Drut

Reinvent the Data Center | Reimagine the Machine

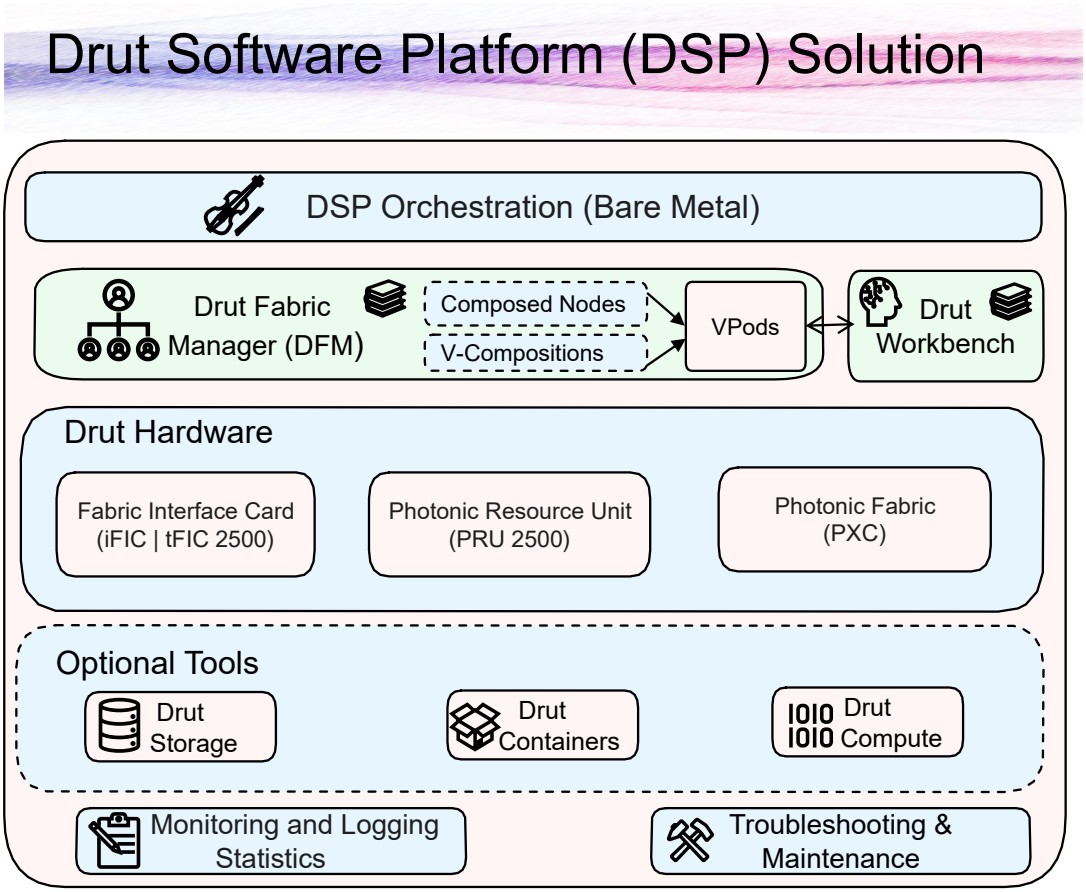
DSP Installation Guide, Version 3.2.0

Contents

Chapter 1. Introduction

The Drut Software Platform (DSP) is a full stack on-prem private cloud infrastructure and platform-as-a-service software solution that allows you to offer cloud-like services and automation to end-users. It is a multi-layered solution anchored by the DSP, which consists of a collection of open-source software services that are used to curate a standards-based environment for next-generation cloud services.

**Hover over each icon/block and click to view the related documentation.*



1. Installing Drut Orchestration *(on page 9)*
2. Installing Drut Fabric Manager *(on page 24)*
3. Installing Drut Storage *(on page 33)*
4. Installing Drut Containers *(on page 21)*
5. Installing Drut Compute *(on page 38)*
6. Setting and Enabling the Drut Monitoring and Logging Services *(on page 17)*
7. View Troubleshooting and Maintenance Procedures *(on page 41)*
8. Installing Drut Workbench *(on page 31)*

Drut sells and supports this software stack as three independent software bundles as part of a managed service offering, which are built for you to own, and we assembled the DSP in a way to avoid vendor and managed service lock-in. The flexibility provided by these solution bundles allow you build your own private cloud infrastructure tailored to your applications and allows you to maintain control at the level that you are comfortable with increasing control as you train your staff.

Chapter 2. Software, Hardware, and System Prerequisites

To get started, we will need a bare metal server to deploy the DSP Orchestration software. You are required to install Ubuntu 20.04 operating system on the bare metal, because the Drut packages are designed to work seamlessly with it.

To begin the installation, first download the dsp_ansible package from Drut Support (support@drut.io) and upload it to your Ubuntu 20.04 machine. After completing the installation, you should be able to log in to the DSP graphical user interface (GUI).

Tip: To allow sudo commands without requiring a password, insert the following line into `/etc/sudoers`:

```
<$USER_NAME> ALL=(ALL) NOPASSWD: ALL
```

Important: We recommend setting up a fresh installation for this version of the platform.

Recommended Hardware Specifications for Single Server DSP Orchestration

The system should be equipped with the following hardware specifications to support DSP orchestration, assuming DSP orchestration, DFM, Prometheus, Postgres, and Grafana are all running on the same machine.

- **CPU:** 40 cores
- **Memory:** 120 GB
- **Storage:** 500 GB (single disk)

  This ensures optimal performance across all integrated components.

Notice: For the specifications for a multi-server deployment, contact Drut Technical Support.

Third-party Hardware

The following is the list of third-party hardware (2, 4, or 8 base metal servers) applicable only for an entry-level installation:

- 2, 4 or 8 bare metal servers:
    - Full length, full height PCIe slot, out of band BMC management port, PXE boot port.
    - Please check with Drut for vendor compatibility.
- Appropriate set of PCIe Resource devices:
    - GPU, DPU, APU, NVMe, and so on.
    - Please check with Drut for vendor compatibility.
- Out of band management network.
- In band data network, as per your application needs.

Drut Hardware

The following hardware is required only for a complete Drut Type 1 installation and is dependent on the specific installation type:

- Drut iFIC 2500 (Initiator Fabric Interface Card) per Bare metal system.
- Drut PRU 2500 (Photonic Resource Unit chassis (1,2 or 4)).
- Drut tFIC 2500 (target Fabric Interface Card) per Resource Unit chassis.
- Drut Photonic Fabric (PXC, plus cabling).

Drut Software

The following Drut software is necessary:

- Drut Software Platform (DSP), version 3.2.0.
- DFM, version 1.3.x.

Hardware Setup

Ensure the following hardware is setup before proceeding with installing the Drut software:

- Connect your servers, control devices, and other networked hardware to switches or routers. Ensure correct ports are used based on your network design.
- Verify that your power supply can handle the total load of all devices in your datacenter environment. Ensure that devices are powered.

Chapter 3. Environment Setup

Follow these steps to set up the environment:

1. Install virtual environment dependencies. Although Ubuntu 20.04 comes with Python 3 pre-installed, you need to run the following command on the host machine's command line to install Python 3 in venv:

   ```
   sudo apt install python3 python3.8-venv python3-pip
   ```

2. Download the dsp_ansible package from `https://<software_download_path>` to the Ubuntu machine and execute the following command to unpack the source file:

   ```
   tar -xvf <source_file>.tar.gz
   ```

3. Change the directory to `dsp/` by executing the following command:

   ```
   cd dsp/
   ```

4. To install various components of DSP, you need to modify the installation YAML files. By default, after unpacking DSP, only example YAML files (`*.yaml.example`) are provided. These must be copied and renamed to the actual configuration files before proceeding with the installation..
   All the example `.yaml` files are available under the `dsp/vars` folder. Execute the following command in your terminal to rename and copy each `.example` file in the same directory, removing the `.example` extension.

   ```
   find ./vars -type f -name "*.example" -exec bash -c 'dest="./vars/$(basename
    "$0" .example)"; [ ! -e "$dest" ] && cp "$0" "$dest"' {} \;
   ```

5. To create the *dsp_env* environment and install the necessary required packages, execute the following script:

   ```
   bash ./create_env.sh
   ```

6. Activate the *dsp_env* environment by running the following command:

   ```
   source ~/dsp_env/bin/activate
   ```

   Important: You must activate the *dsp_env* environment each time you start a new shell. To exit the *dsp_env* environment, execute the following command:

   ```
   deactivate
   ```

Perform the Installing and Configuring DSP Orchestration *(on page 9)* procedure.

