

# ibi<sup>TM</sup> WebFOCUS<sup>®</sup> DSML Services - Container Edition

## Installation and Deployment Guide

Version 9.3.5 | July 2025

# Contents

---

<b>Contents</b>	<b>2</b>
<b>Introduction</b>	<b>4</b>
<b>Requirements and Prerequisites</b>	<b>6</b>
<b>Key Components/Systems</b>	<b>10</b>
<b>Creating or Using Docker Images</b>	<b>12</b>
Building and Loading WebFOCUS DSML Services - Container Edition images using Podman	13
<b>Installing ibi WebFOCUS DSML Services - Container Edition</b>	<b>15</b>
Building the Docker Images	17
Deploying ibi WebFOCUS DSML Services - Container Edition on Kubernetes Cluster	19
Deploying ibi WebFOCUS DSML Services - Container Edition Using Docker Compose	21
Deploying ibi WebFOCUS DSML Services - Container Edition with Red Hat OpenShift Cluster	22
Deploying ibi WebFOCUS DSML Services- Container Edition with Podman and Podman Compose On RHEL 9	28
Installing Podman on RHEL 9	29
Troubleshooting Podman Installation	30
Installing Podman Compose	31
Deploying ibi WebFOCUS DSML Services - Container Edition Using Podman Compose	32
<b>Testing and Verifying</b>	<b>34</b>
<b>Upgrading ibi WebFOCUS DSML Services - Container Edition</b>	<b>37</b>

<b>Uninstalling ibi WebFOCUS DSML Services - Container Edition .....</b>	<b>38</b>
<b>Usage and Environment Considerations .....</b>	<b>39</b>
<b>Resource (CPU and Memory) Parameters .....</b>	<b>40</b>
<b>Third-Party Licenses for the Container Image .....</b>	<b>43</b>
<b>ibi Documentation and Support Services .....</b>	<b>47</b>
<b>Legal and Third-Party Notices .....</b>	<b>48</b>

# Introduction

---

ibi™ WebFOCUS® DSML Services - Container Edition provides a scalable, microservices-based analytical platform designed for container-based deployments. This new offering packages WebFOCUS® DSML Services - Container Edition software components in a container image, including all required Operating System (OS) components and libraries, which can be deployed and run in a Kubernetes cluster.

The following are several benefits and advantages you can use in WebFOCUS DSML Services - Container Edition:

- **Automatic deployments:** Identify which (and how many) WebFOCUS components (for example, WebFOCUS Reporting Server) need to be deployed and where (for example, specific clusters).
- **Reduce operational and infrastructure costs:** Add or remove services and resources to the Kubernetes cluster as needed. Data sources can exist in the cloud (private, public), or on-premises.
- **Increase adoption and usage:** Develop specific BI applications for targeted users on-premises and deploy to the cloud as needed. When you install WebFOCUS on Kubernetes, development and testing can occur within internal clusters or external clusters.
- **Portability:** Fully portable between cloud and on-premises environments.
- **Dynamic scaling:** Rapidly adjust to increased user and data demands.

Existing Kubernetes services that are provided by cloud providers can also be used (for example, Amazon® Elastic Kubernetes Service (EKS)).

Deploying WebFOCUS DSML Services - Container Edition on Kubernetes provides a variety of deployment options in the private cloud, which can be centrally managed and administered based on dynamic workloads.

Docker images enable key components and functionality of the WebFOCUS platform to be isolated, packaged into specific containers and run on Kubernetes. As a result, each WebFOCUS component can be scaled independently to accommodate users, data, and data sources on demand.

## Kubernetes Overview

Kubernetes is an open-source system for running, managing, and orchestrating containerized applications in a cluster of servers (known as a Kubernetes cluster). Kubernetes clusters can run in any cloud environment (for example, private, public, hybrid) or on-premises.

A Kubernetes cluster consists of a set of worker machines, called nodes that run containerized applications. Every cluster has at least one worker node.

The size of a Kubernetes cluster (determined by the number of nodes) depends on the application workloads that are running. For example, each node can represent an 8 core / 64-GB RAM system. Pods are the basic objects in a Kubernetes cluster, which consists of one or more software containers. The worker nodes host the Pods that are the components of the application workload. Usually, only one container runs in a Pod. However, multiple containers can be run in a Pod if needed (depending on specific environment requirements). If one Pod fails, Kubernetes can automatically replace that Pod with a new instance.

Key benefits of Kubernetes include automatic scalability, efficient load balancing, high availability, failover/fault tolerance, and deploying updates across your environment without any disruptions to users.

# Requirements and Prerequisites

---

This section describes the requirements and prerequisites for deploying WebFOCUS DSML Services - Container Edition in a Kubernetes cluster.

Following are the list of requirements and prerequisites for deployment.

- [Docker Version 19.x upto 28.x](#)
- [Kubernetes Cluster Version 1.23.x upto 1.32.x](#)
- [Containerd Runtime](#)
- [kubectl](#)
- [Kubernetes Context and Kubeconfig](#)
- [Helm Version 3.5.x upto 3.16.x](#)

## Docker Version 19.x upto 28.x

If Docker is not installed, use the links below and follow the steps to install Docker on your machine:

<https://docs.docker.com/engine/install>

To verify the Docker version currently installed, enter the following command in your terminal window:

```
$> docker version
```

## Kubernetes Cluster Version 1.23.x upto 1.32.x

Users can consult with their Kubernetes administrator to provision the best available Kubernetes cluster on their environment (such as HA, non-HA control-plane etc.). A Kubernetes cluster running on the Amazon® Elastic Kubernetes Service (EKS) is recommended.

- The Kubernetes cluster must use CNI that supports UDP transport.
- Kubernetes must have access to the registry where WebFOCUS DSML Services -

Container Edition component images are available.

- The Kubernetes cluster must support dynamic provision of volumes (pv).
- The common volume should be 30 to 40 GB.
- If you are using a back-end data source/database that is running outside (external) of the Kubernetes cluster, then ensure it is running somewhere where the cluster has access to it.
- Each node in the Kubernetes cluster must be sufficiently powerful:

**i Note:** These specifications are recommended for production only, for development/poc purposes you can deploy WebFOCUS DSML Services - Container Edition on regular laptop/desktops.

