the deployment procedure.

3.4 SABRE Server Encryption

CMP is capable of encryption of all outgoing files and decryption of all incoming files using PGP encryption following the OpenPGP standard ([RFC 4880](#)) for encrypting and decrypting data. PGP is an asymmetric algorithm meaning it requires a key pair (public and private) to support encryption and decryption. Payloads are encrypted using the public key and can only be decrypted using the matching private key along with a passphrase. There are many tools available which implement the OpenPGP standard and allow for the generation of keypairs and encryption/decryption of data using these keys. GNU Privacy Guard (GnuPG or GPG) is one such freely available and commonly used tool that you may wish to use.

When CMP is installed encryption is configured for all files and default public and private keys are installed. Those keys reside on each SABRE host in the following locations:

- Public key used for encrypting files: `/etc/mdsglobal/sabre/gpg/pubring.gpg`
- Private key used to decrypt files: `/etc/mdsglobal/sabre/gpg/secring.gpg`

The password to use in conjunction with the private key for decryption can be obtained from MDS Global Professional Services prior to installation.

Important

The default keys are designed for test purposes only. Before processing sensitive data through CMP or using CMP in production it is the responsibility of operators of CMP to update the properties in the Administration Console to use keys that have been handled with appropriate security since generation. For enhanced security different key pairs should be configured for each CMP module rather than relying on a single system-wide setting as per the default deployment. When changing key pairs, different file names should be configured rather than overwriting the default key files above, as the default files will be overwritten by a CMP upgrade.

3.5 Network Communication

3.5.1 Intra Stack Communication

It is important that components of the CMP stack can communicate with one another. The Intra Stack Communications table describes which components require access to the other components, over which port. If there are firewalls (other than the default Linux firewalls, as these are updated by the CMP installation) used between target hosts, the communication over the specified ports must be allowed by the firewall. If any of the ports are changed in the inventory file, the new value must be used when configuring firewalls.

Access must be allowed for each server in the corresponding [Ansible Source hosts group](#).

CMP Ansible Host Groups	
Group	Description
CMP Database	CMP PostgreSQL Database server.
JBoss/Wildfly Enterprise Application Platform	JBoss server instances. Required for AgentView Interface Layer, Published Interfaces Layer, Business Configuration and CMP Web Services - SOAP groups.
JBoss Web Server (JWS)/Tomcat	JBoss Web Server instances. Required for AgentView group.
Identity Server	Identity server. CMP uses WSO2 Identity Server for this purpose.
Shared Services	Shared Services application instances.
SAM Services	SAM Services application instances.
Voucher Management Services	Voucher Management Services application instances.
Role Extender	Role extender application instances.
AgentView	AgentView application instances.
Bulk Action Console	The Bulk Action console application instances.
CMP Web Services - SOAP	SOAP Web Services application instances.
CMP Web Services - REST	REST Web Services application instances.
Business Configuration	CMP Business Configuration application instances.
AgentView Interface Layer	AgentView Interfaces Layer application instances.
Published Interfaces Layer	Published Interfaces Layer application instances.
SABRE Server	SABRE Server application instances.
CMP Administration Console	CMP Administration Console application instances.
Context Sensitive Help	Context Sensitive Help application instances.
Artemis	Artemis / ActiveMQ application instances.
SPaRC Engine	SPaRC Engine application instances.
Aurora Console	Aurora Product Catalogue Console application instances.
Grafana	Grafana Monitoring application instances.
Prometheus	Prometheus Monitoring application instances.
Product Catalogue Services	Product Catalogue Services application instances.

Intra Stack Communications			
Source Hosts Groups	Destination Host Groups	Port Numbers	Protocol
JBoss/Wildfly JBoss Web Server (JWS)/Tomcat Identity Server Shared Services SAM Services Voucher Management Services Role Extender Bulk Action Console CMP Web Services - REST SABRE Server CMP Administration Console SPaRC Engine Aurora Console Grafana Prometheus Product Catalogue Services Health Check Service	CMP Database	5432	JDBC
JBoss/Wildfly JBoss Web Server (JWS)/Tomcat Role Extender CMP Web Services - REST Bulk Action Console SABRE Server CMP Administration Console	Identity Server	9443	HTTPS
JBoss/Wildfly JBoss Web Server (JWS)/Tomcat SABRE Server CMP Administration Console	Role Extender	8081	HTTPS
CMP Administration Console SABRE Server	SABRE Server CMP Administration Console	27300 27400	HTTPS

Intra Stack Communications			
Source Hosts Groups	Destination Host Groups	Port Numbers	Protocol
JBoss/Wildfly	Health Check Service	21215	HTTPS
JBoss Web Server (JWS)/Tomcat			
Identity Server			
Shared Services			
SAM Services			
Voucher Management Services			
Role Extender			
Bulk Action Console			
CMP Web Services - REST			
SABRE Server			
CMP Administration Console			
Artemis / ActiveMQ			
SPaRC Engine			



The Port Numbers indicated in the table above are defaults, and will be different if changed in the inventory configuration.

3.5.1.1 High Availability Intra Stack Communication

Communication between components in a High Availability deployment is more complicated than described in the table above. Services will communicate via the load balancer and connectivity to and from the load balancers is required, for example you must ensure the necessary ports are open to allow communication. Determining the exact network connectivity requirements in a High Availability deployment is the responsibility of the person performing the installation, based on the load balancer and shared storage configuration they have set up. See also ["Assumptions" on page 1](#) and ["High Availability Prerequisites" on page 16](#).

Even in a High Availability deployment with traffic going between components via a Load balancer, direct intra stack communication needs to be allowed as follows:

Source Host Groups	Destination Load Balancer
AgentView AgentView Interface Layer Bulk Action Console JBoss/Wildfly JBoss Web Server (JWS)/Tomcat Role Extender REST Web Services CMP Administration Console	Identify Server WSO2
AgentView Pentaho Reporting Server WSO2 IS Login	JBoss Web Server (JWS)/Tomcat
Business Configuration SOAP Web Services	JBoss/Wildfly

3.5.2 External Access

To provide user access to CMP, communication over the following ports must be allowed from the client networks. If `use_ssl` is selected in the Inventory Configuration Tool, the HTTPS ports listed need to be allowed.

Target Host Groups	HTTPS Port
JBoss/Wildfly (hosting AgentView, WSO2 Login SSO)	7443
JBoss Web Server (JWS)/Tomcat (hosting Business Configuration, SOAP etc.)	8443
Identity Server	9443
Bulk Action Console	9009
CMP Web Services - REST	9000
SABRE Server	21212
CMP Administration Console	31212
Artemis / ActiveMQ	8161
Aurora Console	9010



External access will also be required for all incoming and outgoing communication from Sabre Jobs that has been deployed as part of the customer specific CMP installation, for example, event records, sending email, reading from external queues. The network requirements in this area need to be determined as part of the CMP implementation project and are outside the scope of this document.

If any of the ports are changed in the inventory file then access needs to be allowed to the newly specified ports instead.



In addition to the SSH access described earlier, the control server also needs to have the above “External Access” to the target hosts.

4.0 Pre-Installation Tasks

Before you install CMP:

- Ensure that the prerequisites are in place (see ["Prerequisites" on page 13](#)).
- Ensure that you are familiar with CMP Ansible automatic deployment concepts.

See ["About CMP Installation" on page 2](#).

- In the case of a CMP version upgrade, ensure that any associated third party upgrade steps, for example for the PostgreSQL database, are completed as described in the release notes of the software version in question.
- Make a copy of the current Virtual Machine(s) (or snapshot) prior to installation/upgrade. This allows for the environment to be restored in the event that a roll-back is required.
- Ensure that these CMP System users and their uid / gid are free on the relevant systems, that is, none of these are already taken by other users.

User Name	UID	GID
wso2is	200	200
sabre	201	201
rolex	202	202
restws	203	203
logcollector	204	204
bulkaction	205	205
healthcheck	206	206
sharedservices	207	207
contextsensitivehelp	208	208
sparc	209	209
artemis	210	210
jboss	211	211
tomcat	212	212
prometheus	213	213
grafana	214	214
mimir	215	215
aurora	216	216

- Download the CMP Ansible installation files from the CMP installation repository in the cloud to the control server:

The files are in a zip package that stores the Ansible deployment playbooks, inventories and other files:

1. Download the Ansible Playbook package using the `curl` command for the release version being installed using the credentials supplied by MDS Global:

where:

<username> is the username in credentials supplied by MDS Global

<cmp version> is the CMP version to be installed, for example, 8.10 or 8.12.

2. Unzip the downloaded package. For example:

```
unzip cmp-ansible-playbook-pkg.zip
```



Important: The latest Ansible Playbook for the release being installed must always be downloaded prior to running the installation process. It is critical that the playbook version corresponds to the version of CMP being installed.

5.0 Installation Tasks

To install CMP, you must:

1. [Prepare the Inventory File using the Inventory Configuration Tool](#)
2. [Pre-Deploy CMP](#)
3. [Deploy CMP](#)
4. [Upgrade an Existing Installation](#)

Important

The Inventory File used for CMP installation must be generated from the release specific version of the online Inventory Configuration Tool. MDS Global does not support installations using manually created or manually modified Inventory Files due to the complications that this causes in trying to identify any syntax or other issues with the content.

5.1 Installation Configuration Tool

The Installation Configuration Tool is a web-based tool that enables you to provide values for the properties in the inventory template file and produces:

- An inventory file - `<filename>.yaml`.
- A environment state file - `<yaml filename>.state.json`.

The environment state file is a JSON file that is a snapshot of the inventory file that is being produced. It is created when you create an inventory file and then uploaded to the tool when you upload that inventory file. The environment state file is only required for advanced use of the installer and can be ignored unless MDS Global provides specific instruction to the contrary.