Chapter 4. Software Installation

Installing DSP Orchestration Software

Installing and Configuring DSP Orchestration

Ensure that you have required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

Perform the following steps to install and configure the DSP Orchestration software:

1. Update the *hosts* file to include the IP address of the Ubuntu machine under the `[all_in_one]` section as per the following scenarios:
    - If you are configuring the DSP Orchestration software in the same machine, update the *hosts* file as shown below:

```
# Define hosts for DSP Orchestration here
[all_in_one]
localhost ansible_user=<username> ansible_connection=local

[postgres:children]
all_in_one

[maas_region_controller:children]
all_in_one

[redis:children]
all_in_one

[rabbitmq:children]
all_in_one

[nfs:children]
all_in_one
```

    - If you are configuring the DSP Orchestration software on a different machine, update the *hosts* file as shown below:

```
# Define hosts for MAAS here
[all_in_one]
```



```
<other_host_ip_address> ansible_user=<username>

[postgres:children]
all_in_one

[maas_region_controller:children]
all_in_one

[redis:children]
all_in_one

[rabbitmq:children]
all_in_one

[nfs:children]
all_in_one
```

Note: Ensure that the host machine has access to the SSH keys.

2. Update the variable values in the `vars/dsp-orc.yaml` configuration file as per the following table:

```
---
#DSP
dsp_orc_snap_path: "<DRUT_MAAS_SNAP_HTTP_URL_OR_LOCAL_PATH>"
dsp_username: drut
dsp_password: drut
```

Variable Declaration	Description
`dsp_orc_s-nap_path`	Provide the absolute path to the DSP Orchestration snapshot.
`dsp_user-name`	By default, the administrator username is set to *drut*. However, you can modify it to any preferred value based on your requirements.
`dsp_pass-word`	By default, the administrator password is set to *drut*. However, you can modify it to any preferred value based on your requirements.

3. To install DSP Orchestration and other necessary packages, run the following command:


```
ansible-playbook -i hosts playbooks/setup_all.yaml -e @./vars/dsp-orc.yaml
```

This command installs the following software packages along with the login credentials and is displayed on the screen.

- MaaS
- PostgreSQL
  - MaaS Database
  - DFM Database
  - AI Workbench Database
- Redis
- Rabbitmq
  - Rabbitmq - DFM user
  - Rabbitmq - Drut Workbench user
- NFS

4. After the installation is complete, update the DSP Orchestration SSH key with the SSH key of the ansible host that is running.

You can access the DSP GUI at `https://<x.x.x.x>:5443/DRUT/r`. where `<x.x.x.x>` represents the IP address of the host where the DSP Orchestration software is installed.

Important: The default DSP orchestration widget is configured and populated in the **DSP Orchestration > Application** tab. The Prometheus, Grafana, and Loki applications are not configured for local DSP orchestration at this point in time.

Perform the Configuring DSP Orchestration and Network *(on page 11)* procedure.

Configuring DSP Physical Network

Configuring DSP Orchestration and Network

Ensure that you have the required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

The following installation instructions are based on the assumption that you are using a rack with one switch and ten servers.

Note: You can extend this configuration to a variety of network configurations based on your requirements.

You must configure the required networks to complete the configuration. The following table illustrates an example of a network configuration:

Table 1. Example: Sample Network Configuration

Network Name	VLAN ID	Subnet
PXE	Default (`0`)	`192.168.0.0/24`
Admin (ADMIN)	`10`	`192.168.10.0/24`
Public (PUB)	`20`	`192.168.20.0/24`
Internal (INT)	`30`	`192.168.30.0/24`
Out of Band (OOB)	`40`	`192.168.40.0/24`
Storage Data (SData)	`50`	`192.168.50.0/24`
Storage Cluster (SClust)	`60`	`192.168.60.0/24`
Compute (COM)	`70`	`192.168.70.0/24`

Note: You can choose to specify the network subnets even before deployment.

A network with the specified minimum required configurations is set up.

Perform the Notes on Configuring your External Top-of-Rack (TOR) Switch *(on page 12)* procedure.

Notes on Configuring your External Top-of-Rack (TOR) Switch

Perform the following steps to configure your external TOR switch in DSP Orchestration:

1. Create a VLAN interface for each network with a static IP, using `.1` addressing for each VLAN. For example, you can specify the subnet for ADMIN network as `192.168.10.1`.

   Tip: This IP address can route the traffic to the uplink (corporate) or between the VLANs.

2. Configure the VLAN tags for the ports where the cables are connected.
3. Ensure that all the networks have proper routing to reach the internet and the corporate network.
4. Update the firewall rule to reach the public/admin network in this switch:
   a. Log in to the corporate network firewall.
   b. Locate the existing firewall rule for accessing the public/admin network.

   c. Edit the firewall rule to allow access to the public/admin network in this switch.
   d. Save the changes to the firewall rule.
   e. Test the updated firewall rule to ensure connectivity to the public/admin network in this switch.
5. Assuming DSP Orchestration manages all the interfaces except OOB, make sure you enable DHCP for OOB.
6. If the switch does not handle DHCP, you will have to manually assign a static IP address for each VLAN in DSP Orchestration, using `.2` addressing for each VLAN. For example, you can specify the IP address as `192.168.10.2`.

Establishes VLAN interfaces with the specified static IPs to ensure proper VLAN tagging on switch ports, configure routing for network access, update firewall rules for connectivity, manage interfaces with MAAS, and enable DHCP for the OOB network. This setup should effectively route traffic between VLANs and provide necessary connectivity to the corporate network and internet.

Perform the Completing Post-installation Configurations *(on page 13)* procedure.

Completing Post-installation Configurations

Perform the following steps to configure DSP Orchestration:

1. Access the DSP GUI at `https://<x.x.x.x>:5443/DRUT/r` where `<x.x.x.x>` represents the host where the DSP Orchestration software is installed. You will be prompted to enter a username and password.
2. Enter the username and password that you specified during the Installing and Configuring DSP Orchestration *(on page 9)* process. Upon successful login, the **Welcome to Drut Software Platform (DSP)** page is displayed.

   Important: If you choose to proceed with the setup, follow the steps below. Alternatively, if you prefer to complete the setup later, click **Skip Setup** at the bottom of the page.

3. By default the **Region name** group is built based on the name of the host server on which DSP Orchestration is running. You can choose to change it to different name or leave it as is.
4. In the **Connectivity** group, set the **DNS forwarder** value to `8.8.8.8` to enable the resolution of domains not managed by the DSP Orchestration software and click **Save and continue**.

   Notice: If the switch does not handle DHCP, you will have to manually assign a static IP address for each VLAN in DSP Orchestration, using `.2` addressing for each VLAN.

For example, you can specify the IP address as `192.168.10.2`.

5. On the following screen, choose the necessary Ubuntu images or CentOS images as required and click **Continue**.

   Note: For uploading custom images, contact Drut Technical Support.

6. On the next page click **Finish setup**. The **SSH Keys for <USER>** page appears.
7. Open a terminal window and execute the following command to generate an ssh key:

   ```
   ssh-keygen
   ```

   Hit the Enter key twice. The ssh keys are created. Execute the following command to copy the ssh key:

   ```
   cat ~/.ssh/id_rsa.pub
   ```

8. In the DSP Orchestration portal, click **Upload** and paste the generated ssh key.
9. In DSP Orchestration, a "Space" is a logical grouping of subnets that helps control network access and communication between machines. To create a space for each network and assign it accordingly, perform the following steps:

   a. Select **Manage > Subnets** from the menu in the DSP Orchestration Portal. The **Subnets** page appears.

   b. In the **Subnets** page, click **Add**, and then select **Space**. The **Add Space** page appears.

   c. Enter an appropriate name for the space in the **Name** box and click **Add Space**. The space with the specified name is added to the list of spaces.