- 32 GB RAM
- 8 CPU 3.x Ghz
- 50 GB Available Disk Space

**i Note:** The individual who is responsible for installing WebFOCUS DSML Services - Container Edition must have admin privileges over the Kubernetes namespace where WebFOCUS DSML Services is installed.

To verify the Kubernetes version currently installed, enter the following command in your terminal window:

```
$> kubectl version
```

## Containerd Runtime

Starting with Kubernetes version 1.25, containerd replacing Docker as the default runtime. With this change, Kubernetes architecture should be more streamlined and more compatible with other container runtimes, increasing its flexibility and efficiency.

```
$> kubectl get nodes -o wide
```

NAME	STATUS	ROLES	AGE	VERSION	OS- IMAGE	KERNAL- VERSIO N	CONTAINER- RUNTIME
mynode	Ready	control- plane, master	22h	v1.25.7+k3s 1	Ubuntu 22.04 LTS	5.15.0- 27- generic	containerd://1.6. 15-k3s1



**Note:** If your images are stored in the registry, then no need to run the docker save command. It is only required when you are running on a single node cluster, building and using images locally. For the multi-node cluster, your images are stored in some Docker registry.

You must use the docker save command after building WebFOCUS - Container Edition images from the build\_images.sh script.

The following are the commands for available images on containerd.

```
$> docker save ibi2020/webfocus:chart-9.3.5 | sudo ctr -n k8s.io images
import -
$> docker save ibi2020/webfocus:nlq-9.3.5 | sudo ctr -n k8s.io images
import -
$> docker save ibi2020/webfocus:metadata-9.3.5 | sudo ctr -n k8s.io
images import -
$> docker save ibi2020/webfocus:ml-functions-9.3.5 | sudo ctr -n k8s.io
images import -
$> docker save ibi2020/webfocus:analytics-9.3.5 | sudo ctr -n k8s.io
images import -
$> docker save ibi2020/webfocus:ollama-9.3.5 | sudo ctr -n k8s.io
images import -
```

## kubectl

The Kubernetes command-line tool, kubectl, allows you to run commands against Kubernetes clusters. You can use kubectl to deploy applications, inspect and manage cluster resources, and view logs.

If kubectl is not installed, use the following link to download and install kubectl (Linux, macOS):

<https://kubernetes.io/docs/tasks/tools>



## Kubernetes Context and Kubeconfig

Ensure that your context is set. To display a list of contexts currently being used, enter the following command in your terminal window:

```
$> kubectl config get-contexts
```

If required, enter the following command in your terminal window to set the context:

```
$> kubectl config use-context <my-cluster-name>
```

### Note:

- You can pass context directly with a `helmfile sync/destroy` command using a `--kube-context` parameter, it sets `kubectl` context otherwise it uses the current context by default.
- If you are using `webfocusce` tool, then pass the `--context` parameter.

## Helm Version 3.5.x upto 3.16.x

Helm is the “package manager” for Kubernetes, which is used to install charts (“packages”) into Kubernetes. You can consider charts as packaged applications. Charts are your versioned, pre-configured application resources, which can be deployed as one unit. A Helm chart (.tgz file) contains templates that define Kubernetes resources to install.

Install helm based on your Operating System from the given link.

<https://helm.sh/docs/intro/install>

To verify the Helm version currently installed, enter the following command in your terminal window:

```
$> helm version
```

# Key Components/Systems

---

Before deploying WebFOCUS DSML Services - Container Edition, ensure that the following systems are available in your environment with the corresponding components installed accordingly.

**Note:**

- If required, you can also run everything from a single machine (from building the Docker images to deploying WebFOCUS DSML Services - Container Edition in a Kubernetes cluster).
- If you are running on low resources and in a cloud environment, you can remove the `environments/resource-limit.yaml.gotmpl` line in the cloud environment, which is located in the `<webfocus-dsml-ce>/scripts/helmfile/helmfile.yaml` file. You can also update the default values for the allotted resources as required. For more information, see [Resource \(CPU and Memory\) Parameters](#).

1. **Build Server:** After obtaining the `.tar` file, this machine is used to build the required Docker images for WebFOCUS DSML Services - Container Edition on the image registry.

Recommended Specifications:

- 16 GB RAM
- 4 CPU 3.x Ghz
- 50 GB Available Disk Space

Installed Software

- Docker

2. **Deployment Server:** Using the Helm charts that are provided with each Docker image (as is or modified as per end-user requirements), this machine is used to deploy WebFOCUS DSML Services - Container Edition components on the Kubernetes cluster.

Recommended Specifications:

- 32 GB RAM
- 8 CPU 3.x Ghz
- 100 GB Available Disk Space

#### Installed Software

- Helm
- Kubectl

3. **Kubernetes Cluster:** Ensure that your Kubernetes cluster is ready/available and each node in your cluster where WebFOCUS DSML Services - Container Edition is being deployed supports the following recommended specifications:

- 32 GB RAM
- 8 CPU 3.x Ghz
- 100 GB Available Disk Space

# Creating or Using Docker Images

---

Pre-built WebFOCUS DSML Services - Container Edition images are included in the download package. You can download ready-to-use images .tar file from the eDelivery site. It helps provide a consistent environment across development, testing, and production environments. You can also create your own images using the `build-images.sh` script.

Following are the ways to build docker images:


- [Downloading Shipped Images to Load in Docker](#)
- [Building the Docker Images](#)

## Downloading Shipped Images to Load in Docker

With the use of pre-built container images, deployment becomes a straightforward process, eliminating the need to build and configure images from scratch, saving significant time and effort.

### Procedure

1. Download the `IBI_dsm1ce_images_9.3.5.tar` image file.

 **Note:** The minimum disk space requirement for shipped images is 40 GB.

2. Load images to Docker using the following command:

```
$> docker load -i IBI_dsm1ce_images_9.3.5.tar  
$> docker images
```

### Result

You can see 6 images loaded to your docker.

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
ibi2020/webfocus	chart-9.3.5	f96beb0b82f0	3 days ago	277MB
ibi2020/webfocus	ollama-9.3.5	1408d9890ac2	3 days ago	5.9GB
ibi2020/webfocus	nlq-9.3.5	05d3f2ca18bf	3 days ago	479MB
ibi2020/webfocus	analytics-9.3.5	9aaa510b9455	3 days ago	362MB
ibi2020/webfocus	metadata-9.3.5	cc27a1d7438e	3 days ago	577 MB
ibi2020/webfocus	ml-functions-9.3.5	38afab92e5c1	3 days ago	2.2GB

## Building and Loading WebFOCUS DSML Services - Container Edition images using Podman

**i Note:** You can either build the WebFOCUS DSML Services - Container Edition images using Podman or download the IBI\_dsmfce\_images\_9.3.5.tar file and load it with Podman, as outlined in the steps below.

