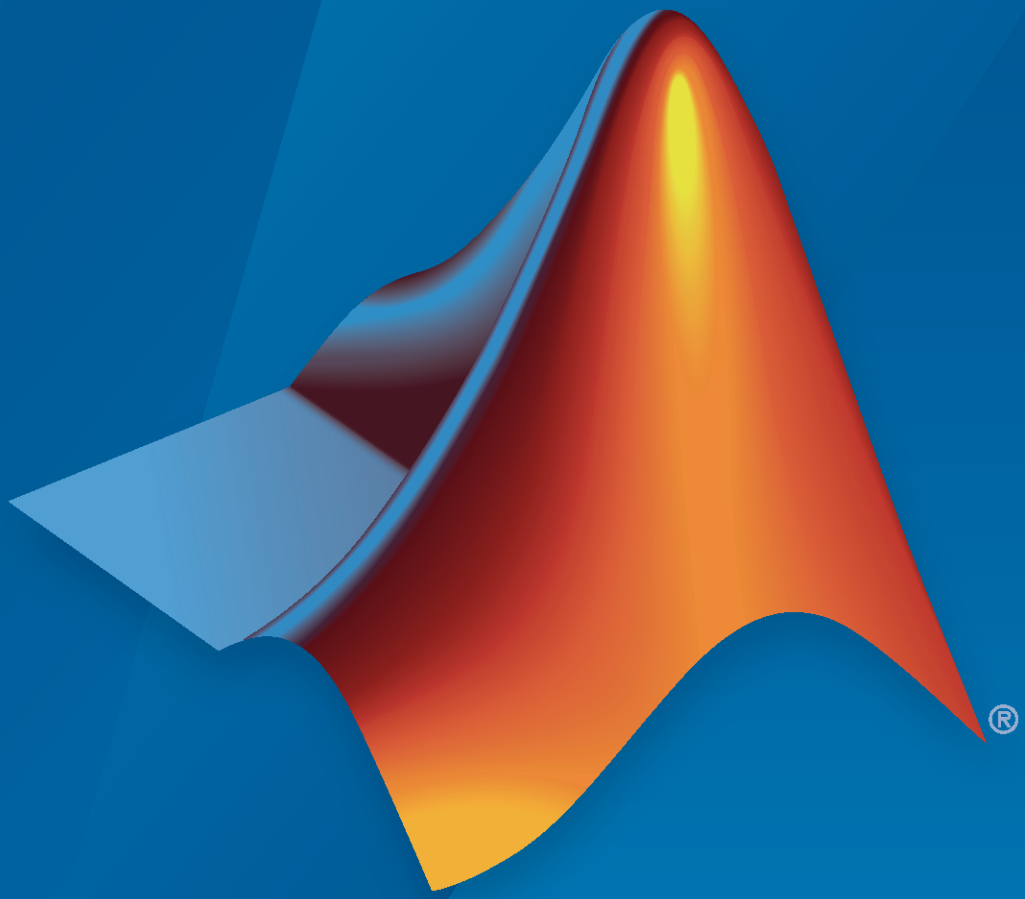
MATLAB® Online Server™

Administrator's Guide



MATLAB®

R2023a

MathWorks®

How to Contact MathWorks

Latest news: `www.mathworks.com`

Sales and services: `www.mathworks.com/sales_and_services`

User community: `www.mathworks.com/matlabcentral`

Technical support: `www.mathworks.com/support/contact_us`

Phone: 508-647-7000

The MathWorks, Inc.
1 Apple Hill Drive
Natick, MA 01760-2098

MATLAB® Online Server™ Administrator's Guide

Revision History

March 2020	Online only	New for Version 1.0 (R2020a)
September 2020	Online only	Revised for Version 1.1 (R2020b)
March 2021	Online only	Revised for Version 1.2 (R2021a)
September 2021	Online only	Revised for Version 1.3 (R2021b)
March 2022	Online only	Revised for Version 1.4 (R2022a)
September 2022	Online only	Revised for Version 1.5 (R2022b)
March 2023	Online only	Revised for Version 1.6 (R2023a)

Contents

1 Introduction

2 Installation

3 Setup and Configuration

4 Server Management

5 Troubleshooting

6 Installation Configuration Properties

7 System Administration Commands

Introduction

MATLAB Online Server Product Description

Host MATLAB Online on-premise or in a cloud environment

MATLAB Online Server lets you host MATLAB Online™ on-premise or in your cloud environment. With MATLAB Online, your users can run MATLAB from their web browsers without downloading, installing, or configuring desktop software on their own computers.

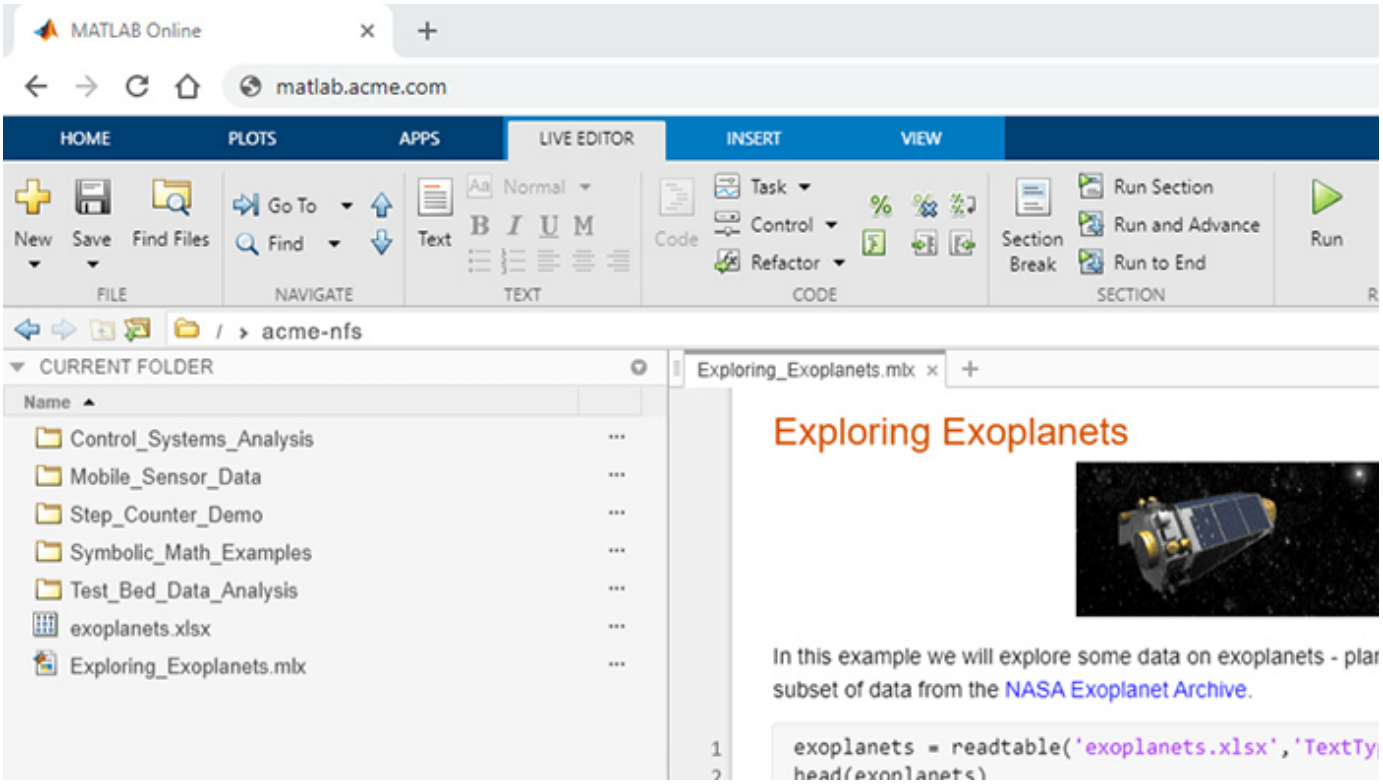
MATLAB Online Server uses a microservice architecture that supports both horizontal and vertical scaling based on your hardware configuration. To ensure efficient resource usage, you can configure your hardware to account for variable or infrequent MATLAB usage. The server integrates with your existing network file system and authentication services.

MATLAB Online provides access to the full MATLAB programming language and desktop environment, as well as to MATLAB add-on products.

Host MATLAB Online on Your Infrastructure

MATLAB Online Server lets you centrally host and manage MATLAB Online for the MATLAB users in your organization, enabling them to access MATLAB from their web browsers. You can integrate MATLAB Online Server with your network file system and authentication protocol, configure hardware resources, and define access for your users.

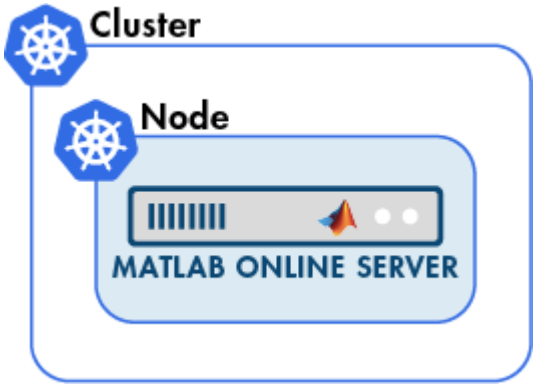
After you configure MATLAB Online Server, your users can access MATLAB Online from a browser by providing their organizational sign-on credentials. MATLAB Online provides an interactive MATLAB desktop environment for iterative analysis and prototyping. This access includes the Live Editor for creating scripts that combine code, output, and formatted text in an executable notebook. MATLAB Online Server can be hosted on your on-premise infrastructure, or it can be hosted on public cloud infrastructures but under your private accounts.



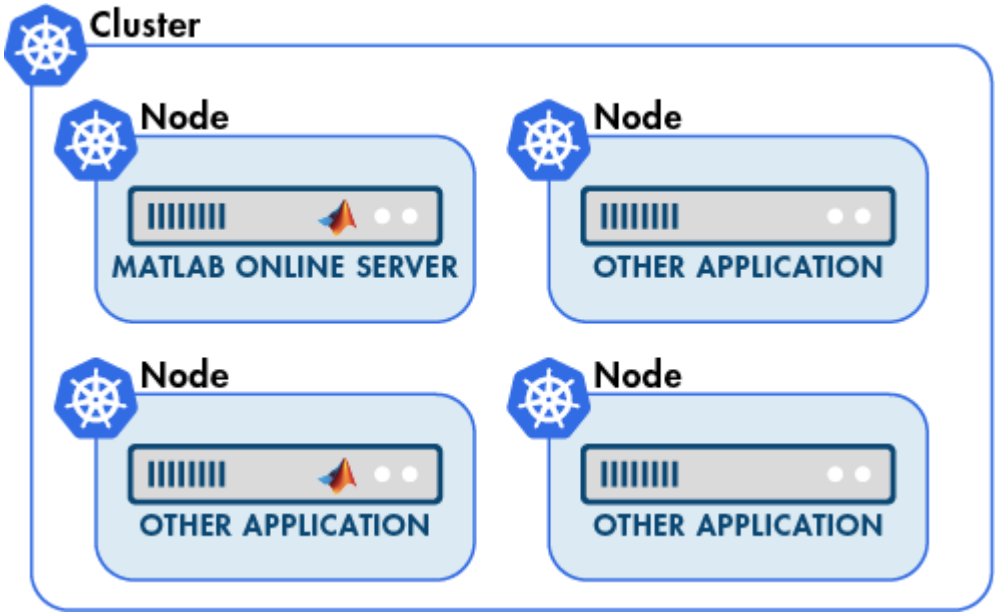
Kubernetes Architecture

MATLAB Online Server runs on Kubernetes, a container orchestration engine for automating the deployment, management, and scaling of applications. In this architecture, a Kubernetes cluster hosts one or more virtual or physical machines called nodes. Using the Kubernetes API, you can centrally manage, monitor, and scale compute resources assigned to these nodes. MATLAB Online Server is deployed as a node within a Kubernetes cluster.

If your organization does not already use Kubernetes, then you can create a new Kubernetes cluster and deploy MATLAB Online Server as a single node within that cluster.



If your organization already has a Kubernetes cluster running on a cloud service (for example, AWS® or Azure®), then you can deploy MATLAB Online Server as a new node in that cluster along with other applications.



MATLAB Online Server Components

MATLAB Online Server is composed of a set of containerized microservices and applications. These components run in pods, which are related containers running in Kubernetes that together form larger applications.

MATLAB Online Server components come in two main types: the core services and the MATLAB pool.

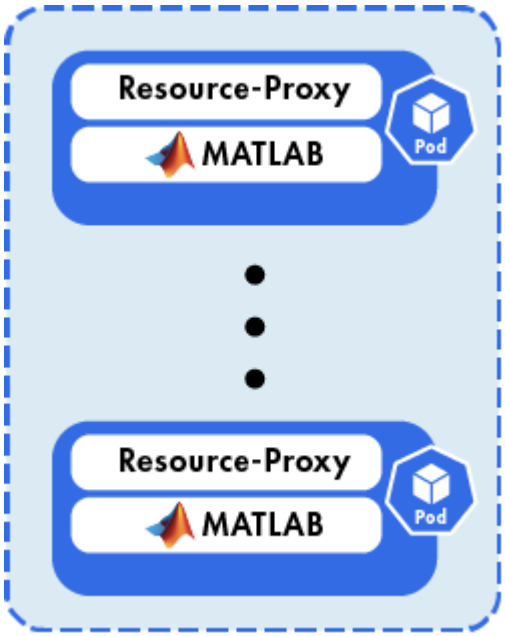
Core Services

The core services performs core server operations such as authenticating users, assigning MATLAB compute resources to users, and checking out licenses. Each service runs in a separate pod.

Service	Description
`Core-UI`	Provides the login screen and other user interface (UI) capabilities.
`Resource`	Enables clients to acquire and then release MATLAB instances from the MATLAB Pool. This component maintains information on all MATLAB instances, such as the used or available instances and the users associated with certain instances.
`Gateway`	Maps MATLAB Online Server clients to their assigned MATLAB instances. After a mapping is created, the Gateway ensures that any request from a given client is forwarded to its assigned MATLAB instance.
`AuthNZ`	Authenticates and authorizes actions of MATLAB Online Server users and some MATLAB Online Server components.
`Licensing`	Communicates with the MathWorks® License Manager, which uses FlexNet Publisher (formerly FLEXlm®) to check for MATLAB Online Server licenses.

MATLAB Pool

The MATLAB pool group is a collection of pods in which each pod is dedicated to one MATLAB end user. When a user logs in, the server assigns a MATLAB pod from this resource pool to that user. That pod runs multiple services that together serve that user a MATLAB instance.

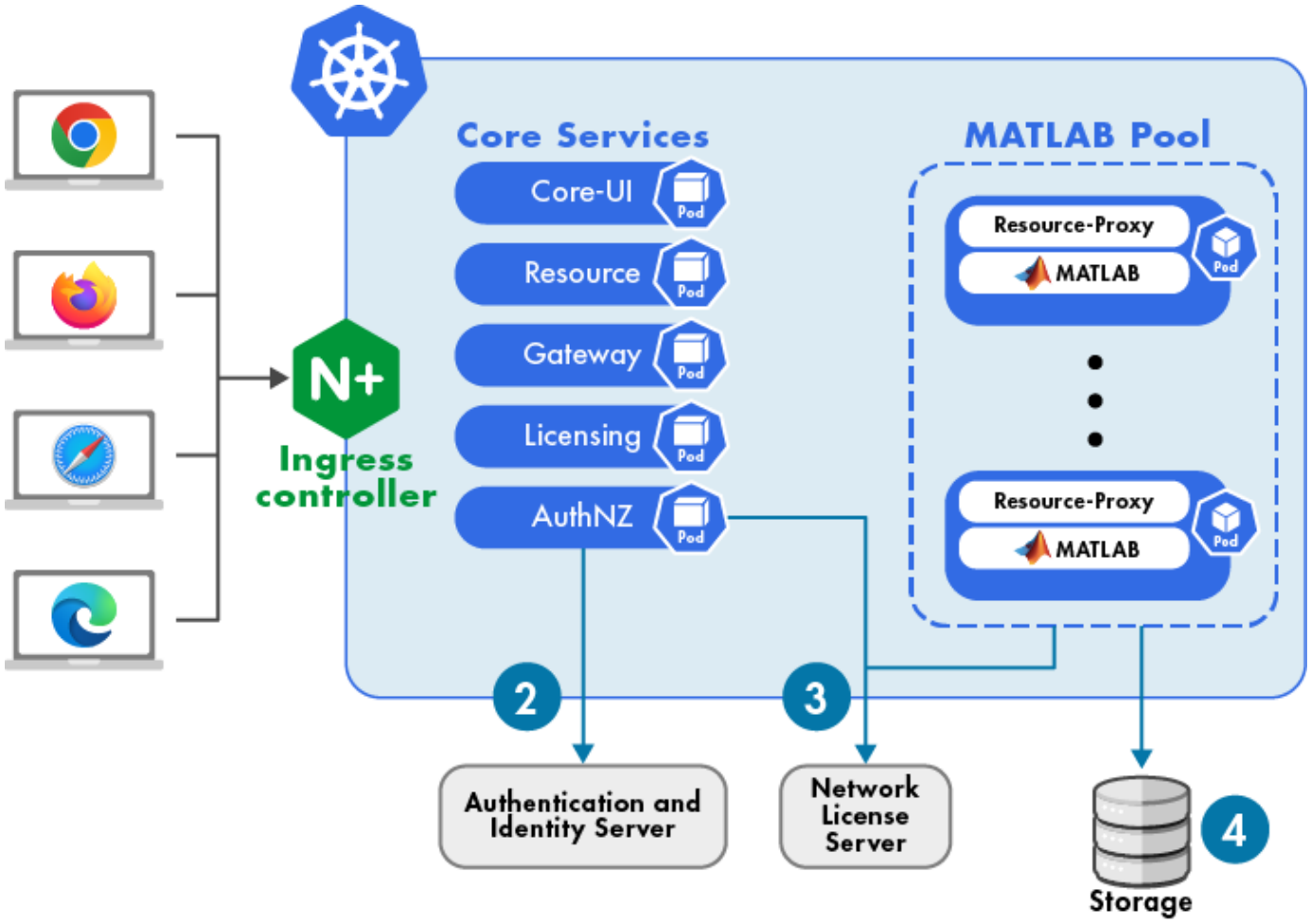


Component	Description
`Resource-Proxy`	Acts as a proxy server to the MATLAB service.
`MATLAB`	Contains the MATLAB Runtime engine.

Login Workflow

The high-level workflow for when a user logs into the system is as follows:

1. The user accesses MATLAB Online from a web browser at an organization-specific URL. The `Core-UI` component serves the landing page containing the request for the end user's username and password. The incoming request goes through an Ingress controller, which acts as a load balancer and routes requests to the appropriate components. All incoming requests to the Kubernetes cluster go through this controller.
2. The controller passes the user's credentials to the `AuthNZ` component, which passes them to the server of an identity provider, such as Active Directory®, for authentication.
3. If the authentication is successful, the `Resource` component assigns the user an available MATLAB instance (that is, a pod) from the MATLAB resource pool, and the `Licensing` component checks out a license from the network license server.
4. The server mounts user file storage and shared file systems and serves the MATLAB user interface to the user. Any subsequent requests by the user go through the Gateway component, which forwards them to their assigned MATLAB instance.



Server Configuration

Configuration settings for components reside in Helm® YAML files. Helm is the package manager for Kubernetes that is used to deploy services onto Kubernetes clusters. YAML files are a plain-text, `key: value` file format used to encode data. In MATLAB Online Server, YAML file names must be all lowercase and their contents must be strictly formatted.

To apply MATLAB Online Server service-specific configuration, edit the settings in the following YAML file.

matlab-online-server-root/overrides/*cluster*/*namespace*/*service-name*.yaml

Here:

- *matlab-online-server-root* is the MATLAB Online Server installation folder.
- *cluster* is the name of the Kubernetes cluster.
- *namespace* is the namespace of the MATLAB Online Server deployment.

For example, this sample path is for the YAML file used to configure the MATLAB resource pool.

```
/opt/matlab_online_server/overrides/matlab-online-server/mathworks/matlab-pool.yaml
```

You can configure the root folder, cluster name, and server namespace during the installation process.

Server Management

You can manage services and images for MATLAB Online Server using the `mosadm` command. The syntax for `mosadm` is as follows:

```
mosadm command option1 ... optionN
```

Use `mosadm` to perform commands such as:

- Load, list, and push the Docker® images used to build containerized applications.
- Deploy, upgrade, and undeploy services.
- Get system information.

To see the available options, use the `mosadm help` command.

Server Installation

To get started with MATLAB Online Server, install the server on a single-node Kubernetes cluster for development and testing purposes. See “Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2.

If your organization has an existing cloud-managed Kubernetes cluster, you can then migrate your installation to the cloud service used by your organization or perform a fresh installation. See “Install MATLAB Online Server on Cloud-Managed Kubernetes” on page 2-16.

If you encounter issues during the installation process:

- See the “Troubleshooting” documentation.
- Contact Technical Support. See “Contact Technical Support About MATLAB Online Server Issues” on page 5-2.

See Also

Related Examples

- “Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2
- “Install MATLAB Online Server on Cloud-Managed Kubernetes” on page 2-16
- “Contact Technical Support About MATLAB Online Server Issues” on page 5-2

MATLAB Online Server System Requirements

Intended Audience

The installation and configuration of MATLAB Online Server is intended for an IT administrator with experience in these areas:

- Installation and management of Linux® systems
- Configuration of LDAP or other authentication protocols
- Configuration of networking settings such as IP addresses and DNS entries
- Observation of operational systems using logs and monitoring systems

Supported Platforms

MATLAB Online Server is compatible with Linux only.

Supported operating systems:

- Ubuntu® 18.04 LTS, 20.04 LTS
- Red Hat® Enterprise Linux™ 7.7–7.9
- CentOS® 7.7–7.8

Supported architectures:

- x86-64

Software Requirements

Kubernetes

The table shows the Kubernetes versions supported for the last six MATLAB Online Server releases.

MATLAB Online Server Version	Kubernetes Versions Supported
R2020b	1.15–1.17
R2021a	1.17–1.18
R2021b	1.17–1.20
R2022a–R2023a	1.18–1.23

For instructions on installing Kubernetes, see “Install Kubernetes” on page 2-4.

Docker

The installation of MATLAB Online Server requires Docker version 20.

For instructions on installing Docker for your Linux distribution, see Install Docker Engine in the Docker documentation.

Note For Red Hat Enterprise Linux distributions, Docker Enterprise is recommended.

NFS

MATLAB Online Server supports network file system (NFS) versions 2 and 3.

This requirement applies only if you are configuring MATLAB file storage using an NFS drive. For more details, see “NFS” on page 3-39.

Hardware Requirements

Each computer on which you install MATLAB Online Server should meet or exceed the minimum hardware recommendations. System requirements for MATLAB Online Server installation can vary based on many factors, including the number of users and the number and size of MATLAB Docker images. If the setup program determines that your computer does not meet the following recommendations, you get a warning, but you can continue with the setup process.

Installation Type	Processor	CPU or GPU	RAM	Free Disk Space
Single installation	64-bit	8-core, 2.0 GHz or higher	32 GB	120 GB
Per concurrent MATLAB session	64-bit	4-core, 2.0 GHz or higher	8 GB	6 GB

Core count is based on physical cores, which can represent actual server hardware or cores on a virtual machine (VM). Hyperthreading is ignored for the purposes of counting cores.

Note You need a minimum of four cores for MATLAB Online Server and Kubernetes. If your computer does not meet these requirements, then the setup program will not install MATLAB Online Server.

In addition, do not install MATLAB Online Server in a cluster where other resource-intensive applications are also running.

License Requirements

MATLAB Online Server

MATLAB Online Server is provisioned as a Concurrent license in units of server instances. A server instance corresponds to a URL endpoint within your organization that your users can navigate to for the purpose of accessing MATLAB Online. There are no licensing limits to the number of users you can serve with one server instance, but as noted before, your ability to scale depends on the underlying hardware configuration you select.

MATLAB Online Server is configured to work with the network license manager. You can use the network license manager that you install using the MathWorks product installer or reuse your existing license manager.

MATLAB Online

In addition to a MATLAB Online Server license, the use of MATLAB licenses that use a network license manager is required. Supported license types are Concurrent, Network Named User, and Enterprise licenses.

See Also

Related Examples

- “Host MATLAB Online on Your Infrastructure” on page 1-3
- “Contact Technical Support About MATLAB Online Server Issues” on page 5-2

External Websites

- MATLAB Online Server Specifications and Limitations

Installation

Perform Minimal MATLAB Online Server Installation on Single Machine

This topic explains the steps you must perform to set up MATLAB Online Server on a single machine for testing and evaluation. To add nodes to the cluster, see Production environment in the Kubernetes documentation.

After MATLAB Online Server is installed, individual MATLAB instances are hosted by a Docker container, and the MATLAB Online Server stack is managed by Kubernetes.

Note These instructions include `sudo` for commands that by default require administration privileges. Depending on how your environment is configured, `sudo` might not be required in some cases.

Verify Installation Prerequisites

Before beginning the installation, verify the following:

- Your system is running one of the supported operating systems and meets the minimum system requirements described in “MATLAB Online Server System Requirements” on page 1-9.
- Your operating system is hosted at a static IP address. Kubernetes requires a static IP address to function properly. With dynamic IP addresses, when the IP address changes, the Kubernetes server fails and must be reinstalled.
- You have the latest version of Docker installed. For instructions on installing Docker for your Linux distribution, see Install Docker Engine in the Docker documentation.

Install Network License Manager

Follow the Linux instructions for installing on an offline machine described in “Install License Manager on License Server” in the Help Center. Configure the server to be accessed over the network.

- You can use an existing network license server or you can create a new installation.
- The server and the MATLAB workers need access to their MathWorks licenses. Those licenses can be either co-located on the same license server or on independent servers.
- Specify the MATLAB and server license hostname and ports using the `MLM_LICENSE_FILE` environment variable format of `<port>@<host>[,<port>@<other-host>]`. The hostname and port must be accessible from inside the Kubernetes cluster.
- If the license server is configured correctly, then the license service running in the Kubernetes cluster will run without any errors and then will check out the license when it starts. If the license is invalid or the license server cannot be reached, then the license service exits and then Kubernetes automatically attempts to restart the license service periodically until it succeeds.
- The other services require that the license server is running, but if the license is temporarily unavailable, they continue to run. After an approximately two-hour grace period, the services go into a hibernation state and then reject requests with a failed status. The logs of the services indicate when they enter or exit this hibernation or grace period state.

Install MATLAB Online Server

Follow these steps to download and then extract the MATLAB Online Server installer and files.

1. Go to the Downloads page on mathworks.com.
2. Under **Select Release**, select the current release, for example, R2023a.
3. Expand the **Get MATLAB Online Server** section. This section appears only if your account has a license for MATLAB Online Server software.
4. Click **Download** to download the ZIP file.
5. In the download folder or folder of your choice, extract the installation files. For example, this command downloads the ZIP file to your home directory (`~/`).

   ```
   unzip R2023a_matlab_online_server.zip -d ~/
   ```

   The unzipped `matlab_online_server` folder is the root folder of your MATLAB Online Server installation. Navigate into this folder. For example, if you unzipped the folder into your home directory, run this command:

   ```
   cd ~/matlab_online_server
   ```

   Confirm that this folder contains the following files and subfolders:

   ```
   ls
   ```

   - `attributions.txt`
   - `charts/`
   - `data/`
   - `install.config`
   - `install.log`
   - `mosadm`
   - `overrides/`
   - `thirdpartylicenses.txt`

   The `mosadm` command is the main utility to perform the remaining installation steps. You must run this command from your MATLAB Online Server installation folder. For details on the various operations it can perform, run this command:

   ```
   ./mosadm help
   ```

Install MATLAB

Download and install the version of MATLAB that your MATLAB Online users will connect to by using the `mosadm install-matlab` command.

Note This command requires an Internet connection. If your machine is not connected to the Internet, follow the instructions in “Install Products Using File Installation Key” instead.

By default, `mosadm install-matlab` installs the latest version of MATLAB into the `/MATLAB` folder on your local machine. If the `/MATLAB` folder does not exist, the command creates it.

```
./mosadm install-matlab
```

For additional options for installing MATLAB, see the `mosadm install-matlab` documentation. For example, you can install additional products or specify the MATLAB version you want to install.

Install Kubernetes

Download and install the required Kubernetes components.

1. Create the single-node Kubernetes architecture.

   ```
   sudo ./mosadm bootstrap-node
   ```

   This command downloads and then installs dependencies and the Kubernetes components. It then initializes a single-node Kubernetes cluster on the machine and customizes the cluster for MATLAB Online Server. It can take several minutes for the installation to finish.

   By default, the Docker daemon is configured using the properties in the file `/etc/docker/daemon.json`, and the bootstrap-node command overwrites any customizations.

   See `mosadm bootstrap-node` for more details on the bootstrap process.

   To preserve existing customizations, use the following command instead:

   ```
   sudo ./mosadm bootstrap-node --preserve-docker-config
   ```

2. At this step, you are asked to accept the terms and conditions for installing MATLAB Online Server. If you agree, answer "Y." If you do not, answer "N," and the installer exits. If you do not agree, you cannot install MATLAB Online Server.

3. To verify successful installation, confirm the existence of the `/etc/kubernetes/admin.conf` file. This file exists only after the cluster is created successfully.

   ```
   ls /etc/kubernetes/admin.conf
   ```

   ```
   /etc/kubernetes/admin.conf
   ```

4. Check that the pods are in the ready state. Each pods should have a `READY` state of `1/1` containers with a `STATUS` of `Running`. If you do not see the correct list of pods, check the version number for the install files. It can take several minutes for all pods to come online.

   ```
   sudo kubectl --kubeconfig /etc/kubernetes/admin.conf get pods --all-namespaces
   ```

   The command returns a display similar to the following:

   ```
   NAMESPACE     NAME                                       READY   STATUS    RESTARTS   AGE
   kube-system   coredns-78fcd69978-m2tph                   1/1     Running   0          51s
   kube-system   etcd-ip-172-31-31-154                      1/1     Running   1          65s
   kube-system   kube-apiserver-ip-172-31-31-154            1/1     Running   1          65s
   kube-system   kube-controller-manager-ip-172-31-31-154   1/1     Running   1          68s
   kube-system   kube-router-w2mk4                          1/1     Running   0          52s
   kube-system   kube-scheduler-ip-172-31-31-154            1/1     Running   1          65s
   ```

5. Merge the `kube-config` file into your home folder so that you can execute the `kubectl` command without having to specify the `--kubeconfig /etc/kubernetes/admin.conf` flag going forward.

   ```
   sudo ./mosadm merge-kube-config
   ```

Install NGINX Ingress Controller

The NGINX ingress controller processes incoming requests to the Kubernetes cluster and acts as a load balancer for MATLAB Online Server.

If you are using your own ingress controller, skip this step and see “Configure NGINX Ingress Controller” on page 3-61 instead.

Install the controller.

```
./mosadm install-ingress
```

This command installs the controller and displays the configuration options used for the controller. Sample output:

```
# Config file for a ingress-nginx helm deploy
rbac:
  create: true

controller:
  hostNetwork: true
  ...
```

Verify that the controller is installed correctly. `mathworks` is the default namespace used for the installation. You can customize this name in a later step.

```
kubectl get pods --namespace mathworks
```

Sample output:

```
NAME                                                 READY   STATUS    RESTARTS   AGE
mathworks-ingress-nginx-controller-7fdcd49d74-zngv4   1/1     Running   0          54s
```

Configure Server Installation

Edit the `install.config` file, located in the folder where you unzipped the installer. Sample edit command:

```
nano install.config
```

Before editing this file, consider creating a backup of `install.config` so that you have a record of the default settings.

At a minimum, modify these parameters:

Machine Configuration

Parameter	Description	Default
DOMAIN_BASE	Fully qualified domain name used to access MATLAB Online Server from a web browser. For example: `matlabonline.mycompany.com`. Specifying an IP address as the domain base is not supported. It is recommended that the domain name is registered to your organization and that the IP address of the server or load balancer are available on the DNS server of the organization. To test that your domain is working, it is possible to temporarily update the host entry of your client machine to have the IP address and the fully qualified domain name. Then, you can test accessing MATLAB Online Server instance from a browser in your client using this domain name.	`matlab.domain.com`

Authentication Configuration

Parameter	Description	Default
`ML_PASSWORD`	MATLAB Online sign-in password. The username for this bootstrapping mode is `admin`. For security purposes, it is recommended that you replace the default administration password with a more secure one. Identity provider configuration for authenticating MATLAB Online users is covered in “Configure User Authentication in MATLAB Online Server” on page 3-6.	`ML_PASSWORD=password`

License Configuration

Parameter	Description	Default
`MOS_LICENSE_SERVER`	MATLAB Online Server license server. Specify the port and hostname IP address in the format `port@host`. **Note** If you have an Enterprise License, to correctly track usage, MathWorks requires two license servers: one for MATLAB Online Server (`MOS_LICENSE_SERVER` parameter) and one for MATLAB (`MATLAB_LICENSE_SERVER` parameter). With the two-server configuration, you must also have multiple pools (at least 2). See “Configure Multiple Versions of MATLAB” on page 3-21. **Example**: `14650@192.166.248.2`	`"27000@flexlm"`
`MATLAB_LICENSE_SERVER`	MATLAB license server details. Specify the port and hostname IP address in the format `port@host`. **Example**: `27000@172.0.0.1`	`"27000@flexlm"`

Additional Configuration Options

You can optionally modify additional parameters based on your configuration. For example, you can update the namespace to use for the deployment by updating the `NAMESPACE` parameter. For a complete list of parameters, see “Installation Configuration Properties” on page 6-2.

Load Docker Images

Load all Docker images and files shipped through the MathWorks container registry onto the computer.

1. Log in to the MathWorks container registry, `containers.mathworks.com`, using your MathWorks account credentials.

```
sudo docker login containers.mathworks.com
```

2 Copy the necessary Helm charts from the container registry. Helm is an open source project that helps deploy services into Kubernetes. This command creates a folder named `charts` in the current folder and then copies the charts there.

```
sudo ./mosadm copy-helm-charts
```

3 Load Docker images. This command loads the images from the MathWorks container registry into the local Docker cache.

```
sudo ./mosadm load-docker-images
```

4 (Optional) Verify that you have enough disk space to build and load the MATLAB image by checking the size of your `/MATLAB` folder. Use the `du` (disk usage) command. For example:

```
du /MATLAB --summarize --human-readable
```

5 From the local installation of MATLAB that you installed in the “Install MATLAB” on page 2-3 step, build an image containing MATLAB. This step can take several minutes.

Note To build this image, your machine must have a `TMPDIR` environment variable that specifies a writable temporary directory mounted in `exec` mode. For details on applying this configuration, see the `mosadm build-matlab-image` documentation.

```
sudo ./mosadm build-matlab-image /MATLAB
```

Here, `/MATLAB` is the path to the folder containing your installation of MATLAB.

6 Check which images were loaded from the container registry and then built by listing all MATLAB Online Server images.

```
sudo ./mosadm list-docker-images
```

Configure Overrides

With the cluster and Docker images ready to use, before the deploying the services, load the configuration overrides.

1 Create the chart configuration files for the deployments. This command creates the YAML override files that you can use to configure the server in the `overrides/`*`cluster`*`/`*`namespace`* folder of your MATLAB Online Server installation, where:

- *`cluster`* is the name of your cluster (`matlab-online-server` for single-node installations).
- *`namespace`* is the value of the `NAMESPACE` parameter in your `install.config` file (default = `mathworks`).

```
sudo ./mosadm generate-overrides
```

2 Confirm that `overrides/`*`cluster`*`/`*`namespace`* contains the following files and subfolders.

```
ls ./overrides/cluster/namespace
```

- `all.yaml`
- `authnz.yaml`
- `core-ui.yaml`

- `gateway.yaml`
- `license.yaml`
- `matlab-pool.yaml`
- `namespace.yaml`
- `resource.yaml`

The YAML files enable you to override the default configuration and customize the server. For example, you can configure storage and authentication. To customize the server before deployment, see “Setup and Configuration”. Otherwise, proceed to the next step.

Deploy MATLAB Online Server

1. To install all the MATLAB Online Server components, run this command:

   ```
   ./mosadm deploy
   ```

   This command uses Helm to install all the charts, customized with any overrides that you generated in the previous step.

2. Check that the services are all running with `kubectl`, replacing *`your-namespace`* with the namespace you used for MATLAB Online Server (default = `mathworks`). Depending on how you configured the server and the platform on which you installed the server, the exact pods you see might differ from the ones shown here.

   ```
   kubectl get pods --namespace your-namespace
   ```

   ```
   NAME                                                  READY   STATUS    RESTARTS   AGE
   your-namespace-authnz-7994c9866d-675fb                1/1     Running   0          10m
   your-namespace-core-ui-cfdcccc4c-5bhrc                1/1     Running   0          10m
   your-namespace-gateway-88ffd446d-mbf2l                1/1     Running   0          10m
   your-namespace-gateway-proxy-6f85db9cbb-8ftbr         1/1     Running   0          10m
   your-namespace-gateway-proxy-6f85db9cbb-mdhr7         1/1     Running   0          10m
   your-namespace-license-5cc85b97cd-zg4vd               1/1     Running   0          10m
   your-namespace-matlab-pool-9cc6b6465-9rdz8            2/2     Running   0          6m54s
   your-namespace-matlab-pool-9cc6b6465-t7wp2            2/2     Running   0          3m40s
   your-namespace-matlab-pool-helpsearch-8479fbdc88-4r6sd  1/1     Running   0          6m54s
   your-namespace-matlab-pool-ui-8484bbbd4d-t6777        1/1     Running   0          6m54s
   your-namespace-resource-78f9b97745-fzwlq              1/1     Running   0          10m
   ```

The pods running MATLAB can take several minutes to fully start (2/2 containers). It is important to wait until the pods are in the ready state. If any of the containers do not start running, check the license server and the `install.config` settings. See “Resolve MATLAB Pod Issues” on page 5-8.

Verify MATLAB Online Server Installation

1. Make sure the address in the `DOMAIN_BASE` parameter defined in `install.config` is registered and can be routed to the IP address of the host machine. Run this command, replacing *`domain`* with the value of the `DOMAIN_BASE` parameter.

   ```
   ping -c 1 matlab.domain.com
   ```

   This command returns the IP address of the host machine. In some types of installation, the ping does not receive packets. If you have such an installation, this command returns the IP address of the server you ping.

2. Open a browser on another machine and then go to the address where you are hosting MATLAB Online:

matlab.domain.com/matlabonline

The address is the `DOMAIN_BASE` parameter defined in `install.config`.

3. Your browser window shows a prompt to select a password only. For a single-machine installation, use the username `admin` and the password you specified in the `ML_PASSWORD` parameter of `install.config` (the default is `password`).

If you can now access your internal version of MATLAB Online, installation is complete. If you cannot access MATLAB Online, or you run into any other technical error, contact MathWorks Support.

Set Up Server

With the installation process and initial configuration complete, you must now complete additional tasks to set up the server for use in your organization. For example, you need to:

- Configure user authentication by specifying your identity provider details.
- Configure the installed MATLAB versions and specify which user groups can access them.
- Configure persistent storage and which directories, drives, and files users can access.
- Customize the MATLAB Online sign-in screen.

To get started setting up the server, see “Set Up MATLAB Online Server After Installation” on page 3-2.

See Also

Related Examples

- “Install Add-Ons for MATLAB in MATLAB Online Server” on page 2-38
- “Install Support Packages for MATLAB in MATLAB Online Server” on page 2-36
- “Host MATLAB Online on Your Infrastructure” on page 1-3
- “Use MATLAB Online Hosted by Your Organization”
- “Uninstall MATLAB Online Server” on page 2-42

Perform Offline MATLAB Online Server Installation

This topic explains the steps you must perform to set up MATLAB Online Server on a single machine in a restricted or air-gapped environment.

MATLAB Online Server installer provides a way to bootstrap a single-node Kubernetes cluster on a node. However, this method pulls many third-party applications, for example, libraries, container images, and configuration files. This workflow might not work when network security measures meant to physically isolate the server from unsecure networks are employed.

