

Paragon Insights Installation Guide

Published
2024-10-24

RELEASE
4.3.0

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

Juniper Networks, the Juniper Networks logo, Juniper, and Junos are registered trademarks of Juniper Networks, Inc. in the United States and other countries. All other trademarks, service marks, registered marks, or registered service marks are the property of their respective owners.

Juniper Networks assumes no responsibility for any inaccuracies in this document. Juniper Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

Paragon Insights Installation Guide

4.3.0

Copyright © 2024 Juniper Networks, Inc. All rights reserved.

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

1

[About This Guide | v](#)

[Installing Paragon Insights](#)

[Paragon Insights Installation Overview | 2](#)

[Paragon Insights Installation Requirements | 2](#)

[Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X | 16](#)

[Installing Paragon Insights On Ubuntu | 19](#)

[Installing Paragon Insights On CentOS | 23](#)

[Paragon Insights Interactive Installation Prompts | 30](#)

[Using the Silent Installer | 33](#)

[Starting and Stopping Paragon Insights | 42](#)

[Checking Paragon Insights Service Status | 43](#)

[Upgrading Paragon Insights Release 3.X with Kubernetes-based Installation to Release 4.X | 46](#)

[Migration from a Single-Master Setup to a Multi-Master Setup | 48](#)

[Before You Begin | 48](#)

[Steps for Migration | 49](#)

[Multi-NIC | 50](#)

[Migration of Paragon Insights Release 3.2.0 \(with Docker Compose\) to Paragon Insights Release 4.0.0 | 50](#)

[Migration from Release 2.X to Release 3.X or Release 4.X | 53](#)

[Overview | 53](#)

[Case 1: Release 2.X Single-node \(Docker Compose\) to Release 3.X Single-node \(Docker Compose\) Migration | 54](#)

[Case 2: Automated Migration from 2.X to 3.X and 4.X \(Kubernetes\) | 55](#)

[Case 3: Manual Migration from 2.X to 3.X and 4.X \(Kubernetes\) | 56](#)

Installing HealthBot Release 3.2.0 | 57

Installing Paragon Insights On Ubuntu | 59

Installing Paragon Insights On CentOS | 62

Uninstalling or Reconfiguring Paragon Insights | 65

Uninstalling Paragon Insights | 66

Reconfiguring Paragon Insights | 67

Paragon Insights Command-Line Options | 70

2

Linux Kernel Upgrade Procedures

Ubuntu Kernel Upgrade | 75

CentOS Kernel Upgrade | 81

About This Guide

Use this guide to install Paragon Insights on a Linux server.

1

CHAPTER

Installing Paragon Insights

[Paragon Insights Installation Overview | 2](#)

[Paragon Insights Installation Requirements | 2](#)

[Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X | 16](#)

[Paragon Insights Interactive Installation Prompts | 30](#)

[Using the Silent Installer | 33](#)

[Starting and Stopping Paragon Insights | 42](#)

[Checking Paragon Insights Service Status | 43](#)

[Upgrading Paragon Insights Release 3.X with Kubernetes-based Installation to Release 4.X | 46](#)

[Migration from a Single-Master Setup to a Multi-Master Setup | 48](#)

[Migration of Paragon Insights Release 3.2.0 \(with Docker Compose\) to Paragon Insights Release 4.0.0 | 50](#)

[Migration from Release 2.X to Release 3.X or Release 4.X | 53](#)

[Installing HealthBot Release 3.2.0 | 57](#)

[Uninstalling or Reconfiguring Paragon Insights | 65](#)

[Paragon Insights Command-Line Options | 70](#)

Paragon Insights Installation Overview

Paragon Insights (Formerly HealthBot) is a highly automated and programmable device-level diagnostics and network analytics tool that aggregates and correlates large volumes of time-sensitive telemetry data, providing a multidimensional and predictive view of the network. Paragon Insights collects telemetry data using various methods, including Junos Telemetry Interface, NETCONF, OpenConfig, SNMP, NetFlow, sFlow, and syslog.

This guide describes how to install Paragon Insights and is intended for network operators and administrators who install, configure, and manage the network security infrastructure; it specifically explains how to:

- Install Paragon Insights using the interactive or silent installer
- Upgrade Paragon Insights from previous version
- Start and stop Paragon Insights services, and check their status
- Adjust Paragon Insights setup parameters
- Reinstall and uninstall Paragon Insights

RELATED DOCUMENTATION

| [Paragon Insights Installation Requirements](#) | 2

Paragon Insights Installation Requirements

IN THIS SECTION

- [Paragon Insights Hardware Requirements](#) | 3
- [Paragon Insights Software Requirements](#) | 4
- [Web Browser Requirements](#) | 7
- [Network Requirements](#) | 8
- [Network Device Requirements](#) | 11
- [Multinode Installation](#) | 16

You can perform an online or offline installation of Paragon Insights.

- **Online Installation**—You need an active internet connection to run an online installation of Paragon Insights. You must download the necessary online package (.deb for Ubuntu or .rpm for CentOS/RedHat) from the Juniper Networks [Software Downloads](#) page and install them. You can then proceed to run the `sudo healthbot setup` and `healthbot start` commands.
- **Offline Installation**—You do not need an active internet connection during offline installation. However, you need an active internet connection to download and install the necessary Paragon Insights package (.deb for Ubuntu or .rpm for CentOS/RedHat) from the Juniper Networks [Software Downloads](#) page and install them. After you install the package (.deb or .rpm), you must download the offline install package to the `/var/local/healthbot` directory and rename the file to **healthbot-offline.tgz**. You can then run the `sudo healthbot setup` and `healthbot start` commands without an active internet connection.

For Paragon Insights (formerly HealthBot) to install successfully, the following hardware and software components are required on the host machine.

Paragon Insights Hardware Requirements

You can install Paragon Insights on either a physical or a virtual machine.

Proof-of-concept (POC) system—supports up to two device groups and three devices per device group:

- RAM: 20 GB
- Disk space: 100 GB available on the `/var/` partition
- Free disk space must be at least 20% of total disk space at all times.
- CPU cores: 8

Production system (minimum system requirements):

- RAM: 32 GB
- Disk space: 250 GB SSD available on the `/var/` partition
- Free disk space must be at least 20% of total disk space at all times.
- Recommended minimum IOPS for the disk(s): 1000
- CPU cores: 16

NOTE: Paragon Insights is a cloud-native application that leverages a microservices-based architecture that allows scale-out and multinode deployment. Depending on your specific requirements and use case, you can add more nodes to the Kubernetes cluster.

For more information on rules and telemetry sensors supported in a Junos OS and Junos OS Evolved software release for a specific hardware platform, see [Telemetry Sensor Explorer](#).

Paragon Insights Software Requirements

You can install Paragon Insights on Ubuntu, Red Hat Enterprise Linux (RHEL), and CentOS versions of Linux.

For Ubuntu:

- The following Ubuntu releases are qualified to work with Paragon Insights:

Installation	Paragon Insights Release	Ubuntu Version
Online Installation	Release 4.3.0	Ubuntu 20.04.3 and Ubuntu 18.04.04
	Release 4.2.0 and earlier	Ubuntu 18.04.04 and Ubuntu 16.04.01
Offline Installation	Release 4.3.0	Ubuntu 20.04.3 and Ubuntu 18.04.04
	Release 4.2.0 and earlier	Ubuntu 18.04.04 and Ubuntu 16.04.01

- The kernel version must be 4.4.19 or greater.
- We recommend installing Ubuntu as one large disk partition.

If multiple partitions are used, Paragon Insights data is written to the `/var/local/healthbot/` directory and Paragon Insights log files are written to `/var/lib/docker/containers`.

- Disable the swap memory by running the `swapoff -a` command to bring up kubelet service.

You must then remove swap memory entries from the `/etc/fstab` folder.

NOTE: You might need to reboot the server for this setting to take effect.

For RHEL:

- The following RHEL releases are qualified to work with Paragon Insights:

Installation	Paragon Insights Release	RHEL Release
Online Installation	Release 4.1.0 and later	RHEL 7.5 and RHEL 8.2
	Release 4.0.0	Not supported
	Release 3.2.0 and earlier	RHEL 7.5
Offline Installation	Release 4.0.0 and later	RHEL 7.5 and RHEL 8.3
	Release 3.2.0 and earlier	RHEL 7.5

NOTE: All Paragon Insights releases except Release 4.0.0 support both online and offline installations. Paragon Insights Release 4.0.0 supports only offline installation.

- The following system utilities must be installed manually if they are not already present:
tar, bash, ln, ssh-keygen, curl, vi, wget, openssl, openssh-server, and rsync
- Enter the following configuration line in the file `/etc/sysctl.conf`: `vm.max_map_count=262144`.
- Disable the swap memory by running the `swapoff -a` command to bring up kubelet service.
You must then remove swap memory entries from the `/etc/fstab` folder.

NOTE: You might need to reboot the server for this setting to take effect.

For CentOS:

- The following CentOS releases are qualified to work with Paragon Insights:

Installation	Paragon Insights Release	CentOS Release
Online Installation	Release 4.3.0	CentOS 7.3
	Release 4.2.x and Release 4.1.0	CentOS 7.3 and CentOS 8.2
	Release 4.0.0	Not supported
	Release 3.2.0 and earlier	CentOS 7.3
Offline Installation	Release 4.3.0	Not supported
	Release 4.2.x, Release 4.1.0, and Release 4.0.0	CentOS 7.3 and CentOS 8.3
	Release 3.2.0 and earlier	CentOS 7.3

NOTE: All Paragon Insights releases except Releases 4.3.0 and 4.0.0 support both online and offline installations. Paragon Insights Release 4.3.0 supports only online installation and release 4.0.0 supports only offline installation.

- The kernel version must be 4.4.19 or greater.

With Paragon Insights Release 4.0.0, for CentOS 8 server and RHEL 8, kernel upgrade is not required.

- The following system utilities must be installed manually if they are not already present:

tar, bash, ln, ssh-keygen, curl, vi, wget, openssl, openssh-server, and rsync

- Disable the swap memory by running the `swapoff -a` command to bring up kubelet service.

You must then remove swap memory entries from the `/etc/fstab` folder.

NOTE: You might need to reboot the server for this setting to take effect.

- Enter the following configuration line in the file `/etc/sysctl.conf`: `vm.max_map_count=262144`.

- On a scaled production server, we recommend that you configure the OS settings in the `limits.conf` and `sysctl.conf` files. These values set the soft and hard memory limits for influx DB memory requirements. If you do not set these limits, you might see errors such as “out of memory” or “too many open files” because of default system limits.

```
cat /etc/security/limits.conf
# End of file
*          hard    nofile    1048576
*          soft    nofile    1048576
root       hard    nofile    1048576
root       soft    nofile    1048576
influxdb   hard    nofile    1048576
influxdb   soft    nofile    1048576
```

```
cat /etc/sysctl.conf
fs.file-max = 2097152
vm.max_map_count=262144
fs.inotify.max_user_watches=524288
fs.inotify.max_user_instances=512
```

- Run the `sysctl -w net.bridge.bridge-nf-call-iptables=1` command on all nodes.

Add the `net.bridge.bridge-nf-call-iptables=1` to the file `/etc/sysctl.conf` to ensure that it persists across server reboots.

NOTE: You might need to reboot the server for this setting to take effect.