### Procedure

1. To build WebFOCUS DSML Services - Container Edition images using Podman, navigate to the /scripts directory, set the environment variable TOOL=podman before building image and run the ./build-images.sh command.

```
$> cd /IBI_dsmfce_9.3.5/scripts
$> export TOOL=podman
$> ./build-images.sh
```

**i Note:** If you cannot set the TOOL then it defaults to Docker, and images builds using Docker.

2. To load WebFOCUS DSML Services - Container Edition images using Podman, run the `podman load` command.
  - a. Download the `IBI_dsm1ce_images_9.3.5.tar` image file.
  - b. You can load all WebFOCUS DSML Services - Container Edition images using the following command:

```
podman load -i IBI_dsm1ce_images_9.3.5.tar
```

3. Verify images after building or loading images using the following command:

```
podman images
```

# Installing ibi WebFOCUS DSML Services - Container Edition

---

WebFOCUS provides a containerized data science and machine learning (DSML) platform, which can also be deployed and run with ibi™ WebFOCUS® - Container Edition in a Kubernetes cluster.

WebFOCUS DSML Services - Container Edition consists of the following services:

- **Instant Insights (autoanalytics):** Enables you to run advanced analyses and generate visualizations and narratives on your data sets, without manually preparing and analyzing your data, or having prior knowledge of data science or statistics.
- **Machine Learning Functions (ml-functions):** When creating a Data Flow, you can easily run predictive analytics on your data sets using Machine Learning functions, without prior knowledge of advanced statistics.
- **Metadata Classification (metadata):** Examines your data and assigns classifications to the columns, which can then be used to match columns from separate data sources. You can classify data that you upload, and use it to match fields in a Union in a Data Flow.
- **Natural Language Query (nlq):** Allows you to ask questions of your data using everyday language. This provides you with valuable insights and allows you to make informed business decisions. NLQ translates natural language into SQL code that can be ran against a database.
- **Chart Recommendation for Visualisation Query (chart\_rec):** Changing the Visualization of Query, the system defines recommended visualization for the result and the result gets generated in that visualized format. Currently we support Bar Chart and Line Chart Visualization formats. The recommendations are based on the following rules.

Format	Variables in the Dataset	Number of Records
Bar chart	An ordered data variable that can be identified as a dimension.	Less than 21
Line chart	An ordered data variable that can be identified as a dimension. A temporal data variable.	Less than 50



# Building the Docker Images

WebFOCUS DSML Services - Container Edition is packaged and provided as a .tar file for the following Linux platform:

```
x86-64 (IBI_dsmIce_9.3.5_linux_x86_64.tar)
```

**i Note:** The process of building your Docker images for WebFOCUS DSML Services - Container Edition and pushing them to a Docker registry can take place on a separate machine from the one that is used to deploy to the Kubernetes cluster.

## Procedure

1. Download and extract the WebFOCUS DSML Services - Container Edition .tar file to a location on your file system.

```
$> tar -xf IBI_dsmIce_9.3.5_linux_x86_64.tar
```

Once you have downloaded and unzipped the .tar file, you must build the Docker image that is required by WebFOCUS DSML Services - Container Edition.

2. Navigate to the <webfocus-dsmIce>/scripts subdirectory:

```
$> cd scripts
```

3. List the contents of the <webfocus-dsmIce>/scripts subdirectory.

```
$> ls
```

The following contents are listed, which includes the build script (build-images.sh).

```
build-images.sh  dsmIce-chart
```

4. Enter the following command to run the build-images.sh script, which creates the Docker image for WebFOCUS DSML Services - Container Edition:

```
$> ./build-images.sh
```

The following Docker images are created:

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
ibi2020/webfocus	chart-9.3.5	1b0dc23fc605	4 days ago	277MB
ibi2020/webfocus	ollama-9.3.5	61fed7e11765	4 days ago	10.5GB
ibi2020/webfocus	nlq-9.3.5	7eed45c95588	4 days ago	479MB
ibi2020/webfocus	analytics-9.3.5	5c7145c19fd2	4 days ago	363MB
ibi2020/webfocus	metadata-9.3.5	313a1e15593c	4 days ago	577MB
ibi2020/webfocus	ml-functions-9.3.5	cd6aa1db6007	4 days ago	2.22GB

# Deploying ibi WebFOCUS DSML Services - Container Edition on Kubernetes Cluster

Once you have created your Docker image for WebFOCUS DSML Services - Container Edition, you can deploy it to your Kubernetes cluster. Enter the following command in your terminal window/console:

```
$> helm install dsml dsml-chart --set global.ACCEPT_EUA=Y -n <namespace> --create-namespace
```

**i Note:** You must accept global.ACCEPT\_EUA=Y, otherwise the deployment fails. Either you can set it in the helm chart or on the command line:

```
--set global.ACCEPT_EUA=Y
```

where:

<namespace> is the webfocus namespace.

This command deploys WebFOCUS DSML Services - Container Edition without resource limits. However, if you want to deploy WebFOCUS DSML Services - Container Edition to a Kubernetes cluster using Helm chart with resource limits, then pass the resource-values.yaml file path.

Run the following command to deploy the WebFOCUS DSML Services - Container Edition images:

```
$> helm install dsml -f dsml-chart/resource-values.yaml dsml-chart --set global.ACCEPT_EUA=Y -n <namespace> --create-namespace
```

where:

<namespace> is the webfocus namespace.

**i Note:** If your WebFOCUS cluster and WebFOCUS DSML Services - Container Edition are deployed on the same namespace, WebFOCUS selects WebFOCUS DSML Services - Container Edition automatically.

If you are running on any other namespace or are hosted somewhere else, check and update the `pyserv_url` setting in the ibi™ WebFOCUS® Reporting Server `eda.cfg` file with the correct URL.

# Deploying ibi WebFOCUS DSML Services - Container Edition Using Docker Compose

Following are the steps to deploy WebFOCUS DSML Services - Container Edition using Docker Compose.