Note the following:

- The MATLAB Online Server single-machine installer assumes that the operating system is one of the Supported Platforms on page 1-9.
- After MATLAB Online Server is installed, individual MATLAB instances are hosted by a Docker container, and the MATLAB Online Server stack is managed by Kubernetes.

Verify Installation Prerequisites

Complete the following software checks before downloading and then configuring the MATLAB Online Server installer. The tasks can be done in any order.

- Make sure you can run commands using sudo.
- The environment is running one of the Supported Platforms on page 1-9.
- Docker is installed. For the minimum version supported by MATLAB Online Server, see “MATLAB Online Server System Requirements” on page 1-9.
- Make sure that the RHEL® or CentOS node has access to the internet for building the offline installer has the same configuration or is a replica of the offline machine.
- Verify that the RHEL or CentOS node is running in air-gapped environment.
- Verify that your operating system is hosted at a static IP address. Kubernetes requires a static IP address to function properly. With dynamic IP addresses, when the IP address changes, the Kubernetes server fails and must be reinstalled.

Install Network License Manager

Follow the Linux instructions for installing on an offline machine described in “Install License Manager on License Server” in the Help Center. Configure the server to be accessed over the network.

- You can use an existing network license server or you can create a new installation.
- The server and the MATLAB workers need access to their MathWorks licenses. Those licenses can be either co-located on the same license server or on independent servers.
- Specify the MATLAB and server license hostname and ports using the `MLM_LICENSE_FILE` environment variable format of `<port>@<host>[,<port>@<other-host>]`. The hostname and port must be accessible from inside the Kubernetes cluster.
- If the license server is configured correctly, then the license service running in the Kubernetes cluster will run without any errors and then will check out the license when it starts. If the license is invalid or the license server cannot be reached, then the license service exits and then Kubernetes automatically attempts to restart the license service periodically until it succeeds.

- The other services require that the license server is running, but if the license is temporarily unavailable, they continue to run. After an approximately two-hour grace period, the services go into a hibernation state and then reject requests with a failed status. The logs of the services indicate when they enter or exit this hibernation or grace period state.

Install MATLAB

Download MATLAB for offline installation and then install the software on the offline server.

Follow the Linux instructions for “Install Products Using File Installation Key” in the Help Center. Make sure you install MATLAB into the folder `/MATLAB`.

Install MATLAB Online Server

Follow these steps to download and then extract the MATLAB Online Server installer and files.

1. Go to the Downloads page on mathworks.com.
2. Under **Select Release**, select the current release, for example, R2023a.
3. Expand the **Get MATLAB Online Server** section. This section appears only if your account has a license for MATLAB Online Server software.
4. Click **Download** to download the ZIP file.
5. In the download folder or folder of your choice, extract the installation files. For example, this command downloads the ZIP file to your home directory (`~/`).

   ```
   unzip R2023a_matlab_online_server.zip -d ~/
   ```

   The unzipped `matlab_online_server` folder is the root folder of your MATLAB Online Server installation. Navigate into this folder. For example, if you unzipped the folder into your home directory, run this command:

   ```
   cd ~/matlab_online_server
   ```

   Confirm that this folder contains the following files and subfolders:

   ```
   ls
   ```

   - `attributions.txt`
   - `charts/`
   - `data/`
   - `install.config`
   - `install.log`
   - `mosadm`
   - `overrides/`
   - `thirdpartylicenses.txt`

   The `mosadm` command is the main utility to perform the remaining installation steps. You must run this command from your MATLAB Online Server installation folder. For details on the various operations it can perform, run this command:

   ```
   ./mosadm help
   ```

Build Offline Installer

Next, on the same machine, build the installer for MATLAB Online Server with the following command:

```
sudo ./mosadm build-offline-installer
```

This command downloads all the required artifacts for building the installer:

- Kubernetes repo (for Kubernetes executables)
- Yum repos configured on your system (RHEL repos)
- Docker registry
  - Docker hub
  - gcr.io
- Helm charts
  - GitHub®

Refer to the "Installed Utilities" section of “Bootstrap Node for Ubuntu Systems” on page 7-3 for more details on these artifacts.

After the build command completes, check that the following folders are present:

- *`matlab_online_server`*`/data/kubepacker-direct-install-rhel-1.2.0/files/opt/mw/offlineinstaller/`
- *`matlab_online_server`*`/data/com.3p.*`

Output from the build can be viewed in the terminal. If the build was not successful, try rebuilding, and if it fails again, contact MathWorks Support.

Install MATLAB Online Server on Air-Gapped Machine

1. Copy the *`matlab_online_server`* folder to the offline server where you want to host MATLAB Online Server. For example:

   ```
   scp -r matlab_online_server
   username@server.com/home/rheluser/matlab_online_server

   scp -r matlab/R2023a username@server.com/MATLAB
   ```

2. Log off of the first machine and log in to the air-gapped machine.
3. In the *`matlab_online_server`* folder, open the `install.config` file and set the `OFFLINE_MODE` parameter to `true`.
4. Install MATLAB Online Server.

   Follow the MATLAB Online Server install workflow starting with “Install Kubernetes” on page 2-4.

Set Up Server

With the installation process and initial configuration complete, you must now complete additional tasks to set up the server for use in your organization. For example, you need to:

- Configure user authentication by specifying your identity provider details.
- Configure the installed MATLAB versions and specify which user groups can access them.
- Configure persistent storage and which directories, drives, and files users can access.
- Customize the MATLAB Online sign-in screen.

To get started setting up the server, see “Set Up MATLAB Online Server After Installation” on page 3-2.

See Also

Related Examples

- “Install Add-Ons for MATLAB in MATLAB Online Server” on page 2-38
- “Install Support Packages for MATLAB in MATLAB Online Server” on page 2-36
- “Host MATLAB Online on Your Infrastructure” on page 1-3
- “Use MATLAB Online Hosted by Your Organization”
- “Uninstall MATLAB Online Server” on page 2-42

Install MATLAB Online Server on Cloud-Managed Kubernetes

These instructions are for installing MATLAB Online Server on cloud-managed Kubernetes.

If you are installing the server on Red Hat OpenShift® specifically, see “Install MATLAB Online Server on Red Hat OpenShift” on page 2-26.

Verify Installation Prerequisites

Make sure your system meets the following requirements:

- Push and pull access to the remote Docker repository
- Existing Kubernetes cluster on Azure AKS, Amazon EKS, or Google GKE
  - Supported node

    Refer to Supported Platforms on page 1-9.
  - Support for network policy

    MATLAB Online Server creates NetworkPolicy resources, which are silently ignored if the cluster is not created with network policy enabled. Network policy specifies which pods can communicate with each other. Because it is important from a security standpoint, network policy is enabled by default in MATLAB Online Server.

    If your cluster does not have a container network interface (CNI) that supports network policies, then MATLAB Online Server functions properly but not securely. This configuration is not recommended.

    Refer to documentation from your specific cloud provider for more details on how to enable network policy in the Kubernetes cluster.
  - Prerequisite for GKE only: To deploy the NGINX Ingress Controller on GKE cluster, create a `clusterrolebinding` for the specific user as shown:

    ```
    kubectl create clusterrolebinding cluster-admin-binding \
    > --clusterrole cluster-admin \
    > --user <userid>
    ```

On the machine that will communicate with the managed cluster to deploy MATLAB Online Server (client machine), complete the following software checks before downloading and then configuring MATLAB Online Server. These tasks can be done in any order.

- Make sure you can access the cluster’s `kubeconfig` file from the client machine.
- Make sure the client machine OS is supported. Refer to Supported Platforms on page 1-9.
- Install Helm v3.
- Install Kubectl.
- Install Docker.
- Check that the machine meets the software and hardware requirements described in “MATLAB Online Server System Requirements” on page 1-9.

Install Network License Manager

Follow the Linux instructions for installing on an offline machine described in “Install License Manager on License Server” in the Help Center. Configure the server to be accessed over the network.

- You can use an existing network license server or you can create a new installation.
- The server and the MATLAB workers need access to their MathWorks licenses. Those licenses can be either co-located on the same license server or on independent servers.
- Specify the MATLAB and server license hostname and ports using the `MLM_LICENSE_FILE` environment variable format of `<port>@<host>[,<port>@<other-host>]`. The hostname and port must be accessible from inside the Kubernetes cluster.
- If the license server is configured correctly, then the license service running in the Kubernetes cluster will run without any errors and then will check out the license when it starts. If the license is invalid or the license server cannot be reached, then the license service exits and then Kubernetes automatically attempts to restart the license service periodically until it succeeds.
- The other services require that the license server is running, but if the license is temporarily unavailable, they continue to run. After an approximately two-hour grace period, the services go into a hibernation state and then reject requests with a failed status. The logs of the services indicate when they enter or exit this hibernation or grace period state.

Install MATLAB

Download and install the version of MATLAB that your MATLAB Online users will connect to by using the `mosadm install-matlab` command.

Note This command requires an Internet connection. If your machine is not connected to the Internet, follow the instructions in “Install Products Using File Installation Key” instead.

By default, `mosadm install-matlab` installs the latest version of MATLAB into the `/MATLAB` folder on your local machine. If the `/MATLAB` folder does not exist, the command creates it.

```
./mosadm install-matlab
```

For additional options for installing MATLAB, see the `mosadm install-matlab` documentation. For example, you can install additional products or specify the MATLAB version you want to install.

Install MATLAB Online Server

Follow these steps to download and then extract the MATLAB Online Server installer and files.

1. Go to the Downloads page on mathworks.com.
2. Under **Select Release**, select the current release, for example, R2023a.
3. Expand the **Get MATLAB Online Server** section. This section appears only if your account has a license for MATLAB Online Server software.
4. Click **Download** to download the ZIP file.
5. In the download folder or folder of your choice, extract the installation files. For example, this command downloads the ZIP file to your home directory (`~/`).

```
unzip R2023a_matlab_online_server.zip -d ~/
```

The unzipped `matlab_online_server` folder is the root folder of your MATLAB Online Server installation. Navigate into this folder. For example, if you unzipped the folder into your home directory, run this command:

```
cd ~/matlab_online_server
```

Confirm that this folder contains the following files and subfolders:

```
ls
```

- `attributions.txt`
- `charts/`
- `data/`
- `install.config`
- `install.log`
- `mosadm`
- `overrides/`
- `thirdpartylicenses.txt`

The `mosadm` command is the main utility to perform the remaining installation steps. You must run this command from your MATLAB Online Server installation folder. For details on the various operations it can perform, run this command:

```
./mosadm help
```

Install NGINX Ingress Controller

The NGINX ingress controller processes incoming requests to the Kubernetes cluster and acts as a load balancer for MATLAB Online Server.

If you are using your own ingress controller, skip this step and see “Configure NGINX Ingress Controller” on page 3-61 instead.

Install the controller.

```
./mosadm install-ingress
```

This command installs the controller and displays the configuration options used for the controller. Sample output:

```
# Config file for a ingress-nginx helm deploy
rbac:
  create: true

controller:
  hostNetwork: true
  ...
```

Verify that the controller is installed correctly. `mathworks` is the default namespace used for the installation. You can customize this name in a later step.

```
kubectl get pods --namespace mathworks
```

Sample output:

```
NAME                                                READY   STATUS    RESTARTS   AGE
mathworks-ingress-nginx-controller-7fdcd49d74-zngv4   1/1     Running   0          54s
```

Configure Server Installation

Edit the file `install.config`, located in the folder where you unzipped the installer. Before editing this file, consider creating a backup of `install.config` so that you have a record of the default settings.

In the file, modify these parameters:

Machine Configuration

Parameter	Description	Default
`DOMAIN_BASE`	Fully qualified domain name used to access MATLAB Online Server from a web browser. For example: `matlabonline.mycompany.com`. Specifying an IP address as the domain base is not supported. It is recommended that the domain name is registered to your organization and that the IP address of the server or load balancer are available on the DNS server of the organization. To test that your domain is working, it is possible to temporarily update the host entry of your client machine to have the IP address and the fully qualified domain name. Then, you can test accessing MATLAB Online Server instance from a browser in your client using this domain name.	`matlab.domain.com`

Authentication Configuration

Parameter	Description	Default
`ML_PASSWORD`	MATLAB Online sign-in password. The username for this bootstrapping mode is `admin`. For security purposes, it is recommended that you replace the default administration password with a more secure one. Identity provider configuration for authenticating MATLAB Online users is covered in “Configure User Authentication in MATLAB Online Server” on page 3-6.	`ML_PASSWORD=password`

License Configuration

Parameter	Description	Default
`MOS_LICENSE_SERVER`	MATLAB Online Server license server. Specify the port and hostname IP address in the format `port@host`. **Note** If you have an Enterprise License, to correctly track usage, MathWorks requires two license servers: one for MATLAB Online Server (`MOS_LICENSE_SERVER` parameter) and one for MATLAB (`MATLAB_LICENSE_SERVER` parameter). With the two-server configuration, you must also have multiple pools (at least 2). See “Configure Multiple Versions of MATLAB” on page 3-21. **Example**: `14650@192.166.248.2`	`"27000@flexlm"`
`MATLAB_LICENSE_SERVER`	MATLAB license server details. Specify the port and hostname IP address in the format `port@host`. **Example**: `27000@172.0.0.1`	`"27000@flexlm"`

Remote Docker Registry Configuration

If you have a remote registry within the company, configure the following information about the registry.

Parameter	Description	Default
`DOCKER_REGISTRY`	Remote Docker registry to push Docker images to **Example**: `myregistry.mycompany.com`	`docker-registry.com`

Parameter	Description	Default
DOCKER_REPOSITORY	Remote Docker repository **Example**: mos	repo-prefix
IMAGE_PULL_SECRET	User-friendly name for the Kubernetes secret object. When you deploy MATLAB Online Server, this object is created from registry credentials. This secret enables the nodes to pull images from the remote registry. **Example**: mymossecret	mwdockerregistry

Security Configuration

Configure your environment with Transport Layer Security (TLS). Apply the settings that are appropriate for your organization.

Parameter	Description	Default
IS_TLS_ENABLED	Flag to enable or disable Transport Layer Security (TLS), specified as false or true.	false
TLS_KEY_FILE	Path to the TLS key file. This parameter applies only if IS_TLS_ENABLED is set to true.	/opt/tls.key
TLS_CERT_FILE	Path to the TLS certification file. This parameter applies only if IS_TLS_ENABLED is set to true.	/opt/tls.crt

Additional Configuration Options

You can optionally modify additional parameters based on your configuration. For example, you can update the namespace to use for the deployment by updating the NAMESPACE parameter. For a complete list of parameters, see “Installation Configuration Properties” on page 6-2.

Prepare Docker Images

Load the required Docker images from the MathWorks container registry and push them to your remote registry.

Load Docker Images

Load all Docker images and files shipped through the MathWorks container registry onto the computer.

1. Log in to the MathWorks container registry, `containers.mathworks.com`, using your MathWorks account credentials.

   ```
   sudo docker login containers.mathworks.com
   ```

2. Copy the necessary Helm charts from the container registry. Helm is an open source project that helps deploy services into Kubernetes. This command creates a folder named `charts` in the current folder and then copies the charts there.

   ```
   sudo ./mosadm copy-helm-charts
   ```

3. Load Docker images. This command loads the images from the MathWorks container registry into the local Docker cache.

   ```
   sudo ./mosadm load-docker-images
   ```

4. (Optional) Verify that you have enough disk space to build and load the MATLAB image by checking the size of your `/MATLAB` folder. Use the `du` (disk usage) command. For example:

   ```
   du /MATLAB --summarize --human-readable
   ```

5. From the local installation of MATLAB that you installed in the “Install MATLAB” on page 2-17 step, build an image containing MATLAB. This step can take several minutes.

   Note To build this image, your machine must have a `TMPDIR` environment variable that specifies a writable temporary directory mounted in `exec` mode. For details on applying this configuration, see the `mosadm build-matlab-image` documentation.

   ```
   sudo ./mosadm build-matlab-image /MATLAB
   ```

   Here, `/MATLAB` is the path to the folder containing your installation of MATLAB.

6. Check which images were loaded from the container registry and then built by listing all MATLAB Online Server images.

   ```
   sudo ./mosadm list-docker-images
   ```

Push Docker Images to Remote Registry

To allow cluster nodes to use Docker images, you must provide credentials to access the registry. If you have logged in to the remote registry from your client machine at least one time, then the file at `~/.docker/config.json` has the registry credentials.

1. Copy the file with the following command:

   ```
   cp ~/.docker/config.json ./dockerconfig.json
   ```

   Contents from this file are used to create a Kubernetes secret by the name <IMAGE_PULL_SECRET> that is configured in the `install.config` file.

2. (If you are not using the Amazon ECR registry, skip this step.) Create an ECR repository. Amazon ECR registry expects repositories for all images to be created in advance. For example, you can create an ECR repository by using this command:

   ```
   ./mosadm list-docker-images | sed -E 's|mathworks/mos(.*?):.*|\1|' |
   sed -n '1!p' | xargs -L 1 aws ecr create-repository --repository-name
   ```

3. Push Docker images to the remote registry (configured in `install.config`) with the following command:

```
./mosadm push-docker-images
```

Configure Overrides

With the cluster and Docker images ready to use, before the deploying the services, load the configuration overrides.

Enter the `mosadm generate-overrides` command with the option `--skip-matlab-image`, as shown:

```
./mosadm generate-overrides --skip-matlab-image
```

The `mosadm` command looks for the MATLAB Docker image locally even when the image is available in the remote Docker repository. So that `mosadm` can proceed without its having to download the image locally, passing the flag `--skip-matlab-image` is necessary.

Deploy MATLAB Online Server

1. To install all the MATLAB Online Server components, run this command:

   ```
   ./mosadm deploy
   ```

   This command uses Helm to install all the charts, customized with any overrides that you generated in the previous step.

2. Check that the services are all running with `kubectl`, replacing *`your-namespace`* with the namespace you used for MATLAB Online Server (default = `mathworks`). Depending on how you configured the server and the platform on which you installed the server, the exact pods you see might differ from the ones shown here.

   ```
   kubectl get pods --namespace your-namespace
   ```

   ```
   NAME                                                  READY   STATUS    RESTARTS   AGE
   your-namespace-authnz-7994c9866d-675fb                1/1     Running   0          10m
   your-namespace-core-ui-cfdcccc4c-5bhrc                1/1     Running   0          10m
   your-namespace-gateway-88ffd446d-mbf2l                1/1     Running   0          10m
   your-namespace-gateway-proxy-6f85db9cbb-8ftbr         1/1     Running   0          10m
   your-namespace-gateway-proxy-6f85db9cbb-mdhr7         1/1     Running   0          10m
   your-namespace-license-5cc85b97cd-zg4vd               1/1     Running   0          10m
   your-namespace-matlab-pool-9cc6b6465-9rdz8            2/2     Running   0          6m54s
   your-namespace-matlab-pool-9cc6b6465-t7wp2            2/2     Running   0          3m40s
   your-namespace-matlab-pool-helpsearch-8479fbdc88-4r6sd 1/1    Running   0          6m54s
   your-namespace-matlab-pool-ui-8484bbbd4d-t6777        1/1     Running   0          6m54s
   your-namespace-resource-78f9b97745-fzwlq              1/1     Running   0          10m
   ```

The pods running MATLAB can take several minutes to fully start (2/2 containers). It is important to wait until the pods are in the ready state. If any of the containers do not start running, check the license server and the `install.config` settings. See “Resolve MATLAB Pod Issues” on page 5-8.

Connect to MATLAB Online Server

After you have installed MATLAB Online Server successfully, provide a way to access MATLAB Online from the browser.

Run the following command to get the external IP of the cluster, which is required to configure the DNS:

```
kubectl get service --namespace mathworks
```

You can route the DOMAIN_BASE parameter, defined in `install.config`, to the external IP shown in the output of the command above.

Verify MATLAB Online Server Installation

1. Make sure the address in the DOMAIN_BASE parameter defined in `install.config` is registered and can be routed to the IP address of the host machine. Run this command, replacing *`domain`* with the value of the DOMAIN_BASE parameter.

   ```
   ping -c 1 matlab.domain.com
   ```

   This command returns the IP address of the host machine. In some types of installation, the ping does not receive packets. If you have such an installation, this command returns the IP address of the server you ping.
2. Open a browser on another machine and then go to the address where you are hosting MATLAB Online:

   `matlab.domain.com``/matlabonline`

   The address is the DOMAIN_BASE parameter defined in `install.config`.
3. Your browser window shows a prompt to select a password only. For a single-machine installation, use the username `admin` and the password you specified in the ML_PASSWORD parameter of `install.config` (the default is `password`).

If you can now access your internal version of MATLAB Online, installation is complete. If you cannot access MATLAB Online, or you run into any other technical error, contact MathWorks Support.

Set Up Server

With the installation process and initial configuration complete, you must now complete additional tasks to set up the server for use in your organization. For example, you need to:

- Configure user authentication by specifying your identity provider details.
- Configure the installed MATLAB versions and specify which user groups can access them.
- Configure persistent storage and which directories, drives, and files users can access.
- Customize the MATLAB Online sign-in screen.

To get started setting up the server, see “Set Up MATLAB Online Server After Installation” on page 3-2.

See Also

Related Examples

- “Install Add-Ons for MATLAB in MATLAB Online Server” on page 2-38
- “Install Support Packages for MATLAB in MATLAB Online Server” on page 2-36
- “Host MATLAB Online on Your Infrastructure” on page 1-3
- “Use MATLAB Online Hosted by Your Organization”
- “Uninstall MATLAB Online Server” on page 2-42

Install MATLAB Online Server on Red Hat OpenShift

These instructions are for installing MATLAB Online Server on the Red Hat OpenShift platform.

Verify Installation Prerequisites

Make sure your system meets the following requirements:

- Push and pull access to the remote Docker repository
- Existing OpenShift 4.x cluster with:
  - Support for network policy
  - Service account that has access to "privileged" security context constraints.

    You can create this access with the following commands (the commands require the `oc` command tool):

```
>oc create namespace mathworks
>oc create sa custom-sa --namespace mathworks
>oc adm policy add-scc-to-user privileged -z custom-sa --namespace mathworks
```

Also complete the following software checks on the client machine before downloading and then configuring MATLAB Online Server. The tasks can be done in any order.

- Make sure you can access the cluster's `kubeconfig` file from the client machine.
- Make sure the client machine OS is supported. Refer to Supported Platforms on page 1-9.
- Install Helm 3.
- Install Kubectl.
- Install Docker.
- Check that the machine meets the software and hardware requirements described in "MATLAB Online Server System Requirements" on page 1-9.

Install Network License Manager

Follow the Linux instructions for installing on an offline machine described in "Install License Manager on License Server" in the Help Center. Configure the server to be accessed over the network.

- You can use an existing network license server or you can create a new installation.
- The server and the MATLAB workers need access to their MathWorks licenses. Those licenses can be either co-located on the same license server or on independent servers.
- Specify the MATLAB and server license hostname and ports using the `MLM_LICENSE_FILE` environment variable format of `<port>@<host>[,<port>@<other-host>]`. The hostname and port must be accessible from inside the Kubernetes cluster.
- If the license server is configured correctly, then the license service running in the Kubernetes cluster will run without any errors and then will check out the license when it starts. If the license is invalid or the license server cannot be reached, then the license service exits and then Kubernetes automatically attempts to restart the license service periodically until it succeeds.
- The other services require that the license server is running, but if the license is temporarily unavailable, they continue to run. After an approximately two-hour grace period, the services go

into a hibernation state and then reject requests with a failed status. The logs of the services indicate when they enter or exit this hibernation or grace period state.

Install MATLAB

Download and install the version of MATLAB that your MATLAB Online users will connect to by using the `mosadm install-matlab` command.

Note This command requires an Internet connection. If your machine is not connected to the Internet, follow the instructions in "Install Products Using File Installation Key" instead.

By default, `mosadm install-matlab` installs the latest version of MATLAB into the `/MATLAB` folder on your local machine. If the `/MATLAB` folder does not exist, the command creates it.

```
./mosadm install-matlab
```

For additional options for installing MATLAB, see the `mosadm install-matlab` documentation. For example, you can install additional products or specify the MATLAB version you want to install.

Install MATLAB Online Server

Follow these steps to download and then extract the MATLAB Online Server installer and files.

1. Go to the Downloads page on mathworks.com.
2. Under **Select Release**, select the current release, for example, R2023a.
3. Expand the **Get MATLAB Online Server** section. This section appears only if your account has a license for MATLAB Online Server software.
4. Click **Download** to download the ZIP file.
5. In the download folder or folder of your choice, extract the installation files. For example, this command downloads the ZIP file to your home directory (`~/`).

   ```
   unzip R2023a_matlab_online_server.zip -d ~/
   ```

   The unzipped `matlab_online_server` folder is the root folder of your MATLAB Online Server installation. Navigate into this folder. For example, if you unzipped the folder into your home directory, run this command:

   ```
   cd ~/matlab_online_server
   ```

   Confirm that this folder contains the following files and subfolders:

   ```
   ls
   ```

   - `attributions.txt`
   - `charts/`
   - `data/`
   - `install.config`
   - `install.log`
   - `mosadm`

- `overrides/`
- `thirdpartylicenses.txt`

The `mosadm` command is the main utility to perform the remaining installation steps. You must run this command from your MATLAB Online Server installation folder. For details on the various operations it can perform, run this command:

```
./mosadm help
```

Configure Server Installation

Edit the file `install.config`, located in the folder where you unzipped the installer. Before editing this file, consider creating a backup of `install.config` so that you have a record of the default settings.

In the file, modify these parameters:

Machine Configuration

Parameter	Description	Default
`DOMAIN_BASE`	Fully qualified domain name used to access MATLAB Online Server from a web browser. For example: `matlabonline.mycompany.com`. Specifying an IP address as the domain base is not supported. It is recommended that the domain name is registered to your organization and that the IP address of the server or load balancer are available on the DNS server of the organization. To test that your domain is working, it is possible to temporarily update the host entry of your client machine to have the IP address and the fully qualified domain name. Then, you can test accessing MATLAB Online Server instance from a browser in your client using this domain name.	`matlab.domain.com`

Authentication Configuration

Parameter	Description	Default
`ML_PASSWORD`	MATLAB Online sign-in password. The username for this bootstrapping mode is `admin`. For security purposes, it is recommended that you replace the default administration password with a more secure one. Identity provider configuration for authenticating MATLAB Online users is covered in “Configure User Authentication in MATLAB Online Server” on page 3-6.	`ML_PASSWORD=password`

License Server Configuration

Parameter	Description	Default
`MOS_LICENSE_SERVER`	MATLAB Online Server license server. Specify the port and hostname IP address in the format `port@host`. **Note** If you have an Enterprise License, to correctly track usage, MathWorks requires two license servers: one for MATLAB Online Server (`MOS_LICENSE_SERVER` parameter) and one for MATLAB (`MATLAB_LICENSE_SERVER` parameter). With the two-server configuration, you must also have multiple pools (at least 2). See “Configure Multiple Versions of MATLAB” on page 3-21. **Example**: `14650@192.166.248.2`	`"27000@flexlm"`
`MATLAB_LICENSE_SERVER`	MATLAB license server details. Specify the port and hostname IP address in the format `port@host`. **Example**: `27000@172.0.0.1`	`"27000@flexlm"`

Remote Docker Registry Configuration

If you have a remote registry within the company, configure the following information about the registry.

Parameter	Description	Default
`DOCKER_REGISTRY`	Remote Docker registry to push Docker images to **Example**: `myregistry.mycompany.com`	`docker-registry.com`

Parameter	Description	Default
DOCKER_REPOSITORY	Remote Docker repository **Example**: mos	repo-prefix
IMAGE_PULL_SECRET	User-friendly name for the Kubernetes secret object. When you deploy MATLAB Online Server, this object is created from registry credentials. This secret enables the nodes to pull images from the remote registry. **Example**: mymossecret	mwdockerregistry

Security Configuration

Configure your environment with Transport Layer Security (TLS). Apply the settings that are appropriate for your organization.

Parameter	Description	Default
IS_TLS_ENABLED	Flag to enable or disable Transport Layer Security (TLS), specified as false or true.	false
TLS_KEY_FILE	Path to the TLS key file. This parameter applies only if IS_TLS_ENABLED is set to true.	/opt/tls.key
TLS_CERT_FILE	Path to the TLS certification file. This parameter applies only if IS_TLS_ENABLED is set to true.	/opt/tls.crt

Additional Configuration Options

You can optionally modify additional parameters based on your configuration. For example, you can update the namespace to use for the deployment by updating the NAMESPACE parameter. For a complete list of parameters, see “Installation Configuration Properties” on page 6-2.

Prepare Docker Images

Load the required Docker images from the MathWorks container registry and push them to your remote registry.

Load Docker Images

Load all Docker images and files shipped through the MathWorks container registry onto the computer.

1. Log in to the MathWorks container registry, `containers.mathworks.com`, using your MathWorks account credentials.

   ```
   sudo docker login containers.mathworks.com
   ```

2. Copy the necessary Helm charts from the container registry. Helm is an open source project that helps deploy services into Kubernetes. This command creates a folder named `charts` in the current folder and then copies the charts there.

   ```
   sudo ./mosadm copy-helm-charts
   ```

3. Load Docker images. This command loads the images from the MathWorks container registry into the local Docker cache.

   ```
   sudo ./mosadm load-docker-images
   ```

4. (Optional) Verify that you have enough disk space to build and load the MATLAB image by checking the size of your `/MATLAB` folder. Use the `du` (disk usage) command. For example:

   ```
   du /MATLAB --summarize --human-readable
   ```

5. From the local installation of MATLAB that you installed in the “Install MATLAB” on page 2-27 step, build an image containing MATLAB. This step can take several minutes.

   Note To build this image, your machine must have a `TMPDIR` environment variable that specifies a writable temporary directory mounted in `exec` mode. For details on applying this configuration, see the `mosadm build-matlab-image` documentation.

   ```
   sudo ./mosadm build-matlab-image /MATLAB
   ```

   Here, `/MATLAB` is the path to the folder containing your installation of MATLAB.

6. Check which images were loaded from the container registry and then built by listing all MATLAB Online Server images.

   ```
   sudo ./mosadm list-docker-images
   ```

Push Docker Images to Remote Registry

To allow cluster nodes to use Docker images, you must provide credentials to access the registry. If you have logged in to the remote registry from your client machine at least one time, then the file at `~/.docker/config.json` has the registry credentials.

1. Copy the file with the following command:

   ```
   cp ~/.docker/config.json ./dockerconfig.json
   ```

   Contents from this file are used to create a Kubernetes secret by the name <IMAGE_PULL_SECRET> that is configured in the `install.config` file.

2. Push Docker images to the remote registry (configured in `install.config`) with the following command:

   ```
   ./mosadm push-docker-images
   ```

Configure Overrides

With the cluster and Docker images ready to use, before the deploying the services, load the configuration overrides.

1. Enter the `mosadm generate-overrides` command with the option `--skip-matlab-image`, as shown:

   ```
   ./mosadm generate-overrides --skip-matlab-image
   ```

   The `mosadm` command looks for the MATLAB Docker image locally even when the image is available in the remote Docker repository. So that `mosadm` can proceed without its having to download the image locally, passing the flag `--skip-matlab-image` is necessary.

2. Update the file `./overrides/<cluster>/<namespace>/all.yaml` to apply the following setting:

   ```
   global:
     allowArbitraryUserId: true
   ```

   MATLAB Online Server runs its containers with a specific UserId, whereas OpenShift randomizes the container's UserId based on a range that is different for different projects in OpenShift. You must set this property to true to allow OpenShift to choose an arbitrary UserId.

3. Update the file `./overrides/<cluster>/<namespace>/all.yaml` to add the ingress controller and its annotations.

   MATLAB Online Server supports Bring-Your-Own-Ingress, and OpenShift clusters come with an ingress controller. To use the OpenShift ingress controller, set these fields for the version of OpenShift you are using.

   OpenShift 4.7 or Earlier

   ```
   global:
     ingressController:
       name: openshift-ingress
       annotations:
         haproxy.router.openshift.io/timeout:"365s"
   ```

   OpenShift 4.8 or Later

   ```
   global:
     ingressController:
       name: openshift-default
       annotations:
         haproxy.router.openshift.io/timeout:"365s"
   ```

   MATLAB Online Server expects the timeout to be set to `"365s"` because the default timeout of openshift-ingress is not sufficient for MATLAB Online Server workflows.

4. Update the override file `./overrides/<cluster>/<namespace>/matlab-pool.yaml`.

   The changes you must implement are required for the following reasons:

   - The MATLAB pool pod needs to run with a specific UserId. Because the pod also runs a privileged container to set up user storage and proxy to MATLAB, it needs to use a custom service account that has access to "privileged" security context constraints.

- The apparmor security profile, which is enabled by default, must be disabled since it is not supported on Red Hat.

```
# Allow use of custom service accounts
customServiceAccount:
  enabled: true
  name: custom-sa
security:
  apparmor:
    enabled: false
```

You can change the service account name in this example to the name you created for the account.

Deploy MATLAB Online Server

1. To install all the MATLAB Online Server components, run this command:

   ```
   ./mosadm deploy
   ```

   This command uses Helm to install all the charts, customized with any overrides that you generated in the previous step.

2. Check that the services are all running with `kubectl`, replacing *`your-namespace`* with the namespace you used for MATLAB Online Server (default = `mathworks`). Depending on how you configured the server and the platform on which you installed the server, the exact pods you see might differ from the ones shown here.

   ```
   kubectl get pods --namespace your-namespace
   ```

   ```
   NAME                                                  READY   STATUS    RESTARTS   AGE
   your-namespace-authnz-7994c9866d-675fb                1/1     Running   0          10m
   your-namespace-core-ui-cfdcccc4c-5bhrc                1/1     Running   0          10m
   your-namespace-gateway-88ffd446d-mbf2l                1/1     Running   0          10m
   your-namespace-gateway-proxy-6f85db9cbb-8ftbr         1/1     Running   0          10m
   your-namespace-gateway-proxy-6f85db9cbb-mdhr7         1/1     Running   0          10m
   your-namespace-license-5cc85b97cd-zg4vd               1/1     Running   0          10m
   your-namespace-matlab-pool-9cc6b6465-9rdz8            2/2     Running   0          6m54s
   your-namespace-matlab-pool-9cc6b6465-t7wp2            2/2     Running   0          3m40s
   your-namespace-matlab-pool-helpsearch-8479fbdc88-4r6sd   1/1     Running   0          6m54s
   your-namespace-matlab-pool-ui-8484bbbd4d-t6777        1/1     Running   0          6m54s
   your-namespace-resource-78f9b97745-fzwlq              1/1     Running   0          10m
   ```

The pods running MATLAB can take several minutes to fully start (2/2 containers). It is important to wait until the pods are in the ready state. If any of the containers do not start running, check the license server and the `install.config` settings. See “Resolve MATLAB Pod Issues” on page 5-8.

Connect to MATLAB Online

After you have installed MATLAB Online Server successfully, provide a way to access MATLAB Online from the browser.

The OpenShift ingress controller creates a service object to access the cluster from outside. To get the service, issue the following command:

```
oc get services --namespace openshift-ingress
```

This command shows an IP address in the External IP column. Using the DNS service of your choice (for example, Route53), map `DOMAIN_BASE` to this IP address.

Verify MATLAB Online Server Installation

1. Make sure the address in the `DOMAIN_BASE` parameter defined in `install.config` is registered and can be routed to the IP address of the host machine. Run this command, replacing *`domain`* with the value of the `DOMAIN_BASE` parameter.

   ```
   ping -c 1 matlab.domain.com
   ```

   This command returns the IP address of the host machine. In some types of installation, the ping does not receive packets. If you have such an installation, this command returns the IP address of the server you ping.

2. Open a browser on another machine and then go to the address where you are hosting MATLAB Online:

   `matlab.domain.com``/matlabonline`

   The address is the `DOMAIN_BASE` parameter defined in `install.config`.

3. Your browser window shows a prompt to select a password only. For a single-machine installation, use the username `admin` and the password you specified in the `ML_PASSWORD` parameter of `install.config` (the default is `password`).

If you can now access your internal version of MATLAB Online, installation is complete. If you cannot access MATLAB Online, or you run into any other technical error, contact MathWorks Support.

Set Up Server

With the installation process and initial configuration complete, you must now complete additional tasks to set up the server for use in your organization. For example, you need to:

- Configure user authentication by specifying your identity provider details.
- Configure the installed MATLAB versions and specify which user groups can access them.
- Configure persistent storage and which directories, drives, and files users can access.
- Customize the MATLAB Online sign-in screen.

To get started setting up the server, see “Set Up MATLAB Online Server After Installation” on page 3-2.

See Also

Related Examples

- “Install Add-Ons for MATLAB in MATLAB Online Server” on page 2-38
- “Install Support Packages for MATLAB in MATLAB Online Server” on page 2-36
- “Host MATLAB Online on Your Infrastructure” on page 1-3
- “Use MATLAB Online Hosted by Your Organization”
- “Uninstall MATLAB Online Server” on page 2-42

Install Support Packages for MATLAB in MATLAB Online Server

A support package is a type of add-on that enables MATLAB users to use a MathWorks product with specific third-party hardware and software. This topic describes how to install support packages in MATLAB Online so that they are accessible by all users.

Prerequisites

Before you begin, make sure you have done the following tasks:

- Installed the Linux version of MATLAB from the Downloads page and launched MATLAB successfully.
- Learned what support packages are and how to manage them on the Linux version of desktop MATLAB.
- Installed and configured MATLAB Online Server.

Install Procedure

Step 1. Install MATLAB

Skip this step if you have already installed the Linux version of MATLAB as specified in Prerequisites.

1. Create an empty folder `/MATLAB`.
2. Install MATLAB in the folder `/MATLAB` with the toolboxes you want.
3. Note that `/MATLAB` is `matlabroot`, not `/MATLAB/<releasename>`
4. Create a folder inside `/MATLAB` named `SupportPackages`:

   ```
   /MATLAB/SupportPackages
   ```

Step 2. Change Support Package Root

1. Open MATLAB from `/MATLAB/bin`.
2. Change the support package root by issuing the following command:

   ```
   matlabshared.supportpkg.setSupportPackageRoot('/MATLAB/SupportPackages')
   ```
3. Restart MATLAB.

Step 3. Install Support Packages

1. From within the restarted MATLAB, sign in to your MathWorks account.
2. Install any required support packages using the instructions in “Get and Manage Add-Ons” on the MathWorks website.

   Support packages that install third party applications on the machine are not supported.
3. To verify that the support packages are installed successfully, launch MATLAB and then execute the following command:

   ```
   supportPackages = matlabshared.supportpkg.getInstalled
   ```
4. To verify that the support packages are available in MATLAB Online, launch MATLAB in the browser and then execute the following command:

```
addons = matlab.addons.installedAddons
```

5. Exit MATLAB.
6. Delete the licenses folder under `/MATLAB`.

Step 4. Build MATLAB Image

1. From the installation you just performed, build the MATLAB image using the following command:

   ```
   sudo ./mosadm build-matlab-image /MATLAB
   ```

   If you have a multi-node installation, you can retag the Docker images and then push them to a remote Docker registry using the `mosadm push-docker-images` command. Replace *`registryName`* with the name of your Docker registry and *`repositoryName`* with the name of your repository.

   ```
   mosadm push-docker-images --deployment-image-registry registryName --deployment-image-repository repositoryName
   ```

2. Use `mosadm` to redeploy the license service using the following commands:

   ```
   ./mosadm undeploy license
   ./mosadm deploy license
   ```

See Also

Related Examples

- “Install Add-Ons for MATLAB in MATLAB Online Server” on page 2-38

Install Add-Ons for MATLAB in MATLAB Online Server

MATLAB add-ons encompass a wide variety of resources, including products, apps, support packages, and toolboxes. Add-ons extend the capabilities of MATLAB with functionality for additional tasks and applications. This topic describes how to install add-ons in MATLAB Online so that they are accessible by all users.

Prerequisites

Before you begin, make sure that you:

- Understand what add-ons are and how to manage them on the Linux version of the MATLAB Desktop. For more information, see “Get and Manage Add-Ons” in the Help Center.
- Installed and configured MATLAB Online Server.

Create Docker Image with Add-Ons Installed

1. Go to the Downloads at mathworks.com. Install the Linux version of MATLAB in the folder `/MATLAB`.
2. Install the required add-ons using the “Get and Manage Add-Ons” instructions on mathworks.com.

