JUNIPER NETWORKS | Engineering Simplicity

Juniper Paragon Automation 2.3.0 Installation and Upgrade Guide

Published
2025-11-02

RELEASE
2.3.0

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Juniper Paragon Automation 2.3.0 Installation and Upgrade Guide
2.3.0

The information in this document is current as of the date on the title page.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at https://support.juniper.net/support/eula/. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

About This Guide

Use this guide to install Paragon Automation on VMware ESXi 8.0 servers. This guide explains how to:

- Install and upgrade Paragon Automation.
- Update the base-OS
- Shut down and reboot the cluster.
- Repair and replace nodes.
- Back up and restore the configuration.

RELATED DOCUMENTATION

Paragon Automation User Guide

Paragon Automation Release Notes, Release 2.3.0

CHAPTER 1

Introduction

IN THIS CHAPTER

Paragon Automation Installation Overview

Juniper® Paragon™ Automation is a WAN automation solution that enables enterprise and service provider networks to meet the challenges posed by an increase in volume, velocity, and types of traffic. Paragon Automation delivers an experience-first and automation-driven network that provides a high-quality experience to network operators.

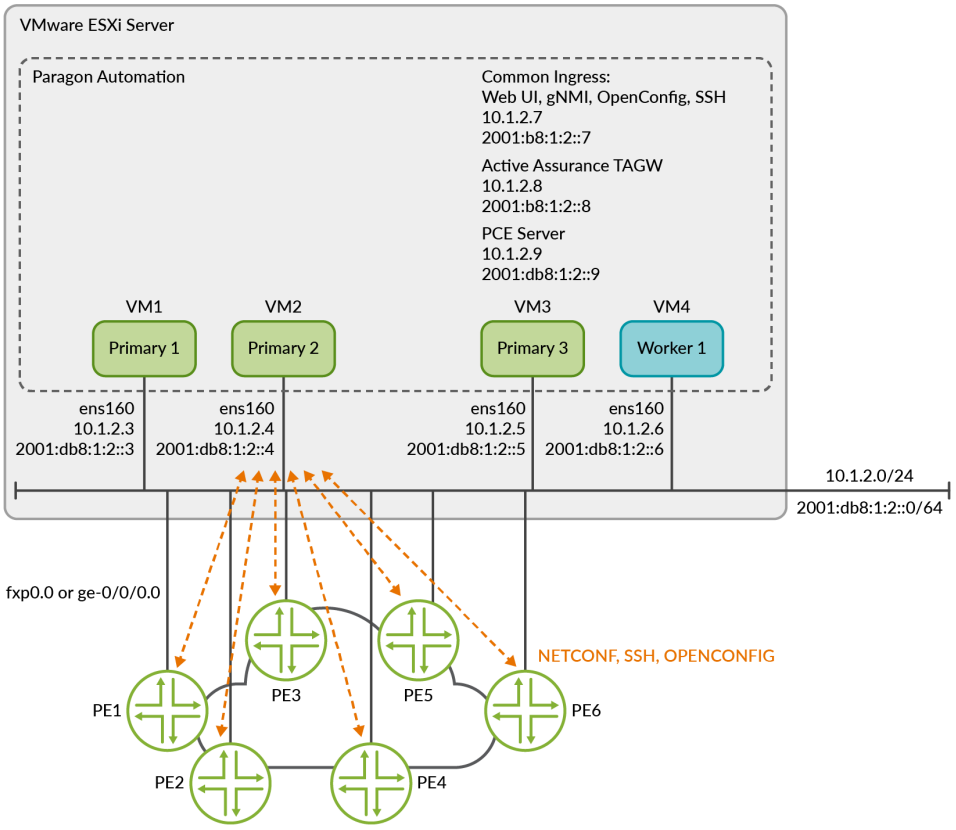
This guide describes how to install Paragon Automation and is intended for system administrators and network operators who install and manage the network infrastructure.

You deploy Paragon Automation as a set of on-premises (customer managed) nodes or virtual machines (VMs). A Kubernetes cluster is deployed inside these VMs during the installation of Paragon Automation. The Kubernetes cluster is a collection of microservices that interact with one another through APIs and that gets created automatically during Paragon Automation installation.

Inter-node communication within the cluster is implemented using APIs, and SSH, while the communication between Paragon Automation and the managed devices uses SSH, NETCONF, OpenConfig, and gNMI.

Figure 1 on page 3 shows a typical Paragon Automation cluster deployment along with communication protocols. While the illustration shows two servers, you can deploy the cluster on a single server as well.

Figure 1: Paragon Automation Deployment



Paragon Automation Installation

To install Paragon Automation:

1. Download the installation bundle to your local desktop. The installation bundle comprises an OVA file. Use the OVA directly or extract the OVF and **.vmdk** files to create your VMs.
2. Create and configure the VMs on ESXi 8.0 servers using the OVA (or OVF and **.vmdk**) bundle.
3. Deploy a Paragon Automation cluster on the VMs using the Paragon Shell CLI.
4. Log in to the Paragon Automation Web GUI.

An IT or system administrator with permissions to create VMs in the VMware ESXi Server installs and maintains the Paragon Automation cluster. The IT or system administrator is responsible for tasks that are related to installation and administration. Paragon Automation can be deployed in an air-gap environment where there is no access to the Internet.

You do not have to create the VMs and then use the OVA (or OVF and **.vmdk**) bundle. You will be creating the VMs from the OVA (or OVF and **.vmdk**) bundle. In other words, you do not need to install

the VMs with any particular operating system, deploy additional components such as Docker, create and configure the interfaces, configure NTP, and so on, separately. All these tasks are done automatically as part of the VM-creation process from the OVA (or OVF and **.vmdk**) bundle.

Paragon Shell CLI

Paragon Automation provides a custom containerized MGD (cMGD) user shell, called Paragon Shell. A system administrator can use Paragon Shell to deploy and configure the Paragon Automation cluster. The Paragon Shell CLI is installed and available after the VMs are created on the VMware ESXi Server using the OVA (or OVF and **.vmdk**) bundle. The software bundle is prepackaged with all the packages that are required to create the node VMs and deploy the Paragon Automation cluster. Paragon Shell is installed on the Linux base OS.

You can use Paragon Shell to:

- Deploy the Paragon Automation cluster.
- Upgrade, back up, restore, and edit the cluster configuration.
- Create and edit users.
- Configure monitoring with the collection of metrics from different types of sources and forward the collected data to designated sinks or destinations.
- Retrieve cluster information for troubleshooting.

VMs are appliances containing both the Linux base OS as well as all required application code. When you create and log in to the VMs, you are placed in Paragon Shell, by default. When you exit Paragon Shell, you are placed in the Linux root shell.

NOTE: Exercise caution while executing commands from the Linux root shell. Commands executed from the Linux root shell are not supported unless explicitly mentioned in the documentation.

The configuration files used to deploy the cluster are stored in the **/root/epic/config** folder on the VM from which you deployed the cluster.

This guide explains how to:

- Install and upgrade Paragon Automation.
- Shut down and reboot the cluster.
- Repair and replace nodes.
- Back up and restore a configuration.

RELATED DOCUMENTATION

2

CHAPTER

System Requirements

IN THIS CHAPTER

Paragon Automation Implementation

To determine the resources required to implement Paragon Automation, you must understand the fundamentals of the Paragon Automation underlying infrastructure.

Paragon Automation is a collection of microservices that interact with one another through APIs and run within containers in a Kubernetes cluster. A Kubernetes cluster is a set of nodes or virtual machines (VMs) running containerized applications.

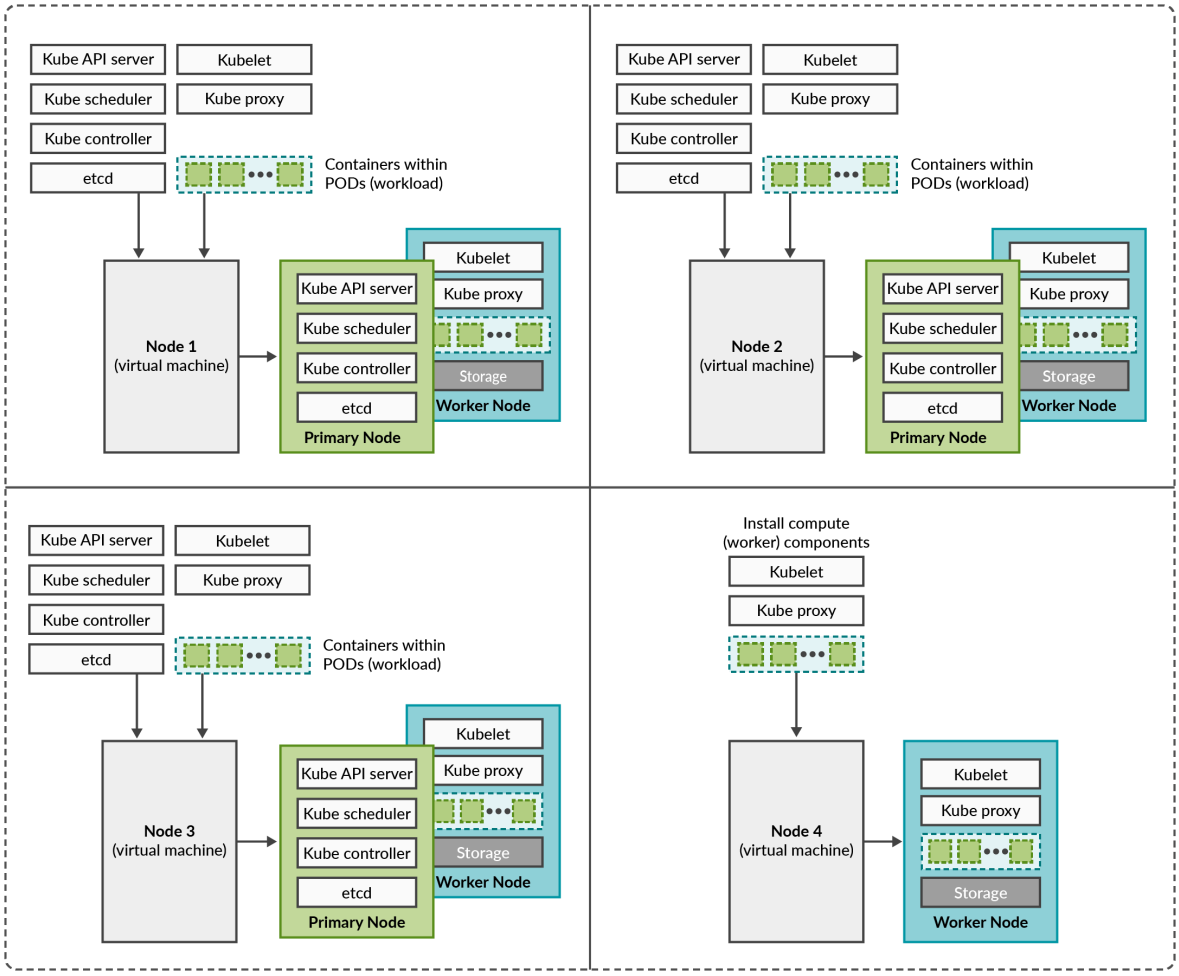
A Kubernetes cluster comprises one or more primary and worker nodes.

- **Control plane (primary) node**—The primary node performs the Kubernetes control-plane functions.
- **Compute (worker) node**—The worker node provides resources to run the pods. Worker nodes do not have control-plane function.

The two types of nodes can be deployed separately or co-located in the same VM. A single node can function as both primary and worker if the components required for both roles are installed in the same node.

In Paragon Automation, by default, the primary nodes also serve as worker nodes.

Figure 2: Kubernetes Cluster Nodes and Roles



You need to consider the intended system's capacity (number of devices to be managed, use cases, and so on), the level of availability required, and the expected system's performance, to determine the following cluster parameters:

- Total number of nodes in the cluster
- Amount of resources on each node (CPU, memory, and disk space)
- Number of nodes acting as primary and worker nodes

The amount of resources on each node are described later in this guide in "Paragon Automation System Requirements" on page 10.

Paragon Automation Implementation

Paragon Automation is implemented on top of a Kubernetes cluster, which consists of one or more primary nodes and one or more worker nodes. Paragon Automation is implemented as a multinode cluster. At minimum, three nodes that function as both primary and worker nodes and one node that functions as a worker-only node is required for a functional cluster.

NOTE: The four-node cluster is the recommended and supported implementation.

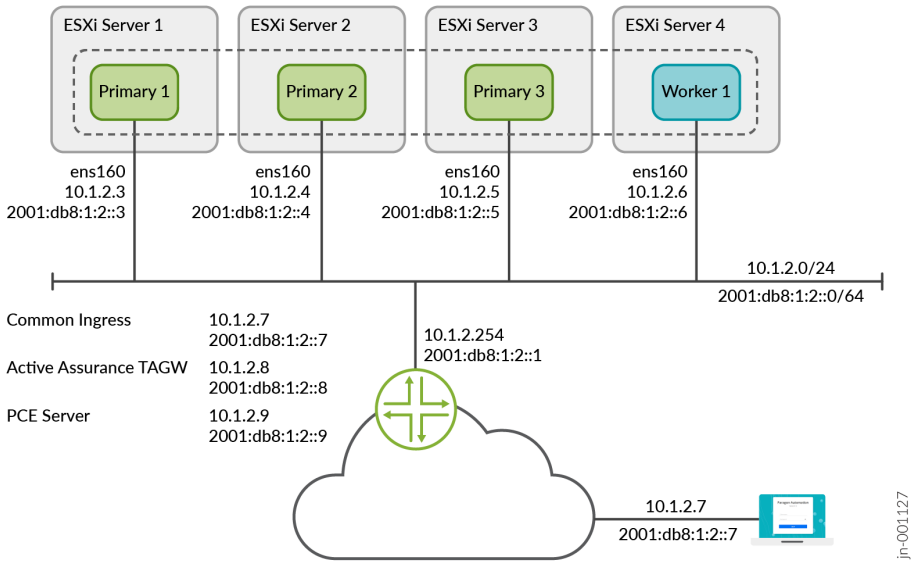
This implementation not only improves performance but allows for high availability within the cluster:

- **Control plane high availability**—The three nodes that function as both primary and worker nodes provide the required control plane redundancy. We do not support more than three primary nodes.
- **Workload high availability**—For workload high availability and workload performance, you must have more than one worker. In Paragon Automation, the three nodes that function as both primary and worker nodes and the one node that serves as a worker-only node provide the required workload high availability.
- **Storage high availability**—For storage high availability, all the nodes provide Ceph storage.

The Paragon Automation cluster remains functional when a single node fails and when the maximum latency between nodes is **less than 25 ms**.

To ensure that the cluster remains functional when a server fails, implement the cluster on four servers. The recommended implementation to maintain both server and node high availability is illustrated in Figure 3 on page 10. This ensures that the cluster remains functional even if one of the servers fail.

Figure 3: Server and Node High Availability



NOTE: In this release, single node failure and restart might cause inconsistencies in the device inventory.

RELATED DOCUMENTATION

Paragon Automation System Requirements | 10

Install Paragon Automation | 20

Paragon Automation System Requirements

IN THIS SECTION

Before you install the Paragon Automation software, ensure that your system meets the requirements that we describe in these sections.

Software Requirements

Use VMware ESXi 8.0 to deploy Paragon Automation.