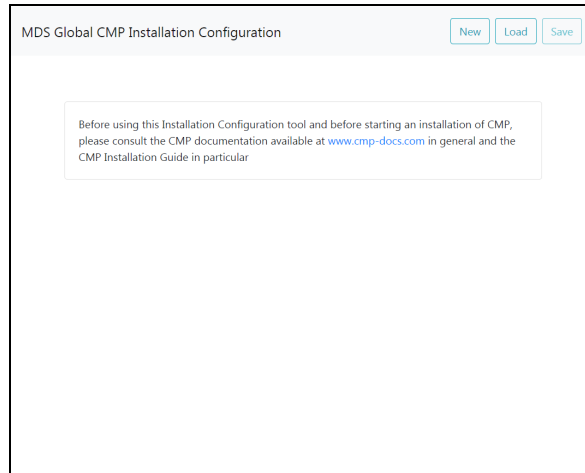
5.1.1 Preparing an Inventory File with the Installation Configuration Tool

To prepare an inventory file using the Installation Configuration Tool, follow these steps:

1. Access the tool via the URL: <https://inventory-config.cmp-docs.com>.

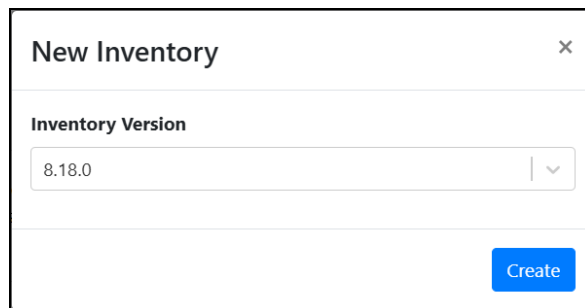
The tool is password protected. Use the same cmp-docs.com Google account used to access the CMP Documentation set.

The tool opens in the browser.



2. To create a new inventory file, click **New**.

The **New Inventory** dialog opens.

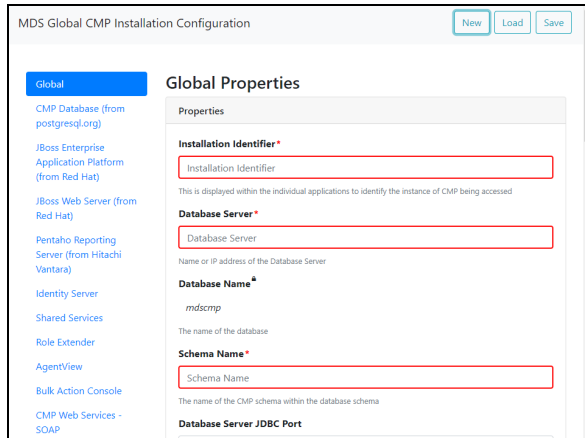


3. Select the CMP release version for which to create an inventory file from the Inventory Version drop-down list, and then select *Create*.

The MDS Global CMP Installation Configuration tool opens. On the left is a navigation pane with links to configure each of the components of CMP, for example **CMP Database**, **JBoss Enterprise** or **AgentView**. The list of components reflects the table of **CMP Groups**.

On the right are the properties, for which you can enter values.

4. Define the **Global Properties**.



MDS Global CMP Installation Configuration

Global

CMP Database (from postgresql.org)

JBoss Enterprise Application Platform (from Red Hat)

JBoss Web Server (from Red Hat)

Pentaho Reporting Server (from Hitachi Vantara)

Identity Server

Shared Services

Role Extender

AgentView

Bulk Action Console

CMP Web Services - SOAP

Global Properties

Properties

Installation Identifier*

Installation Identifier

This is displayed within the individual applications to identify the instance of CMP being accessed

Database Server*

Database Server

Name or IP address of the Database Server

Database Name*

mdscmp

The name of the database

Schema Name*

Schema Name

The name of the CMP schema within the database schema

Database Server JDBC Port

For a description of the properties for each component, see the relevant topic in this section. The explanatory on screen text will also help you. Default property values are pre-filled.

The colours of the fields are also a guide:

Mandatory property fields are highlighted in red:

Database Server

A property field that has the focus is highlighted in blue:

Database Server

A property field for which you have entered a value is highlighted in green:

cmp-gorilla-test1.example.com

A hardcoded value is highlighted in black:

Database Name

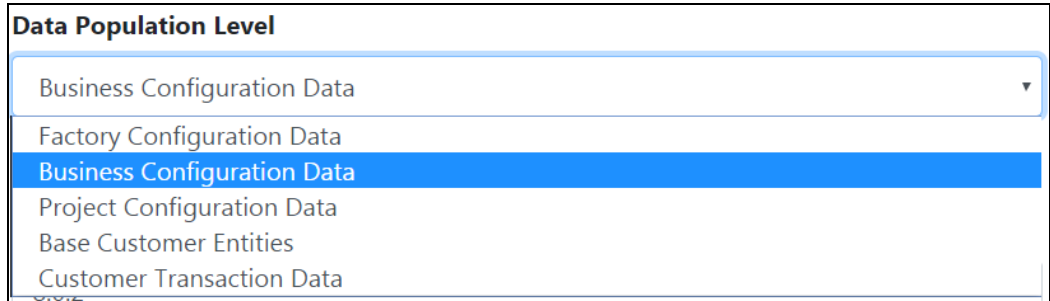
mdscmp

The name of the database

- Enter the property values for the component.
Enter a value for a property as follows:
 - For a text value, such as a hostname, database name or URL, enter the text in the field.

cmp-gorilla-test1.example.com

- To choose from a set of predefined (enum) values, select from a drop-down list.



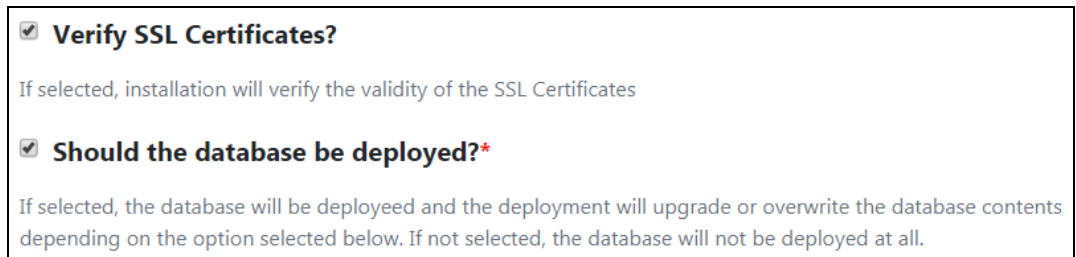
Data Population Level

- Business Configuration Data
- Factory Configuration Data
- Business Configuration Data**
- Project Configuration Data
- Base Customer Entities
- Customer Transaction Data



A typical production installation will require a data population level of Business Configuration Data to be configured.

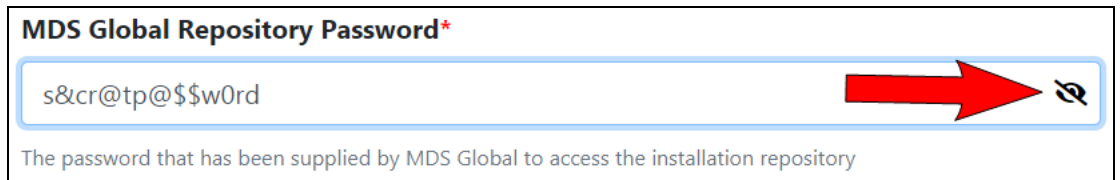
- To set a property to true/false or enable/disable, select or deselect a checkbox.



☒ **Verify SSL Certificates?**
If selected, installation will verify the validity of the SSL Certificates

☒ **Should the database be deployed?***
If selected, the database will be deployed and the deployment will upgrade or overwrite the database contents depending on the option selected below. If not selected, the database will not be deployed at all.

- When entering a masked property, such as a password, you can toggle the mask/unmask icon to reveal and check your entry.

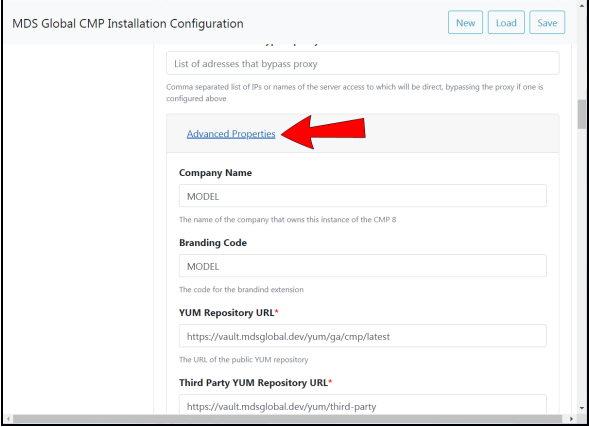


MDS Global Repository Password*

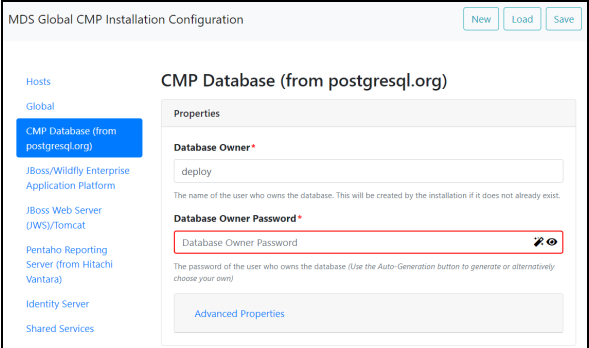
s&cr@tp@\$w0rd

The password that has been supplied by MDS Global to access the installation repository

6. Click the **Advanced Properties** link at the bottom of the **Global Properties** screen to expand the **Advanced Properties** list to update the advanced properties.



7. Once you have completed the **Global Properties** screen, click the **Hosts** link on the left-hand pane.
8. Enter the host name on which the selected CMP component will be installed (selected from the Software drop-down list to the right of the Host field).
You must specify at least one host per CMP component. To remove a host, click the corresponding **Delete**  icon.
9. Next, click the **CMP Database** component link on the left-hand pane to add the relevant properties.



10. Repeat the steps to enter values for all CMP components until everything is configured.
11. Select AgentView from the left-hand menu. Select Advanced Properties to configure the branding. The standard CMP font and branding is deployed by default but this can be overwritten by changing the default font, colour, background images, and so on, and/or by uploading customer specific branding.
12. Click **Save**.

The **Download Inventory** window is displayed.

13. Enter a name for the inventory file.

Download Inventory ×

File Name*

cmpinventory

The name of the inventory .yaml file. The corresponding state file will be stored with the name <filename>.state.json

Vault Password

Vault Password 

Enter the Vault Password if you would like to encrypt sensitive properties. Leave blank to save inventory unencrypted.