   By default, MATLAB add-ons are installed in the following folder:

   ```
   ~/MATLAB Add-Ons
   ```
3. After you install the add-ons successfully, exit MATLAB.
4. Copy the folder where the add-ons were installed into the MATLAB installation directory, `/MATLAB`.
5. Under `/MATLAB`, delete the licenses folder.
6. Using the following command, build the MATLAB image from the installation you just performed:

   ```
   sudo ./mosadm build-matlab-image /MATLAB
   ```

If you have a multi-node installation, you can retag the Docker images and then push them to a remote Docker registry using the `mosadm push-docker-images` command. Replace *`registryName`* with the name of your Docker registry and *`repositoryName`* with the name of your repository.

```
mosadm push-docker-images --deployment-image-registry registryName --deployment-image-repository repositoryName
```

Configure MATLAB Online Server to Use Installed Add-Ons

Each storage profile can optionally define an add-ons directory with the following YAML structure:

```
addons:
  directory: add-ons-directory
```

Ensure that *add-ons-directory* is available in the MATLAB container during runtime in one of the these ways:

- Inside the Docker container
- Through a special mount using the storage profile
- Chosen dynamically by the administrator

The benefits and drawbacks to the first two procedures are the same: the add-ons you select are installed and then rolled out to the entire organization, but MATLAB users cannot install any custom add-ons.

Alternatively, you can choose the add-ons directory during user sign in, as long as users have write access to the directory. Using this procedure, MATLAB users can install any custom add-on, but you cannot roll out any add-ons to the users. The add-ons must be installed by the users themselves.

Note If the addons: *add-ons-directory* value is empty or not defined, *startDirectory*/MATLAB Add-Ons is the default directory for the add-ons.

From the following options, select the one that works best for your organization.

Option 1: Read-Only Add-Ons Directory Inside Docker Container

During this procedure, an add-ons directory is created inside the MATLAB installation directory and created as a Docker image. The add-ons you select are installed and then rolled out to the entire organization, but MATLAB users cannot install any custom add-ons.

In matlab-pool.yaml (override), add the add-ons directory to the storage profile as shown:

```
storage:
  profiles:
    - name: nfsHome
      startDirectory: "/home/${subject.subjectId}"
      mounts:
        - name: home
          permissionType: user
          mountPath: "/home/${subject.subjectId}"
          uid: "${subject.uid}"
          gid: "${subject.gid}"
          type: nfs
          server: "nfs"
          subPath: "/exports/home/${subject.subjectId}"
          mountOptions:
"rw,relatime,vers=3,rsize=1048576,wsize=1048576,namlen=255,acregmi
n=600,acregmax=600,acdirmin=600,acdirmax=600,hard,nocto,noacl,prot
o=tcp,timeo=600,retrans=2,mountproto=tcp,local_lock=none,nolock"
      restrictNavigationOutsideMounts: false
      addons:
        directory: "/MATLAB/MATLAB Add-Ons"
```

Option 2: Read-Only Add-Ons Directory Through Special Mount

The add-ons you select are installed and then rolled out to the entire organization. MATLAB users cannot install any custom add-ons.

1 Create a MATLAB Docker image. See **Prepare** Docker **Images** in “Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2.

2 Mount the add-ons directory through an external mount using Storage profiles.

In the `matlab-pool.yaml` (override), in the storage profile configuration, add the add-ons directory as shown:

```
storage:
  profiles:
    - name: nfsHome
      startDirectory: "/home/${subject.subjectId}"
      mounts:
        - name: home
          permissionType: user
          mountPath: "/home/${subject.subjectId}"
          uid: "${subject.uid}"
          gid: "${subject.gid}"
          type: nfs
          server: "nfs"
          subPath: "/exports/home/${subject.subjectId}"
          mountOptions:
"rw,relatime,vers=3,rsize=1048576,wsize=1048576,namlen=255,acregmi
n=600,acregmax=600,acdirmin=600,acdirmax=600,hard,nocto,noacl,prot
o=tcp,timeo=600,retrans=2,mountproto=tcp,local_lock=none,no
lock"
       - name: addons
          mountPath: /AddOns
          type: nfs
          permissionType: none
          server: "nfs"
          subPath: "/exports/matlab/R2021a/addons"
          mountOptions:
"ro,relatime,vers=3,rsize=1048576,wsize=1048576,namlen=255,acregmi
n=600,acregmax=600,acdirmin=600,acdirmax=600,hard,nocto,noacl,prot
o=tcp,timeo=600,retrans=2,sec=sys,mountproto=tcp,local_lock=none,n
olock"
      restrictNavigationOutsideMounts: false
      addons:
        directory: "/AddOns"
```

Option 3: Writable User Installed Add-Ons Directory

You choose the add-ons directory during user sign in. This directory must have write access for all users.

Using this procedure, MATLAB users can install any custom add-on, but you cannot roll out any add-ons to the users. The add-ons must be installed by the users themselves.

1 Create a MATLAB Docker image. See **Prepare** Docker **Images** in “Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2.

2 In the matlab-pool.yaml (override), in the storage profile configuration, add the add-ons directory as shown:

```
storage:
  profiles:
    - name: nfsHome
      startDirectory: "/home/${subject.subjectId}"
      mounts:
        - name: home
          permissionType: user
          mountPath: "/home/${subject.subjectId}"
          uid: "${subject.uid}"
          gid: "${subject.gid}"
          type: nfs
          server: "nfs"
          subPath: "/exports/home/${subject.subjectId}"
          mountOptions:
"rw,relatime,vers=3,rsize=1048576,wsize=1048576,namlen=255,a
cregmin=600,acregmax=600,acdirmin=600,acdirmax=600,hard,noct
o,noacl,proto=tcp,timeo=600,retrans=2,mountproto=tcp,local_l
ock=none,nolock"
      restrictNavigationOutsideMounts: false
      addons:
        directory: "/home/${subject.subjectId}/MATLAB Add-Ons
```

See Also

Related Examples

- “Install Support Packages for MATLAB in MATLAB Online Server” on page 2-36

Uninstall MATLAB Online Server

The process for uninstalling MATLAB Online Server depends on how you installed the server.

- If you installed the server on a single machine, see “Uninstall Server on Single Machine” on page 2-42.
- If you installed the server on an existing cloud-managed Kubernetes cluster, see “Uninstall Server on Cloud-Managed Kubernetes” on page 2-42.

Uninstall Server on Single Machine

If you installed MATLAB Online Server on a single machine, where the server runs in a single-node Kubernetes cluster, undeploy the cluster by using the `mosadm reset-node` function. You can use this function only if you installed the server using the `mosadm bootstrap-node` function.

```
sudo ./mosadm reset-node
```

This function:

- Stops the `kubelet` program running on the machine
- Removes the Kubernetes cluster running locally
- Resets the IP tables

The `mosadm reset-node` function does not uninstall any third-party Linux packages that the `mosadm bootstrap-node` function installed. These packages include:

- `unzip`
- `gettext`
- `jq`
- `ca-certificates`
- `curl`
- `software-properties-common`
- `git`
- `nfs-common`
- `ipvsadm`
- `kubelet` (1.22.5)
- `kubeadm` (1.22.5)
- `kubectl` (1.22.5)
- `helm` (3.7.0)

For a list of additional components that remain installed, see “Software Components That Remain Installed” on page 2-43.

Uninstall Server on Cloud-Managed Kubernetes

Follow these steps if you installed MATLAB Online Server in a cloud-managed Kubernetes cluster, where the server is one of several nodes in this cluster.

1. Undeploy the server by removing all running services from the Kubernetes cluster.

   ```
   ./mosadm undeploy
   ```

2. (Optional) If you installed the NGINX ingress controller during installation, uninstall the controller.

   ```
   ./mosadm uninstall-ingress
   ```

This procedure does not uninstall all components that were initially installed. For more details, see “Software Components That Remain Installed” on page 2-43.

Software Components That Remain Installed

After you uninstall MATLAB Online Server, these components remain installed:

- Databases connected to the server
- File storage set up for users
- Docker registries
- MathWorks license manager
- MATLAB installations that were mounted to the server

  To uninstall MATLAB, see “Uninstall Products from Linux Systems”.

See Also

Related Examples

- “Contact Technical Support About MATLAB Online Server Issues” on page 5-2
- “Host MATLAB Online on Your Infrastructure” on page 1-3

Setup and Configuration

Set Up MATLAB Online Server After Installation

After installing MATLAB Online Server using the instructions provided in “Installation”, you must perform several post-installation configuration tasks before users can access MATLAB Online.

Configure User Authentication

To authenticate user access in MATLAB Online Server, you must configure an identity provider that MATLAB Online Server can use to authenticate users. The identity provider allows the use of organizational credentials through either a username and password prompt or a single sign-on password.

For details on configuring your identity provider, see “Configure User Authentication in MATLAB Online Server” on page 3-6.

Configure File Storage for Users

By default, user storage is ephemeral. After a user signs out of a session, all data inside the directory is deleted permanently. To enable persistent storage, you can specify storage profiles on your NFS drives. Alternatively, you can use persistent volumes to provide per-user storage, shared storage, or a combination of the two.

To enable persistent storage and to configure which drives, directories, and files are available to users after they log in, see “Configure File Storage for Users in MATLAB Online Server” on page 3-35.

Configure MATLAB

MATLAB Online Server can support multiple versions of MATLAB. These versions can correspond to a specific release (for example, R2023a) or to a release with a specific configuration (for example, R2023a with GPU support). After users sign in, they can select from any MATLAB versions to which they have access through a version chooser UI that you set up.

During this configuration process, at a minimum, you must specify the license server used to check out MATLAB licenses. If your server supports multiple MATLAB versions, you must also:

- Specify which MATLAB documentation release to serve across all MATLAB versions.
- Specify a unique display name for each MATLAB version. This display name appears in the browser title bar and the version chooser UI.
- Configure which user groups are allowed to access the different versions of MATLAB.

For details on configuring one or more MATLAB versions, see “Configure MATLAB in MATLAB Online Server” on page 3-17.

Add Software Dependencies

By default, MATLAB Online Server mounts MATLAB from a Docker image that is embedded in the server. This image does not contain the software dependencies that might be required for certain MATLAB workflows, such as a compiler for code generation. To add these dependencies, see “Add Software Dependencies to MATLAB Image” on page 3-31.

Customize MATLAB Online Login Screen

MATLAB Online Server users can access MATLAB over the web through MATLAB Online as soon as they log in with their company credentials. You can customize this login screen to display, for example, the name and logo of your company. For details, see “Customize MATLAB Online Login Screen” on page 3-4.

Additional Configuration Options

Depending on your server setup, you might need to perform additional configuration steps. For example, you can configure high-availability for the server and control network communication between the server and other applications in your organization. For a complete list of configuration options, see “Setup and Configuration”.

See Also

Related Examples

- “Use MATLAB Online Hosted by Your Organization”
- “Monitor MATLAB Online Server Using Kubernetes Dashboard” on page 4-2

Customize MATLAB Online Login Screen

MATLAB Online Server users can access MATLAB over the web via MATLAB Online as soon as they sign in with the credentials that are configured in “Configure User Authentication in MATLAB Online Server” on page 3-6. The default sign-in page for MATLAB Online, with no customization, is shown in the following image:

You can customize the appearance of this sign-in page by specifying your organization name and logo.

Specify Organization Name

To specify your organization name, edit the settings in the file `all.yaml` and then add the `organizationName` attribute to the global config file.

For example:

```
global:
  organizationName: "MathWorks"
```

Specify Logo

To specify your organization logo, you can choose either to link to the logo or use a logo file.

Link to Logo

Edit the `core-ui.yaml` file and then add the following configuration setting:

```
loginPage:
  logo:
    url: "<url to the logo>"
```

Use Logo File

1. Copy or rename the logo file to `loginPage.logo.file`.

2 Copy this file to the following location before deploying the services:

```
<Overrides_dir>/<cluster>/<namespace>/core-ui/binary/loginPage.logo.file
```

Implement Customized Sign-In Page

To implement the new sign-in page, redeploy MATLAB Online Server using the following command:

```
./mosadm upgrade core-ui
```

The next time you start MATLAB Online from MATLAB Online Server, the sign-in page shows your organization name and logo. The following image is an example of a customized sign-in page for MathWorks MATLAB Online Server on MATLAB Online Server:

See Also

Related Examples

- “Configure MATLAB in MATLAB Online Server” on page 3-17
- “Configure User Authentication in MATLAB Online Server” on page 3-6

Configure User Authentication in MATLAB Online Server

To authenticate user access in MATLAB Online Server, you must configure an identity provider that MATLAB Online Server can use to authenticate and allow users. The identity provider allows the use of user credentials for the organization through a user name/password prompt or single sign-on, depending on the type of Identity provider configured.

At least one identity provider must be configured to sign in to MATLAB Online Server. When you issue the command to generate overrides, the command generates a configuration for the identity provider of type local (see “Local” on page 3-7 for details) with the user name `admin` and `password` that are defined in `install.config`.

To apply the identity provider configuration, edit the settings in the file *<matlab_online_server_install_dir>/overrides/<cluster>/<namespace>/*`authnz.yaml`. For example, `/opt/matlab_online_server/overrides/matlab-online-server/mathworks/authnz.yaml`.

Identity Provider

Identity Provider Specification

Attribute	Default Value	Optional	Description
`identityProviders`	[]	No	The list of identity providers configured for use with MATLAB Online Server

In YAML format:

```
identityProviders:
-    <list of identity provider configurations>
```

Attribute	Default Value	Optional	Description
`name`	""	No	The name of the identity provider
`displayName`	""	Yes	How the identity provider's name is displayed, specified using characters `A-Z`, `a-z`, and `0-9`, with the symbols `_`, `-`, or `.` in between these characters. The maximum `displayName` can be 63 characters long.
`type`	""	No	The type of the identity provider. Based on the type, the identity provider properties vary; see next section.

Identity Provider Types

MATLAB Online Server supports the following types of identity providers:

- LOCAL

- LDAP
- SAML

Local

The LOCAL (**local**) identity provider type sets up the accounts that are in memory. For this identity provider, the accounts must be defined with the properties shown in the following table.

Attribute	Default Value	Optional	Description
`subjectId`	""	No	The userID or user name for the account
`displayName`	""	No	Display name for the account
`password`	""	No	Account password
`groups`	[]	Yes	The groups that the user is associated with For example: [“group1”,”group2”]
`extra`	{}	Yes	The extra metadata that may be added to the user account when they sign in. For example: `{"uid":"1001","department":"MATLAB Online Server"}`

Values in the settings for groups and extra are user-defined and not validated in any way.

In YAML format:

```
identityProviders:
  - id: local
    type: local
    displayName: local
    accounts:
      - subjectId: admin
        displayName: Admin
        password: admin
        groups: ["admin"]
        extra: {}
      - subjectId: stateful
        password: stateful
        extra: {}
```

LDAP

Lightweight Directory Access Protocol, also known as LDAP, is used to manage user identities. Identity management for MATLAB Online Server can apply user information from sources that you specify. The LDAP (**ldap**) identity provider type provides access control and tells MATLAB Online Server how to map user information tables to user information that MATLAB Online Server needs to manage these users.

For more about LDAP settings, visit https://ldap.com/.

The LDAP identity provider type requires extra properties, shown in the following table:

Attribute	Default Value	Optional	Description
`host`	""	No	The host address for the LDAP, for example: "ldapserver.yourcompany.com".
`port`	""	No	Port is the LDAP port at the host address, for example: “389” for regular LDAP and “636” for secure LDAP.
`useStartTls`	false	Yes	The useStartTLS attribute is a Boolean flag that indicates if your LDAP server requires start TLS for connection. You must add the trusted certificates to the MATLAB Online Server trust store. See “Add Certificates to MATLAB Online Server Trust Store” on page 3-11.
`useSsl`	false	Yes	The useSSL attribute is a Boolean flag that indicates if your LDAP server requires SSL for connection. You must add the trusted certificates to the MATLAB Online Server trust store. See “Add Certificates to MATLAB Online Server Trust Store” on page 3-11.
`baseDn`	""	No	The baseDn attribute is the starting point for the search tree in your LDAP environment. For example: “dc=ldap,dc=yourcompany,dc=com

Attribute	Default Value	Optional	Description
`filter`	""	No	Specifies an LDAP search filter. Use the search filter to scan a subset of users from the LDAP database. This allows you to reduce the scope of the LDAP search and control which organizational user may access MATLAB Online Server. • You can leverage variable substitution to configure how the username is provided to LDAP, for example: “(CN={0})”. • Specify the search filter as attribute=value, for example,CN=test*, matches all users that have a common name (CN) attribute that starts with "test". • Use parentheses to combine multiple filter expression in an AND (&) or OR(|) clause. For example, “(&(CN={0})(department=foo))” matches all users that are in department "foo". The default search filter is objectClass=organizationalPerson. For more information on search filters, see LDAP filters at https://ldap.com/ldap-filters.
`adminDn`	""	No	The administrator account used to establish the connection to the LDAP server, for example: “cn=readonly”. This account is used to query LDAP for the list of users. The account requires only read access to the directory and does not attempt to store information back into LDAP.
`adminPassword`	""	No	The administrator account password used to establish the connection to the LDAP server.

Attribute	Default Value	Optional	Description

`subjectAttributeMapping`	`{}`	No	The subject attribute mapping allows configurations to set the values for the user attributes that map to your internal LDAP values.
			
			For example, your user's first name might be stored as *`displayFirstName`* in the LDAP system.
			
			To map this in MATLAB Online Server set `displayName: "displayFirstName"`.
			
			Enclose each `subjectAttributeMapping` value in quotes. Valid fields are as follows:
			
			`subjectAttributeMapping:`
			`  subjectId: "subject_id"`
			`  displayName: "display_name"`
			`  groups: "groups"`
			`  extra:`
			`  email: "email_address"`
			`  uid: "uid_number"`
			`  gid: "gid_number"`
			
			`displayName` is an optional attribute that controls what is displayed for the username in the upper-right corner of MATLAB Online Server.
			
			The `subjectId` is the default mapping field for the user account, normally "cn".
			
			The extra parameters are for additional customization relating to the network files system (NFS):
			
			• extra.uid – the mapping to a Unix UID id

In YAML format:

```
identityProviders:
  - id: "ldap"
    type: "ldap"
    displayName: "LDAP"
    host: "ldap"
    port: "636"
    useStartTls: true
    useSsl: true
    baseDn: "dc=matlabonlineserver,dc=mwcloudtest,dc=com"
    filter: "(uid={0})"
    adminDn: "cn=readonly,dc=matlabonlineserver,dc=mwcloudtest,dc=com"
    adminPassword: "readonly"
    subjectAttributeMapping:
      subjectId: "uid"
      displayName: "cn"
      groups: "groups"
      extra:
        uid: "uidNumber"
```

Add Certificates to MATLAB Online Server Trust Store

When you use LDAPS or LDAP with TLS, you must add the certificates to the global trust store. These certificates are validated against the LDAP/LDAPS server to establish a secure connection.

You must add the certificate information, shown in the following table, to authnz.yaml.

Attribute	Default Value	Optional	Description
`validateCertificate`	true	Yes	`validateCertificate` property enables the certificate verification for the authnz service while communicating to the external identity providers. As an example, when LDAP is being used with useSSL= true or useTLS = true, the secure communication will be validated. If this property is disabled, the communication will still be LDAPS, but the certificate verification will be skipped.

Attribute	Default Value	Optional	Description
`trustedCertificates`	""	Yes	The server certificates that are required for communicating with the external identity providers in the authnz service must be provided using the `trustedCertificates` property. For example, when LDAP is being used with useSSL= true or useTLS = true, the secure communication must be validated and server certificates must be provided using this property. If the certificates are not provided, the LDAPS communication fails.

In YAML format:

```
global:
  tls:
    validateCertificate: true
    trustedCertificates: |
          -----BEGIN CERTIFICATE-----
          MIIC5DCCAk2gAwIBAgIUZhnmeMmayNSTcAI2hgyxQt6GotUwDQYJKoZIhvcNAQEL
          ...
          VA/d/fQ+yxUjlDBc6ly/OwVFtIr0QykE
          -----END CERTIFICATE-----
          -----BEGIN CERTIFICATE-----
          BQAwgYMxCzAJBgNVBAYTAlVTMQswCQYDVQQIDAJNQTEPMA0GA1UEBwwGTmF0aWNr
          ...
          EdUg4pCYWUyFgGA/QCg4EniQEMN
          -----END CERTIFICATE-----
```

The certificates must be formatted using PEM. For details on PEM, see:

- PEM format requirements for certificates and domain keys
- PEM, DER, CRT, and CER: X.509 Encodings and Conversions

SAML

Security Assertion Markup Language, also known as SAML, is used to communicate with an identity provider. The SAML configuration for MATLAB Online Server ensures that the identities used to control access to MATLAB Online Server are managed by the Enterprise Identify Provider.

For more information about SAML settings, see https://wiki.oasis-open.org/security/.

The SAML identity provider type has extra properties, as shown in the following table:

Attribute	Optional or Required	Description
`assertionConsumerPath`	Required	The assertion consumer service (ACS) endpoint is a location where the SSO tokens are sent. ACS is applicable to all SAML versions and both the IdP-initiated and SP-initiated SSO profiles. **Default**: `"/authnz/saml/code"`
`corsAllowOriginDomain`	Required	The fully qualified domain name of the server that communicates with MATLAB Online Server, for example, `samlintegrationhost.yourcompanydomain.com`. **Default**: `""`
`relyingPartyId`	Required	The hostname for the server that integrates with the SAML Identity Provider. This value is typically the hostname of the DNS entry for your MATLAB Online Server Kubernetes cluster, but depending on the integration product or service being used to expose the Identity Provider via the SAML system, this value can also be an arbitrary identifier. **Default**: `""`

Attribute	Optional or Required	Description

`subjectAttributeMapping`	Required	The subject attribute mapping lets you set values for the user attributes that map to your organizational SAML assertion values.
		
		For example, your user's first name might be stored as *displayFirstName* in the SAML system.
		
		To map this in MATLAB Online Server, set `displayName: displayFirstName`.
		
		Enclose each `subjectAttributeMapping` field value in quotes. Valid fields are as follows:
		
		`subjectAttributeMapping:`
		`  subjectId: "subject_id"`
		`  displayName: "display_name"`
		`  groups: "groups"`
		`  extra:`
		`  email: "email_address"`
		`  uid: "uid_number"`
		`  gid: "gid_number"`
		
		`displayName` is an optional attribute that controls what is displayed for the user name in the upper-right corner of MATLAB Online Server.
		
		`subjectId` is the default mapping field for the user account. This value comes from the SAML assertions sent as part of a successful login.
		
		`groups` is an optional attribute that defines what groups the user belongs to.
		
		`extra` contains customization fields relating to the network file system (NFS):
		
		• `email` — Email address of user
		• `uid` — the mapping to a Unix user ID (UID)
		• `gid` — the mapping to a Unix group ID (GID)
		
		Default: `{}`

`supportIdpInitiated`	Optional	Specify as `true` if you are using an IdP-initiated SSO profile. For SP-initiated profiles, set as `false` or omit this field. **Default**: `""`

Attribute	Optional or Required	Description
`idpIssuer`	Optional (Required if `supportIdpInitiated` is `true`)	IdP issuer for IdP-initiated relay state. This value has the format *`idpIssuerName`*`|`*`idpRedirectUrl`*, where *`idpIssuerName`* is an arbitrary string used to identify the IDP issuer, and *`idpRedirectUrl`* is the URL that the issuer redirects users to. Separate these values with a pipe character (`|`). For example: `"idp-issuer-name|https://<IdP issuer redirect URL>"`. This field applies only when `supportIdpInitiated` is set to `true`. **Default**: `""`
`idpMetadataUrl`	Optional (You must specify either `idpMetadataUrl` or `idpMetadataXml`, but not both)	URL to the required SAML metadata for SP-initiated or IdP-initiated profiles. **Default**: `""`
`idpMetadataXml`	Optional (You must specify either `idpMetadataUrl` or `idpMetadataXml`, but not both)	The value of `idpMetadataXml` is a block of XML that is exported by the SAML Identity Provider. You can paste the exported XML into this field, but the field must be properly formatted YAML. Otherwise, `mosadm` is unable to deploy the AuthNZ details. Depending on the integration product or service being used to expose the Identity Provider via SAML, there might be newline characters. Delete those newline characters from the metadata XML before pasting the code into `authnz.yaml`. **Default**: `""`

In YAML format:

```
identityProviders:
  - id: saml
    type: saml
    assertionConsumerPath: "/service/assertionConsumer"
    corsAllowOriginDomain: "samlintegrationhost.yourcompanydomain.com"
    relyingPartyId: "matlab.domain.com"
    idpMetadataXml: |
        <xml content/>
    subjectAttributeMapping:
      displayName: "name"
      subjectId: "myNameId"
      groups: "groups"
      extra:
        email: "email"
        uid: "uidNumber"
        gid: "gidNumber"
    supportIdpInitiated: true
    idpUsesHttps: true
    idpIssuer: "idp-issuer-name|https://<IdP issuer redirect URL>"
    idpMetadataUrl: "https://<IdP metadata>"
```

This code specifies both `idpMetadataUrl` and `idpMetadataXml` for illustrative purposes. For your YAML file, specify only one or the other of these fields.

See Also

External Websites

- https://en.wikipedia.org/wiki/Authentication
- https://en.wikipedia.org/wiki/Transport_Layer_Security

Configure MATLAB in MATLAB Online Server

MATLAB Online Server enables you to configure the versions of MATLAB that users can access and the settings for those versions. To set up and configure MATLAB:

1. Update the MATLAB Pool YAML configuration files that correspond to the versions of MATLAB you are setting up.
   - If the server runs a single version of MATLAB, see “Configure Single Version of MATLAB” on page 3-17.
   - If the server runs multiple MATLAB versions, see “Configure Multiple Versions of MATLAB” on page 3-21.
2. If the server runs multiple MATLAB versions, specify which release of the MATLAB documentation to serve across all versions. See “Specify MATLAB Documentation” on page 3-24. MATLAB Online Server does not support specifying different MATLAB documentation releases for each version.
3. Deploy the changes. See “Deploy MATLAB Configuration” on page 3-24.

Prerequisites

- You installed MATLAB Online Server and installed at least one version of MATLAB using one of the procedures listed under “Installation”.
- You are familiar with core MATLAB Online Server concepts, such as MATLAB pools, pods, and YAML override files. For more details, see “Host MATLAB Online on Your Infrastructure” on page 1-3.

Configure Single Version of MATLAB

To configure a single version of MATLAB running on the server, update the MATLAB Pool settings file that corresponds to that version of MATLAB. The file is located at this path.

`server_root/overrides/cluster/namespace/matlab-pool.yaml`

Here:

- *`server_root`* is the root folder of your MATLAB Online Server installation.
- *`cluster`* is the name of your Kubernetes cluster.
- *`namespace`* is the Kubernetes namespace you used to deploy MATLAB Online Server.

By modifying the YAML fields in this file, you can override the default MATLAB settings. This table shows the available configuration settings.

Configuration	Instructions

Configure License Server. Configure the MathWorks License Manager server used to check out MATLAB licenses.	In the `matlab-pool.yaml` file, in the `flexlm` field, update the `servers` subfield with the license server port and address information. Use the format `port@address`. For example:
	`flexlm:`
	`  servers: "27000@flexlm.company.com"`
	To list multiple servers, separate the server information with a colon. For example:
	`flexlm:`
	`  servers: "27000@flexlm1.company.com:27000@flexlm2.company.com"`

Configure Prewarmed MATLAB Instances. Configure the number of prewarmed MATLAB instances that are ready for assignment when a user logs in.	In the `matlab-pool.yaml` file, update the `replicaCount` field. `replicaCount: 2` The default count of 2 means that two MATLAB pods are ready for assignment. For example, suppose two users log in at the same time. Each pod assigned to them contains a MATLAB instance that is ready to use within seconds of logging in. If two more users log in at the same time, their logins are delayed until two new pods can start up. The maximum number of active users that can log in at one time is limited to the number of MATLAB licenses. The total number of MATLAB instances that can be assigned is limited to the number of licenses available and the hardware configured for the MATLAB Pool. If the server is unable to assign a MATLAB pod to a user, then the user receives a message that no MATLAB instances are available at that time and to try logging in later.

Configuration	Instructions

Customize Title Bar. Customize the MATLAB title bar text that displays on the browser tab when a user signs in.	In the `matlab-pool.yaml` file, update the `displayName` field. For example:
	`poolConfig:`
	`  displayName: "R2023a"`
	This field accepts numbers, letters, dashes (`-`), underscores (`_`), and periods (`.`). For example: `"R2023a_4-CPU_16-GB"`.
	After the server establishes a MATLAB session, the display name appears as the title of the browser tab in this format:
	MATLAB Online – R2023a

Configure Network Ports. Configure additional ports to enable MATLAB to have access to.	In the `networkPolicy` field of the `matlab-pool.yaml` file, uncomment the `additionalAllowedPorts` and specify the additional ports and optional protocols allowed. For example:
	`networkPolicy:`
	`  enabled: true`
	`  additionalAllowedPorts:`
	`  - port: 1433`
	`  protocol: TCP`
	`  - port: 1433`
	`  protocol: UDP`

Set MATLAB Installation Source. Specify the source of the MATLAB Pool pod installation. MATLAB Online Server uses this information to mount the MATLAB installation so that it can interact with it.	In the `matlab-pool.yaml` file, update the `mountType` field to one of the mounting options described in “Set MATLAB Installation Source” on page 3-26.
Configure Storage. Configure the drives, directories, and files that are available to users after they log in to MATLAB.	In the `matlab-pool.yaml` file, define storage profiles for each resource available to users in separate `storage` fields, as described in “Configure File Storage for Users in MATLAB Online Server” on page 3-35.
Configure Hardware Resources. Allocate hardware resources and provide limits for memory and CPU configuration of MATLAB.	In the `matlab-pool.yaml` file, update the `resourceQuota` field, as described in “Configure MATLAB Hardware Resources” on page 3-29.

Configuration	Instructions

Configure Authorization. Customize which user groups can access MATLAB.	In the `matlab-pool.yaml` file, in the `resourceDefinition` section, update the `allowedGroups` field. For example:
	`resourceDefinition:`
	`  allowedGroups:`
	`  - "team1"`
	`  - "team3*"`
	This sample YAML enables access to any `"team1"` user and any user whose team name starts with `"team3"` (`team31`, `team32`, `team3abc`, and so on) by using a wildcard pattern. MATLAB Online Server uses the glob pattern, which is an industry standard for specifying wildcard patterns. For more details, see https://en.wikipedia.org/wiki/Glob_(programming).

Configure MATLAB Session Timeouts. Configure how long before MATLAB sessions time out.	In the `matlab-pool.yaml` file, in the `resourceDefinition` section, update these fields. Units are in minutes.
	`resourceDefinition:`
	`  maxLifetimeMinutes: 240`
	`  defaultLifetimeMinutes: 120`
	`  maxInactivityTimeoutMinutes: 30`
	`  defaultInactivityTimeoutMinutes: 15`
	For more details, see “Configure MATLAB Session Settings” on page 3-59.

<table>
<tr><th>Configuration</th><th>Instructions</th></tr>
<tr><td>Set Environment Variables and Metadata. Set MATLAB environment variables and pod metadata.</td><td>In the <code>matlab-pool.yaml</code> file, in the <code>resourceDefinition</code> section:
To set Kubernetes tags, update the <code>podLabels</code> field.
To set Kubernetes annotations, update the <code>podAnnotations</code> field.
To set MATLAB environment variables, update the <code>env</code> field.
Use this sample <code>matlab-pool.yaml</code> code and update the field values for your MATLAB pod.<pre>resourceDefinition:
  ownerMetadata:
    podLabels:
      - name: username
        value: ${subject.subjectId}
      - name: environment
        value: "prod"
    podAnnotations:
      - name: username
        value: ${subject.subjectId}
    env:
      - name: MOS_USERNAME
        value: ${subject.subjectId}
      - name: environment
        value: "prod"</pre>The <code>${subject.subjectId}</code> format enables the AuthNZ service to dynamically populate the user metadata when that user signs in. For more details about how placeholders work, see “Placeholders” on page 3-48.</td></tr>
</table>

Configure Multiple Versions of MATLAB

MATLAB Online Server can support multiple versions of MATLAB. When users have access to multiple MATLAB versions, after logging in, they are provided an option to choose the MATLAB version they want to use (for example, R2023a, R2023a with GPU, R2023b).

The workflow for configuring multiple MATLAB versions is as follows:

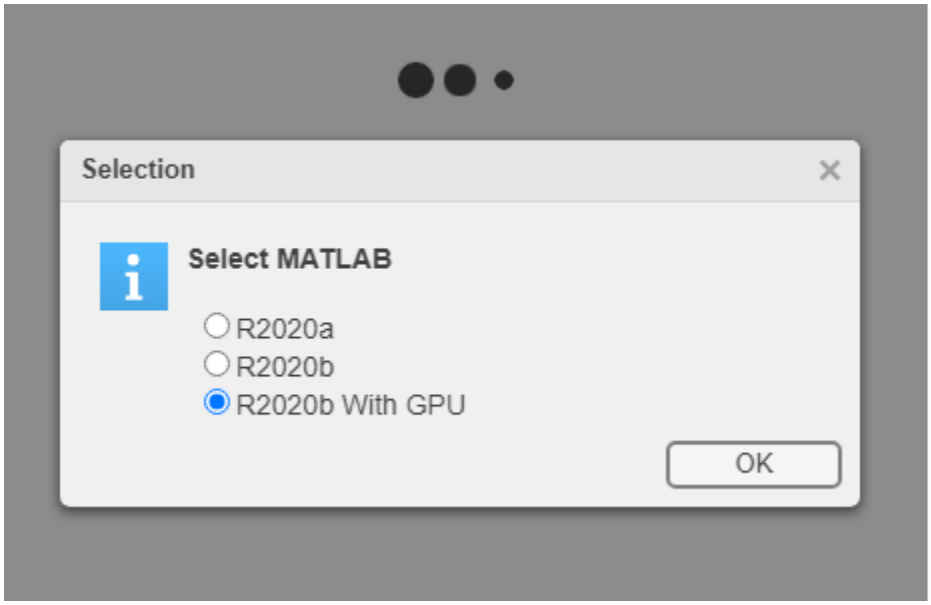
1. Enable the MATLAB version chooser UI that end users see after logging in to MATLAB.
2. Create the folder structure used to store the configuration files for each MATLAB version.
3. Configure the MATLAB version that was installed during the MATLAB Online Server installation.
4. Install and configure additional MATLAB versions.

Enable MATLAB Version Chooser UI

The MATLAB version chooser UI enables MATLAB end users to select their MATLAB version after signing in to MATLAB Online. To enable the chooser UI, in the `core-ui.yaml` override file, set this field:

```
multiplePoolSupport:
  enabled: true
```

The options that end users can choose from depend on how many and what kind of pools you create. This sample UI chooser enables MATLAB end users to select from three different versions of MATLAB Online.



When the end user clicks **OK**, MATLAB Online Server launches the selected version of MATLAB in the browser.

Create MATLAB Pool Folder Structure

To support multiple versions of MATLAB, instead of specifying a single `matlab-pool.yaml` file, you must create a separate MATLAB Pool YAML file for each version. These files must reside in a folder named `matlab-pool`.

To set up this folder structure:

1. Navigate to the folder that contains the `matlab-pool.yaml` file. For example:

   ```
   cd ~/matlab_online_server/overrides/matlab-online-server/mathworks
   ```

2. Create a directory named `matlab-pool`.

   ```
   mkdir matlab-pool
   ```

3. Move the file `matlab-pool.yaml` to the `matlab-pool` directory.

   ```
   mv matlab-pool.yaml matlab-pool
   ```

Configure Existing MATLAB Version

Configure the version of MATLAB that you installed when you first installed MATLAB Online Server.

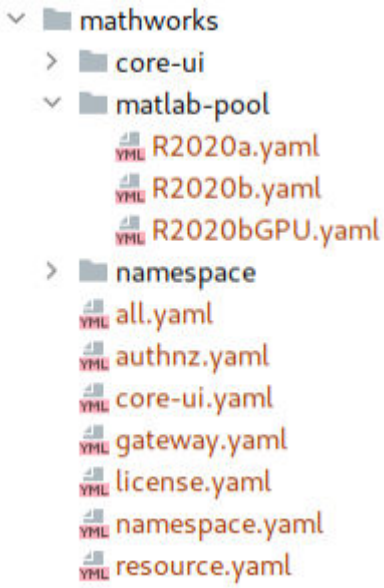
1. Rename the existing `matlab-pool/matlab-pool.yaml` file to `matlab-pool/poolname.yaml`, where *`poolname`* is a descriptive name of the pool. Typically, this name corresponds to the name of a MATLAB release, along with any additional characters that describe the pool. For example:
   - `r2023a.yaml`
   - `r2023agpu.yaml`
   - `r2023b.yaml`
   - `r2023bgpu.yaml`
2. In the renamed YAML file, configure any options for this version of MATLAB. For a list of configurable options, see “Configure Single Version of MATLAB” on page 3-17.
3. Configure the display name of the version that users see in the version chooser UI when they log in. The display name also appears in the title bar of the MATLAB Online browser. In the YAML file for this MATLAB pool, uncomment the `poolConfig` section and update the `displayName` field with a unique name for the release.

```
poolConfig:
  displayName: "R2023a"
```

Install and Configure Additional MATLAB Versions

After you update the first version of MATLAB, you can install and configure the additional versions of MATLAB. Repeat these steps for as many MATLAB versions as you plan to support.

1. On the server, install the version of MATLAB you want to create a pool for. For installation instructions, see “Install Products Using Internet Connection”.
2. Copy the YAML file you configured in “Configure Existing MATLAB Version” on page 3-23. Rename this file based on the release and supported features of the MATLAB version. For example, this sample `matlab-pool` folder supports three different versions of MATLAB.



3. In the copied YAML file, configure any options for this new MATLAB pool. For a list of configurable options, see “Configure Single Version of MATLAB” on page 3-17.
4. In the YAML file, update the `displayName` field with the appropriate display name for that version. This name must be unique for each MATLAB pool you create.

Specify MATLAB Documentation

If your server includes only one version of MATLAB, skip this step. If the server supports only one version, then it serves the MATLAB documentation corresponding to its release.

If your server supports multiple MATLAB versions, then you must specify which release of the MATLAB documentation you want all versions to serve. MATLAB Online Server does not support serving a different help release for each MATLAB version.

1. Select the version of MATLAB (for example, R2023a) from which you want to serve the help content. Open the respective YAML override file (for example, `r2023a.yaml`), and add the following field.

   ```
   includeHelpSearch: true
   ```

2. Configure all MATLAB pools to use the help content from the selected version. In the `all.yaml` file, add the following configuration.

   ```
   extraMatlabStartupFlags: -r "docRelease=com.mathworks.mlservices.MLHelpServices.setDocRelease('RNNNNx');
   docRelease=com.mathworks.mlservices.MLHelpServices.getDocRelease;clear docRelease;"
   includeHelpSearch: false
   ```

   Replace *RNNNNx* with the MATLAB version you use for the help content, for example, R2023a.

Deploy MATLAB Configuration

To have your MATLAB configuration changes take effect, you must redeploy the MATLAB Pool configurations you updated. MATLAB users receive the changes the next time they sign on.

To redeploy all versions of MATLAB running on the server, use the `mosadm upgrade` command.

```
./mosadm upgrade matlab-pool
```

- If the server hosts only one version of MATLAB, then this command redeploys the `matlab-pool.yaml` file for that version.
- If the server hosts multiple versions of MATLAB, then this command redeploys all the YAML files for those versions found in the `matlab-pool` folder.

To deploy newly installed versions of MATLAB, use the `--install` flag to deploy only that version and keep all currently deployed versions running on the server.

```
./mosadm upgrade matlab-pool --install
```

If you modified YAML overrides files in addition to the MATLAB pool files, such as the `core-ui.yaml` file when specifying multiple MATLAB versions, then you can redeploy the entire server by using `mosadm deploy` and `mosadm undeploy`.

```
./mosadm undeploy
```

```
./mosadm deploy
```

See Also

`mosadm build-matlab-image` | `mosadm upgrade`

Related Examples

- “Set Up MATLAB Online Server After Installation” on page 3-2
- “Add Software Dependencies to MATLAB Image” on page 3-31
- “Configure User Authentication in MATLAB Online Server” on page 3-6
- “Customize MATLAB Online Login Screen” on page 3-4
- “Configure Network Policies” on page 3-62

Set MATLAB Installation Source

Specify the source of the MATLAB Pool pod installation in the `mountType` field of the `matlab-pool.yaml` file. MATLAB Online Server uses this information to mount the MATLAB installation so that it can interact with it.