   The following table illustrates an example of the space name associated with each network:

Network Name	VLAN ID	Subnet	DSP Orchestration Space Name
PXE	Default (`0`)	`192.168.0.0/24`	`pxe-space`
Admin (ADMIN)	`10`	`192.168.10.0/24`	`admin-space`
Public (PUB)	`20`	`192.168.20.0/24`	`public-space`
Internal (INT)	`30`	`192.168.30.0/24`	`internal-space`

Network Name	VLAN ID	Subnet	DSP Orchestration Space Name
Out of Band (OOB)	`40`	`192.168.40.0/24`	`oob-space`
Storage Data (SData)	`50`	`192.168.50.0/24`	`sdata-space`
Storage Cluster (SClust)	`60`	`192.168.60.0/24`	`scluster-space`
Compute (COM)	`70`	`192.168.70.0/24`	`compute-space`

10. Choose the necessary VLAN and assign the newly created space as required.
11. Enable DHCP for the required networks.
12. Create PXE subnets and enable DHCP.
13. Setup the required configurations for the server such as:

    a. Ensure that you properly connect the network cables and enable the PXE boot sequence by default.

    b. Ensure efficient use of BMC manager in DFM by providing common user names and passwords for all BMCs.

    c. Power up all the servers.

Commissioning, Configuring, and Deploying the Servers

After some time, you will be able to view all the servers on the **Hardware > Machines** page in the DSP Orchestration portal.

The DSP Orchestration portal generates and displays random host names by default. However, you have the option to provide a more meaningful and useful host name if needed.

Perform the following steps to complete the configuration tasks:

1. Start commissioning each server.
2. Configure the network as required by following these steps:
    a. Add any additional VLANs that the server requires.
    b. Select the appropriate network and choose **Auto assign**.
3. Configure the storage as needed. If necessary, configure LVM storage.
4. Pool and tag the server to assign appropriate needs.
5. After commissioning and configuring the servers, you have the option to deploy any OS on the server or keep the server in a ready state to receive requests.

Registering Machines as KVM Hosts

When the server is in ready-state, execute the following steps:

1. Deploy OS by registering the machines as KVM hosts.
2. From the DSP Orchestration software, click **Hardware** and select **KVM**. All available KVM hosts are displayed on the KVM page under the **LXD** and **Virsh** tabs.
3. Select the required VM from the list of KVM hosts. The selected VM details are displayed on the screen.
4. Select the **Settings / KVM Host Settings** tab, apply the relevant tag to the host, and click **Save changes** in the DSP Orchestration portal.

Configuring the DSP Orchestration Cloud

Ensure that you have required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

Update the `dsp_config.yaml` configuration file to provide the generic credentials for creating Virtual Machines (VMs).

Update the `vars/dsp_config.yaml` configuration file to include the following DSP Orchestration cloud parameters:

```
---
# DSP config
dsp_url: "https://<ip_address>:<port_no>"
dsp_username: drut
dsp_password: drut
dsp_primary_network_space: public-api
skip_allocation: true
```

Variable Declaration	Description
`dsp_url`	Provide the IP address `https://<x.x.x.x>:<port>`where the DSP Orchestration software is installed.
`dsp_user-name`	Provide the username to log into the `dsp_url` portal.
`dsp_pass-word`	Provide the password to log into the `dsp_url` portal.

Variable Declaration	Description
`dsp_primary_network_space`	Provide a space name that has a logical grouping of subnets that helps control network access and communication between machines.
`skip_allocation`	To use a machine that already exists in the DSP Orchestration cloud, make sure that the machine is in ready state with the provided host name, and set the `skip_allocation` parameter to *true*. If you intend to create new VMs from `dsp_config.yaml` *(on page 16)*, set the `skip_allocation` parameter to *false*. To create Virtual Machines (VMs) on the go, you need to deploy the Operating System (OS) on a Bare Metal machine and register it as a Kernel Virtual Machine (KVM) host with specific tags.

The `vars/dsp_config.yaml` configuration file is updated with the specified credentials for creating Virtual Machines (VMs).

Perform the Setting-up the Monitoring and Logging Services *(on page 17)* procedure.

Setting-up the Monitoring and Logging Services

Ensure that you have the required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

Perform the following steps to install Drut's monitoring and logging functions:

1. Update the `vars/dsp_config.yaml` *(on page 16)* and `vars/monitor_services.yaml` configuration file parameters as follows:

```
---
# MONITOR SERVICE
monitor_default_machines_config:
  cpu_count: 4
  tags: "<TAG_NAME>" # SET KVM HOST TAG
  memory: 10240
  storage: sba:100
  interfaces: eth0:space=public-space
```



```
  pool: "DSP-Monitor"  # The pool will be created if it does not exist, or you can add a
 custom pool name.
# By default, the following in-built names are used. If you want to specify the
 Prometheus, Grafana,
# and Loki hostnames names according to your requirements, uncomment the relevant section
 and provide
# the necessary details.
# To Prometheus
# prometheus_machine:
#   - hostname: "DSP-prometheus"
#     ansible_group: prometheus

# To Grafana
# grafana_machine:
#   - hostname: "DSP-grafana"
#     ansible_group: grafana

# admin_user: "drut"
# admin_password: "drut"

# To Loki
# loki_machine:
#   - hostname: "DSP-loki"
#     ansible_group: loki
```

2. To install the monitoring and logging services, execute the following command:

```
ansible-playbook playbooks/setup_monitor_services.yaml -e @./vars/dsp_config.yaml -e
 @./vars/monitor_services.yaml
```

Tip: If you wish to install a specific server of interest, use the `-t <server_name>` option. For example:

```
ansible-playbook playbooks/setup_monitor_services.yaml -e @./vars/dsp_config.yaml
 -e @./vars/monitor_services.yaml -t prometheus,grafana,loki
```

The Prometheus, Grafana, and Loki applications are configured and the corresponding widgets are registered and populated in the **DSP Orchestration > Application** tab and a summary with login credentials are displayed on the screen.

Perform the Enabling Monitoring Services for DSP Orchestration *(on page 19)* procedure.

Enabling Monitoring Services for DSP Orchestration

Ensure that you have the required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

Perform the following steps to enable monitoring services in DSP Orchestration portal:

1. If you want to configure monitoring services on the same machine, update the *hosts* file as shown below:

```
# Define hosts for DSP Orchestration here
[all_in_one]
localhost ansible_connection=local  # <Other machine IP> or localhost
 ansible_connection=local

[postgres:children]
all_in_one

[maas_region_controller:children]
all_in_one

[redis:children]
all_in_one

[rabbitmq:children]
all_in_one

[nfs:children]
all_in_one
```

   If you want to configure the monitoring services on a different machine, update the *hosts* file as shown below:

```
# Define hosts for DSP Orchestration here
[all_in_one]
<other_host_ip_address> ansible_user=<username>

[postgres:children]
```



```
all_in_one


[maas_region_controller:children]
all_in_one


[redis:children]
all_in_one


[rabbitmq:children]
all_in_one


[nfs:children]
all_in_one
```

Note: Ensure that the host machine has access to the SSH keys.

2. Execute the following command if you want to enable monitoring services on a remote machine on which an instance of DSP Orchestration is running:

```
ansible-playbook -i hosts playbooks/enable_maas_monitoring.yaml -e
 @./vars/dsp_config.yaml
```

Important: It is important that you update the `vars/dsp_config.yaml` *(on page 16)* with the DSP Orchestration server details.

If custom names are defined in the `vars/monitor_services.yaml` file, execute the following command to apply the changes:

```
ansible-playbook -i hosts playbooks/enable_maas_monitoring.yaml -e
 @./vars/dsp_config.yaml -e @./vars/monitor_services.yaml
```

The Rabbitmq, Redis, and Postgresql services are configured and the corresponding widgets are updated with the monitoring services and is populated in the **DSP Orchestration > Application** tab.

Perform the Configuring Drut Containers *(on page 21)* procedure.