Hardware Requirements

This section describes the minimum hardware resources that are required on each node VM in the Paragon Automation cluster, for evaluation purposes or for small deployments.

The compute, memory, and disk requirements of the cluster nodes can vary based on the intended capacity of the system. The intended capacity depends on the number of devices to be onboarded and monitored, types of sensors, and frequency of telemetry messages. If you increase the number of devices, you'll need higher CPU and memory capacities.

NOTE: To get a scale and size estimate of a production deployment and to discuss detailed dimensioning requirements, contact your Juniper Partner or Juniper Sales Representative. Juniper Paragon Automation Release 2.3.0 supports a scale of a maximum of 1000 devices.

Each of the four nodes in the cluster must have:

- 16-core vCPU
- 32-GB RAM
- 300-GB SSD

NOTE: SSDs are mandatory.

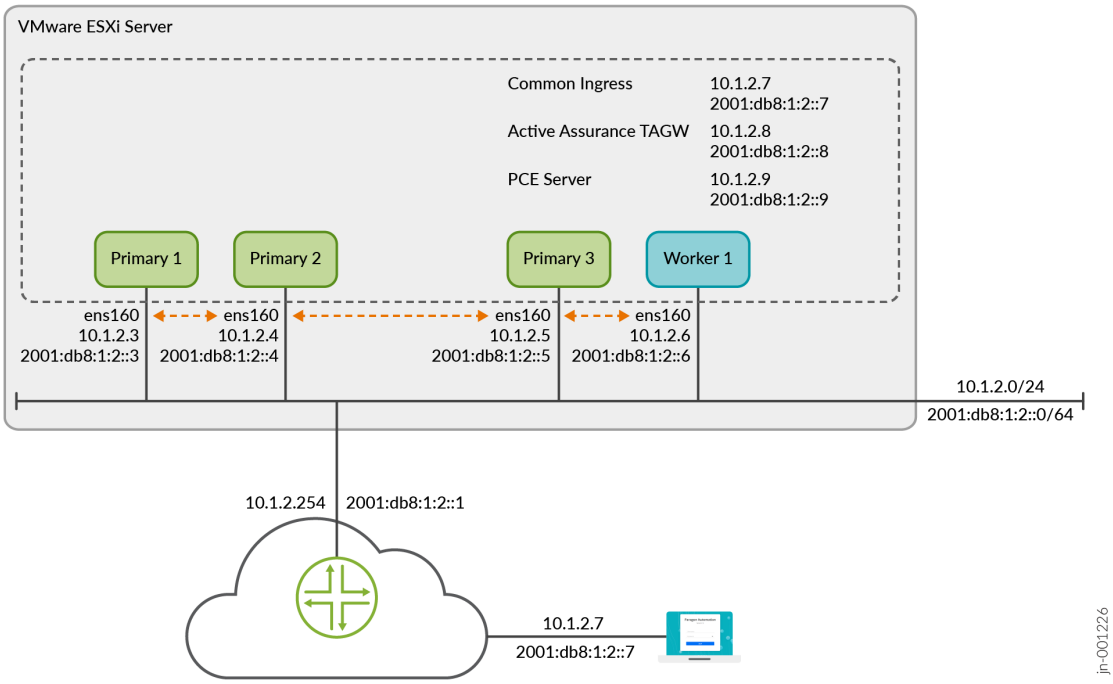
The VMs do not need to be on the same ESXi server, but need to be able to communicate over the same L2 network. The servers must have enough CPU, memory, and disk space to accommodate the hardware resources listed in this section. For node and server high-availability, deploy the four VMs on four servers.

Network Requirements

The four nodes must be able to communicate with each other through SSH. The nodes must be able to sync to an NTP server. SSH is enabled automatically during the VM creation, and you will be asked to enter the NTP server address during the cluster creation. Ensure that there is no firewall blocking NTP or blocking SSH traffic between the nodes in case they are on different servers.

Figure 4 on page 12 illustrates the IP addresses required to install Paragon Automation.

Figure 4: IP Addressing Requirements



You need to have the following addresses available for the installation, all in the *same subnet*.

- Four interface IP addresses, one for each of the four nodes
- Internet gateway IP address

- Three virtual IP (VIP) addresses for:
  - Generic ingress IP address shared between gNMI, OC-TERM (SSH connections from devices), and the Web GUI—This is a general-purpose VIP address that is shared between multiple services and used to access Paragon Automation from outside the cluster.
  - Paragon Active Assurance Test Agent gateway (TAGW)—This VIP address serves HTTP-based traffic to the Paragon Active Assurance Test Agent endpoint.
  - PCE server—This VIP address is used to establish Path Computational Element Protocol (PCEP) sessions between Paragon Automation and the devices. The PCE server VIP configuration is necessary to view dynamic topology updates in your network in real-time. For information on establishing BGP-LS peering and PCEP sessions, see Dynamic Topology Workflow.

  The VIP addresses are added to the outbound SSH configuration that is required for a device to establish a connection with Paragon Automation. The outbound SSH commands for OC-TERM and gNMI both use VIP addresses.
- Hostnames mapped to the VIP addresses—Along with VIP addresses, you can also enable devices to connect to Paragon Automation using hostnames. However, you must ensure that the hostnames and the VIP addresses are correctly mapped in the DNS and your device is able to connect to the DNS. If you configure Paragon Automation to use hostnames, the hostnames take precedence over VIP addresses and are added to the outbound SSH configuration used during onboarding devices.

Configure IPv6 Addresses

In this release, you can configure the Paragon Automation cluster using IPv6 addresses in addition to the existing IPv4 addresses. With IPv6 addressing configured, you can use IPv6 addresses for OC-TERM, gNMI, the Active Assurance TAGW,and access to the Web GUI. You must have the following additional addresses available at the time of installation:

- Four interface IPv6 addresses, one for each of the four nodes
- Internet gateway IPv6 address
- Two IPv6 VIP addresses for generic ingress and Active Assurance TAGW
- Hostnames mapped to the IPv6 VIP addresses—You can also use hostnames to connect to IPv6 addresses. You must ensure that the hostnames are mapped correctly in the DNS to resolve to the IPv6 addresses.

If hostnames are not configured and IPv6 addressing is enabled in the cluster, the IPv6 VIP addresses are added to the outbound SSH configuration, used for device onboarding, instead of IPv4 addresses.

NOTE: We do not support configuring an IPv6 address for the PCE server.

In addition to the listed IP addresses and hostnames, you need to have the following information available with you at the time of installation:

- Primary and secondary DNS server addresses for IPv4 and IPv6 (if needed)
- NTP server information

Figure 4 on page 12 illustrates the IP and VIP addresses required to install a Paragon Automation cluster.

Communication within and from outside of the cluster

You must allow intracluster communication between the nodes. In particular, you must keep the ports listed in Table 1 on page 14 open for communication.

Table 1: Ports That Firewalls Must Allow for Intracluster Communication

Port	Usage	From	To	Comments
Infrastructure Ports				
22	SSH for management	All cluster nodes	All cluster nodes	Require a password or SSH-key
2222 TCP	Paragon Shell configuration sync	All cluster nodes	All cluster nodes	Require password or SSH-key
443 TCP	HTTPS for registry	All cluster nodes	Primary nodes	Anonymous read access Write access is authenticated
2379 TCP	etcd client port	Primary nodes	Primary nodes	Certificate-based authentication

Table 1: Ports That Firewalls Must Allow for Intracluster Communication *(Continued)*

Port	Usage	From	To	Comments
2380 TCP	etcd peer port	Primary nodes	Primary nodes	Certificate-based authentication
5473	Calico CNI with Typha	All cluster nodes	All cluster nodes	—
6443	Kubernetes API	All cluster nodes	All cluster nodes	Certificate-based authentication
7472 TCP	MetalLB metric port	All cluster nodes	All cluster nodes	Anonymous read only, no write access
7946 UDP	MetalLB member election port	All cluster nodes	All cluster nodes	—
8443	HTTPS for registry data sync	Primary nodes	Primary nodes	Anonymous read access Write access is authenticated
9345	rke2-server	All cluster nodes	All cluster nodes	Token based authentication
10250	kubelet metrics	All cluster nodes	All cluster nodes	Standard Kubernetes authentication
10260	RKE2 cloud controller	All cluster nodes	All cluster nodes	Standard Kubernetes authentication

Table 1: Ports That Firewalls Must Allow for Intracluster Communication *(Continued)*

Port	Usage	From	To	Comments
32766 TCP	Kubernetes node check for PCE service local traffic policy	All cluster nodes	All cluster nodes	Read access only
Calico CNI Ports				
4789 UDP	Calico CNI with VXLAN	All cluster nodes	All cluster nodes	—
5473 TCP	Calico CNI with Typha	All cluster nodes	All cluster nodes	—
51820 UDP	Calico CNI with Wireguard	All cluster nodes	All cluster nodes	—

The following ports must be open for communication from outside the cluster.

Table 2: Ports That Firewalls Must Allow for Communication from Outside the Cluster

Port	Usage	From	To
179 TCP	Topology visualization and traffic engineering using the topology information	Paragon cluster node IP address	Router IP address to which you want to set up BGP peering from Paragon Automation. You can use the router management IP address or the router interface IP address.

Table 2: Ports That Firewalls Must Allow for Communication from Outside the Cluster *(Continued)*

Port	Usage	From	To
443	Web GUI + API	External user computer/desktop	Web GUI Ingress VIP address(es)
443	Paragon Active Assurance Test Agent	External network devices	Paragon Active Assurance Test Agent VIP address
2200	OC-TERM	External network devices	Web GUI Ingress VIP address(es)
4189	PCE Server	External network devices	PCE Server VIP address
6800	Paragon Active Assurance Test Agent	External network devices	Paragon Active Assurance Test Agent VIP address
32767	gNMI	External network devices	Web GUI Ingress VIP address(es)

Web Browser Requirements

The latest versions of Google Chrome, Mozilla Firefox, and Safari.

NOTE: We recommend that you use Google Chrome.

RELATED DOCUMENTATION

Paragon Automation Implementation | 7

CHAPTER 3

Install

IN THIS CHAPTER

Install Paragon Automation

SUMMARY

This topic describes the steps you must perform to install Paragon Automation using an OVA (or OVF and **.vmdk**) bundle.

IN THIS SECTION

You (system administrator) can install Paragon Automation by downloading an OVA bundle. Use the OVA bundle to deploy the node virtual machines (VMs) on one or many VMware ESXi servers. Alternatively, extract the OVF and **.vmdk** files from the OVA bundle and use the files to deploy the node VMs. Paragon Automation runs on a Kubernetes cluster with at least three primary and worker nodes and one worker-only node. The installation is air-gapped but you need Internet access to download the OVA bundle to your computer.

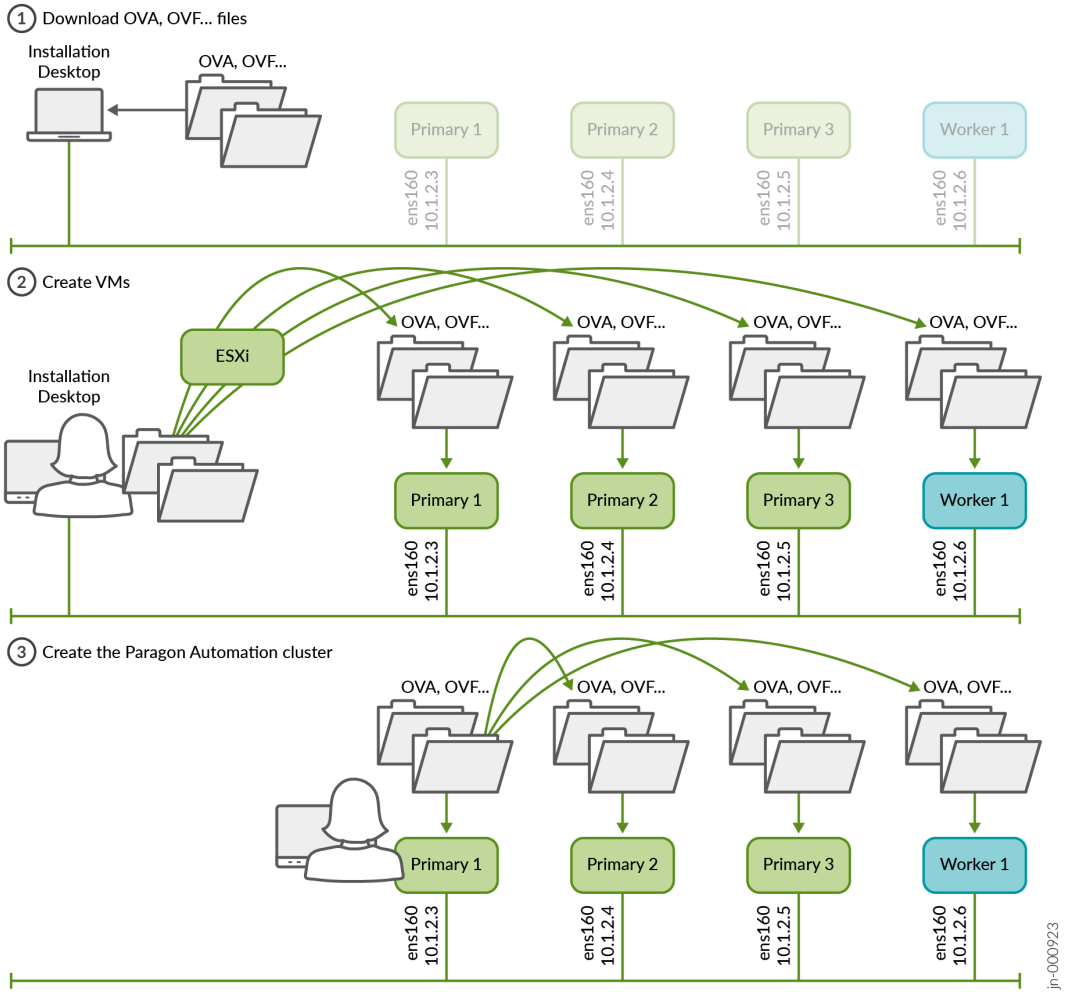
If you are installing from a remote machine, you can instead create a local desktop installer VM, either on the same server where you want to install Paragon Automation or on a different server. The local desktop installer VM can be a basic Ubuntu desktop VM. This avoids having to download the files into your remote machine, and then running the installation from the remote machine. This is described in the "Prepare the Nodes" on page 22 section.

To install Paragon Automation, you must create the node VMs using the downloaded OVA (or OVF and **.vmdk**) files. The software download files come prepackaged with the base OS and all packages required to create the VMs and deploy your Paragon Automation cluster. The VMs have Ubuntu 22.04.5 LTS (Jammy Jellyfish) Linux base OS.

Once the VMs are created, you must configure each VM in the same way. When all the VMs are configured and prepared, you deploy the Paragon Automation cluster. You can deploy the cluster using Paragon Shell.

Figure 5 on page 21 illustrates the Paragon Automation installation process at a high-level.

Figure 5: Installation Process



NOTE: You create the VMs directly using the OVA (or OVF and **.vmdk**) bundle. You do not need to create the VMs separately with any running software (for example, Ubuntu), or explicitly create interfaces, install Docker separately and so on. These are all automatically created and configured when you create the VMs using the OVA (or OVF and **.vmdk**) bundle.