This table shows the available configuration options.

MATLAB Installation Source	`mountType` Value	Description
Container image	`"embedded"` (default)	• Enables easy export of MATLAB installation across Kubernetes clusters. • Requires large image installation. See "Mount MATLAB from Built Container Image" on page 3-27.
One or more Kubernetes nodes	`"host"`	• Enables use of existing MATLAB installation, without a large image installation. • Export across Kubernetes clusters is more difficult. See "Mount MATLAB from Kubernetes Nodes" on page 3-27.
NFS server	`"nfs"`	• Enables use of existing MATLAB installation, without a large image installation. • Performance might be slower because installation is on a network drive. See "Mount MATLAB from NFS Server" on page 3-27.
Persistent volume claim (PVC)	`"pvc"`	• Can be used with cloud-provided Kubernetes clusters. • Performance might be slower because installation is on a network drive. See "Mount MATLAB from Persistent Volume Claim" on page 3-28.

Mount MATLAB from Built Container Image

With this mounting configuration, you build a MATLAB container image and make the image available inside MATLAB Online Server. This is the default option.

Use this configuration to make it easy to export the MATLAB installation across Kubernetes clusters, such as when:

- Exporting to different cloud-provided Kubernetes clusters
- Working across different Kubernetes clusters within the same version of MATLAB

For details on building a MATLAB image in the container, see `mosadm build-matlab-image`.

To use this configuration, set these `matlab-pool.yaml` fields:

```
matlab:
  installation:
    mountType: "embedded"
```

To specify the container image that MATLAB Online Server uses, set these fields:

```
images:
  matlabEmbedded:
    image: "com.mathworks.matlabonlineserver.matlab-image"
    tag: "R2023a"
    registry:
    repository:
```

- Update `image` and `tag` with the information for your image.
- To use the global image settings, leave `registry` and `repository` empty.

Mount MATLAB from Kubernetes Nodes

If MATLAB is installed on the same machine as your single-node or multinode Kubernetes cluster, you can make the MATLAB installation available to MATLAB Online Server directly.

For cloud-provided Kubernetes installations, unless the nodes are customized to support this configuration, this mounting type is not recommended.

To use this configuration, set these `matlab-pool.yaml` fields:

```
matlab:
  installation:
    mountType: "host"
    path: "/MATLAB"
```

- Update `path` with the path to the MATLAB installation folder. For multinode Kubernetes installations, the `path` value must be the same across all nodes.

Mount MATLAB from NFS Server

Use this configuration when the MATLAB installation is on an NFS server that the Kubernetes cluster can access. This option makes the MATLAB installation directly available to MATLAB Online Server without having to build a large image.

Performance might be slower with this option when compared to the direct installation options, because MATLAB Online Server loads MATLAB from a network location.

To use this configuration, set these `matlab-pool.yaml` fields:

```
matlab:
  installation:
    mountType: "nfs"
    server: "nfs-server"
    path: "/exports/MATLAB/R2023a"
```

- Update `server` with the IP address of the NFS server of a fully-qualified server name.
- Update `path` with the subpath on the NFS server that contains the MATLAB installation.

Mount MATLAB from Persistent Volume Claim

If the previous configurations are not available, you can make the MATLAB installation available through a Kubernetes Persistent Volume Claim (PVC). You can back the PVC with a different storage provider by using a Container Storage Interface (CSI). For more details on PVC mounting, see “Persistent Volume Claim (PVC)” on page 3-44.

Consider using this option for cloud-provided Kubernetes installations. Performance might be slower with this option when compared to the direct installation options, because MATLAB Online Server loads MATLAB from a network location.

To use this configuration, set these `matlab-pool.yaml` fields:

```
matlab:
  installation:
    mountType: "pvc"
    claimName: "matlab-r2023a-pvc"
    path: ""
```

- Update `claimName` with the name of the PVC where the MATLAB installation is available. For multinode Kubernetes clusters, the PVC must be available to all nodes.
- Update `path` with the path within the PVC where the MATLAB installation is available.

See Also

Related Examples

- “Configure MATLAB in MATLAB Online Server” on page 3-17
- “Configure File Storage for Users in MATLAB Online Server” on page 3-35

Configure MATLAB Hardware Resources

When new MATLAB and storage use cases arise, you might need to change the hardware resources allocated to MATLAB Online Server service. To change the hardware resource allocation, update the memory and CPU configuration settings in the `matlab-pool.yaml` file.

The MATLAB pool has two containers that control the hardware resources allocated:

- The resource-proxy container provisions the storage for each user session.
- The MATLAB container assigns MATLAB compute resources to each user.

The default memory and CPU configuration for these containers are as follows:

```
resourceQuota:
  matlab-pool-resource-proxy:
    requests:
      cpu: "200m"
      memory: "150Mi"
    limits:
      cpu: "500m"
      memory: "512Mi"
  matlab-pool-matlab:
    requests:
      cpu: "200m"
      memory: "2Gi"
    limits:
      #cpu:
      #memory:
```

With these default settings:

- The Resource-Proxy container requests a minimum of 200m CPU and 150Mi of memory, with limits of 500m CPU and 512Mi of memory on the node.
- The MATLAB container requests a minimum of 200m CPU and 2Gi of memory on the node, with no limit set on these resources. To add resource limits, remove the comment markers (#) and add values appropriate for your environment and infrastructure. For more details, see “Add Limits to MATLAB Container” on page 3-29.

For more about CPU and memory configuration in Kubernetes, see these links:

- CPU: https://kubernetes.io/docs/concepts/configuration/manage-compute-resources-container/#meaning-of-cpu
- Memory: https://kubernetes.io/docs/concepts/configuration/manage-compute-resources-container/#meaning-of-memory
- Resource Management: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers

Add Limits to MATLAB Container

Because the default MATLAB container configuration has no resource limits specified, it is possible for a container to use up all CPU and memory on the node.

To set a limit on CPU and memory requests, in the `matlab-pool.yaml` file, uncomment the `cpu` and `memory` fields for the MATLAB container and set CPU and memory values appropriate for your environment and infrastructure. For example:

```
resourceQuota:
  matlab-pool-matlab:
    requests:
      cpu: "200m"
      memory: "2Gi"
    limits:
      cpu: "2000m"
      memory: "7Gi"
```

With these settings, the MATLAB container requests a minimum of 200m CPU and 2Gi of memory and can request a maximum of 2000m CPU and 7Gi of memory. Therefore, the container cannot use all the resources on the node.

Increase Limits of Resource-Proxy

If the amount of data that is mounted through storage profiles is excessive, such as more than 10 TB, the underlying Linux storage mechanism, FUSE, might need more memory to successfully provision the storage and required permissions.

To increase the limit on requests for CPU and memory, in the `matlab-pool.yaml` file, specify new `cpu` and `memory` field values that are appropriate for your environment and infrastructure. For example:

```
resourceQuota:
  matlab-pool-resource-proxy:
    requests:
      cpu: "250m"
      memory: "300Mi"
    limits:
      cpu: "750m"
      memory: "1Gi"
```

With these settings, the Resource-Proxy container requests a minimum of 250m CPU and 300Mi of memory and can request a maximum of 750m CPU and 1Gi of memory. Therefore, the storage provisioner can use more hardware resources than the default maximum of 500m CPU and 512Mi of memory.

See Also

Related Examples

- “Configure MATLAB in MATLAB Online Server” on page 3-17

Add Software Dependencies to MATLAB Image

By default, the MATLAB Docker image does not contain the software dependencies required for certain MATLAB workflows, such as a compiler for code generation. You can add such dependencies by customizing the MATLAB Docker image.

Prerequisites

- You installed MATLAB Online Server and loaded the default MATLAB image from the MathWorks container registry.
- Docker is installed on your machine. Familiarity with Dockerfiles is helpful but not required. For more details, see Dockerfile reference.

Note These instructions prefix `docker` and `mosadm` commands with `sudo`. Depending on how your environment is configured, `sudo` administration privileges might not be required in all cases.

Prepare MATLAB Image

1. Navigate to your MATLAB Online Server folder. For example:

   ```
   cd ~/matlab_online_server
   ```

2. List the Docker images loaded into your MATLAB Online Server installation.

   ```
   sudo ./mosadm list-docker-images
   ```

   Verify that the list includes these images:

   - The base image containing the libraries required to run MATLAB, such as the UI display and window manager. The tag you see after the colon (`:`) might vary.

     ```
     containers.mathworks.com/matlab-online-server/mos-matlab-omnibus-image:1.13.0-bionic
     ```

   - The image containing MATLAB itself. This image is a child of the base MATLAB image. The tag you see might vary.

     ```
     containers.mathworks.com/matlab-online-server/mos-matlab-image:1.13.0
     ```

   If you do not see these images listed, reload the Docker images that ship with the MATLAB Online Server installer.

   ```
   sudo ./mosadm load-docker-images
   ```

3. Create a backup of the base image using this Docker command. Run this command as a single line. If necessary, update the tag name to the one you see.

   ```
   sudo docker tag containers.mathworks.com/matlab-online-server/mos-matlab-omnibus-image:1.13.0-bionic
     containers.mathworks.com/matlab-online-server/mos-matlab-omnibus-image:1.13.0-bionic-backup
   ```

Create Dockerfile Template

A Dockerfile is a text file containing commands that Docker executes prior to building the image. In this file, you can specify commands that build all the necessary dependencies for the MATLAB image.

1. In your `matlab_online_server` folder, create a folder named `docker` and navigate into it.

```
mkdir docker && cd docker
```

2 Create a file named `Dockerfile` (no extension) and open it for editing.

```
touch Dockerfile && nano Dockerfile
```

3 Create a template Dockerfile by copying these lines into the file.

```
FROM containers.mathworks.com/matlab-online-server/mos-matlab-omnibus-image:1.13.0-bionic
USER root

# START IMAGE CUSTOMIZATION



# END IMAGE CUSTOMIZATION

USER matlab
WORKDIR /home
CMD ["/usr/local/bin/startup.sh"]
```

This table describes the actions that these commands perform.

Command	Description
`FROM mathworks/...image:1.13.0-bionic`	Specifies that you are building from the MATLAB image. Docker applies the customization commands to this image.
`USER root`	Temporarily enables root administration privileges so that Docker can execute the software dependency commands. Depending on your environment configuration and the commands specified, these privileges might not be necessary. After executing the Dockerfile, Docker disables these privileges.

`USER matlab`
`WORKDIR /home`
`CMD ["/usr/local/bin/startup.sh"]`
	Enables the Docker image to build inside MATLAB Online Server. These lines must be at the end of the Dockerfile.

Add Software Dependency Commands

In the body of the Dockerfile, write the commands that add the software dependencies. These samples Dockerfiles show dependencies you can add for common MATLAB workflows. They are written for the Ubuntu operating system, but you can perform similar commands in other Linux operating systems.

Install Required Software Package

In this file, replace *`package`* with the name of the software package you want to install (for example, `curl`).

```
FROM containers.mathworks.com/matlab-online-server/mos-matlab-omnibus-image:1.13.0-bionic

USER root

# START IMAGE CUSTOMIZATION

RUN apt-get update && apt-get install -y package

# END IMAGE CUSTOMIZATION

USER matlab
WORKDIR /home
CMD ["/usr/local/bin/startup.sh"]
```

Install GCC Compiler to Enable MEX Workflows

These commands copy the gcc binaries onto your machine, set the necessary environment variables, and install the gcc compiler.

```
FROM containers.mathworks.com/matlab-online-server/mos-matlab-omnibus-image:1.13.0-bionic

USER root

# START IMAGE CUSTOMIZATION

ENV GCC_VERSION gcc-6.3.0

# Copy gcc-6.3 binaries from the gcc:6.3 image
COPY --from=gcc:6.3 /usr/local/bin /usr/bin/${GCC_VERSION}/bin
COPY --from=gcc:6.3 /usr/local/include /usr/bin/${GCC_VERSION}/include
COPY --from=gcc:6.3 /usr/local/lib/gcc /usr/bin/${GCC_VERSION}/lib/gcc
COPY --from=gcc:6.3 /usr/local/lib64 /usr/bin/${GCC_VERSION}/lib64
COPY --from=gcc:6.3 /usr/local/libexec /usr/bin/${GCC_VERSION}/libexec
COPY --from=gcc:6.3 /usr/local/share /usr/bin/${GCC_VERSION}/share

ENV GCC_HOME /usr/bin/${GCC_VERSION}/bin/

RUN update-alternatives --install /usr/bin/gcc gcc ${GCC_HOME}/gcc 90 \
    && update-alternatives --install /usr/bin/g++ g++ ${GCC_HOME}/g++ 90 \
    && update-alternatives --install /usr/bin/gfortran gfortran ${GCC_HOME}/gfortran 90

# END IMAGE CUSTOMIZATION

USER matlab
WORKDIR /home
CMD ["/usr/local/bin/startup.sh"]
```

Customize MATLAB and Java Options at Startup

These commands add scripts to the Docker image. Here, it is assumed that the MATLAB startup script (`matlabrc.m`) and Java options file (`java.opts`) are in the same folder as your Dockerfile.

```
FROM containers.mathworks.com/matlab-online-server/mos-matlab-omnibus-image:1.13.0-bionic

USER root

# START IMAGE CUSTOMIZATION

COPY matlabrc.m /MATLAB/toolbox/local/matlabrc.m
COPY java.opts /MATLAB/bin/glnxa64/java.opts

# END IMAGE CUSTOMIZATION

USER matlab
WORKDIR /home
CMD ["/usr/local/bin/startup.sh"]
```

Build MATLAB Image

After you update the Dockerfile with all required dependencies, build the base MATLAB image from the folder containing the Dockerfile. Make sure you have required disk space on the Docker data disk.

```
sudo docker build . --tag containers.mathworks.com/matlab-online-server/mos-matlab-omnibus-image:1.13.0-bionic
```

If you mounted MATLAB from an image available inside MATLAB Online Server (the default configuration), then you must also rebuild the MATLAB image. If you mounted MATLAB from an external source, skip this step. For information on configuring the MATLAB installation source, see “Set MATLAB Installation Source” on page 3-26.

```
cd ..
sudo ./mosadm build-matlab-image /MATLABInstallPath
```

If you are using a remote Docker registry, push the customized images to your remote registry by using the `mosadm push-docker-images` command.

See Also

`mosadm load-docker-images` | `mosadm list-docker-images` | `mosadm push-docker-images`

Related Examples

- “Configure MATLAB in MATLAB Online Server” on page 3-17

External Websites

- Dockerfile reference

Configure File Storage for Users in MATLAB Online Server

A storage profile is a specification defined by the MATLAB Online Server administrator that determines how and which drives, directories, and files are available to the resource (MATLAB) after users sign on. The storage profiles are defined in the storage section of your `matlab-pool.yaml` file. These profiles are mounted when the user signs in and the remote resource (MATLAB) is allocated to them.

The login information provides more data regarding the user at the time of mounting the profile to the Storage service (part of the MATLAB pod). This information is used to resolve and mount user-specific directories.

Storage profiles allow you to specify this dynamic configuration through the use of *placeholders*. These placeholders are resolved dynamically while the directories are mounted with the same information from a user who has signed in (from the AuthNZ service). For more information about placeholders, see “Placeholders” on page 3-48.

Depending on your configuration of storage profiles, additional training for the MATLAB user might be required. If the MATLAB user saves data in the persistent folders or mounts, then when the user signs into a new session, all saved data is restored. However, data that the user stores in a nonpersistent folder, such as `/tmp`, does not persist to the next session. Familiarize your users with the concept of data persistence so that they know which folders to use for data they want to preserve between sessions.

Profile Specification

Attribute	Default Value	Optional	Note
`name`	`""`	No	Name of the storage profile.
`startDirectory`	`"/"`	Yes	Start directory (user path of the MATLAB user) after the storage profile is mounted. See example and more below this table.
`restrictNavigationOutsideMounts`	`false`	Yes	If `true`, MATLAB users are not able to navigate (cd) to other directories other than those specified in mounts and MATLAB directory.
`mounts`	`<Nil>`	Yes	Directories to be mounted and then made available to the MATLAB user (see “Mount Specifications” on page 3-37).
`addons`	`<Nil>`	Yes	Directory where MATLAB add-ons are stored (see “Specification for Add-Ons” on page 3-37).

Attribute	Default Value	Optional	Note
`allowedGroups`	`<Nil>`	Yes	List of the groups that user should belong to so that they can use this profile These groups are compared against the list of groups that are configured through `authnz` (see “Placeholders” on page 3-48). • If `allowedGroups` is empty (default), the storage profile can be mounted by any user. • If there are multiple values in `allowedGroups`, a user that is part of any group listed can mount this storage profile. • The complete/exact group name that is returned in the SubjectAuthResponse should be included in `allowedGroups` as well. For example, in the LDAP scenario, the group name is the full Distinguished Name (DN): **(CN=group2,OU=Group,OU=Mail,DC=ad,DC=domain,DC=com)**

In YAML format:

```
storage:
  profiles:
    - name: <name>
      mounts:
        <list of mounts>
      startDirectory: <start directory>
      restrictNavigationOutsideMounts: <true/false>
      addons:
        directory: <addons directory>
      allowedGroups:
      - <name_of_group_1>
      - <name_of_group_2>
```

startDirectory—User session and Preferences directory. This folder contains the preferences, settings, history, and layout files associated with the user. For more about MATLAB user settings, see the MATLAB documentation in the Help Center.

MATLAB Online Server provides the ability to store the MATLAB preferences and some of the session information as part of the storage profile configuration.

For each storage profile, `startDirectory` can be customized. This field sets the session directory when the user signs into MATLAB Online Server.

When this directory is persistent, the history, layout preferences, and more will be automatically restored when the user next signs in.

Format:

```
startDirectory: "/home/${subject.subjectId}"
```

Specification for Add-Ons

Attribute	Default Value	Optional	Note
`directory`	`<startDirectory>/MATLAB Add-Ons`	Yes	Directory where MATLAB add-ons are stored.

Mount Specifications

Attribute	Default Value	Optional	Note
`name`	`""`	No	Name of the mount.
`mountPath`	`""`	No	Path of the mount. This info is available to the MATLAB user as the mounted path, after the user signs in.
`type`	`""`	No	Type of the mount. Based on the type, the mount properties can vary. For more information, see “Mount Types” on page 3-38.
`permissionType`	`""`	No	Type of the permissions that are given to the mounted directory after the user has signed in. Based on the permission type, the mount properties can vary. For more information, see “Mount Types” on page 3-38.

Attribute	Default Value	Optional	Note
`createIfNotExist` • This feature is applicable only when the dynamic placeholders are used . • This feature is applicable only when the permissionType is of “User.”	`false`	Yes	Dynamically create folders while provisioning the storage for MATLAB user, if the folder does not exist with the same UID permissions of the user.
`timeoutInSeconds`	5	Yes	Number of seconds that the server waits before terminating a running executable.

Mount Types

MATLAB Online Server supports the following mount types:

- `"empty"` — Create an empty directory that is deleted after the user signs out.
- `"host"` — Mount a directory from the node's host.
- `"nfs"` — Mount a directory from the network file system (NFS).
- `"delegated"` — Mount a directory by calling an executable.
- `"pvc"` — Mount a persistent volume claim (PVC) that is available on the Kubernetes cluster.

To disable certain mount types, in the `resource.yaml` YAML file, add or update the `storage` field and specify the `disallowedMountTypes` subfield. For example, this sample YAML disables the delegated and PVC mount types.

```
storage:
  disallowedMountTypes:
    - "delegated"
    - "pvc"
```

Empty

The Empty (**empty**) mount type creates an empty directory. This storage is ephemeral storage, meaning that after a user signs out of the session, all data inside the directory is deleted permanently.

This mount type does not require any extra properties.

In YAML format:

```
storage:
  profiles:
    - name: <name>
      mounts:
        - name: code
          mountPath: /code
          type: empty
          permissionType: <permission type>
      startDirectory: <start directory>
      restrictNavigationOutsideMounts: <true/false>
      addons:
        directory: <addons directory>
```

Host

The Host (**`host`**) mount type mounts a directory from the Nodes host. This mount type expects the directory (**`hostPath`**) to be present on the Node; otherwise, the MATLAB pod does not start.

This mount type has these extra properties:

- `hostPath` (required) — The folder being mounted
- `subPath` (optional) — The specific folder within **`hostPath`** that is available to the MATLAB user

In YAML format:

```
storage:
  profiles:
     - name: <name>
       mounts:
       - name: node_folder
         mountPath: /share
         hostPath: /share
         type: host
         subPath: "/exports/${subject.subjectId}"
         permissionType: <permission type>
       startDirectory: <start directory>
       restrictNavigationOutsideMounts: <true/false>
       addons:
         directory: <addons directory>
```

NFS

The NFS (**`nfs`**) mount type mounts an NFS directory and then makes it available to the MATLAB user.

This mount type expects that the NFS server allows mounting the NFS directories onto the MATLAB pods. The NFS administrator might need to allow access to the NFS server by the Kubernetes cluster, through the NFS exports list.

This mount type requires the following extra properties:

- `server` (required) — Name of NFS server
- `subPath` (required) — The folder within **`server`** being made available to the MATLAB user
- `mountOptions` (required) — Options for configuring the NFS mount. To view the supported NFS versions, see “Software Requirements” on page 1-9.

In YAML format:

```
storage:
  profiles:
    - name: <name>
      mounts:
        - name: nfsHome
          mountPath: /share
          type: nfs
          server: "nfs"
          subPath: "/exports/home/${subject.subjectId}"
          mountOptions:
"rw,relatime,vers=3,rsize=1048576,wsize=1048576,namlen=255,acregmin=60
0,acregmax=600,acdirmin=600,acdirmax=600,hard,nocto,noacl,proto=tcp,ti
meo=600,retrans=2,mountproto=tcp,local_lock=none,nolock"
          permissionType: <permission type>
      startDirectory: <start directory>
      restrictNavigationOutsideMounts: <true/false>
      addons:
        directory: <addons directory>
```

Delegated

The Delegated (`delegated`) mount type mounts a drive or directory using a specified executable. You can use this executable to customize the storage mount using options not natively supported by the other mount types. For example, your executable can mount an NFS drive and specify mount options using the Common Internet File System (CIFS) protocol, which the `"nfs"` mount type does not support.

By default, this mount type is not enabled. To enable it, add or update the `storage` field of your `resource.yaml` file to specify the `disallowedMountTypes` subfield, and do not include the `"delegated"` option in this subfield. For example, this sample YAML field enables all mount types.

```
storage:
  disallowedMountTypes: []
```

In addition to the default storage mount properties, this mount type has this additional property:

- `args` (required) — List of one or more command-line arguments used to call the executable. The first argument is the path to the executable file. Specify any additional options used to call the executable in subsequent arguments.

In YAML format:

```
storage:
  profiles:
    - name: <profile name>
      mounts:
        - name: <mount name>
          mountPath: <mount path>
          type: delegated
          permissionType: none
          args:
          -  "/opt/matlabonlineserver/my_mount_script.sh"
          timeoutInSeconds: 10
```

The executable that you write must support these options:

- `--mos-mount-dir <folder>` argument that specifies the folder being mounted
- `start` operation that mounts the folder
- `stop` operation that unmounts the folder

Your MATLAB Online Server installation includes a sample customizable shell script for mounting an NFS drive. The script is located in the resource proxy Docker image, which controls the storage layer for MATLAB users. It is located at this path:

`mos-resource-proxy-image/main/src/main/scripts/example_mount.sh`

The `example_mount.sh` script has these contents.

```
#!/bin/bash

usage()
{
    echo "usage   : example_mount [[-h]]"
    echo "example : ./example_mount.sh or ./example_mount.sh -h or"
    echo "./example_mount.sh start --mos-mount-dir /tmp or ./example_mount.sh stop --mos-mount-dir /tmp"
}

function run() {
    echo "$@"
    $@
}

MOS_MOUNT_DIR=""
MOUNT_OPERATION=""
NFS_SERVER="myserver"
NFS_SUBPATH="mysubpath"

while [ "$1" != "" ]; do
    case $1 in
        -h | --help )                       usage
                                            exit
                                            ;;
        --mos-mount-dir )                   shift
                                            MOS_MOUNT_DIR=$1
                                            ;;
        start )                             MOUNT_OPERATION=$1
                                            ;;
        stop )                              MOUNT_OPERATION=$1
                                            ;;
        # Other arguments can be processed the same way
        * )                                 usage
                                            exit 1
    esac
    shift
done

if [ -z "$MOS_MOUNT_DIR" ]
then
echo "Mount directory is empty"
exit 1
fi

case "${MOUNT_OPERATION}" in
    "start")
        echo "Starting mount operation on directory ${MOS_MOUNT_DIR}"
        run mkdir -p "$MOS_MOUNT_DIR"
        run sudo mount -t nfs -o rw,relatime,vers=3,rsize=1048576,wsize=1048576,namlen=255,acregmin=600, \
acregmax=600,acdirmin=600,acdirmax=600,hard,nocto,noacl,proto=tcp,timeo=600,retrans=2,mountproto=tcp, \
local_lock=none,nolock ${NFS_SERVER}:${NFS_SUBPATH} "${MOS_MOUNT_DIR}"
        ;;
    "stop")
        echo "Stopping mount operation on directory ${MOS_MOUNT_DIR}"
        run sudo umount "${MOS_MOUNT_DIR}"
        ;;
    *)
        echo "Not a supported mount operation"
        exit 1
        ;;
esac

exit 0
```

In this script, you can customize the NFS_SERVER and NFS_SUBPATH variables to set the NFS folder you want to mount. Placeholders for specifying user names are supported. For more details, see “Placeholders” on page 3-48. Sample variable values are:

```
NFS_SERVER = "nfs"
NFS_SUBPATH = "exports/home/${subject.subjectId}"
```

By default, the executables you write can run the `mount`, `unmount`, `chown`, and `chmod` commands without requiring a password. If your executable uses other commands that normally require a password, you can customize the resource proxy image to run these commands without a password.

Caution Disabling password prompts for commands that normally require `sudo` administration privileges can be a security risk.

Suppose you want your executable to run `mkdir` without having to enter a password on folders that are normally password protected. You can write a Dockerfile to disable the password prompt in the resource proxy image and then rebuild the image.

1. Navigate to the folder containing your script.
2. Create a file named `Dockerfile` (no extension) and open it for editing.

   ```
   touch Dockerfile && nano Dockerfile
   ```

3. Create a backup of the resource proxy image using this Docker command. Run this command as a single line. Update the image names with the name of your resource proxy image.

   ```
   sudo docker tag containers.mathworks.com/matlab-online-server/mos-resource-proxy-image:1.13.0-bullseye
     containers.mathworks.com/matlab-online-server/mos-resource-proxy-image:1.13.0-bullseye-backup
   ```

4. Copy the following lines of code into your Dockerfile and customize them for your executable.

   - FROM — Specify the name of the resource proxy image you are referencing.
   - COPY — Specify the name of your script and the location where you want to copy the script to. In your storage profile, the first item in the `"args"` field references this location.
   - RUN — Specify the commands for which you want to disable the password requirements. To disable passwords on additional commands, add more RUN lines. These lines update the `sudo` policy on your machine. For more details, see Sudoers Manual.
   - USER — Temporarily switch to the root user to perform the RUN operation. `mwuser` is the default user for the resource proxy.

   ```
   FROM containers.mathworks.com/matlab-online-server/mos-resource-proxy-image:1.13.0-bullseye-backup

   USER root

   COPY myscript.sh /opt/matlabonlineserver/myscript.sh

   RUN chmod +x /opt/matlabonlineserver/mount.sh
   RUN echo "mwuser  ALL=(ALL) NOPASSWD: /usr/bin/mkdir" >> /etc/sudoers

   USER mwuser
   ```

5. Build and tag the image. For example, this sample tag inserts the word `"delegated"` at the end to record the change.

   ```
   sudo docker build . --tag
   containers.mathworks.com/matlab-online-server/mos-resource-proxy-image:1.13.0-bullseye-delegated
   ```

6. If you are using a remote registry, build and push the customized image to your Docker registry using these `docker` commands.

   ```
   sudo docker push . containers.mathworks.com/matlab-online-server/mos-resource-proxy-image:1.13.0-bullseye-delegated
   ```

7. In the `matlab-pool.yaml` file, add or update the `"images"` field with the customized resource-proxy image. For example:

```
images:
  resourceProxy:
    image: mos-resource-proxy-image
    tag: 1.13.0-bullseye-delegated
```

8 Undeploy and redeploy the `matlab-pool.yaml` file to apply your changes.

```
./mosadm undeploy
./mosadm deploy
```

Persistent Volume Claim (PVC)

PVC (`pvc`) mount type mounts a Persistent Volume Claim that is available on the Kubernetes cluster and makes it available to the MATLAB user.

Refer to the following link for detailed information on Persistent Volumes: https://v1-22.docs.kubernetes.io/docs/concepts/storage/persistent-volumes

Note MATLAB Online Server expects that the Persistent Volume Claim is allowed for:

- Mounting onto the MATLAB pods (same PVC)
  - Some PV types might restrict the scaling based on the PV configuration
- Read and write files
- Retaining data after exiting the session (using persistentVolumeReclaimPolicy)

If Persistent Volume or Persistent Volume Claim is created in Kubernetes with a different configuration, you must reconfigure the object to satisfy the above conditions.

This mount type requires the following extra properties:

- claimName
  - The name of the Persistent Volume Claim
- subPath
  - The folder within the data volume (Persistent Volume Claim) that is available for the MATLAB user

These extra properties are shown in YAML format in the following example:

```
storage:
  profiles:
    - name: <name>
      mounts:
        - name: home
          mountPath: /share
          type: pvc
          claimName: <pvc name>
          subPath: "/exports/home/${subject.subjectId}"
          permissionType: <permission type>
      startDirectory: <start directory>
      restrictNavigationOutsideMounts: <true/false>
      addons:
        directory: <addons directory>
```

When using a Persistent Volume Claim, keep these points in mind:

- You can specify the same Persistent Volume Claim more than once in a storage profile. Within the `mounts` field, specify each claim as a separate mount.
- If you specify the same Persistent Volume Claim in more than one storage profile, each claim must have a different mount name.
- The same Persistent Volume Claims defined in the storage profile are mounted in all MATLAB pods.
- The storage resources request that is defined in the Persistent Volume or Persistent Volume Claim must be sufficient to meet the entire storage needs for all users.

Permission Types

The following permission types are supported: None, Fixed, and User.

None

The None (**none**) permission type performs a mount with default access to the MATLAB process only.

This permission type is ideal for empty mount types (ephemeral storage) or anonymous read-only directories that are mounted.

This permission type does not require any extra properties.

In YAML format:

```
storage:
  profiles:
    - name: <name>
      mounts:
        - name: code
          mountPath: /code
          type: empty
          permissionType: none
      startDirectory: <start directory>
      restrictNavigationOutsideMounts: <true/false>
      addons:
        directory: <addons directory>
```

Fixed

The Fixed (**fixed**) permission type performs a mount and then shares the directory with the same permissions that are specified (**uid**, **gid**) to the MATLAB process only.

This permission type is ideal for shared read-only directories.

This permission type requires the following extra properties:

- uid
- gid

In YAML format:

```
storage:
  profiles:
    - name: <name>
      mounts:
      - name: public
        mountPath: /public
        type: nfs
        server: "nfs"
        subPath: "/exports/nfs_share"
        mountOptions:
"ro,relatime,vers=3,rsize=1048576,wsize=1048576,namlen=255,acreg
min=600,acregmax=600,acdirmin=600,acdirmax=600,hard,nocto,noacl,
proto=tcp,timeo=600,retrans=2,sec=sys,mountproto=tcp,local_lock=
none,nolock"
        permissionType: fixed
        uid: 1001
        gid: 1000
      startDirectory: <start directory>
      restrictNavigationOutsideMounts: <true/false>
      addons:
        directory: <addons directory>
```

User

The User (**user**) permission type performs a mount and then shares the directory with the same permissions as that of the user (**uid**, **gid**) to the MATLAB process only if the placeholders are resolved. For more information, see “Placeholders” on page 3-48.

This permission type is ideal for mounting user-specific directories, such as an NFS home folder, which is readable and writable by the signed-in user only.

This permission type requires the following extra properties:

- uid
- gid

In YAML format:

```
storage:
  profiles:
  - name: <name>
    mounts:
      - name: public
        mountPath: /public
        type: nfs
        server: "nfs"
        subPath: "/exports/home/${subject.subjectId}"
        mountOptions:
"rw,relatime,vers=3,rsize=1048576,wsize=1048576,namlen=255,acreg
min=600,acregmax=600,acdirmin=600,acdirmax=600,hard,nocto,noacl,
proto=tcp,timeo=600,retrans=2,mountproto=tcp,local_lock=none,nol
ock"
        permissionType: user
        uid: "${subject.uid}"
        gid: "${subject.gid}"
    startDirectory: <start directory>
    restrictNavigationOutsideMounts: <true/false>
    addons:
      directory: <addons directory>
```

Disallow UIDs

MATLAB Online Server includes a security feature that disallows certain UIDs to create storage mounts.

You can add a configurable list of UIDs that are not allowed to set up the storage permissions. For example, if a user with uid 0 signs in, the permissions on the binded folder are given the root permissions (as uid = 0); this feature disallows this permission binding for uid 0.

The following shows the default list of disallowed Uids (in YAML format); you can override the values with your specific disallowed Uids:

```
storage:
  disallowedUids:
    - "0"
    - "1000600715"
    - "1000600903"
    - "1000600912"
```

Storage Profiles: Dynamic Folder Creation

Currently, MATLAB Online Server provides a way to mount the file system available to the MATLAB user with the dynamic placeholders support of storage profiles. This feature extends the ability to create the folders automatically in their storage mount with the appropriate user permissions.

With the createIfNotExist attribute, you can dynamically create folders while provisioning the storage for the user, if the folder does not exist with the same UID permissions of the user.

For example, assume the following storage profile is defined and is applicable for the user when they sign in:

```
storage:
  profiles:
    - name: <name>
      mounts:
        - name: public
          mountPath: /public
           type: nfs
           server: "nfs"
           subPath: "/exports/home/${subject.subjectId}"
           mountOptions:
"rw,relatime,vers=3,rsize=1048576,wsize=1048576,namlen=255,acregmin=600,acregm
ax=600,acdirmin=600,acdirmax=600,hard,nocto,noacl,proto=tcp,timeo=600,retrans=
2,mountproto=tcp,local_lock=none,nolock"
          permissionType: user
          uid: "${subject.uid}"
          gid: "${subject.uid}"
          createIfNotExist: true
      startDirectory: <start directory>
      restrictNavigationOutsideMounts: <true/false>
      addons:
        directory: <addons directory>
```

- The placeholder "${subject.subjectId}" is resolved to the user name, for example, username1, when a user signing in to MATLAB Online Server as username1. The NFS server and the subpath “nfs:/exports/home” are mounted by the storage service
- If the folder /exports/home/username1 exists in nfsserver “nfs,” then the storage service uses that folder. Otherwise, the folder is created after placeholder resolution with the permissions “${subject.uid}” (for example, 3904).
- The folder permissions on the Linux file system (in the NFS server) are:

  ```
  ls -al /exports/home
  drwxr-xr-x 1     3904  3904   20 Feb 18 19:13 /exports
  /home/username1/
  ```

The createIfNotExist attribute is applicable only when:

- Dynamic placeholders are used
- permissionType is “User”


```
ls -al /exports/home
drwxr-xr-x 1     3904  3904   20 Feb 18 19:13 /exports
/home/username1/
```


Placeholders

Placeholders are temporary markers for values that are resolved dynamically while the directories are mounted during storage setup.

For more information on the AuthNZ IdP (Identify Provider) configuration and the Subject response, see “Configure User Authentication in MATLAB Online Server” on page 3-6.

A sample response from AuthNZ follows. In this case, with the same information for the signed-in user from the AuthNZ service.

```
{
   "mwtype":"authnz/SubjectAuthResponse",
   "subject":{
      "subjectId":"username1",
      "displayName":"username1",
      "groups": [
      "group1",
      "group2"
    ],
      "extra":{
         "uid":[
            "3904"
         ],
         "gid":[
            "3904"
         ]
      },
      "idp":"ldap"
   },
   "authToken":"xxxxxxx-xxxxxxxx-xxxxxx",
   "authenticated":true
}
```

From the above response, the following placeholders can be resolved:

- subject.subjectId
- subject.displayName
- subject.groups
- subject.uid
- subject.gid

As another example, consider a storage unit with one NFS directory per user that must be mounted, where:

- Each user has a different home directory that is differentiated by the user name.
- The home directory of each user is locked to their UID and GID.

The storage profile can be constructed as follows:

```
storage:
  profiles:
    - name: nfsHome
      startDirectory: "/home/${subject.subjectId}"
      mounts:
        - name: home
          permissionType: user
          mountPath: "/home/${subject.subjectId}"
          uid: "${subject.uid}"
          gid: "${subject.gid}"
          type: nfs
          server: "nfs"
          subPath: "/exports/home/${subject.subjectId}"
          mountOptions:
"rw,relatime,vers=3,rsize=1048576,wsize=1048576,namlen=255,acreg
min=600,acregmax=600,acdirmin=600,acdirmax=600,hard,nocto,noacl,
proto=tcp,timeo=600,retrans=2,mountproto=tcp,local_lock=none,nol
ock"
      restrictNavigationOutsideMounts: false
```

Note that mountPath cannot be started with a placeholder.

Storage Profiles: Examples

The following examples demonstrate creating storage profiles based on the following types of storage systems (backed by Kubernetes Persistent Volumes):

- NFS
- AWS
  - EBS
  - EFS
- Azure
  - Azure Files

Results of these examples may vary based on variable information, such as Kubernetes version, features on the Kubernetes cluster, updates by the cloud provider that are more recent that these examples, and so on.

Each example includes the following:

- Prerequisites:
  - Persistent Volumes, Persistent Volume Claims needs the appropriate storage class (CSI Driver) to gets created successfully
  - Refer to the following link for detailed information on Persistent Volumes: https://v1-22.docs.kubernetes.io/docs/concepts/storage/persistent-volumes
- Create Persistent Volume Claim
  - Provides a sample yaml file that can be used to create the Persistent Volume Claim and other dependent objects, if required.
  - Provides a command to create these objects in Kubernetes.

- Storage Profile
  - A storage profile is a specification defined by the MATLAB Online Server administrator that determines how and which drives, directories, and files are available to the resource (MATLAB) after users sign on
  - The storage profiles are defined in the MATLAB Pool configuration:

    `<OVERRIDES_DIR>/<cluster>/<namespace>/matlab-pool.yaml`
  - After MATLAB Online Server is deployed successfully with the new storage profiles, when a MATLAB user signs in to MATLAB Online Server, the working directory of the user will be /home/<username>.
- Cleanup (optional)
  - Provides commands for removing the Persistent Volume Claim and the dependent Kubernetes objects, if needed

NFS - PVC Storage Profile

The following example provides instructions to create a Persistent Volume Claim where the data is stored in an NFS server. This newly created Persistent Volume Claim in MATLAB Online Server can provide data persistence to the MATLAB user upon signing in.

1. Prerequisites:

   The cluster must install the required NFS CSI driver. There are many Kubernetes CSI drivers that are available, and the instructions differ based on which is installed.

   This example uses the default NFS CSI driver created by the Kubernetes project:

   a. Add helm repo:

   ```
   helm repo add csi-driver-nfs https://raw.githubusercontent.com/kubernetes-csi/csi-driver-nfs/master/charts
   ```

   b. Install CSI driver:

   ```
   helm install csi-driver-nfs csi-driver-nfs/csi-driver-nfs --namespace kube-system
   ```

2. Create Persistent Volume Claim

   Create Persistent Volume Claim using YAML and kubetctl.