Enabling Monitoring Services for DSP Orchestration Using Tags (Optional)

1. To install only DSP Orchestration Software on either a local or a remote machine, execute the following command:

```
ansible-playbook -i hosts playbooks/enable_maas_monitoring.yaml -e @./vars/dsp-orc.yaml
 -t maas_postgres
```

2. To install only Drut Postgres and then install DFM database on either a local or a remote machine, execute the following command:

```
ansible-playbook -i hosts playbooks/enable_maas_monitoring.yaml -t drut_postgres
```

3. To install only Redis on either a local or a remote machine, execute the following command:

```
ansible-playbook -i hosts playbooks/enable_maas_monitoring.yaml -t redis
```

4. To install only RabbitMQ on either a local or a remote machine, execute the following command:

```
ansible-playbook -i hosts playbooks/enable_maas_monitoring.yaml -t rabbitmq
```

5. To install Drut Postgres with DFM database, RabbitMQ, and Redis together on either a local or a remote machine, execute the following command:

```
ansible-playbook -i hosts playbooks/enable_maas_monitoring.yaml -t
 rabbitmq,drut_postgres,redis
```

Installing Drut Containers

Configuring Drut Containers

Ensure that you have the required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

Perform the following steps to configure the Drut Containers software.

After setting up the parameters in the `vars/dsp_config.yaml` *(on page 16)* file, you need to update the `vars/k8s.yaml` configuration file to include the following Drut Containers configuration parameters:

```
---
# K8s

# This is used to create the load balancer and will use this IP to connect to the K8s master.
virtual_ip: <VIRTUAL_IP>

# This is the network interface participating in the negotiation of the virtual IP, e.g., eth0.
```



```
network_interface: <NETWORK_INTERFACE>
k8_ha_enabled: true

k8s_default_machines_config:
  cpu_count: 15
  tags: "<TAG_NAME>" # SET KVM HOST TAG
  memory: 32768
  storage: sba:100
  interfaces: eth0:space=public-api;eth1:space=admin-api
  pool: "DSP-K8" # The pool will be created if it does not exist, or you can add a custom pool
 name.

# If you want to specify the K8s machines host names, ansible group names, and CPU counts
 according to
# your requirements, uncomment this section and provide the necessary details.

# By default, the following in-built names are used. If you want to specify the
 DSP-kube-master1, DSP-kube-master2,
# and DSP-kube-masterN; and so on hostnames names according to your requirements, uncomment the
 relevant section and provide
# the necessary details.

# k8s_machines:
#   - hostname: "DSP-kube-master1"
#     ansible_group: masters
#     cpu_count: 4
#   - hostname: "DSP-kube-master2"
#     ansible_group: masters
#     cpu_count: 4
#   - hostname: "DSP-kube-master3"
#     ansible_group: masters
#     cpu_count: 4
#   - hostname: "DSP-kube-worker1"
#     ansible_group: workers
#   - hostname: "DSP-kube-worker2"
#     ansible_group: workers
```



```
#   - hostname: "DSP-kube-worker3"
#     ansible_group: workers
```

Important:

- **High Availability (HA)** is currently supported. You need at least three masters and multiple workers.
- Make sure to reserve the virtual IP address under DSP Orchestration.

The Drut Containers software is configured for creating clusters. The corresponding widget is registered and populated in the **DSP Orchestration > Application** tab.

Perform the Installing Drut Containers Clusters *(on page 23)* procedure.

Installing Drut Containers Clusters

Ensure that you have the required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

Perform the following steps to install the Drut Containers clusters.

After you configure the Drut Containers, execute the following script to install Drut Containers clusters:

```
ansible-playbook playbooks/setup_k8s.yaml -e @./vars/dsp_config.yaml -e @./vars/k8s.yaml
```

If custom names are defined in the `vars/monitor_services.yaml` file, execute the following command to apply the changes:

```
ansible-playbook playbooks/setup_k8s.yaml -e @./vars/dsp_config.yaml -e @./vars/k8s.yaml -e
 @./vars/monitor_services.yaml
```

The Drut Containers clusters are installed on the host machine. After the execution of the above command, a `k8s_hosts` file is generated by default, which serves as an inventory for DFM and VRM installation or upgrade purposes.

Here is a sample of what the contents of a `k8s_hosts *(on page 23)*` file might look like:

```
# Ansible managed
[masters]
drut-1-kube-master1  ansible_host=10.51.10.41
drut-1-kube-master2  ansible_host=10.51.10.46
drut-1-kube-master3  ansible_host=10.51.0.56
```



```
[workers]
drut-1-kube-worker1  ansible_host=10.51.0.57
drut-1-kube-worker2  ansible_host=10.51.0.58
drut-1-kube-worker3  ansible_host=10.51.0.3

[all:vars]
virtual_ip=10.51.0.21
k8_ha_enabled=True
pool=K8scluster
# Ansible managed
```

If you need to regenerate the `k8s_hosts` file any time, execute the following command:

```
ansible-playbook playbooks/generate_inv.yaml  -e @./vars/dsp_config.yaml -e @./vars/k8s.yaml -e type=k8s
```

The packages are configured and the corresponding widgets are registered and populated in the **DSP Orchestration > Application** tab and a summary with login credentials are displayed on the screen.

Perform the Installing DFM on Drut Containers Clusters *(on page 24)* procedure.

Installing DFM on Drut Containers Clusters

Ensure that you have the required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

Perform the following steps to install and manage DFM clusters:

1. Once the `k8s_hosts *(on page 23)*` file is generated, setup the parameters in the `vars/dsp_config.yaml *(on page 16)*` file, and update the `vars/fm.yaml *(on page 24)*` configuration file as illustrated below to install DFM on all hosts listed in the `k8s_hosts *(on page 23)*` file:

   ```
   ---
   # To FM
   drut_fm_url: "<DRUT_FM_HELM_PACKAGE_HTTP_URL_OR_LOCAL_PATH>"

   fm_db_host_ip: "<FM_DB_HOST_IP>" # DFM database Host IP
   fm_mq_host_ip: "<FM_MQ_HOST_IP>" # DFM RabbitMQ Host IP
   fm_redis_host_ip: "<FM_REDIS_HOST_IP>" # DFM Redis HOST IP
   nfs_host_ip: "<NFS_HOST_IP>" # DFM NFS Host IP
   ```



```
bmc_username: "<BMC_USER_NAME>" # Provide the BMC username
bmc_password: "<BMC_PASSWORD>" # Provide the BMC password
```

2. Execute the following command to install DFM:

   ```
   ansible-playbook -i k8s_hosts playbooks/install_fm.yaml -e @./vars/fm.yaml -e
    @./vars/dsp_config.yaml
   ```

   If custom names are defined in the `vars/monitor_services.yaml` file, execute the following command to apply the changes:

   ```
   ansible-playbook -i k8s_hosts playbooks/install_fm.yaml -e @./vars/fm.yaml -e
    @./vars/dsp_config.yaml -e @./vars/monitor_services.yaml
   ```

   DFM is installed on all the clusters listed in the `k8s_hosts` file. The corresponding widget is registered, the fabric details are pulled into the orchestration software, and is populated in the **DSP Orchestration > Application** tab and a summary with login credentials are displayed on the screen.
3. **Optional:** To uninstall DFM from the Drut Containers clusters, execute the following command:

   ```
   ansible-playbook -i k8s_hosts playbooks/uninstall_fm.yaml -e @./vars/fm.yaml
   ```

4. **Optional:** To uninstall DFM from the Drut Containers clusters and cleanup the database, execute the following command:

   ```
   ansible-playbook -i k8s_hosts playbooks/uninstall_fm.yaml -e @./vars/fm.yaml -e
    @./vars/dsp_config.yaml -e cleardb=true
   ```

   The corresponding widget is de-registered and removed from the **DSP Orchestration > Application** tab and a summary with login credentials are displayed on the screen.
5. **Optional:** To upgrade the version of DFM on the Drut Containers clusters update the `drut_fm_url` variable in the `fm.yaml` with the package version you are upgrading to, and then execute the following command:

   ```
   ansible-playbook -i k8s_hosts playbooks/upgrade_fm.yaml -e @./vars/fm.yaml
   ```

   Note: You will need a minimum of twice the number of cores that DFM is using.

   If custom names are defined in the `vars/monitor_services.yaml` file, execute the following command to apply the changes:

   ```
   ansible-playbook -i k8s_hosts playbooks/upgrade_fm.yaml -e @./vars/fm.yaml -e
    @./vars/monitor_services.yaml
   ```

Perform the Adding New Worker Nodes to a Drut Containers Cluster *(on page 26)* procedure.

Adding New Worker Nodes to a Drut Containers Cluster

Ensure that you have the required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

Perform the following steps to add worker nodes to the DFM clusters.