Web Browser Requirements

Paragon Insights is supported on the following 64-bit web browsers:

Table 1: Supported Web Browsers

Browser	Supported Version(s) (Macintosh)	Supported Version(s) (Windows)
Chrome	90 and later	90 and later

Table 1: Supported Web Browsers (Continued)

Browser	Supported Version(s) (Macintosh)	Supported Version(s) (Windows)
Firefox	83 and later	83 and later
Safari	14.0.3 and later	-

Network Requirements

- For Kubernetes-based installations, including multinode installations:
 - All nodes must run NTP or other time-synchronization at all times.
 - An Internet connection is required for all nodes during the initial Ubuntu (.deb) or CentOS/Red Hat (.rpm) software extraction process. This is not a requirement for the healthbot setup portion of the installation
 - One static IP address per node.
(Optional) A hostname that resolves to the corresponding IP address.
 - Ensure that internet protocol version 6 (IPv6) is enabled.
 - An SSH server must be running on all nodes.
 - All nodes must be in the same subnet.
 - For a multi-master installation:
 - You must always enter an odd number of master nodes.
 - You need a virtual IP address to configure high availability (HA) between the master nodes. This virtual IP address must be different from the virtual IP address that you specify to access various Paragon Insights services.
 - All master nodes must be in same subnet. Ensure that you place the master nodes on different racks so that there is no impact if there are power outages
 - You need to determine the number of master nodes before you start the installation process.
 - For a multinode installation, a virtual (unused) IP address in the same subnet as the nodes is needed. This is the address on which the Web GUI is accessed.

- A common SSH user name and password is needed for all nodes. The `healthbot setup` command (discussed later) must be run as this user.
- Docker version 18.09.3 or later is required

NOTE: Starting with Paragon Insights Release 4.0.0, Docker upgrade is not required.

Run any one of the following commands to verify the Docker version installed:

```
$ docker version
```

or

```
$ docker --version
```

NOTE: Verify the SELinux mode. If it is set to enforcing, change it to permissive. This is required to allow Docker commands to execute later in this procedure.

```
$ getenforce
Enforcing
$ sudo setenforce 0
$ getenforce
Permissive
```

- Open the following firewall ports, as appropriate:
 - JTI (native GPB), for telemetry collection - per your source and destination port settings
 - gRPC (OpenConfig), for telemetry collection - TCP port 32767
 - NETCONF/SSH, for telemetry collection - TCP port 830
 - SNMP, for telemetry collection - UDP port 161
 - Syslog messages - UDP port 514
 - NetFlow, for telemetry collection - UDP port of your choice.
Must be different for each NetFlow host.
 - sFlow, for telemetry collection - UDP port of your choice

NOTE: Default ports are listed above; adjust as needed if you use non-default ports.

- Enable these ports to allow intra-cluster and inter-cluster communication.

This ensures that the cluster setup does not crash due to SSH timeout.

See [Table 2 on page 10](#) for information and requirements on ports for single-master setups. You must enable these ports for single-master setups.

For more information on ports for multi-master setups, see [Table 3 on page 11](#). You must enable the ports listed in both [Table 2 on page 10](#) and [Table 3 on page 11](#) for multi-master setups.

Table 2: Ports for Single-Master and Multi-Master Setups

Direction	Ether Type	Internet Protocol	Port Range	Remote IP Prefix	Description
Ingress	IPv4	TCP	22	0.0.0.0/0	SSH
Ingress	IPv4	ICMP	any	0.0.0.0/0	ICMP probes
Ingress	IPv4	TCP	8080	0.0.0.0/0	Paragon Insights GUI and REST API server
Ingress	IPv4	TCP	6443		Communicate with worker nodes in the cluster
Ingress	IPv4	TCP	179		BGP used by calico for route discovery
Ingress	IPv4	TCP	10250		Kubelet API communication
Ingress	IPv4	TCP	8443		Kubernetes metrics server
Ingress	IPv4	TCP	7005		Paragon Insights common services

Table 2: Ports for Single-Master and Multi-Master Setups (Continued)

Direction	Ether Type	Internet Protocol	Port Range	Remote IP Prefix	Description
Ingress	IPv4	IPIP	any		Overlay network setup by calico
Egress	IPv4	any	any	0.0.0.0/0	Allow all IPv4 outbound traffic
Egress	IPv6	any	any	::/0	Allow all IPv6 outbound traffic

Table 3: Additional Ports for Multi-Master Setups

Direction	Ether Type	Internet Protocol	Port Range	Remote IP Prefix	Description
Ingress	IPv6	TCP	2379		etcd client requests
Ingress	IPv4	TCP	2380		etcd peer communication

Network Device Requirements

Junos Devices

Paragon Insights collects data from devices running Junos OS using multiple data collection methods, called sensors. Each sensor type requires a certain Junos OS version, and configuration added to the devices, to enable a connection to the Paragon Insights server.

Native GPB

- Junos OS Version: 15.1 or later
- Required configuration—configure a sensor profile for each relevant related rule in Paragon Insights:

```
##Streaming Server Profile
set services analytics streaming-server COLLECTOR-1 remote-address <HealthBot-server-address>
set services analytics streaming-server COLLECTOR-1 remote-port 22000
```



```

##Export Profile
set services analytics export-profile EXP-PROF-1 local-address <local-router-IP>
set services analytics export-profile EXP-PROF-1 local-port 22001
set services analytics export-profile EXP-PROF-1 reporting-rate 30
set services analytics export-profile EXP-PROF-1 format gpb
set services analytics export-profile EXP-PROF-1 transport udp
##Sensor Profile
set services analytics sensor SENSOR-1 server-name COLLECTOR-1
set services analytics sensor SENSOR-1 export-name EXP-PROF-1
set services analytics sensor SENSOR-1 resource <resource> # example /junos/system/linecard/
interface/

```

See [Configuring a Junos Telemetry Interface Sensor](#) for more information.

NetFlow (IPFIX)

- Junos OS Version: 14.1R1 or later for MX Series Routers. For complete Junos version and platform support information, see:
 - [Configuring Flow Aggregation on MX, M, vMX and T Series Routers and NFX250 to Use Version 9 Flow Templates](#)
 - [Configuring Flow Aggregation to Use IPFIX Flow Templates on MX, vMX and T Series Routers, EX Series Switches and NFX250, and SRX Devices](#)
 - [Understanding Inline Active Flow Monitoring - TechLibrary](#)
- Required configuration—
 - Configure a NetFlow v9 or IPFIX template
 - Apply the template to enable traffic sampling
 - Associate the sampling instance with the FPC
 - Specify which traffic interface to sample

The following samples are for an IPFIX configuration. Lines that start with “##” are comments and are used to point out details in the configuration.

IPFIX Template Configuration

```

set services flow-monitoring version-ipfix template IPv4-TEMPLATE ipv4-template

```

Apply IPFIX Template to Enable Traffic Sampling

```
set forwarding-options sampling instance IPFIX-IPv4-INSTANCE input rate 10 set forwarding-
options sampling instance IPFIX-IPv4-INSTANCE family inet output flow-server 10.XX.XX.200 port
2055
set forwarding-options sampling instance IPFIX-IPv4-INSTANCE family inet output flow-server
10.XX.70.XX version-ipfix template IPv4-TEMPLATE
set forwarding-options sampling instance IPFIX-IPv4-INSTANCE family inet output inline-jflow
source-address 198.XX.XX.1
```

10.XX.70.XX = Paragon Insights server

port 2055; use this value in Paragon Insights GUI (device group config)

inline-jflow = Enable inline flow monitoring for traffic from the designated address

198.XX.XX.1 = traffic interface that does the exporting; use this value in Paragon Insights GUI (device config)

Associate Sampling Instance with the FPC

```
set chassis fpc 0 sampling-instance IPFIX-IPv4-INSTANCE
```

Specify the Traffic Interface to Sample and Direction of Sampled Traffic

```
set interfaces ge-0/0/0 unit 0 family inet sampling input
set interfaces ge-0/0/0 unit 0 family inet sampling output
```

OpenConfig

- Junos OS Version: 16.1 or later
 - The OpenConfig sensor requires that the Junos device have the OpenConfig and network agent packages installed. These packages are built into Junos OS Releases 18.2X75, 18.3, and later. For releases between 16.1 and 18.2X75 or 18.3, you must install the packages.

To verify whether you have these packages, enter the following command:

To verify whether you have these packages, enter the following command:

```
user@host> show version | match "Junos:|openconfig|na telemetry"
```

```
Junos: 19.2R1.8
```

```
JUNOS na telemetry [19.2R1.8]
JUNOS Openconfig [19.2R1.8]
```

See [Understanding OpenConfig and gRPC on Junos Telemetry Interface](#) for more information.

- Network agent is not supported on PPC platforms (MX104, MX80, and so on)
- Refer to the following topics of the Junos Telemetry Interface User Guide if the OpenConfig and network agent packages are not installed.
 - To install the OpenConfig package, see [Installing the OpenConfig Package](#).
 - To install the network agent manager package, see [Installing the Network Agent Package \(Junos Telemetry Interface\)](#).
- After you have installed the packages, enable OpenConfig on the MX240 by running the following command:

```
set system services extension-service request-response grpc clear-text port number
```

Network agent is not supported on PPC platforms (MX104, MX80, and so on)

iAgent (NETCONF)

- Junos OS Version: 11.4 or later
- Required configuration:

```
set system services netconf ssh
```

SNMP

- Junos OS Version: Any release
- Required configuration:

```
set snmp community public
```

Syslog

- Junos OS Version: Any release

- Required configuration:

```
set system syslog host 10.x.x.1 any any
set system syslog host 10.x.x.1 allow-duplicates
set system syslog host 10.x.x.1 structured-data
```

10.x.x.1 = Paragon Insights server

BEST PRACTICE: Structured syslog is highly recommended because it avoids text parsing by the Paragon Insights server.

Cisco Device Support

Paragon Insights can collect telemetry data from Cisco IOS XR devices. To use these devices with Paragon Insights, you must configure the gRPC server and the openconfig-interfaces sensors. Paragon Insights does not automatically configure these for you.

The following example shows a sensor group `sg1` created for gRPC dial-in configuration with the YANG model for interfaces. An `hbot_interfaces` subscription associates the `sg1` sensor group with an interval of 10 seconds to stream data.

NOTE: The following reference configuration is loaded on a device running Cisco IOS XR software version 6.3.2.

```
!
grpc
port 32767
!
telemetry model-driven
sensor-group sg1
  sensor-path openconfig-interfaces:interfaces/interface
!
subscription hbot_interfaces_
  sensor-group-id sg1 sample-interval 10000
!
!
ssh server v2
end
```

Multinode Installation

Starting with release 3.0.0 Paragon Insights uses Kubernetes for all HA/multinode installations. In order to install Paragon Insights on more than one server, you must install using Kubernetes. All of the needed configuration for the clusters is performed by Kubernetes. During the setup phase of the installation process, you must identify the Kubernetes master node(s) and worker node(s).

Starting with Release 4.0.0, while installing Paragon Insights you can choose to have multiple master nodes. While running the `healthbot setup` command, you are prompted to specify hostnames or IP addresses of the master nodes. If you choose to have multiple master nodes, you must also specify the virtual IP address that is required for configuring high availability (HA) between the master nodes. If you are using the silent installer, in the configuration file you can specify the virtual IP address in the **master_virtual_ip** field.

Starting from Release 4.0.0, while installing Paragon Insights you can specify multiple virtual IP addresses (unused) so that you can connect to various services in Paragon Insights and thereby monitor devices that are in different subnets. If you are using the silent installer, in the configuration file you can specify multiple virtual IP addresses in the **load_balancer_ip** field.