   - Save the following lines in a YAML file, for example, “nfs-pv.yaml.”

```
apiVersion: v1
    kind: PersistentVolume
    metadata:
      name: pv-example-nfs
    spec:
      capacity:
        storage: 10Gi
      accessModes:
        - ReadWriteMany
      persistentVolumeReclaimPolicy: Retain
      mountOptions:
        - relatime
        - vers=3
        - rsize=1048576
        - wsize=1048576
        - namlen=255
        - acregmin=600
        - acregmax=600
        - acdirmin=600
        - acdirmax=600
        - hard
        - nocto
        - noacl
        - proto=tcp
        - timeo=600
        - retrans=2
        - mountproto=tcp
        - local_lock=none
        - nolock
      csi:
        driver: nfs.csi.k8s.io
        readOnly: false
        volumeHandle: unique-volumeid  # make sure it's a unique id in the cluster
        volumeAttributes:
          server: examplenfsserver
          share: /local-ssd/nfs_pv
    ---
    kind: PersistentVolumeClaim
    apiVersion: v1
    metadata:
      name: pvc-example-nfs
    spec:
      accessModes:
        - ReadWriteMany
      resources:
        requests:
          storage: 10Gi
      volumeName: pv-example-nfs
      storageClassName: ""
```

- Create the Persistent Volume and Persistent Volume Claim using the following command:

  `kubectl apply --namespace` *`your-namespace`* `-f nfs-pv.yaml`

3 Storage Profile

In your *`overrides/cluster/namespace/`*`matlab-pool.yaml` file, add these lines to mount the volume in your `home` directory.

```
storage:
  profiles:
    - name: home
      mounts:
        - name: home
          mountPath: "/home/${subject.subjectId}"
          type: pvc
          claimName: "pvc-example-nfs"
          subPath: "/data/${subject.subjectId}"
          createIfNotExist: true
          permissionType: user
          uid: "${subject.uid}"
          gid: "${subject.uid}"
      startDirectory: "/home/${subject.subjectId}"
```

4 Cleanup

   a Undeploy all applications that are using Persistent Volume. For undeploying MATLAB Online Server, matlab-pool, use the following command:

   ```
   ./mosadm undeploy matlab-pool
   ```

   b Remove Persistent Volume and Persistent Volume Claim:

   ```
   kubectl delete -f
   ```

   c Uninstall the NFS CSI driver:

   ```
   helm uninstall csi-driver-nfs -n kube-system
   ```

EBS (Elastic Block Storage) PVC Storage Profile

The following example provides instructions to create a Persistent Volume Claim where the data is stored in an AWS EBS volume, and for using this newly created Persistent Volume Claim in MATLAB Online Server to provide data persistence to the MATLAB user upon signing in.

1 Prerequisites:

   The cluster must install the required EBS CSI Driver. To install the driver, refer to the information at Amazon EBS CSI driver.

   You might need to set permissions to allow MATLAB Online Server to create the EBS volumes and attach them to the Kubernetes cluster through IAM roles.

2 Create Persistent Volume Claim

   Create Persistent Volume Claim using yaml and kubetctl.

   a Save the following lines in a yaml file, for example, “ebs-pv.yaml.”

```
apiVersion: v1
   kind: PersistentVolumeClaim
   metadata:
     name: pvc-example-ebs-eks
   spec:
     accessModes:
       - ReadWriteOnce
     storageClassName: gp2
     resources:
       requests:
         storage: 100Gi
```

b Create Persistent Volume and Persistent Volume Claim using the command:

`kubectl apply --namespace` *`your-namespace`* `-f ebs-pv.yaml`

Note that you do not need to create a Persistent Volume explicitly; the GP2 storage class automatically provisions an EBS volume when a Persistent Volume Claim is created.

3 Storage Profile

In your *`overrides/cluster/namespace/`*`matlab-pool.yaml` file, add these lines to mount the volume in your `home` directory.

```
storage:
  profiles:
    - name: home
      mounts:
        - name: home
          mountPath: "/home/${subject.subjectId}"
          type: pvc
          claimName: "pvc-example-ebs-eks"
          subPath: "/data/${subject.subjectId}"
          createIfNotExist: true
          permissionType: user
          uid: "${subject.uid}"
          gid: "${subject.uid}"
      startDirectory: "/home/${subject.subjectId}"
```

4 Cleanup

For more information, see Amazon EBS CSI driver.

EFS (Elastic File Storage) PVC Storage Profile

The following example provides instructions to create a Persistent Volume Claim where the data is stored in an AWS EFS volume, and for using this newly created Persistent Volume Claim in MATLAB Online Server to provide data persistence to the MATLAB user upon signing in.

1 Prerequisites:

The cluster must install the required EFS CSI Driver. To install the driver, refer to the information at Amazon EFS CSI driver.

You might need to set permissions to allow MATLAB Online Server to create the EFS volumes and attach them to the Kubernetes cluster through IAM roles.

2 Persistent Volume Claim

Create Persistent Volume Claim using yaml and kubetctl.

a Save the following lines in a YAML file, for example, “efs-pv.yaml.”

```
kind: StorageClass
    apiVersion: storage.k8s.io/v1
    metadata:
      name: efs-sc
    provisioner: efs.csi.aws.com
    ---
    apiVersion: v1
    kind: PersistentVolume
    metadata:
      name: pvc-example-efs-eks
    spec:
      capacity:
        storage: 5Gi
      volumeMode: Filesystem
      accessModes:
        - ReadWriteMany
      persistentVolumeReclaimPolicy: Retain
      storageClassName: efs-sc
      csi:
        driver: efs.csi.aws.com
        volumeHandle: <EFS Volume ID >  # fs-8416d972
    ---
    apiVersion: v1
    kind: PersistentVolumeClaim
    metadata:
      name: pvc-example-efs-eks
    spec:
      accessModes:
        - ReadWriteMany
      storageClassName: efs-sc
      resources:
        requests:
          storage: 5Gi
```

b Create Persistent Volume and Persistent Volume Claim using the following command:

```
kubectl --namespace your-namespace apply -f az-files-pv.yaml
```

3 Storage Profile

In your *overrides/cluster/namespace*/matlab-pool.yaml file, add these lines to mount the volume in your home directory.

```
storage:
  profiles:
    - name: home
      mounts:
        - name: home
          mountPath: "/home/${subject.subjectId}"
          type: pvc
          claimName: "pvc-example-efs-eks"
          subPath: "/data/${subject.subjectId}"
          createIfNotExist: true
          permissionType: user
          uid: "${subject.uid}"
          gid: "${subject.uid}"
      startDirectory: "/home/${subject.subjectId}"
```

4 Cleanup

For more information, see Amazon EBS CSI driver.

Azure Files PVC Storage Profile

The following example provides instructions to create a Persistent Volume Claim where the data is stored in an Azure Files volume, and for using this newly created Persistent Volume Claim in MATLAB Online Server to provide data persistence to the MATLAB user upon signing in.

1 Prerequisites

The cluster must install the required Azure Files CSI Driver. To install the driver, see Use Azure Files Container Storage Interface (CSI) drivers in Azure Kubernetes Service (AKS).

You must allow permissions for MATLAB Online Server to access the Azure Files volumes and attach them to the Kubernetes cluster. Create a Kubernetes secret with the appropriate credential information (storage account and the storage key):

```
kubectl create secret generic --namespace your-namespace
az-files-example-secret --from-literal=azurestorageaccountname=<STORAGEACCT>
--from-literal=azurestorageaccountkey=<STORAGEKEY>
```

2 Persistent Volume Claim

Create Persistent Volume Claim using yaml and kubetctl.

To provide access to the Azure Files, perform the following steps:

a Save the following lines in a YAML file, for example, “az-files-pv.yaml".

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-example-az-files
spec:
  capacity:
    storage: 5Gi
  accessModes:
    - ReadWriteMany
  azureFile:
    secretName: az-files-example-secret
    shareName: <AZ-Files-Share-Name>
    readOnly: false
  mountOptions:
  - dir_mode=0777
  - file_mode=0777
  - uid=1000
  - gid=1000
  - mfsymlinks
  - nobrl
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-example-az-files
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: ""
  resources:
    requests:
      storage: 5Gi
```

b Create Persistent Volume and Persistent Volume Claim using the following command:

```
kubectl --namespace your-namespace apply -f az-files-pv.yaml
```

3 Storage Profile

In your *overrides/cluster/namespace/*`matlab-pool.yaml` file, add these lines to mount the volume in your `home` directory.

```
storage:
  profiles:
    - name: home
      mounts:
        - name: home
          mountPath: "/home/${subject.subjectId}"
          type: pvc
          claimName: "pvc-example-az-files"
          subPath: "/data/${subject.subjectId}"
          createIfNotExist: true
          permissionType: user
          uid: "${subject.uid}"
          gid: "${subject.uid}"
      startDirectory: "/home/${subject.subjectId}"
```

4 Cleanup

For more information, see Use Azure Files Container Storage Interface (CSI) drivers in Azure Kubernetes Service (AKS).

See Also

Related Examples

- “Set MATLAB Installation Source” on page 3-26
- “Configure MATLAB in MATLAB Online Server” on page 3-17
- “Configure User Authentication in MATLAB Online Server” on page 3-6

Configure MATLAB Session Settings

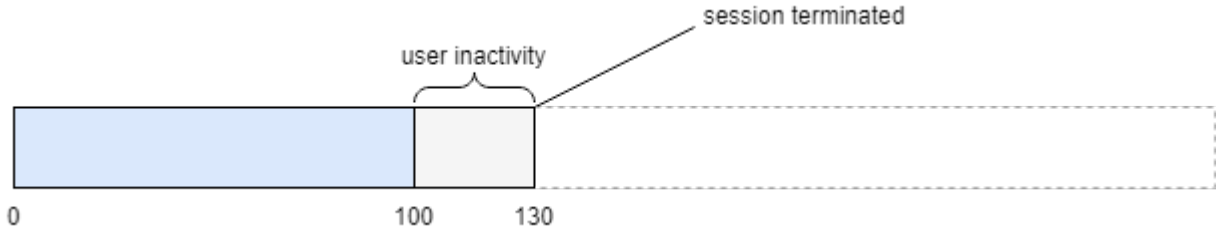
In the `resourceDefinition` section of the `matlab-pool.yaml` file, these fields configure how long before MATLAB sessions time out. The default field values are shown. Units are in minutes.

```
resourceDefinition:
  maxLifetimeMinutes: 240
  defaultLifetimeMinutes: 120
  maxInactivityTimeoutMinutes: 30
  defaultInactivityTimeoutMinutes: 15
```

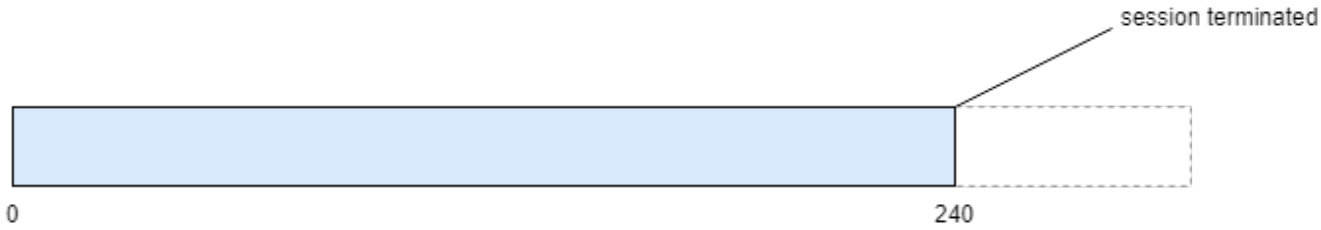
This table describes the type of session timeouts you can configure and which YAML fields to update.

Goal	Field to Update
Set the number of minutes that MATLAB Online Server waits before terminating the MATLAB session. MATLAB Online Server terminates the session regardless of whether the user is still active.	`maxLifetimeMinutes` **Note** For simplicity, consider setting `defaultLifetimeMinutes` to the same value as `maxLifetimeMinutes`.
Set the number of minutes that the MATLAB Online user interface waits for activity before terminating the MATLAB session. Session activity is defined as any mouse or keyboard movements or any API requests.	`maxInactivityTimeoutMinutes` **Note** For simplicity, consider setting `defaultInactivityTimeoutMinutes` to the same value as `maxInactivityTimeoutMinutes`.

This diagram shows a session that terminates due to inactivity. After 100 minutes, the user becomes inactivate. After 30 consecutive minutes of inactivity (`maxInactivityTimeoutMinutes` and `default InactivityTimeoutMinutes` are set to 30), the MATLAB Online interface terminates the session.



This diagram shows a session that terminates when it reaches the maximum session time. The user remains active for the entire session, but after 240 minutes (`maxLifetimeMinutes` = 240), the MATLAB Online interface terminates the session.



See Also

Related Examples

- “Configure MATLAB in MATLAB Online Server” on page 3-17

Configure NGINX Ingress Controller

The NGINX ingress controller processes incoming requests to the Kubernetes cluster and acts as a load balancer for MATLAB Online Server. Using the `mosadm install-ingress` command, you can install a controller that is preconfigured to work with MATLAB Online Server.

If your cluster already has an NGINX ingress controller installed, you must update your controller settings to work with MATLAB Online Server. Settings can vary by controller, but you can use these sample values to update the settings in your own controller.

```
use-http2: "true"
hsts: "false"
hsts-include-subdomains: "false"
proxy-connect-timeout: "365"
proxy-read-timeout: "365"
proxy-send-timeout: "365"
proxy-next-upstream: "error timeout http_502 http_503 http_504"
proxy-body-size: "100m"
```

Note Because MATLAB Online Server has host networking enabled, the number of NGINX ingress controllers must not exceed the number of nodes in the cluster.

See Also

`mosadm install-ingress`

Related Examples

- “Host MATLAB Online on Your Infrastructure” on page 1-3

Configure Network Policies

A network policy specifies how groups of pods are allowed to communicate with each other and with other network endpoints. You can modify the network policies for these MATLAB Online Server services running inside a Kubernetes cluster:

- AuthNZ (authentication)
- Core-UI (UI-related files)
- Gateway (proxies the requests to the appropriate resource)
- MATLAB Pool (remote MATLAB resources)
- Resource (acquires the MATLAB resource for the user)

This topic describes how to enable network policies and customize rules for providing communication with external systems, for example, LDAP, databases, and other internal services.

Prerequisites

Before you begin, make sure you have completed the following tasks:

- Install and then configure MATLAB Online Server. See Single Machine Installation on page 2-2.
- Familiarize yourself with Kubernetes network policies. See Network Policies at kubernetes.io.
- Create a Kubernetes cluster with a network policy plugin installed.
  - If you have a Kubernetes cluster that was created using mosadm bootstrap-node, Kube Router is preconfigured as a network policy plugin.
  - If you have a managed Kubernetes cluster (either from a cloud provider or a custom setup in-house Kubernetes cluster), you must install and then validate the network policy adapter before following the instructions in this section. (If you need to, refer to Install a Network Policy Provider at kubernetes.io. Make sure you that the instructions you follow align with the Kubernetes version you are using).
  - If you are already using the network policy adapter, then you can proceed without any further setup.

Network Policy Configuration Workflow

Securing network connections revolves around defining egress and ingress rules. For a Kubernetes pod:

- Ingress refers to incoming connections to the pod.
- Egress refers to outgoing connections from the pod.

In Kubernetes network policy, you create "allow" rules for ingress and egress independently (egress, ingress, or both).

This topic focuses on customizing egress rules. If you have use cases to customize the ingress rules, contact MathWorks Support.

To customize the egress rules for a service, follow this workflow.

1. Open the YAML override file for the service:

   `matlab_online_server/overrides/`*`cluster`*`/`*`namespace`*`/`*`service-name`*`.yaml`

   The YAML file you need to update depends on the service or deployment you are enabling communication with.

2. Open the default egress policy for the service:

   `matlab_online_server/charts/`*`service-name`*`-chart/values.yaml`

3. Copy the `networkPolicy` field from the `values.yaml` file into the `overrides/.../`*`service-name`*`.yaml` file.

   Any egress rules defined in *`service-name`*`.yaml` override the default rules defined in `values.yaml`. If you do not copy the `networkPolicy` field from `values.yaml` to the override file, the default egress rules for that service instance do not apply.

   Note For the MATLAB Pool service only, you can allow additional ports without having to override the default network policy. For more details, see “Allow Additional Ports” on page 3-63.

4. In the `egress` subfield of `networkPolicy`, customize the egress rules based on your network policy.

   - For more details on network policies you can update, see the examples in this topic.
   - For more advanced customizations of egress rules not covered in the examples, see Network Policies in the Kubernetes documentation.

5. Redeploy MATLAB Online Server by using the `mosadm` command.

   ```
   ./mosadm undeploy
   ```

   ```
   ./mosadm deploy
   ```

Enable and Disable Network Policies

Network policies are enabled by default for all services and deployments.

To enable or disable all network policies, make the following YAML configuration part of the override for that service:

```
networkPolicy:
 enabled: true
```

You can make the above configuration part of `all.yaml`.

Allow Additional Ports

To allow outgoing connections to additional ports, add the port number and optional protocol to the `ports` field. For example:

```
networkPolicy:
  enabled: true
  egress:
    - ports:
      - port: 1433
        protocol: TCP
      - port: 1433
        protocol: UDP
```

For the MATLAB pool only, you can allow connections to ports in addition to the ones listed in the default policy. In the `networkPolicy` field, uncomment the `additionalAllowedPorts` and specify the additional ports and optional protocols allowed. For example:

```
networkPolicy:
  enabled: true
  additionalAllowedPorts:
    - port: 1433
      protocol: TCP
    - port: 1433
      protocol: UDP
```

For the MATLAB pool, you can modify egress rules using either the `additionalAllowedPorts` field or the `egress` field, but you cannot include both.

Block IP Addresses

To block outgoing communication to certain IP addresses, add the block addresses to the except field. For example:

```
networkPolicy:
  enabled: true
    egress:
      to:
      - ipBlock:
          cidr: 0.0.0.0/0
          except:
            - 169.254.169.0/24
            - 192.168.0.0/24
```

Integrate with Databases

Database Toolbox™ provides functions and an app for exchanging data with relational and nonrelational databases. The toolbox enables this exchange by automatically converting between database and MATLAB data types.

Database Toolbox (on Linux) supports any JDBC-compliant relational database, as well as NoSQL databases including Cassandra®, MongoDB®, and Neo4j®.

For instructions on connecting to a database using MATLAB, see the Database Toolbox documentation in the Help Center.

Databases expose a port to communicate with JDBC-compliant clients, such as MATLAB. For clients to communicate with the database server, you must customize the firewall and network policy to open the port to communicate. Customize the network policy in the `matlab-pool.yaml` override file:

`matlab_online_server/overrides/`*`cluster`*`/`*`namespace`*`/matlab-pool.yaml`

For example, suppose you want to enable communication between MATLAB and a MySQL database server with these details:

- Server: mymsql.company.com
- JDBC port: 3306 (the standard one)

For MATLAB to communicate with the database server, the policy must allow access to the following ports:

- 53 – mwsql.company.com needs a DNS lookup
- 3306 – JDBC client (MATLAB) needs to communicate with the JDBC server (database)

The overall network policy configuration is specified as shown:

```
networkPolicy:
  enabled: true
  egress:
    - to:
      - ipBlock:
          cidr: 0.0.0.0/0
          except:
            - 169.254.169.0/24
      ports:
      # allow dns access
      - port: 53
        protocol: TCP
      - port: 53
        protocol: UDP
      # allow standard http/https port access
      - port: 80
        protocol: TCP
      - port: 443
        protocol: TCP
      # allow access to MathWorks License Manager server port
      - port: 27000
        protocol: TCP
      - port: 27001
        protocol: TCP
      # allow access on a variety of nfs related ports
      - port: 111
        protocol: UDP
      - port: 111
        protocol: TCP
      - port: 612
        protocol: UDP
      - port: 2049
        protocol: TCP
      - port: 2049
        protocol: UDP
      - port: 20048
        protocol: UDP
      - port: 20048
        protocol: TCP
      - port: 33668
        protocol: UDP
      - port: 33743
        protocol: TCP
      - port: 38494
        protocol: UDP
      - port: 36663
        protocol: TCP
      - port: 38494
        protocol: UDP
      - port: 40051
        protocol: TCP
      - port: 52241
        protocol: TCP
      - port: 3306
        protocol: TCP
```

For more details, see “Integrate MATLAB Online Server with Database Toolbox” on page 3-77.

Enable Communication with Internal LDAP Server

Lightweight Directory Access Protocol (LDAP) is an open, cross-platform protocol for accessing information from a server.

AunthNZ is one of the services in MATLAB Online Server that is responsible for communicating with the LDAP server to authenticate the user.

LDAP server details:

- Server: myldap.company.com
- LDAP port: 343
- LDAPS port: 643
- Override file: authnz.yaml

For AuthNZ to communicate to the LDAP server, the policy must allow access to the following ports:

- 53 – myldap.company.com needs a DNS lookup
- 343, 643 – AuthNZ needs to bind to the LDAP(s) server

The overall network policy configuration is specified as shown:

```
networkPolicy:
  egress:
    - ports:
      # allow dns access
      - port: 53
        protocol: TCP
      - port: 53
        protocol: UDP
      # allow standard http/https port access
      - port: 80
        protocol: TCP
      - port: 443
        protocol: TCP
      # allow standard ldap(s) port access
      - port: 343
        protocol: TCP
      - port: 643
        protocol: TCP
```

See “Configure User Authentication in MATLAB Online Server” on page 3-6 for more details.

Integrate with External Web Services

The MATLAB RESTful web services functions webread, websave, webwrite, and weboptions allow non-programmers to access many web services using HTTP GET and POST methods. However, some interactions with a web service are more complex and require functionality not supported by the RESTful web services functions. Use the HTTP interface classes for writing customized web access applications.

For more information on web access using MATLAB, see the MATLAB documentation in the Help Center.

These web services also expose a port (typically 80 and 443) to communicate with clients, such as MATLAB. For clients to communicate with these services, you must customize the firewall and network policy to open the port to communicate. Update the network policy in the `matlab-pool.yaml` override file:

`matlab_online_server/overrides/`*`cluster`*`/`*`namespace`*`/matlab-pool.yaml`

For more details on configuring MATLAB network communication, see “Configure MATLAB in MATLAB Online Server” on page 3-17.

View Default Network Policies

The preferred method of customizing network policies is to update the `overrides/.../`*`service-name`*`.yaml` files. However, when overriding egress rules, it is recommended that you view the existing network policies and copy over any policies that still apply.

Default AuthNZ Network Policies

The AuthNZ microservice has the following default egress rules defined in `matlab_online_server/charts/authnz-chart/values.yaml`:

```
networkPolicy:
  egress:
    - ports:
      # allow dns access
      - port: 53
        protocol: TCP
      - port: 53
        protocol: UDP
      # allow standard http/https port access
      - port: 80
        protocol: TCP
      - port: 443
        protocol: TCP
      # allow standard ldap(s) port access
      - port: 389
        protocol: TCP
      - port: 636
        protocol: TCP
      # allow redis master/replicas
      - port: 6379
        protocol: TCP
      # allow redis sentinel
      - port: 26379
        protocol: TCP
```

As specified by this policy override, the AuthNZ microservice can talk to the following ports:

- 53 – Used to make DNS lookups
- 80, 443 – Used to communicate with other HTTP(s) services (for example, http(s):// www.mathworks.com)
- 389, 636 – Used to communicate with the LDAP(s) server (for example, ldap:389)
- 6379, 26397 – Used to communicate with Redis™ services

Default MATLAB Pool Network Policies

The MATLAB Pool microservice has the following default egress rules defined in `matlab_online_server/charts/matlab-pool-chart/values.yaml`:

```
defaultNetworkPolicy:
  egress:
    - to:
      - ipBlock:
          cidr: 0.0.0.0/0
          except:
            - 169.254.169.0/24
      ports:
      # allow dns access
      - port: 53
        protocol: TCP
      - port: 53
        protocol: UDP
      # allow standard http/https port access
      - port: 80
        protocol: TCP
      - port: 443
        protocol: TCP
      # allow access to the flexlm server port
      - port: 27000
        protocol: TCP
      - port: 27001
        protocol: TCP
      # allow access on a variety of nfs related ports
      - port: 111
        protocol: UDP
      - port: 111
        protocol: TCP
      - port: 612
        protocol: UDP
      - port: 2049
        protocol: TCP
      - port: 2049
        protocol: UDP
      - port: 20048
        protocol: UDP
      - port: 20048
        protocol: TCP
      - port: 33668
        protocol: UDP
      - port: 33743
        protocol: TCP
      - port: 38494
        protocol: UDP
      - port: 36663
        protocol: TCP
      - port: 38494
        protocol: UDP
      - port: 40051
        protocol: TCP
      - port: 52241
        protocol: UDP
```

As specified in this policy override, MATLAB can talk to the following ports and IP addresses.

Ports:

- 53 – Used to make DNS lookups
- 80, 443 – Used to communicate with other HTTP(s) services (for example, http(s)://www.mathworks.com)
- 27000, 27001 – Used to communicate with the network license manager for checking out MATLAB licenses
  - These ports are controlled by license.dat.
  - See “Resolve License Service Issues” on page 5-12 for information on how to make sure these ports are fixed in the license file.
- 111, 612, 2049, 20048, 33668, 36663, 33743, 38494, 40051, 52241 – Used for communicating with the NFS server
  - These are standard ports, and their actual values can vary based on the internal NFS configuration.
  - These ports can be derived leveraging the Linux commands portmap or lsof on the NFS server.

IP addresses:

- Allows access to all the IP addresses except the following range: 169.254.169.0/24
  - In this example, the IP range 169.254.169.0/24 is blocked to prohibit access to the Amazon EC2® metadata service (if the node is running on AWS).

See Also

Related Examples

- “Configure User Authentication in MATLAB Online Server” on page 3-6
- “Configure MATLAB in MATLAB Online Server” on page 3-17
- “Integrate MATLAB Online Server with Database Toolbox” on page 3-77

Enable High Availability in MATLAB Online Server

MATLAB Online Server is deployed as a collection of microservices on a Kubernetes cluster. Because these services are stateless, maintaining state through cookies and communicating with each other, all their data is stored in memory and not backed up. If a pod restarts or fails, MATLAB users might need to sign in again. To prevent this behavior, you can enable the high availability (HA) for the server.

In an HA environment, MATLAB Online Server replicates each pod running an individual service. If one pod fails, the server switches to a replicated pod to keep the server running. This improved reliability comes at the expense of using additional computing resources.

Although you can enable HA for MATLAB Online Server regardless of the underlying Kubernetes infrastructure, your production environment should have a highly available Kubernetes cluster. For information on creating a highly available cluster, see the Kubernetes documentation for your provider.

How Services Work in High Availability Environment

MATLAB Online Server run these core services:

- `AuthNZ` – Authentication
- `Core-UI` – UI-related files
- `Gateway` – Proxies the requests to the appropriate resource
- `MATLAB Pool` – Remote MATLAB resources
- `Resource` – Acquires the MATLAB resource for the user

Although MATLAB Online Server services are stateless, by default, the `AuthNZ`and `Gateway` services store in-memory information about logged-in users and the mapping between user sessions and their respective MATLAB instances. When HA is enabled, for these services, MATLAB Online Server starts four pods (two `AuthNZ` pods, two `Gateway` pods) of a Redis service that works as a highly available cache. In this configuration, a failure of a single `AuthNZ` or `Gateway` pod does not affect the stored information about current sessions and their mapped tokens and MATLAB instances. When a Kubernetes cluster has multiple worker nodes, MATLAB Online Server pods are configured to be spread across several nodes using anti-affinity.

Note High availability does not affect the persistence of MATLAB code and data that users write, store, and execute on the server. To enable users to save data between session, you can mount persistent, individualized folders for them. For details, see “Configure File Storage for Users in MATLAB Online Server” on page 3-35.

Because the core services start as single pods, MATLAB Online Server is not highly available (HA) by default. When you enable HA, the server replicates these pods, either in a local Redis cache server or an external Redis server.

Enable High Availability Using Local Redis Cache (Default)

To enable the default HA configuration for MATLAB Online Server, where the server replicates pods in a local Redis cache server, follow these steps.

1. In the `install.config` file of your MATLAB Online Server installation folder, set the `IS_HA_ENABLED` property to `true`. With this setting, the generated overrides for each MATLAB Online Server service have a replica count set to 2.

   ```
   IS_HA_ENABLED=true
   ```

2. Copy the Helm charts to add the Redis pod to the updated MATLAB Online Server configuration.

   ```
   sudo ./mosadm copy-helm-charts
   ```

3. Generate the new YAML override files, skipping the MATLAB image.

   ```
   sudo ./mosadm generate-overrides --skip-matlab-image
   ```

4. Undeploy and redeploy the server.

   ```
   ./mosadm undeploy

   ./mosadm deploy
   ```

Enable High Availability Using External Redis Server

To use an external Redis server to enable HA, then you must first disable the local Redis deployment, and then manually configure the external deployment in the appropriate YAML overrides files.

1. In the `install.config` file, set `IS_HA_ENABLED` to `true`, and add the `DEPLOY_REDIS` property and set it to `false`.

   ```
   IS_HA_ENABLED=true
   DEPLOY_REDIS=false
   ```

2. Copy the Helm charts to add the Redis pod to the updated MATLAB Online Server configuration.

   ```
   sudo ./mosadm copy-helm-charts
   ```

3. Generate the new YAML override files, skipping the MATLAB image.

   ```
   sudo ./mosadm generate-overrides --skip-matlab-image
   ```

4. (Optional) If you are deploying MATLAB Online Server on Red Hat OpenShift, then you must also add the following lines to the generated `./overrides/`*`cluster`*`/`*`namespace`*`/redis.yaml` file:

   ```
   serviceAccount:
     create: false
     name: custom-sa
   ```

   The `custom-sa` field value is a service account that is created following the instructions for installing MATLAB Online Server on Red Hat OpenShift. Those instructions are:

   ```
   oc create sa custom-sa --namespace your-namepsace
   oc adm policy add-scc-to-user privileged -z custom-sa --namespace your-namespace
   ```

5. Open the `authnz.yaml` and `gateway.yaml` override files.

   - `./overrides/`*`cluster`*`/`*`namespace`*`/authnz.yaml`
   - `./overrides/`*`cluster`*`/`*`namespace`*`/gateway.yaml`

6. In these files, locate the `cache` section. The default `cache` section enables MATLAB Online Server to generate a Redis configuration with the type `FailoverRedis` in those two files, as shown here:

```
cache:
  type: FailoverRedis
  failoverredis:
    mastername: redis-sentinel-master
    addrs: mathworks-redis:26379
```

This cache type, `FailoverRedis`, enables communication with the Redis server deployed within MATLAB Online Server, using the configuration specified under `failoverredis`.

7 Update the cache type and configuration for your external Redis server. Specify one of these configurations.

- `BasicRedis` — Single Redis instance, where all items are stored in the same instance. Sample YAML:

  ```
  cache:
    type: BasicRedis
    basicredis:
      addr: your-redis-server:your-redis-port
      tls: true
      password: your-redis-password
  ```

- `ClusterRedis` — Multiple Redis instances, where items are sharded across all instances and each instance has only a subset of items. Sample YAML:

  ```
  cache:
    type: ClusterRedis
    clusterredis:
      addrs: your-redis-server:your-redis-port
      tls: true
      password: your-redis-password
  ```

  The `addrs` field, under the `ClusterRedis` type, can be assigned to a single `server:port` value or to multiple comma-separated `servers-i:port-i` values.

Note Not all Redis configurations provided by cloud platforms support HA by default. If you are using an external Redis server, use one with an HA configuration enabled. For secure communication with any external Redis server, you must set the `tls` field to `true`.

8 To apply your changes, undeploy and redeploy the server.

```
./mosadm undeploy

./mosadm deploy
```

See Also

Related Examples

- “Configure File Storage for Users in MATLAB Online Server” on page 3-35

External Websites

- Redis

Integrate MATLAB Online Server with Parallel Computing Toolbox and MATLAB Parallel Server

Using “MATLAB Parallel Server” and “Parallel Computing Toolbox”, your users of MATLAB Online can enable parallel computing to speed up their workflows. As an administrator, to enable use of these products, you must update the MATLAB Online Server configuration.

By default, the Helm charts that define the MATLAB Online Server cluster enable support for Parallel Computing Toolbox™ and the automatic creation of local `parpool` clusters. You can change this behavior by editing the `matlab-pool.yaml` override file.

To enable access to MATLAB Parallel Server™ clusters for MATLAB Online Server users, you must add a `NetworkPolicy` egress rule to the `matlab-pool.yaml` file in the override folder.

Configure Support in MATLAB Pool

To enable a version of MATLAB to have access to parallel computing resources, in the `matlab-pool.yaml` file of your MATLAB Online Server install overrides directory, specify these fields:

```
pct:
  enabled: true
  defaultPoolSize: 2
  defaultPoolAutoCreate: true

matlabProcessConfiguration:
  enableLocalParCluster: true
```

These values configure the installation so that:

- Parallel Computing Toolbox is available to all MATLAB Online Server users.
- The default pool size of the Parallel Computing Toolbox pools is 2.
- If the user runs MATLAB code containing `'parfor'` or `'spmd'`, the server automatically creates a default parallel pool using `parpool`.
- The default `parcluster`, accessed through `parpool`, is set to `'local'`.

Additionally, this configuration enables the MATLAB process to use local MATLAB workers on the same Kubernetes node that the MATLAB Docker containers are running on.

To configure MATLAB instances to use an on-premise cluster but not enable the use of local MATLAB workers on the Kubernetes nodes running MATLAB, set these values instead.

```
pct:
  enabled: true

matlabProcessConfiguration:
  enableLocalParCluster: false
```

Enable Use of On-Premise MJS or HPC Clusters

MATLAB Online Server is installed with a set of network policies that restrict the egress of network connections from nodes that the MATLAB Docker containers are running on. To enable the use of MJS

or access to other HPC job schedulers, you must update the `matlab-pool.yaml` overrides file. For more information, see “How Parallel Computing Software Runs a Job” (Parallel Computing Toolbox).

This YAML example opens up egress for network connections to an on-premise host running the MathWorks MJS job scheduler with the IP address of `172.28.152.244`. Other egress rules from the `matlab-pool/values.yaml` file are copied in.

```
networkPolicy:
  enabled: true
  egress:
    # This first rule enables egress to all ports on the 172.28.152.244 host
    - to:
      - ipBlock:
      cidr: 172.28.152.244/32
    # This next rule is the rule for the other MATLAB Online Server internal nodes
    - to:
      - ipBlock:
      cidr: 0.0.0.0/0
      except:
        - 169.254.169.0/24
      ports:
        # allow dns access
        - port: 53
          protocol: TCP
        - port: 53
          protocol: UDP
        # allow standard http/https port access
        - port: 80
          protocol: TCP
        - port: 443
          protocol: TCP
        # allow access to MathWorks License Manager server port (ports controlled by license.dat)
        - port: 27000
          protocol: TCP
        - port: 27001
          protocol: TCP
        # allow access on a variety of nfs related ports (derived from portmap/lsof in nfs server container)
        - port: 111
          protocol: UDP
        - port: 111
          protocol: TCP
        - port: 612
          protocol: UDP
        - port: 2049
          protocol: TCP
        - port: 2049
          protocol: UDP
        - port: 20048
          protocol: UDP
        - port: 20048
          protocol: TCP
        - port: 33668
          protocol: UDP
        - port: 33743
          protocol: TCP
        - port: 38494
          protocol: UDP
        - port: 36663
          protocol: TCP
        - port: 38494
          protocol: UDP
        - port: 40051
          protocol: TCP
        - port: 52241
          protocol: UDP
```

MATLAB End User Support

After you integrate MATLAB with Database Toolbox, provide end users with information on using parallel computing in the MATLAB Online Server environment. See “Use Parallel Computing in MATLAB Online Hosted by Your Organization”.

See Also

Related Examples

- “Configure Network Policies” on page 3-62

Integrate MATLAB Online Server with Database Toolbox

Using “Database Toolbox”, MATLAB Online end users can connect to a JDBC-compliant databases to execute queries. As an administrator, to allow proper communication to the database, you must update the MATLAB Online Server configuration.

Prerequisites

1. You installed the Database Toolbox product. For instructions, see “Install Add-Ons for MATLAB in MATLAB Online Server” on page 2-38.
2. You have database host and port information so that you can properly configure the back end. Many database vendors have a default port, but it is possible that this port is different in your environment.

Basic Configuration

By default, MATLAB Online Server allows the MATLAB process access to the minimum number of network ports required for it to run properly. To enable interactions with an external database, update the configuration YAML files to allow egress from the Kubernetes cluster to the desired resource.

1. Update the overrides file `matlab-pool.yaml` file with the required port information for this database. Make your changes in the `networkPolicy` section of the YAML file. See “Configure Network Policies” on page 3-62. For example:

   ```
   networkPolicy:
     enabled: true
     additionalAllowedPorts:
       # allow MySQL access on standard port
       - port: 3306
         protocol: TCP
       - port: 3306
         protocol: UDP
       … <any additional port configurations>
   ```

2. Deploy the change to the cluster using the following command. Replace *`your-namespace`* with your MATLAB Online Server namespace.

   ```
   ./mosadm upgrade matlab-pool --namespace your-namespace
   ```

3. Validate the change by confirming that your changes appear in the Kubernetes network policy. To view this network policy, run this command, replacing *`your-namespace`* with your MATLAB Online Server namespace.

   ```
   kubectl describe networkPolicy --namespace your-namespace
   ```

Custom Configuration

To specify a custom configuration, provide a storage mount for all MATLAB Online users with read-only drivers. You can make version-specific drivers for your environment readily available to your end users. To create a read-only mount that contains JDBC drivers for your MATLAB Online Server installation, see NFS configuration on page 3-39.

MATLAB End User Support

After you integrate MATLAB with Database Toolbox, provide end users with information on connecting to their database. See “Perform Database Queries in MATLAB Online Hosted by Your Organization”.

See Also

Related Examples

- “Configure Network Policies” on page 3-62
- “Install Add-Ons for MATLAB in MATLAB Online Server” on page 2-38

Server Management

Monitor MATLAB Online Server Using Kubernetes Dashboard

The Kubernetes Dashboard (also known as Web UI) provides a UI that you can use to monitor your cluster. These instructions show how to install and configure the dashboard for a MATLAB Online Server single-node configuration. You can use these instructions as the basis for a larger deployment and set of security configurations in your environment.

Prerequisites

- You have MATLAB Online Server installed on a single node, as described in “Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2.
- The Kubernetes `kubectl` command is configured, as described in “Install Kubernetes” on page 2-4 of the installation instructions.

Install and Set Up Dashboard on Single-Node Configuration

These instructions walk you through the basic installation and setup of the Kubernetes Dashboard. For detailed instructions and security configuration options, in the Kubernetes documentation, see https://kubernetes.io/docs/tasks/access-application-cluster/web-ui-dashboard.

1. Deploy the Kubernetes Dashboard. On the https://github.com/kubernetes/dashboard page, copy the installation command for the latest version of the dashboard and execute it in your MATLAB Online Server terminal. The command has this format.

   ```
   kubectl apply -f ...
   ```

2. Verify that the installation succeeds by checking the list of pods in your namespaces.

   ```
   kubectl get pods --all-namespaces
   ```

   Verify that the list displays two pods in a new `kubernetes-dashboard` namespace. For example:

   ```
   NAMESPACE              NAME                                         READY   STATUS    RESTARTS        AGE
   ...
   kubernetes-dashboard   dashboard-metrics-scraper-7c857855d9-7zj45   1/1     Running   0               10m
   kubernetes-dashboard   kubernetes-dashboard-6b79449649-lpz7t        1/1     Running   0               10m
   ...
   ```

3. Create a Kubernetes service account in your cluster. This account enables authentication into the dashboard.

   To create this account, in your MATLAB Online Server root folder, create a file named `matlab-online-server-admin-account.yaml` and enter this information, exactly as shown.

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: matlab-online-server-admin
  namespace: kube-system
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: matlab-online-server-admin
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
  - kind: ServiceAccount
    name: matlab-online-server-admin
    namespace: kube-system
```

Apply the file to your Kubernetes cluster.

```
kubectl apply -f matlab-online-server-admin-account.yaml
```

4. Retrieve the bearer token for the created `matlab-online-server-admin` account. Execute this command, and copy or save the returned token. In a later step, you use this token to access the dashboard locally.

```
kubectl -n kube-system describe secret $(kubectl -n kube-system get secret |
grep matlab-online-server-admin | awk '{print $1}') | grep token: | awk '{print $2}'
```

5. Start a proxy to enable access to the Kubernetes Dashboard.

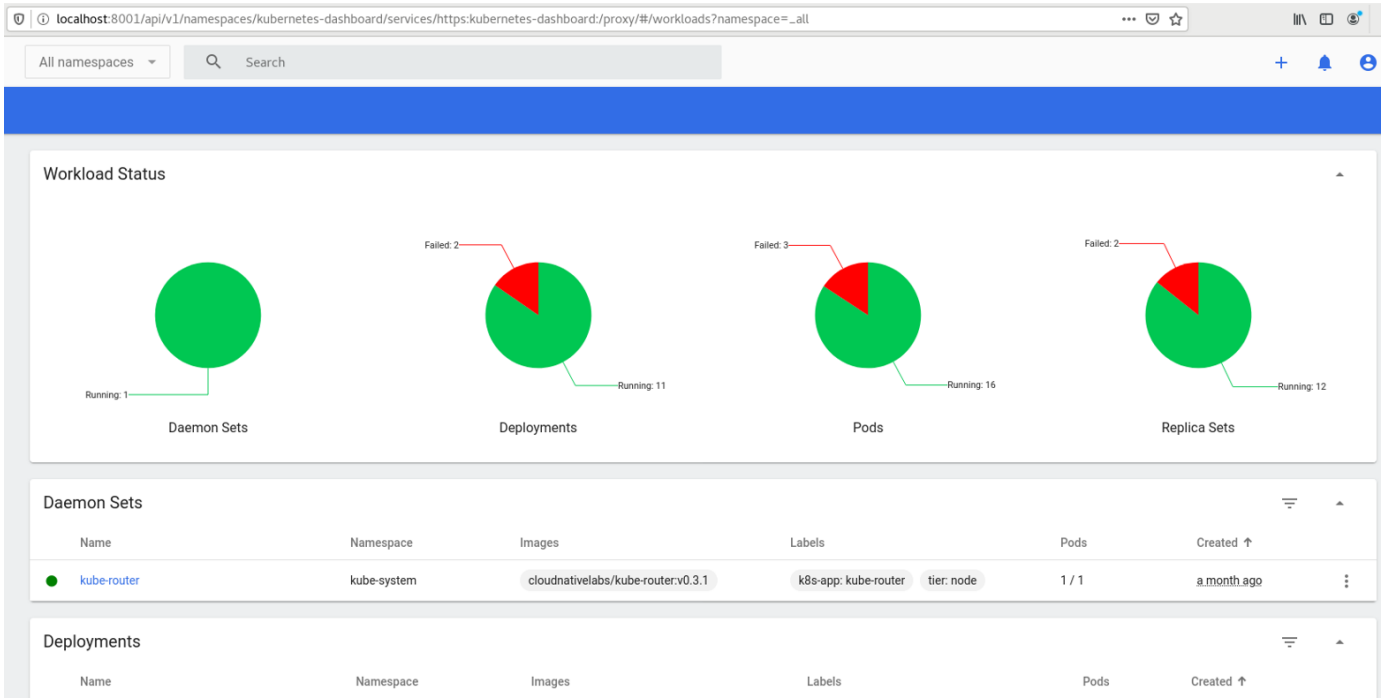
```
kubectl proxy
```

6. Access the Kubernetes Dashboard on a local browser at this URL.

```
https://localhost:8001/api/v1/namespaces/kubernetes-dashboard/services/https:kubernetes-dashboard:/proxy/#/login
```

7. When prompted for the authentication method, select **Token** and enter the bearer token you retrieved in a previous step.