NOTE: Release 2.3.0 is no longer available for download from the software download site. We recommend that you install release 2.4.1 or upgrade to release 2.4.1.

To install Juniper Paragon Automation Release 2.4.1 afresh, download the **paragon-2.4.1.9371.ova** file from the Juniper Paragon Automation software download site. Perform the steps described in the *Installation and Upgrade Guide* to install release 2.4.1 and log in to the Web GUI. See Install Paragon Automation for information.

To prepare the VMs, perform the following steps.

Prepare the Nodes

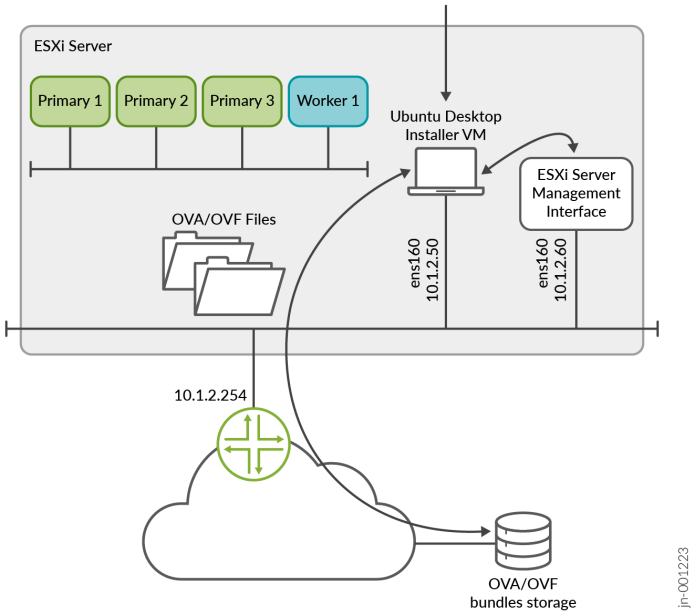
To prepare the nodes you must create and configure the VMs.

1. Download the **Paragon Automation Installation OVA** file from the Juniper Paragon Automation software download site. The OVA is used to create the node VMs and deploy your cluster.

   Note that the actual filename will include the release date in it, such as **paragon-2.3.0-*builddate*.ova**.

   The file is large in size, and it might take considerable time to download it and then create the VMs from your computer. So, we recommend that you create a local installer VM, which can be a basic Ubuntu desktop VM, either on the same server where you want to install Paragon Automation or on a different server. You must be able to download the OVA file to this local installer VM and you must have enough space on the VM to store the file. Configure connectivity to the management IP addresses of the servers as show in Figure 6 on page 23.

Figure 6: Local Installer VM to download the OVA/OVF files



2. (Optional) Validate the integrity of the OVA file. If you are using an Ubuntu desktop, use the following command:

```
root@ubuntu:~$ sha512sum paragon-2.3.0-builddate.ova
7deda68aae8ba6399aa95d5365a659a8d579c5562811ebe588972cf0c5107337628370d78dcbdb56ab8ea97e73b759
7f3a5ff06e9f501706bd8954b7454b86d2 paragon-2.3.0-builddate.ova
```

   Verify that the number displayed onscreen is the same as the SHA512 checksum number available on the Juniper Paragon Automation software download site. Click **Checksums** to view the valid SHA512 checksum.

3. After verifying the integrity, create the node VMs. You can use the OVA as it is to create the VMs.

   Alternatively, extract and use the OVF and **.vmdk** files from the OVA to create your VMs. To extract the files, use the following command:

```
tar -xvf paragon-2.3.0-builddate.ova
```

   If your installation desktop is running Windows, you can download and use the tar utility from https://gnuwin32.sourceforge.net/packages/gtar.htm to extract the files.

NOTE: If you are using a stand-alone ESXi 8.0 server without vCenter, due to a limitation of the VMware host client, you cannot upload large OVA files to the client. In such cases, you **must** extract and use the OVF, **.vmdk**, and **.nvram** files to create your VMs.

4. From your Web browser, connect and log in to the VMware ESXi 8.0 server where you will install Paragon Automation.

   If you are using a local installer VM, use the browser in the VM to connect to the VMware ESXi server.

5. **Create the node VMs.**

   Perform the following steps to create the VMs.

   a. Right-click the **Host** icon and select **Create/Register VM**.

      The New virtual machine wizard appears.

   b. On the Select creation type page, select **Deploy a virtual machine from an OVF or OVA file**.

      Click **Next**.

   c. On the Select OVF and VMDK files page, enter a name for the node VM.

      Click to upload or drag and drop the OVA file (or the OVF and **.vmdk** files).

      Review the list of files to be uploaded and click **Next**.

   d. On the Select storage page, select the appropriate datastore that can accommodate 300-GB SSD for the node VM. Note that SSD is mandatory.

      Click **Next**. The extraction of files takes a few minutes.

   e. On the Deployment options page:

      - Select the virtual network to which the node VM will be connected.
      - Select the **Thick** disk provisioning option.
      - Enable the VM to power on automatically.

      Click **Next**.

   f. On the Ready to complete page, review the VM settings.

      Click **Finish** to create the node VM.

> **NOTE**: If you used the OVF and **.vmdk** files to create your VMs and the VM creation failed, retry creating the VMs with the **.nvram** file. On step "5.c" on page 24, upload the **.nvram** file along with the OVF and **.vmdk** files. For standalone ESXi 8.0 servers without vCenter, you must upload the **.nvram** file as well.

g. To power on the VM, right-click the newly created VM on the Inventory page, and click **Power > Power on**.

h. Repeat steps "5.a" on page 24 through "5.g" on page 25 for the other three node VMs.

   Alternatively, if you are using VMware vCenter, you can right-click the VM, and click the **Clone > Clone to Virtual Machine** option to clone the newly created VM. Clone the VM three times to create the remaining node VMs.

   Enter appropriate VM names when prompted.

i. (Optional) Verify the progress of the VM creation in the Recent tasks section at the bottom of the page. When a VM is created, it appears in the VMware Host Client inventory under Virtual Machines.

j. When all the VMs have been created, verify that the VMs have the correct specifications and are powered on.

6. **Configure the node VMs.**

   When all the node VMs are created, perform the following steps to configure the VMs.

a. Connect to the node VM Web console of the first VM. You are logged in as root automatically.

b. You are prompted to change your password immediately. Enter and re-enter the new password. You are automatically logged out of the VM.

> **NOTE**: We recommend that you enter the same password for all the VMs. If you configure different passwords for the VMs, enter the different passwords correctly when requested to "Generate SSH keys" on page 33 for the cluster nodes when deploying the cluster.

c. When prompted, log in again as root user with the newly configured password.

d. Configure the following information when prompted.

Table 3: VM Configuration Wizard

Prompt	Action
Do you want to set up a Hostname? (y/n)	Enter **y** to configure a hostname.
Please specify the Hostname	Enter an identifying hostname for the VM. For example: Primary1. The hostname should be under 64 characters and can include alphanumeric and some special characters. If you do not enter a hostname, a default hostname in the format `controller-<VM-IP-address-4th-octet>` is assigned. **NOTE**: Since you are deploying the cluster from one node and entering the IP addresses of the other nodes during the cluster configuration process, the roles are assigned automatically. The first three nodes to be configured are the primary and worker nodes and the last node is the worker-only node. The hostnames (and whether or not they match the role of the node), will not affect the operations of the cluster. However, for management purposes, we recommend that you pay attention to how you name the nodes and the order in which you entered their addresses during the cluster creation steps. We do not support changing the hostname after the cluster has been installed.
Do you want to set up Static IP (preferred)? (y/n)	Enter **y** to configure an IP address for the VM.
Please specify the IP address in CIDR notation	Enter the IP address in the CIDR notation. For example, 10.1.2.3/24. **NOTE**: If you enter 10.1.2.3 instead of 10.1.2.3/24, you will see an Invalid IP address error message.
Please specify the Gateway IP	Enter the gateway IP address.

Table 3: VM Configuration Wizard *(Continued)*

Prompt	Action
Please specify the Primary DNS IP	Enter the primary DNS IP address.
Please specify the Secondary DNS IP	Enter the secondary DNS IP address.
Do you want to set up IPv6? (y/n)	Enter `y` to configure IPv6 addresses. If you don't want to configure IPv6 addresses, enter `n` and proceed to Step "6.e" on page 27.
Please specify the IPv6 address in CIDR notation	Enter the IPv6 address in the CIDR notation. For example, 2001:db8:1:2::3/64. **NOTE**: If you enter 2001:db8:1:2::3 instead of 2001:db8:1:2::3/64, you will see an Invalid IP address error message.
Please specify the Gateway IPv6	Enter the gateway IPv6 address.
Please specify the Primary DNS IPv6	Enter the primary DNS IPv6 address.
Please specify the Secondary DNS IPv6	Enter the secondary DNS IPv6 address.

e. When prompted if you are sure to proceed, review the information displayed, type **y** and press Enter.

f. You are logged into Paragon Shell.

g. Repeat steps "6.a" on page 25 through "6.f" on page 27 for the other three VMs.

7. (Optional) Verify connectivity between the nodes. Log in again to all the node VMs. If you have been logged out, log in again as root with the previously configured "password" on page 25. You are placed in Paragon Shell operational mode. Type `exit` to enter the Linux root shell of the nodes. Ping the other three nodes from each node using the `ping` *`static-ipv4-address`* command to verify that the nodes can connect to each other.

8. (Optional) Before you proceed to deploy the cluster, verify that the NTP server(s) is reachable. On any one of the cluster nodes, type `start shell`. At the `#` prompt, ping the server using the `ping` *`ntp-servers-name-or-address`* command. If the ping is unsuccessful, use an alternate NTP server.

You have completed the node preparation steps and are ready to deploy the cluster.

Deploy the Cluster Using Paragon Shell

Perform the following steps to deploy the Paragon Automation cluster using Paragon Shell CLI.

1. Go back to the first node VM (*Primary1*). If you have been logged out, log in again as root with the previously configured "password" on page 25. You are placed in Paragon Shell operational mode.

```
**********************************************************************
                WELCOME TO PARAGON SHELL!
     You will now be able to execute Paragon CLI commands!
**********************************************************************
root@eop>
```

2. To configure the cluster, enter the configuration mode in Paragon Shell.

```
root@eop> configure
Entering configuration mode

[edit]
```

3. Configure the following cluster parameters.

```
root@eop# set paragon cluster nodes kubernetes 1 address 10.1.2.3

[edit]
root@eop# set paragon cluster nodes kubernetes 2 address 10.1.2.4

[edit]
root@eop# set paragon cluster nodes kubernetes 3 address 10.1.2.5

[edit]
root@eop# set paragon cluster nodes kubernetes 4 address 10.1.2.6
```

```
[edit]
root@eop# set paragon cluster ntp ntp-servers pool.ntp.org

[edit]
root@eop# set paragon cluster common-services ingress ingress-vip 10.1.2.7

[edit]
root@eop# set paragon cluster applications active-assurance test-agent-gateway-vip 10.1.2.8

[edit]
root@eop# set paragon cluster applications web-ui web-admin-user "user-admin@juniper.net"

[edit]
root@eop# set paragon cluster applications web-ui web-admin-password Userpasswd

[edit]
```

Where:

The IP addresses of `kubernetes` nodes with indexes 1 through 4 must match the "static IP addresses" on page 26 that are configured on the node VMs. The Kubernetes nodes with indexes 1,2, and 3 are the primary and worker nodes, the node with index 4 is the worker-only node.

`ntp-servers` is the NTP server to synchronize to.

`web-admin-user` and `web-admin-password` are the e-mail address and password that the first user can use to log in to the Web GUI.

`ingress-vip` is the VIP address for generic common ingress and is used to connect to the Web GUI.

`test-agent-gateway-vip` is the VIP address for the Paragon Active Assurance Test Agent gateway (TAGW).

The VIP addresses are added to the outbound SSH configuration that is required for a device to establish a connection with Paragon Automation.

4. Configure the PCE server VIP address.

```
root@eop# set paragon cluster applications pathfinder pce-server pce-server-vip 10.1.2.9

[edit]
```

Where:

`pce-server-vip` is the VIP address that is used by the PCE server to establish Path Computational Element Protocol (PCEP) sessions between Paragon Automation and the devices.

NOTE: Configure the PCE server VIP address to view your network topology updates in real-time.

You can also configure the VIP address at any time post cluster deployment. For information on how to configure the PCE server VIP address after cluster deployment, see *Configure a PCE Server*.

5. (Optional) Configure IPv6 addresses.

```
root@eop# set paragon cluster kubernetes address-family cluster-ipv6-enabled true

[edit]
root@eop# set paragon cluster common-services ingress ingress-vip-ipv6 2001:db8:1:2::7

[edit]
root@eop# set paragon cluster applications active-assurance test-agent-gateway-vip-ipv6
2001:db8:1:2::8

[edit]
root@eop# set paragon cluster install prefer-ipv6 true

[edit]
```

Where:

`cluster-ipv6-enabled` enables usage of IPv6 addresses for the cluster making the cluster dual-stack.

`ingress-vip-ipv6` is the IPv6 VIP address for generic common ingress and is used to connect to the Web GUI.

`test-agent-gateway-vip-ipv6` is the IPv6 VIP address for the Active Assurance TAGW.

`prefer-ipv6` configures preference for IPv6 addresses over IPv4 addresses. When set to `true`, and if hostnames are not configured, IPv6 VIP addresses are added to the outbound SSH configuration.

6. If you want to configure hostnames for generic ingress and Paragon Active Assurance TAGW, configure the following:

```
root@eop# set paragon cluster common-services ingress system-hostname ingress-vip-dns-
hostname
```

```
[edit]
root@eop# set paragon cluster applications active-assurance test-agent-gateway-hostname
nginx-ingress-controller-hostname

[edit]
```

Where:

`system-hostname` is the hostname for the generic ingress virtual IP (VIP) address.

`test-agent-gateway-hostname` is the hostname for the Paragon Active Assurance TAGW VIP address.

When you configure hostnames, the hostnames take precedence over VIP addresses and are added to the outbound SSH configuration. The hostnames can resolve to either IPv4 or IPv6 VIP addresses or both.

7. (Optional) Configure the following settings for SMTP-based user management.

```
root@eop# set paragon cluster mail-server smtp-relayhost smtp.relayhost.com

[edit]
root@eop# set paragon cluster mail-server smtp-relayhost-username relayuser

[edit]

root@eop# set paragon cluster mail-server smtp-relayhost-password relaypassword
[edit]

root@eop# set paragon cluster mail-server smtp-allowed-sender-domains paragonautomation.net

[edit]
root@eop# set paragon cluster mail-server smtp-sender-email no-reply@paragonautomation.net
[edit]
root@eop# set paragon cluster mail-server smtp-sender-name Juniper Paragon Automation

[edit]
root@eop# set paragon cluster papi papi-local-user-management false

[edit]
root@eop# set paragon cluster mail-server smtp-enabled true

[edit]
```

Where:

`smtp-allowed-sender-domains` are the e-mail domains from which Paragon Automation sends e-mails to users.

`smtp-relayhost` is the name of the SMTP server that relays messages.

`smtp-relayhost-username` (optional) is the username to access the SMTP (relay) server.

`smtp-relayhost-password` (optional) is the password for the SMTP (relay) server.