RELATED DOCUMENTATION

[Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X](#) | 16

[Using the Silent Installer](#) | 33

Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X

IN THIS SECTION

- [Installing Paragon Insights On Ubuntu](#) | 19
- [Installing Paragon Insights On CentOS](#) | 23

This topic provides information on how to:

- Perform a fresh installation of Paragon Insights releases 4.0.0, 4.1.0, 4.2.0, or 4.3.0.
- Seamlessly upgrade Paragon Insights from an earlier 4.X release to a later release.

NOTE: If you are currently using Paragon Insights with Docker Compose-based installation, then you must perform a fresh installation of Release 4.X. If you are using Paragon Insights with Kubernetes-based installation, follow the instructions provided in the ["Upgrading Paragon Insights Release 3.X with Kubernetes-based Installation to Release 4.X"](#) on page 46 topic.

You can perform an online or offline installation of Paragon Insights.

- **Online Installation**—You need an active internet connection to run an online installation of Paragon Insights. You must download the necessary online package (.deb for Ubuntu or .rpm for CentOS/RedHat) from the Juniper Networks [Software Downloads](#) page and install them. You can then proceed to run the `sudo healthbot setup` and `healthbot start` commands.
- **Offline Installation**—You do not need an active internet connection during offline installation. However, you need an active internet connection to download and install the necessary Paragon Insights package (.deb for Ubuntu or .rpm for CentOS/RedHat) from the Juniper Networks [Software Downloads](#) page and install them. After you install the package (.deb or .rpm), you must download the offline install package to the `/var/local/healthbot` directory and rename the file to **healthbot-offline.tgz**. You can then run the `sudo healthbot setup` and `healthbot start` commands without an active internet connection.

The Paragon Insights (formerly HealthBot) software package is available for download as a Debian (.deb) file for installation on Ubuntu, or a Red Hat Package Manager (.rpm) file for installation on CentOS and Red Hat Enterprise Linux (RHEL).

Before You Begin

You will need the following details for the installation:

- **Deployment type**—Single node or multinode installation. Multinode installations are useful for load distribution and scaling.
- **SSL certificate and private key**—Supply your own key, or have Paragon Insights create one for you.
- **Host IP address**—The server IP address, for SSH and Web UI access.
- If you are using Paragon Insights to install a new Kubernetes cluster, ensure that the hostnames of the nodes resolve to the IP addresses of the nodes used in the Kubernetes cluster. You can use DNS to map the hostname to the IP address or add the entries to the local `/etc/hosts` file.

- If you are installing Paragon Insights into an existing Kubernetes cluster, you need information about the existing Kubernetes cluster:
 - Docker registry name
 - The path to a kubeconfig file configured with the existing cluster details
 - A user account with administrator privileges for the Kubernetes cluster

NOTE: The following points regarding installation on an existing Kubernetes cluster must be considered:

- Paragon Insights has only been qualified with the Calico container network interface (CNI). It should work with other CNI plugins, but has not been tested.
- If your Kubernetes cluster uses other CNI plugins, such as Flannel, you might notice long playbook deployment times on some Kubernetes versions. One potential workaround, is to disable transmit (tx) and receive (rx) checksum offloading on VXLAN interfaces. An example of this, using Flannel: `ethtool --offload flannel.1 tx off rx off`.

For discussions regarding the latency issue, see:

- [Kubernetes Issue](#)
- [Flannel Issue](#)
- [LKML.ORG](#)
- If there are a large number of device groups in Paragon Insights, there is the possibility that some of the Kubernetes pods might not get scheduled if the nodes are saturated by the maximum pods per node limit. By default, this limit is 110 pods per node in most Kubernetes distributions.

As a workaround, you can increase the maximum pods per node setting. Refer to the documentation from your Kubernetes distribution for details on how to modify this setting.
- If your CNI plugin is configured with a static IP CIDR block for each node, make sure the number of IP addresses in the block allocated to the node is at least double the size of the maximum pods per node setting.

Additional requirements:

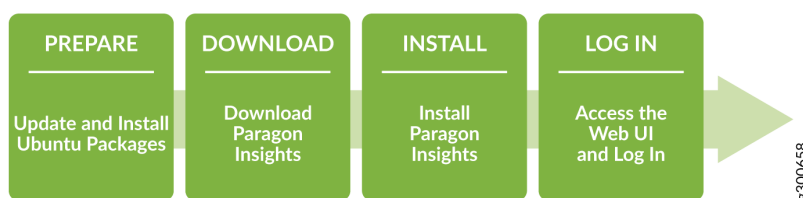
- You must have a Juniper.net user account with permissions to download the Paragon Insights software package.
- The Paragon Insights server must have access to the Internet during the software extraction process of installation (.deb for Ubuntu or .rpm for CentOS/Red Hat).

Paragon Insights creates a `/var/local/healthbot/` directory as part of the installation process. This directory contains all the Paragon Insights-related files.

Installing Paragon Insights On Ubuntu

The general workflow for installing Paragon Insights on Ubuntu is shown in [Figure 1 on page 19](#).

Figure 1: Installation Workflow - Paragon Insights on Ubuntu



NOTE: For multinode installations, perform the following tasks only on the primary node. If you are logged in as root or using root privileges, remove the `sudo` portion of each command below.

Part 1 - PREPARE

1. Ensure that you have at least 60GB of free disk space. We recommend 100GB of free disk space.
2. Set the environment variable, `HB_EXTRA_MOUNT1`, to allow Paragon Insights to access directories outside of the `/var/local/healthbot` directory.

To set the environment variable:

```
root@hb_server:/ export HB_EXTRA_MOUNT1=/root/.kube/config
```

3. Set the environment variables— `HB_IAM_SKIP_MAIL_VERIFICATION` and `HB_IAM_DISABLE_SMTP_SETTINGS`— as false to enable e-mail verification of users.

To set the environment variable:

```
root@hb_server:/ export HB_IAM_SKIP_MAIL_VERIFICATION=false
root@hb_server:/ export HB_IAM_DISABLE_SMTP_SETTINGS=false
```


4. Ensure that the Ubuntu package lists on your host system are latest.

```
$ sudo apt-get update
```

5. (Optional) Install the **wget** package. This tool will be used later to download the Paragon Insights software package. On some installations, **wget** is installed by default.

```
$ sudo apt-get install -y wget
```

6. Install Docker CE.

The following commands install the latest stable version on x86 machines:

```
$ sudo apt-get install -y apt-transport-https ca-certificates curl gnupg-agent software-properties-common
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
$ sudo apt-get update
$ sudo apt-get install -y docker-ce docker-ce-cli containerd.io
```

To verify that Docker is installed and running:

```
$ docker run hello-world
```

To verify the Docker version installed:

```
$ docker version
```

or

```
$ docker --version
```

For more information, see [Get Docker CE for Ubuntu](#) and [Manage Docker as a Non-root User](#).

7. (Optional) Add your user account to the Docker group. This is required for a later installation step. If installing and running with root privileges, this step is not required.

```
$ sudo usermod -aG docker $USER
```

Log out of the current session, and log back in.

Part 2 - DOWNLOAD

1. Download the Paragon Insights package from the Juniper Networks [Software Download](#) page to a temporary directory (like /var/tmp/) on the server. Note that downloading software requires a Juniper.net account.

```
wget -O /<temp-directory>/healthbot-<version>.deb "<URL-from-the-software-download-page>"
```

NOTE: You can also download the Paragon Insights package locally and push it to the server.

Part 3 - INSTALL

1. Install the .deb package using the following format:

```
$ sudo apt-get install -y /<path-to-deb-file>/healthbot-<version>.deb
```

Paragon Insights checks that prerequisite software is installed on your host device during this step. If any required software is not found, Paragon Insights will prompt you before installing those missing software packages.

For example:

```
root@ubuntu:/var/tmp#
```

```
$ sudo apt-get install ./healthbot-4.3.0-1.deb

Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'healthbot' instead of './healthbot-4.3.0-1.deb'
The following NEW packages will be installed:
  healthbot
0 upgraded, 1 newly installed, 0 to remove and 82 not upgraded.
Need to get 0 B/8,418 MB of archives.
After this operation, 20.3 GB of additional disk space will be used.
Get:1 /homes/sharanyab/healthbot-4.3.0-1.deb healthbot all 4.3.0-1 [8,418 MB]
(Reading database ... 142792 files and directories currently installed.)
Preparing to unpack .../healthbot-4.3.0-1.deb ...
```

```
Unpacking healthbot (4.3.0-1) ...
Setting up healthbot (4.3.0-1) ...
```

NOTE: If you see the following error when running the `sudo apt-get` command, you can ignore it:

“Can't drop privileges for downloading as file '/home/user/healthbot-3.0.0-1.deb' couldn't be accessed by user '_apt'. - pkgAcquire::Run (13: Permission denied)”

For more information, see [Bug 1543280](#).

2. Download the offline install package that is required for offline installation from the Juniper Networks [Software Downloads](#) page. Once downloaded, change the name of the file to **healthbot-offline.tgz** and place it in the **/var/local/healthbot** directory.
3. Enter `sudo healthbot setup` to configure your installation. You can also use `sudo healthbot -v setup` to display more detailed information.

While running the `sudo healthbot setup` command, you are prompted to answer several questions as part of the setup process. For more information, see ["Paragon Insights Interactive Installation Prompts"](#) on page 30.

An example of the setup command from a multinode installation:

An example of the setup command from a single-node installation:

Paragon Insights installation is now complete. If any errors occurred, they will be listed in the outputs above.

4. Enter `healthbot start` to start Paragon Insights services including the Web UI.

For example:

5. Enter `healthbot status` to verify that the Paragon Insights services are up and running.

For example:

```
$ healthbot status
```

Name of service	Status

alerta	Up
api-server	Up
argo-server	Up

auditlog	Up
config-server	Up
configmanager	Up
debugger	Up
gateway	Up
grafana	Up
hb-proxy-syslog	Up
hbmon	Up
iam	Up
inference-engine	Up
influxdb(10.221.133.53)	Up
ingest-snmp-proxy	Up
license-client	Up
metallb(controller)	Up
metallb(speaker)	Up
mgd	Up
node-exporter	Up
postgres	Up
redis(rfr-redis)	Up
redis(rfs-redis)	Up
redisoperator	Up
reports	Up
tsdb-shim	Up
udf-farm	Up
workflow-controller	Up

Part 4 - LOG IN

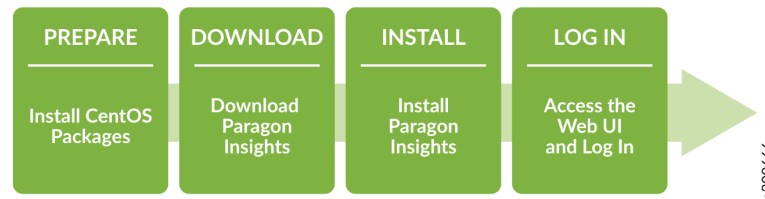
1. Open a browser and enter the <https://server-IP:8080> URL to access the Paragon Insights UI, where *server-IP* is the virtual IP address you entered, to connect to Paragon Insights services, during installation. Log in to the Paragon Insights Web UI using the credentials: **Username:** *admin*, **Password:** *Admin123!*. These are one-time credentials. When you enter them, Paragon Insights prompts you to change the password and gives instructions about the recommended length, case changes, and so on.

NOTE: Starting from Paragon Insights Release 4.1.0, username is not case sensitive.

Installing Paragon Insights On CentOS

The general workflow for installing Paragon Insights on CentOS is shown in [Figure 2 on page 24](#).

Figure 2: Installation Workflow - Paragon Insights on CentOS



NOTE: For multinode installations, perform the following tasks only on the primary node. If you are logged in as root or using root privileges, remove the `sudo` portion of each command below.