To add new worker nodes to a Drut Containers cluster, execute the following command to update the `k8s.yaml *(on page 21)*` configuration file with the new worker details:

```
ansible-playbook playbooks/add_k8s_worker.yaml -e @./vars/dsp_config.yaml -e @./vars/k8s.yaml
```

CAUTION: We recommend that you do not change or remove any existing machine details.

Installing DX-VRM

Introduction

DX introduces new features that allow for virtual disaggregation of GPU server resources, which enables the essential technology, known as "vPODs" or virtual PODs, enabling dynamic allocation of resources according to software workloads. It facilitates direct data transfer between application memory spaces within GPUs in the same machine or other machines without utilizing the CPU. This capability includes:

- Point-to-point link
- Reconfigurability
- Distribution across multiple machines

Components of DX-VRM

The DX-VRM software comprises of the following components:

vRM (Virtual Resource Manager)

This service configures and manages off -the-shelf resources for the DFM. vRM is as a Drut Container docker service running on the DFM server.

vRUH (Virtual Resource Unit Host):

vRUH is a machine (VM or bare metal), that hosts the vRUG service for each device. The vRUH can be deployed as a VM on the same server as DFM.

vRUG (Virtual Resource Unit Group)

Each vRUG is a collection of emulators for each off-the-shelf server, represented by virtual Drut components such as vPRU and vFIC. It consists of a series of Drut Container docker services that are installed and executed on the vRUH.

Prerequisites

After you have finished all the steps in the order presented here, you can move forward with the DX-VRM installation:

1. Installing and Configuring DSP Orchestration *(on page 9)*
2. Configuring DSP Orchestration and Network *(on page 11)*
3. Notes on Configuring your External Top-of-Rack (TOR) Switch *(on page 12)*
4. Completing Post-installation Configurations *(on page 13)*
5. Commissioning, Configuring, and Deploying the Servers *(on page 15)*
6. Registering Machines as KVM Hosts *(on page 16)*
7. Configuring the DSP Orchestration Cloud *(on page 16)*
8. Setting-up the Monitoring and Logging Services *(on page 17)*
9. Enabling Monitoring Services for DSP Orchestration *(on page 19)*
10. Configuring Drut Containers *(on page 21)*
11. Installing Drut Containers Clusters *(on page 23)*
12. Installing DFM on Drut Containers Clusters *(on page 24)*
13. Adding New Worker Nodes to a Drut Containers Cluster *(on page 26)*

Installing DX-VRM Service

Ensure that you have required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

Follow these steps to install DX-VRM services on the same Drut Containers where DFM is installed.

1. Once the `k8s_hosts` file is generated, update the `vars/vrm.yaml` configuration file as illustrated below to install DX-VRM on all hosts listed in the `k8s_hosts` file:

```
---# To VRM
vrm_helm_url: "<DRUT_VRM_HELM_PACKAGE_HTTP_URL_OR_LOCAL_PATH>"
vrm_namespace: "virtual-resource-manager"
REDIS_SERVER: "<DEFAULT_FM_REDIS_HOST_IP>" # Should be FM Redis server IP
vrm_deploy_time: 120
vrm_retry_deploy_interval: 30
set_default_zones: false
```



```
dxlite_zone_name: VRM
dxlite_rack_name: Rack1
```

Important:

If you set the `set_default_zones` value to *true*, the zone and rack names defined in the `dxlite_zone_name` and `dxlite_rack_name` variables will be created automatically in DFM.

If you set the `set_default_zones` value to *false*, the zone and rack names specified in the `dxlite_zone_name` and `dxlite_rack_name` variables are ignored. Instead, zone and rack names specified in the `vars/dx_lite.yaml` *(on page 29)* configuration file (as described in the Setting-up DX-VRM Machines *(on page 29)* procedure) will be used for creating zones and racks.

2. Execute the following command to install the DX-VRM on the Drut Containers where the DFM is installed:

```
ansible-playbook -i k8s_hosts playbooks/install_vrm.yaml -e @./vars/vrm.yaml -e
 @./vars/dsp_config.yaml
```

Note: The corresponding DX-VRM widget is registered and populated on the **DSP Orchestration's > Application** tab.

If custom names are defined in the `vars/monitor_services.yaml` file, execute the following command to apply the changes:

```
ansible-playbook -i k8s_hosts playbooks/install_vrm.yaml -e @./vars/vrm.yaml -e
 @./vars/dsp_config.yaml -e @./vars/monitor_services.yaml
```

Tip: If you want to uninstall the DX-VRM, execute the following command:

```
ansible-playbook -i k8s_hosts playbooks/uninstall_vrm.yaml -e @./vars/vrm.yaml
```

Note: The corresponding DX-VRM is de-registered and the widget is removed from the **DSP Orchestration's > Application** tab.

DX-VRM service will be installed in the same Drut Containers where DFM is installed and will be discoverable by DFM and a summary with login credentials are displayed on the screen.

Perform the Setting-up DX-VRM Machines *(on page 29)* procedure.

Setting-up DX-VRM Machines

Ensure that you have required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

Perform the following steps to install DX-VRM software:

1. Update the `vars/dsp_config.yaml (on page 16)` and `vars/dx_lite.yaml` configuration file parameters as illustrated below:

```
# VRM
vRM_URL: "https://<IPX.X.X.X:PORT>/api/virtual-resource-manager"
vRM_USERNAME: "admin"
vRM_PASSWORD: "admin"
vRM_CONFIG: "{{vRM_URL}}/config/"
vRM_VRUH: "{{vRM_URL}}/virtual-resource-unit-host/"
vRM_VRUG: "{{vRM_URL}}/virtual-resource-unit-group/"
vRM_TASK: "{{vRM_URL}}/task-response/"
Feasibility_Check_url: "{{vRM_VRUG}}check-feasibility/"


# VRUH
vRUH:
  - "S47"


# VRUG
vRUG:
  - hostname: "S51"
    resource_block_count: 2
    zone_fqgn: "Drut.VRM"
    rack_fqgn: "Drut.VRM.Rack1"
    # List of each VM details this list should match with resource_block_count and its
 optional
    vm_details:
      - hostname: ABC1
        cpu_count: 10
        memory: 102400
        storage: sda:50
```



```
      interfaces: eth0:space=public-api
    - hostname: ABC
      cpu_count: 10
      memory: 10240
      storage: sda:50
      interfaces: eth0:space=public-api
- hostname: "R2S16-bare-105"
  resource_block_count: 2
  zone_fqgn: "Drut.VRM"
  rack_fqgn: "Drut.VRM.Rack1"
#   vm_details:
#     - hostname: ABC
#       cpu_count: 100
#       memory: 1024000
#       storage: sda:5000
#       interfaces: eth0:space=public-space
#     - hostname: ABCD
#       cpu_count: 10
#       memory: 10240
#       storage: sda:50
#       interfaces: eth0:space=public-space
```

Tip: The `vRM_URL` information can be retrieved from the summary report generated by the `vars/vrm.yaml` configuration file, as described in the Installing DX-VRM Service *(on page 27)* procedure.

2. To install DX-VRM, run the following command:

```
ansible-playbook playbooks/setup_dx_lite_machines.yaml -e @./vars/dsp_config.yaml -e
 @./vars/dx_lite.yaml
```

Tip: If you want to delete DX-VRM machines, execute the following command:

```
# Run this script to delete a VRUG
ansible-playbook playbooks/delete_dx_lite_machines.yaml -e
 @./vars/dsp_config.yaml -e @./vars/dx_lite.yaml -e vrug_host=<HOSTNAME>


# Run this script to delete a VRUH HOST
```



```
ansible-playbook playbooks/delete_dx_lite_machines.yaml -e
 @./vars/dsp_config.yaml -e @./vars/dx_lite.yaml -e vrugh_host=<HOSTNAME>
```

Important: You will not be able to delete a VRUG or VRUH resource if the managers are in composed state.

The VRUG and VRUGH machine collections are deployed and are now available to VRM. These machines are ready for the deployment of their respective managers. Additionally, the managers will be integrated into the DFM, including the establishment of PCIe peer connections.