`smtp-allowed-sender-domains` are the e-mail domains from which Paragon Automation sends e-mails to users.

`smtp-sender-email` is the e-mail address that appears as the sender's e-mail address to the e-mail recipient.

`smtp-sender-name` is the name that appears as the sender's name in the e-mails sent to users from Paragon Automation.

`papi-local-user-management` enables or disables local user management.

`mail-server smtp-enabled` enables or disables SMTP.

NOTE: SMTP configuration is optional at this point. SMTP settings can be configured after the cluster has been deployed also. For information on how to configure SMTP after cluster deployment, see *Configure SMTP Settings in Paragon Shell*.

8. (Optional) Install custom user certificates. Note, before you install user certificates, you must copy the custom certificate file and certificate key file to the Linux root shell of the node from which you are deploying the cluster. Copy the files to the **/root/epic/config** folder.

```
root@eop# set paragon cluster common-services ingress user-certificate use-user-certificate
true

[edit]
root@eop# set paragon cluster common-services ingress user-certificate user-certificate-
filename "certificate.cert.pem"

[edit]
root@eop# set paragon cluster common-services ingress user-certificate user-certificate-key-
filename "certificate.key.pem"
```

```
[edit]
```

Where:

`user-certificate-filename` is the user certificate filename.

`user-certificate-key-filename` is the user certificate key filename.

NOTE: Installing certificates is optional at this point. You can configure Paragon Automation to use custom user certificates after cluster deployment also. For information on how to install user certificates after cluster deployment, see "Install User Certificates" on page 42.

9. Commit the configuration and exit configuration mode.

```
root@eop# commit
commit complete

[edit]
root@eop# exit
Exiting configuration mode

root@eop>
```

10. Generate the configuration files.

```
root@eop> request paragon config
Paragon inventory file saved at /epic/config/inventory
Paragon config file saved at /epic/config/config.yml
```

The **inventory** file contains the IP addresses of the VMs.

The **config.yml** file contains minimum Paragon Automation cluster configuration parameters that are required to deploy a cluster.

The `request paragon config` command also generates a **config.cmgd** file in the **config** directory. The **config.cmgd** file contains all the `set` commands that you executed in "3" on page 28. If the **config.yml** file is inadvertently edited or corrupted, you can redeploy your cluster using the `load set config/config.cmgd` command in the configuration mode.

11. Generate SSH keys on the cluster nodes.

When prompted, enter the SSH password for the VMs. Enter the same "password" on page 25 that you configured to log in to the VMs.

```
root@eop> request paragon ssh-key
Setting up public key authentication for ['10.1.2.3','10.1.2.4','10.1.2.5','10.1.2.6']

Please enter SSH username for the node(s): root
Please enter SSH password for the node(s):
                        password

checking server reachability and ssh connectivity ...
Connectivity ok for 10.1.2.3
Connectivity ok for 10.1.2.4
Connectivity ok for 10.1.2.5
Connectivity ok for 10.1.2.6
SSH key pair generated in 10.1.2.3
SSH key pair generated in 10.1.2.4
SSH key pair generated in 10.1.2.5
SSH key pair generated in 10.1.2.6
copied from 10.1.2.3 to 10.1.2.3
copied from 10.1.2.3 to 10.1.2.4
copied from 10.1.2.3 to 10.1.2.5
copied from 10.1.2.3 to 10.1.2.6
copied from 10.1.2.4 to 10.1.2.3
copied from 10.1.2.4 to 10.1.2.4
copied from 10.1.2.4 to 10.1.2.5
copied from 10.1.2.4 to 10.1.2.6
copied from 10.1.2.5 to 10.1.2.3
copied from 10.1.2.5 to 10.1.2.4
copied from 10.1.2.5 to 10.1.2.5
copied from 10.1.2.5 to 10.1.2.6
copied from 10.1.2.6 to 10.1.2.3
copied from 10.1.2.6 to 10.1.2.4
copied from 10.1.2.6 to 10.1.2.5
copied from 10.1.2.6 to 10.1.2.6
```

NOTE: If you have configured different passwords for the VMs, ensure that you enter corresponding passwords when prompted.

12. Deploy the cluster.

```
root@eop> request paragon deploy cluster
Process running with PID: 231xx03
To track progress, run 'monitor start /epic/config/log'
After successful deployment, please exit Paragon-shell and then re-login to the host to
finalize the setup
```

The cluster deployment begins and takes over an hour to complete.

NOTE: If deployment does not start, check the contents of the **inventory** and **config.yml** files in the **/root/epic/config** folder in the Linux root shell. If the contents of the files are empty or incorrect, reboot the node using the `request paragon reboot type node` command. Once the node is back online, log in to rerun the `request paragon config` command from Paragon Shell and reattempt to deploy the cluster using `request paragon deploy cluster`.

13. (Optional) Monitor the progress of the deployment onscreen.

```
root@eop> monitor start /epic/config/log
```

The progress of the deployment is displayed. Deployment is complete when you see an output similar to this onscreen.

```
<output snipped>

PLAY RECAP *********************************************************************
10.1.2.3             : ok=1938 changed=828  unreachable=0    failed=0    rescued=0
ignored=4
10.1.2.4             : ok=185  changed=89   unreachable=0    failed=0    rescued=0
ignored=0
10.1.2.5             : ok=180  changed=88   unreachable=0    failed=0    rescued=0
ignored=0
10.1.2.6             : ok=185  changed=89   unreachable=0    failed=0    rescued=0
ignored=0

Thursday 19 September 2024  13:24:12 +0000 (0:00:00.586)       1:42:44.252 ****
===============================================================================
user-registry : Push Docker Images from local registry to paragon registry - 810.60s
```

```
kubernetes/addons/rook : Wait for Object-Store ----------------------- 403.28s
Install Helm Chart -------------------------------------------------- 201.88s
jcloud/airflow2 : Install Helm Chart -------------------------------- 186.19s
kubernetes/addons/postgres-operator : Make sure postgres is fully up and accepting request
using regular user - 114.11s
Check if kafka container is up -------------------------------------- 111.98s
Save installer config to configmap ---------------------------------- 98.89s
delete existing install config-map - if any ------------------------- 93.80s
Install Helm Chart -------------------------------------------------- 76.39s
kubernetes/multi-master-rke2 : start rke2 server on 1st master --------- 66.62s
systemd ------------------------------------------------------------- 65.02s
jcloud/papi : Install Helm Chart ------------------------------------ 64.71s
Wait until all node is ready ---------------------------------------- 62.86s
user-registry : Push Helm Charts to paragon registry ---------------- 57.56s
kubernetes/addons/metallb : Apply MetalLB configuration ------------- 57.12s
paa/timescaledb : Make sure postgres is fully up and accepting request using regular user
-- 57.11s
kubernetes/multi-master-rke2 : start rke2 server on other master ------- 57.07s
kubernetes/multi-master-rke2 : start rke2 server on other master ------- 49.41s
Create Kafka Topics ------------------------------------------------- 48.06s
kubernetes/addons/helper-commands : Install Pathfinder Utility scripts -- 45.21s
Playbook run took 0 days, 1 hours, 42 minutes, 44 seconds
registry-8228


root@eop>
```

Alternatively, if you did not choose to monitor the progress of the deployment onscreen using the `monitor` command, you can view the contents of the log file using the `file show /epic/config/log` command. The last few lines of the log file must look similar to "the sample output" on page 35. We recommend that you check the log file periodically to monitor the progress of the deployment.

14. Upon successful completion of the deployment, the application cluster is created. Log out of the VM and log in again to Paragon Shell.

    The console output displays the Paragon Shell welcome message and the IP addresses of the four nodes (called Controller-1 through Controller-4), the Paragon Active Assurance TAGW VIP address, the Web admin user e-mail address, and Web GUI IP address. If IPv6 addresses are configured, the welcome message displays the IPv6 VIP addresses as well.

```
Welcome to Juniper Paragon Automation OVA
```

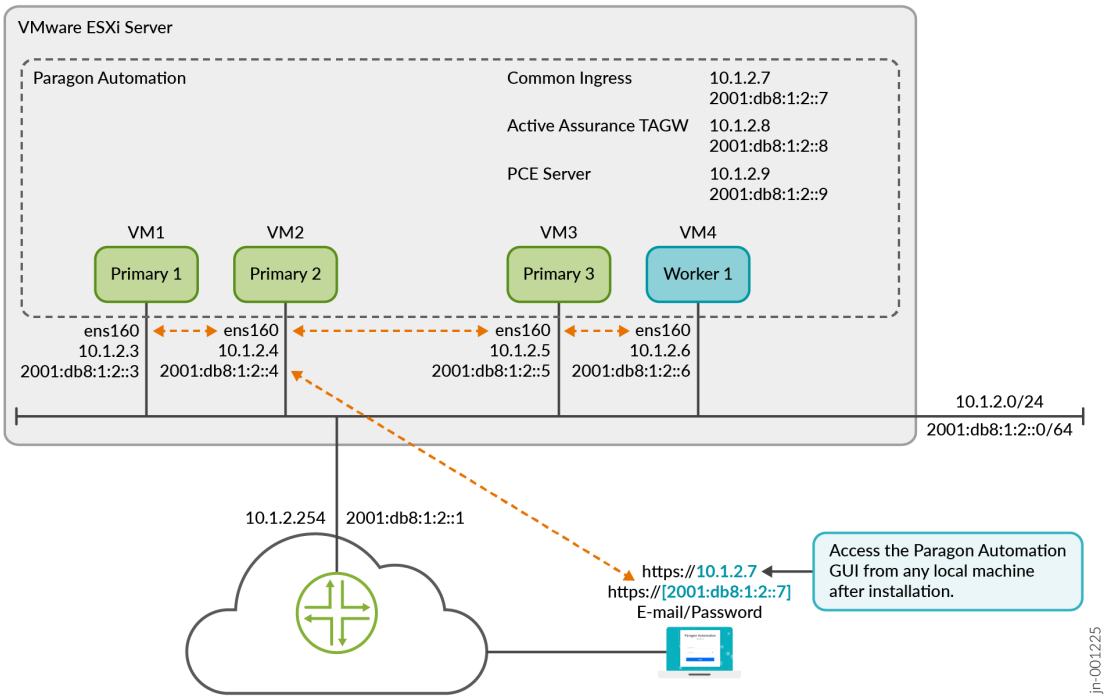
```
This Controller IP: 10.1.2.3, 2001:db8:1:2::3
This VM 10.1.2.3, 2001:db8:1:2::3 is part of an EPIC on-prem system with IPv6 enabled.
================================================================================================
Controller IP    :  10.1.2.3, 10.1.2.4, 10.1.2.5, 10.1.2.6
PAA Virtual IP   :  10.1.2.8, 2001:db8:1:2::8
UI               :  https://10.1.2.7, https://[2001:db8:1:2::7]
Web Admin User   :  admin-user@juniper.net
================================================================================================
ova: 20241017_1231
build: eop-release-2.3.0.8298.g4407b53b58

************************************************************
                WELCOME TO PARAGON SHELL!
     You will now be able to execute Paragon CLI commands!
************************************************************
root@Primary1>
```

The CLI command prompt displays your login username and the node hostname that you configured previously. For example, if you entered Primary1 as the hostname of your primary node, the command prompt is `root@Primary1 >`.

Figure 7 on page 38 illustrates a Paragon Automation cluster post-installation.

Figure 7: Paragon Automation Cluster



NOTE: If you are accessing the Web GUI from an external IP address, outside the Paragon Automation network, you must use NAT to map the external IP address to the Web GUI IP address.

You can now verify the cluster deployment and log in to the Web GUI. See "Log in to the Web GUI" on page 38.

Log in to the Web GUI

After the cluster has been deployed, you can verify the deployment (optionally), and log in to the Web GUI using the information you entered during the deployment.

1. (Optional) Verify the deployment.

a. Verify cluster details.

```
root@Primary1 > show paragon cluster details
Storage and Controller Node IPs: 10.1.2.3, 10.1.2.4, 10.1.2.5, 10.1.2.6
```

b. Verify cluster node details.

```
root@Primary1 > show paragon cluster nodes
NAME           STATUS   ROLES                       AGE     VERSION
Primary1   Ready    control-plane,etcd,master   4h12m   v1.31.1+rke2r1
Primary2   Ready    control-plane,etcd,master   4h12m   v1.31.1+rke2r1
Primary3   Ready    control-plane,etcd,master   4h11m   v1.31.1+rke2r1
Worker1    Ready    influxdb-worker,worker      4h11m   v1.31.1+rke2r1
```

2. Log in to the Web GUI.

a. Enter the common ingress VIP address in a Web browser to access the Paragon Automation login page. The common ingress IP address, that you configured during installation, can be either IPv4 or IPv6. The URLs to the UI is displayed on your console welcome message after the cluster deployment is complete.

To use the IPv4 address to connect to the Web GUI, enter **https://** ***ingress-vip*** in the URL. For example, **https://10.1.2.7**.

To use the IPv6 address to connect to the Web GUI, enter **https://[*ingress-vip-ipv6*]** in the URL. Ensure that you enclose the IPv6 address within square brackets. For example, **https://[2001:db8:1:2::7]**.

Alternatively, if you have configured hostnames, enter **https://** ***ingress-vip-dns-hostname*** to connect to the Web GUI.

b. Enter the Web admin user e-mail and password that you "configured" on page 28 previously to log in to Paragon Automation. The Web admin user e-mail is also displayed on your console after the cluster deployment is complete.

You are logged in to the Paragon Automation GUI and are directed to the New Account page from where you can create a new Organization. For more information, see *User Activation and Login*.

For a list of cluster-related tasks that you can perform using Paragon Shell post installation of the Paragon Automation cluster, see "Post Installation Tasks" on page 40.

SEE ALSO

Paragon Automation System Requirements | 10

Upgrade Paragon Automation | 45

Adopt a Device

Post Installation Tasks | 40

Post Installation Tasks

IN THIS SECTION

After you install the Paragon Automation cluster, you can use Paragon Shell to perform the following additional optional tasks:

- Ensure that the installed cluster is healthy and operational.

  Execute the `request paragon health-check` command. The `Overall Cluster Status` must be `GREEN`. For example:

```
root@primary1> request paragon health-check
Health status checking...

=======================================================
Get node count of Kubernetes cluster.
=======================================================

OK
There are 4 nodes in the cluster.
...
<output snipped>
...
=======================================================
Verifying Elasticsearch
=======================================================
```

```
OK
Opensearch test...
Checking health status at opensearch-cluster-master.common:9200...
Opensearch is healthy (green).
OPENSEARCH VERIFICATION PASS

=======================================================
Overall cluster status
=======================================================

GREEN
```

- Configure the PCE server. See *Configure PCE Server*.
- Configure SMTP-based user management. See *Configure SMTP Settings in Paragon Shell*.
- Install custom user certificates. See "Install User Certificates" on page 42.
- Back up your Paragon Automation cluster configuration. See "Back Up Using Paragon Shell" on page 79.
- Configure monitoring in Paragon Automation to collect metrics from different types of sources and forward the collected data to designated sinks. See Configure Monitoring.
- Update the Victoria Metrics cluster configuration parameters. See "Update Victoria Metrics" on page 41.

Update Victoria Metrics

Update the Victoria Metrics cluster configuration parameters. Victoria-metrics is a time series database. Different microservices populate the database and query data using the APIs provided by Victoria metrics.