8. Log in to the dashboard. You can now familiarize yourself with the different namespaces for your environment and the administrative features in the Kubernetes Dashboard.



Install and Set Up Dashboard on Managed Kubernetes Provider

The instructions for setting up the Kubernetes Dashboard depends on your provider.

- **Amazon Web Services (AWS)** — Set up Kubernetes Dashboard using the Elastic Kubernetes Service managed by AWS. In the Amazon documentation, see //docs.aws.amazon.com/eks/latest/userguide/dashboard-tutorial.html.
- **Microsoft® Azure** — Set up Kubernetes Dashboard using the Azure Kubernetes Service managed by Azure. In the Microsoft documentation, see //docs.microsoft.com/azure/aks/kubernetes-dashboard.
- **Google Kubernetes Engine** — The open-source Kubernetes Dashboard on the Google Kubernetes Engine managed by Google is deprecated. Google recommends using the Google Cloud console instead.

See Also

Related Examples

- “Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2
- “Install MATLAB Online Server on Cloud-Managed Kubernetes” on page 2-16

Check Pod Status

Check the status of the Kubernetes cluster pods. Before you can check the pods, you must have `kubectl` installed and configured to access the MATLAB Online Server Kubernetes cluster and have `helm` installed on the same nodes as `kubectl`.

Get Pod List

MATLAB Online Server microservices run on Kubernetes cluster pods. To display a list of these pods, run the `kubectl` command with `get pods`.

Syntax

Issue the following command, replacing the variable *`your-namespace`* with the namespace you used to deploy MATLAB Online Server:

```
kubectl --namespace your-namespace get pods
```

Example Output

This command results in a display of running pods and statuses:

```
NAME                                              READY     STATUS    RESTARTS      AGE
your-namespace-authnz-authnz-5bd9867d69-2trk2      1/1      Running      0      17h
your-namespace-core-ui-core-ui-6679d5ff7f-rdtfb    1/1      Running      0      17h
your-namespace-gateway-65697586f7-vsvls            1/1      Running      0      17h
your-namespace-license-84d4877898-mmqll            1/1      Running      0      17h
your-namespace-matlab-pool-5858d796b9-lqslb        2/2      Running      0      14m
your-namespace-matlab-pool-apparmor-loader-6prmt   1/1      Running      0      17h
your-namespace-matlab-pool-apparmor-loader-99zcq   1/1      Running      0      17h
your-namespace-matlab-pool-helpsearch-5689dd4cdf-t77rh  1/1 Running      0      17h
your-namespace-matlab-pool-ui-689fb5d9ff-tsrh4     1/1      Running      0      17h
your-namespace-resource-resource-56f5745747-dctps  1/1      Running      0      17h
```

The information returned can be broken down as follows:

- The following services are MATLAB Online Server core services:
  - authnz
  - core-ui
  - gateway
  - license
  - helpsearch
  - ui
  - resource

All these services are expected to have the value 1/1 under the READY column.

- The MATLAB Pool pod hosts the MATLAB process and it is expected to have the value 2/2 under the READY column. MATLAB Online Server assigns a separate pod for each MATLAB Online Server user.
- matlab-pool-apparmor-loader is a specific pod that MATLAB Online Server deploys for each Kubernetes node. The display shown in the previous example lists two pods of this type because its Kubernetes cluster has two nodes.

The name of each pod has suffix at the end as a unique identifier. As an example, the authnz pod name ends with the suffix 5bd9867d69-2trk2. If a new pod is created, then you get a new suffix identifier at the end of its name.

Get List of Other Required Services

While the command `kubectl` lists the main MATLAB Online Server services, running that command does not make visible other required services, such as namespace. Instead, check the deployment status of other required services by running the `helm list` command.

Syntax

In this example, replace *`your-namespace`* with the namespace you used for MATLAB Online Server:

```
helm list | egrep "your-namespace|nginx"
```

Example Output

This command results in a display of both core and additional services and their status.

```
your-namespace-authnz      1    Thu Jan 16 16:51:39 YYYY    DEPLOYED    mos-authnz-1.0.0              1.0.0          your-namespace
your-namespace-core-ui     1    Thu Jan 16 16:51:40 YYYY    DEPLOYED    mos-core-ui-1.0.0             1.0.0          your-namespace
your-namespace-gateway     1    Thu Jan 16 16:51:41 YYYY    DEPLOYED    mos-gateway-1.0.0             1.0.0          your-namespace
your-namespace-license     1    Thu Jan 16 16:51:42 YYYY    DEPLOYED    mos-license-1.0.0             1.0.0          your-namespace
your-namespace-matlab-pool 1    Thu Jan 16 16:51:43 YYYY    DEPLOYED    mos-matlab-pool-1.0.0         1.0.0          your-namespace
your-namespace-namespace   1    Thu Jan 16 16:51:45 YYYY    DEPLOYED    mos-namespace-1.0.0           1.0.0          your-namespace
your-namespace-resource    1    Thu Jan 16 16:51:46 YYYY    DEPLOYED    mos-resource-1.0.0            1.0.0          your-namespace
nginx-ingress              3    Sat Nov 23 15:21:31 YYYY    DEPLOYED    nginx-ingress-1.25.0             0.26.1c        nginx-ingress
```

Make sure each service has the value DEPLOYED.

Get Pod Descriptions

To get more details about each pod, run the following command:

```
kubectl describe pod podId
```

Syntax

For example, issue the following command, replacing *`your-namespace`* with the namespace you used for MATLAB Online Server:

```
kubectl --namespace=your-namespace describe pod your-namespace-matlab-pool-5858d796b9-lqslb
```

This command provides details about the MATLAB Pool pod identified with *`your-namespace`*`-matlab-pool-5858d796b9-lqslb`.

Example Output

Pod Descriptors	
Name	*`your-namespace`*-matlab-pool-5858d796b9-lqslb
Namespace	*`your-namespace`*
Node	private-node-3/123.45.67.89
Start Time	Fri, 17 Jan YYYY 10:25:20 -0500

Pod Descriptors	
Status	Running
IP	987.65.43.21
Containers	matlab: Container ID: docker:// cf5d62aeb248a87ddac5fdd775cae7c94e8df06816ad68c5d1db78b87f903cb4
Port	8080/TCP

Pod Descriptors	
Events	Type Reason Age From Message ---- ------ ---------- ------ ---- --- Normal Scheduled <unknown> default-scheduler Successfully assigned <your-namespace>/<your-namespace>-matlab- pool-5858d796b9-lqslb to private-node-3 Normal Pulling 19m kubelet, private-node-3 Pulling image "containers.mathworks.com/matlab-online-server/ mos-matlab-omnibus-image:1.13.0-bionic" Normal Pulled 19m kubelet, private-node-3 Successfully pulled image "containers.mathworks.com/matlab-online-server/ mos-matlab-omnibus-image:1.13.0-bionic" Normal Created 19m kubelet, private-node-3 Created container matlab Normal Pulling 19m kubelet, private-node-3 Pulling image "containers.mathworks.com/matlab-online-server/ mos-matlab-omnibus-image:1.13.0-bionic" Normal Pulling 19m kubelet, private-node-3 Pulling image "containers.mathworks.com/matlab-online-server/ mos-matlab-omnibus-image:1.13.0-bionic" Normal Pulled 19m kubelet, private-node-3 Successfully pulled image "containers.mathworks.com/matlab-online-server/ mos-matlab-omnibus-image:1.13.0-bionic" Normal Created 19m kubelet, private-node-3 Created container resource-proxy Normal Started 19m kubelet, private-node-3 Started container resource-proxy Normal Started 19m kubelet, private-node-3 Started container matlab Normal Pulled 19m kubelet, private-node-3 Successfully pulled image "containers.mathworks.com/matlab-online-server/ mos-matlab-omnibus-image:1.13.0-bionic" Normal Created 19m kubelet, private-node-3 Created container display Normal Started 19m kubelet, private-node-3 Started container display Normal Pulling 19m kubelet, private-node-3 Pulling image "containers.mathworks.com/matlab-online-server/ mos-matlab-omnibus-image:1.13.0-bionic" Normal Pulled 19m kubelet, private-node-3 Successfully pulled image "containers.mathworks.com/matlab-online-server/ mos-matlab-omnibus-image:1.13.0-bionic" Normal Created 19m kubelet, private-node-3 Created container windowmanager Normal Started 19m kubelet, private-node-3 Started container windowmanager

See Also

Related Examples

- “Resolve MATLAB Pod Issues” on page 5-8
- “Resolve Evicted or Terminated Pod Issues” on page 5-20

- “Resolve Pod Creation Stuck Issues” on page 5-16
- “Resolve Out-of-Resource Conditions” on page 5-22

External Websites

- Kubernetes

Update MATLAB Online Server

An update is a software change to the currently released version. MATLAB Online Server updates are released on a semi-regular basis to add functionality or provide bug fixes.

You must have MATLAB Online Server installed before you can install an update. See Single Machine Installation on page 2-2.

The method you use to update MATLAB Online Server creates two independent deployments (A and B) and then switches MATLAB Online traffic to the new deployment, resulting in little to no downtime for your end users.

To install and deploy a MATLAB Online Server updates, you:

1. Create a new server instance to deploy.
2. Switch from the old instance, referred to in this topic as deployment A, to the new instance, deployment B.

Configure New Instance

1. Choose a deployment environment.
   - If both the A and B deployments are on the same Kubernetes cluster, use a different Kubernetes namespace (NAMESPACE) for the new deployment. For example,
     - Existing deployment (A): **mathworksa**
     - New deployment (B): **mathworksb**
   - If you are bootstrapping a single-node cluster (mosadm bootstrap-node), use a new server.
   - If you are using a new Kubernetes cluster, configure the Kubernetes context on the terminal (KUBECONFIG config).
2. Choose a different domain name (DOMAIN_BASE) for the new deployment. For example:
   - Existing deployment (A): **matlab.domain.com**
   - New deployment (B): **matlab2.domain.com**

Download, Install, and Deploy MATLAB Online Server Update

1. Go to //www.mathworks.com/products/matlab-online-server.html.
2. Download the MATLAB Online Server update file.
3. Extract the MATLAB Online Server update to a new folder.
4. Edit install.config for deployment B, using the same information you used for deployment A, except:
   - For DOMAIN_BASE, use the new domain (for example, matlab2.domain.com).
   - If you are using the same Kubernetes cluster, use the new NAMESPACE (for example, mathworksb).
5. Deploy MATLAB Online Server on the new server or new namespace.

   You must keep a couple of things in mind:

- The number of namespaces cannot exceed the number of clusters.
- Make sure you use the correct mosadm for the deployment you are working with. For example:
  - Deployment A: Use mosadm from the MATLAB Online Server A folder.
  - Deployment B: Use mosadm from the MATLAB Online Server B folder.

Deploy MATLAB Online Server:

a Execute the following command to install all MATLAB Online Server components:

```
./mosadm deploy
```

This command uses Helm to install all the charts, customized with any overrides that you generated in the previous step.

b Check that the services are all running with kubectl, replacing *`your-namespace`* with the namespace you used for MATLAB Online Server:

```
kubectl get pods --namespace your-namespace
```

The pods running MATLAB can take several minutes to fully start (2/2 containers). It is important to wait until the pods are in the ready state. If any of the containers do not start running, check the license server and the `install.config` settings. See “Resolve MATLAB Pod Issues” on page 5-8.

6 Verify that you have successfully installed the new MATLAB Online Server instance by navigating to the new DOMAIN_BASE (for example, matlab2.domain.com).

Switch to New Deployment

1 In the B deployment, in the <OVERRIDES_DIR>/<cluster>/<namespace>/all.yaml file, update the domainBase value (matlab2.domain.com) to the original DOMAIN_BASE from deployment A (matlab.domain.com).

2 Make deployment B ready for DNS cutover by issuing the following command:

```
mosadm upgrade --namespace mathworksb
```

3 Verify that you have successfully updated all pods; see “Check Pod Status” on page 4-5.

4 To test with a restricted set of users or on a single machine before switching the DNS to the new deployment, perform a local verification. See the note after these instructions.

5 Perform the DNS cutover to redirect the traffic to the new deployment (B) from the old deployment (A) using your vendor’s instructions to update DNS records. Depending on your DNS provider, switching the DNS might require taking the MATLAB Online server offline for some time.

After the DNS switch, all MATLAB users must log in again.

If you encounter any failures, see the troubleshooting topics under “Server Management” for further investigation. You can also start this procedure over.

Note Local Verification

Optionally, before switching the DNS to the new deployment, you can test with a restricted set of users, or on a single machine. For example, update your operating system's hosts file to point your original domain base with the new server's IP address. To do so, update the hosts file:

- Linux: `/etc/hosts`
- Windows®: `<Windows Install>\System32\drivers\etc\hosts`
- Mac OS: `/etc/hosts`

Add the following line:

```
<new ip address> <old DOMAIN_BASE>
```

For example:

```
52.201.253.135 matlab.domain.com
```

This update means that when DOMAIN_BASE is accessed from this machine, the requests go to deployment B. You can now verify the MATLAB Online Server installation for deployment B, using the verification process in the "Verify MATLAB Online Server Installation" on page 2-10 section of the installation instructions.

See Also

Related Examples

- "Update MATLAB Release" on page 4-13

Update MATLAB Release

MathWorks might release MATLAB updates following the general release. An update is an incremental update to the current release. An upgrade is a change from one major release to the next.

Use this procedure to apply MATLAB release updates to a MATLAB Online Server instance.

1. On a Linux machine, download and then install the latest MATLAB update for the release of your MATLAB Online Server installation.
2. Create a backup of the working MATLAB image. This action might include pushing it to another repository or pushing it to the same repository but with a different tag.

   Note If the MATLAB update is not successful, you need the backup image to restore MATLAB in MATLAB Online Server. Do not skip this step.

   - You can use the following commands to save a backup of the existing MATLAB image:

     ```
     docker tag <MATLAB_IMAGE_NAME>:<MATLAB_IMAGE_TAG> <MATLAB_IMAGE_NAME>:<MATLAB_IMAGE_TAG_UPDATED>
     ```

     For example:

     ```
     docker tag containers.mathworks.com/matlab-online-server/mos-matlab-image:1.13.0
     containers.mathworks.com/matlab-online-server/mos-matlab-image:1.13.0-backup
     ```

   - For a remote registry, run the following command also:

     ```
     docker push <MATLAB_IMAGE_NAME>:<MATLAB_IMAGE_TAG_UPDATED>
     ```

     For a remote registry, the two commands together might look similar to this example:

     ```
     docker tag remote-registry.com/matlab-online-server/mos-matlab-image:1.13.0
     remote-registry.com /matlab-online-server/mos-matlab-image:1.13.0-backup

     docker push remote-registry.com /matlab-online-server/mos-matlab-image:1.13.0-backup
     ```

3. Use `mosadm` to build a new MATLAB Docker image. Run this command, replacing *`MATLAB-install-location`* with your actual MATLAB install location:

   ```
   mosadm build-matlab-image MATLAB-install-location
   ```

   This command overrides the existing MATLAB image.
4. Use `mosadm` to redeploy the MATLAB Pool pod using the following commands:

   ```
   mosadm undeploy matlab-pool

   mosadm deploy matlab-pool
   ```

See Also

Related Examples

- “Update MATLAB Online Server” on page 4-10

Troubleshooting

Contact Technical Support About MATLAB Online Server Issues

If you experience technical issues when installing, configuring, or maintaining MATLAB Online Server, email `matlab-online-server-support@mathworks.com`.

In the email, provide system information, including UNIX® system information, the present working directory (`pwd`), and disk usage. To obtain this information, use the `mosadm gather-system-info` command. This command outputs the information to the terminal and logs it to the `install.log` file.

```
./mosadm gather-system-info
```

In addition to system information, include in your email as much of the following information as you can:

- MATLAB Online Server license number
- MATLAB Online Server version
- Debugging logs
- MATLAB version and all supplemental toolbox names

See Also

Related Examples

- “Manage Server Logs” on page 5-3

Manage Server Logs

Each microservice in MATLAB Online Server outputs log information as it runs. Checking log information can help debug issue that occur while the server is running.

Check Logs

You can get the content of server logs by using the file system, dumping log information to system output, or by streaming information to system output.

Access Logs Using the File System

Access the single node on which you installed MATLAB Online Server (for example, using ssh), and then access the folder /var/log/containers/. This folder contains logs for all the MATLAB Online Server microservices.

Issue the following command, replacing *`your-namespace`* with the namespace you used for MATLAB Online Server:

```
ls /var/log/containers/your-namespace
```

Examples:

```
your-namespace-core-ui-6679d5ff7f-rdtfb_your-namespace_core-ui-165cfda8ff5c4d2b01838d0acb573df881e04a08c329afcda1b96b9c15
your-namespace-gateway-65697586f7-vsvls_your-namespace_gateway-78afff6773618381f1fc6ba05e63eceaf3f3a6fd5de3defd6f50b28a97
your-namespace-matlab-pool-apparmor-loader-zdqwk_your-namespace_apparmor-loader-87115017a99407c3bf647712c61426d9f226a8b4b
your-namespace-matlab-pool-helpsearch-5689dd4cdf-t77rh_your-namespace_helpsearch-963806b3d47815d8c7df217770b38be10c37fc30
your-namespace-matlab-pool-ui-689fb5d9ff-tsrh4_your-namespace_webgui-ec90a23bbd0db048115f6fae4113084a8df6d0ef860fc03dc54e
```

Dump Logs to System Output

Use the `kubectl` command to access the logs for individual pods. The following example demonstrates the command to dump the content of the logs belonging to the `matlab-pool-helpsearch` pod. Replace *`your-namespace`* with the namespace you used for MATLAB Online Server.

```
kubectl --namespace= your-namespace logs your-namespace -matlab-pool-helpsearch-5689dd4cdf-t77rh
```

If a pod has more than one container, for example, matlab-pool, you can first extract the list of its containers, and then dump the content of its container logs by specifying the name of the container.

For example, to get the list of containers for the MATLAB Pool pod, enter the following command, replacing *`your-namespace`* with the namespace you used for MATLAB Online Server:

```
kubectl --namespace= your-namespace get pods your-namespace -matlab-pool-5858d796b9-lqslb
-o jsonpath='{.spec.containers[*].name}'
```

The results list four containers inside the MATLAB Pool pod. The containers are listed left to right with a space between each container name:

```
matlab resource-proxy display windowmanager
```

To get logs for the MATLAB container that is running inside the MATLAB Pool pod, run the following command, replacing *`your-namespace`* with the namespace you used for MATLAB Online Server:

```
kubectl --namespace= your-namespace logs your-namespace -matlab-pool-5858d796b9-lqslb -c matlab
```

Stream Logs to System Output

To continuously monitor the contents of logs, stream their contents by adding the -f flag to the kubectl command as follows, replacing *your-namespace* with the namespace you used for MATLAB Online Server:

```
kubectl --namespace= your-namespace logs -f your-namespace -matlab-pool-helpsearch-5689dd4cdf-t77rh
kubectl --namespace= your-namespace logs -f your-namespace -matlab-pool-5858d796b9-lqslb -c matlab
```

Configure Log Shipping

MATLAB Online Server does not come with any log shipping capabilities. To persist the MATLAB Online Server application logs for debugging and future use or in integrating with Splunk®, follow the procedures described earlier.

Option 1: Persistence of Logs on Host

MATLAB Online Server comprises various services. Logs from multiple services can be fetched using kubectl and stored in files on disk.

For logs of MATLAB pods that are in use, perform the following steps:

1. Run the following command to get the status of in-use pods:

   ```
   kubectl get pods -l inUse=true
   ```

   This action returns the status of pods in use:

   ```
   NAME                                        READY   STATUS    RESTARTS   AGE
   your-namespace-matlab-pool-f479b5b96-d6kfk   2/2     Running   0          90m
   ```

2. To retrieve the logs of a particular pod, run the following commands, replacing *your-namespace* with the namespace you used for MATLAB Online Server, depending on which logs you want to retrieve:

   - License service logs

     ```
     kubectl logs <license_pod_name> -n your-namespace >> <path_to_license_log_file_on_host>
     ```

   - AuthNZ service logs

     ```
     kubectl logs <authnz_pod_name> -n your-namespace >> <path_to_authnz_log_file_on_host>
     ```

   - MATLAB logs

     ```
     kubectl logs <matlab_pod_name> -n your-namespace -c matlab >> <path_to_matlab_log_file_on_host>
     ```

   - Resource proxy logs

     ```
     kubectl logs <matlab_pod_name> -n your-namespace -c resource-proxy >>
     <path_to_resource_proxy_log_file_on_host>
     ```

These examples illustrate storing log files on the host. You can persist logs on any other location, for example, an NFS server of your organization, where they can be exported with logging tools of your choice for more analysis and for setting up alerts.

Option 2: Integrate with Splunk (Advanced)

Follow these instructions to integrate Splunk with MATLAB Online Server. For more information on Splunk, see Splunk Connect for Kubernetes on EKS!.

Note Depending on the platform you have chosen, you might have to make changes for Splunk integration to work. For resources, check the documentation from your product vendor.

1. Create a minimum of two Splunk indices:
   - One events index, which handles logs and objects (you can also create two separate indices for logs and objects)
   - One metrics index

   If you do not configure these indices, Splunk Connect for Kubernetes uses the defaults created in your HEC (HTTP Event Collector) token.
2. Create an HEC token if you do not already have one.
3. Create a namespace for Splunk Connect for Kubernetes with the following command:

   ```
   kubectl create ns mos-splunk-connect-k8s
   ```
4. Create the file `values.yaml` with the following contents (type lines exactly as shown):

```
kubernetes:
 clusterName: <cluster-name>
logLevel: warn
 global:
  splunk:
   hec:
    token: <hec-token>
    host: <splunk-host>
    port: <splunk-port>
    protocol: http
    insecureSSL: true
    indexName: <splunk-index-name>
journalLogPath: /var/log/journal
logs:
 matlab-pool:
  from:
   pod: "*-matlab-pool"
   container: matlab
  multiline:
   firstline: /^\[#\|\d{4}-\d{1,2}/
  bootstrap:
   from:
    journald:
     unit: kubeadm.service
   timestampExtraction:
    regexp: \w(?<time>[0-1]\d[0-3]\d [^\s]*)
     format: "%m%d %H:%M:%S.%N"
    sourcetype: kube:kubeadm
  resource-proxy:
   from:
    pod: "*-matlab-pool"
    container: resource-proxy
   multiline:
    firstline: /^\[#\|\d{4}-\d{1,2}/
   bootstrap:
    from:
     journald:
      unit: kubeadm.service
     timestampExtraction:
      regexp: \w(?<time>[0-1]\d[0-3]\d [^\s]*)
      format: "%m%d %H:%M:%S.%N"
     sourcetype: kube:kubeadm
```

5 Install Splunk Connect for Kubernetes.

```
helm install --name mos-splunk-connect --namespace mos-splunk-connect-
k8s -f values.yaml https://github.com/splunk/splunk-connect-for-
kubernetes/releases/download/1.3.0/splunk-connect-for-kubernetes-1.3.0.tgz
```

Note This procedure is just one method to integrate Splunk with Kubernetes. There are other methods that you can choose based on your requirements.

There are also various log shipping options with Kubernetes that you can choose to run, depending on your organization's needs and availability of tools.

Configure Log Rotation

Kubernetes does not rotate logs automatically. You can configure log rotation manually. For example, you can deploy logrotate as a daemon service on a MATLAB Online Server node and then configure it to run each hour.

MATLAB Online Server logs are under the folder `/var/log/containers/`. Use the logrotate config file to configure the logs. By default, this file is `/etc/logrotate.conf`. For more information, see “Check Pod Status” on page 4-5.

See Also

`mosadm gather-system-info`

Related Examples

- “Contact Technical Support About MATLAB Online Server Issues” on page 5-2

Resolve MATLAB Pod Issues

The MATLAB pod comprises four Docker containers, one of which is MATLAB Runtime.

When the services are running successfully and have no issues, then the status of pods is `Running`.

However, if the services are having issues, pods can still show the status `Running`. This can be because, for example, the MATLAB Docker image was built incorrectly, or there are issues connecting with the network license manager server.

Run the following command to display pod status, replacing *`your-namespace`* with the namespace you used for MATLAB Online Server:

```
kubectl get pods --namespace your-namespace
```

This command outputs information similar to that shown in the following example:

```
 NAME                                              READY    STATUS    RESTARTS      AGE
your-namespace-core-ui-core-ui-5bd9b7c88b-fqbl4    1/1     Running     0          111s
your-namespace-authnz-authnz-7c7999589c-x74d9      1/1     Running     0          111s
your-namespace-gateway-669dd7865f-t6l74            1/1     Running     0          110s
your-namespace-license-548f7b6655-8skdt            1/1     Running     0          108s
your-namespace-matlab-pool-8c679f76-c4h4n          2/2     Running     0          107s
your-namespace-matlab-pool-8c679f76-cckkp          2/2     Running     0          107s
your-namespace-matlab-pool-ui-579fc8ff49-2bz25     1/1     Running     0          107s
your-namespace-ingress-nginx-controller-kcwwd      1/1     Running     0          104s
your-namespace-resource-resource-894447df7-r9ngb   1/1     Running     0          103s
```

Problems and Resolutions

Check the possible problems and solutions in this section to help diagnose and resolve common issues.

License Checkout Issues

Although a license checkout error is a failing scenario, the MATLAB pod status still shows as Running because a prewarmed pool of MATLAB instances might run successfully but fail to check out a license when a user signs in.

To determine if a license checkout error is the issue, perform the following assessment:

1. Run the following command to get the MATLAB container logs, replacing *`matlab_pod_name`* with the name of your MATLAB Online Server MATLAB pod and *`your-namespace`* with the namespace you used for MATLAB Online Server:

   ```
   kubectl logs matlab_pod_name --namespace your-namespace -c matlab
   ```

2. Check for License Manager Error -<n>: License Checkout Failed in the logs. If this error has occurred, then it is highly likely that MATLAB is having issues checking out the license.

   Logs can be lengthy. You can use a text-searching tool to get the information you need faster. For example, you can search the logs for a license checkout error:

   ```
   kubectl logs matlab_pod_name --namespace your-namespace -c matlab | grep 'License Checkout'
   ```

   You can also search for a license manager error:

   ```
   kubectl logs matlab_pod_name --namespace your-namespace -c matlab | grep 'License Manager Error'
   ```

If you see License Manager Error -<n>: License Checkout Failed in the logs, check the specific error message, and then consult the following table.

Error Message	Resolution Steps
Invalid license file syntax	Check to make sure that the license file syntax is appropriate (that is, check the network license manager logs). Then, making sure that you use the correct license file, try again.

Cannot connect to license server	• Try to connect to the license server locally.
	• Check to see if the network firewall rules for the license server are set correctly to allow connection from the MATLAB Online Server cluster.
	• Network policy on a MATLAB pod is enabled by default, which allows it to talk to the network license manager only on ports 27000 and 27001 by default (controlled by license.dat). Make sure your network license manager server is listening on those ports. If your server is listening on other ports, you need to change the following configuration in the `matlab-pool` override and then restart the pod:
	`networkPolicy:`
	`  enabled: true`
	`  additionalAllowedPorts:`
	`  - port: <flexlm_port_here>`
	`  protocol:`

Incorrect MATLAB Image

The MATLAB Image is built from a local MATLAB install and can sometime have issues if it was built from an incorrect MATLAB location or version that is not supported by MATLAB Online Server.

To determine if an incorrect MATLAB image is the problem, perform the following assessment.

1. Run the following command to display a pod status list, replacing *`your-namespace`* with the namespace you used for MATLAB Online Server:

   ```
   kubectl get pods --namespace your-namespace
   ```

2. Examine the pod status information, as shown in this example:

   ```
   NAME                                              READY   STATUS    RESTARTS   AGE
   your-namespace-authnz-authnz-569df55f78-pqklk     1/1     Running   0          47h
   your-namespace-core-ui-core-ui-84c59ddd45-fwf4t   1/1     Running   0          47h
   your-namespace-gateway-84dbf7c8d-6gjcz            1/1     Running   0          47h
   your-namespace-license-85ffb6c595-q6x4f           1/1     Running   0          47h
   your-namespace-matlab-pool-apparmor-loader-jzqsg  1/1     Running   0          40s
   your-namespace-matlab-pool-f479b5b96-59xmj        3/4     Running   0          40s
   your-namespace-matlab-pool-helpsearch-546cb5fd8-jkx4p  1/1     Running   0          40s
   your-namespace-matlab-pool-ui-7b6d688f98-rfdz5    1/1     Running   0          40s
   your-namespace-ingress-nginx-controller-5c6b5f775f-7ls87  1/1     Running   0          27h
   your-namespace-resource-resource-8554f87bbf-g4ssq   1/1     Running   0          47h
   ```

3. If you see MATLAB pods in a 3/4 Running state, as shown in the example log, run the following command to learn why. Replace *`matlab_pod_name`* with your MATLAB Online Server MATLAB pod name and *`your-namespace`* with the namespace you used for MATLAB Online Server:

```
kubectl logs matlab_pod_name --namespace your-namespace -c matlab
```

This command outputs a log similar to this example:

```
Starting MATLAB
+ exec /MATLAB/bin/matlab -prewarm -c '"27000@flexlm.mwcloudtest.com"' -nodesktop -softwareopengl -nosplash
MATLAB is selecting SOFTWARE OPENGL rendering.
...

< M A T L A B (R) >
Copyright 1984-2023 The MathWorks, Inc.
...

Warning: Unrecognized command line option: prewarm.

To get started, type doc.
For product information, visit www.mathworks.com.
```

The logs show that MATLAB was not able to start successfully. To verify that this is what happened, you can run the following command to look at the logs of resource-proxy, which is another container in the MATLAB pod that proxies requests to MATLAB.

Issue this command, replacing *`matlab_pod_name`* with your MATLAB Online Server MATLAB pod name and *`your-namespace`* with the namespace you used for MATLAB Online Server:

```
kubectl logs matlab_pod_name --namespace your-namespace -c resource-proxy
```

Examine the logs output by this command. If you see the following messages, the backend (in this case, MATLAB) is not ready and is not functioning properly.

```
YYYY-MM-DD 19:06:51,037 INFO matlabonlineserver.resource_proxy Backend is not ready
YYYY-MM-DD 19:06:51,507 INFO matlabonlineserver.resource_proxy Backend is not ready
YYYY-MM-DD 19:06:54,498 WARNING matlabonlineserver.http.client Jetty request failed: Connection refused
YYYY-MM-DD 19:07:04,499 WARNING matlabonlineserver.http.client Jetty request failed: Connection refused
YYYY-MM-DD 19:07:11,037 INFO matlabonlineserver.resource_proxy Backend is not ready
YYYY-MM-DD 19:07:11,507 INFO matlabonlineserver.resource_proxy Backend is not ready
YYYY-MM-DD 19:07:14,500 WARNING matlabonlineserver.http.client Jetty request failed: Connection refused
YYYY-MM-DD 19:07:24,500 WARNING matlabonlineserver.http.client Jetty request failed: Connection refused
YYYY-MM-DD 19:07:31,037 INFO matlabonlineserver.resource_proxy Backend is not ready
YYYY-MM-DD 19:07:31,507 INFO matlabonlineserver.resource_proxy Backend is not ready
```

In the example log, the message `"Backend is not ready"` indicates that MATLAB is not yet ready.

For resolution, check to see if you are building the MATLAB image from the correct MATLAB install location.

Storage Mount Errors

When all services and MATLAB are in a Running state, but you are encountering issues using MATLAB Online, it is possible that there was a problem with mounting the storage on MATLAB at the time of user sign on.

To determine if storage mounting on MATLAB is the problem, perform the following assessment:

1. Run the following command to generate pod logs, replacing *`matlab_pod_name`* with the name of your MATLAB Online Server MATLAB pod and *`your-namespace`* with the namespace you used for MATLAB Online Server.

   ```
   kubectl logs matlab_pod_name --namespace your-namespace -c resource-proxy
   ```

2. Examine the output from this command:

   ```
   YYYY-MM-DD 20:44:39,124 INFO matlabonlineserver.storage Setting up the profile: nfsHome
   YYYY-MM-DD 20:44:39,124 INFO matlabonlineserver.storage Setting up the mount: home
   YYYY-MM-DD 20:44:39,124 INFO matlabonlineserver.storage Starting NFS mount
   ```

```
YYYY-MM-DD 20:44:39,125 INFO matlabonlineserver.storage Executing: mount -o rw,
relatime,vers=3,rsize=1048576,wsize=1048576,namlen=255,acregmin=600,acregmax=600,
acdirmin=600,acdirmax=600,hard,nocto,noacl,proto=tcp,timeo=600,retrans=2,mountproto=tcp,
local_lock=none,nolock nfs:/exports/home/oin /mounts/homeoin
YYYY-MM-DD 20:44:49,167 INFO matlabonlineserver.storage mount exit value= 137, mount result= , mount error=
```

3. If you see the mount error message, as displayed in the last line of the example shown, check that the ports allowed by the network policy for MATLAB Pool include ports for your NFS server setup. For more details on network policies, see “Configure Network Policies” on page 3-62.

MATLAB Pod Not in Running State

If none of the possible issues described in this section help you to resolve the problem, try the following:

- See “Resolve MATLAB Pod Issues” on page 5-8.
- Contact MathWorks Support.

1. Run the following `kubectl` commands, replacing *`license_pod_name`* with the name of the pod that is stuck and *`your-namespace`* with the namespace you used for MATLAB Online Server:

   ```
   kubectl describe pod license-pod-name --namespace your-namespace
   kubectl logs license-pod-name --namespace your-namespace
   ```

2. Keep the output nearby to discuss with a support technician.

See Also

Related Examples

- “Resolve Pod Creation Stuck Issues” on page 5-16
- “Resolve Evicted or Terminated Pod Issues” on page 5-20
- “Resolve Out-of-Resource Conditions” on page 5-22

Resolve License Service Issues

MATLAB Online Server requires you to have valid licenses for both MATLAB Online Server and MATLAB before you install either. After you have completed installation with valid licenses, your users can sign in successfully to MATLAB Online.

License Service Pod State is Running but Users Cannot Connect to MATLAB Online

When the services are running successfully and can connect with the network license manager license server, then the status of pods is Running.

However, if the services are having issues connecting with network license manager, the status of the pods is still shown as Running. This is expected, and it means that the MATLAB Online Server is running successfully but is not authorized because it is unable to connect to the network license manager.

```
NAME                                                  READY   STATUS    RESTARTS   AGE
your-namespace-authnz-7994c9866d-675fb                1/1     Running   0          10m
your-namespace-core-ui-cfdcccc4c-5bhrc                1/1     Running   0          10m
your-namespace-gateway-88ffd446d-mbf2l                1/1     Running   0          10m
your-namespace-gateway-proxy-6f85db9cbb-8ftbr         1/1     Running   0          10m
your-namespace-gateway-proxy-6f85db9cbb-mdhr7         1/1     Running   0          10m
your-namespace-ingress-nginx-controller-7fdcd49d74-bnbls 1/1     Running   0          10m
your-namespace-license-5cc85b97cd-zg4vd               1/1     Running   0          10m
your-namespace-matlab-pool-9cc6b6465-9rdz8            2/2     Running   0          10m
your-namespace-matlab-pool-9cc6b6465-t7wp2            2/2     Running   0          10m
your-namespace-matlab-pool-helpsearch-8479fbdc88-4r6sd   1/1     Running   0          10m
your-namespace-matlab-pool-ui-8484bbbd4d-t6777        1/1     Running   0          10m
your-namespace-resource-78f9b97745-fzwlq              1/1     Running   0          10m
```

If the license service is not operational, other services also do not work as expected. This is because the license service checks out the MATLAB Online Server license, which in turn allows all other services to run successfully.

Get More Information

Run the following command to get more information, replacing *`license-pod-name`* with the name of your license pod and *`your-namespace`* with the namespace you used for MATLAB Online Server:

```
kubectl logs license-pod-name --namespace your-namespace
```

If you get output similar to the following display, then it is likely that the license service is unable to check out the MATLAB Online Server license.

```
00000006 YYYY-Mmm-DD 21:10:55.104617 0x00007fa8d8fd5700 mos::license I
Checking out license
00000007 YYYY-Mmm-DD 21:10:55.104626 0x00007fa8d8fd5700 mos::license I
Server: 27000@flexlm
00000008 YYYY-Mmm-DD 21:11:15.237586 0x00007fa8d8fd5700
startup::license_manager I enter checkout for feature: MATLAB_Online_Server
The server is running.
Error while trying to check out the license: , will retry in 30 seconds.
00000009 YYYY-Mmm-DD 21:11:28.314607 0x00007fa8d8fd5700 mos::license I
Registered service
00000010 YYYY-Mmm-DD 21:11:56.040342 0x00007fa8d203c700 mos::license I Check
license request mathworks-gateway-669dd7865f-t6l74
00000011 YYYY-Mmm-DD 21:11:58.430979 0x00007fa8c71fa700 mos::license I
Checking out license
00000012 YYYY-Mmm-DD 21:11:58.431050 0x00007fa8c71fa700 mos::license I
Server: 27000@flexlm
00000013 YYYY-Mmm-DD 21:12:03.866191 0x00007fa8d303e700 mos::license I Check
license request from mathworks-authnz-authnz-7c7999589c-x74d9
```

```
00000014 YYYY-Mmm-DD 21:12:04.565262 0x00007fa8c71fa700
startup::license_manager I enter checkout for feature: MATLAB_Online_Server
00000015 YYYY-Mmm-DD 21:12:04.869012 0x00007fa8d283d700 mos::license I Check
license request from mathworks-resource-resource-894447df7-r9ngb
Error while trying to check out the license: ^^, will retry in 30 seconds.
```

Possible Issues

- MATLAB Online Server is unable access license server when both servers are on the same machine

  If you installed MATLAB Online Server and the network license manager server on the same machine, license checkouts from MATLAB or MATLAB Online Server cluster might be blocked because the firewall prevents access to the license server ports.