## Procedure

1. Download the WebFOCUS DSML Services - Container Edition tar file and extract it.
2. To deploy, use the following commands.

```
$> cd <webfocus-dsml-ce>/scripts/docker-compose  
$> docker compose up
```

3. To check the running container, use the following command:

```
$> docker compose ps
```

4. To undeploy the cluster, use the following command:

```
$> docker compose down
```

If you want to run nginx on SSL configuration. For SSL configuration Certificate and Key files are required, and follow the below step.

Following command creates `dsml.crt` and `dsml.key` file.

```
openssl req -x509 -newkey rsa:4096 -keyout dsml.key -out dsml.crt -  
sha256 -days 3650 -nodes
```

You can use CA signed certificate instead of self signed certificate, see <https://docs.gunicorn.org/en/20.1.0/settings.html#ssl> for Gunicorn SSL configuration

# Deploying ibi WebFOCUS DSML Services - Container Edition with Red Hat OpenShift Cluster

Red Hat OpenShift, a hybrid cloud application platform powered by Kubernetes, brings together tested and trusted services to reduce the friction of developing, modernizing, deploying, running, and managing applications. Red Hat OpenShift delivers a consistent experience across public cloud, on-premises, hybrid cloud, and edge architecture.

## Prerequisites

- Helm
- Docker/Podman

## System Requirements

- Memory: At least 32 GB of RAM.
- CPU: At least 8 CPU cores.
- Disk space: Approximately 80 GB of free disk space.

## Supported Platforms

A Linux distribution (RHEL, Fedora, Ubuntu, CentOS, or similar) with a 64-bit architecture.



**Caution:** Code snippets in the PDF format might have undesired line breaks due to space constraints. You must verify code snippets before directly copying and pasting them in your program.

## Setting up CRC

### Procedure

1. Select the operating system from the dropdown list and download the Red Hat OpenShift local installer from the following link:

<https://console.redhat.com/openshift/create/local>

2. Extract the downloaded CRC tar file to your specified location and add it to the path using the following command:

```
$ tar xvf crc-linux-amd64.tar.xz
crc-linux-2.16.0-amd64/
crc-linux-2.16.0-amd64/LICENSE
crc-linux-2.16.0-amd64/crc

$ mkdir -p ~/local/bin

$ mv crc-linux-*-amd64/crc ~/local/bin/

$ export PATH=$HOME/local/bin:$PATH

$ crc version
CRC version: 2.16.0+05b62a75
OpenShift version: 4.12.9
Podman version: 4.4.1

$ echo 'export PATH=$HOME/local/bin:$PATH' >> ~/.bashrc
```

3. Navigate to the extracted folder and set up the Red Hat OpenShift cluster using the following command:

```
$> crc setup
```

4. Check the default configuration of CPU, memory, and disk space using the following command:

```
$> crc config view
```

Update the Red Hat OpenShift cluster default configuration to the recommended configuration for CPU, memory, and disk space using the following command:

```
$> crc config set memory 16000
$> crc config set cpus 6
$> crc config set disk-size 80
```

5. You need a pull secret to start the CRC, which you can download or copy from <https://console.redhat.com/openshift/create/local>, and run the following command:

```
$> crc start -p <path-to-your-pull-secret-file>
```

```
Started the OpenShift cluster.

The server is accessible via web console at:
  https://console-openshift-console.apps-crc.testing

Log in as administrator:
  Username: kubeadmin
  Password: MMRq-QQcMM-VjCna-gn9yX

Log in as user:
  Username: developer
  Password: developer

Use the 'oc' command line interface:
  $ eval $(crc oc-env)
  $ oc login -u developer https://api.crc.testing:6443
```

## Internal Registry for Getting Images

Use the following steps to deploy the WebFOCUS DSML Services - Container Edition using internal registry images.

### Procedure

1. Push the images using Docker or Podman:
  - a. Update the Docker configuration to allow push images to the unsecure Red Hat OpenShift local registry using the following command:

```
$> vi ~/etc/docker/daemon.json
```

After opening the daemon.json, add the following entry:

```
{
  "insecure-registries" : [ "default-route-openshift-image-
registry.apps-crc.testing:443" ]
}
```

Now, restart the docker using the following command:



```
$> sudo systemctl daemon-reload
$> sudo systemctl restart docker
```

- b. To push the images using Podman, use the following command:

```
vi /etc/containers/registries.conf

[registries.insecure]
registries = ['default-route-openshift-image-registry.apps-
crc.testing:443']
```

2. Log in to the Red Hat OpenShift cluster as a developer user using the following command:

```
$> eval $(crc oc-env)
$> oc login -u developer https://api.crc.testing:6443
```

3. Create the new Red Hat OpenShift project using the following command:

```
$> oc new-project webfocus
```

4. Download the IBI\_dsm1ce\_9.3.5.tar from [Tibco eDelivery](#).

Navigate to <webfocus-dsm1-ce>/scripts and build the Docker images using the following command:

```
$> ./build-images.sh
```

Verify the images using the following command:

```
$> docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
ibi2020/webfocus	chart-9.3.5	f96beb0b82f0	3 days ago	277MB
ibi2020/webfocus	ollama-9.3.5	1408d9890ac2	3 days ago	5.9GB

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
ibi2020/webfocus	nlq-9.3.5	05d3f2ca18bf	3 days ago	479MB
ibi2020/webfocus	analytics-9.3.5	9aaa510b9455	3 days ago	362MB
ibi2020/webfocus	metadata-9.3.5	cc27a1d7438e	3 days ago	577MB
ibi2020/webfocus	ml-functions-9.3.5	38afab92e5c1	3 days ago	2.2GB

5. Tag the images and push it to the CRC internal registry using the following command:

```
$> docker login -u `oc whoami` -p `oc whoami --show-token` default-route-openshift-image-registry.apps-crc.testing:443

$> docker image tag ibi2020/webfocus:chart-9.3.5 default-route-openshift-image-registry.apps-crc.testing:443/webfocus/repo:chart-9.3.5
docker image push default-route-openshift-image-registry.apps-crc.testing:443/webfocus/repo:chart-9.3.5

$> docker image tag ibi2020/webfocus:ollama-9.3.5 default-route-openshift-image-registry.apps-crc.testing:443/webfocus/repo:ollama-9.3.5
docker image push default-route-openshift-image-registry.apps-crc.testing:443/webfocus/repo:ollama-9.3.5