Installing Drut Workbench

Ensure that you have the required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder. You need to have the following components before proceeding with installing Drut Workbench

1. Install Postgres, NFS, and Rabbitmq by executing the following command:

   ```
   ansible-playbook -i hosts playbooks/setup_all.yaml -e @./vars/dsp-orc.yaml -t
    aiwb_postgres,aiwb_rabbitmq,aiwb_nfs
   ```

   Remember: You can skip this step if you have already executed the `setup_all.yaml` script.

2. Once the `k8s_hosts` file is generated, setup up the parameters in the `vars/dsp_config.yaml` *(on page 16)* file, and update the `vars/aiwb.yaml` configuration file as illustrated below to install Drut Workbench on all hosts listed in the `k8s_hosts` file:

   ```
   ---
   # To AI Workbench
   aiworkbench_url: "<DRUT_AI_WORKBENCH_HELM_URL>"
   aiwb_postgres_host: "<DB_HOST_IP>"
   aiwb_rabbitmq_ip: "<RABBITMQ_HOST_IP>"
   nfsserver_host: "<NFS_HOST_IP>"
   nexusrepo_registry_url: "<NEXUS_URL>"
   nexusrepo_rouser: "<NEXUS_RO_USER>"
   nexusrepo_ropassword: "<NEXUS_RO_PASSWORD>"
   nexusrepo_rwuser: "<NEXUS_RW_USER>"
   nexusrepo_rwpassword: "<NEXUS_RW_PASSWORD>"
   ```



```
nexusrepo_reponame: "<NEXUS_REPO_NAME>"
nexusrepo_path: workbench/workloads
enable_https: true # If set to true, you will be using the https protocol; else you will
 be using the http  protocol
```

3. Execute the following command to install the Drut Workbench on the Drut Containers using the `k8s_hosts` inventory file:

```
ansible-playbook -i k8s_hosts playbooks/install_aiworkbench.yaml -e @./vars/aiwb.yaml -e
 @./vars/dsp_config.yaml
```

Note: The corresponding Drut Workbench widget is registered and populated on the **DSP Orchestration's > Application** tab.

If custom names are defined in the `vars/monitor_services.yaml` file, execute the following command to apply the changes:

```
ansible-playbook -i k8s_hosts playbooks/install_aiworkbench.yaml -e
 @./vars/aiwb.yaml -e @./vars/dsp_config.yaml -e @./vars/monitor_services.yaml
```

Tip: If you want to uninstall the Drut Workbench, execute the following command:

```
ansible-playbook -i k8s_hosts playbooks/uninstall_aiworkbench.yaml -e
 @./vars/aiwb.yaml
```

If you want to uninstall the Drut Workbench and clean the database, execute the following command:

```
ansible-playbook -i k8s_hosts playbooks/uninstall_aiworkbench.yaml -e
 @./vars/aiwb.yaml -e cleardb=true
```

Note: The corresponding Drut Workbench is de-registered and the widget is removed from the **DSP Orchestration > Application** tab.

Configuring the VPods for Drut Workbench

Ensure that you have the required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

1. After setting up the parameters in the `vars/dsp_config.yaml` *(on page 16)* file, update the `vars/vpod.yaml` configuration file as illustrated below to enable OS deployment on the VPod machines:

```
---
distro: "ubuntu/jammy"
vpod_name: <VPOD_NAME>
aiwb_server: https://<AIWB_SERVICE_IP:PORT>
nexus_server: <NEXUS_SERVER_IP:PORT>
```

2. Execute the following command to configure the VPod machines for Drut Workbench:

```
ansible-playbook playbooks/setup_vpod_machines.yaml -e @./vars/dsp_config.yaml -e
 @./vars/vpod.yaml
```

The OS is deployed on the machines in the VPod and all the necessary Nvidia drivers, Docker, and Cuda toolkit is installed on the machines, and the VPod is made available for Drut Workbench.

Installing Drut Storage

The procedures in this section contain information on how to perform the following tasks for installing and configuring the Drut Storage software:

Installing Drut Storage Using Scripts

Ensure that you have the required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

Perform the following steps to install Drut Storage cluster:

1. Update the configuration file parameters of `dsp_config.yaml` *(on page 16)* and `vars/ceph.yaml` as follows:

```
---
# CEPH
# dashboard_password: Dsp@123 # Uncomment this line and insert a password of your choice
ceph_mon_ip_space: <DSP-NETWORK-SPACE-NAME> # Configurable ex: public-api
```



```
ceph_cluster_network: <STORAGE_NETWORK_SUB_NET_CIDR> # Subnet designated for cluster
 replication, recovery, and heartbeats (configurable).

# If the create_openstack_config value is set to true, a configuration file and a
 corresponding pool
# are created within the cluster. Additionally, Drut Storage serves as the backend for
 Drut Compute.

# If the create_openstack_config value is set to false, the configuration file and the
 corresponding
# pool can be created within the cluster at a later time as needed.
create_openstack_config: false

ceph_default_machines_config:
  cpu_count: 4
  tags: : "<TAG_NAME>" #SET KVM HOST TAG
  memory: 10240
  storage: sda:50,sdb:50,sdc:50,sdd:50
  interfaces: eth0:space=public-api;eth1:space=storage-data
  pool: "DSP-Storage"   # The pool will be created if it does not exist, or you can add a
 custom pool name.

# If you want to specify the ceph machines host names, ansible group names, and labels
 according to
# your requirements, uncomment this section and provide the necessary details.

# ceph_machines:
#   - hostname: "DSP-ceph-control1"
#     ansible_group: ceph-admin
#     labels: "['_admin', 'mon', 'mgr', 'monitoring']"
#   - hostname: "DSP-ceph-control2"
#     ansible_group: ceph-nodes
#     labels: "['mon', 'rgw', 'mgr', 'iscsi', 'osd']"
#   - hostname: "DSP-ceph-control3"
#     ansible_group: ceph-nodes
#     labels: "['mon', 'rgw', 'iscsi', 'osd']"
#   - hostname: "DSP-ceph-control4"
```



```
#     ansible_group: ceph-nodes
#     labels: "['mon', 'rgw', 'iscsi', 'osd']"
```

2. Execute the following command to install Drut Storage:

```
ansible-playbook playbooks/ceph_setup_cluster.yaml -e @./vars/dsp_config.yaml -e
 @./vars/ceph.yaml
```

   If the `create_openstack_config` value was set to *false* in the `vars/ceph.yaml` *(on page 33)* configuration file, execute the following commands in the specified order to create a configuration file and a corresponding pool within the cluster:

```
ansible-playbook playbooks/generate_inv.yaml  -e @./vars/dsp_config.yaml -e
 @./vars/ceph.yaml -e type=dsp_storage
```

```
ansible-playbook -i dsp_storage_hosts playbooks/setup_openstack_pools.yml -e
 @./vars/ceph.yaml
```

   The corresponding widget is registered and is populated in the **DSP Orchestration > Application** tab.

3. To update the `ceph_machines` section in the `vars/ceph.yaml` configuration file with the new host details, execute the following command:

```
ansible-playbook playbooks/ceph_addhost.yaml -e @./vars/dsp_config.yaml -e
 @./vars/ceph.yaml
```

Installing Drut Storage Manually as an Administrator

Ensure that you have the required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

Perform the following steps to manually install the Drut Storage as an administrator. Using this procedure, you can bootstrap an admin node, install the required packages, and add nodes to the cluster.

1. Make sure you have at least two interfaces:
   a. Drut Storage mon/admin network (PUB network).
   b. Drut Storage cluster network (SClust network).
2. Deploy the necessary machines from the DSP Orchestration page.

Bootstrapping an Admin Node

Ensure that you have the required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

Perform the following steps to bootstrap an admin node.