Save

The inventory file will be saved as <filename>.yaml. The corresponding environment state file will be saved as <filename>.state.json.

14. To encrypt properties in the file, enter the vault file password.
15. Click **Save**.

The file is validated. An error message will alert you to any issues.

Download Inventory ×

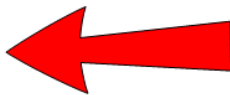
File Name*

The name of the inventory .yaml file. The corresponding state file will be stored with the name <filename>.state.json

Vault Password

 👁

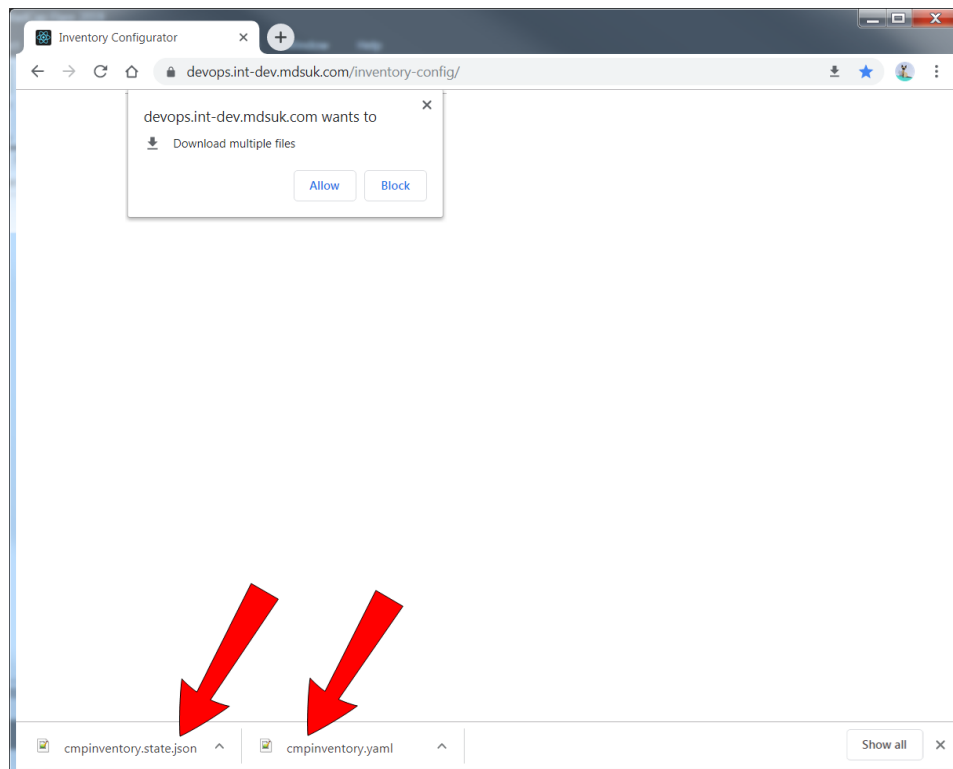
Enter the Vault Password if you would like to encrypt sensitive properties. Leave blank to save inventory unencrypted.

At least one host must be specified in group aview_iface 

Save

16. Fix any issues and save the file.

The files are saved to your Downloads folder.

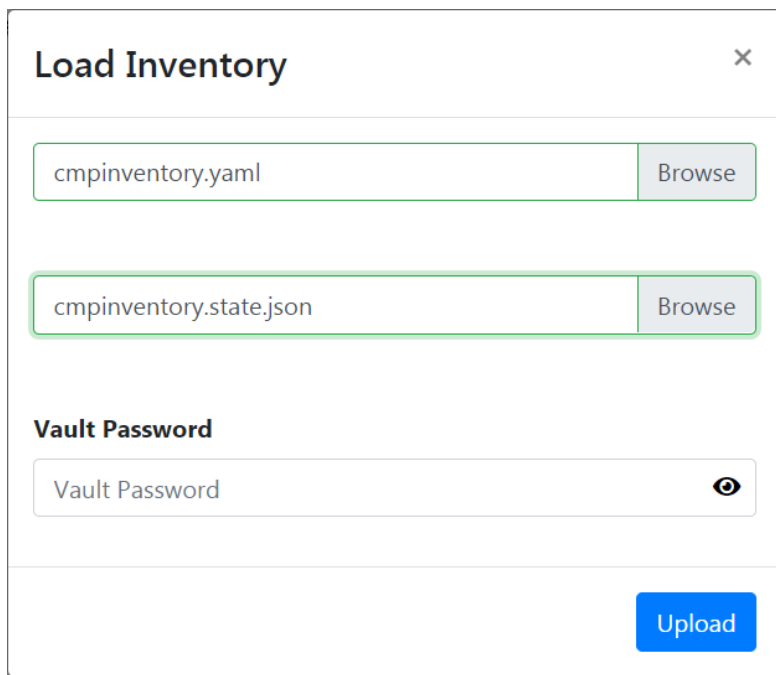


17. The final step is to copy the inventory file to the Ansible control server and use it as part of the installation process, following the instructions in ["Deploy CMP" on page 44](#).

5. 1. 2 Working with an Existing Inventory File

To upload and edit an inventory file that was created previously with the tool, follow these steps:

1. Open the Installation Configuration Tool and click **Load**.
The **Load Inventory** window is displayed.
2. To **Select an Inventory File**, click the corresponding **Browse** button.
3. Navigate to the inventory file and select it.
4. Select the corresponding environment state file.



5. If the file is encrypted, enter the **Vault Password**.
6. Click **Upload**.

The file is uploaded and the tool uses the environment state file to detect whether the inventory file has been changed outside of the tool or whether it differs from the model embedded in the tool. Property values that are different are highlighted in yellow:

JBoss HTTPS Port (if SSL is in use)

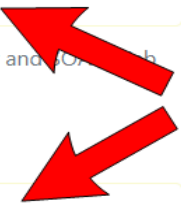
8444

The port that Business Configuration, AgentView Interfaces Layer, Published Interfaces Layer and SOAP Web Services (JBoss applications) will listen for HTTPS connections on if SSL is in use

JBoss HTTP Port(if SSL not in use)

8081

The port that Business Configuration, AgentView Interfaces Layer, Published Interfaces Layer and SOAP Web Services (JBoss applications) will listen for HTTP connections on if SSL is not in use



7. Edit the inventory by entering property values as described in ["Preparing an Inventory File with the Installation Configuration Tool" on page 35](#)
8. Save your changes.

5.2 Pre-Deploy CMP

Before you install CMP, there are a number of steps that must be completed to ensure a smooth deployment.

These steps are managed using the `pre-deploy.yaml` file, and must always be executed before a full installation using the `deploy.yaml`, as it is required to pre-configure certain aspects of the system and run various validations of the overall system readiness for a deployment to take place.

Multiple Ansible playbooks are run when the pre-deploy is executed, such as *CMP System Deployment Pre Checks on the Inventory*, *CMP System Deployment Pre Checks on the Hosts*, and *Generate the required CMP System users and groups*, amongst others.

 The `pre-deploy.yaml` can be executed at any time before the `deploy.yaml` is run, meaning it could be executed multiple times, hours or even days before a full deployment is performed.

Once the `pre-deploy.yaml` has been executed, the ansible tasks to deploy CMP can be run.

 When the pre-deploy is executed, based on the option selected from within the Inventory Configuration Tool - CMP Repository Source, if this set to **Cached** it will configure an Apache HTTP Web Server and sync down all required external repositories onto a locally configured system. For more on caching, see [Install Cached version](#).

5.3 Install Cached version

You can improve the performance and robustness of a CMP 8 installation or upgrade by running the CMP 8 installer in a pre-install mode. In doing so, the Generate Repo Host task prepares the environment for local caching prior to the main installation or upgrade. The Apache web service is installed and configured, and the necessary repository structure is created. This allows the artefacts from the CMP repositories and the required third party repositories to be downloaded locally before the CMP 8 installation.

There is an Ansible installer option allowing you to install CMP using a cached version of the required Red Hat Package Manager. This allows for a deployment to happen without ever needing to contact external systems and go outside the network.

This is facilitated by splitting the deployment into two parts, `pre-deploy.yaml` which is run before an installation, and `deploy.yaml` for the system install.

 The `pre-deploy.yaml` can be executed at any time before the `deploy.yaml` is run, meaning it could be executed multiple times, hours or even days before a full deployment is performed.

When the pre-deploy is executed, based on the option selected from within the Inventory Configuration Tool - CMP Repository Source, if this set to **Cached** it will configure an Apache HTTP Web Server and sync down all required external repositories onto a locally configured system. The first time this is run can take some time as it syncs everything down, but subsequent executions will only sync any changes and run much quicker. The main deploy will then install the required rpms from this local location instead of making calls to external repositories. The main advantage of this is to prevent installer failures due to unknown external influences when repositories can't be reached or other external network related connectivity issues.

 If the option selected from within the Inventory Configuration Tool - CMP Repository Source is set to **From Source**, then the installation will continue to install components from the external repositories as previously.

For the hosting of the internal cached repositories, we recommend using the Ansible controller node as this will already be present and negates the need for setting up another additional box. However, there is the option to use any system you choose.

To host these cached repositories, it will require additional disk space on the chosen box, approximate sizes for the repositories are 70GB for Rocky Linux and 220GB for RedHat on the initial setup. These will grow over time, so allow for this in your configuration.

5.4 Deploy CMP

To deploy the CMP stack to the target system, run the following command from the directory that the Ansible Playbook has unzipped from:

```
ansible-playbook --ask-vault-password-file=<vault_password_file>
-i <inventory_file> -b playbooks/deploy.yaml
```

where <inventory file> is the file that has been generated by the Installation Configuration Tool.

See "[Common Ansible Command Line Parameters](#)" below to get details about combinations of the command line parameters that can be used.