Part 1 - PREPARE

1. This procedure assumes you are installing Paragon Insights using a non-root user. If this user does not already have `sudo` privileges, set them now.

```

$ su
Password: <root-pwd>
# sudo usermod -aG wheel <non-root-username>
<stay logged in as root>

```

2. While logged in as root, set `secure_path` in the `/etc/sudoers` file to include `/usr/local/bin/` so that `sudo` will be able to locate the Paragon Insights script during the healthbot setup step, later in this procedure.

```

# visudo
<scroll down to Defaults secure_path = /sbin:/bin:/usr/sbin:/usr/bin>
<type i to enter edit mode>
<scroll to end of line and add :/usr/local/bin >
<type Esc, then :wq, and press Enter>
# exit
exit
$ <Log out of the current session, and log back in>

```

3. Install the **epel-release** repository. This is required for installing Python3 and other packages.

```

$ sudo yum install -y epel-release

```

4. Install the yum-utils package used for handling repositories and extending package management.

```
$ sudo yum install -y yum-utils
```

5. Install the wget package. This tool will be used later to download the Paragon Insights software package.

```
$ sudo yum install -y wget
```

6. Install Docker CE.

```
$ sudo yum install -y yum-utils \ device-mapper-persistent-data \ lvm2
$ sudo yum-config-manager \ --add-repo \ https://download.docker.com/linux/centos/docker-
ce.repo
$ sudo yum install docker-ce docker-ce-cli containerd.io
$ sudo systemctl start docker
$ sudo systemctl enable docker
```

To verify that Docker is installed and running:

```
$ sudo docker run hello-world
```

For full instructions, see [Get Docker CE for CentOS](#) and [Manage Docker as a Non-root User](#).

Part 2 - DOWNLOAD

1. Download the Paragon Insights package from the Juniper Networks [Software Download](#) page to a temporary directory (like /var/tmp) on the server. Note that downloading software requires a Juniper.net account.

```
wget -O /<temp-directory>/healthbot-<version>.noarch.rpm "<URL-from-the-software-download-
page>"
```

NOTE: You can also download the Paragon Insights package locally and push it to the server.

Part 3 - INSTALL

1. Install the .rpm package using the following format:

```
$ sudo yum install -y /<path-to-rpm-file>/healthbot-<version>.noarch.rpm
```

Paragon Insights checks that prerequisite software is installed on your host device during this step. If any required software is not found, Paragon Insights will prompt you before installing those missing software packages. Paragon Insights uses an Internet connection to download any missing packages.

An example, from CentOS:

```
$ sudo yum install -y healthbot-4.3-0.1.noarch.rpm
```

```
Last metadata expiration check: 1:22:13 ago on Tue 23 Nov 2021 03:27:33 PM UTC.
```

```
Dependencies resolved.
```

```
=====
Package
Architecture
Version
Repository
```

```
Size
=====
```

```
Installing:
```

```
healthbot
```

```
noarch
```

```
4.3-0.1
```

```
@commandline
```

```
5.3 G
```

```
Transaction Summary
```

```
=====
Install 1 Package
```

```
Total size: 5.3 G
```

```
Installed size: 16 G
```

```
Downloading Packages:
```

```
Running transaction check
```

```
Transaction check succeeded.
```

```
Running transaction test
```

```
Transaction test succeeded.
```

```
Running transaction
```

```
Preparing :
```

```

1/1
Running scriptlet :
healthbot-4.3-0.1.noarch

1/1
Installing :
healthbot-4.3-0.1.noarch

1/1
Running scriptlet :
healthbot-4.3-0.1.noarch

1/1
Verifying :
healthbot-4.3-0.1.noarch

1/1

Installed:
healthbot-4.3-0.1.noarch

'infra_helm_tiller_image': ["gcr.io/kubernetes-helm/tiller:v2.16.1", "svl-
artifactory.juniper.net/healthbot-registry/third_party/k8s/tiller:v2.16.1"],
Complete!

```

2. Paragon Insights supports offline installation for Kubernetes-based installations.

Download the offline install package that is required for offline installation from the Juniper Networks [Software Downloads](#) page. Once downloaded, change the name of the file to **healthbot-offline.tgz** and place it in the **/var/local/healthbot** directory.

3. Enter `sudo healthbot setup` to configure your installation. You can also use `sudo healthbot -v setup` to display more detailed information.

While running the `sudo healthbot setup` command, you are prompted to answer several questions as part of the setup process. For more information, see ["Paragon Insights Interactive Installation Prompts" on page 30](#).

An example of the setup command from a multinode installation:

An example of the setup command from a single-node installation:

Paragon Insights installation is now complete. If any errors occurred, they will be listed in the outputs above.

4. Enter `healthbot start` to start Paragon Insights services including the Web UI.

```
healthbot start
Stopping HealthBot
Generated service manifests
Updated orchestrator details
Successfully started redis
Started configmanager successfully
Successfully published files to config manager
Restarted config manager successfully
Started configmanager successfully
Successfully generated manifest file
Successfully deployed TSDB
Successfully published tsdb manifest files
Started postgres cluster successfully
Deployed services successfully
Reconciling load balancer successfully
Started load balancer successfully
Started all services successfully
Created default databases
UI can be accessed at following URL(s): https://10.xxx.xxx.186:8080
```

5. Enter `healthbot status` to verify that the Paragon Insights services are up and running.

For example:

```
$ healthbot status
```

Name of service	Status

alerta	Up
api-server	Up
argo-server	Up
auditlog	Up
config-server	Up
configmanager	Up
debugger	Up
gateway	Up
grafana	Up
hb-proxy-syslog	Up
hbmon	Up

iam	Up
inference-engine	Up
influxdb(10.221.133.53)	Up
ingest-snmp-proxy	Up
license-client	Up
metallb(controller)	Up
metallb(speaker)	Up
mgd	Up
node-exporter	Up
postgres	Up
redis(rfr-redis)	Up
redis(rfs-redis)	Up
redisoperator	Up
reports	Up
tsdb-shim	Up
udf-farm	Up
workflow-controller	Up

6. Verify that SELinux is set to Enforcing. If not, change it to Enforcing..

```
$ getenforce
Permissive
$ sudo setenforce 1
$ getenforce
Enforcing
```

Part 4 - LOG IN

1. Open a browser and enter the <https://server-IP:8080> URL to access the Paragon Insights UI, where *server-IP* is the virtual IP address you entered, to connect to Paragon Insights services, during installation. Log in to the Paragon Insights Web UI using the credentials: **Username:** *admin*, **Password:** *Admin123!*. These are one-time credentials. When you enter them, Paragon Insights prompts you to change the password and gives instructions about the recommended length, case changes, and so on.

NOTE: Starting from Paragon Insights Release 4.1.0, username is not case sensitive.

Paragon Insights Interactive Installation Prompts

IN THIS SECTION

- Prompts for Paragon Insights Release 4.3.0 | 30
- Prompts for Paragon Insights Release 4.2.0 | 31
- Prompts for Paragon Insights Release 4.0.0 and 4.1.0 | 32
- Prompts for HealthBot Release 3.2.0 (Docker Compose Installation) | 32
- Prompts for HealthBot Release 3.2.0 (Kubernetes-based Installation) | 33

While running the `sudo healthbot setup` command, you are prompted to answer several questions as part of the setup process.

The default values for each question are shown in square brackets []. If a choice is required, the default option is capitalized. For example, [Y/n].

For information on installing Paragon Insights (formerly HealthBot), see ["Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X" on page 16](#).

Prompts for Paragon Insights Release 4.3.0

- Paragon Insights free model/freemium license tier is not available in this version. A valid license (trial or commercial) is needed to use Paragon Insights features.

The format of license key has been changed due to business reasons in this version. The system may have contained the older format of license keys, which are not valid now. You can log into the Juniper Agile Licensing Portal and choose to revoke the existing license and then re-activate the license to a new software version (that will provide you with a new format license key). This newly activated license can be added to the system post upgrade.

Do you want to proceed? [Y/n]

- Do you want system to create and use a self-signed SSL certificate? [Y/n]
- Enter hostnames/ip address of the kubernetes master node(s):
- Enter comma separated hostnames/ip addresses of kubernetes worker nodes.

- Do you want HealthBot services to run on master node? [Y/n]

All nodes must be in the same subnet. Abort the setup if all the nodes are not within same subnet.

- Please enter password for user "root" to connect to the nodes:

You are prompted to confirm the password:

Confirm password:

- Enter comma separated virtual IP(s):

NOTE: A virtual IP address is required to connect to Paragon Insights services. If you have multiple interfaces connected to different networks, specify a virtual IP address from each network that you need connectivity from.

Prompts for Paragon Insights Release 4.2.0

- Paragon Insights free model/freemium license tier is not available in this version. A valid license (trial or commercial) is needed to use Paragon Insights features.

Do you want to proceed? [Y/n]

- Do you want system to create and use a self-signed SSL certificate? [Y/n]
- Enter hostnames/ip address of the kubernetes master node(s):
- Enter comma separated hostnames/ip addresses of kubernetes worker nodes.
- Do you want HealthBot services to run on master node? [Y/n]

All nodes must be in the same subnet. Abort the setup if all the nodes are not within same subnet.

- Please enter password for user "root" to connect to the nodes:

You are prompted to confirm the password:

Confirm password:

- Enter comma separated virtual IP(s):

NOTE: A virtual IP address is required to connect to Paragon Insights services. If you have multiple interfaces connected to different networks, specify a virtual IP address from each network that you need connectivity from.

Prompts for Paragon Insights Release 4.0.0 and 4.1.0

- Do you want system to create and use a self-signed SSL certificate? [Y/n]
- Do you want to install on kubernetes? [y/N]

Enter y.

- Enter hostnames/ip address of the kubernetes master node(s):
- Enter comma separated hostnames/ip addresses of kubernetes worker nodes.
- Do you want HealthBot services to run on master node? [Y/n]

All nodes must be in the same subnet. Abort the setup if all the nodes are not within same subnet.

- Please enter password for user "regress" to connect to the nodes:

You are prompted to confirm password:

Confirm password:

- Enter comma separated virtual IP(s):

NOTE: A virtual IP address is required to connect to Paragon Insights services. If you have multiple interfaces connected to different networks, specify a virtual IP address from each network that you need connectivity from.

Prompts for HealthBot Release 3.2.0 (Docker Compose Installation)

- Do you want system to create and use a self-signed SSL certificate? [Y/n]
- Do you want to install on kubernetes? [y/N]

Enter N to begin Docker Compose installation.

- Please enter host IP address. This must be the IP address for establishing SSH connection to the host:
- (Repeated)Do you want system to create and use a self-signed SSL certificate? [Y/n]
- Use IPv6 docker network? [y/N]
- Enter subnet for IPv4 Docker network (optional):

Prompts for HealthBot Release 3.2.0 (Kubernetes-based Installation)

- Do you want system to create and use a self-signed SSL certificate? [Y/n]
 - Do you want to install on kubernetes? [y/N]
- Enter y to begin kubernetes-based installation.
- Do you have Kubernetes cluster already? [Y/n]
 - Enter hostnames/ip address of the kubernetes master node(s):
 - Enter comma separated hostnames/ip addresses of kubernetes worker nodes.
 - Do you want HealthBot services to run on master node? [Y/n]

RELATED DOCUMENTATION

[Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X](#) | 16

Using the Silent Installer

IN THIS SECTION

- [How to Perform a Silent Installation](#) | 41

This topic provides information on how to:

- Perform a fresh installation of Paragon Insights releases 4.0.0, 4.1.0, 4.2.0, or 4.3.0 by using the silent installer.
- Seamlessly upgrade Paragon Insights from an earlier 4.X release to a later release.