1. Execute the following commands in the order they are listed below:

```
sudo apt install -y curl
curl --silent --remote-name --location
 https://raw.githubusercontent.com/ceph/ceph/quincy/src/cephadm/cephadm
chmod +x cephadm
sudo ./cephadm add-repo --release quincy
sudo ./cephadm install
```

2. Execute the following command to bootstrap the admin node:

```
sudo cephadm bootstrap --mon-ip <admin_node_ip> --cluster-network <cluster_network>
```

   For example: `sudo cephadm bootstrap --mon-ip 10.52.0.2 --cluster-network 10.202.0.0/20`

3. Make sure to make a note of the `ceph.pub` key file path or create a backup for future reference.

Installing Required Packages on Other Machines

Ensure that you have the required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

Perform the following steps to install the required packages on other machines that you intend to include in the cluster:

1. Execute the following commands in the order they are listed below to install the required packages on the cluster machines:

```
sed -i 's/#PermitRootLogin prohibit-password/PermitRootLogin yes/g' /etc/ssh/sshd_config
 && service ssh restart
sudo apt-get update
sudo apt-get install ca-certificates curl gnupg lsb-release
sudo mkdir -m 0755 -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor
 -o /etc/apt/keyrings/docker.gpg
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
 https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo
 tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
sudo chmod a+r /etc/apt/keyrings/docker.gpg
```



```
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin
 docker-compose-plugin -y
sudo apt-get install ceph-common
```

2. **Optional:** Perform this step if you are using a custom or local image for installation. Execute the following commands in the order they are listed below:

```
scp -r fabricm@10.1.10.107:/etc/docker/certs.d /etc/docker/
sudo cp /etc/docker/certs.d/drutio\:443/ca.crt /usr/local/share/ca-certificates/
sudo update-ca-certificates
sudo systemctl restart docker
sudo docker login drutio:443
```

3. Copy the public key `ceph.pub` to each node by executing the following command:

```
ssh-copy-id -f -i /etc/ceph/ceph.pub root@<all_nodes>
```

Adding Nodes to the Cluster

Perform the following steps to add nodes to the cluster.

1. Login to the Drut Storage portal: `https://<admin_node_ip>:8443` using your credentials.
2. From the left-navigation pane, select **Dashboard > Cluster > Hosts**.
   The **Hosts List** tab appears in the content pane displaying all available hosts.
3. Click **Add** on the top-left corner of the content pane.
   The **Add Host** dialog appears.
4. Enter the node details as per the following table and click **Add Host**.

Table 2. Add Host Dialog Options

Option	Description
Hostname	Enter the host name.
Network address	Enter the host's IP address.
Labels	Add required labels to run relevant services on this host. Add labels such as `_admin`, `grafana`, `iscsi`, `mds`, `mgr`, `mon`, `nfs`, `osd`, `rbd`, `rbd-mirror`, or `rgwv`.
Maintenance Mode	Select this option to place the host in maintenance mode (stops all Drut Storage daemons on host). By default this option is *deselected*.

The node is added to the hosts list.

5. From the left-navigation pane, select **Dashboard > Cluster > Services**.
   The **Services** page appears in the content pane displaying all available services.

Installing Drut Compute Using Kolla Ansible

Ensure that you have the required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

Perform the following steps to install Drut Compute using Kolla Ansible platform:

1. Update the `/dsp_config.yaml *(on page 16)*` and `vars/openstack.yaml` configuration file parameters as follows:

   Important: Ensure that the `neutron_external_interface` and `network_interface` parameters are available under every `openstack_machines` section.

```
---
# If setup_external_prometheus is enabled, it will create a job in the external
 Prometheus to scrape data
# from the OpenStack cluster. You need to configure the following settings to use the
 external Prometheus.
dsp_monitor_enable: true
horizon_password: drut
default_openstack_settings: |
  enable_neutron_provider_networks: "yes"
  kolla_base_distro: "ubuntu"
  openstack_tag_suffix: ""
  kolla_internal_vip_address: "<VIP_for_openstack_HA>"
  enable_central_logging: "yes"
  enable_heat: "no"
  enable_cinder: "yes"
  enable_cyborg: "no"
  enable_grafana: "yes"
  enable_prometheus: "yes"
  enable_designate: "no"
  designate_backend: "bind9"
  designate_ns_record:
```



```
    - "ns1.drut.openstack.io"
  enable_redis: "no"
  # Ceph config
  # If you do not want to use Ceph, disable the configuration below.
  glance_backend_ceph: "yes"
  cinder_backend_ceph: "yes"
  nova_backend_ceph: "yes"
  # Glance
  ceph_glance_keyring: "ceph.client.dsp_drut_glance.keyring"
  ceph_glance_user: "dsp_drut_glance"
  ceph_glance_pool_name: "DSP_images"
  # Cinder
  ceph_cinder_keyring: "ceph.client.dsp_drut_cinder.keyring"
  ceph_cinder_user: "dsp_drut_cinder"
  ceph_cinder_pool_name: "DSP_volumes"
  ceph_cinder_backup_keyring: "ceph.client.dsp_drut_backup.keyring"
  ceph_cinder_backup_user: "dsp_drut_backup"
  ceph_cinder_backup_pool_name: "DSP_backups"
  # Nova
  ceph_nova_keyring: "ceph.client.dsp_drut_cinder.keyring"
  ceph_nova_user: "dsp_drut_cinder"
  ceph_nova_pool_name: "DSP_vms"
  enable_prometheus_openstack_exporter_external: "yes"
  # To Enable SSL/HTTPS
  kolla_enable_tls_internal: "yes"
  kolla_enable_tls_external: "yes"
  kolla_enable_tls_backend: "yes"
  kolla_copy_ca_into_containers: "yes"
  openstack_cacert: "/etc/ssl/certs/ca-certificates.crt"
  # Custom docker registry settings:
  docker_registry: "gcr.io"
  docker_namespace: "sampletesttrialproject/openstack.kolla"


openstack_default_machines_config:
  cpu_count: 16
  tags: "DSP-KVM"
  memory: 32768
```



```
  storage: sda:50
  interfaces: eth0:space=public-api;eth1:space=storage-data,mode=unconfigured
  pool: "DSP-OS"
  neutron_external_interface: eth1
  network_interface: eth0


openstack_machines:
  - hostname: "DSP-openstack-control1"
    ansible_group: control, network, storage
  - hostname: "DSP-openstack-control2"
    ansible_group: control, network, monitoring
  - hostname: "DSP-openstack-control3"
    ansible_group: monitoring, storage
  - hostname: "DSP-openstack-control4"
    ansible_group: compute
```

2. **Optional:** To use the monitoring and logging services, you need to setup the Setting-up the Monitoring and Logging Services *(on page 17)* functionality.
3. Execute the following script to install Drut Compute using Kolla Ansible:

   ```
   setup_openstack.sh
   ```

   The corresponding widget is registered and is populated in the **DSP Orchestration > Application** tab.
4. Execute the following script to add new host details in the `vars/openstack.yaml` under the `openstack_machines` section:

   ```
   add_openstack_node.sh
   ```


Chapter 5. Troubleshooting and Maintenance

Removing Packages, Services, and Databases When Installation Errors are Encountered

Execute the respective code snippets to address any installation errors you may have encountered:

```
#!/bin/bash

# To Stop PostgreSQL Service
sudo systemctl stop postgresql

# To Remove PostgreSQL packages
sudo apt remove postgresql-cl* -y --purge
sudo apt remove postgres* --purge -y

# To Remove RabbitMQ packages
sudo apt-get remove --purge rabbitmq-server

# To Remove Redis packages
sudo apt remove redis* -y
sudo apt remove libhiredis0.14:amd64 -y
sudo apt-get purge --auto-remove redis-server -y

# To Remove DSP Orchestration snap package
sudo snap remove maas --purge
```

Backing-up DSP Orchestration and DFM Databases

Perform the following steps to backup your DSP Orchestration and DFM databases.

1. Execute the following command to backup your DSP Orchestration database:

   ```
   ansible-playbook -i hosts playbooks/backups.yaml -e @./vars/maas_backup.yaml
   ```

   The ssh keys, netplan, and DSP Orchestration database (postgres mass database) are backed-up and the `vars/maas_backup.yaml` file is updated accordingly.
2. Execute the following command to backup your FM database:


```
ansible-playbook playbooks/fm_backup.yaml -e @./vars/fm_backup.yaml --extra-vars
 `fm_db_host=<Add_fm_db_host_IP>`
```

Important: By default, the `login_user` argument is set to `drut`. If the DFM database host is using a different `login_user` value other than the default value, then the command must be appended with `-e `login_user=<login_user_name>`` value.

The DFM database is backed-up and the `vars/fm_backup.yaml` file is updated accordingly.

Creating Drut Storage Users and Pools for Drut Compute

Perform the following steps to create Drut Storage users and pools for Drut Compute.