1. Log in to the node from which you deployed the Paragon Automation cluster.
2. Type `configure` to enter configuration mode.
3. Update the Victoria Metrics parameters. Use the following commands.

   - `root@primary1# set paragon cluster insights victoria-metrics vm-cluster-replication-factor` *`replication_factor`*

- ```
  root@primary1# set paragon cluster insights victoria-metrics vm-cluster-retention-period retention_period
  ```

- ```
  root@primary1# set paragon cluster insights victoria-metrics vm-cluster-storage-instances number_of_pods
  ```

  For more information on `vm-cluster-replication-factor`, `vm-cluster-retention-period`, and `vm-cluster-storage-instances`, see *set paragon cluster insights victoria-metrics*.

4. Type `commit and-quit` to commit the configuration and exit configuration mode.
5. Regenerate the configuration files.

```
root@primary1> request paragon config
Paragon inventory file saved at /epic/config/inventory
Paragon config file saved at /epic/config/config.yml
```

6. Deploy the updated configuration.

```
root@primary1> request paragon deploy cluster input "-v -t victoriametrics-cluster"
Process running with PID: 23xx022
To track progress, run 'monitor start /epic/config/log'
```

RELATED DOCUMENTATION

> *Configure a PCE Server*
>
> *Configure SMTP Settings in Paragon Shell*

Install User Certificates

You can configure Paragon Automation to use custom user certificates. You can install user certificates either when you deploy the Paragon Automation cluster or post deployment.

To upload and install custom user certificates, perform the following steps:

1. Log in to the Linux root shell of the node from which you deployed the cluster.

2. Copy the custom user certificate file and key file to the **root/epic/config** directory.
3. Type `cli` to log in to Paragon Shell and type `configure` to enter configuration mode.
4. Configure the following parameters, commit, and exit the configuration mode.

```
root@Primary1# set paragon cluster common-services ingress user-certificate use-user-
certificate true

root@Primary1# set paragon cluster common-services ingress user-certificate user-certificate-
filename "certificate.cert.pem"

root@Primary1# set paragon cluster common-services ingress user-certificate user-certificate-
key-filename "certificate.key.pem"

root@Primary1# commit
commit complete

[edit]
root@Primary1# exit
Exiting configuration mode

root@Primary1>
```

Where:

certificate.cert.pem is the user certificate filename.

certificate.key.pem is the user certificate key filename.

5. Regenerate the configuration files.

```
root@Primary1> request paragon config
Paragon inventory file saved at /epic/config/inventory
Paragon config file saved at /epic/config/config.yml
```

6. Deploy the certificates.

```
root@Primary1> request paragon deploy cluster input  “-t  ingress-controller”
Process running with PID: 23xx022
To track progress, run 'monitor start /epic/config/log'
```

4

CHAPTER

Upgrade and Update

IN THIS CHAPTER

Upgrade Paragon Automation

IN THIS SECTION

You can upgrade your existing Juniper Paragon Automation Release 2.2.0 installation to Release 2.3.0 using Paragon Shell. The upgrade functionality enables you to upgrade your Paragon Automation installation and all the applications running on it.

We do not support upgrading from Juniper Paragon Automation Releases 2.0.0 and 2.1.0 to Release 2.3.0.

NOTE: Release 2.3.0 is no longer available for download from the software download site. We recommend that you install release 2.4.1 or upgrade to release 2.4.1.

If you want to upgrade from release 2.2.0, upgrade to release 2.4.1 by downloading the **upgrade_paragon-release-2.4.1.9371.g4407b53b58.tgz** file available on the software download site. See Upgrade Paragon Automation for information.

The upgrade process is automated by a set of Paragon Shell commands and carries out the required pre-upgrade system checks, retrieves the upgrade package, and executes the upgrade process on the cluster nodes. You can upgrade using an upgrade file that is either downloaded locally on your primary node or downloaded directly from a Web page.

During an upgrade, it is important that no change activities including onboarding of devices, provisioning of services or changing other configurations are done in the system. The upgrade will automatically reboot all components and there will be short unavailability during that time. The upgrade process does not affect the traffic through the network and once the upgrade is complete, the devices and services are not reconfigured.

We recommend that you back up your configuration before upgrading. For information on backing up your current configuration, see "Back Up and Restore Paragon Automation" on page 74.

To upgrade your Paragon Automation cluster:

1. Upgrade your installation and all the applications running on it using either the "local" on page 48 option or the "url" on page 51 option.
2. Upgrade "Paragon Shell and the OVA System Files" on page 54.

Prerequisites to the Upgrade Process

Before you upgrade the Paragon Automation cluster, ensure the following.

- Paragon Shell is accessible and operational.
- The cluster nodes have the following free disk space available:
  - The primary node from which the cluster was deployed must have 15% of the total disk space + three times the upgrade file size free.
  - The other two primary and worker nodes must have 15% of the total disk space + the same amount as the upgrade file size free.
  - The worker node must have 15% of the total disk space free.
- The cluster is healthy and operational. Execute the `# health-check` command from the Linux root shell. The `Overall Cluster Status` must be `GREEN`. For example:

```
root@primary1:~# health-check
Health status checking...

=======================================================
Get node count of Kubernetes cluster.
=======================================================

OK
There are 4 nodes in the cluster.

<output snipped>

=======================================================
Verifying Elasticsearch
=======================================================

OK
```

```
Opensearch test...
Checking health status at opensearch-cluster-master.common:9200...
Opensearch is healthy (green).
OPENSEARCH VERIFICATION PASS

=====================================================
Overall cluster status
=====================================================

GREEN
```

- Disable and delete previous OpenSearch backup files to free up space.

  1. Disable OpenSearch backup.

     ```
     # kubectl patch cronjob opensearch-backup-cron -n common -p '{"spec": {"suspend": true}}'
     ```

  2. Delete the periodic backup job.

     ```
     # kubectl delete job -n common -l app=opensearch-backup-cron
     ```

  3. Delete all existing OpenSearch backup files.

     ```
     # kubectl exec -i -n common -c opensearch-backup  $(kubectl get po -n common -l
     app=opensearch-backup -o jsonpath={.items[0].metadata.name}) -- bash -c 'rm -rf /opt/
     paragon/opensearch-backup/*'
     ```

- (Optional) Check the current build and OVA version of your existing setup from Paragon Shell.

  ```
  root@primary> show paragon version
  ova: 20240502_0230
  build: eop-release-2.2.0.8141.g0de84d1c80
  Client Version: v1.28.6+rke2r1
  Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
  Server Version: v1.29.3+rke2r1
  ```

Upgrade Paragon Automation using **either** the "local" on page 48 option or the "url" on page 51 option.

Upgrade using the `local` Option

Use this option for air-gapped environments where your Paragon Automation installation does not have access to the Internet. However, you need to be able to copy the **upgrade_paragon-release-*build-id*.tgz** and **upgrade_paragon-release-*build-id*.tgz.psig** files to your primary node.

1. Log in as root user to the primary node from which the current cluster was installed. You are logged in to Paragon Shell.
2. Type `exit` to exit from Paragon Shell to the Linux root shell.
3. Copy the **upgrade_paragon-release-*build-id*.tgz** and **upgrade_paragon-release-*build-id*.tgz.psig** files, of the version to which you want to upgrade, to the **/root/epic/temp** folder.

   You might need to download the **upgrade_paragon-release-*build-id*.tgz** and **upgrade_paragon-release-*build-id*.tgz.psig** files from the *Juniper Software Download* site to your local computer before copying it to the primary node.
4. (Optional) Use the `gpg --verify` command to validate the digital signature of the upgrade file. For example:

   ```
   root@primary1:~# gpg --verify upgrade_paragon-release-2.3.0.8213.g458486e9da.tgz.psig
   upgrade_paragon-release-2.3.0.8213.g458486e9da.tgz
   gpg: Signature made Tue Apr 23 01:00:09 2024 UTC
   gpg:                using RSA key 4B7B22C9C4FE32CF
   gpg: Good signature from "Northstar Paragon Automation 2024 ca@juniper.net" [ultimate]
   ```

   Here `primary1` is the installer primary node. Validation takes a couple of minutes to complete.
5. Type `cli` to enter Paragon Shell.
6. Use the following command to upgrade Paragon Automation.

   `request paragon cluster upgrade local filename upgrade_paragon-release-`*`build-id`*`.tgz`

   For example:

   ```
   root@primary1> request paragon cluster upgrade local filename upgrade_paragon-
   release-2.3.0.8213.g458486e9da.tgz
   Using local file /root/epic/temp/upgrade_paragon-release-2.3.0.8213.g458486e9da.tgz for
   upgrade
   Upgrade is in progress ...
   Updated to build: paragon-release-2.3.0.8213.g458486e9da
   Paragon Cluster upgrade is successful!
   ```

```
Please continue to primary host node to upgrade Paragon-shell and update OVA system files by:
/root/epic/upgrade_paragon-shell_ova-system.sh
```

Here `primary1` is the installer primary node.

Your Paragon Automation installation and all the applications running on it are upgraded.

Note that, the upgrade process takes a little over an hour to complete. Also, if you get disconnected from the VM during the upgrade process, you can periodically check the upgrade log file until you see an output similar to this:

```
root@primary1:~# cat /root/upgrade/upgrade.log
<output snipped>
…
PLAY RECAP *********************************************************************
10.1.2.3              : ok=1819 changed=430  unreachable=0    failed=0    rescued=0
ignored=2
10.1.2.4              : ok=185  changed=26   unreachable=0    failed=0    rescued=0
ignored=0
10.1.2.5              : ok=185  changed=26   unreachable=0    failed=0    rescued=0
ignored=0
10.1.2.6              : ok=177  changed=25   unreachable=0    failed=0    rescued=0
ignored=0

Saturday 03 December 2024  09:41:53 +0000 (0:00:00.665)       1:26:57.926 *******
===============================================================================
user-registry : Push Docker Images from local registry to paragon registry - 532.34s
jcloud/airflow2 : Install Helm Chart ---------------------------------- 278.28s
Install Helm Chart ---------------------------------------------------- 147.88s
delete existing install config-map - if any --------------------------- 111.87s
Save installer config to configmap ------------------------------------ 98.15s
jcloud/papi : Install Helm Chart -------------------------------------- 97.77s
Create Kafka Topics --------------------------------------------------- 79.97s
user-registry : Push Helm Charts to paragon registry ------------------ 78.70s
systemd --------------------------------------------------------------- 67.23s
kubernetes/addons/helper-commands : Install Pathfinder Utility scripts -- 44.65s
kubernetes/addons/helper-commands : Copy profiler to /opt/paragon/bin -- 39.79s
registry : Copy nginx image on 10.1.2.4 ------------------------------- 37.46s
registry : Copy nginx image on 10.1.2.5 ------------------------------- 37.04s
registry : Copy nginx image on 10.1.2.6 ------------------------------- 36.80s
registry : Copy nginx image on 10.1.2.3 ------------------------------- 36.03s
Install Helm Chart ---------------------------------------------------- 34.49s
registry : Copy zot image on 10.1.2.4 --------------------------------- 33.29s
```

```
registry : Copy zot image on 10.1.2.5 ---------------------------------- 32.46s
registry : Copy zot image on 10.1.2.6 ---------------------------------- 31.67s
registry : Copy zot image on 10.1.2.3 ---------------------------------- 30.25s
Playbook run took 0 days, 1 hours, 26 minutes, 57 seconds
registry-14272
Application Cluster upgraded to version build: paragon-release-2.3.0.8213.g458486e9da!!!
```

7. Ensure that the upgraded cluster is healthy and operational.

   Execute the `request paragon health-check` command. The `Overall Cluster Status` must be `GREEN`.

   For example:

```
root@primary1> request paragon health-check
Health status checking...

=========================================================
Get node count of Kubernetes cluster.
=========================================================

OK
There are 4 nodes in the cluster.
...
<output snipped>
...
=========================================================
Verifying Elasticsearch
=========================================================

OK
Opensearch test...
Checking health status at opensearch-cluster-master.common:9200...
Opensearch is healthy (green).
OPENSEARCH VERIFICATION PASS

=========================================================
Overall cluster status
=========================================================

GREEN
```

8. Upgrade Paragon Shell and the OVA system files. Go to "Upgrade Paragon Shell and the OVA System Files" on page 54.

Upgrade using the `url` Option

Use this option if your Paragon Automation installation has access to the Internet and the upgrade file is in a remote location.

1. Log in as root user to the primary node from which the current cluster was installed. You are logged in to Paragon Shell.
2. Use the following command to upgrade Paragon Automation.

   `request paragon cluster upgrade url "`*`https://juniper.software.download.site/upgrade_paragon-release-build-id.tgz?query_string`*`"`

   For example:

```
root@primary1> request paragon cluster upgrade url "https://cdn.juniper.net/software/paragon-
images/upgrade_paragon-release-2.3.0.8213.g458486e9da.tgz?query_string"
Upgrading paragon cluster from  https://cdn.juniper.net/software/paragon-images
Downloading tarball file        upgrade_paragon-release-2.3.0.8213.g458486e9da.tgz
Download file size: 19,526,900,113 bytes
Current disk Usage:
             Total: 263,622,004,736 bytes
              Used: 83,496,677,376 bytes
         Available: 168,297,881,600 bytes
Please wait for current download to finish... (File is large. It may take a while.)
Upgrade tarball file is downloaded.
Upgrade is in progress ...
Updated to build: paragon-release-2.3.0.8213.g458486e9da
Paragon Cluster upgrade is successful!
Please continue to primary host node to upgrade Paragon-shell and update OVA system files by:
/root/epic/upgrade_paragon-shell_ova-system.sh
```

   Here `primary1` is the installer primary node.

   Your Paragon Automation installation and all the applications running on it are upgraded.