NOTE: If you are currently using Paragon Insights with Docker Compose-based installation, then you must perform a fresh installation of Release 4.X. If you are using Paragon Insights with Kubernetes-based installation, follow the instructions provided in the ["Upgrading Paragon Insights Release 3.X with Kubernetes-based Installation to Release 4.X"](#) on page 46 topic.

Paragon Insights (formerly HealthBot) provides a silent installer as a convenient way to streamline the installation process without requiring user intervention.

To use the silent installer, you can use the default configuration file, included in the downloaded .deb or .rpm package, or create your own custom configuration. The default configuration file, **healthbot.conf**, is located on the Paragon Insights server in the **/var/local/healthbot/** after the interactive installation is complete.

The following is an example configuration file for single-node installations.

```
##### Multi-node settings for Kubernetes installations #####
# Absence of multi-node key indicates single node installation.
# Un-comment below for multi-node installation. Refer documentation
# for more information. This is not a mandatory key.
# Use the use_cluster stanza if you already have a kubernetes cluster
# If you want HealthBot to install the kubernetes cluster, use the cluster stanza

multi_node:
  cluster:
    master_nodes: ['10.xx.xx.28']
    worker_nodes: []
    allow_master_scheduling: True
    load_balancer_ip: ['10.xx.xx.28']
    master_virtual_ip: None
    ssh_user: root
    ssh_pw: '<Insert password>'
    pod_cidr: 10.xx.xx.0/16
    service_cidr: 10.xx.xx.0/12

# use_cluster:
# registry: <Insert registry name>
```


[illegible]


```

    service_cidr: 10.xx.xx.0/12

#   use_cluster:
#       registry: <Insert registry name>
#       kubeconfig: <Insert path to kubeconfig file>
#       # Uncomment the following stanza if want HealthBot to install loadbalancer
#       load_balancer:
#           load_balancer_ip: <Insert a Virtual IP here>

##### Single node settings for docker-compose installations #####
##### Container network settings #####
# Refer: https://docs.docker.com/engine/reference/commandline/network_create/.
# For single-node installation, we use default driver.

# Un-comment below if you want to control subnet of healthbot's containers.
# If devices in your network have IP address from the private IP address
# block, use this option to avoid IP address conflict of containers with
# network devices.
# This is not a mandatory key.

# network:
#   subnet: 10.xx.xx.0/16
#   gateway: 10.xx.xx.2

##### Master settings #####
# Specify below host's ip address on which sshd is listening on the default
# port. Don't use localhost or loop-back address. This is a mandatory key.

# host_ip: <ip>

# Specify below HTTPS certificate and private key file location.
# This is a mandatory key.

https:
    cert_file: /var/local/healthbot/certs/cert.pem
    key_file: /var/local/healthbot/certs/key.pem

*****
$ sudo healthbot setup --c <path-to-file>/healthbot.conf
i Paragon Insights free model/freemium license tier is not available in this version. A valid
license(trial or commercial) is needed to use Paragon Insights features

```



```
# pod_cidr: 10.xx.xx.0/16
# service_cidr: 10.xx.xx.0/12
```

to:

```
multi_node:
  cluster:
    master_nodes: [10.xx.xx.200]
    worker_nodes: [10.xx.xx.82]
    allow_master_scheduling: True
    load_balancer_ip: 10.xx.xx.201
    ssh_user: hbuser
    ssh_pw: password1
    pod_cidr: 10.xx.xx.0/16
    service_cidr: 10.xx.xx.0/12
```

To run the silent installer package:

How to Perform a Silent Installation

1. Follow the steps for installing the Paragon Insights package as described in ["Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X" on page 16](#) until you reach the setup step.
2. Edit your custom configuration file and change the setting values as needed. You can use the file `/var/local/healthbot/healthbot.conf` as a starting point.
3. Run the silent installer using the following command:

```
$ sudo healthbot setup -c <path and filename to custom configuration file>
```

For example:

```
$ sudo healthbot setup -c $HOME/healthbot/healthbot-custom.conf
```

NOTE: The `sudo healthbot setup -c` command prompts you for user credentials. If you do not want to be prompted, add the credential information to your custom configuration file. For example:

```
username: <username> password: <password>
```

Any errors that occur during the installation are displayed in the terminal window.

RELATED DOCUMENTATION

[Paragon Insights Installation Requirements | 2](#)

[Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X | 16](#)

Starting and Stopping Paragon Insights

To start Paragon Insights (formerly HealthBot), enter the following command:

```
$ healthbot start
```

This command starts the common services, starts the services for all device groups and networks groups, and then launches the Paragon Insights Web UI. For example:

```
$ healthbot start
Stopping HealthBot
Configured secrets
Generated service manifests
Updated orchestrator details
Successfully started redis
Started configmanager successfully
Successfully published files to config manager
Restarted config manager successfully
Started configmanager successfully
Successfully generated manifest file
Successfully deployed TSDB
Successfully published tsdb manifest files
Started postgres cluster successfully
```

```

Successfully updated SNMP engine data
Deployed services successfully
Reconciling load balancer successfully
Started load balancer successfully
Started all services successfully
Created default databases
UI can be accessed at following URL(s): https://10.xx.xx.17:8080

```

You can use the `healthbot -v start` command to display detailed information on your screen.

To stop the Paragon Insights application, enter the following command:

```
$ healthbot stop
```

This command stops all Paragon Insights services, including the device group and network group services. This command does not close the Paragon Insights Web UI if it is open. For example:

```

$ healthbot stop
Deleting healthbot namespace
All services stopped successfully
Removed group common services

```

RELATED DOCUMENTATION

[Uninstalling or Reconfiguring Paragon Insights | 65](#)

[Paragon Insights Command-Line Options | 70](#)

Checking Paragon Insights Service Status

A dedicated service, `healthbot_hbmon`, monitors the state of Paragon Insights (formerly HealthBot). It monitors the CPU and memory usage of the individual Paragon Insights services in terms of percentages, and triggers an alarm at predefined settings. Alarms are viewable on the Paragon Insights Web UI. For example, for both CPU and memory usage:

- 100% or greater is critical.

- 95-100% is major.
- 75-95% is minor.

If the state goes from critical to normal, the alarm is automatically closed. For more information about monitoring alarms, see *Alerts and Notifications*.

To view the default Paragon Insights services, use the `healthbot status` command:

```
$ healthbot status
```

Name of service	Status

alerta	Up
api-server	Up
argo-server	Up
auditlog	Up
config-server	Up
configmanager	Up
debugger	Up
gateway	Up
grafana	Up
hb-proxy-syslog	Up
hbmon	Up
iam	Up
inference-engine	Up
influxdb(10.221.133.53)	Up
ingest-snmp-proxy	Up
license-client	Up
metallb(controller)	Up
metallb(speaker)	Up
mgd	Up
node-exporter	Up
postgres	Up
redis(rfr-redis)	Up
redis(rfs-redis)	Up
redisoperator	Up
reports	Up
tsdb-shim	Up
udf-farm	Up
workflow-controller	Up

Similarly, the `healthbot status --device-group healthbot` and `healthbot status --network-group healthbot` commands show the default Paragon Insights services. For example:

```
$ healthbot status --device-group healthbot
Name of service          Status
-----
analytical-engine        Up
hb-syslog                 Up
iagent                   Up
itsdb                    Up
job-scheduler-training   Up
jtimon                   Up
resource-discovery       Up
```

```
$ healthbot status --network-group healthbot
Name of service          Status
-----
itsdb                    Up
resource-discovery       Up
```

After the device groups and network groups are defined, you can view their services. The following example shows Paragon Insights services running on a device group named `edge` and a network group named `13vpn`:

```
$ healthbot status --device-group edge
Name of service          Status
-----
analytical-engine        Up
iagent                   Up
itsdb                    Up
job-scheduler-training   Up
jtimon                   Up
resource-discovery       Up
```

```
$ healthbot status --network-group 13vpn
Name of service          Status
-----
kapacitor                Up
analytical-engine        Up
```

The Paragon Insights Web UI displays the names of the device groups and network groups that you created.

RELATED DOCUMENTATION

[Paragon Insights Command-Line Options](#) | 70

Alerts and Notifications

Upgrading Paragon Insights Release 3.X with Kubernetes-based Installation to Release 4.X

NOTE: You can upgrade only Paragon Insights (formerly HealthBot) Release 3.X with Kubernetes-based installation to Release 4.X. You cannot upgrade a Docker Compose-based installation to Release 4.X.

If you are currently using Paragon Insights with Docker Compose-based installation, then you must perform a fresh installation of Release 4.X. To perform a fresh installation of Paragon Insights or to upgrade Paragon Insights 4.X to a later release, see ["Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X" on page 16](#) or ["Using the Silent Installer" on page 33](#) topics.

To upgrade from Paragon Insights Release 3.x with Kubernetes-based installation to Paragon Insights Release 4.x:

1. Take a backup of the **keycloak-hb.json** file. To take a backup:
 - a. Identify the keycloak pod in the namespace.

```
/var/local/healthbot/healthbot k get pods | grep keycloak
```

- b. Connect to the keycloak pod.

```
/var/local/healthbot/healthbot k exec -it <pod-name> bash
```

- c. Run the following command to generate the backup file:

```
/opt/jboss/keycloak/bin/standalone.sh -Dkeycloak.migration.action=export -
Dkeycloak.migration.provider=singleFile -Dkeycloak.migration.file=/tmp/keycloak-hb.json -
Djboss.socket.binding.port-offset=999 -Dkeycloak.migration.realmName=healthbot
```

- d. Terminate the process using **ctrl-c** after you receive the following message:

```
Keycloak 9.0.0 (WildFly Core 10.0.3.Final) started in 17548ms - Started 590 of 885 services (601
services are lazy, passive or on-demand)
```

- e. Disconnect from the keycloak pod and copy the file from the Keycloak pod to your local system.

```
healthbot k cp <Keycloak pod>:/tmp/keycloak-hb.json /var/local/healthbot/keycloak-hb.json
```

2. Follow the instructions mentioned in ["Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X" on page 16](#) to upgrade to Paragon Insights Release 4.X.

NOTE: When you run the **sudo healthbot setup** command, you need not enter the installation configuration as the installer already has the required information.

3. Log in to the GUI.

The default password for the *admin* user is *Admin123!* and the *admin* user must reset the password on first login. The default password for users other than the *admin* user is *Change123!*.

4. If you are the *admin* user, you can import the user configuration to Paragon Insights Release 4.X using the following command:

```
/var/local/healthbot/healthbot migrate-user -f <keycloak-backup-filename> -u <administrative-
username> -p <password>
```

You are now upgraded to Paragon Insights Release 4.X and you have migrated the data.

The following are the changes after you upgrade and migrate your data to Paragon Insights Release 4.X:

- The existing users of Paragon Insights Release 3.X.X are added to the IAM server. However, the existing user's credentials are not migrated.
- The existing user roles of Paragon Insights Release 3.X.X are not migrated. The existing user is mapped to default roles that are available in Paragon Insights Release 4.X.

- The existing user groups of Paragon Insights Release 3.X.X are not migrated. The existing user groups are associated with a default role, that is, the *hadmin* user group will be migrated with the default role of *sp-admin* and any other groups will be migrated with role of *sp-operator*. After the upgrade is complete, the administrator can manually reconfigure the group permission.

NOTE: If you are using LDAP (Lightweight Directory Access), users and user groups are not added to the LDAP server.