5. 4. 1 Common Ansible Command Line Parameters

The `ansible-playbook` is quite flexible, allowing users to control the way it accesses the target environment. Depending on the deployment scenario, a combination of the command line parameters that are described in the following table can be used:

Parameter	Description	Example
-b	Runs all the tasks in the play with elevated access (using sudo).  This is mandatory parameter. It must always be used when running Ansible playbooks.	-b
-k	Prompt user for connection password. If specified, the user will be prompted to enter the password for the user specified in the <code>ansible_ssh_user</code> parameter	-k
-K	Prompt user for the sudo password. If specified, user will be prompted to enter the remote username password. Usually it is defined by the <code>ansible_ssh_user</code> parameter. Can be required when the user in question requires a password to gain the sudo access but the SSH private key is used to access the remote system.	-K
--ask-vault-pass	Prompt user for the vault password. If specified, the user will be prompted to enter the vault password.	--ask-vault-pass
--vault-password-file	Points Ansible command to the vault password file.	--vault-password-file=~/.vaultpwd
-i	Points Ansible to the inventory file.	-i inventory/test.yaml

Rather than deploying all CMP components in one go as described above, you can deploy individual components of the system using the playbooks files stored in the `playbooks` directory. The `deploy.yaml` playbook referenced in the overall installation above simply orchestrates these component-specific playbooks.



Important: Execution of lower level playbooks individually rather than using the orchestration in `deploy.yaml` can result in system misconfiguration if not done correctly. Zero downtime upgrade of a High Availability installation is also not possible by executing playbooks individually. Therefore only the



use of `deploy.yaml` is supported for production environments unless MDS Global gives explicit instruction to execute a specific lower level playbook.

If the playbooks are executed individually, it is important to note that they have the following dependencies:

- Three components must be installed in order:
 1. Database (`deploy-db.yaml`)
 2. WSO2IS (`deploy-wso2is.yaml`)
 3. Role-extender (`deploy-role-extender.yaml`)
- JBoss must be deployed prior to the following: agent-view-installation-layer, published-interfaces-layer, SOAP WS and configuration-centre.
- JWS must be deployed prior to agent-view and wso2is-login.
- Once JBoss and JWS are deployed, then the dependant components and any remaining components can be deployed in any order.

This table describes the deployment playbooks and summarises the dependencies between them:

Playbook Name	Depends On	Description
deploy-db.yaml		Deploys the postgres data-base
deploy-wso2is.yaml	deploy-db.yaml	Deploys the WSO2 Identity Server
deploy-role-extender.yaml	deploy-db.yaml deploy-wso2is.yaml	Deploys the Role Extender Application
deploy-health-check.yaml	deploy-db.yaml	Deploys the Health Check Service
deploy-sam-services.yaml	deploy-db.yaml	Deploys the SAM Services
deploy-voucher-management-services.yaml	deploy-db.yaml	Deploys the Voucher Management Services
deploy-shared-services.yaml	deploy-db.yaml	Deploys the Shared Services
deploy-jboss.yaml contains deployment roles for: agent-view-interfaces-layer pil configuration-centre soap-ws	deploy-db.yaml deploy-wso2is.yaml deploy-role-extender.yaml	Deploys the JBoss/Wildfly Application Server along with AgentView Interfaces Layer, Published Interfaces Layer, Configuration Centre and SOAP Web Services
deploy-jws.yaml contains deployments roles for: wso2is-login agent-view	deploy-db.yaml deploy-wso2is.yaml deploy-role-extender.yaml deploy-jboss.yaml	Deploys the JBoss/Tomcat Web Server along with WSO2 Identity Server Custom Login

Playbook Name	Depends On	Description
		application for SSO and AgentView
deploy-rest-ws.yaml	deploy-db.yaml deploy-wso2is.yaml deploy-role-extender.yaml	Deploy REST Web Services
deploy-bulk-action-console.yaml	deploy-db.yaml deploy-wso2is.yaml deploy-role-extender.yaml	Deploys the Bulk Action Console Application
deploy-context-sensitive-help.yaml		Deploys the Context Sensitive Help Application
deploy-artemis.yaml		Deploys the Artemis / ActiveMQ Application
deploy-sparc-engine.yaml	deploy-db.yaml deploy-artemis.yaml	Deploys the SPaRC Engine Application
deploy-sabre-server.yaml	deploy-db.yaml deploy-wso2is.yaml deploy-role-extender.yaml deploy-artemis.yaml	Deploy the Sabre Server Application
deploy-sabre-console.yaml	deploy-db.yaml deploy-wso2is.yaml deploy-role-extender.yaml deploy-artemis.yaml deploy-sabre-server.yaml deploy-licensing-post.yaml deploy-node-exporter.yaml deploy-grafana.yaml deploy-prometheus.yaml deploy-product-catalog-rest.yaml	Deploys the Administration Console Application

Playbook Name	Depends On	Description
	deploy-aurora-console.yaml	

The individual playbooks can be executed independently, providing that the playbooks they depend upon have already been executed.

5.4.2 Summary File

Once the `deploy.yaml` script has successfully completed execution, a new file will be produced. The file outlines the summary that includes the URLs of the installed components as well as location of the component log files. The summary filename matches the name of the inventory file used (without the `.yaml` extension) with the suffix, `summary.txt`.

6.0 Upgrading an Existing Installation

Prerequisites:

- Check in the release notes of the new CMP version that an upgrade path from the currently installed version exists. If there is no direct upgrade path then it will be necessary to perform a series of upgrades following defined upgrade paths, for example, if 8.10 - 8.12 is not supported so an 8.10 - 8.11 upgrade would need to be performed, followed by an 8.11 - 8.12 upgrade.
- If the release notes for the target release state that PostgreSQL or Linux major versions need to be upgraded, then complete those upgrades prior to starting the installation.
- Ensure that the Ansible version on the control server is aligned to the requirements of the target CMP release.
- Complete the pre-installation tasks as described in the release note of the CMP version to be upgraded to.
- Review the *Installation Configuration Tool* section of this guide to identify new parameters, or changes to existing parameters, that you may need for your upgrade.
- Update the original inventory file used for the last installation or upgrade of the target system using the Installation Configuration tool online at <https://inventory-config.cmp-docs.com>:
 1. Open the Installation Configuration Tool and select the CMP version to be upgraded to
 2. Click on load and select the original inventory file used for the last installation or upgrade of the target system
 3. Acknowledge the warning that the CMP version to be installed by the inventory file is being updated
 4. Make any changes required to the properties to be used for the installation paying particular attention to update values for new mandatory properties added since the previous release version

To upgrade an existing CMP installation, use the same `ansible-playbook` command to deploy a new installation:

```
ansible-playbook --ask-vault-password-file=<vault_password_file>  
-i <inventory_file> -b playbooks/deploy.yaml
```

The `playbooks/deploy.yaml` file can be executed multiple times. At each run the playbooks will:

- Deploy the latest CMP RPMs
- Recreate the necessary configuration file

It is safe to run the `deploy.yaml` playbook multiple times as long as the principal properties in the inventory file used during the execution are not modified. The principal properties of the inventory file are:

- Group hosts
- Ports allocated to each component
- Credentials used within the CMP components

The group hosts must not be modified under any circumstances (except where adding additional hosts). If there is a need to change a host in the group, the CMP stack must currently be uninstalled first using the old inventory file. After that the hosts can be changed and the `deploy.yaml` playbook executed again with the modified inventory file.

Ports in theory can be changed, however the old ports must be excluded from the firewall configuration (if firewall is enabled on the target host).

Database credentials can be changed in the inventory file only to reflect manual changes. For example, if an administrator changes a database user password, the new password in question must be modified in the inventory file prior to the next upgrade.

The same applies to the application credentials.

6.1 Certificate Updates

In a similar way to running an upgrade, you can also run a deployment to ONLY update the SSL Certificates installed in the various backend systems. To do this, first make sure the new certificates are in the correct location as defined in your inventory file under the `ssl_certificates` section.

Then use the same `ansible-playbook` command as previously but adding to the end of it `--tags ssl_configuration -e ssl_update=true` as shown below:

```
ansible-playbook --vault-password-file=<vault_password_file> -i <inventory_file> -b playbooks/deploy.yaml --tags ssl_configuration -e ssl_update=true
```

6.2 Patch Updating

There is an option to run a deployment to ONLY patch update the installed CMP components where necessary. This option uses the Ansible tags in order to skip running unnecessary tasks and run only those tasks required for updating the RPMs that may have been patched for the CMP system.

This method runs through the playbook, installs only those RPMs that have been patched/updated and performs a service restart, thereby reducing downtime by affecting only those updated components.

When you are ready to patch update, first ensure you read and adhere to the above notes on updating an existing installation (although unless explicitly stated in the patch

release note, a patch will not require any change to the inventory file that was used for the last installation) and then use the same ansible-playbook command as previously but adding to the end of it `-e patch_update=true --tags installation` as shown in the following example:

```
ansible-playbook --ask-vault-password-file=<vault_password_file>  
-i <inventory_file> -b playbooks/deploy.yaml -e patch_update=true  
--tags installation
```

6.0 Copying Database-Based Configuration

It is possible to copy database-based configuration, which is stored in a database rather than in property files, from one installation of CMP to another.

This is typically used in testing environments, but it can also be used to copy CMP data between installations. Copying a database-based configuration ensures consistency across environments, and saves time by providing a ready-to-use system with existing records.

Before you begin, ensure that:

- The source and target versions of CMP are the same, for example, if the source installation is V 8.15.0 Patch Level 2, the target installation must also be V 8.15.0 Patch Level 2.

You can also compare the module versions of the database in the Admin Console in *System Configuration->Modules->Database*. The versions for the **cmp-data-base-pkg** should be identical.

- You have a mechanism for transferring a file from the source installation's database machine to the target installation's database machine, such as SSH.
- You are able to log onto both the source and target databases using a profile with sufficient database privileges to read and write data to all tables within the database and run procedures.

To copy the configuration data from one installation to another, follow these steps:

1. Run the `SAVE_CMP_CONFIG_DATA( level_param VARCHAR, file_path VARCHAR)` function. This function will exist in the same schema as the CMP tables.

`<level_param>` can have one of four values, namely:

- a. `Factory_Configuration`
- b. `System_Configuration`
- c. `Business_Configuration`
- d. `Project_Configuration`

These levels specify the different levels of data that you are copying. `Factory_Configuration` is the most basic CMP config, and `Project_Configuration` is the most complex.

`<file_path>` is a fully qualified name, including path, to a text file that the function will create. This file should not already exist.