$> docker image tag ibi2020/webfocus:nlq-9.3.5 default-route-openshift-image-registry.apps-crc.testing:443/webfocus/repo:nlq-9.3.5
docker image push default-route-openshift-image-registry.apps-crc.testing:443/webfocus/repo:nlq-9.3.5

$> docker image tag ibi2020/webfocus:analytics-9.3.5 default-route-openshift-image-registry.apps-crc.testing:443/webfocus/repo:analytics-9.3.5
docker image push default-route-openshift-image-registry.apps-crc.testing:443/webfocus/repo:analytics-9.3.5

$> docker image tag ibi2020/webfocus:metadata-9.3.5 default-route-openshift-image-registry.apps-
```

```
crc.testing:443/webfocus/repo:metadata-9.3.5
docker image push default-route-openshift-image-registry.apps-
crc.testing:443/webfocus/repo:metadata-9.3.5

$> docker image tag ibi2020/webfocus:ml-functions-9.3.5 default-
route-openshift-image-registry.apps-
crc.testing:443/webfocus/repo:ml-functions-9.3.5
docker image push default-route-openshift-image-registry.apps-
crc.testing:443/webfocus/repo:ml-functions-9.3.5
```

Verify the completion using the following command:

```
$> oc get imagestreams repo -n webfocus -o jsonpath='{range
.status.tags[*]}{.tag}{"\n"}'
```

6. Deploy WebFOCUS DSML Services - Container Edition using the following command:

```
$> helm upgrade --install dsml -f dsml-chart/images-values.yaml
dsml-chart -n <namespace>
```

7. All pods should be running fine as below after successful deployment of WebFOCUS DSML Services - Container Edition.

NAME	READY	STATUS	RESTARTS	AGE
autoanalytics-6ccb995db-72nb6	1/1	Running	0	114s
chart-7499d677f8-4fn9b	1/1	Running	0	114s
metadata-786c987c9d-zgchm	1/1	Running	0	114s
ml-functions-75cf8c98b8-lt6ps	1/1	Running	0	114s
nlq-5bfd486886-sdm4s	1/1	Running	0	114s
ollama-7f4fb5bcd-b-8mkg2	1/1	Running	0	114s

8. Create a route to access the appserver using the following command:

```
$> oc expose svc dsml-common -n <namespace>
```

View the route using the following command:

```
$> oc get route -n <namespace>
```

For more information on the route, see [Route Configuration](#).

9. You can increase the timeout limit of a route by using the following command:

```
$> oc annotate route dsml-common --overwrite
haproxy.router.openshift.io/timeout=5m -n <namespace>
```

**Note:** The annotation `haproxy.router.openshift.io/timeout` is specific to Red Hat OpenShift's HAProxy-based router. It configures the HTTP timeout for the route. This is the maximum time that the router waits for a response from the backend service before terminating the connection.

10. Redirect to <http://localhost:31099/metadata/v1/system> in any browser for confirmation.

## Deploying ibi WebFOCUS DSML Services-Container Edition with Podman and Podman Compose On RHEL 9

Podman is a lightweight, daemonless container engine compatible with Docker, offering a flexible and secure environment for running containers, including rootless operation. Podman Compose, a tool for defining and running multi-container Podman applications, simplifies the orchestration of WebFOCUS DSML Services components. This deployment method is useful for non-production environments, local development, and streamlined container orchestration.

# Installing Podman on RHEL 9

This section covers the installation and initial testing of Podman on RHEL 9.

## Step 1: Update the System

Ensure that your system is up to date before proceeding.

```
sudo dnf update -y
```

## Step 2: Install Podman

Install the Podman package using DNF.

```
sudo dnf install -y podman
```

## Step 3: Verify Installation

Verify that Podman is installed correctly by checking its version.

```
podman --version
```

## Optional: Enable and Start the Podman Service

For socket activation or rootless containers, enable and start the Podman service.

```
systemctl --user enable --now podman.socket
```

**i Note:** This step requires a non-root user and lingering to be enabled.

```
loginctl enable-linger $USER
```

## Step 4: Test Podman

Run a test container to verify that Podman is functioning correctly.

```
podman run --rm -it alpine sh
```

# Troubleshooting Podman Installation

This section addresses common errors encountered during Podman installation and testing.

## Error: Copying System Image

If you encounter the following error:

```
Error: copying system image from manifest list: writing blob: adding
layer with blob
"sha256:fe07684b16b82247c3539ed86a65ff37a76138ec25d380bd80c869a1a4c73236
"/"/"/"sha256:fd2758d7a50e2b78d275ee7d1c218489f2439084449d895fa17eede6c6
1ab2c4": unpacking failed (error: exit status 1; output: potentially
insufficient UIDs or GIDs available in user namespace (requested 0:42
for /etc/shadow): Check /etc/subuid and /etc/subgid if configured
locally and run "podman system migrate": lchown /etc/shadow: invalid
argument)
```

This indicates an issue with UID/GID mapping. Follow the steps below to resolve it.

## Configure Subuid and Subgid Properly

1. Check your username:

```
whoami
```

Let us assume your username is "John".

2. Edit /etc/subuid and /etc/subgid:

Ensure that the following entries exist.

```
john:100000:65536
```

Add these entries if they are missing.

```
sudo sh -c 'echo "john:100000:65536" >> /etc/subuid'  
sudo sh -c 'echo "john:100000:65536" >> /etc/subgid'
```



**Note:** Ensure that there are no duplicate entries or overlapping ranges.

### 3. Run Podman System Migration:

After fixing the UID/GID ranges, run the following command.

```
podman system migrate
```

This updates your user storage and configuration based on the new mappings.

## Installing Podman Compose

This section guides you through the installation of Podman Compose on RHEL 9.

### Steps to Enable EPEL on RHEL 9

Podman Compose is available in the Extra Packages for Enterprise Linux (EPEL) repository.

### Install EPEL Repo Manually

Download and install the epel-release RPM.

```
sudo dnf install -y https://dl.fedoraproject.org/pub/epel/epel-release-  
latest-9.noarch.rpm
```

### Enable EPEL and Refresh the Cache

Enable the EPEL repository and refresh the DNF cache.

```
sudo dnf config-manager --enable epel  
sudo dnf makecache
```

## Install Podman Compose

Install Podman Compose using DNF.