Migration from a Single-Master Setup to a Multi-Master Setup

IN THIS SECTION

- [Before You Begin | 48](#)
- [Steps for Migration | 49](#)
- [Multi-NIC | 50](#)

Paragon Insights does not support an upgrade from a single-master setup to a multi-master setup. With Paragon Insights Release 4.0.0, you can only migrate from a single-master setup to a multi-master setup.

These topics provide information on migration.

Before You Begin

- For production deployments, it is recommended that you have dedicated master nodes for the Kubernetes control plane.
- All prerequisites (node-related) provided in the ["Paragon Insights Installation Requirements"](#) on [page 2](#) topic are met.

Steps for Migration

Follow these steps to migrate from a single-master setup to a multi-master setup

1. Backup Config data from the single-master setup.

The Config files must be backed up in a folder other than `/var/local/healthbot/` to avoid losing data during uninstallation.

For more information on backup, see *Paragon Insights Configuration – Backup and Restore*.

2. Backup Times Series Database (TSDB) data from the single-master setup.

```
root@linux $ export HB_EXTRA_MOUNT3=/root/backup_all
root@linux $ healthbot -v tsdb backup --all --path <tsdb_backup_path>
```

3. Uninstall Paragon Insights.

```
"yum remove <healthbot>"/apt-get remove <healthbot>
```

4. Reinstall Paragon Insights.

Run `healthbot setup`, install Kubernetes, and run `healthbot start`.

- For more information on installing Paragon Insights, see ["Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X" on page 16](#).
- When you run `healthbot setup`, you are prompted to specify hostnames or IP addresses of the master nodes. If you choose to have multiple master nodes, you must also specify the virtual IP address that is required for configuring high availability (HA) between the master nodes. However, if you use the silent installer, you can specify the virtual IP address in the `master_virtual_ip` field of the configuration file.
- To add more than one virtual IP during `healthbot setup`, see ["Multi-NIC" on page 50](#).

5. Restore TSDB backup.

```
root@linux $ healthbot k exec -it svc/tsdb-shim "mkdir /tmp/userBackups"

root@linux $ healthbot -v tsdb restore --all --path <tsdb_backup_path>
```

6. Restore configuration and helper files.

For more information, see *Paragon Insights Configuration – Backup and Restore*.

Multi-NIC

Paragon Insights Release 4.0.0 supports multi-NIC (network interface card).

In a scenario where telemetry information of devices that you are monitoring is generated from one network, and you access the web UI from another network, Paragon Insights allows you to enter multiple virtual IP addresses when you run the `healthbot setup`. This ensures segregation of traffic between the networks.

For this feature to work, you will need multiple virtual IP addresses for every subnet. Virtual IP addresses for these subnets can be provided when you run `healthbot setup`.

For example, when you run `healthbot setup`, you are prompted to enter a virtual IP address. In this case, you must enter more than one virtual IP address. After you run `healthbot start`, all services will be available on all virtual IP addresses.

Migration of Paragon Insights Release 3.2.0 (with Docker Compose) to Paragon Insights Release 4.0.0

You can migrate Paragon Insights Release 3.2.0 (Docker Compose installation) to Paragon Insights Release 4.0.0.

Since Paragon Insights Release 4.0.0 does not support a Docker Compose based installation, you can only migrate (and not upgrade) from a Release 3.2.0 Docker Compose installation to Release 4.0.0.

NOTE: Config/Data/User/Helper files must be migrated from Paragon Insights Release 3.2.0 Docker Compose installation to Paragon Insights Release 4.0.0.

For more information on upgrade that is not Docker Compose related, see ["Upgrading Paragon Insights Release 3.X with Kubernetes-based Installation to Release 4.X" on page 46](#).

Follow these steps to upgrade to Paragon Insights Release 4.0.0:

1. Backup Config data.

The Config files must be backed up in a folder other than `/var/local/healthbot/` to avoid losing data during uninstallation.

- a. Backup configuration and helper files.

For more information, see *Paragon Insights Configuration – Backup and Restore*.

b. Backup Times Series Database (TSDB) data.

```
root@linux $ export HB_EXTRA_MOUNT3=/root/backup_all
root@linux $ healthbot -v tsdb backup --all --path <tsdb_backup_path>
```

For more information, see *Paragon Insights Configuration – Backup and Restore*.

c. Backup Keycloak users.

You backup Keycloak users when you are migrating existing users to the new IAM service in Paragon Insights Release 4.0.0. However, you can skip this step if user migration is not required.

To backup Keycloak users:

- i. Backup the keycloak-hb.json file.
- ii. Run the following command to identify the keycloak container ID in the namespace.

```
root@linux $ docker ps | grep keycloak
```

For steps "1.c.iii" on page 51 through "1.c.v" on page 52, replace *<keycloak docker-container id>* with the Keycloak container ID that you identified.

- iii. Connect to the keycloak container.

```
root@linux $ docker exec -it <keycloak docker-container id> bash
```

- iv. Run the following command from the keycloak container to generate the backup file.

```
root@linux $ docker exec -it <keycloak docker-container id> bash

root@67dfd489d4 # /opt/jboss/keycloak/bin/standalone.sh -
Dkeycloak.migration.action=export -Dkeycloak.migration.provider=singleFile -
Dkeycloak.migration.file=/tmp/keycloak-hb.json -Djboss.socket.binding.port-
offset=999 -Dkeycloak.migration.realmName=healthbot
    Terminate the process using ctrl-c after you receive the following
message:
    Keycloak 9.0.0 (WildFly Core 10.0.3.Final) started in 17548ms -
    Started 590 of 885 services (601 services are lazy, passive or on-demand)
```


- v. Disconnect from the keycloak container and copy the file from the keycloak container to your local system.

```
root@linux $ docker cp <Keycloak docker-container id>:/tmp/keycloak-hb.json /root/
backup/keycloak-hb.json
```

The backup process is not complete. The backed up data is stored in the host machine.

2. Uninstall Paragon Insights Release 3.2.0.

```
"yum remove <package-name>"/apt-get remove <package-name>
```

3. Install Paragon Insights Release 4.0.0.

Run `healthbot setup`, install Kubernetes, and run `healthbot start`.

For more information, see ["Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X" on page 16](#).

4. Restore TSDB backup.

```
root@linux $ healthbot k exec -it svc/tsdb-shim "mkdir /tmp/userBackups"

root@linux $ healthbot -v tsdb restore --all --path <tsdb_backup_path>
```

5. Migrate keycloak users to IAM users.

- a. Log in to the Paragon Insights graphical user interface (GUI) using the credentials: Username: *admin*, Password: *Admin123!*.

When you enter these one-time credentials, Paragon Insights prompts you to change the password. The default password for users other than the admin user is *Change123!*.

- b. If you are the admin user, you can import user configuration to Paragon Insights Release 4.0.0 by running the following command:

```
/var/local/healthbot/healthbot migrate-user -f <keycloak-backup-filename> -u
<administrative-username> -p <password>
```

<password>

6. Restore configuration and helper files.

For more information, see *Paragon Insights Configuration – Backup and Restore*.

Migration from Release 2.X to Release 3.X or Release 4.X

IN THIS SECTION

- [Overview | 53](#)
- [Case 1: Release 2.X Single-node \(Docker Compose\) to Release 3.X Single-node \(Docker Compose\) Migration | 54](#)
- [Case 2: Automated Migration from 2.X to 3.X and 4.X \(Kubernetes\) | 55](#)
- [Case 3: Manual Migration from 2.X to 3.X and 4.X \(Kubernetes\) | 56](#)

Overview

You can migrate to Release 3.X or 4.X from Release 2.X.

Paragon Insights (formerly HealthBot) Release 2.X was based entirely on Docker and Docker Compose for the management of the back end microservices for both single-node and multinode installations. Starting with Paragon Insights Release 3.0.0, Kubernetes is used to manage the Docker components in multinode installations. This change introduces some incompatibilities between the existing 2.X data and the 3.X and 4.X formats, specifically:

- The underscore, “_”, character is no longer supported for Device Group or Network Group names.
- Device Group and Network Group names are no longer case-sensitive. So, in 3.X and 4.X, the names DeviceGroup and deviceGroup are considered the same.
- Differences in the time-series databases (TSDB) between the versions can cause problems.

We have implemented procedures for detecting these incompatibilities and transforming the data during specific data migration processes.

The following three use cases describe procedures for migrating the 2.X data into a 3.X installation or a 4.X installation.



WARNING: The procedures and functions outlined below are provided only on a best-effort basis. There is no guarantee that the data will migrate without some errors. In the event that the migration function encounters incompatibilities that it cannot resolve, an alert message regarding the need for manual intervention is displayed.

Ensure that you are familiar with ["Paragon Insights Installation Requirements" on page 2](#) and ["Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X" on page 16](#). These topics provide greater detail about some of the steps outlined in the procedures below.

Case 1: Release 2.X Single-node (Docker Compose) to Release 3.X Single-node (Docker Compose) Migration

Before you begin, you must have:

- A working 2.X installation
- Downloaded either the .deb file for Ubuntu or the .rpm file for CentOS to a temporary location on your 2.X server

See ["Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X" on page 16](#) for details.

The commands that you need to run to perform the migration on Ubuntu are:

NOTE: Single-node Paragon Insights installations are not recommended for production systems. If you perform this type of upgrade, you cannot upgrade later to a multinode installation with Docker Compose.

```
$ sudo apt-get install -y /<path-to-deb-file>/healthbot-<version>.deb
$ sudo healthbot setup
```

Answer “n” to the question about using Kubernetes.

The setup procedure contains the functions needed to detect the incompatibilities and transform the data. If incompatibilities are discovered, you are asked to confirm the data transformation attempt.

```
$ healthbot start
```

Check to see if your data was properly migrated.

The commands that you need to run to perform the migration on CentOS are:

```
sudo yum install -y /<path-to-rpm-file>/healthbot-<version>.rpm
sudo healthbot setup
```

Answer “n” to the question about using Kubernetes.

The setup procedure contains the functions needed to detect the incompatibilities and transform the data. If incompatibilities are discovered, you are asked to confirm the data transformation attempt.

```
$ healthbot start
```

Check to see if your data was properly migrated.

Case 2: Automated Migration from 2.X to 3.X and 4.X (Kubernetes)

Before you begin, you must have:

- A working 2.X installation
- Run `$ healthbot stop` on the 2.X server
- Downloaded either the `.deb` file for Ubuntu or the `.rpm` file for CentOS to your 3.X or 4.X server

See ["Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X" on page 16](#) for details.

- Run the interactive installers on your 3.X or 4.X server.

During the `$ sudo healthbot setup` phase of the install, answer “Y” to the question about installing with Kubernetes.

See ["Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X" on page 16](#) for details.

When `healthbot setup` reports success, do not run `healthbot start`. Instead, run the following commands:

- Run `$ healthbot migrate --host <ip address or hostname of 2.X installation>`

For example: `$ healthbot migrate --host 10.xx.xx.124`

The migration function immediately prompts you for an ssh username and password to use to connect to the 2.X server. The credentials you provide must have root privileges. Once connected to the 2.X server, it collects the following files needed for migration and places them in the appropriate directories on the new server:

- Configuration files
- TSDB files
- Helper files

Once collected, the migration function scans the configuration and TSDB files for incompatibilities. If any are found, it tries to transform them to make them compatible with 3.X or 4.X.

- Run `$ healthbot start`
- Check to see if your data migrated properly.