2. Transfer the file specified in `<file_path>` to a location accessible to the target database box.

3. Run the `LOAD_CMP_CONFIG_DATA (file_path TEXT)` function.

This function will exist in the same schema as the CMP tables in the target database. The `<file_path>` parameter specifies the fully qualified name of the file produced by the `SAVE_CMP_CONFIG_DATA` function and should refer to the location where the file was transferred in step 2.

This function does not raise an error if the data being copied already exists in the target tables, that is, a PK Key conflict.

7.0 Troubleshooting

Typically, if anything is wrong with the host environment, the playbook stops running and the installation fails.

Ansible produces a status report that you can consult for troubleshooting. The lines in the status report are colour-coded as follows:

- **Yellow**

Yellow text indicates that Ansible made changes to the target host. For example:

```

ansible@alpha22:~/cmp-8-deployment-ga
TASK [postgres : Create symbolic link to the pg_wal directory] *****
Thursday 30 May 2019  09:10:30 +0100 (0:00:00.095)          0:02:00.210 *****
skipping: [alpha24.mdsuk.com]

TASK [postgres : Copy the modified postgresql service unit file] *****
Thursday 30 May 2019  09:10:30 +0100 (0:00:00.097)          0:02:00.308 *****
changed: [alpha24.mdsuk.com]

TASK [postgres : Reload the service data after reconfiguring service unit] *****
Thursday 30 May 2019  09:10:31 +0100 (0:00:00.911)          0:02:01.219 *****
changed: [alpha24.mdsuk.com]

TASK [postgres : Start PostgreSQL service] *****
Thursday 30 May 2019  09:10:32 +0100 (0:00:00.645)          0:02:01.864 *****
ok: [alpha24.mdsuk.com]

TASK [postgres : Create database configuration script] *****
Thursday 30 May 2019  09:10:32 +0100 (0:00:00.544)          0:02:02.409 *****
changed: [alpha24.mdsuk.com]

```

- **Blue**

Blue text indicates that the task was not executed because the condition attached to the file was false. For example:

ansible@alpha22:~/cmp-8-deployment-ga

```

TASK [postgres : Create symbolic link to the pg_wal directory] *****
Thursday 30 May 2019  09:10:30 +0100 (0:00:00.095)      0:02:00.210 *****
skipping: [alpha24.mdsuk.com]

TASK [postgres : Copy the modified postgresql service unit file] *****
Thursday 30 May 2019  09:10:30 +0100 (0:00:00.097)      0:02:00.308 *****
changed: [alpha24.mdsuk.com]

TASK [postgres : Reload the service data after reconfiguring service unit] *****
Thursday 30 May 2019  09:10:31 +0100 (0:00:00.911)      0:02:01.219 *****
changed: [alpha24.mdsuk.com]

TASK [postgres : Start PostgreSQL service] *****
Thursday 30 May 2019  09:10:32 +0100 (0:00:00.645)      0:02:01.864 *****
ok: [alpha24.mdsuk.com]

TASK [postgres : Create database configuration script] *****
Thursday 30 May 2019  09:10:32 +0100 (0:00:00.544)      0:02:02.409 *****
changed: [alpha24.mdsuk.com]

```

If the task that caused the error is followed by a blue line '**...ignoring**', this means that task in question has the `ignore_errors` property set to `true`, so even in the case of an error, playbook execution will not be interrupted. This is normal execution for the CMP installation.

- **Green**

Green text indicates that the task was executed but the state of the target host did not change. For example:

 ansible@alpha22:~/cmp-8-deployment-ga

```
TASK [postgres : Create symbolic link to the pg_wal directory] *****
Thursday 30 May 2019  09:10:30 +0100 (0:00:00.095)      0:02:00.210 *****
skipping: [alpha24.mdsuk.com]

TASK [postgres : Copy the modified postgresql service unit file] *****
Thursday 30 May 2019  09:10:30 +0100 (0:00:00.097)      0:02:00.308 *****
changed: [alpha24.mdsuk.com]

TASK [postgres : Reload the service data after reconfiguring service unit] *****
Thursday 30 May 2019  09:10:31 +0100 (0:00:00.911)      0:02:01.219 *****
changed: [alpha24.mdsuk.com]

TASK [postgres : Start PostgreSQL service] *****
Thursday 30 May 2019  09:10:32 +0100 (0:00:00.645)      0:02:01.864 *****
ok: [alpha24.mdsuk.com]

TASK [postgres : Create database configuration script] *****
Thursday 30 May 2019  09:10:32 +0100 (0:00:00.544)      0:02:02.409 *****
changed: [alpha24.mdsuk.com]
```

- Red

Red text indicates that there was an error. For example:

```

TASK [firewall-rule : Check if firewalld is running] *****
Thursday 30 May 2019  09:19:00 +0100 (0:00:00.088)      0:01:27.672 *****
fatal: [alpha24.mdsuk.com]: FAILED! => (
  "changed": true,
  "cmd": "service firewalld status",
  "delta": "0:00:00.020468",
  "end": "2019-05-30 09:19:00.913781",
  "rc": 3,
  "start": "2019-05-30 09:19:00.893313"
)

STDOUT:

• firewall.service - firewalld - dynamic firewall daemon
  Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset: enabled)
  Active: inactive (dead)
  Docs: man:firewalld(1)

STDERR:

Redirecting to /bin/systemctl status firewalld.service

MSG:

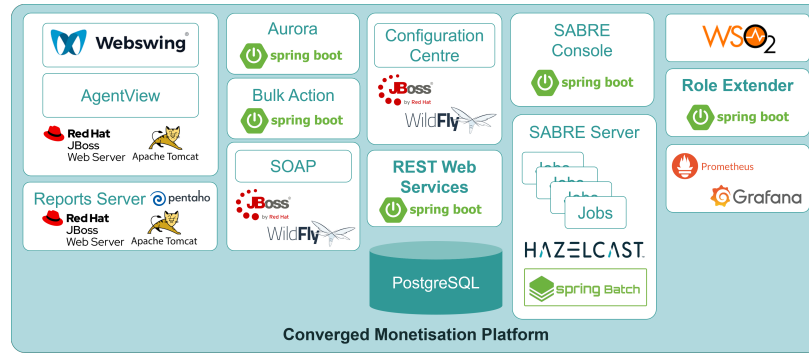
non-zero return code

```

Some tasks can fail by design; but this is normal as the error is ignored by the play.

Once the deployment has completed, if CMP does not work as expected, you need to go through the application logs. The location of the component logs is in the summary file produced by the Ansible play. For more information, see ["Summary File" on page 49](#)

As part of troubleshooting, it may be necessary to examine the CMP technical architecture to determine the components that each part of the system depends on. For example, if AgentView doesn't work then the AgentView Interfaces layer, WSO2 and the database are all potential sources of error depending on exactly where the application problem is seen. For more information, see the *CMP Technical Architecture Guide*.



Examining the technical architecture of CMP can help to troubleshoot an installation

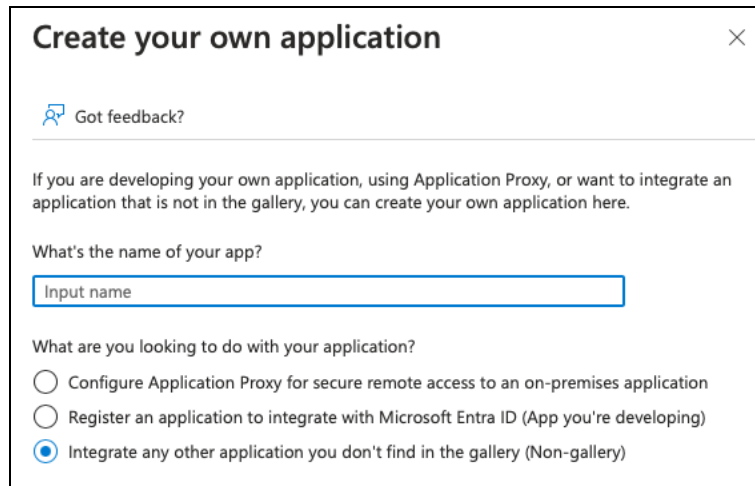
8.0 Configure Azure Active Directory as a SAML Federated Identity Provider in WSO2

The following steps describe how to integrate Azure Active Directory with WSO2 as a SAML Federated Identity Provider, giving CMP 8 SAML SSO (Single Sign-On) capabilities via Azure Active Directory.

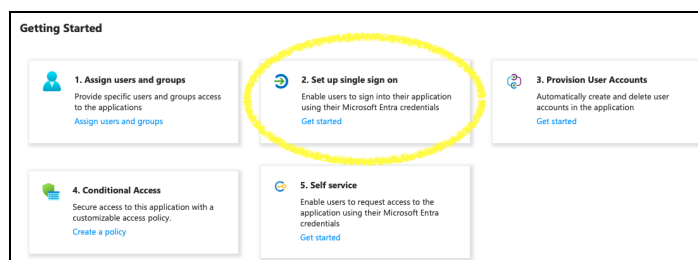
1. Set up Azure Active Directory

The first step is to configure an application in Azure to communicate with the WSO2 server during the SAML-based federation.

1. Log in to the [Microsoft Azure management portal](#) with your existing Office 365 admin credentials.
2. From the Microsoft Azure Dashboard, navigate to *Enterprise Applications* and select **New Application**.
3. Select **Create your own application**.



4. Enter a name for your application, for example, *CMP 8*, select *Integrate any other application you don't find in the gallery (Non-gallery)*, and then select **Create**.
5. When the application has been created, select *Get Started* from the **Set up single sign-on** menu option.



6. Select *SAML* as the single sign-on method.

You are redirected to the SAML-based Sign-on page.

7. Add the following values in the Basic SAML Configuration section, then save and exit.

- Identifier (Entity ID): Enter A Unique Entity Identifier (e.g. cmp8wso2saml)
- Reply URL (Assertion Consumer Service URL): https://<wso2_base_url>/commonauth
- Sign-on URL: https://<wso2_base_url>/samlso
- Logout Url: https://<wso2_base_url>/samlso

Your configuration for this application should look like the one below (you may need to refresh the page to view the new configurations).

Basic SAML Configuration		 Edit
Identifier (Entity ID)	cmp8wso2saml	
Reply URL (Assertion Consumer Service URL)	https://wso2.greenhouse.mdsglobal.dev/commonauth	
Sign on URL	https://wso2.greenhouse.mdsglobal.dev/samlso	
Relay State (Optional)	Optional	
Logout Url (Optional)	https://wso2.greenhouse.mdsglobal.dev/samlso	

Next, configure the claims which will be passed in the SAML response to APIM Server during the federation. A new claim must be added to support application roles.