```
sudo dnf install -y podman-compose
```

# Deploying ibi WebFOCUS DSML Services - Container Edition Using Podman Compose

To deploy WebFOCUS DSML Services - Container Edition using podman compose, perform the following steps.

**i Note:** It is recommended only for non production deployment.

1. Download the WebFOCUS DSML Services - Container Edition tar file and extract it.
2. Update the docker compose environment file, navigate to <webfocus-dsml-ce>/scripts/docker-compose.

```
$> cd <webfocus-dsml-ce>/scripts/docker-compose
```

3. Build WebFOCUS DSML Services images, refer [Building and Loading WebFOCUS DSML Services - Container Edition images using Podman](#).
4. Deploy it on the server and client using the following command:

```
podman compose up -d
```

5. Undeploy the cluster using the following command:

```
podman compose down
```



6. Check containers using below command:

```
podman ps
```

# Testing and Verifying

After deployment, to check the status of the pod that is running WebFOCUS DSML Services - Container Edition, enter the following command in your terminal window/console:

```
$> kubectl -n webfocus get pods -l app.kubernetes.io/component=dsml
```

A status of the pod is returned, as shown in the following example.

NAME	READY	STATUS	RESTARTS	AGE
autoanalytics-6ccb995db-72nb6	1/1	Running	0	114s
chart-7499d677f8-4fn9b	1/1	Running	0	114s
metadata-786c987c9d-zgchm	1/1	Running	0	114s
ml-functions-75cf8c98b8-lt6ps	1/1	Running	0	114s
nlq-5bfd486886-sdm4s	1/1	Running	0	114s
ollama-7f4fb5bcdb-8mkg2	1/1	Running	0	114s

To invoke the WebFOCUS DSML Services - Container Edition endpoint, enter the following command in your terminal window/console:

```
$> kubectl -n webfocus exec -it edaserver-0 -- /bin/bash
```

```
[ibi@edaserver-0 temp]$ curl http://dsml-common/autoanalytics/v1/system
{"data":{"buildDate":"2021-09-18T10:26:03Z","gitCommit":"a30fe8cf455","version":"1.4.1"}}
```

You can also check if the WebFOCUS DSML Services - Container Edition is running by accessing the following URL from a web browser or using Curl (exposed via NGINX running on NodePort 31088).

```
http://<host-name>:<host-port>/metadata/v1/system
```

```
{"data":{"buildDate":"2025-06-13T11:56:39Z","gitCommit":"563a91913e9","version":"9.3.5"}}
```

```
http://<host-name>:<host-port>/machinelearning/v1/system
```

```
{"data":{"buildDate":"2025-07-10T05:05:08Z","gitCommit":"02e91913744","version":"9.3.5"}}
```

```
http://<host-name>:<host-port>/autoanalytics/v1/system
```

```
{"data":{"buildDate":"2025-07-09T04:37:43Z","gitCommit":"1c9abf826d4","version":"9.3.5"}}
```

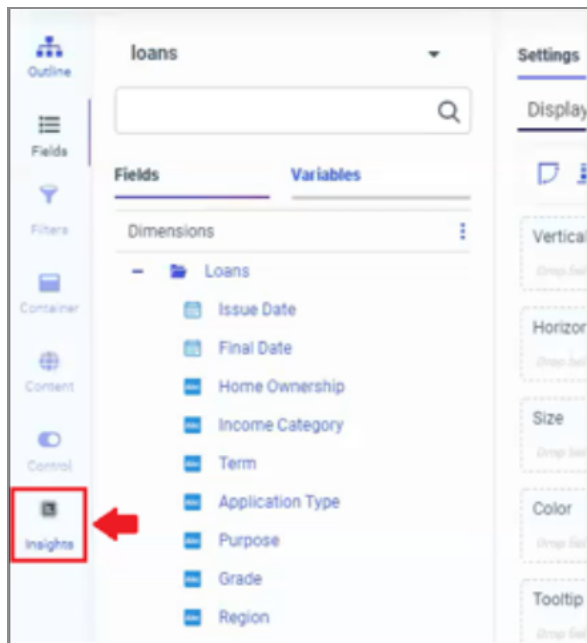
```
http://<host-name>:<host-port>/nlq/v1/system
```

```
{"data":{"buildDate":"2025-07-09T12:15:19Z","gitCommit":"649d47680b7","version":"9.3.5"}}
```

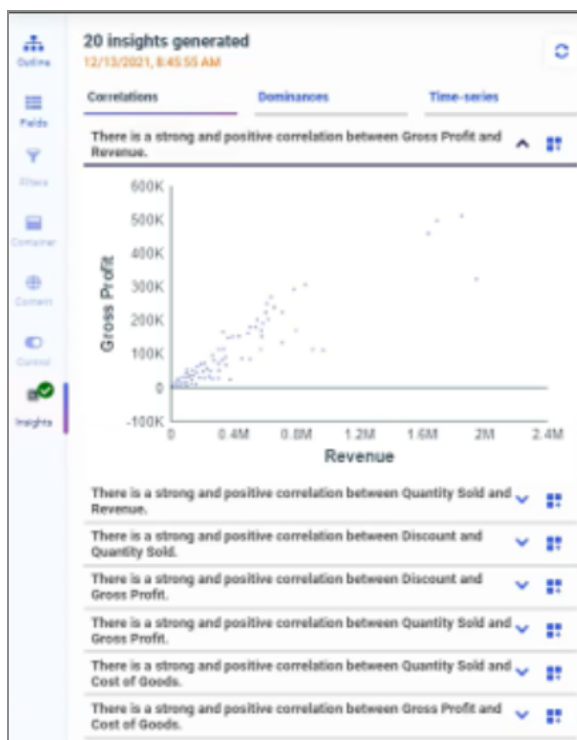
```
http://<host-name>:<host-port>/chart/v1/system
```