Execute the following commands in the order they are listed below:

```
ceph osd pool create volumes
ceph osd pool create images
ceph osd pool create backups
ceph osd pool create vms

rbd pool init volumes
rbd pool init images
rbd pool init backups
rbd pool init vms

ceph auth get-or-create client.glance mon `profile rbd` osd `profile rbd pool=images` mgr
 `profile rbd pool=images`
ceph auth get-or-create client.cinder mon `profile rbd` osd `profile rbd pool=volumes, profile
 rbd pool=vms, profile rbd-read-only pool=images` mgr `profile rbd pool=volumes, profile rbd
 pool=vms`
ceph auth get-or-create client.cinder-backup mon `profile rbd` osd `profile rbd pool=backups`
 mgr `profile rbd pool=backups`
```

Cleaning-up the DFM

Execute the following script to clean-up the DFM database:

```
clean_database.sh
```


Removing Drut Storage Host from the Cluster

Perform the following steps to remove a Drut Storage host from the cluster.

1. A host can be safely removed from the cluster, after all the daemons are removed from the host. To remove all daemons from the host, execute the following command:

   ```
   ceph orch host drain *<host>*
   ```

   The `_no_schedule` label is applied to the host. All OSDs on the host are scheduled for removal.
2. To monitor the progress of the OSD removal operation, execute the following command:

   ```
   ceph orch osd rm status
   ```
3. To verify if all the daemons are removed from the host, execute the following command:

   ```
   ceph orch ps <host>
   ```
4. After all the daemons are removed from the host, execute the following command to remove the host from the cluster:

   ```
   ceph orch host rm <host>
   ```

Removing an Existing Drut Compute Node or Host

Perform the following steps to remove an existing Drut Compute node or host.

1. For each host being removed, find Neutron routers on that host, move them, and disable the L3 agent by executing the following script:

   ```
   host=<remove_node_name>
   target_host=<target_node_name>

   source /etc/kolla/admin-openrc.sh

   l3_id=$(openstack network agent list --host $host --agent-type l3 -f value -c ID)
   target_l3_id=$(openstack network agent list --host $target_host --agent-type l3 -f value
    -c ID)

   echo $l3_id
   echo $target_l3_id

   openstack router list --agent $l3_id -f value -c ID | while read router; do
     openstack network agent remove router $l3_id $router --l3
   ```



```
  openstack network agent add router $target_l3_id $router --l3
done


openstack network agent set $l3_id --disable

dhcp_id=$(openstack network agent list --host $host --agent-type dhcp -f value -c ID)
target_dhcp_id=$(openstack network agent list --host $target_host --agent-type dhcp -f
 value -c ID)

echo $dhcp_id
echo $target_dhcp_id

openstack network list --agent $dhcp_id -f value -c ID | while read network; do
  openstack network agent remove network $dhcp_id $network --dhcp
  openstack network agent add network $target_dhcp_id $network --dhcp
done
```

2. Remove existing compute nodes by performing the following steps:

Remember: Before removing any compute nodes from the system, it is recommended that you either migrate or destroy any instances that they are hosting.

   a. For each host, disable the compute service to ensure that no new instances are scheduled to it by executing the following command:

```
openstack compute service set <host> nova-compute --disable
```

   b. To migrate live instances to another host, execute the following command:

```
openstack server list --all-projects --host <host> -f value -c ID | while read
 server; do
  openstack server migrate --live-migration $server
done
```

   Verify that the migrations were successful.

3. Stop all services running on the hosts being removed by executing the following command:

```
kolla-ansible -i openstack_hosts stop --yes-i-really-really-mean-it --limit <node_name>
```

4. Remove the hosts from the Ansible inventory (`openstack_hosts` and `vars/openstack.yaml`).

Note: All groups are listed as a comma-separated list in the `groups_names`column, before removing the host from the `openstack_hosts` file.

5. Execute the following command to reconfigure the remaining controllers to update the membership of clusters such as MariaDB and RabbitMQ.

```
kolla-ansible -i openstack_hosts reconfigure --limit <groups_names>
```

Tip: It is recommended to use a suitable limit, such as `--limit control`.

6. Execute the following script on each host to clean-up the services.

```
host=<remove_node_name>
openstack network agent list --host $host -f value -c ID | while read id; do
  openstack network agent delete $id
done

openstack compute service list --os-compute-api-version 2.53 --host $host -f value -c ID
 | while read id; do
  openstack compute service delete --os-compute-api-version 2.53 $id
done
```

Creating an External Flat Network on Drut Compute

Perform the following steps to create an external flat network on Drut Compute.

1. Login to the Drut Compute portal as admin user.
2. Select **Admin > Network > Networks** from the left-navigation menu.
   The Drut Compute portal's **Networks** content-pane is displayed listing all available networks.
3. In the **Networks** content-pane, click the **Create Network** button.
   The **Create Network** popup appears with the following tabs:
   - **Network**
   - **Subnet**
   - **Subnet Details**

   In the next steps, you will have to enter and select the minimum required fields for creating a flat external network.
4. In the **Create Network** popup, under the **Network** tab, specify the values of the fields as described in the following table, and click **Next** to proceed with the next steps:

Table 3. Networks > Create Network Popup > Network Tab Options

Field	Description
Name	Specify the name of the external network.
Project*	Select **Admin** option.
Provider Network Type*	Select **Flat** option.
Physical Network*	Enter **physnet1**.
Shared	Select this option.
External Network	Select this option.

5. In the **Create Network** popup, under the **Subnet** tab, specify the values of the fields as described in the following table, and click **Next** to proceed with the next steps:

Table 4. Networks > Create Network Popup > Subnet Tab Options

Field	Description
Subnet Name	Enter the subnet name.
Network Address	10.83.0.0/24 **Note:** The network address entered here must match with the `neutron_external_interface` address specified in the `vars/openstack.yaml` file.

6. In the **Create Network** popup, under the **Subnet** tab, specify the values of the fields as described in the following table, and click **Create** to complete the network creation process:

Table 5. Networks > Create Network Popup > Subnet Details Tab Options

Field	Description
Enable DHCP	Deselect this option.
Allocation Pools	Enter **10.83.0.150,10.83.0.250** based on the reservation in the DSP Orchestration for this network.
DNS Name Servers	Enter **8.8.8.8** or you can specify any of your corporate DNS servers.

Installing and Managing DFM Manually

Ensure that you have required environment as described in the Environment Setup *(on page 8)* procedure and make sure that you are in the `dsp/` folder.

Perform the following steps to install and manage DFM manually:

1. To install DFM *with NFS* server enabled, execute the following command:

```
helm install --name-template=fmrelease --namespace=fabric-manager --set
 global.fm_primary_db_host=<database machine IP>,global.fm_mq_host=<rabbitmq
 IP>,global.fm_redis_host=<redis machine IP>,logs.nfs.host=<nfs machine
 IP>,logs.nfs.path=/fabricm/logs,global.northbound_security.enabled=<true |
 false>,fm_optical_health_monitor_interval=60 <fabric-manager-x.x.x-xx.tgz>
```

2. After the installation is complete, execute the following commands in the order they are listed below to register the element service (internal service) with the DFM database:

```
kubectl exec -it fmrelease-fm-resource-manager-xxxx -n fabric-manager -- bash
```

```
curl -H "Content-Type: application/json"  -H "Authorization: Basic YWRtaW46YWRtaW4="
 -d'{"RemoteRedfishServiceUri" :"https://
fmrelease-fabric-element-service:19090/redfish/v1"}'  -X POST
 https://localhost:9811/redfish/v1/Managers
```

3. To install DFM *without NFS* server enabled, execute the following command:

```
helm install --namespace=fabric-manager fmrelease fabric-manager-<FM_RELEASE_VERSION>.tgz
 --set
global.fabricm_host=<FM_HOST>,logs.nfs.enabled=false,fm_optical_health_monitor_interval=60
,global.northbound_security.enabled=true
```

4. To view the installed version of DFM, execute the following command:

```
helm list -n fabric-manager
```

5. To upgrade to a new DFM version, execute the following command:

```
helm upgrade --debug fmrelease --namespace=fabric-manager --set
 global.fabricm_host=<FM_HOST>,logs.nfs.enabled=false,global.northbound_security.enabled=t
rue /tmp/fabric-manager-upgrade.tgz --force
```

6. To uninstall DFM, execute the following command:

```
helm uninstall fmrelease --namespace=fabric-manager
```