8. Select **Edit** in *Attributes & Claims* and select *Add new claim*.
9. Add the following values and then save your changes:
 - Name: memberOf
 - Namespace: <http://schemas.xmlsoap.org/ws/2005/05/identity/claims>
 - Value: user.assignedroles

Home > CMP 8 | SAML-based Sign-on > SAML-based Sign-on > Attributes & Claims >

Manage claim

 Save
  Discard changes
  Got feedback?

Name *

Namespace

Choose name format

Source * ☒ Attribute ☐ Transformation ☐ Directory schema extension

Source attribute *

Claim conditions

Advanced SAML claims options

- Download the raw certificate and save to a location for later use when setting up the Federated IDP in WSO2.

3 SAML Certificates

Token signing certificate  Edit

Status	Active
Thumbprint	BBA8C183C9812C344DE2595A98ABC5E41B8B0C63
Expiration	05/07/2027, 09:56:20
Notification Email	ken.casey@outlook.ie
App Federation Metadata Url	https://login.microsoftonline.com/7f57cca3-31a2-... 
Certificate (Base64)	Download
Certificate (Raw)	Download
Federation Metadata XML	Download

Verification certificates (optional) 

Required	No
Active	0
Expired	0

- From the Azure dashboard, navigate to *App Registrations > All applications*, and select the app that was created in the above step.

In order to conform with the role based authorisation within CMP 8, application roles to be used must be configured in Azure for each of the CMP 8 security groups.

- Select *Manage > App Roles* and create a new role for each of the security groups defined in the documentation [here](#). Default app roles can be removed if desired, but this is not necessary.

Home > App registrations > CMP 8

CMP 8 | App roles

Search + Create app role Got feedback?

Overview
Quickstart
Integration assistant
Diagnose and solve problems
Manage
Branding & properties
Authentication
Certificates & secrets
Token configuration
API permissions
Expose an API
App roles
Owners
Roles and administrators
Manifest
Support + Troubleshooting

App roles
App roles are custom roles to assign permissions to users or apps. The application defines and publishes the app roles and interprets them as permissions during authorization.

How do I assign App roles

Display name	Description	Allowed member ty...	Value	ID	State
agent-view-backoffice	agent-view-backoffice	Users/Groups	agent-view-backoffice	73f26b5a-14ea-4651-...	Enabled
agent-view-consumer-csa	agent-view-consumer-csa	Users/Groups	agent-view-consumer...	8b0d4229-979a-41fa-...	Enabled
agent-view-consumer-user	agent-view-consumer-user	Users/Groups	agent-view-consumer...	57bc5149-ae49-4f67-a...	Enabled
agent-view-enterprise-admin	agent-view-enterprise-admin	Users/Groups	agent-view-enterprise...	884ad92c-8ed2-40a0-...	Enabled
agent-view-enterprise-csa	agent-view-enterprise-csa	Users/Groups	agent-view-enterprise...	5100dae4-23ed-4eb0-...	Enabled
agent-view-enterprise-user	agent-view-enterprise-user	Users/Groups	agent-view-enterprise...	4deb72b4-0ada-4681-...	Enabled
agent-view-manager	agent-view-manager	Users/Groups	agent-view-manager	bdad67c2-5375-46c4-...	Enabled
configuration_centre_read_only	configuration_centre_read_only	Users/Groups	configuration_centre_r...	fc35bc3e-c4f5-4ff6-b3...	Enabled
configuration_centre_admin	configuration_centre_admin	Users/Groups	configuration_centre_a...	627d2320-d270-4eb4-...	Enabled
config_centre_admin_user	config_centre_admin_user	Users/Groups	config_centre_admin_...	1a71c356-96f6-4fa0-a...	Enabled
sabre_console_admin	sabre_console_admin	Users/Groups	sabre_console_admin	bbaf66c7-c11d-4c94-...	Enabled
sabre_console_advanced_ops	sabre_console_advanced_ops	Users/Groups	sabre_console_advanc...	599ec3b2-c4b4-485e-...	Enabled
sabre_console_basic_ops	sabre_console_basic_ops	Users/Groups	sabre_console_basic_o...	e4be1106-a6fb-4c34-...	Enabled
sabre_console_read_only	sabre_console_read_only	Users/Groups	sabre_console_read_o...	f344dfa4-e87f-41f3-be...	Enabled



Azure AD integration can only be configured for UI modules of CMP 8 and is not compatible with Web Service based modules such as SOAP and REST. Therefore, only Security Groups for AgentView, Business Configuration and Batch need be configured in Azure.

13. To add users and assign roles from the Azure dashboard, navigate back to Enterprise Applications and select the Application that was created earlier.
14. Navigate to Manage > Users and groups, and select *Add user/group* to provide access to CMP 8 to specific users from Azure AD. To assign a user multiple roles, you will need to add multiple entries.

Home > Enterprise applications | All applications > CMP 8

CMP 8 | Users and groups

Overview
Deployment Plan
Diagnose and solve problems
Manage
Properties
Owners
Roles and administrators
Users and groups
Single sign-on
Provisioning
Application proxy
Self-service
Custom security attributes
Security
Activity
Troubleshooting + Support

The application will appear for assigned users within My Apps. Set 'visible to users' to no in properties to prevent this.

Assign users and groups to app-roles for your application here. To create new app-roles for this application, use the [application registration](#)

First 200 shown, search all users & groups

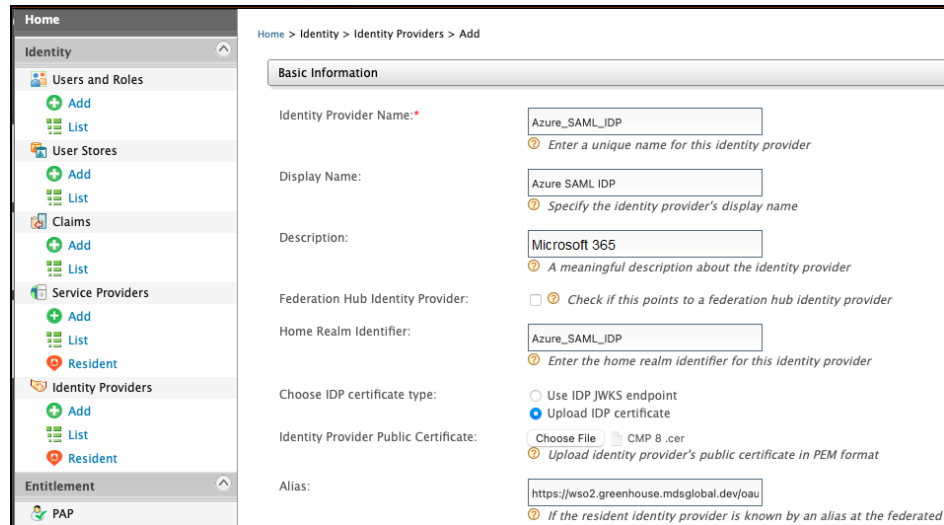
Display name	Object type	Role assigned
☐ Ken Casey	User	agent-view-enterprise-admin
☐ Ken Casey	User	config_centre_admin_user
☐ Ken Casey	User	sabre_console_admin

The configurations needed for the federation on Microsoft Azure are complete.

Navigate to Manage > Single sign-on to view all of the necessary details required to configure the Identity Provider in WSO2.

2. Set up Identity Provider in WSO2

1. Log in to the WSO2 Management Console using the admin credentials.
2. Navigate to *Identity Providers* on the left-hand pane, and select **Add**.



3. Enter the following details under *Basic Information*:
 - Identity Provider Name: (*Required) e.g. Azure_SAML_IDP
 - Display Name: e.g. Azure SAML IDP
 - Description: e.g. Microsoft 365 Login
 - Home Realm Identifier: e.g. Azure_SAML_IDP. Note: This must be the same as *Identity Provider Name*.
 - Alias: <https://wso2.greenhouse.mdsglobal.dev/oaui>
 - Select **Upload IDP certificate** from *Choose IDP certificate type*, and upload the raw certificate that was downloaded earlier from Azure.
4. Expand the *Claim Configuration* accordion, and select **Define Custom Claim Dialect** to map and configure the Azure AD claims with WSO2.
5. Select **Add Claim Mapping**, and enter the following custom mappings:

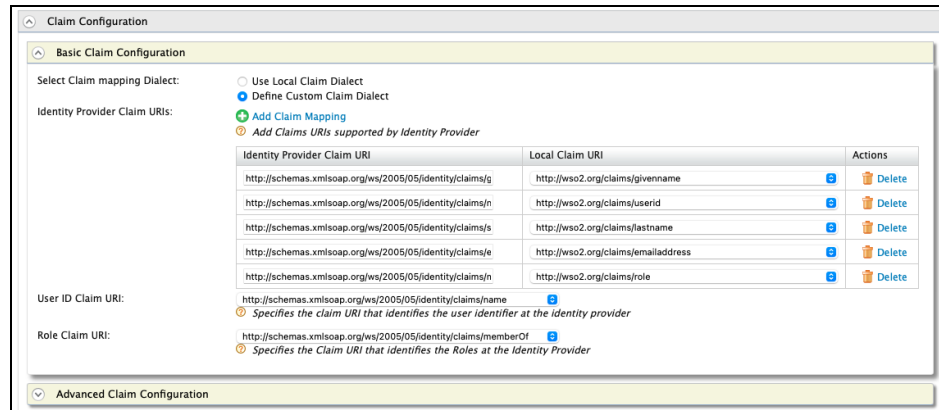
Identity Provider Claim URI	Local Claim URI
http://s-chem-as.xmlsoap.org/ws/2005/05/identity/claims/givenname	http://wso2.org/claims/givenname
http://s-chem-as.xmlsoap.org/ws/2005/05/identity/claims/name	http://wso2.org/claims/userid
http://s-chem-	http://wso2.org/claims/lastname

```

as.xmlsoap.org-
g/ws/2005/05/identity/claims/surname
http://s-
chem-
as.xmlsoap.org-
g/ws/2005/05/identity/claims/emailaddress
http://s-
chem-
as.xmlsoap.org-
g/ws/2005/05/identity/claims/memberOf
http://wso2.or-
g/claims/emailaddress
http://wso2.or-
g/claims/role

```

Set `http://s-chemas.xmlsoap.org/ws/2005/05/identity/claims/name` as the User ID Claim URI and `http://s-chemas.xmlsoap.org/ws/2005/05/identity/claims/memberOf` as the Role Claim URI.