Case 3: Manual Migration from 2.X to 3.X and 4.X (Kubernetes)

Before you begin, you must have:

- A working 2.X installation
- Run `$ healthbot stop` on the 2.X server
- Downloaded either the `.deb` file for Ubuntu or the `.rpm` file for CentOS to your 3.X or 4.X server

See ["Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X" on page 16](#) for details.

- Run the interactive installers on your 3.X or 4.X server.

During the `$ sudo healthbot setup` phase of the install, answer "Y" to the question about installing with Kubernetes.

See ["Using the Interactive Installers to Install or Upgrade to Paragon Insights Release 4.X" on page 16](#) for details.

When `healthbot setup` reports success, do not run `healthbot start`. Instead, run the following commands on the 2.X server:

- `$ tar czf /tmp/influx.tgz /var/local/healthbot/etc/data/influxdb`
- `$ tar czf /tmp/helper.tgz /var/local/healthbot/input`

- `$ scp /tmp/helper.tgz /tmp/influx.tgz /var/local/healthbot/config/healthbot.json <username>@<ip or hostname of 3.x or 4.x server>:</writable/directory/on/3.x or 4.x/server>`

For example: `$ scp /tmp/helper.tgz /tmp/influx.tgz /var/local/healthbot/config/healthbot.json`

`user@10.100.101.12:/var/tmp/` where user is a valid username on the host with permission to write in the `/var/tmp/` directory.

- On the 3.X or 4.X server, run: `$ healthbot migrate -c </path/to/healthbot.json> -t </path/to/influx.tgz> -hf </path/to/helper.tgz>`

For example: `$ healthbot migrate -c /var/tmp/healthbot.json -t /var/tmp/influx.tgz -hf /var/tmp/helper.tgz`

The migration function scans the configuration and TSDB files for incompatibilities. If any are found, it requests confirmation from the user before it tries to transform them to make them compatible with 3.X or 4.X.

- Run `$ healthbot start`
- Check to see if your data migrated properly.

Installing HealthBot Release 3.2.0

IN THIS SECTION

- [Installing Paragon Insights On Ubuntu | 59](#)
- [Installing Paragon Insights On CentOS | 62](#)

The Paragon Insights (formerly HealthBot) software package is available for download as a Debian (.deb) file for installation on Ubuntu, or a Red Hat Package Manager (.rpm) file for installation on CentOS and Red Hat Enterprise Linux (RHEL).

Before You Begin

You will need the following details for the installation:

- Deployment type—Single node or multinode installation. Multinode installations are useful for load distribution and scaling.
- SSL certificate and private key—Supply your own key, or have Paragon Insights create one for you.
- Host IP address—The server IP address, for SSH and Web UI access.

- If you are using Paragon Insights to install a new Kubernetes cluster, ensure that the hostnames of the nodes resolve to the IP addresses of the nodes used in the Kubernetes cluster. You can use DNS to map the hostname to the IP address or add the entries to the local `/etc/hosts` file.
- If you are installing Paragon Insights into an existing Kubernetes cluster, you need information about the existing Kubernetes cluster:
 - Docker registry name
 - The path to a kubeconfig file configured with the existing cluster details
 - A user account with administrator privileges for the Kubernetes cluster

NOTE: The following points regarding installation on an existing Kubernetes cluster must be considered:

- Paragon Insights has only been qualified with the Calico container network interface (CNI). It should work with other CNI plugins, but has not been tested.
- If your Kubernetes cluster uses other CNI plugins, such as Flannel, you might notice long playbook deployment times on some Kubernetes versions. One potential workaround, is to disable transmit (tx) and receive (rx) checksum offloading on VXLAN interfaces. An example of this, using Flannel: `ethtool --offload flannel.1 tx off rx off`.

For discussions regarding the latency issue, see:

- [Kubernetes Issue](#)
- [Flannel Issue](#)
- [LKML.ORG](#)
- If there are a large number of device groups in Paragon Insights, there is the possibility that some of the Kubernetes pods might not get scheduled if the nodes are saturated by the maximum pods per node limit. By default, this limit is 110 pods per node in most Kubernetes distributions.

As a workaround, you can increase the maximum pods per node setting. Refer to the documentation from your Kubernetes distribution for details on how to modify this setting.
- If your CNI plugin is configured with a static IP CIDR block for each node, make sure the number of IP addresses in the block allocated to the node is at least double the size of the maximum pods per node setting.

Additional requirements:

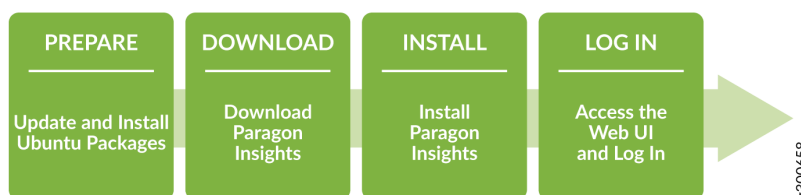
- You must have a Juniper.net user account with permissions to download the Paragon Insights software package.
- The Paragon Insights server must have access to the Internet during the software extraction process of installation (.deb for Ubuntu or .rpm for CentOS/Red Hat).

Paragon Insights creates a `/var/local/healthbot/` directory as part of the installation process. This directory contains all the Paragon Insights-related files.

Installing Paragon Insights On Ubuntu

The general workflow for installing Paragon Insights on Ubuntu is shown in [Figure 3 on page 59](#).

Figure 3: Installation Workflow - Paragon Insights on Ubuntu



NOTE: For multinode installations, perform the following tasks only on the primary node. If you are logged in as root or using root privileges, remove the `sudo` portion of each command below.

Part 1 - PREPARE

1. Ensure that you have at least 60GB of free disk space. We recommend 100GB of free disk space.
2. Set the environment variable, `HB_EXTRA_MOUNT1`, to allow Paragon Insights to access directories outside of the `/var/local/healthbot` directory.

To set the environment variable:

```
root@hb_server:/ export HB_EXTRA_MOUNT1=/root/.kube/config
```

3. Set the environment variables— `HB_IAM_SKIP_MAIL_VERIFICATION` and `HB_IAM_DISABLE_SMTP_SETTINGS`— as false to enable e-mail verification of users.

To set the environment variable:

```
root@hb_server:/ export HB_IAM_SKIP_MAIL_VERIFICATION=false
root@hb_server:/ export HB_IAM_DISABLE_SMTP_SETTINGS=false
```

4. Ensure that the Ubuntu package lists on your host system are latest.

```
$ sudo apt-get update
```

5. (Optional) Install the **wget** package. This tool will be used later to download the Paragon Insights software package. On some installations, **wget** is installed by default.

```
$ sudo apt-get install -y wget
```

6. Install Docker CE.

The following commands install the latest stable version on x86 machines:

```
$ sudo apt-get install -y apt-transport-https ca-certificates curl gnupg-agent software-properties-common
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
$ sudo apt-get update
$ sudo apt-get install -y docker-ce docker-ce-cli containerd.io
```

For more information, see [Get Docker CE for Ubuntu](#).

7. (Optional) Add your user account to the Docker group. This is required for a later installation step. If installing and running with root privileges, this step is not required.

```
$ sudo usermod -aG docker $USER
```

Log out of the current session, and log back in.

Part 2 - DOWNLOAD

1. Download the Paragon Insights package from the Juniper Networks [software download page](#) to a temporary directory (like /var/tmp/) on the server. Note that downloading software requires a Juniper.net account.

```
wget -O /<temp-directory>/healthbot-<version>.deb "<URL-from-the-software-download-page>"
```

NOTE: You can also download the Paragon Insights package locally and push it to the server, if preferred.

Part 3 - INSTALL

1. Install the .deb package using the following format:

```
$ sudo apt-get install -y /<path-to-deb-file>/Paragon Insights-<version>.deb
```

Paragon Insights checks that prerequisite software is installed on your host device during this step. If any required software is not found, Paragon Insights will prompt you before installing those missing software packages.

NOTE: If you see the following error when running the `sudo apt-get` command, you can ignore it:

"Can't drop privileges for downloading as file '/home/user/healthbot-3.0.0-1.deb' couldn't be accessed by user '_apt'. - pkgAcquire::Run (13: Permission denied)"

For more information, see [Bug 1543280](#).

2. Paragon Insights supports offline installation for Kubernetes-based installations.

Download the **Healthbot Package-Offline Support** file from the [Paragon Insights Downloads](#) page. Once downloaded, change the name of the file to **healthbot-offline.tgz** and place it in the **/var/local/healthbot** directory.

3. Enter `sudo healthbot setup` to configure your installation. You can also use `sudo healthbot -v setup` to display more detailed information.

While running the `sudo healthbot setup` command, you are prompted to answer several questions as part of the setup process.

To proceed with Docker Compose installation, enter `N` when you see this prompt: Do you want to install on kubernetes? [y/N]. To proceed with kubernetes-based installation, enter `y`.

For more information, see ["Paragon Insights Interactive Installation Prompts" on page 30](#).

4. Enter `healthbot start` to start Paragon Insights services including the Web UI.
5. Enter `healthbot status` to verify that the Paragon Insights services are up and running.

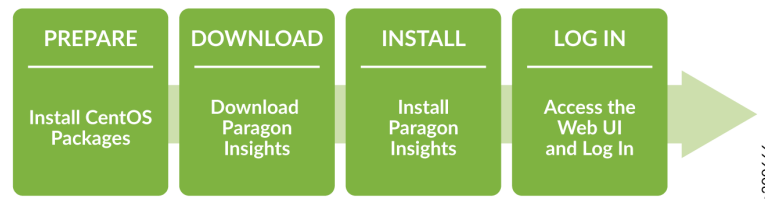
Part 4 - LOG IN

1. Open a browser and enter the <https://server-IP:8080> URL to access the Paragon Insights UI, where *server-IP* is the virtual IP address you entered, to connect to Paragon Insights services, during installation. Log in to the Paragon Insights Web UI using the credentials: **Username:** *admin*, **Password:** *Admin123!*. These are one-time credentials. When you enter them, Paragon Insights prompts you to change the password and gives instructions about the recommended length, case changes, and so on.

Installing Paragon Insights On CentOS

The general workflow for installing Paragon Insights on CentOS is shown in [Figure 4 on page 62](#).

Figure 4: Installation Workflow - Paragon Insights on CentOS



NOTE: For multinode installations, perform the following tasks only on the primary node. If you are logged in as root or using root privileges, remove the `sudo` portion of each command below.

Part 1 - PREPARE

1. This procedure assumes you are installing Paragon Insights using a non-root user. If this user does not already have `sudo` privileges, set them now.

```
$ su
Password: <root-pwd>
```

```
# sudo usermod -aG wheel <non-root-username>
<stay logged in as root>
```

2. While logged in as root, set `secure_path` in the `/etc/sudoers` file to include `/usr/local/bin/` so that `sudo` will be able to locate the Paragon Insights script during the healthbot setup step, later in this procedure.

```
# visudo
<scroll down to Defaults secure_path = /sbin:/bin:/usr/sbin:/usr/bin>
<type i to enter edit mode>
<scroll to end of line and add :/usr/local/bin >
<type Esc, then :wq, and press Enter>
# exit
exit
$ <Log out of the current session, and log back in>
```

3. Install the **epel-release** repository. This is required for installing Python3 and other packages.

```
$ sudo yum install -y epel-release
```

4. Install the `yum-utils` package used for handling repositories and extending package management.

```
$ sudo yum install -y yum-utils
```

5. Install the `wget` package. This tool will be used later to download the Paragon Insights software package.

```
$ sudo yum install -y wget
```

6. Verify the SELinux mode, and if set to enforcing change it to permissive. This is required to allow Docker commands to execute later in this procedure.

```
$ getenforce
Enforcing
$ sudo setenforce 0
$ getenforce
Permissive
```

7. Install Docker CE.

```
$ sudo yum install -y yum-utils \ device-mapper-persistent-data \ lvm2
$ sudo yum-config-manager \ --add-repo \ https://download.docker.com/linux/centos/docker-
ce.repo
$ sudo yum install docker-ce docker-ce-cli containerd.io
$ sudo systemctl start docker
$ sudo systemctl enable docker
```

For more information, see [Get Docker CE for CentOS](#).