Note that, the upgrade process takes a little over an hour to complete. Also, if you get disconnected from the VM during the upgrade process, you can periodically check the upgrade log file until you see an output similar to this:

```
root@primary1:~# cat /root/upgrade/upgrade.log
<output-snipped>
…
PLAY RECAP *********************************************************************
10.1.2.3              : ok=1908 changed=533  unreachable=0    failed=0    rescued=0
ignored=2
10.1.2.4              : ok=187  changed=31   unreachable=0    failed=0    rescued=0
ignored=0
10.1.2.5              : ok=187  changed=31   unreachable=0    failed=0    rescued=0
ignored=0
10.1.2.6              : ok=179  changed=29   unreachable=0    failed=0    rescued=0
ignored=0

Friday 27 November 2024  20:19:15 +0000 (0:00:00.295)       1:36:36.571 ******
===============================================================================
Create Kafka Topics ------------------------------------------------- 2866.24s
user-registry : Push Docker Images from local registry to paragon registry - 472.90s
Install Helm Chart -------------------------------------------------- 100.75s
Install Helm Chart --------------------------------------------------- 89.25s
jcloud/airflow2 : Install Helm Chart --------------------------------- 71.12s
systemd -------------------------------------------------------------- 63.44s
jcloud/papi : wait for papi rest api --------------------------------- 61.69s
delete existing install config-map - if any -------------------------- 51.48s
Save installer config to configmap ----------------------------------- 50.79s
user-registry : Push Helm Charts to paragon registry ----------------- 37.01s
jcloud/papi : Install Helm Chart ------------------------------------- 32.68s
Wait for common-utils to be running ---------------------------------- 28.40s
paragon-shell-config : Load paragon-shell initial configs on master node -- 26.93s
kubernetes/addons/helper-commands : Install Pathfinder Utility scripts -- 25.73s
Install Helm Chart -------------------------------------------------- 22.19s
command ------------------------------------------------------------- 21.97s
Wait for opensearch-backup to be running ----------------------------- 21.82s
kubernetes/addons/helper-commands : Copy profiler to /opt/paragon/bin -- 18.28s
systemcheck : Get Disk IOPS ------------------------------------------ 17.12s
kubernetes/addons/resource-reservation : Apply resource-reservation kube-cfg file -- 14.67s
Playbook run took 0 days, 1 hours, 36 minutes, 36 seconds
```

```
registry-8862
Application Cluster upgraded to version build: eop-release-2.3.0.8213.g458486e9da!!!
```

3. Ensure that the upgraded cluster is healthy and operational.

   Execute the `request paragon health-check` command. The `Overall Cluster Status` must be `GREEN`. For example:

```
root@primary1> request paragon health-check
Health status checking...

=========================================================
Get node count of Kubernetes cluster.
=========================================================

OK
There are 4 nodes in the cluster.
...
<output snipped>
...
=========================================================
Verifying Elasticsearch
=========================================================

OK
Opensearch test...
Checking health status at opensearch-cluster-master.common:9200...
Opensearch is healthy (green).
OPENSEARCH VERIFICATION PASS

=========================================================
Overall cluster status
=========================================================

GREEN
```

4. Upgrade Paragon Shell and the OVA system files. Go to "Upgrade Paragon Shell and the OVA System Files" on page 54.

Upgrade Paragon Shell and the OVA System Files

When your Paragon Automation installation and all the applications running on it are successfully upgraded, you must upgrade Paragon Shell and the OVA system files.

1. Exit from the installer primary node Paragon Shell to the Linux root shell by typing `exit`.

2. Execute the Paragon Shell upgrade shell script.

```
root@primary1:~# bash /root/epic/upgrade_paragon-shell_ova-system.sh
Upgrading paragon-shell...
Updating paragon-shell for primary1......
 Container paragon-shell  Stopping
 Container paragon-shell  Stopped
 Container paragon-shell  Removing
 Container paragon-shell  Removed
 paragon-shell Pulling
 paragon-shell Pulled
 Container paragon-shell  Creating
 Container paragon-shell  Created
 Container paragon-shell  Starting
 Container paragon-shell  Started
Updating paragon-shell for primary2......
 Container paragon-shell  Stopping

<output snipped>

primaryname    update-status
primary1       ok
primary3       ok
primary2       ok
primary4       ok
paragon-shell upgrade successful!
Updating OVA system files...
OVA system files update successful!
```

   Paragon Shell and the OVA system files are upgraded.

3. (Optional) Check the build and OVA version of your upgraded setup from Paragon Shell.

```
root@primary> show paragon versionova: 20241128_0439
ova-patch: 20241213_0349
```

```
build: eop-release-2.3.0.8213.g458486e9da
Client Version: v1.29.6
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
Server Version: v1.31.1+rke2r1
```

Now proceed to perform the post cluster upgrade tasks.

Post Cluster Upgrade Tasks

After upgrading the cluster and Paragon Shell OVA, perform the following tasks to complete the upgrade process.

1. Update the base OS. See "Update the OS" on page 55.
2. Upgrade the service designs, update the network implementation plan, and recreate the resource and service instances. See *Update the Network Implementation Plan and Recreate Service Instances After Upgrade*.
3. Restart all Active Assurance Monitors one by one. Wait until each restarted monitor is running and producing metrics before restarting the next monitor.

RELATED DOCUMENTATION

Update the OS | 55

Update the Network Implementation Plan and Recreate Service Instances After Upgrade

Install Paragon Automation | 20

Back Up and Restore Paragon Automation | 74

Update the OS

You create the node virtual machines (VMs) in a Paragon Automation cluster using the OVA (or OVF and **.vmdk**) software download files. The files are pre-packaged with all the utilities, OS, and software required to create the VMs. The VMs are created with Ubuntu 22.04.5 LTS (Jammy Jellyfish) Linux base OS. You might need to update the base OS to maintain the security, stability, performance, and compatibility of the Kubernetes cluster. Juniper provides you with the required OS update file to enable

you to update the OS on your node VMs. The OS update functionality in Paragon Automation includes the following updates:

- Linux kernel update
- OpenSSL or OS security update
- Any third-party packages required by Paragon Automation
- All packages that are part of the base OS

To update the OS, perform the following steps:

1. Download the **paragon-ubuntu-22-04-update-*date*.tar.gz** and **paragon-ubuntu-22-04-update-*date*.tar.gz.psig** files from the Juniper Software Download site on to your computer or local installer VM.
2. Log in to any one of the Paragon Automation cluster nodes as the root user.
3. Type `exit` to exit Paragon Shell to the Linux root shell.
4. Copy the **paragon-ubuntu-22-04-update-*date*.tar.gz** and **paragon-ubuntu-22-04-update-*date*.tar.gz.psig** files to the cluster node.
5. Update the OS on the current node using the copied files.

   ```
   root@node# update-os paragon-ubuntu-22-04-update-date.tar.gz paragon-ubuntu-22-04-update-date.tar.gz.psig
   ```

   The process takes around 15-30 minutes to complete depending on the size of the updated packages. Once complete, the OS on the other three nodes of the cluster is also automatically updated.

Verify the OS update

You can perform any of the following steps to verify that the OS update process is successful, and the cluster operation is unaffected.

- The Paragon Automation cluster remains operational while updating the OS. To verify if the update process has succeeded, check the **/var/log/apt/history.log** log file to see the timestamp of the last update and updated packages. For example:

  ```
  Start-Date: 2024-09-18  15:50:57

  Commandline: apt upgrade

  Install: ubuntu-pro-client-l10n:amd64 (31.2.2~22.04, automatic), ubuntu-pro-client:amd64 (31.2.2~22.04, automatic)
  ```

```
Upgrade: dpkg:amd64 (1.21.1ubuntu2.1, 1.21.1ubuntu2.3), libxtables12:amd64 (1.8.7-1ubuntu5,
1.8.7-1ubuntu5.2), initramfs-tools-core:amd64 (0.140ubuntu13.1, 0.140ubuntu13.4), udev:amd64
(249.11-0ubuntu3.7, 249.11-0ubuntu3.12), coreutils:amd64 (8.32-4.1ubuntu1,
8.32-4.1ubuntu1.2), libmm-glib0:amd64 (1.20.0-1~ubuntu22.04.1, 1.20.0-1~ubuntu22.04.3),
python3-tz:amd64 (2022.1-1ubuntu0.22.04.0, 2022.1-1ubuntu0.22.04.1), openssh-client:amd64
(1:8.9p1-3ubuntu0.6, 1:8.9p1-3ubuntu0.7), iptables:amd64 (1.8.7-1ubuntu5, 1.8.7-1ubuntu5.2),
python3-distupgrade:amd64 (1:22.04.16, 1:22.04.19), apt:amd64 (2.4.8, 2.4.12),
sosreport:amd64 (4.4-1ubuntu1.22.04.1, 4.5.6-0ubuntu1~22.04.2), cryptsetup-bin:amd64
(2:2.4.3-1ubuntu1.1, 2:2.4.3-1ubuntu1.2), git:amd64 (1:2.34.1-1ubuntu1.9,
1:2.34.1-1ubuntu1.10), libunwind8:amd64 (1.3.2-2build2, 1.3.2-2build2.1), libldap-
common:amd64 (2.5.16+dfsg-0ubuntu0.22.04.2, 2.5.17+dfsg-0ubuntu0.22.04.1), ufw:amd64
(0.36.1-4build1, 0.36.1-4ubuntu0.1), sg3-utils:amd64 (1.46-1build1, 1.46-1ubuntu0.22.04.1),
grub-pc-bin:amd64 (2.06-2ubuntu7.1, 2.06-2ubuntu7.2), libfwupd2:amd64 (1.7.9-1~22.04.1,
1.7.9-1~22.04.3), libapt-pkg6.0:amd64 (2.4.8, 2.4.12), initramfs-tools-bin:amd64
(0.140ubuntu13.1, 0.140ubuntu13.4), apparmor:amd64 (3.0.4-2ubuntu2.2, 3.0.4-2ubuntu2.3),
libip4tc2:amd64 (1.8.7-1ubuntu5, 1.8.7-1ubuntu5.2), libapparmor1:amd64 (3.0.4-2ubuntu2.2,
3.0.4-2ubuntu2.3), openssh-server:amd64 (1:8.9p1-3ubuntu0.6, 1:8.9p1-3ubuntu0.7),
irqbalance:amd64 (1.8.0-1build1, 1.8.0-1ubuntu0.2), python-apt-common:amd64 (2.4.0ubuntu1,
2.4.0ubuntu3), libgpgme11:amd64 (1.16.0-1.2ubuntu4, 1.16.0-1.2ubuntu4.2), libldap-2.5-0:amd64
(2.5.16+dfsg-0ubuntu0.22.04.2, 2.5.17+dfsg-0ubuntu0.22.04.1), libudev1:amd64
(249.11-0ubuntu3.7, 249.11-0ubuntu3.12), motd-news-config:amd64 (12ubuntu4.3, 12ubuntu4.6),
libsgutils2-2:amd64 (1.46-1build1, 1.46-1ubuntu0.22.04.1), libc6:amd64 (2.35-0ubuntu3.6,
2.35-0ubuntu3.7), locales:amd64 (2.35-0ubuntu3.6, 2.35-0ubuntu3.7), fwupd-signed:amd64
(1.51~22.04.1+1.2-3ubuntu0.2, 1.51.1~22.04.1+1.4-0ubuntu0.1), cloud-init:amd64
(23.1.2-0ubuntu0~22.04.1, 23.4.4-0ubuntu0~22.04.1), sg3-utils-udev:amd64 (1.46-1build1,
1.46-1ubuntu0.22.04.1), open-vm-tools:amd64 (2:12.1.5-3~ubuntu0.22.04.4,
2:12.3.5-3~ubuntu0.22.04.1), base-files:amd64 (12ubuntu4.3, 12ubuntu4.6), mdadm:amd64
(4.2-0ubuntu1, 4.2-0ubuntu2), cryptsetup-initramfs:amd64 (2:2.4.3-1ubuntu1.1,
2:2.4.3-1ubuntu1.2), python3-apt:amd64 (2.4.0ubuntu1, 2.4.0ubuntu3), snapd:amd64
(2.58+22.04.1, 2.61.3+22.04), systemd-hwe-hwdb:amd64 (249.11.3, 249.11.5), python3-distro-
info:amd64 (1.1build1, 1.1ubuntu0.2), libcryptsetup12:amd64 (2:2.4.3-1ubuntu1.1,
2:2.4.3-1ubuntu1.2), multipath-tools:amd64 (0.8.8-1ubuntu1.22.04.1, 0.8.8-1ubuntu1.22.04.4),
distro-info-data:amd64 (0.52ubuntu0.2, 0.52ubuntu0.6), libip6tc2:amd64 (1.8.7-1ubuntu5,
1.8.7-1ubuntu5.2), grub2-common:amd64 (2.06-2ubuntu7.1, 2.06-2ubuntu7.2), cryptsetup:amd64
(2:2.4.3-1ubuntu1.1, 2:2.4.3-1ubuntu1.2), distro-info:amd64 (1.1build1, 1.1ubuntu0.2),
openssh-sftp-server:amd64 (1:8.9p1-3ubuntu0.6, 1:8.9p1-3ubuntu0.7), grub-common:amd64
(2.06-2ubuntu7.1, 2.06-2ubuntu7.2), ubuntu-standard:amd64 (1.481, 1.481.1), libc-bin:amd64
(2.35-0ubuntu3.6, 2.35-0ubuntu3.7), http://netplan.io :amd64 (0.105-0ubuntu2~22.04.3,
0.106.1-7ubuntu0.22.04.2), python3-apport:amd64 (2.20.11-0ubuntu82.4, 2.20.11-0ubuntu82.5),
initramfs-tools:amd64 (0.140ubuntu13.1, 0.140ubuntu13.4), ubuntu-server:amd64 (1.481,
1.481.1), apt-utils:amd64 (2.4.8, 2.4.12), ubuntu-release-upgrader-core:amd64 (1:22.04.16,
1:22.04.19), libfwupdplugin5:amd64 (1.7.9-1~22.04.1, 1.7.9-1~22.04.3), ubuntu-advantage-
```

```
tools:amd64 (27.13.6~22.04.1, 31.2.2~22.04), ethtool:amd64 (1:5.16-1, 1:5.16-1ubuntu0.1), git-
man:amd64 (1:2.34.1-1ubuntu1.9, 1:2.34.1-1ubuntu1.10), grub-pc:amd64 (2.06-2ubuntu7.1,
2.06-2ubuntu7.2), kpartx:amd64 (0.8.8-1ubuntu1.22.04.1, 0.8.8-1ubuntu1.22.04.4), python3-
problem-report:amd64 (2.20.11-0ubuntu82.4, 2.20.11-0ubuntu82.5), libnetplan0:amd64
(0.105-0ubuntu2~22.04.3, 0.106.1-7ubuntu0.22.04.2), apport:amd64 (2.20.11-0ubuntu82.4,
2.20.11-0ubuntu82.5), ubuntu-minimal:amd64 (1.481, 1.481.1), update-notifier-common:amd64
(3.192.54.5, 3.192.54.8), python3-debian:amd64 (0.1.43ubuntu1, 0.1.43ubuntu1.1)

End-Date: 2024-09-18  15:52:27
```

- Verify that cluster-operation is unaffected by checking that all pods are in `Running` status using the following command in the Linux root shell.

```
# kubectl get pods -A
```

- Verify that the cluster is healthy and operational using the following command in the Linux root shell.

```
# health-check
```

Repair or Replace Cluster Nodes

IN THIS SECTION

You can repair and replace faulty nodes in your Paragon Automation cluster using Paragon Shell. This topic describes how to repair and replace nodes in your cluster.

Repair Nodes

To repair a faulty node in your existing Paragon Automation cluster.

1. Log in to the Linux root shell of the faulty node.

   If you are unable to log in to the faulty node, go to step "4" on page 59.

2. Stop and kill all RKE2 services on the faulty node.

```
root@node-f:~# rke2-killall.sh
root@node-f:~# rke2-uninstall.sh
```

3. Clear the data on the disk partition used for Ceph.

```
root@node-f:~# wipefs -a -f /dev/partition
root@node-f:~# dd if=/dev/zero of=/dev/partition bs=1M count=100
```

   Use **/dev/sdb** if you used the OVA bundle to deploy your cluster.

4. Log in to the Linux root shell of the node you deployed the cluster from. Note that you can repair the faulty node from any functional node in the cluster.

5. Delete the faulty node from the cluster.

```
root@primary1:~# kubectl delete node faulty-node-hostname
```

   Where *faulty-node-hostname* is the hostname of the node you want to repair.

6. Type `cli` to enter Paragon Shell.

   If you are not logged in to the node from which you deployed the cluster, then log out of the current node, and log in to the installer node.