```
{"data":{"buildDate":"2025-07-09T04:42:30Z","gitCommit":"fe77251fa23","version":"9.3.5"}}
```

To verify WebFOCUS DSML Services - Container Edition functionality in your WebFOCUS instance, create a new visualization, which opens ibi™ WebFOCUS® Designer where you are initially prompted to select a data source. After selecting a data source and WebFOCUS® Designer opens, you will notice that the Insights tab is displayed in the left pane, as shown in the following image.



Clicking the Insights tab automatically generate insights on the data source that is currently being used in WebFOCUS Designer, as shown in the following image.



# Upgrading ibi WebFOCUS DSML Services - Container Edition

---

To upgrade WebFOCUS DSML Services - Container Edition to a new version in your Kubernetes cluster:


## Procedure

1. Build the required Docker images for the new version of WebFOCUS DSML Services - Container Edition (packaged as a .tar file), as described in Building the Docker Images.

Once you have created your Docker images from the new .tar, you can upgrade the previous version of WebFOCUS DSML Services - Container Edition that has been deployed in your Kubernetes cluster using the new images.

2. Enter the following commands in your terminal window/console:

```
$> cd scripts  
$> helm upgrade dsml dsml-chart -n <namespace>
```

 **Note:** If you used -f dsml-chart/resource-values.yaml or any other customized file when previously deploying WebFOCUS DSML Services - Container Edition, then use the same parameter when upgrading to the new version.

# Uninstalling ibi WebFOCUS DSML Services - Container Edition

---

To uninstall WebFOCUS DSML Services - Container Edition from your Kubernetes cluster, enter the following command in your terminal window/console:

```
$> helm -n webfocus delete dsml
```

# Usage and Environment Considerations

---

This section describes the usage and considerations for WebFOCUS DSML Services - Container Edition in your Kubernetes cluster.

## Scaling

If you are planning or have a requirement to scale your WebFOCUS DSML Services - Container Edition deployment, ensure the following:

1. You are using a private Docker registry to pull images.
2. Update the image pull Secret in your configuration (`platform.dockerconfigjson`).
3. Your storage provider is using the ReadWriteMany access mode.

## Resource Limits

Be aware of resource limits and make the appropriate adjustments according to your environment.

## Resource (CPU and Memory) Parameters

This section lists and describes the resource parameters that control CPU and memory (RAM) usage.

By default, resource limits can only be applied to the cloud environment. There are no resource limits for the dev environment. However, you can modify this default behavior if required by including `resource-limit.yaml.gotmpl` for the dev environment in Helmfile.

**i Note:** Limits and requests for ephemeral-storage are measured in bytes. You can express storage as a plain integer or as a fixed-point number using one of the following suffixes: E, P, T, G, M, K

You can also use the following power-of-two equivalents: Ei, Pi, Ti, Gi, Mi, Ki

For example, the following represent roughly the same value: 128974848, 129e6, 129M, 123Mi

Parameter	Description	Default Value
auto-analytics.resources.requests.memory	auto-analytics pod/container reserves requested memory in the system.	4Gi
auto-analytics.resources.requests.cpu	auto-analytics pod/container reserves the requested CPU in the system (2 = 2 CPU).	2
auto-analytics.resources.limits.memory	auto-analytics container max limit for the memory (RAM).	4Gi
auto-analytics.resources.limits.cpu	auto-analytics container max limit for the CPU.	2
metadata.resources.requests.memory	metadata pod/container reserves requested memory in the system.	4Gi



Parameter	Description	Default Value
metadata.resources.requests.cpu	metadata pod/container reserves the requested CPU in the system (1 = 1 CPU).	2
metadata.resources.limits.memory	metadata container max limit for the memory (RAM).	4Gi
metadata.resources.limits.cpu	metadata container max limit for the CPU (2 = 2 CPU).	2
chart-rec.resources.requests.memory	chart-rec pod/container reserves requested memory in the system.	4Gi
chart-rec.resources.requests.cpu	chart-rec pod/container reserves the requested CPU in the system.	2
chart-rec.resources.limits.memory	chart-rec container max limit for the memory (RAM).	4Gi
chart-rec.resources.limits.cpu	chart-rec container max limit for the CPU.	2
ml-functions.resources.requests.memory	ml-functions pod/container reserves requested memory in the system.	4Gi
ml-functions.resources.requests.cpu	ml-functions pod/container reserves the requested CPU in the system.	2
ml-functions.resources.limits.memory	ml-functions container max limit for the memory (RAM).	4Gi
ml-functions.resources.limits.cpu	ml-functions container max limit for the CPU.	2
nlq.resources.requests.memory	nlq pod/container reserves requested memory in the system.	4Gi

Parameter	Description	Default Value
nlq.resources.requests.cpu	nlq pod/container reserves the requested CPU in the system.	2
nlq.resources.limits.memory	nlq container max limit for the memory (RAM).	4Gi
nlq.resources.limits.cpu	nlq container max limit for the CPU.	2
dsml-common.resources.requests.memory	dsml-common pod/container reserves requested memory in the system.	1Gi
dsml-common.resources.requests.cpu	dsml-common pod/container reserves requested CPU in the system.	1
dsml-common.resources.limits.memory	dsml-common container max limit for the memory (RAM).	1Gi
dsml-common.resources.limits.cpu	dsml-common container max limit for the CPU.	1

# Third-Party Licenses for the Container Image

## Analyze container images and their associated licenses

### Base container image

WebFOCUS DSML Services - Container Edition includes a redhat/ubi9 official container image as the common base image layer for building images. For details on Debian licenses and software package types, see the [license information](#).

As with all container images, the Red Hat ubi9 container image may contain other software (such as bash, glibc, zlib, and others from the base distribution, along with any direct or indirect dependencies of the primary software included in the built image) that is subject to other licenses.

**Note:** The image user is responsible for ensuring that any use of the image complies with all relevant licenses for all software contained within.

## Additional software packages

Building images often installs additional software packages (fetched from the official distribution software repositories, from other user-added repositories, or from specific locations), in addition to installing the packages provided by the base image. You can inspect the Dockerfiles to identify these additional packages. For example, when you read the Dockerfile, you see a list of packages that are installed in the image, as specified in the Dockerfile.

**Note:** Each such specified package can, in turn, install other software packages as dependencies.

## Manually retrieve installed package information

You can inspect a container image and retrieve its contents with standard container and package management tools.

Use the `yum-query` command to retrieve the full list of installed packages in a container image.

Example: To retrieve the list of installed packages in the WebFocus Reporting server image, run the following command:

```
$> docker run --rm --entrypoint yum <image-name> list installed
```

## Result

This system is not registered with an entitlement server. You can use a subscription-manager to register.

Installed Packages		
acl.x86_64	2.3.1-3.el9	@System
alsa-lib.x86_64	1.2.7.2-1.el9	@ubi-9-appstream-rpms
alternatives.x86_64	1.20-2.el9	@System
audit-libs.x86_64	3.0.7-103.el9	@System
avahi-libs.x86_64	0.8-12.el9	@ubi-9-baseos-rpms
basesystem.noarch	11-13.el9	@System
bash.x86_64	5.1.8-6.el9_1	@System
bzip2-libs.x86_64	1.0.8-8.el9	@System
ca-certificates.noarch	2022.2.54-90.2.el9_0	@System
cmake-filesystem.x86_64	3.20.2-7.el9	@ubi-9-appstream-rpms
copy-jdk-configs.noarch	4.0-3.el9	@ubi-9-appstream-rpms

## Manually retrieve installed packages licenses

Use the `rpm` command to retrieve the license for any package, use the following command:

```
docker run --rm --entrypoint rpm <image-name> rpm -qi <package-name> |
grep License
```

Example: To retrieve the license for the installed package `libXtst-1.2.3-16.el9.x86_64`, run the following command:

```
docker run --rm --entrypoint rpm <image-name> rpm -qi libXtst-1.2.3-
16.el9.x86_64 | grep License
```

```
Output:
License      : GPLv2+
License      : MIT
```

```
docker run --rm --entrypoint /bin/bash <image-name> -c 'for package in
$(rpm -qa); do echo "Package: $package"; rpm -qi $package | grep
License; done'
```

## Manually retrieve installed files

### Procedure

1. Use the command `docker` to extract the contents of a container for further inspection.

**i Note:** If you create a new container from an image, you can inspect it without running the container.

Example: To create a temporal container called `temp-container`, which is based on the `unknown-image:latest` image, run the following command:

```
$> docker create --name temp-container unknown-image:latest
```

2. Extract the container file system as a `.tar` file. Following command is an example.

```
$> docker export temp-container > temp-container.tar
```

You can also directly extract the files list without creating a .tar archive. Following command is an example.

```
$> docker export temp-container | tar t > temp-container-files.txt
```

The previous commands create and export the contents of a stopped container. Both commands are direct ways to extract the image's final file system, which is a composite view of a container instance.

Alternatively, to create an image .tar file, run the `docker image save` command. Following command is an example:

```
$> docker image save unknown-image:latest > temp-image.tar
```

The previous commands produce an archive that exposes the image format, not the containers created from it. The .tar file includes a `manifest.json` file that describes the image's layers and a set of separate directories containing the content of each of the individual layers. This method is helpful when you want to evaluate each layer's role in building the image.

For more information, see the [Docker CLI](#) documentation.

For more information on the container image format, see the [OCI image format specification](#).

## Manually retrieve installed package sources

You can use the `rpm` command to retrieve the source for an installed package.

Example: To retrieve the source for the installed package, use the following command:

```
docker run --rm --entrypoint /bin/bash <image-name> -c "for package in
\$(rpm -qa); do echo -e \"Package: \$package Source RPM: \$(rpm -qi
\$package | awk '/Source RPM/ {print \$4}')\"; done"
```

# ibi Documentation and Support Services

---

For information about this product, you can read the documentation, contact Support, and join Community.

## How to Access ibi Documentation

Documentation for ibi products is available on the [Product Documentation website](#), mainly in HTML and PDF formats.

The [Product Documentation website](#) is updated frequently and is more current than any other documentation included with the product.

## Product-Specific Documentation

The documentation for this product is available on the [ibi™ WebFOCUS® DSML Services - Container Edition Documentation](#) page.

## How to Contact Support for ibi Products

You can contact the Support team in the following ways:

- To access the Support Knowledge Base and getting personalized content about products you are interested in, visit our [product Support website](#).
- To create a Support case, you must have a valid maintenance or support contract with a Cloud Software Group entity. You also need a username and password to log in to the [product Support website](#). If you do not have a username, you can request one by clicking **Register** on the website.

## How to Join ibi Community

ibi Community is the official channel for ibi customers, partners, and employee subject matter experts to share and access their collective experience. ibi Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from ibi products. For a free registration, go to [ibi Community](#).

# Legal and Third-Party Notices

---

SOME CLOUD SOFTWARE GROUP, INC. (“CLOUD SG”) SOFTWARE AND CLOUD SERVICES EMBED, BUNDLE, OR OTHERWISE INCLUDE OTHER SOFTWARE, INCLUDING OTHER CLOUD SG SOFTWARE (COLLECTIVELY, “INCLUDED SOFTWARE”). USE OF INCLUDED SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED CLOUD SG SOFTWARE AND/OR CLOUD SERVICES. THE INCLUDED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER CLOUD SG SOFTWARE AND/OR CLOUD SERVICES OR FOR ANY OTHER PURPOSE.

USE OF CLOUD SG SOFTWARE AND CLOUD SERVICES IS SUBJECT TO THE TERMS AND CONDITIONS OF AN AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER AGREEMENT WHICH IS DISPLAYED WHEN ACCESSING, DOWNLOADING, OR INSTALLING THE SOFTWARE OR CLOUD SERVICES (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH LICENSE AGREEMENT OR CLICKWRAP END USER AGREEMENT, THE LICENSE(S) LOCATED IN THE “LICENSE” FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE SAME TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of Cloud Software Group, Inc.

ibi, the ibi logo, WebFOCUS, and TIBCO are either registered trademarks or trademarks of Cloud Software Group, Inc. in the United States and/or other countries.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only. You acknowledge that all rights to these third party marks are the exclusive property of their respective owners. Please refer to Cloud SG’s Third Party Trademark Notices (<https://www.cloud.com/legal>) for more information.

This document includes fonts that are licensed under the SIL Open Font License, Version 1.1, which is available at: <https://scripts.sil.org/OFL>

Copyright (c) Paul D. Hunt, with Reserved Font Name Source Sans Pro and Source Code Pro.

Cloud SG software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the “readme” file for the availability of a specific version of Cloud SG software on a specific operating system platform.



THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. CLOUD SG MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S), THE PROGRAM(S), AND/OR THE SERVICES DESCRIBED IN THIS DOCUMENT AT ANY TIME WITHOUT NOTICE.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "README" FILES.

This and other products of Cloud SG may be covered by registered patents. For details, please refer to the Virtual Patent Marking document located at <https://www.cloud.com/legal>.

Copyright © 2021-2025. Cloud Software Group, Inc. All Rights Reserved.