Claim Configuration

Basic Claim Configuration

Select Claim mapping Dialect:

- ☐ Use Local Claim Dialect
- ☒ Define Custom Claim Dialect

Identity Provider Claim URIs:

- [Add Claim Mapping](#)
- [Add Claims URIs supported by Identity Provider](#)

Identity Provider Claim URI	Local Claim URI	Actions
<code>http://schemas.xmlsoap.org/ws/2005/05/identity/claims/g</code>	<code>http://wso2.org/claims/givenname</code>	Add Delete
<code>http://schemas.xmlsoap.org/ws/2005/05/identity/claims/n</code>	<code>http://wso2.org/claims/userid</code>	Add Delete
<code>http://schemas.xmlsoap.org/ws/2005/05/identity/claims/s</code>	<code>http://wso2.org/claims/lastname</code>	Add Delete
<code>http://schemas.xmlsoap.org/ws/2005/05/identity/claims/e</code>	<code>http://wso2.org/claims/emailaddress</code>	Add Delete
<code>http://schemas.xmlsoap.org/ws/2005/05/identity/claims/n</code>	<code>http://wso2.org/claims/role</code>	Add Delete

User ID Claim URI:

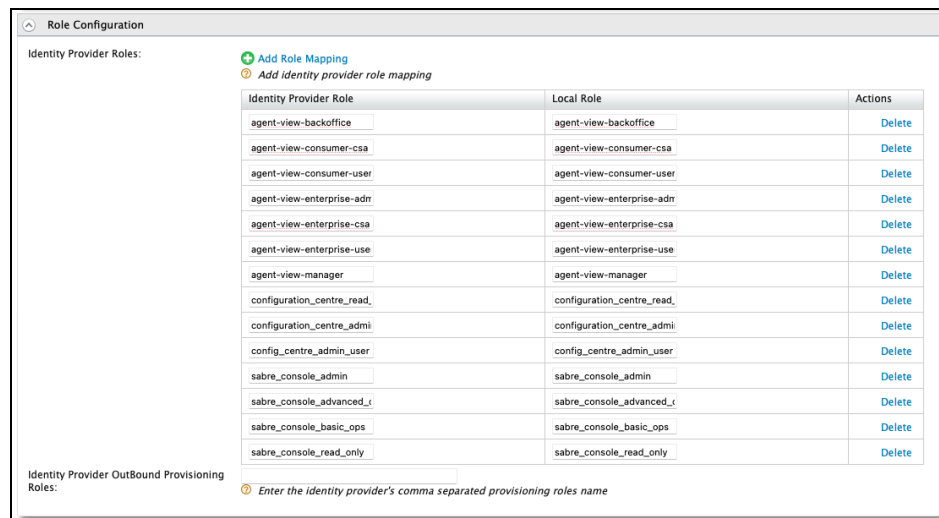
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name` [Add](#)

Role Claim URI:

`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/memberOf` [Add](#)

Advanced Claim Configuration

- Expand the *Role Configuration* accordion to define the role mapping from Azure AD to WSO2. A mapping must be registered for each defined role.



Role Configuration

Identity Provider Roles:

- [Add Role Mapping](#)
- [Add identity provider role mapping](#)

Identity Provider Role	Local Role	Actions
agent-view-backoffice	agent-view-backoffice	Delete
agent-view-consumer-csa	agent-view-consumer-csa	Delete
agent-view-consumer-user	agent-view-consumer-user	Delete
agent-view-enterprise-adm	agent-view-enterprise-adm	Delete
agent-view-enterprise-csa	agent-view-enterprise-csa	Delete
agent-view-enterprise-use	agent-view-enterprise-use	Delete
agent-view-manager	agent-view-manager	Delete
configuration_centre_read	configuration_centre_read	Delete
configuration_centre_admi	configuration_centre_admi	Delete
config_centre_admin_user	config_centre_admin_user	Delete
sabre_console_admin	sabre_console_admin	Delete
sabre_console_advanced_x	sabre_console_advanced_x	Delete
sabre_console_basic_ops	sabre_console_basic_ops	Delete
sabre_console_read_only	sabre_console_read_only	Delete

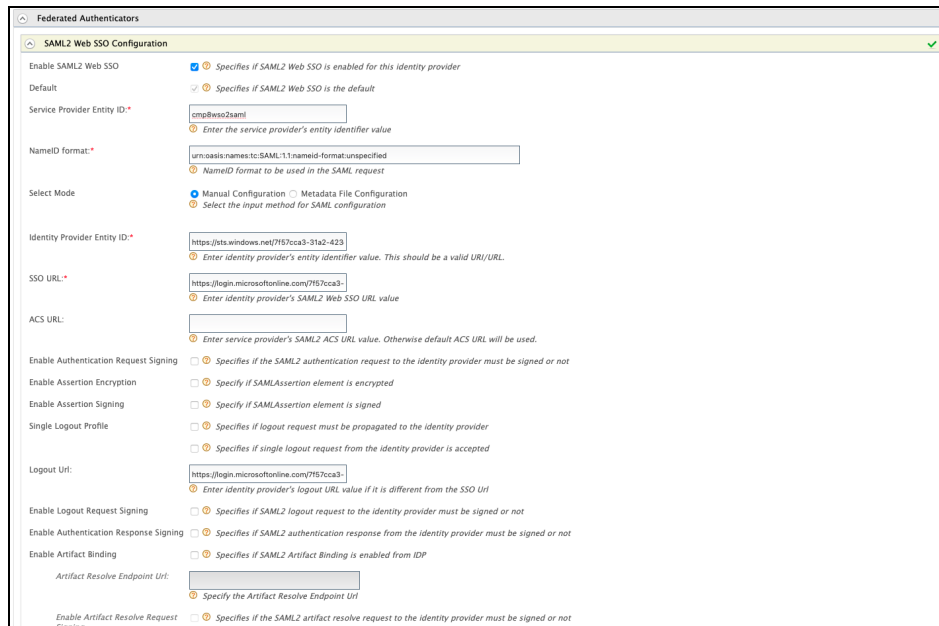
Identity Provider OutBound Provisioning Roles:

[Enter the identity provider's comma separated provisioning roles name](#)



This is required regardless of whether the roles have been defined with the same value or not.

7. Expand *Federated Authenticators*, and then expand *SAML2 Web SSO Configuration* to configure the Federated SAML2 Authenticator. Input the following in the respective fields:
 - Enable SAML2 Web SSO: true
 - Service Provider Entity ID: <"Identifier (Entity ID)" from Azure SAML Configuration (1)>
 - Identity Provider Entity ID: <"Microsoft Entra Identifier" from Azure SAML Configuration (4)>
 - SSO URL: <"Login URL" from Azure SAML Configuration (4)>
 - Logout URL: <"Logout URL" from Azure SAML Configuration (4)>
 - Keep the remaining configurations as is.



Federated Authenticators

SAML2 Web SSO Configuration

Enable SAML2 Web SSO ☒ Specifies if SAML2 Web SSO is enabled for this identity provider

Default ☒ Specifies if SAML2 Web SSO is the default

Service Provider Entity ID: Enter the service provider's entity identifier value

NameID format: NameID format to be used in the SAML request

Select Mode ☒ Manual Configuration ☐ Metadata File Configuration Select the input method for SAML configuration

Identity Provider Entity ID: Enter identity provider's entity identifier value. This should be a valid URL/URL.

SSO URL: Enter identity provider's SAML2 Web SSO URL value

ACS URL: Enter service provider's SAML2 ACS URL value. Otherwise default ACS URL will be used.

Enable Authentication Request Signing ☐ Specifies if the SAML2 authentication request to the identity provider must be signed or not

Enable Assertion Encryption ☐ Specifies if SAMLAssertion element is encrypted

Enable Assertion Signing ☐ Specifies if SAMLAssertion element is signed

Single Logout Profile ☐ Specifies if logout request must be propagated to the identity provider

Logout URL: Enter identity provider's logout URL value if it is different from the SSO URL

Enable Logout Request Signing ☐ Specifies if SAML2 logout request to the identity provider must be signed or not

Enable Authentication Response Signing ☐ Specifies if SAML2 authentication response from the identity provider must be signed or not

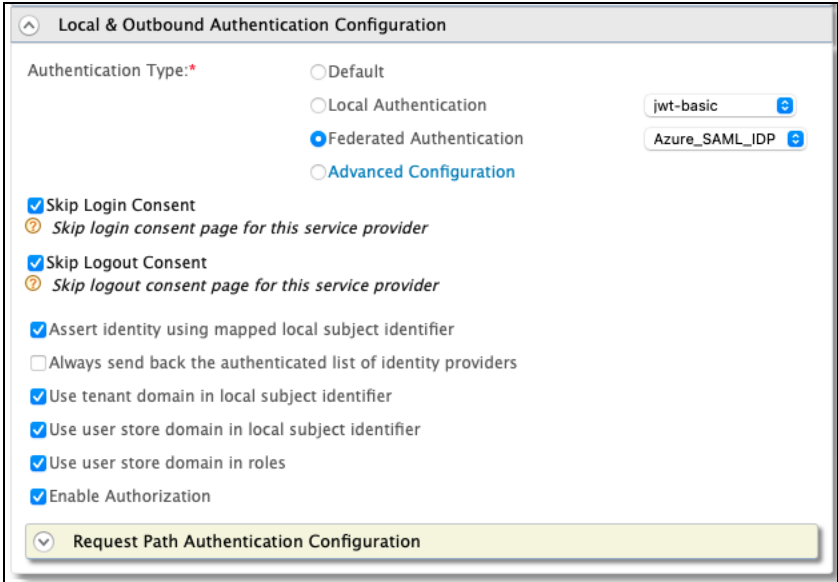
Enable Artifact Binding ☐ Specifies if SAML2 Artifact Binding is enabled from IDP

Artifact Resolve Endpoint URL: Specify the Artifact Resolve Endpoint URL

Enable Artifact Resolve Request ☐ Specifies if the SAML2 artifact resolve request to the identity provider must be signed or not

8. Click **Register** to register the IDP.
3. **Configure Service Providers to Support Azure SSO**
 1. Navigate to *Service Providers > List* to view the defined set of Service Providers for CMP 8 which are configured during the CMP 8 installation. These represent each module within the CMP 8 suite. The modules which can support Azure SSO Integration, and need to be modified, include:

- sabre-console
 - bulk-action-console
 - configuration-centre
 - agent-view
2. Select **Edit** on the agent-view service provider and expand the *Local & Outbound Authentication Configuration* accordion to configure the service provider for Federated Authentication.
 3. Configure the service provider as displayed below, and then select **Update** to save.