7. Repair the node from Paragon Shell.

```
user@primary1> request paragon repair-node address ip-address-of-faulty-node
```

   Where *ip-address* is the IP address of the node that you want to repair.

Replace Faulty Nodes

IN THIS SECTION

You can replace a faulty node with a new node. To replace a node, you must prepare the new node, delete the faulty node and then add the new node to the cluster.

1. Log in to the Linux root shell of one of the functional nodes of the cluster and delete the faulty node.

```
root@primary1:~# kubectl delete node faulty-node-hostname
```

   Where *faulty-node-hostname* is the hostname of the node you want to replace.

2. **Prepare the new node.**

   Perform the following steps to prepare the new node before replacing a faulty node.

   The node which is replacing the faulty node can have a new IP address or the same IP address as the faulty node, but you will still need to create and prepare the node VM.

   a. Log in to the VMware ESXi 8.0 server on which you have installed Paragon Automation.

   b. **Create the new node VM.**

      Perform the following steps to create the VM.

      i. Right-click the **Host** icon and select **Create/Register VM**.

         The New virtual machine wizard appears.

      ii. On the Select creation type page, select **Deploy a virtual machine from an OVF or OVA file**.

         Click **Next**.

      iii. On the Select OVF and VMDK files page, enter a name for the node VM.

         Click to upload or drag and drop the OVA file or the OVF file along with the **.vmdk** file.

         Review the list of files to be uploaded and click **Next**.

iv. On the Select storage page, select the appropriate datastore that can accommodate 300-GB SSD for the node VM.

Click **Next**. The extraction of files takes a few minutes.

v. On the Deployment options page:

- Select the virtual network to which the node VM will be connected.
- Select the **Thick** disk provisioning option.
- Enable the VM to power on automatically.

Click **Next**.

vi. On the Ready to complete page, review the VM settings.

Click **Finish** to create the node VM.

vii. (Optional) Verify the progress of the VM creation in the Recent tasks section at the bottom of the page.

viii. When all the VM has been created, verify that it has the correct specifications and is powered on.

c. **Configure the new node VM.**

Perform the following steps to configure the node VM.

i. Connect to the node VM console. You are logged in as root automatically.

ii. You are prompted to change your password immediately. Enter and re-enter the new password. You are automatically logged out of the VM.

NOTE: We recommend that you enter the same password for as the VMs in your existing cluster.

iii. When prompted, log in again as root user with the newly configured password.

iv. Configure the following information when prompted.

Table 4: VM Configuration Wizard

Prompt	Action
Do you want to set up a Hostname? (y/n)	Enter **y** to configure a hostname.
Please specify the Hostname	Enter an identifying hostname for the VM. If you do not enter a hostname, a default hostname in the format `controller-<VM-IP address 4th octet>` `is assigned`.
Do you want to set up Static IP (preferred)? (y/n)	Enter **y** to configure the IP address for the VM. This IP address can be different or the same as the faulty node.
Please specify the IP address in CIDR notation	Enter the IP address in the CIDR notation. For example, 10.1.2.3/24. The IP address must be in the same subnet as the IP addresses of your existing Paragon Automation cluster.
Please specify the Gateway IP	Enter the gateway IP address.
Please specify the Primary DNS IP	Enter the primary DNS IP address.
Please specify the Secondary DNS IP	Enter the secondary DNS IP address.
Do you want to set up IPv6? (y/n)	Enter `y` to configure IPv6 addresses. If you don't want to configure IPv6 addresses, enter `n` and proceed to Step "2.c.v" on page 63. **NOTE**: Configure IPv6 addresses only if your existing cluster is configured with IPv6 addresses.

Table 4: VM Configuration Wizard *(Continued)*

Prompt	Action
Please specify the IPv6 address in CIDR notation	Enter the IPv6 address in the CIDR notation. For example, 2001:db8:1:2::3/64. **NOTE**: If you enter 2001:db8:1:2::3 instead of 2001:db8:1:2::3/64, you will see an Invalid IP address error message.
Please specify the Gateway IPv6	Enter the gateway IPv6 address.
Please specify the Primary DNS IPv6	Enter the primary DNS IPv6 address.
Please specify the Secondary DNS IPv6	Enter the secondary DNS IPv6 address.

v. When prompted if you are sure to proceed, review the information displayed, type **y** and press Enter.

You have prepared the node and can now replace the faulty node with the newly prepared node.

3. **Replace the Faulty Node:**

   Log in to the node from which you deployed your existing Paragon Automation cluster. You are placed in Paragon Shell.

4. If the IP address of the new node is same as the IP address of the faulty node, go to step "5" on page 65. If the IP address of the new node is different from the IP address of the faulty node, perform the following steps.

   a. To edit the cluster, type `configure` to enter the configuration mode.

```
user@primary1> configure
Entering configuration mode

[edit]
user@primary1#
```

b. Delete the faulty node.

```
user@primary1# delete paragon cluster nodes kubernetes 3
```

Where *3* is the index number of the node that you want to delete.

c. Add the new node to the cluster configuration in place of the node you deleted and commit the configuration.

```
user@primary1# set paragon cluster nodes kubernetes 3 address 10.1.2.11

user@primary1# commit
commit complete
```

Where *10.1.2.11* is the IP address of the new node.

d. (Optional) Verify the cluster configuration.

```
user@primary1# show paragon cluster nodes
kubernetes 1 {
    address 10.1.2.3;
}
kubernetes 2 {
    address 10.1.2.4;
}
kubernetes 3 {
    address 10.1.2.11;
}
kubernetes 4 {
    address 10.1.2.6;
}
```

e. Exit configuration mode and regenerate the configuration files.

```
user@primary1# exit
Exiting configuration mode

user@primary1> request paragon config
Paragon inventory file saved at /epic/config/inventory
Paragon config file saved at /epic/config/config
```

5. Regenerate SSH keys on the cluster nodes.

   When prompted, enter the SSH password for all the existing VMs and the new VM. Enter the same password that you configured to log in to the VMs.

```
user@primary1> request paragon ssh-key
Please enter comma-separated list of IP addresses: 10.1.2.3,10.1.2.4,10.1.2.6,10.1.2.11
Please enter SSH username for the node(s): root
Please enter SSH password for the node(s): password
checking server reachability and ssh connectivity ...
Connectivity ok for 10.1.2.3
Connectivity ok for 10.1.2.4
Connectivity ok for 10.1.2.6
Connectivity ok for 10.1.2.11

<output snipped>
```

6. Type `configure` to enter configuration mode and replace the node.

```
user@primary1> configure
Entering configuration mode

[edit]
user@primary1# request paragon replace-node address 10.1.2.11
Process running with PID: 23xx032
To track progress, run 'monitor start /epic/config/log'
```

   Where *10.1.2.11* is the IP address of the new node.

Replace-node failure scenario

You might encounter issues with node-replacement when the IP address (or hostname) of the replacement node is different from the IP address (or hostname) of the faulty node; perform the following additional steps to fix the issue.

1. Log in to the Linux root shell of the node from where you deployed the cluster.
2. Delete the local volume pvc associated with the faulty node, if any.
   a. Use the `# kubectl describe pv -A` command to determine if there is any lingering pvc associated with the faulty node and note the pvc name.
   b. Use the `# kubectl get pvc -A` command to find the namespace associated with the pvc name.

c. Use the `# kubectl delete pvc -n namespace pvc_name` command to delete the pvc.

3. Run the following command to check if the status of Rook and Ceph pods installed in the rook-ceph namespace is not `Running` or `Completed`.

```
$ kubectl get po -n rook-ceph
```

4. Remove the failing OSD processes.

```
$ kubectl delete deploy -n rook-ceph rook-ceph-osd-number
```

5. Connect to the toolbox.

```
$ kubectl exec -ti -n rook-ceph $(kubectl get po -n rook-ceph -l app=rook-ceph-tools -o
jsonpath={..metadata.name}) -- bash
```

6. Identify the failing OSD.

```
$ ceph osd status
```

7. Mark out the failed OSD.

```
$ ceph osd out osd-ID-number
```

8. Remove the failed OSD.

```
$ ceph osd purge osd-ID-number --yes-i-really-mean-it
```

SEE ALSO

Install Paragon Automation | 20

Shut Down and Reboot Nodes

You may need to shut down or reboot one or more or all the node VMs for reasons such as performing a scheduled maintenance activity, upgrading your ESXi server, recovering from a node failure, and so on. This topic describes how to gracefully shut down and restart your node VMs and the entire Paragon Automation cluster.

Shut down a node VM

1. Log in to the node VM. You are placed in Paragon Shell.
2. (Optional) Check the health of your cluster. Execute the `request paragon health-check` command. The `Overall Cluster Status` must be `GREEN` or `AMBER`.
3. Shut down the node using the `request paragon shutdown type node` command.

Reboot a node VM

1. Log in to the node VM. You are placed in Paragon Shell.
2. Reboot the node using the `request paragon reboot type node` command.
3. (Optional) After the VM has rebooted, verify that the cluster is in good health. Execute the `request paragon health-check` command. The `Overall Cluster Status` must be `GREEN` or `AMBER`.

Shut down the cluster

1. Log in to any node VM. You are placed in Paragon Shell.
2. (Optional) Check the health of your cluster. Execute the `request paragon health-check` command. The `Overall Cluster Status` must be `GREEN` or `AMBER`.
3. Shut down the cluster using the `request paragon shutdown type cluster` command.

Reboot the cluster

1. Log in to the node VM. You are placed in Paragon Shell.
2. Reboot the node using the `request paragon reboot type cluster` command.
3. (Optional) After the cluster has rebooted, verify that the cluster is in good health. Execute the `request paragon health-check` command. The `Overall Cluster Status` must be `GREEN` or `AMBER`.

Increase TimescaleDB PVC Size

SUMMARY

Use the steps detailed in this topic to manually increase the size for TimescaleDB persistent volume claim (PVC).

IN THIS SECTION

The minimum recommended system requirements to deploy a Paragon Automation cluster is described in "Paragon Automation System Requirements" on page 10. If you want to scale up your cluster you can increase the hardware resources that are required on each node VM when you are creating the VMs for the first time as per your requirement. If you have already deployed your cluster, use the information detailed in this topic to increase storage size, specifically Ceph and consequently TimescaleDB PVC.

Ceph storage is used for both object storage and the PVC for the pods. ~50% of Ceph storage is allocated for TimescaleDB PVC.

To increase the size of the PVC, perform the following steps:

1. "Increase Ceph storage." on page 68
2. "Update the allocation quota for Timescale DB PVC in Ceph storage." on page 71
3. "Increase the Timescale DB PVC size." on page 71

Increase Ceph Storage

Increase the total Ceph storage size.

1. Increase the virtual disk size from your ESXi server.

   Ceph uses the second disk attached to the node virtual machine (VMs). Ensure that you resize the correct virtual disk. You do not need to reboot the VM.
2. Verify the new virtual disk size.
   a. Log in to the Linux root shell of the node VM.

b. Use the `lsblk` command to verify that the OS detects the new disk size. For example:

```
root@vm4:~# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
loop0    7:0    0   64M  1 loop /snap/core20/2379
loop1    7:1    0 63.7M  1 loop /snap/core20/2434
loop2    7:2    0   87M  1 loop /snap/lxd/29351
loop3    7:3    0 89.4M  1 loop /snap/lxd/31333
loop4    7:4    0 38.8M  1 loop /snap/snapd/21759
loop5    7:5    0 44.3M  1 loop /snap/snapd/23258
sr0     11:0    1    4M  0 rom
nbd0    43:0    0    0B  0 disk
nbd1    43:32   0    0B  0 disk
nbd2    43:64   0    0B  0 disk
nbd3    43:96   0    0B  0 disk
nbd4    43:128  0    0B  0 disk
nbd5    43:160  0    0B  0 disk
nbd6    43:192  0    0B  0 disk
nbd7    43:224  0    0B  0 disk
vda    252:0    0  700G  0 disk
├─vda1 252:1    0    1M  0 part
└─vda2 252:2    0  700G  0 part /var/lib/kubelet/pods/fde8c46d-f069-4203-bd4e-3897d5915559/
volume-subpaths/config/network/0
....
                                 /export/local-volumes/pv1
                                 /
vdb    252:16   0   55G  0 disk
```

In this example, the OS detects that the `vdb` was increased from 50-GB to 55-GB.

3. Restart Ceph OSD to detect the size change. For example:

a. Verify the existing OSD size.

i. Launch the Rook tools pod.

```
root@vm1:~# kubectl exec -ti -n rook-ceph $(kubectl get po -n rook-ceph -l app=rook-
ceph-tools -o jsonpath={..metadata.name}) -- bash
```

ii. Retrieve the current OSD size.

```
bash-4.4$ ceph osd status
ID  HOST USED  AVAIL  WR OPS  WR DATA  RD OPS  RD DATA  STATE
 0  vm2  2846M  47.2G      0     2633       0        0   exists,up
 1  vm3  3132M  46.9G      1     28.6k      0     1135k  exists,up
 2  vm4  3065M  47.0G      4     23.9k      3      201k  exists,up
 3  vm1  2897M  47.1G      1     2698       1      979k  exists,up
```

In this example, you are modifying the OSD on vm4. Here, we still see original size, which is ~50-GB (USED+AVAIL).

b. Go back to the Linux root shell and determine the OSD pod that runs on the node for which the disk size is to be increased (vm4).

```
root@vm1:~# kubectl get pod -A -o wide | grep osd | grep vm4
...
rook-ceph          rook-ceph-osd-2-787df64c87-bkjt8                          2/2
Running            2 (4d3h ago)     13d     10.1.2.8     vm4   <none>           <none>
...
```

c. Restart the pod.

```
root@vm1:~# kubectl rollout restart deploy -n rook-ceph rook-ceph-osd-2-787df64c87-bkjt8
```

d. Verify the new size.

i. Launch the Rook tools pod and execute `ceph osd status` again.

```
bash-4.4$ ceph osd status
ID  HOST USED  AVAIL  WR OPS  WR DATA  RD OPS  RD DATA  STATE
 0  vm2  2924M  47.1G      0     4403       0        0   exists,up
 1  vm3  3202M  46.8G      0      426k      0     1140k  exists,up
 2  vm4  1474M  53.5G      0      477       1      186k  exists,up
 3  vm1  2977M  47.0G      3     21.4k      3     1008k  exists,up
```

Here, the OSD on vm4 has increased to ~55-GB.

ii. Verify that the total size has also increased.

```
bash-4.4$ ceph status
...
  data:
...
    usage:   11 GiB used, 194 GiB / 205 GiB avail
```

Here, the total size has increased from 200-GB to 205-GB.

NOTE: In this example, you increased the size of Ceph storage on one node VM. We recommend that you increase the size on all the node VMs to make the storage value consistent across all the nodes. Repeat these steps for the remaining three VMs.

Update Timescale DB PVC Quota

If you increase the total size of Ceph storage, you must update the allocation quota between object storage and PVC. Use Paragon Shell to increase the allocation quota. For example:

```
root@vm1> request paragon deploy cluster input "-t rook-quota"
Process running with PID: 1830232
To track progress, run 'monitor start /epic/config/log'
```

Increase Timescale DB PVC

You can increase the PVC size for TimescaleDB by using Paragon Shell. When you are installing Paragon Automation afresh, override the default PVC size and add the `set paragon cluster custom-config keyValue paa-timescaledb-storage-size string` *size* configuration before deploying the Paragon Automation cluster.