  One workaround is to modify the firewall to enable access to the network license manager ports. By default, the network license manager listens on ports 27000 and 27001. For example, to enable these ports in Ubuntu, use this command:

  ```
  sudo ufw allow 27000 && sudo ufw allow 27001
  ```

- Network license manager server listening on ports other than 27000/27001

  Network policy is enabled by default for the license service, which allows communicating with network license manager only on port 27000 and with the MLM daemon on 27001. The default policy assumes that the license file has the following SERVER lines:

  ```
  SERVER <HOSTID> ID=0 27000
  DAEMON MLM "<FLEXLM_INSTALL_LOCATION>/etc/MLM" port=27001
  ```

  If your network license manager server is listening on ports other than 27000 and the MLM daemon on other than 27001, update the license service override file and then restart the pod:

  ```
  networkPolicy:
    enabled: true
    egress:
       ports:
       - port: <flexlm_port_here>
         protocol: TCP
  ```

- The network license manager server is using an incorrect or an expired license file or the network license manager server is misconfigured

  Check the network license manager server logs to see details about the reason for failure. Try to connect to the network license manager server from a local machine using the following commands:

  ```
  nc -vz <FelxLM_DNS_NAME> 27000
  nc -vz <FelxLM_DNS_NAME> 27001
  ```

  In cases where the network license manager server is using an expired license file:

  - Update the license file. See “Update Network License” in the Help Center.
  - Set up both a listener on the network license manager log file and an alert, so that you are notified when the license is about to expire.
  - If the license service pod is not in the Running state, see “Resolve License Service Issues” on page 5-12.

- See “Resolve MATLAB Pod Issues” on page 5-8.

License Checkout Fails Due to Missed Heartbeats

If you are running MATLAB Online Server on a cloud-managed Kubernetes cluster, such as through Azure AKS, the license service might unexpectedly exceed the license checkout count. Exceeding the license count results in failed license checkouts.

When this issue occurs, the logs for the license service contain several `"missed heartbeat"` warnings. A heartbeat is when the license service periodically pings the Kubernetes cluster to confirm that the licenses for the MATLAB Online Server or MATLAB instances running inside it are still checked out. If the license service misses a heartbeat, meaning it is unable to connect, the service checks out a new license. After too many missed heartbeats and license checkouts, the service exceeds the checkout count.

To see if the failed checkouts are due to missed heartbeats, run this command to check the log for the license service:

`kubectl logs` *`license-pod-name`* `--namespace` *`namespace`*

where:

- *`namespace`* is the MATLAB Online Server namespace as listed in the `install.config` file (default = `"mathworks"`).
- *`license-pod-name`* is the name of the pod running the license service. To get the license pod name, run `kubectl get pods --namespace` *`namespace`*.

If the log contains repeated missed heartbeat warnings, such as the ones shown here, then the issue is caused by Kubernetes closing its connection to the license service after a period of inactivity.

```
00000034 YYYY-Mmm-DD 09:26:32.863576 0x00007f2c08c8v400 mos::license I Missed heartbeat: 2
00000035 YYYY-Mmm-DD 09:26:32.863640 0x00007f2c08c8v400 mos::license I Current state is: 1
00000036 YYYY-Mmm-DD 09:27:32.643434 0x00007f2c08c8v400 mos::license I Missed heartbeat: 3
00000037 YYYY-Mmm-DD 09:27:32.643488 0x00007f2c08c8v400 mos::license I Current state is: 1
00000038 YYYY-Mmm-DD 09:28:32.700154 0x00007f2c08c8v400 mos::license I Missed heartbeat: 4
00000039 YYYY-Mmm-DD 09:28:32.700234 0x00007f2c08c8v400 mos::license I Current state is: 1
00000040 YYYY-Mmm-DD 09:29:32.739331 0x00007f2c08c8v400 mos::license I Missed heartbeat: 5
00000041 YYYY-Mmm-DD 09:29:32.739374 0x00007f2c08c8v400 mos::license I Current state is: 1
00000042 YYYY-Mmm-DD 09:30:32.702587 0x00007f2c08c8v400 mos::license I Missed heartbeat: 6
00000043 YYYY-Mmm-DD 09:30:32.702636 0x00007f2c08c8v400 mos::license I Current state is: 1
00000044 YYYY-Mmm-DD 09:31:32.670748 0x00007f2c08c8v400 mos::license I Missed heartbeat: 7
00000045 YYYY-Mmm-DD 09:31:32.670799 0x00007f2c08c8v400 mos::license I Current state is: 1
00000046 YYYY-Mmm-DD 09:32:32.709340 0x00007f2c08c8v400 mos::license I Missed heartbeat: 8
00000047 YYYY-Mmm-DD 09:32:32.709388 0x00007f2c08c8v400 mos::license I Current state is: 1
00000048 YYYY-Mmm-DD 09:33:32.776049 0x00007f2c08c8v400 mos::license I Missed heartbeat: 9
00000049 YYYY-Mmm-DD 09:33:32.776103 0x00007f2c08c8v400 mos::license I Current state is: 1
00000050 YYYY-Mmm-DD 09:34:32.676145 0x00007f2c08c8v400 mos::license I Missed heartbeat: 10
00000051 YYYY-Mmm-DD 09:34:32.676193 0x00007f2c08c8v400 mos::license I Current state is: 1
00000052 YYYY-Mmm-DD 09:35:32.897212 0x00007f2c08c8v400 mos::license I Missed heartbeat: 11
00000053 YYYY-Mmm-DD 09:35:32.897262 0x00007f2c08c8v400 mos::license I Current state is: 1
```

To resolve this issue, specify a shorter interval between heartbeat checks.

In the `license.yaml` and `matlab-pool.yaml` override files, set the `heartbeatIntervalInSeconds` field. If left unspecified, the default interval is 600 seconds (10 minutes). Setting the interval to 120 seconds is sufficient to resolve the issue. The minimum interval is 60 seconds, but more frequent interval consume additional server resources.

The tables show the location of the YAML override files and the values to set.

***matlab-online-server-root*/overrides/*cluster*/*namespace*/license.yaml**

```
flexlm:
  servers: "27000@flexlm.company.com"
  heartbeatIntervalInSeconds: 120
```

***matlab-online-server-root*/overrides/*cluster*/*namespace*/matlab-pool.yaml**

```
flexlm:
  servers: "27000@flexlm.company.com"
  heartbeatIntervalInSeconds: 120
```

After making these changes, you must redeploy the license and MATLAB pool services. For example:

```
./mosadm undeploy license
./mosadm deploy license

./mosadm undeploy matlab-pool
./mosadm deploy matlab-pool
```

Get Help

If you are unable to resolve the issue with these resources, do the following:

1. Have ready the output from the following `kubectl` commands:

   ```
   kubectl describe pod <license-pod-name> --namespace <namespace>
   kubectl logs <license-pod-name> --namespace <namespace>
   ```

2. Contact MathWorks Support.

See Also

Related Examples

- “Contact Technical Support About MATLAB Online Server Issues” on page 5-2

Resolve Pod Creation Stuck Issues

During MATLAB Online Server installation and configuration, pods can sometimes get stuck in one of the following states:

To see pod status, run the following command, replacing *your-namespace* with the namespace you used for MATLAB Online Server:

```
kubectl get pods --namespace your-namespace
```

Consult the following sections for instructions on correcting the pod state when the pod is stuck during creation.

ImagePullBackOff

To find out the exact error message for why the pod status is in `ImagePullBackOff` status, run the following command, replacing *pod_name* with the name of the pod that is stuck and *your-namespace* with the namespace you used for MATLAB Online Server:

```
kubectl describe pod pod_name --namespace your-namespace
```

In the information returned, look at the `Events` section. The following table lists the most likely reasons for the `ImagePullBackOff` error.

Message Type	Sample Error Message	Reason
Invalid container image or no pull access for a private image	`Error response from daemon: repository foobartest4 not found: does not exist or no pull access`	This means either that the specified registry name is nonexistent, or you do not have access to the container. Make sure you have the correct name. If the name is correct, then check that the container registry for this image does not require authentication or, if it requires authentication, make sure that you have the secret properly configured. As a test, use the following command to try to pull the same image from your local machine: `docker pull <docker-image-name:docker-image-tag>`

Message Type	Sample Error Message	Reason
Invalid container image tag	`Warning Failed 10m (x4 over 12m) kubelet, mos-gar-3-pool-1-9781becc-bdb3 Failed to pull image "redis:foobar": rpc error: code = Unknown desc = Error response from daemon: manifest for redis:foobar not found`	This means that a specified image cannot be found. To test if the image has an appropriate tag, try to pull the image locally: `docker pull <docker-image-name:docker-image-tag>`

CrashLoopBackOff

This pod status means that your pod is starting, crashing, starting again, and then crashing again.

To get more information about the problem, run the following command, replacing *`pod_name`* with the name of the pod that is stuck and *`your-namespace`* with the namespace you used for MATLAB Online Server:

```
kubectl describe pod pod_name --namespace your-namespace
```

In the information returned, look at the `Events` section, which is the main part of the log that describes what is happening with the pod. Examples of output for the Events section can be found in “Check Pod Status” on page 4-5.

Next, look at the logs by running the following command, replacing *`pod_name`* with the name of the pod that is stuck:

```
kubectl logs pod_name
```

For a multi-container pod (for example, the MATLAB pod), run the command with the container name:

```
kubectl logs pod_name -c container_name
```

Logs can help you find a reason why the container is crashing. Look at the logs for Java exception or error messages. See “Get Help” on page 5-18 for assistance.

Pending

To get more information about why the pod is in a pending state, run the following command, replacing *`pod_name`* with the name of the pod that is stuck and *`your-namespace`* with the namespace you used for MATLAB Online Server:

```
kubectl describe pod pod_name --namespace your-namespace
```

The pod might be in a `Pending` state because there is not enough CPU, not enough memory, or not enough CPU *and* memory.

If there is not enough CPU, for example, you might find the following entry in the log:

```
Events:
  Type     Reason            Age               From            Message
```

```
  ----     ------            ----               ----               ----
  Warning  FailedScheduling  2s (x6 over 11s) default-scheduler  0/2
nodes are available: 4 Insufficient cpu
```

Possible conditions:

- You have requested more CPU than *any* of the nodes has. For example, if each node in the two-node cluster has 2 CPU cores and you request 4 CPU cores, even though, in total, the cluster capacity is 4 CPU cores, a single node does not meet the request of 4 CPU cores.

  Every pod needs to be scheduled on a single node with 4 CPU cores and because all nodes have only 2 CPU cores, the request cannot be met. Even if you turn on more nodes in your cluster, Kubernetes still is not able to schedule your request.
- There is no more capacity in the cluster for the CPU cores you have requested. For example, if each node in the two-node cluster has 1 CPU core and you request 1 CPU core, then two pods can be scheduled on each node because the requested CPU matches the node's capacity.

  However, if a third pod requests 1 CPU core, there is not enough CPU in the cluster, and you get the `FailedScheduling` state. If this happens, you can turn on more nodes (with 1 CPU core) in the cluster, and your pod can be scheduled.

ContainerCreating

When a pod is just starting, `ContainerCreating` is a normal state. However, it is not normal for a pod to be stuck in the `ContainerCreating` state for longer than a few seconds (approximately 10 to 20 seconds).

To get more information about the problem, run the following command, replacing *`pod_name`* with the name of the pod that is stuck and *`your-namespace`* with the namespace you used for MATLAB Online Server:

```
kubectl describe pod pod_name --namespace your-namespace
```

In the information returned, look at the Events section to find a possible reason for the pod to continue to be in the `ContainerCreating` state.

Get Help

If none of the above issues described in this section help you to resolve the issue, try the following:

- See “Resolve MATLAB Pod Issues” on page 5-8.
- Contact MathWorks Support.

  1. Run the following `kubectl` commands, replacing *`license_pod_name`* with the name of the pod that is stuck and *`your-namespace`* with the namespace you used for MATLAB Online Server:

     ```
     kubectl describe pod license-pod-name --namespace your-namespace
     kubectl logs license-pod-name --namespace your-namespace
     ```
  2. Keep the output nearby to discuss with a support technician.

See Also

Related Examples

- “Resolve MATLAB Pod Issues” on page 5-8
- “Resolve Evicted or Terminated Pod Issues” on page 5-20
- “Resolve Out-of-Resource Conditions” on page 5-22

Resolve Evicted or Terminated Pod Issues

How Pods Become Evicted

When available compute resources are low, some pods can get into an Evicted state. When resources become available on the node, these pods are automatically rescheduled.

To see pod status, run the following command, replacing *`your-namespace`* with the namespace you used for MATLAB Online Server:

```
kubectl get pods --namespace your-namespace
```

How to Clean Up Evicted Pods

- Run a Cron job to delete Evicted pods at regular intervals.
- Manually delete evicted pods using the following command, replacing *`your-namespace`* with the namespace you used for MATLAB Online Server:

  ```
  kubectl get pods --namespace your-namespace | grep Evicted | \
     awk '{print $1}' | xargs kubectl \
     delete pod --namespace your-namespace
  ```

How Pods Become Terminated

A Terminated state indicates that the container completed its execution and has stopped running. A pod enters the Terminated state when it has successfully completed execution or when it has failed.

Kubernetes cleans up Terminated pods when the number of pods exceeds the configured threshold, as determined by `terminated-pod-gc-threshold` in the `kube-controller-manager`. For a single-node installation, `terminated-pod-gc-threshold` is set to 10.

To see the status of pods, run the following command, replacing *`your-namespace`* with the namespace you used for MATLAB Online Server:

```
kubectl get pods --namespace your-namespace
```

How to Clean Up Terminated Pods

You can clean up pods in a `Terminated` state using one of these methods:

- Run a Cron job to delete `Terminated` pods at regular intervals.
- Manually delete terminated pods using the following command, replacing *`your-namespace`* with the namespace you used for MATLAB Online Server.

  ```
  kubectl get pods --namespace your-namespace | grep Terminated | \
     awk '{print $1}' | xargs kubectl \
     delete pod --namespace your-namespace
  ```

Occasionally, manual deletion does not actually delete the terminated pod. In this case, you might have to forcibly delete the pod. Run the following command, replacing *`your-namespace`* with the namespace you used for MATLAB Online Server:

```
kubectl get pods --namespace your-namespace | grep Terminated | \
   awk '{print $1}' | xargs kubectl \
   delete pod --namespace your-namespace --force --grace-period=0
```

For more about the `kubectl` command, see the kubectl Cheat Sheet at `kubernetes.io`.

See Also

Related Examples

- “Resolve Pod Creation Stuck Issues” on page 5-16
- “Resolve MATLAB Pod Issues” on page 5-8
- “Resolve Out-of-Resource Conditions” on page 5-22

Resolve Out-of-Resource Conditions

Depending on how the cluster resources are configured, under certain situations, the cluster might be out of physical resources, which can affect pod status.

To see pod status, run the following command, replacing *`your-namespace`* with the namespace you used for MATLAB Online Server:

```
kubectl get pods --namespace your-namespace
```

How Out-of-Resource Condition Can Occur

Some pod statuses that can indicate that the cluster is out of resources include `Pending`, `Evicted` (see “Resolve Evicted or Terminated Pod Issues” on page 5-20), and “ContainerCreating” on page 5-18.

Kubernetes can proactively monitor for and prevent total starvation of a compute resource. If this situation occurs during pod creation, Kubernetes can reclaim the starved resource by proactively failing one or more pods. The pod status then goes from `Running` to `Evicted`, and the new pods get stuck in the `ContainerCreating` state.

Possible Solution 1. Clean Up Unused Images

An out-of-resource condition can occur if you update the MATLAB image too many times without cleaning up unused images. For example, if the ephemeral storage of the Amazon EC2 instance (node) in AWS is approximately 100 GB, and you spin up a single-node MATLAB Online Server cluster successfully with all pods in the `Running` state, most capacity on the node is occupied by the MATLAB image (approximately 27 GB) for a full MATLAB install.

If you happen to update the image in a few weeks and then perform a MATLAB Pool update, the node downloads yet another approximately 27 GB image (assuming it is an image with a different name). If you do this a few more times without cleaning up unused images, you start seeing that some pods are getting into an `Evicted` state and the pod you are trying to start (MATLAB pod in this case) is stuck in the `ContainerCreating` state because it was able to pull the image but did not have enough resources (memory) to start the container.

Make sure unused images from the node are periodically cleaned up. Run the following command to see the images on the node:

```
docker images
```

Then, use either of the following commands to remove an unused image:

```
docker rmi <image-id>
```

or

```
docker rmi <image:tag>
```

Possible Solution 2. Provision More Nodes or Set Explicit Resource Limits

If resource limits for CPU and memory are not set, users can inadvertently exploit all available resources.

For example, assume that the MATLAB Pool pod is configured with no resource limits (default) as shown:

```
matlabResources:
  requests:
    cpu: 200m
    memory: 2Gi
  limits:
    #cpu: <cpu-limit-here>
    #memory: <memory-limit-here>
```

Under these conditions, when a MATLAB Online user performs an operation that maxes out the resources on the node, other end users cannot sign in to MATLAB Online, as the pod is stuck in the `ContainerCreating` state. Because there is a lack of resources, Kubernetes starts evicting some pods, putting them in the `Evicted` state.

Possible Action 1: Provision More Nodes

Provision more nodes on the cluster to meet the resource needs of the cluster.

Possible Action 2: Set Explicit Resource Limits

Set resource limits appropriately for the MATLAB Pool pod so that users cannot exploit all the resources.

1. Update the `matlab-pool` override file at `<overrides/matlab-online-server/mathworks/matlab-pool.yaml>`. Uncomment the limits section and then update with appropriate limit values.
2. Run the following `mosadm` command to update the node:

   ```
   mosadm upgrade matlab-pool
   ```

See Also

Related Examples

- “Resolve Pod Creation Stuck Issues” on page 5-16
- “Resolve MATLAB Pod Issues” on page 5-8
- “Resolve Evicted or Terminated Pod Issues” on page 5-20

Resolve Security Exception Error When Accessing Help

Problem

Your end user runs a doc command at the MATLAB command prompt, for example:

```
doc plot
```

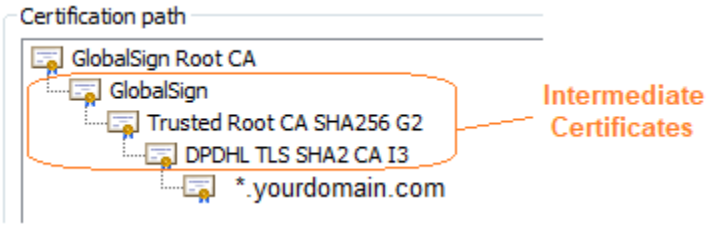
They see an error similar to the following:

```
Error using doc>displayDocPage (line 171)
Java exception occurred:
com.mathworks.webservices.client.core.MathWorksClientException: unable
to connect to host sun.security.validator.ValidatorException: PKIX
path building failed:
sun.security.provider.certpath.SunCertPathBuilderException: unable to
find valid certification path to requested target
. . .
```

Solution

This exception occurs when the Local CA certificate store is missing one or more intermediate CA certificates. For the client to validate the server, Root CA certificates must be available in the local certificate store. Without these intermediate Certificates, the client considers the secure connection untrusted and, in most cases, rejects it.

The figure below shows an example of a certificate path that can be viewed from the Chrome® browser. In this example, the Certificate of `*.yourdomain.com` has `GlobalSign Root CA` as root and three intermediate certificates: `GlobalSign`, `Trusted Root CA SH256 G2`, and `DPDHL TLS SHA2 CA I3`.



To resolve missing certificates:

1. Append the missing intermediate CA certificates to your public key file (for example, `publicKey.pem`).
2. Open your `publicKey.pem` file, and then append only the missing intermediate certificates to it in the reverse order. Do not append the root certificate to this file.

For this example, the file `publicKey.pem` must contains the intermediate certificates in this order:

```
-----BEGIN CERTIFICATE-----
       PDHL TLS SHA2 CA I3
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
       Trusted Root CA SH256 G2
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
       GlobalSign
-----END CERTIFICATE-----
```

See Also

Related Examples

- “Configure User Authentication in MATLAB Online Server” on page 3-6

Installation Configuration Properties

Installation Configuration Properties

MATLAB Online Server includes a configuration file that you can use to configure your installation. This file is named `install.config` and is located in the root of your MATLAB Online Server installation. For example, if you installed the server in your home directory, then it located at `~/matlab_online_server/install.config`.

The tables that follow lists the parameters you can modify or enable. If you edit this file while your server is running, you must redeploy the server for your changes to take effect.

Domain Name

Parameter	Description	Default
`DOMAIN_BASE`	Fully qualified domain name used to access MATLAB Online Server from a web browser. For example: `matlabonline.mycompany.com`. Specifying an IP address as the domain base is not supported. It is recommended that the domain name is registered to your organization and that the IP address of the server or load balancer are available on the DNS server of the organization. To test that your domain is working, it is possible to temporarily update the host entry of your client machine to have the IP address and the fully qualified domain name. Then, you can test accessing MATLAB Online Server instance from a browser in your client using this domain name.	`matlab.domain.com`

Docker

If you have a remote registry within your organization, uncomment these parameters and configure the following information about the registry.

Parameter	Description	Default
DOCKER_REGISTRY	Remote Docker registry to push Docker images to **Example**: myregistry.mycompany.com	docker-registry.com
DOCKER_REPOSITORY	Remote Docker repository **Example**: mos	repo-prefix
IMAGE_PULL_SECRET	User-friendly name for the Kubernetes secret object. When you deploy MATLAB Online Server, this object is created from registry credentials. This secret enables the nodes to pull images from the remote registry. **Example**: mymossecret	mwdockerregistry

Namespace

Parameter	Description	Default
NAMESPACE	Namespace to use for the deployment.	mathworks

Authentication

Parameter	Description	Default
ML_PASSWORD	MATLAB Online sign-in password. The username for this bootstrapping mode is admin. For security purposes, it is recommended that you replace the default administration password with a more secure one. Identity provider configuration for authenticating MATLAB Online users is covered in “Configure User Authentication in MATLAB Online Server” on page 3-6.	ML_PASSWORD=password

License Server

Parameter	Description	Default
`MOS_LICENSE_SERVER`	MATLAB Online Server license server. Specify the port and hostname IP address in the format `port@host`. **Note** If you have an Enterprise License, to correctly track usage, MathWorks requires two license servers: one for MATLAB Online Server (`MOS_LICENSE_SERVER` parameter) and one for MATLAB (`MATLAB_LICENSE_SERVER` parameter). With the two-server configuration, you must also have multiple pools (at least 2). See “Configure Multiple Versions of MATLAB” on page 3-21. **Example**: `14650@192.166.248.2`	`"27000@flexlm"`
`MATLAB_LICENSE_SERVER`	MATLAB license server details. Specify the port and hostname IP address in the format `port@host`. **Example**: `27000@172.0.0.1`	`"27000@flexlm"`

Security

Consider configuring your environment with Transport Layer Security (TLS). Leverage your organization's best practices for key management, certificate authority, and key rotation for MATLAB Online Server. For a local development environment and during configuration of your environment, it is acceptable to temporarily disable TLS to aid in initial deployment.

Parameter	Description	Default
`IS_TLS_ENABLED`	Flag to enable or disable Transport Layer Security (TLS), specified as `false` or `true`.	`false`

Parameter	Description	Default
TLS_KEY_FILE	Path to the TLS key file. This parameter applies only if IS_TLS_ENABLED is set to true.	/opt/tls.key
TLS_CERT_FILE	Path to the TLS certification file. This parameter applies only if IS_TLS_ENABLED is set to true.	/opt/tls.crt

Kubernetes

Parameter	Description	Default
KUBERNETES_CERTIFICATE_NAME	Name of a pre-existing Kubernetes certificate	"" (empty string)

Base Container Image

Parameter	Description	Default
BASE_IMAGE_FLAVOR	The base container images that are used to install and configure MATLAB Online Server, specified as debian or rhel.	debian

Offline Installation

Parameter	Description	Default
OFFLINE_MODE	Run MATLAB Online Server in offline mode, specified as true or false. For more details, see “Perform Offline MATLAB Online Server Installation” on page 2-12.	false

Client-Only Mode

Parameter	Description	Default
CLIENT_ONLY	Run MATLAB Online Server in client-only mode, specified as false or true.	false

Logging

Parameter	Description	Default
LOG_LEVEL	Log level of MATLAB Online Server, specified as `info` or `debug`. For more verbose logs, specify `LOG_LEVEL=debug`.	`info`

High Availability

Parameter	Description	Default
IS_HA_ENABLED	Option to enable or disable high availability (HA) for MATLAB Online Server, specified as `false` or `true`. For more details, see “Enable High Availability in MATLAB Online Server” on page 3-71.	`false`

Dashboard Deployment

Parameter	Description	Default
DEPLOY_DASHBOARDS	Option to deploy dashboards, specified as `false` or `true`.	`false`

Firewall

Parameter	Description	Default
`FIREWALL_ENABLED`	Option to enable or disable a firewall, specified as true or false. This parameter applies only to single-node installations. For more details, see: • “Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2 • “Perform Offline MATLAB Online Server Installation” on page 2-12 When this firewall is enabled, only ports 22 (SSH), 80 (HTTP) and 443 (HTTPS) are enabled. To communicate on any other port, you must use the ufw (uncomplicated firewall command). For example, this command enables communication on port 8443: `sudo ufw allow 8443`	`true`

See Also

Related Examples

- “Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2
- “Perform Offline MATLAB Online Server Installation” on page 2-12
- “Install MATLAB Online Server on Cloud-Managed Kubernetes” on page 2-16
- “Install MATLAB Online Server on Red Hat OpenShift” on page 2-26

System Administration Commands

mosadm bootstrap-node

Install Kubernetes and configure single-node Kubernetes cluster for MATLAB Online Server

Syntax

```
mosadm bootstrap-node
mosadm bootstrap-node option1 ... optionN
```

Description

`mosadm bootstrap-node` installs Kubernetes, initializes a single-node Kubernetes cluster on the machine where it is being installed, and customizes the cluster for MATLAB Online Server.

The exact commands executed by this command depend on the operating system:

- For Ubuntu systems, see “Bootstrap Node for Ubuntu Systems” on page 7-3
- For Red Hat or CentOS systems, see “Bootstrap Node for Red Hat or CentOS Systems” on page 7-5

`mosadm bootstrap-node option1 ... optionN` initializes the node using the specified configuration options.

Note `mosadm bootstrap-node` requires `sudo` administration privileges.

Examples

Bootstrap Node

```
sudo ./mosadm bootstrap-node
```

Input Arguments

`option1 ... optionN` — One or more configuration options

strings

One or more configuration options, specified as strings corresponding to valid configuration options from this table.

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.

Option	Description
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm bootstrap-node` to the command line.
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet, -q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.
`--skip-log-file`	Skip creating `install.log` and logging data to it.
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

More About

Bootstrap Node for Ubuntu Systems

When bootstrap-node is executed on an Ubuntu distribution, the following changes are applied to the machine.

After the operating system is configured with the following settings, the Kubernetes single-node cluster is bootstrapped.

- Coredns is installed (to allow DNS lookups within the Kubernetes pods)
- Kube-proxy is removed
- Kube-router is installed as the container network interface and proxy server, and the network policy provider.
- The current node is untainted to allow Kubernetes to schedule pods.

Extra repositories

The following repositories are added to the package manager apart from the pre-built defaults:

- https://packages.cloud.google.com/apt/doc/apt-key.gpg
- https://packages.cloud.google.com/apt/dists/kubernetes-xenial/main

Updates

`apt-get update` is called within the script to update all the base utilities of the operating system.

Installed Utilities

The following operating system utilities are installed as part of the bootstrap process:

- `unzip`
- `gettext`
- `jq`
- `ca-certificates`
- `curl`
- `software-properties-common`
- `git`
- `nfs-common`
- `ipvsadm`
- `kubelet` (1.22.5)
- `kubeadm` (1.22.5)
- `kubectl` (1.22.5)
- `helm` (3.7.0)

Networking

The script allows and configures the following:

- Bridge networking
- IP forwarding
- Enables the following IPVS modules:
  - Ip_vs_dh
  - ip_vs_ftp
  - ip_vs
  - ip_vs_lblc
  - ip_vs_lblcr
  - ip_vs_lc
  - ip_vs_nq
  - ip_vs_rr
  - ip_vs_sed
  - ip_vs_sh

  - ip_vs_wlc
  - ip_vs_wrr
- Enables firewall
  - By default, only ssh (22), http (80), and https (443) are allowed.
  - All other ports are disabled if `ufw` is already installed (on Ubuntu distribution, `ufw` is installed by default).

Docker Daemon

Unless bootstrap-node is executed with the flag `–preserve-docker-config`, bootstrap node creates or overwrites the file `/etc/docker/daemon.json` with the following properties:

```
{
      "exec-opts": ["native.cgroupdriver=systemd"],
      "log-driver": "json-file",
      "log-opts": {
        "max-size": "100m"
      },
      "storage-driver": "overlay2"
}
```

For more details, refer to the topic Daemon configuration file on Docker docs for more details.

Miscellaneous

- Enables high precision timestamps in syslog.
- Disables swap (disabling swap is required by the Kubernetes/kubelet to bootstrap).

Bootstrap Node for Red Hat or CentOS Systems

Note For Red Hat or CentOS, set the SELINUX profile to permissive mode. The following commands can be executed with sudo privileges.

```
# Set SELinux in permissive mode (effectively disabling it)
1  setenforce 0
2  sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
```

When bootstrap-node is executed on Red Hat or CentOS distributions, the following changes are applied to the machine.

After the operating system is configured with the following settings, Kubernetes single-node cluster is bootstrapped.

- Coredns is installed, to allow DNS lookups within the Kubernetes pods.
- Kube-proxy is removed.
- Kube-router is installed as the container network interface and proxy server, and the network policy provider.
- The current node is untainted to allow Kubernetes to schedule pods.

Extra repositories

The following repositories are added to the package manager apart from the pre-built defaults:

- https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64
  - https://packages.cloud.google.com/yum/doc/yum-key.gpg (key)
  - https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg (key)

Updates

`yum update` is called within the script to update all the base utilities of the operating system.

Installed Utilities

The following operating system utilities are installed as part of the bootstrap process:

- `unzip`
- `gettext`
- `jq`
- `ca-certificates`
- `curl`
- `software-properties-common`
- `git`
- `nfs-common`
- `ipvsadm`
- `kubelet` (1.22.5)
- `kubeadm` (1.22.5)
- `kubectl` (1.22.5)
- `helm` (3.7.0)

Networking

The script allows and configures the following:

- Bridge networking
- IP forwarding
- Enables the following IPVS modules:
  - Ip_vs_dh
  - ip_vs_ftp
  - ip_vs
  - ip_vs_lblc
  - ip_vs_lblcr
  - ip_vs_lc
  - ip_vs_nq
  - ip_vs_rr

  - ip_vs_sed
  - ip_vs_sh
  - ip_vs_wlc
  - ip_vs_wrr
  - br_netfilter
- Enables firewall
  - By default, only ssh (22), http (80), and https (443) are allowed.
  - All other ports are disabled if `firewalld` is already installed (on Red Hat distribution, `firewalld` is installed by default).

Docker Daemon

Unless bootstrap-node is executed with the flag `–preserve-docker-config`, bootstrap node creates or overwrites the file `/etc/docker/daemon.json` with the following properties:

```
{
      "exec-opts": ["native.cgroupdriver=systemd"],
      "log-driver": "json-file",
      "log-opts": {
        "max-size": "100m"
      },
      "storage-driver": "overlay2"
}
```

For more details, refer to the topic Daemon configuration file on Docker docs for more details.

Miscellaneous

- The script enables high precision timestamps in syslog.
- Disables swap (disabling swap is required by the Kubernetes/kubelet to bootstrap).

Version History

Introduced in R2020a

See Also

`mosadm reset-node` | `mosadm renew-node`

Topics

“Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2

mosadm build-matlab-image

Build MATLAB Docker image for use in MATLAB Online Server

Syntax

```
mosadm build-matlab-image
mosadm build-matlab-image matlabInstallLocation
mosadm build-matlab-image ___ option1 ... optionN
```

Description

`mosadm build-matlab-image` builds an image containing MATLAB image from the default MATLAB install location: `/MATLAB`.

`mosadm build-matlab-image matlabInstallLocation` builds a MATLAB image from the specified MATLAB installation location.

`mosadm build-matlab-image ___ option1 ... optionN` builds a MATLAB image using the specified configuration options and any of the previous syntaxes.

Note `mosadm build-matlab-image` interacts with Docker. If you do not have `sudo` administration privileges for Docker, then this command also requires `sudo`.

Examples

Build MATLAB Image from Specified Location

```
sudo ./mosadm build-matlab-image /Programs/MATLAB
```

```
Listing images
Checking the disk space...
Building MATLAB image
Determining the sizes of the folders in '/Programs/MATLAB'
Splitting the directories and files into layers
dockerRootDir: '/var/lib/docker'
...
Building layer 1/2, size: 2047 MB
Saving layer 1/2
...
Layer 1/2 built
Building layer 2/2, size: 1377 MB
Saving layer 2/2
...
Layer 2/2 built
Successfuly built the image containers.mathworks.com/matlab-online-server/mos-matlab-image:1.13.0

Completed build process
```

Input Arguments

`matlabInstallLocation` — MATLAB install location

`/MATLAB` (default) | string

MATLAB install location, specified as a string.

`option1 ... optionN` — One or more configuration options
strings

One or more configuration options, specified as strings corresponding to valid configuration options from these tables.

Specific to mosadm build-matlab-image

Option	Description
`--deployment-image-registry` *`registryName`*	Image registry to use for the deployment. Example: `registry.yourcompany.com`
`--deployment-image-repository` *`repositoryName`*	Image repository to use for the deployment images. Example: `mathworks`
`--extra-reserved-file-system-space` *`numGBs`*	Extra reserved file system space for image, in GBs. Specify `numGBs` as a numeric scalar. **Default**: `0.1`
`--image-flavor` *`imageFlavorName`*	Container image flavor, specified as `debian` or `rhel`. **Default**: `debian`

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm build-matlab-image` to the command line.
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet, -q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.

Option	Description
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.
`--skip-log-file`	Skip creating `install.log` and logging data to it.
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

Tips

To build this image, your machine must have a `TMPDIR` environment variable that specifies a writable temporary directory mounted in `exec` mode. To mount `TMPDIR` in `exec` mode, run this command:

```
mount -o remount,exec /tmp
```

If you are unable to make the `/tmp` folder writable or you cannot mount folders in `exec` mode, then they can set a custom temporary directory by running this code:

```
export TMPDIR="tempDirName"
```

Here, *`tempDirName`* is the name of a directory that is writable and mounted in `exec` mode.

Version History

Introduced in R2020a

See Also

`mosadm install-matlab`

Topics

“Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2
“Add Software Dependencies to MATLAB Image” on page 3-31

mosadm build-offline-installer

Build offline installer for MATLAB Online Server

Syntax

```
mosadm build-offline-installer
mosadm build-offline-installer option1 ... optionN
```

Description

`mosadm build-offline-installer` builds the offline installer for MATLAB Online Server.

`mosadm build-offline-installer option1 ... optionN` builds the offline installer using the specified configuration options.

Note `mosadm build-offline-installer` requires `sudo` administration privileges.

Examples

Build Offline Installer

```
sudo ./mosadm build-offline-installer
```

Input Arguments

`option1 ... optionN` — One or more configuration options

strings

One or more configuration options, specified as strings corresponding to valid configuration options from these tables.

Specific to mosadm build-offline-installer

Option	Description
`--create-zip`	Create a ZIP file of the artifacts generated during the offline installation.

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.

Option	Description
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm build-offline-installer` to the command line.
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet, -q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.
`--skip-log-file`	Skip creating `install.log` and logging data to it.
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

Version History

Introduced in R2020a

See Also

Topics

“Perform Offline MATLAB Online Server Installation” on page 2-12

mosadm copy-helm-charts

Copy Helm charts into install location for MATLAB Online Server

Syntax

```
mosadm copy-helm-charts
mosadm copy-helm-charts option1 ... optionN
```

Description

`mosadm copy-helm-charts` copies Helm charts into the install location for MATLAB Online Server. Helm is an open source project that helps deploy services into Kubernetes.

`mosadm copy-helm-charts option1 ... optionN` copies the Helm charts using the specified configuration options.

Note `mosadm copy-helm-charts` interacts with Docker. If you do not have `sudo` administration privileges for Docker, then this command also requires `sudo`.

Examples

Copy Helm Charts

```
sudo ./mosadm copy-helm-charts
```

Input Arguments

`option1 ... optionN` — One or more configuration options

strings

One or more configuration options, specified as strings corresponding to valid configuration options from these tables.

Specific to mosadm copy-helm-charts

Option	Description
`--image-keywords` *`keyword1 … keywordN`*	Keywords used to filter the images of a chart.

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.

Option	Description
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm copy-helm-charts` to the command line.
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet, -q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.
`--skip-log-file`	Skip creating `install.log` and logging data to it.
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

Version History

Introduced in R2020a

See Also

`mosadm merge-kube-config` | `mosadm generate-overrides`

Topics

“Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2

External Websites

Helm

mosadm deploy

Deploy MATLAB Online Server services to Kubernetes cluster

Syntax

```
mosadm deploy
mosadm deploy serviceName
mosadm deploy ___ option1 ... optionN
```

Description

`mosadm deploy` deploys all MATLAB Online Server services that have override charts, namely authnz, core-ui, gateway, license, matlab-pool, namespace, nginx-ingress, and resource into the Kubernetes cluster for the server.

`mosadm deploy serviceName` deploys only the specified service.

`mosadm deploy ___ option1 ... optionN` deploys services using the specified configuration options and any of the previous syntaxes.

Examples

Deploy All Services

```
./mosadm deploy
```

```
Cluster name matlab-online-server
Cluster namespace mathworks
Charts dir /.../matlab_online_server/charts
Overrides dir /.../matlab_online_server/overrides
Successfully installed release:  mathworks-authnz
Successfully installed release:  mathworks-core-ui
Successfully installed release:  mathworks-gateway
Successfully installed release:  mathworks-license
Successfully installed release:  mathworks-matlab-pool
Successfully installed release:  mathworks-namespace
Successfully installed release:  mathworks-resource
```

Deploy Specific Service

Deploy the MATLAB pool service.

```
./mosadm deploy matlab-pool
```

```
Cluster name matlab-online-server
Cluster namespace mathworks
Charts dir /.../matlab_online_server/charts
```

```
Overrides dir /.../matlab_online_server/overrides
Successfully installed release:  mathworks-matlab-pool
```

Input Arguments

`serviceName` — Name of service

`authnz` | `core-ui` | `gateway` | `license` | `matlab-pool` | `namespace` | `nginx-ingress` | `resource`

Name of service, specified as one of the options in the table.

Name	Description
`authnz`	Authenticates and authorizes actions of MATLAB Online Server users and some MATLAB Online Server components
`core-ui`	Configures the login screen and other UI elements.
`gateway`	Maps MATLAB Online Server clients to their assigned MATLAB instances.
`license`	Communicates with MathWorks License Manager to check for MATLAB Online Server licenses.
`matlab-pool`	Configures settings for MATLAB instances running in MATLAB Online Server.
`namespace`	Configures the namespace used for a MATLAB Online Server deployment
`nginx-ingress`	Configure the NGINX Ingress controller, which acts as a load balancer for MATLAB Online Server.
`resource`	Maintains information on all MATLAB instances.

`option1 ... optionN` — One or more configuration options

strings

One or more configuration options, specified as strings corresponding to valid configuration options from these tables.

Specific to mosadm deploy

Option	Description
`--image-keywords` *`keyword1 … keywordN`*	Keywords used to filter the images of a chart.