Local & Outbound Authentication Configuration

Authentication Type:*

☐ Default
☐ Local Authentication
☒ Federated Authentication
☐ Advanced Configuration

☒ Skip Login Consent
 ? Skip login consent page for this service provider

☒ Skip Logout Consent
 ? Skip logout consent page for this service provider

☒ Assert identity using mapped local subject identifier
☐ Always send back the authenticated list of identity providers

☒ Use tenant domain in local subject identifier
☒ Use user store domain in local subject identifier
☒ Use user store domain in roles
☒ Enable Authorization

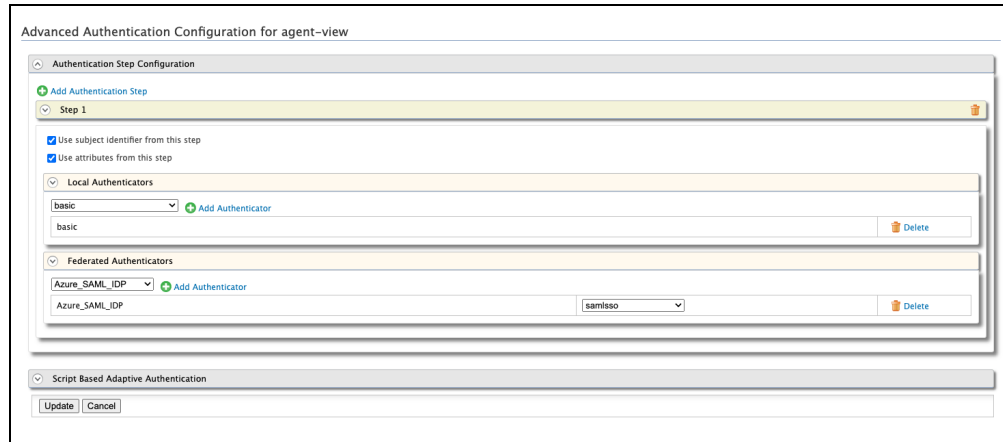

 

4. Repeat the process for all other applicable Service Providers.

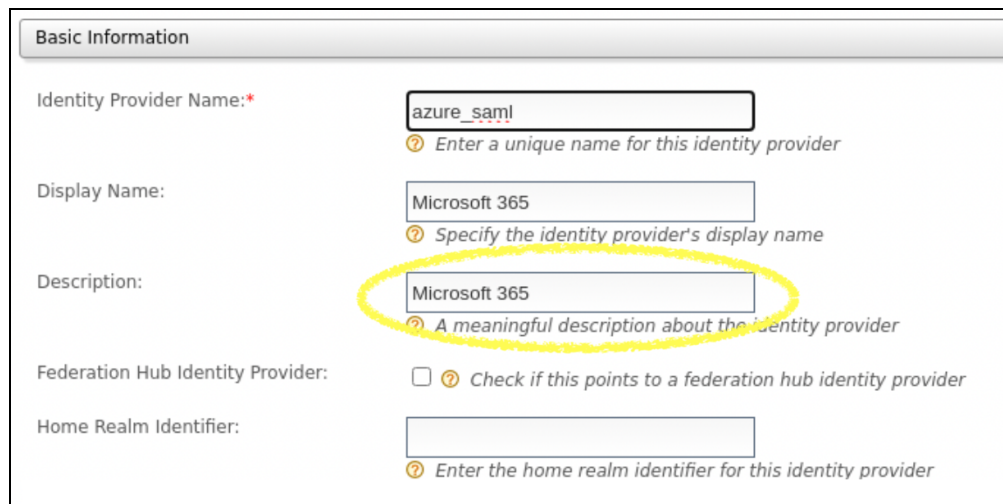
Now when navigating to any of the integrated modules, the user should be redirected to the Microsoft login page for authentication.

Update for 8.16

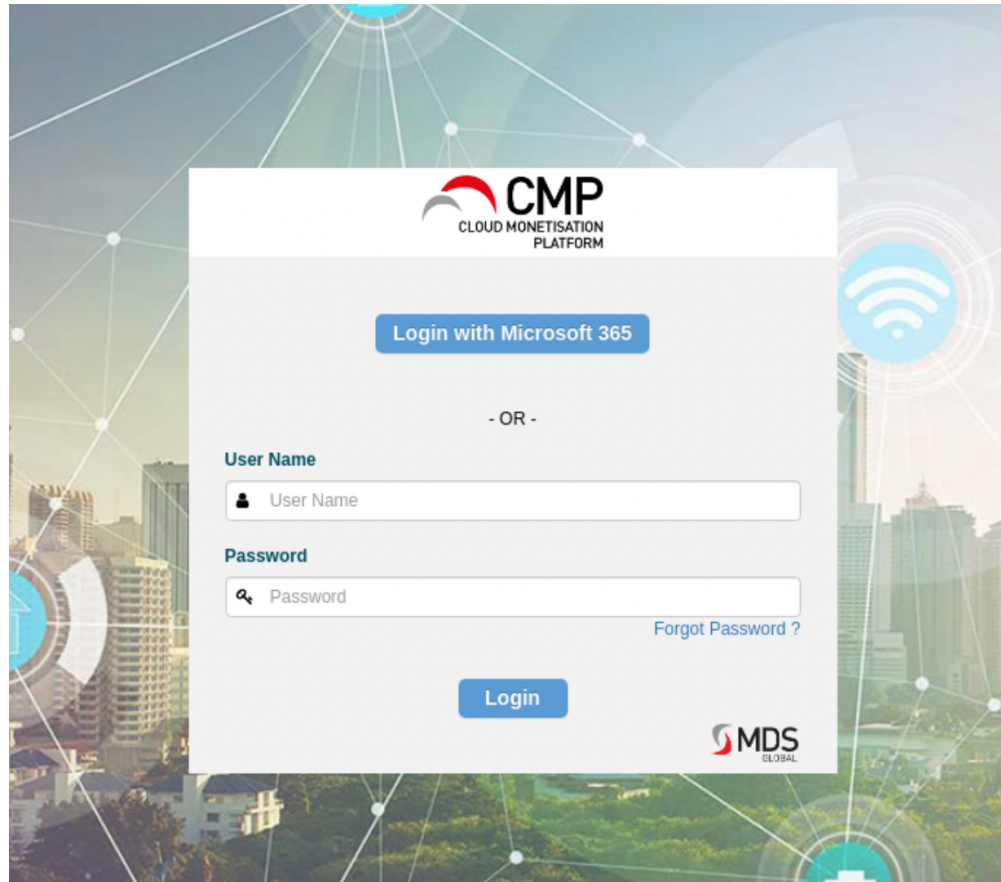
From 8.16 onwards the custom login page will allow for multiple authentication methods to be used for any given service provider. To configure this, you must select *Advanced Configuration* in the *Local & Outbound Authentication Configuration* accordion, and configure the steps as shown below:



The button label will be retrieved from the Identity Provider's description as configured in WSO2 and must be configured.



This will result in the login page displaying multiple options for authentication.



The login screen for the CMP Cloud Monetisation Platform is displayed over a background image of a city skyline with network lines. The interface includes the CMP logo at the top, a Microsoft 365 login button, a separator, and input fields for User Name and Password. A 'Forgot Password?' link and a 'Login' button are also present, along with the MDS Global logo in the bottom right corner.

CMP
CLOUD MONETISATION
PLATFORM

Login with Microsoft 365

- OR -

User Name

Password

[Forgot Password ?](#)

Login

MDS
GLOBAL

9.0 Post-Installation Checks and Tasks

The deployment playbooks have built-in post deployment checks to confirm that all components are started. However, this does not guarantee that application is in a normal state.

It is recommended that post deployment, you manually log into each of the applications that have a UI and confirm that it works as expected by performing a set of standard operations. If there are failures, check the logs as sometimes some of the services may need to be restarted.

CMP installs the following services:

Component	Service Name
PostgreSQL	postgresql-16
WSO2 Identity Server	wso2is
Role Extender	role-extender
JBoss / Wildfly	eap8-standalone
JWS / Tomcat	jws6-tomcat
Health Check Service	health-check
SAM Services	sam-services
Voucher Management Services	voucher-management-services
Shared Services	shared-services
REST Web Services	rest-ws
Product Catalogue / REST Web Services	product-catalog
Bulk Action Console	bulk-action-console
Aurora Console	aurora-console
Context Sensitive Help	context-sensitive-help
Artemis / ActiveMQ	artemis
SPaRC Engine	sparc-engine
SABRE Server	sabre-server
CMP Administration Console	sabre-console
Prometheus	prometheus

All CMP components are configured to start as `systemd` services. The standard `systemctl` command should be used to control the component services.

To check the status of a service - to see if you need to restart it, for example - run the following command:

```
sudo systemctl status <service_name>
```

To start a service, run the following command:

```
sudo systemctl start <service_name>
```

To stop the service, run the following command:

```
sudo systemctl stop <service_name>
```

And to restart the service, run the following command:

```
sudo systemctl restart <service_name>
```

9.0 High Availability Deployment Post-Installation Tasks

High Availability deployments rely upon Load Balancing, PostgreSQL replication and shared storage to be correctly configured before executing the installation process. These steps are not covered in this guide and will vary depending on the exact server landscape being deployed to.

Please consult MDS Global for assistance with planning and executing a High Availability deployment

10.0 Uninstall CMP

To uninstall the CMP stack run the following command:

```
ansible-playbook --ask-vault-password-file=<vault_password_file>  
-i <inventory_file> -b playbooks/uninstall.yaml
```