1. Log in to the installer node VM.
2. Configure the new PVC size in configure mode. For example:

```
root@vm1> configure
Entering configuration mode
```

```
[edit]
root@vm1# set paragon cluster custom-config keyValue paa-timescaledb-storage-size string 40Gi

[edit]
root@vm1# commit
warning: *** operating on 10.1.2.8 ***
warning: *** operation on 10.1.2.8 succeeds ***
warning: *** operating on 10.1.2.7 ***
warning: *** operation on 10.1.2.7 succeeds ***
warning: *** operating on 10.1.2.6 ***
warning: *** operation on 10.1.2.6 succeeds ***
commit complete

[edit]
root@vm1# exit
Exiting configuration mode

root@vm1> request paragon config
Paragon inventory file saved at /epic/config/inventory
Paragon config file saved at /epic/config/config.yml
```

3. Deploy the Paragon Automation cluster as usual.

5
CHAPTER

Backup and Restore

IN THIS CHAPTER

Back Up and Restore Paragon Automation

IN THIS SECTION

This topic describes the backup and restore functionality available for Paragon Automation.

CAUTION: We support only the snapshot feature available on VMware clients to back up and restore your cluster. We recommend that you use *only* the snapshot feature available on VMware clients to back up and restore your cluster.

Back Up and Restore Using VMware Snapshots

IN THIS SECTION

You can use the snapshot functionality available on VMware vSphere and ESXi clients to back up and restore your Paragon Automation cluster and application configuration data. The backup procedure using the snapshot functionality can be performed while the microservices and applications are running and does not affect the operation of the network. The snapshot creates a copy of your cluster and application configuration information. The snapshot restore functionality enables you to restore your cluster on the same set of VMs from which the backup was taken. The VMs are suspended before performing a restore operation and the downtime associated with a restore operation is minimal.

Back Up Using VMware Snapshots

To back up your Paragon Automation cluster and application configuration data, use any one of the following options:

- **On VMware vSphere client**

  1. Log in to any of the cluster nodes.
  2. Use the `request paragon health-check` Paragon Shell command to ensure that your cluster is in good health. The status of the cluster must be `GREEN`.
  3. Log in to the VMware vSphere client which manages the Paragon Automation VMs.
  4. Click on your cluster host in the VM inventory.

     Your cluster node VMs are listed in the **Virtual Machines** tab under VMs.
  5. Select all the VMs, right-click, and click **Take snapshot**. The snapshot creation dialog box appears.
  6. Enter a name for the snapshot and a description. Also, select the **Include virtual machine's memory** check-box.
  7. Click **Create** to create the snapshot. The snapshot creation takes a few minutes.

     You can check the progress of the job under **Recent Tasks** or in the **Task Console**.

- **On VMware ESXi client**

  1. Log in to any of the cluster nodes.
  2. Use the `request paragon health-check` Paragon Shell command to ensure that your cluster is in good health. The status of the cluster must be `GREEN`.
  3. Log in to the VMware ESXi server on which your Paragon Automation VMs are located.
  4. Select a cluster node VM listed under **Virtual Machines**, right-click and click **Take snapshot**. The snapshot creation dialog box appears.
  5. Enter a name for the snapshot and a description. Also, select the **Include virtual machine's memory** check-box.
  6. Click **Create** to create the snapshot.
  7. Repeat steps "4" on page 75 through "6" on page 75 for the other three VMs and enter appropriate snapshot names when prompted.

     NOTE: We recommend that you create snapshots of all the VMs one after the other as close as possible to each other in time.

8. (Optional) You can check the progress of the snapshot jobs under **Recent Tasks** or in the **Task Console**.

- **Using APIs**

  You can also use the `snapshot.create` option to take snapshots of VMs using APIs. For more information, see https://github.com/vmware/govmomi/blob/main/govc/USAGE.md#snapshotcreate.

Restore Using VMware Snapshots

To restore your Paragon Automation cluster and application configuration data which was previously backed up in snapshots, use any one of the following options:

- **On VMware vSphere client**

  1. Log in to the VMware vSphere client which manages the Paragon Automation VMs and where the snapshots were previously taken.
  2. Click your cluster host in the VM inventory.

     Your cluster VMs are listed in the **Virtual Machines** tab under VMs.
  3. Select all the VMs, right-click, and click **Suspend** to suspend the operation of the VMs. Do not restore the VMs when they are powered on.
  4. Select a cluster node VM, right-click, and click **Manage snapshots**. The Manage snapshots page appears with a list of available snapshots.
  5. Select the snapshot that you want to restore and click **Restore snapshot**.

     Ensure that you select a snapshot that is recent and stable.
  6. Repeat steps "4" on page 76 through "5" on page 76 for the other three VMs.

     Alternatively, you can select a VM, right-click and click **Revert to Latest Snapshot** if your immediate last snapshot was the most stable version.

     > **NOTE**: We recommend that you restore snapshots of all the VMs one after the other as close as possible to each other in time. Do not wait for the snapshot restoration of any one VM to complete before restoring the other.

  7. (Optional) You can check the progress of the job under **Recent Tasks** or in the **Task Console**.
  8. Once the restore process is complete, wait for a few minutes for the VMs to power-on and the cluster to stabilize.
  9. Log in to any of the cluster nodes.

10. Type the `show paragon cluster nodes` command in Paragon Shell. Ensure that the status of all the nodes is `Ready`.

11. Type the `show paragon cluster pods` command and ensure that the status of all the pods is either `Running` or `Completed`.

12. Type the `request paragon health-check` command to ensure that your cluster is in good health. The status of the cluster must be `GREEN`.

    NOTE: If the cluster health-check does not return a `GREEN` status, reboot all the nodes using the `request paragon reboot type cluster` command and recheck the status.

- **On VMware ESXi client**

1. Log in to the VMware ESXi server on which your Paragon Automation VMs are located.

2. Select you cluster node VM listed under **Virtual Machines**, right-click and click **Suspend** to suspend the operation of the VM. Do not restore a VM when it is powered on.

3. Right-click the selected VM and click **Manage snapshots**. The Manage snapshots page appears with a list of available snapshots.

4. Select the snapshot that you want to restore and click **Restore snapshot**.

   Ensure that you select a snapshot that is recent and stable.

5. Repeat steps "2" on page 77 through "4" on page 77 for the other three VMs.

   Alternatively, you can select a VM, right-click and click **Revert to Latest Snapshot** if your immediate last snapshot was the most stable version.

   NOTE: We recommend that you restore snapshots of all the VMs one after the other as close as possible to each other in time. Do not wait for the snapshot restoration of any one VM to complete before restoring the other.

6. (Optional) You can check the progress of the restore jobs under **Recent Tasks** or in the **Task Console**.

7. Once the restore process is complete, wait for a few minutes for the VMs to power-on and the cluster to stabilize.

8. Log in to any of the cluster nodes.

9. Type the `show paragon cluster nodes` command in Paragon Shell. Ensure that the status of all the nodes is `Ready`.

10. Type the `show paragon cluster pods` command and ensure that the status of all the pods is either `Running` or `Completed`.

11. Type the `request paragon health-check` command to ensure that your cluster is in good health. The status of the cluster must be `GREEN`.

- **Using APIs**

  You can also use the `snapshot.revert` option to restore backed-up snapshots of VMs using APIs. For more information, see https://github.com/vmware/govmomi/blob/main/govc/USAGE.md#snapshotrevert.

Disaster Recovery

You can restore data on a fresh installation from a backup take on a different setup using the procedure described in this section. You can back up your Paragon Automation cluster and all associated application configuration by exporting it to your ESXi server. Once exported, you can redeploy it on a new installation. During backup and re-installation, your cluster is shutdown and is nonoperational. Perform the following steps to export and redeploy:

1. Verify that the cluster is in good health. Log in to one of the primary nodes and execute the `request paragon health-check` command from the Linux root shell.

   The `Overall Cluster Status` must be `GREEN`.

2. Export your current cluster configuration.

   a. Log in to one of the node VMs. You are placed in Paragon Shell.

   b. Shut down the cluster using the `request paragon shutdown type cluster` command.

   c. Log in to your ESXi server.

   d. Export the configuration of all VMs one after another and enter appropriate names when prompted.

      Perform the steps described in the VMware vSphere documentation at Export an OVF Template.

   The VM configuration is backed up locally as four sets of OVF, **.vmdk**, **.nvram**, and **.mf** configuration files.

3. Redeploy the exported configuration as a fresh installation.

   Perform the steps described in "Create the node VMs" on page 24. Select the corresponding sets of OVF and configuration files for each VM.

   The cluster is re-installed afresh.

4. (Optional) Verify that the cluster is in good health. Log in to one of the cluster nodes and execute the `request paragon health-check` command.

   The `Overall Cluster Status` must be `GREEN`.

Back Up and Restore Using Paragon Shell

IN THIS SECTION

You can also use the backup and restore functionality available in Paragon Shell to back up and restore your Paragon Automation cluster and application configuration data.

CAUTION: The backup and restore functionality available in Paragon Shell is inconsistent and might not work as intended. We recommend that you use the snapshot feature available on VMware clients to back up and restore your cluster. See "Back Up and Restore Using VMware Snapshots" on page 74.

Back Up Using Paragon Shell

You can back up your current Paragon Automation network configuration using Paragon Shell CLI. When you run the backup command, all the application configuration information stored in PostgreSQL,ArangoDB, and Airflow configuration database systems are backed up. The backup command also backs up telemetry information stored in TimescaleDB, OpenSearch, and VictoriaMetrics database systems. The backup procedure can be performed while the microservices and applications are running and does not affect the operation of the network.

To back up your Paragon Automation configuration state.

1. Log in as root user to any of the Paragon Automation nodes.
2. Execute the following command to back up the configuration.

   ```
   root@Primary1> request paragon backup
   ```

Alternatively, you can use the `request paragon backup config` command to back up only configuration information or `request paragon backup telemetry` command to back up only telemetry data.

3. (Optional) View progress of the backup job. The backup process takes a few minutes to complete. In order to not block the terminal for use for the whole duration of the backup job, you are returned to the command prompt before the backup job is complete. You can view the progress status of the backup job by using the following command.

   ```
   root@Primary1> show paragon backup status backup-id backup ID
   ```

Upon completion, a backup file with a filename in the **yyyymmdd-hhmmss** format is created. The backup file is stored in the local persistent **/export/paragon-shell/backup** folder on the node. You'll have to exit out of Paragon Shell to the Linux root shell to navigate to the **backup** folder.

Caveat of the backup process

- Application configurations (such as devices, sites, service orders, and so on) are backed up, but certificates and infrastructure services configurations are not backed up. This information must be kept unchanged before you perform a restore.
- Backup process will capture current infrastructure configurations, but information is used for reference only. The same configuration can be used to instantiate a new setup.

Restore Using Paragon Shell

You can restore your Paragon Automation network configuration from a backup configuration file. To restore from a backup configuration file, all microservices and applications must be stopped, and the cluster is not functional until the databases are restored. Once the databases are flushed and restored to the backed-up configuration, the applications must be restarted, and configuration restored from the databases must be reparsed.

To restore your Paragon Automation configuration from a specific backup configuration file.

1. Log in as root user to the node where the backup file is located.
2. Execute the following command to uninstall all the running applications.

   ```
   root@Primary1> request paragon destroy cluster input “-v -t apps”
   ```

   This command takes some time to complete, and you must wait until all the applications are shut down before proceeding to the next step. To check the status of the applications, perform the following steps.

   a. Type `exit` to exit Paragon Shell to the Linux root shell.

   b. Check each application namespace for running deployments using the `kubectl get deployments -n` *namespace* command, where *namespace* is the name of the application namespace. Use this command for each namespace and the output of every command should be empty, which indicates that no deployments are running. Table 5 on page 81 lists all the namespaces that you must check.

Table 5: Namespace List

airflow	epic	foghorn	gnmi-term
mems	netbox	northstar	oc-term
paa	papi	streams	trust
healthbot			

The command output should display *only* the following deployments:

```
vm-operator-victoria-metrics-operator, vmauth-victoria-metrics-auth, vminsert-victoria-metrics-cluster
```

When all applications are shut down, you are ready to proceed with the restore process.

c. Type `cli` to log in again to Paragon Shell.

3. (Optional) If your backed up file is in a remote location, download the file locally.

   ```
   root@Primary1> request paragon backup download backup-id backup-ID storage-location path user username password password
   ```

   Where:

   path is the path to the remote backup file.

   username and *password* are the login credentials to the remote server.

4. Restore the applications configuration from the backup file.

   ```
   root@Primary1> request paragon restore backup-id backup-ID
   ```

5. (Optional) View progress status of the restore job.

   ```
   root@Primary1> show paragon restore status backup-id backup-ID
   ```

6. Reinstall all the applications.

   ```
   root@Primary1> request paragon deploy cluster input “-v -t apps”
   ```

7. Reparse and synchronize the restored configuration.

   ```
   root@Primary1> request paragon restore sync
   ```

You must perform the restore operation only on the node on which the required backup file is located.

Caveats of the restore process

- When you perform the restore operation, the network configuration is returned to the configuration present in the backup file. From the time the backup was taken, if the network configuration has changed due to new devices being onboarded or new service orders being executed, the network configuration in Paragon Automation might be different from the actual network state. To ensure

that the network configuration in Paragon Automation and the actual network state have minimal mismatch post a restore operation, we recommend that you take regular periodic backups or specific backups after every network intent change.

- You cannot restore data from a release different from the current installed release of Paragon Automation.
- You can restore data on the same setup on which a backup was taken and not on a fresh installation using the procedure described in this section. To restore backed up data on a fresh installation, use the procedure described in "Disaster Recovery" on page 78.
- Since a backup does not store the certificates and infrastructure services configurations, that information must be kept unchanged during restoration.
- Resources allocated to the network won't be preserved after a restore and you must ensure that you release the allocated resources during the window between taking a backup and performing a restore.

View or Delete Backup Files

To view a list of all backup files across all nodes, use the following command:

```
root@Primary1> show paragon backup
```

The node connects to all the other nodes in the Paragon Automation cluster using SSH and displays a list of all backup file names along with the IP address of the node on which the file is located.

To view a list of backup files along with a list of failed backup attempts, use the following command:

```
root@Primary1> show paragon backup include-failure true
```

To delete a backup file, use the following command.

```
root@Primary1> request paragon backup delete backup-id backup-ID
```

You can delete a backup file that is located only on the node on which you execute the command.

Upload or Download Backup Files

Upload your backed-up file to a remote location outside the Paragon Automation cluster, within the same network as the cluster or in a different network. To upload your backup file to a remote location, use the following command:

```
root@Primary1>request paragon backup upload backup-id backup-ID storage-location scp://IP:port/remote-path user username password password
```

To view progress of the backup file upload command, use the following command:

```
root@Primary1> show paragon backup upload status backup-id backup-ID
```

To download your backup file from a remote location, use the following command:

```
root@Primary1>request paragon backup download backup-id backup-ID storage-location scp://IP:port/remote-path user username password password
```

To view progress of the backup file download command, use the following command:

```
root@Primary1> show paragon backup upload status backup-id backup-ID
```

RELATED DOCUMENTATION

Upgrade Paragon Automation | 45

request paragon backup

request paragon restore

show paragon backup