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.

Option	Description
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm deploy` to the command line.
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet, -q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.
`--skip-log-file`	Skip creating `install.log` and logging data to it.
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

Version History

Introduced in R2020a

See Also

`mosadm upgrade` | `mosadm undeploy`

Topics

“Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2
“Configure MATLAB in MATLAB Online Server” on page 3-17

mosadm gather-system-info

Gather system information for MATLAB Online Server

Syntax

```
mosadm gather-system-info
mosadm gather-system-info option1 ... optionN
```

Description

`mosadm gather-system-info` gathers system information for MATLAB Online Server, including UNIX system information, the present working directory, and disk usage. The command outputs this information to the terminal and logs it to the `install.log` file.

`mosadm gather-system-info option1 ... optionN` gathers system information using the specified configuration options.

Note `mosadm gather-system-info` interacts with Docker. If you do not have `sudo` administration privileges for Docker, then this command also requires `sudo`.

Examples

Gather System Information

```
sudo ./mosadm gather-system-info
```

Input Arguments

`option1 ... optionN` — One or more configuration options

strings

One or more configuration options, specified as strings corresponding to valid configuration options from this table.

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.

Option	Description
`--help, -h, help`	Output help for `mosadm gather-system-info` to the command line.
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet, -q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.
`--skip-log-file`	Skip creating `install.log` and logging data to it.
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

Version History

Introduced in R2020a

See Also

`mosadm run-smoke-tests`

Topics

“Contact Technical Support About MATLAB Online Server Issues” on page 5-2
“Manage Server Logs” on page 5-3

mosadm generate-overrides

Generate override files for configuring MATLAB Online Server

Syntax

```
mosadm generate-overrides
mosadm generate-overrides option1 ... optionN
```

Description

`mosadm generate-overrides` generates Helm override files based on `install.config`.

`mosadm generate-overrides option1 ... optionN` generates Helm override files using the specified configuration options.

Note `mosadm generate-overrides` interacts with Docker. If you do not have `sudo` administration privileges for Docker, then this command also requires `sudo`.

Examples

Generate Helm Overrides

```
sudo ./mosadm generate-overrides
```

```
Generating file /.../matlab_online_server/overrides/matlab-online-server/mathworks/all.yaml
Generating file /.../matlab_online_server/overrides/matlab-online-server/mathworks/authnz.yaml
Generating file /.../matlab_online_server/overrides/matlab-online-server/mathworks/core-ui.yaml
Generating file /.../matlab_online_server/overrides/matlab-online-server/mathworks/gateway.yaml
Generating file /.../matlab_online_server/overrides/matlab-online-server/mathworks/license.yaml
Generating file /.../matlab_online_server/overrides/matlab-online-server/mathworks/matlab-pool.yaml
Generating file /.../matlab_online_server/overrides/matlab-online-server/mathworks/resource.yaml
Generating file /.../matlab_online_server/overrides/matlab-online-server/mathworks/namespace.yaml
Generated overrides
```

If you receive an error that MATLAB install or MATLAB Docker image is not found, receive an error, set up the MATLAB install or MATLAB Docker image, and then try the command again.

Input Arguments

`option1 ... optionN` — One or more configuration options

strings

One or more configuration options, specified as strings corresponding to valid configuration options from these tables.

Specific to mosadm generate-overrides

Option	Description
`--deployment-image-registry` *`registryName`*	Image registry to use for the deployment. Example: `registry.yourcompany.com`
`--deployment-image-repository` *`repositoryName`*	Image repository to use for the deployment images. Example: `mathworks`
`--image-flavor` *`imageFlavorName`*	Container image flavor, specified as `debian` or `rhel`. **Default**: `debian`
`--image-keywords` *`keyword1 … keywordN`*	Keywords used to filter the images of a chart.
`--skip-backup`	Skip creating a backup of the existing overrides while generating the new overrides.
`--skip-matlab-image`	Skip checking whether the MATLAB image is already built.

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm generate-overrides` to the command line.
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet, -q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.

Option	Description
`--skip-log-file`	Skip creating `install.log` and logging data to it.
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

Version History

Introduced in R2020a

See Also

`mosadm merge-kube-config` | `mosadm copy-helm-charts`

Topics

“Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2

mosadm help

Display help for MATLAB Online Server system administration commands

Syntax

```
mosadm help
```

Description

`mosadm help` displays details on the operations that the `mosadm` command can perform and its configuration options. It also displays the commands to run when installing MATLAB Online Server.

Alternatively, you can display help by using the commands `mosadm -h` and `mosadm --help`.

Examples

Display Help

Display the help for the `mosadm` system administration commands.

```
./mosadm help
```

```
mosadm installs and manages the MATLAB Online Server installation.

The basic process is as follows:
...

Usage: mosadm
...
```

Version History

Introduced in R2020a

See Also

`mosadm gather-system-info`

Topics

“Host MATLAB Online on Your Infrastructure” on page 1-3

mosadm install-ingress

Install NGINX ingress controller for MATLAB Online Server

Syntax

```
mosadm install-ingress
mosadm install-ingress option1 ... optionN
```

Description

`mosadm install-ingress` installs the NGINX ingress controller for processing incoming requests to the Kubernetes cluster. This controller acts as a load balancer for MATLAB Online Server.

`mosadm install-ingress option1 ... optionN` installs the controller using the specified configuration options.

Examples

Install Controller

```
./mosadm install-ingress
```

Input Arguments

`option1 ... optionN` — One or more configuration options
strings

One or more configuration options, specified as strings corresponding to valid configuration options from this table.

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm install-ingress` to the command line.
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.

Option	Description
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet`, `-q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.
`--skip-log-file`	Skip creating `install.log` and logging data to it.
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

Version History

Introduced in R2020a

See Also

`mosadm uninstall-ingress`

Topics

“Configure NGINX Ingress Controller” on page 3-61
“Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2

External Websites

https://en.wikipedia.org/wiki/Nginx

mosadm install-matlab

Install MATLAB on MATLAB Online Server

Syntax

```
mosadm install-matlab
mosadm install-matlab option1 ... optionN
```

Description

`mosadm install-matlab` installs the latest version of MATLAB for use in MATLAB Online Server. By default, this command installs MATLAB to the `/MATLAB` folder of your local machine.

`mosadm install-matlab option1 ... optionN` installs MATLAB using the specified configuration options. For example, you can specify the folder in which to install MATLAB, the MATLAB release you want to install, and add-on products you want to include in your installation.

Examples

Install Latest Version of MATLAB

Install the latest version of MATLAB into the default `/MATLAB` folder on your machine. If this folder does not exist, the `mosadm install-matlab` command creates it.

```
./mosadm install-matlab
```

Install MATLAB with Specified Options

Install MATLAB R2023a and Database Toolbox to the `/usr/local/MATLAB` folder of your local machine.

```
./mosadm install-matlab --matlab-root /usr/local/MATLAB --matlab-version R2023a --products "MATLAB Database_Toolbox"
```

Input Arguments

`option1 ... optionN` — One or more configuration options
strings

One or more configuration options, specified as strings corresponding to valid configuration options from these tables.

Specific to mosadm install-matlab

Option	Description
`--matlab-root` *`filePath`*	Destination path in which install MATLAB. Specify *`filePath`* as a valid path on the machine. **Default**: `/MATLAB`
`--matlab-version` *`releaseVersion`*	MATLAB release to install. Example: `R2023a`
`--products "`*`product1 ... productN`*`"`	List of products to install, specified as a space-separated list. When specifying multiple products, replace spaces within product names with underscores and enclose the product list in quotes. Example: `"MATLAB Database_Toolbox Parallel_Computing_Toolbox"` **Default**: `MATLAB`

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm install-matlab` to the command line.
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet`, `-q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.
`--skip-log-file`	Skip creating `install.log` and logging data to it.

Option	Description
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

Version History

Introduced in R2023a

See Also

`mosadm build-matlab-image`

mosadm list-docker-images

List Docker images used by MATLAB Online Server

Syntax

```
mosadm list-docker-images
mosadm list-docker-images option1 ... optionN
```

Description

`mosadm list-docker-images` lists all the Docker images in the local Docker cache of your MATLAB Online Server installation.

`mosadm list-docker-images option1 ... optionN` lists Docker images using the specified configuration options.

Note `mosadm list-docker-images` interacts with Docker. If you do not have `sudo` administration privileges for Docker, then this command also requires `sudo`.

Examples

List Docker Images

```
sudo ./mosadm list-docker-images
```

```
Listing images
containers.mathworks.com/matlab-online-server/mos-matlab-image:1.13.0
containers.mathworks.com/matlab-online-server/mos-gateway-image:1.8.0
containers.mathworks.com/matlab-online-server/mos-resource-image:1.13.0
containers.mathworks.com/matlab-online-server/mos-resource-proxy-image:1.13.0-bullseye
containers.mathworks.com/matlab-online-server/mos-matlab-omnibus-image:1.13.0-bionic
containers.mathworks.com/matlab-online-server/mos-authnz-image:1.8.0-stretch
containers.mathworks.com/matlab-online-server/mos-core-ui-image:1.6.3-stretch
containers.mathworks.com/matlab-online-server/mos-hamonitor-image:1.2.0
containers.mathworks.com/matlab-online-server/mos-license-image:20221.1.1-stretch
```

Input Arguments

`option1 ... optionN` — One or more configuration options

strings

One or more configuration options, specified as strings corresponding to valid configuration options from these tables.

Specific to mosadm list-docker-images

Option	Description
`--deployment-image-registry` *`registryName`*	Image registry to use for the deployment. Example: `registry.yourcompany.com`

Option	Description
`--deployment-image-repository` *`repositoryName`*	Image repository to use for the deployment images. Example: `mathworks`
`--image-flavor` *`imageFlavorName`*	Container image flavor, specified as `debian` or `rhel`. **Default**: `debian`

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm list-docker-images` to the command line.
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet`, `-q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.
`--skip-log-file`	Skip creating `install.log` and logging data to it.
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

Version History

Introduced in R2020a

See Also

`mosadm load-docker-images` | `mosadm push-docker-images`

Topics

“Add Software Dependencies to MATLAB Image” on page 3-31

External Websites

Docker

mosadm load-docker-images

Load Docker images into MATLAB Online Server

Syntax

```
mosadm load-docker-images
mosadm load-docker-images option1 ... optionN
```

Description

`mosadm load-docker-images` loads Docker images from the installer into the local Docker cache of the MATLAB Online Server installation.

`mosadm load-docker-images option1 ... optionN` loads Docker images using the specified configuration options.

Note `mosadm load-docker-images` interacts with Docker. If you do not have `sudo` administration privileges for Docker, then this command also requires `sudo`.

Examples

Load Docker Image

```
sudo ./ mosadm load-docker-images
```

```
mos-authnz:1.8.0@default chart images:
Pulling containers.mathworks.com/matlab-online-server/mos-authnz-image:1.8.0-stretch... Done
mos-core-ui:1.9.0@default chart images:
Pulling containers.mathworks.com/matlab-online-server/mos-core-ui-image:1.6.3-stretch... Done
mos-gateway:1.8.0@default chart images:
Pulling containers.mathworks.com/matlab-online-server/mos-gateway-image:1.8.0... Done
mos-hamonitor:1.4.0@default chart images:
Pulling containers.mathworks.com/matlab-online-server/mos-hamonitor-image:1.2.0... Done
mos-matlab-pool:1.13.0@default chart images:
Pulling containers.mathworks.com/matlab-online-server/mos-matlab-omnibus-image:1.13.0-bionic... Done
Pulling containers.mathworks.com/matlab-online-server/mos-resource-proxy-image:1.13.0-bullseye... Done
mos-namespace:1.4.0@default chart images:
mos-resource:1.13.0@default chart images:
Pulling containers.mathworks.com/matlab-online-server/mos-resource-image:1.13.0... Done
mos-license:1.4.0@default chart images:
Pulling containers.mathworks.com/matlab-online-server/mos-license-image:20221.1.1-stretch... Done
```

Input Arguments

`option1 ... optionN` — One or more configuration options

strings

One or more configuration options, specified as strings corresponding to valid configuration options from these tables.

Specific to mosadm load-docker-images

Option	Description
`--deployment-image-registry` *registryName*	Image registry to use for the deployment. Example: `registry.yourcompany.com`
`--deployment-image-repository` *repositoryName*	Image repository to use for the deployment images. Example: `mathworks`
`--do-patch-glibc`	Enable building of custom GNU C library (glibc) patch.
`--image-flavor` *imageFlavorName*	Container image flavor, specified as `debian` or `rhel`. **Default**: `debian`

Common to All mosadm Commands

Option	Description
`--charts-dir` *chartsDir*	Name of MATLAB Online Server charts folder to use.
`--cluster` *clusterName*	Name of Kubernetes cluster to use.
`--data-dir` *dataDir*	Name of MATLAB Online Server `data` folder to use.
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm load-docker-images` to the command line.
`--install` *serviceName*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *configFilePath*	Path to the Kubernetes cluster configuration file.
`--kube-context` *contextName*	Kubernetes cluster context to use.
`--manifest-file` *manifestFileName*	Name of the MATLAB Online Server manifest file.
`--mos-root` *rootDir*	MATLAB Online Server root folder.
`--namespace` *namespaceName*	Name of Kubernetes namespace to use.
`--overrides-dir` *overridesDir*	Name of MATLAB Online Server `overrides` folder.
`--quiet`, `-q`	Output only data.
`--set-file` *key1=path1,...,keyN=pathN*	Set values from respective files specified via the command line.
`--set-string` *key1=val1,...,keyN=valN*	Set string values on the command line.
`--skip-create-namespace` *namespaceName*	Skip checking whether the specified namespace exists.
`--skip-log-file`	Skip creating `install.log` and logging data to it.
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.

Option	Description
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

Version History

Introduced in R2020a

See Also

`mosadm list-docker-images` | `mosadm push-docker-images`

Topics

“Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2

External Websites

Docker

mosadm merge-kube-config

Merge Kubernetes configuration files into MATLAB Online Server installation

Syntax

```
mosadm merge-kube-config
mosadm merge-kube-config option1 ... optionN
```

Description

`mosadm merge-kube-config` merges the cluster Kubernetes configuration located at `/etc/kubernetes/admin.conf` into the MATLAB Online Server Kubernetes configuration located at `~/.kube/config`. The merged configuration makes it easier to run the `kubectl` command-line tool for managing the Kubernetes cluster that MATLAB Online Server runs in.

`mosadm merge-kube-config option1 ... optionN` merges Kubernetes configuration files using the specified configuration options.

Note `mosadm merge-kube-config` requires `sudo` administration privileges.

Examples

Merge Kubernetes Configuration Files

```
sudo ./mosadm merge-kube-config
```

```
Using kubeconfig from /etc/kubernetes/admin.conf
Will merge the current config from ~/.kube/config with the one in
/etc/kubernetes/admin.conf
mkdir -p /home/ubuntu/.kube
cp /etc/kubernetes/admin.conf /home/ubuntu/.kube/config
chown -R ubuntu:ubuntu /home/ubuntu/.kube
```

Input Arguments

`option1 ... optionN` — One or more configuration options

strings

One or more configuration options, specified as strings corresponding to valid configuration options from this table.

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.

Option	Description
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm merge-kube-config` to the command line.
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet, -q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.
`--skip-log-file`	Skip creating `install.log` and logging data to it.
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

Version History

Introduced in R2020a

See Also

`mosadm copy-helm-charts` | `mosadm generate-overrides`

Topics

“Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2

External Websites

Kubernetes

mosadm push-docker-images

Push Docker images loaded from MATLAB Online Server to remote registry

Syntax

```
mosadm push-docker-images
mosadm push-docker-images option1 ... optionN
```

Description

`mosadm push-docker-images` retags and pushes Docker images loaded from the MATLAB Online Server local Docker cache to the configured remote registry. The `mosadm` command uses the Docker registry and Docker repository information from the `install.config` file.

`mosadm push-docker-images option1 ... optionN` pushes Docker images using the specified configuration options.

Note `mosadm push-docker-images` interacts with Docker. If you do not have `sudo` administration privileges for Docker, then this command also requires `sudo`.

Examples

Push Docker Images

Update the following lines in the `install.config` file with your Docker registry and repository information.

```
DOCKER_REGISTRY=<docker-registry-here>
DOCKER_REPOSITORY=<repo-here>
```

Push the Docker images.

```
mosadm push-docker-images
```

Alternatively, you can pass the command as a configuration option as follows:

```
mosadm push-docker-images --deployment-image-registry registryName --deployment-image-repository repositoryName
```

Input Arguments

`option1 ... optionN` — One or more configuration options

strings

One or more configuration options, specified as strings corresponding to valid configuration options from these tables.

Specific to mosadm push-docker-images

Option	Description
`--deployment-image-registry` *`registryName`*	Image registry to use for the deployment. Example: `registry.yourcompany.com`
`--deployment-image-repository` *`repositoryName`*	Image repository to use for the deployment images. Example: `mathworks`
`--image-flavor` *`imageFlavorName`*	Container image flavor, specified as `debian` or `rhel`. **Default**: `debian`

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm push-docker-images` to the command line.
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet, -q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.
`--skip-log-file`	Skip creating `install.log` and logging data to it.
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.

Option	Description
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

Version History

Introduced in R2020a

See Also

`mosadm load-docker-images` | `mosadm list-docker-images`

Topics

“Install MATLAB Online Server on Cloud-Managed Kubernetes” on page 2-16
“Install MATLAB Online Server on Red Hat OpenShift” on page 2-26
“Add Software Dependencies to MATLAB Image” on page 3-31

External Websites

Docker

mosadm renew-node

Renew single-node Kubernetes certificates for MATLAB Online Server

Syntax

```
mosadm renew-node
mosadm renew-node option1 ... optionN
```

Description

`mosadm renew-node` renews the Kubernetes internal cluster certificates for one year from the time of executing the command. After renewing the certificate, the `renew-node` command performs the following actions:

Use `mosadm` to

1. Restarts the Kubernetes static and default pods.
2. Removes the existing `~/.kube` folder.
3. Runs the `./mosadm merge-kube-config` command to rebuild the `~/.kube` folder and merge the cluster Kubernetes configuration (`/etc/kubernetes/admin.conf`) into the user Kubernetes configuration (`~/.kube/config`).

You can use `renew-node` on single-node clusters only. `sudo` privileges are required.

`mosadm renew-node option1 ... optionN` renews the node using the specified configuration options.

Note `mosadm renew-node` requires `sudo` administration privileges.

Examples

Renew Node

```
sudo ./mosadm renew-node
```

```
Removing static pod manifests from /etc/kubernetes/manifests
Waiting for 30s
Copying static pod manifests to /etc/kubernetes/manifests
Waiting for 30s
Removing existing ~/.kube/config
Proceed? Y/N Y
Copying kube config to  /.../.kube/config
```

Input Arguments

`option1 ... optionN` — One or more configuration options
strings

One or more configuration options, specified as strings corresponding to valid configuration options from this table.

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm renew-node` to the command line.
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet, -q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.
`--skip-log-file`	Skip creating `install.log` and logging data to it.
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

Version History

Introduced in R2023a

See Also

`mosadm bootstrap-node` | `mosadm reset-node`

mosadm reset-node

Reset single-node Kubernetes cluster running MATLAB Online Server

Syntax

```
mosadm reset-node
mosadm reset-node option1 ... optionN
```

Description

`mosadm reset-node` resets the single-node Kubernetes cluster that is running MATLAB Online Server and performs these actions:

- Stop the `kubelet` program running on the machine.
- Remove the Kubernetes cluster running locally.
- Reset the IP tables.

Run this command for a single-node cluster that was created using the `mosadm bootstrap-node` command.

After you run this command, you can rerun `mosadm bootstrap-node` to redeploy a single-node Kubernetes cluster and continue working with MATLAB Online Server. The `reset-node` command does not uninstall Linux packages deployed during the `bootstrap-node` operation. To reset your environment fully, re-image the operating system or restore it from a backup.

`mosadm reset-node option1 ... optionN` resets the node using the specified configuration options.

Note `mosadm reset-node` requires `sudo` administration privileges.

Examples

Reset Node

```
sudo ./mosadm reset-node
```

```
[reset] Reading configuration from the cluster...
[reset] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
W0125 18:44:12.253437  105006 configset.go:77]
  Warning: No kubeproxy.config.k8s.io/v1alpha1 config is loaded.
  Continuing without it: configmaps "kube-proxy" not found
[reset] WARNING: Changes made to this host by 'kubeadm init' or 'kubeadm join' will be reverted.
[reset] Are you sure you want to proceed? [y/N]: y
[preflight] Running pre-flight checks
The 'update-cluster-status' phase is deprecated and will be removed in a future release.
  Currently it performs no operation
[reset] Stopping the kubelet service
[reset] Unmounting mounted directories in "/var/lib/kubelet"
[reset] Deleting contents of config directories: [/etc/kubernetes/manifests /etc/kubernetes/pki]
[reset] Deleting files: [/etc/kubernetes/admin.conf
  /etc/kubernetes/kubelet.conf /etc/kubernetes/bootstrap-kubelet.conf
```

```
  /etc/kubernetes/controller-manager.conf /etc/kubernetes/scheduler.conf]
[reset] Deleting contents of stateful directories: [/var/lib/etcd /var/lib/kubelet
  /var/lib/dockershim /var/run/kubernetes /var/lib/cni]

The reset process does not clean CNI configuration. To do so, you must remove /etc/cni/net.d

The reset process does not reset or clean up iptables rules or IPVS tables.
If you wish to reset iptables, you must do so manually by using the "iptables" command.

If your cluster was setup to utilize IPVS, run ipvsadm --clear (or similar)
to reset your system's IPVS tables.

The reset process does not clean your kubeconfig files and you must remove them manually.
Please, check the contents of the $HOME/.kube/config file.
```

Input Arguments

`option1 ... optionN` — One or more configuration options

strings

One or more configuration options, specified as strings corresponding to valid configuration options from this table.

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm reset-node` to the command line.
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet`, `-q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.

Option	Description
`--skip-log-file`	Skip creating `install.log` and logging data to it.
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

Version History

Introduced in R2020a

See Also

`mosadm bootstrap-node` | `mosadm renew-node`

mosadm run-smoke-tests

Run smoke tests on MATLAB Online Server

Syntax

```
mosadm run-smoke-tests
mosadm run-smoke-tests option1 ... optionN
```

Description

`mosadm run-smoke-tests` runs smoke tests on MATLAB Online Server.

Smoke tests reveal simple failures severe enough to impact operations. The smoke test tool that comes with MATLAB Online Server runs a set of tests against a MATLAB Online Server implementation. The tool returns information on installation setup, Kubernetes elements, and the MATLAB Online Server microservices. The information provided by these smoke tests can help focus troubleshooting strategies for MATLAB Online Server.

Note To run smoke tests, verify that your system meets the prerequisites specified in “Smoke Test Prerequisites” on page 7-48

For details on interpreting the results, see “Address Smoke Test Failures” on page 7-48

`mosadm run-smoke-tests option1 ... optionN` runs smoke tests using the specified configuration options.

Note `mosadm run-smoke-tests` interacts with Docker. If you do not have `sudo` administration privileges for Docker, then this command also requires `sudo`.

Examples

Run Default Smoke Tests

```
sudo ./mosadm run-smoke-tests
```

```
YYYY/MM/DD hh:mm:ss Smoke-Tests------------------------------------------------------------------------
YYYY/MM/DD hh:mm:ss    Commands----------------------------------------------------------------------
YYYY/MM/DD hh:mm:ss       docker : PASS
YYYY/MM/DD hh:mm:ss       kubectl : PASS
YYYY/MM/DD hh:mm:ss       helm : PASS
YYYY/MM/DD hh:mm:ss    Commands : PASS
YYYY/MM/DD hh:mm:ss    ------------------------------------------------------------------------------
YYYY/MM/DD hh:mm:ss    Properties--------------------------------------------------------------------
YYYY/MM/DD hh:mm:ss       DomainBaseLookup : PASS
YYYY/MM/DD hh:mm:ss       DomainBaseSecurelyReachable : PASS
YYYY/MM/DD hh:mm:ss    Properties : PASS
...
```

Run Smoke Tests with Verbose Output

```
sudo ./mosadm run-smoke-tests --verbose
```

```
YYYY/MM/DD hh:mm:ss Smoke-Tests----------------------------------------------------------------------
YYYY/MM/DD hh:mm:ss    Commands----------------------------------------------------------------------
YYYY/MM/DD hh:mm:ss       docker-----------------------------------------------------------------------
YYYY/MM/DD hh:mm:ss          docker version : Client:
                              Version:           20.10.12
                              API version:       1.41
                              Go version:        go1.17.3
                              Git commit:        20.10.12-0ubuntu4
                              Built:             Mon Mar  7 17:10:06 YYYY
                              OS/Arch:           linux/amd64
                              Context:           default
                              Experimental:      true
                              ...
```

Input Arguments

`option1 ... optionN` — One or more configuration options

strings

One or more configuration options, specified as strings corresponding to valid configuration options from these tables.

Specific to mosadm run-smoke-tests

Option	Description
`--container-log-tail-lines` *`numLines`*	Number of lines from the end of the container log to display. **Default**: `100`
`--deployment-image-registry` *`registryName`*	Image registry to use for the deployment. Example: `registry.yourcompany.com`
`--deployment-image-repository` *`repositoryName`*	Image repository to use for the deployment images. Example: `mathworks`
`--domain-base` *`domainBaseName`*	Domain where MATLAB Online Server is running.
`--image-flavor` *`imageFlavorName`*	Container image flavor, specified as `debian` or `rhel`. **Default**: `debian`
`--license-port` *`portNum`*	Port number used to connect to MathWorks License Manager. Specify *`portNum`* as a numeric string representing a valid port number. **Default**: `8080`
`--no-certificate-check`	If the certificate deployed at MATLAB Online Server does not match its domain, many tests fail. Specify this argument to ignore this certificate mismatching.

Option	Description
`--no-secure-http`	If no certificate is deployed at MATLAB Online Server, run tests that require HTTP calls using only nonsecure HTTP. Use this option to check whether MATLAB Online Server is deployed properly before deploying the appropriate certificate. If you specify this option, then `mosadm run-smoke-tests` ignores the `--no-certificate-check` option.
`--no-terminal-colors`	Skip using terminal colors.
`--skip-tests` *`testName1 ... testNameN`*	Names of tests to be skipped.
`--time-out` *`numSeconds`*	Number of seconds to wait to start running tests before canceling the operation. **Default**: 5

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm run-smoke-tests` to the command line.
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet`, `-q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.
`--skip-log-file`	Skip creating `install.log` and logging data to it.

Option	Description
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

More About

Smoke Test Prerequisites

Single Node

- Set and export your test username and password environment variables MOS_TEST_USERNAME and MOS_TEST_PASSWORD. Their default values are *admin* and *password* , respectively. The username and password should be valid credentials for signing in to your MATLAB Online Server session via the browser.
- If you have installed MATLAB Online Server using mosadm, mosadm applies the correct Kube-Config and properly sets the environment variable KUBECONFIG.

Managed Kubernetes (EKS, GKE, AKS, ...)

- Set and export your test username and password environment variables MOS_TEST_USERNAME and MOS_TEST_PASSWORD. Their default values are *admin* and *password*, respectively.
- Make sure that you can access your Kubernetes cluster via kubectl.
- Your ~/.kube/config should point to your cluster and namespace Kube-Config file, or your KUBECONFIG environment variable should be set to the right path of your Kube-Config file. Note that KUBECONFIG has priority over ~/.kube/config.

Address Smoke Test Failures

Before proceeding with troubleshooting failures shown by the smoke tests, make sure that the prerequisites stated in the MATLAB Online Server Administrator Guide are satisfied.

Examine the results of the smoke tests and determine if all tests passed or if there are failures that you must investigate.

If you get a **PASS** result on all smoke tests, then you have properly deployed MATLAB Online Server, and you are ready to access it from your browser.

If you get a **FAIL** for one or more sub smoke tests, then you must find the root cause of their failures before proceeding further.

The smoke tests are divided into several parts, where each part has a subset of tests that target a specific part of configuration or behavior of your local setup, Kubernetes configuration, or MATLAB Online Server setup.

If you get a FAIL on any sub smoke test, re-run the smoke tests with the argument –*verbose*. This argument gives you more details on the failures.

Commands Tests

The commands tests check if the commands `docker`, `kubectl`, and `helm` run properly from the machine on which you are running the `mosadm` command. If the `–verbose` option is enabled, you get more details on their versions.

If any of the subtests fails, you can manually check if the related command (`docker`, `kubect,l` or `helm`) is installed on your machine and properly working.

Properties Tests

The properties checks focus on tests related to your MATLAB Online Server `domain`.

- `DomainBaseLookup` checks if your domain is reachable securely via https.
- `DomainBaseSecurelyReachable` checks if your domain is resolvable to an IP address.
- `DomainBaseValidateCertificate` checks if your domain SSL certificate matches your domain name.

While the MATLAB Online Server installation instructions specify that you have a valid SSL certificate that matches your MATLAB Online Server domain name, you may not have it initially ready. However, you can still run the smoke tests.

If you don’t have a valid SSL certificate or you have one but it does not match your domain (for example, you may have a self-signed certificate), you can have the smoke test bypass the certificate requirement by passing the no-secure-http and no-certificate arguments to the smoke tests option:

```
mosadm run-smoke-tests -no-secure-http -no-certificate-check
```

Docker Tests

The Docker tests check properties related your local Docker environment. The test checks for the Docker Root Directory disk space (refer to “Hardware Requirements” on page 1-10 for details on disk space requirements).

If the available disk space under this directory disk partition is low, then that might be a root cause for some deployment failures. Consider increasing this disk space, or re-install or re-configure Docker to use a Root Directory under a partition with more disk space.

Kubernetes Tests

The Kubernetes tests run a set of tests related to your Kubernetes cluster.

- *GetKubernetesClient* checks if the test can communicate with your Kubernetes cluster. If this test fails, then make sure that your KUBECONFIG (or ~/.kube/config) is set properly and you can access it successfully from your installation machine using the kubectl command.
- *CheckKubernetesNamespace* verifies that your configured namespace, which defaults to `mathworks`, exists under your Kubernetes cluster. If the configured namespace is not present under your Kubernetes cluster, then your MATLAB Online Server won’t be deployed properly, or will be deployed but under the wrong namespace.
- *GetPodsList* displays all pods under your configured namespace. This is mainly an informative test that displays all existing pods under your Kubernetes namespace.
- *MOS-Core-ServicesPodsAreReady* lists only the MATLAB Online Server core service pods. If a pod is running properly, the test shows the status READY next to the service. Otherwise, the test shows which underlying container is failing, along with any error messages that apply.

MATLAB Online Server

MATLAB Online Server runs targeted tests against each MATLAB Online core service.

- *AuthNZ* checks if the AuthNZ service can authenticate users via their usernames and passwords. Under “Smoke Test Prerequisites” on page 7-48, you set two environment variables: MOS_TEST_USERNAME and MOS_TEST_PASSWORD. These variables store the valid username and password, respectively, for accessing your MATLAB Online Server from the browser. These variables default to *admin* and *password*.

  If this AuthNZ test fails, even if those environment variables are set properly, then check if the License test is failing. If this is the case, first fix the License service and then re-run the smoke tests. If the License test passes and the AuthNZ test is still failing, then check the AuthNZ service logs for further troubleshooting.
- *CoreUI*, *Gateway*, *MATLABPoolUI* and *Resource* run a set of internal MATLAB Online Server tests. Like *AuthNZ*, if any of these services tests fails, first check if the License service is passing its test before troubleshooting any of them.
- *License* checks if the MATLAB Online Server License service is responding to requests. If this test shows the status PASS but other MATLAB Online Server services are still failing their tests, check if your network license manager server is accessible, via its ports, from your Kubernetes pods. The netcat (nc) command can be helpful in checking if a server is accessible via specific port.

Also check if your license file (license.dat) contains your MATLAB licenses as well as your MATLAB Online Server license.

- *Help* tests if the Help content is accessible by MATLAB Online Server users. Even if these tests are failing, it does not have any impact on the remaining MATLAB Online Server services.

If you get many MATLAB Online Server services smoke tests failing, start by troubleshooting the License service and then proceed further with the remaining services.

As a general best practice, after fixing each single failed sub smoke test, re-run all smoke tests. Some services depend on others, and fixing one of them, like the License, can in many cases fix other failures.

If you continue to see errors in some sub smoke tests of some MATLAB Online Server services, check their logs (see “Configure Log Shipping” on page 5-4), as these files often provide useful information on the root causes of such failures.

If you are still experiencing failures after applying the troubleshooting suggestions, perform the following:

1. Save the *logs* folder, generated by the smoke tests, as a zip file.
2. Run the command `sudo ./mosadm gather-system-info`. This command generates the file `install.log`.
3. Contact MathWorks Support.

Version History

Introduced in R2020a

See Also

`mosadm gather-system-info`

Topics

“Contact Technical Support About MATLAB Online Server Issues” on page 5-2
“Manage Server Logs” on page 5-3

mosadm undeploy

Remove MATLAB Online Server services from Kubernetes cluster

Syntax

```
mosadm undeploy
mosadm undeploy serviceName
mosadm undeploy ___ option1 ... optionN
```

Description

`mosadm undeploy` removes all services that have override charts, namely authnz, core-ui, gateway, license, matlab-pool, namespace, nginx-ingress, and resource.

`mosadm undeploy serviceName` removes only the specified service.

`mosadm undeploy ___ option1 ... optionN` undeploys services using the specified configuration options and any of the previous syntaxes.

Examples

Undeploy All Services

```
./mosadm undeploy
```

Undeploy Specific Service

Undeploy the MATLAB pool service.

```
./mosadm undeploy matlab-pool
```

Input Arguments

`serviceName` — Name of service
`authnz` | `core-ui` | `gateway` | `license` | `matlab-pool` | `namespace` | `nginx-ingress` | `resource`

Name of service, specified as one of the options in the table.

Name	Description
`authnz`	Authenticates and authorizes actions of MATLAB Online Server users and some MATLAB Online Server components

Name	Description
`core-ui`	Configures the login screen and other UI elements.
`gateway`	Maps MATLAB Online Server clients to their assigned MATLAB instances.
`license`	Communicates with MathWorks License Manager to check for MATLAB Online Server licenses.
`matlab-pool`	Configures settings for MATLAB instances running in MATLAB Online Server.
`namespace`	Configures the namespace used for a MATLAB Online Server deployment
`nginx-ingress`	Configure the NGINX Ingress controller, which acts as a load balancer for MATLAB Online Server.
`resource`	Maintains information on all MATLAB instances.

`option1 ... optionN` — One or more configuration options
strings

One or more configuration options, specified as strings corresponding to valid configuration options from these tables.

Specific to mosadm undeploy

Option	Description
`--image-keywords` *`keyword1 … keywordN`*	Keywords used to filter the images of a chart.

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm undeploy` to the command line.
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.

Option	Description
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet`, `-q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.
`--skip-log-file`	Skip creating `install.log` and logging data to it.
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

Version History

Introduced in R2020a

See Also

`mosadm upgrade` | `mosadm deploy`

Topics

“Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2
“Configure MATLAB in MATLAB Online Server” on page 3-17

mosadm uninstall-ingress

Uninstall NGINX ingress controller for MATLAB Online Server

Syntax

```
mosadm uninstall-ingress
mosadm uninstall-ingress option1 ... optionN
```

Description

`mosadm uninstall-ingress` uninstalls the NGINX ingress controller for processing incoming requests to the Kubernetes cluster. You can use this command only if you installed the controller using `mosadm install-ingress`.

`mosadm uninstall-ingress option1 ... optionN` uninstalls the controller using the specified configuration options.

Examples

Uninstall Controller

```
./mosadm uninstall-ingress
```

Input Arguments

`option1 ... optionN` — One or more configuration options
strings

One or more configuration options, specified as strings corresponding to valid configuration options from this table.

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm uninstall-ingress` to the command line.
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.

Option	Description
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet`, `-q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.
`--skip-log-file`	Skip creating `install.log` and logging data to it.
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

Version History

Introduced in R2023a

See Also

`mosadm install-ingress`

Topics

“Uninstall MATLAB Online Server” on page 2-42

mosadm upgrade

Upgrade MATLAB Online Server services in Kubernetes cluster

Syntax

```
mosadm upgrade
mosadm upgrade serviceName
mosadm upgrade ___ option1 ... optionN
```

Description

`mosadm upgrade` upgrades all services currently deployed in the Kubernetes cluster for MATLAB Online Server that have YAML override files. This command upgrades services only if they have configuration changes since their last deployment.

`mosadm upgrade serviceName` upgrades only the specified service.

`mosadm upgrade ___ option1 ... optionN` upgrades services using the specified configuration options and any of the previous syntaxes.

Examples

Upgrade All Services

If you are updating a property that is applicable to all services:

1. In the all-services override file (`all.yaml`), update the desired property.
2. Run: `mosadm upgrade`

```
./mosadm upgrade
```

Upgrade Specific Service

Update the number of available MATLAB instances in the prewarmed MATLAB pool.

First, in the MATLAB Pool override file (`matlab-pool.yaml`), change the `replicaCount` attribute value to the desired number, for example: `replicaCount: 4`

Then, upgrade the MATLAB pool service.

```
./mosadm upgrade matlab-pool
```

Input Arguments

`serviceName` — Name of service
`authnz` | `core-ui` | `gateway` | `license` | `matlab-pool` | `namespace` | `nginx-ingress` | `resource`

Name of service, specified as one of the options in the table.

Name	Description
`authnz`	Authenticates and authorizes actions of MATLAB Online Server users and some MATLAB Online Server components
`core-ui`	Configures the login screen and other UI elements.
`gateway`	Maps MATLAB Online Server clients to their assigned MATLAB instances.
`license`	Communicates with MathWorks License Manager to check for MATLAB Online Server licenses.
`matlab-pool`	Configures settings for MATLAB instances running in MATLAB Online Server.
`namespace`	Configures the namespace used for a MATLAB Online Server deployment
`nginx-ingress`	Configure the NGINX Ingress controller, which acts as a load balancer for MATLAB Online Server.
`resource`	Maintains information on all MATLAB instances.

`option1 ... optionN` — One or more configuration options
strings

One or more configuration options, specified as strings corresponding to valid configuration options from these tables.

Specific to mosadm upgrade

Option	Description
`--image-keywords` *`keyword1 … keywordN`*	Keywords used to filter the images of a chart.

Common to All mosadm Commands

Option	Description
`--charts-dir` *`chartsDir`*	Name of MATLAB Online Server charts folder to use.
`--cluster` *`clusterName`*	Name of Kubernetes cluster to use.
`--data-dir` *`dataDir`*	Name of MATLAB Online Server `data` folder to use.
`--dry-run`	Perform a "dry run" of the operation, printing out the commands that would have run without this option specified.
`--help, -h, help`	Output help for `mosadm upgrade` to the command line.

Option	Description
`--install` *`serviceName`*	If a service by this name is not already deployed, deploy the service during upgrade.
`--kube-config` *`configFilePath`*	Path to the Kubernetes cluster configuration file.
`--kube-context` *`contextName`*	Kubernetes cluster context to use.
`--manifest-file` *`manifestFileName`*	Name of the MATLAB Online Server manifest file.
`--mos-root` *`rootDir`*	MATLAB Online Server root folder.
`--namespace` *`namespaceName`*	Name of Kubernetes namespace to use.
`--overrides-dir` *`overridesDir`*	Name of MATLAB Online Server `overrides` folder.
`--quiet`, `-q`	Output only data.
`--set-file` *`key1=path1,...,keyN=pathN`*	Set values from respective files specified via the command line.
`--set-string` *`key1=val1,...,keyN=valN`*	Set string values on the command line.
`--skip-create-namespace` *`namespaceName`*	Skip checking whether the specified namespace exists.
`--skip-log-file`	Skip creating `install.log` and logging data to it.
`--skip-prompt, -y`	Force acceptance of the terms of use and do not prompt.
`--tmp-dir` *`tempDirName`*	Name of MATLAB Online Server temporary folder to use.
`--values` *`key1=val1,...,keyN=valN`*	Specify one of more values in a YAML file or URL.
`--verbose`	Output extra details to the command line.

Version History

Introduced in R2020a

See Also

`mosadm deploy` | `mosadm undeploy`

Topics

“Perform Minimal MATLAB Online Server Installation on Single Machine” on page 2-2
“Configure MATLAB in MATLAB Online Server” on page 3-17