Part 2 - DOWNLOAD

1. Download the Paragon Insights package from the Juniper Networks [software download page](#) to a temporary directory (like /var/tmp) on the server. Note that downloading software requires a Juniper.net account.

```
wget -O /<temp-directory>/healthbot-<version>.noarch.rpm "<URL-from-the-software-download-
page>"
```

NOTE: You can also download the Paragon Insights package locally and push it to the server, if preferred.

Part 3 - INSTALL

1. Install the .rpm package using the following format:

```
$ sudo yum install -y /<path-to-rpm-file>/healthbot-<version>.noarch.rpm
```

Paragon Insights checks that prerequisite software is installed on your host device during this step. If any required software is not found, Paragon Insights will prompt you before installing those missing software packages. Paragon Insights uses an Internet connection to download any missing packages.

2. Paragon Insights supports offline installation for Kubernetes-based installations.

Download the **Healthbot Package-Offline Support** file from the [Paragon Insights Downloads](#) page. Once downloaded, change the name of the file to **healthbot-offline.tgz** and place it in the **/var/local/healthbot** directory.

3. Enter `sudo healthbot setup` to configure your installation. You can also use `sudo healthbot -v setup` to display more detailed information.

While running the `sudo healthbot setup` command, you are prompted to answer several questions as part of the setup process.

To proceed with Docker Compose installation, enter `N` when you see this prompt: `Do you want to install on kubernetes? [y/N]`. To proceed with kubernetes-based installation, enter `y`.

For more information, see "[Paragon Insights Interactive Installation Prompts](#)" on page 30.

Paragon Insights installation is now complete.

4. Enter `healthbot start` to start Paragon Insights services including the Web UI.
5. Enter `healthbot status` to verify that the Paragon Insights services are up and running.

For example:

6. Reset the SELinux mode to enforcing, if applicable.

Part 4 - LOG IN

1. Open a browser and enter the <https://server-IP:8080> URL to access the Paragon Insights UI, where *server-IP* is the virtual IP address you entered, to connect to Paragon Insights services, during installation. Log in to the Paragon Insights Web UI using the credentials: **Username:** *admin*, **Password:** *Admin123!*. These are one-time credentials. When you enter them, Paragon Insights prompts you to change the password and gives instructions about the recommended length, case changes, and so on.

Uninstalling or Reconfiguring Paragon Insights

IN THIS SECTION

- [Uninstalling Paragon Insights | 66](#)
- [Reconfiguring Paragon Insights | 67](#)

This section describes the following tasks:

Uninstalling Paragon Insights

Paragon Insights on Ubuntu

To uninstall Paragon Insights (formerly HealthBot) on Ubuntu, enter the following command:

```
$ sudo apt-get [remove|purge] healthbot
```

The `remove` and `purge` options both perform the same action. You can use either to uninstall Paragon Insights. For example:

```
$ sudo apt-get purge healthbot
```

Paragon Insights on CentOS

NOTE: When uninstalling Paragon Insights on CentOS, you might see warnings indicating that some of the package files are not found. You can safely ignore these warnings.

To uninstall Paragon Insights on CentOS:

1. Verify the SELinux mode, and if set to enforcing change it to permissive. This is required to allow the uninstall procedure to run.

```
$ getenforce
Enforcing
$ sudo setenforce 0
$ getenforce
Permissive
```

2. Uninstall the Paragon Insights package and its components using the following format:

```
$ sudo yum remove <healthbot-package-name>
```

For example:

```
$ yum list installed | grep healthbot
healthbot.noarch                2.0-2.1                installed
$ sudo yum remove healthbot.noarch
```

```

sudo yum remove healthbot.noarch
Loaded plugins: fastestmirror
Resolving Dependencies
--> Running transaction check
--> Package healthbot.noarch 0:2.0-2.1 will be erased
...
<answer 'y' to prompts as needed>
...
Running transaction
...
Removed healthbot
  Erasing      : healthbot-2.0-2.1.noarch                1/1
  Verifying    : healthbot-2.0-2.1.noarch                1/1

Removed:
  healthbot.noarch 0:2.0-2.1

Complete!

```

NOTE: This procedure does not uninstall all of the software dependencies that were installed for Paragon Insights during installation.

3. Reset the SELinux mode to enforcing, if applicable.

```

$ getenforce
Permissive
$ sudo setenforce 1
$ getenforce
Enforcing

```

Reconfiguring Paragon Insights

Use the `healthbot reconfigure` command to:

- Change from a single-node installation to a multinode installation, or vice versa
- Change the Paragon Insights server's IP address

- Change SSL certificates

For example:

```
$ sudo healthbot reconfigure
```

```
i Please have a look at below messages and acknowledge them to proceed:
```

1. Paragon Insights free model/freemium license tier is not available in this version. A valid license (trial or commercial) is needed to use Paragon Insights features.
2. The format of license key has been changed due to business reasons in this version. The system may have contained the older format of license keys, which are not valid now. You can log into the Juniper Agile Licensing Portal and choose to revoke the existing license and then re-activate the license to a new software version (that will provide you with a new format license key). This newly activated license can be added to the system post upgrade.

```
Do you want to proceed? [Y/n] Y
```

```
Do you want to reconfigure SSL certificates? [y/N]y
```

```
Do you want system to create and use a self-signed SSL certificate? [Y/n] Y
```

```
[executing ..] openssl req -nodes -x509 -newkey rsa:4096 -keyout /var/local/healthbot/certs/
key.pem -out /var/local/healthbot/certs/cert.pem -days 36525 -subj "/C=US/ST=Ca/L=Svl/O=Juniper
Networks/OU=Org/CN=localhost" -extensions SAN -reqexts SAN -config /tmp/ssl.conf
```

```
[executing ..] rm -f /tmp/ssl.conf
```

```
Creating /var/local/healthbot/healthbot.conf config file..
```

```
i Checking space for docker images
```

```
Loading docker images. This may take some time..
```

```
100%|████████████████████████████████████████████████████████████████████████████████| 50/50 [01:43<00:00,
2.08s/it]
```

```
Docker images successfully loaded
```

```
Done making directories
```

```
i User has quit previous setup to reset iptables. Ignoring API server unreachability.
```

```
Do you want to reconfigure kubernetes cluster? [y/N] y
```

```
Installing new Kubernetes cluster
```

```
Existing cluster details found
```

```
Master nodes: ['10.xxx.165.xxx']
```

```
Worker nodes: []
```

```
Master scheduling: True
```

```
Ingress virtual IP: ['10.102.xxx.xxx']
```


You can also reconfigure Paragon Insights using a configuration file. For example:

```
sudo healthbot reconfigure -c /path/configuration-file.conf
```

In this example, Paragon Insights is reconfigured to whatever settings are defined in the configuration file. For more information on the configuration file, see ["Using the Silent Installer" on page 33](#).

Paragon Insights Command-Line Options

Use `-h` (help) to view all available command-line options for the `healthbot` command.

```
root@choc-ubuntu1:~# healthbot
$ healthbot -h
usage: healthbot.py [-h] [-v] [-nc]
                        {ansible,auditlogs,setup,reconfigure,status,remove,start,stop,uninstall,gen-
compose,list-plugins,load-plugin,remove-plugin,backup-grafana,restore-grafana,modify-workflow-
engine,modify-uda-engine,modify-udf-engine,restart,cli,copy,logs,mgd,tsdb,add-node,remove-
node,get-nodes,runtime,cluster-health,patch,version,publish-license-files,connection-state-
reset,k,debug,sync-channel-to-master,sync-channel,get-k8s-dashboard-token,postgres,reload-udp-
ingest-common-config,cluster-exec,reset-password,migrate-user,create-health-database}
                        ...

positional arguments:
  {ansible,auditlogs,setup,reconfigure,status,remove,start,stop,uninstall,gen-compose,list-
plugins,load-plugin,remove-plugin,backup-grafana,restore-grafana,modify-workflow-engine,modify-
uda-engine,modify-udf-engine,restart,cli,copy,logs,mgd,tsdb,add-node,remove-node,get-
nodes,runtime,cluster-health,patch,version,publish-license-files,connection-state-
reset,k,debug,sync-channel-to-master,sync-channel,get-k8s-dashboard-token,postgres,reload-udp-
ingest-common-config,cluster-exec,reset-password,migrate-user,create-health-database}
                        sub-command help

setup

Setup healthbot
reconfigure
    Reconfigure healthbot
status
    Show healthbot status
remove
```

```

    Remove/Delete services for a group
start
    Start healthbot. This command will start common services and services for configured
device and network groups
stop
    Stop all running healthbot services
uninstall
    Uninstall healthbot
gen-compose
    Generate compose files of services for the given group
list-plugins
    List plugins available in Healthbot
load-plugin
    Load a plugin
remove-plugin

Remove a plugin
backup-grafana
    Backup Grafana data
restore-grafana
    Restore Grafana data
modify-workflow-engine.  Modify WORKFLOW engine
modify-uda-engine
    Modify UDA engine
modify-udf-engine
    Modify UDF engine
restart
    Restart the service(s) for a group
cli
    Gain cli access to a service
logs
    Access service logs
mgd
    Spin up MGD container for writing rules
tsdb
    update TSDB settings
add-node
    add new worker nodes to kubernetes cluster. eg: healthbot add-node node1 node2
remove-node
    remove existing worker nodes from kubernetes cluster.eg: healthbot remove-node node1
get-nodes
    Get the list of nodes in current kubernetes cluster
runtime

```

```

    print runtime details
cluster-health
    Check cluster health
patch
    patch healthbot service
version
    Print healthbot version information
publish-license-files    Publish license files
connection-state-reset  Command to reset connection states manually
k
    Run kubectl commands
debug
    Debug healthbot
sync-channel-to-master  Force sync config manager's channel to master node (applicable for
Kubernetes environment)
sync-channel
    Force sync given channel to all nodes (applicable for Kubernetes environment)
get-k8s-dashboard-token  Prints kubernetes dashboard token
postgres
    Postgres commands
reload-udp-ingest-common-config
    This command helps a service reload its config. Its
    support is very limited now
cluster-exec
    Execute commands on nodes
reset-password
    Reset password for a user
migrate-user
    Migrate user configuration from HealthBot 3.2.x
create-health-database  Check cluster health

optional arguments:
-h, --help            show this help message and exit
-v, --verbose         Increase output verbosity
-nc, --no-color       No color

```

You can also use `-h` with an option to view more information. For example:

```
$ healthbot status -h
usage: healthbot status [-h]
                        [--network-group NETWORK_GROUP | --device-group DEVICE_GROUP]
                        [-j]

optional arguments:
  -h, --help            show this help message and exit
  --network-group NETWORK_GROUP
                        Show network group status
  --device-group DEVICE_GROUP
                        Show device group status
  -j, --json            Display data in JSON format.
```

RELATED DOCUMENTATION

[Starting and Stopping Paragon Insights | 42](#)

[Uninstalling or Reconfiguring Paragon Insights | 65](#)

2

CHAPTER

Linux Kernel Upgrade Procedures

Ubuntu Kernel Upgrade | 75

CentOS Kernel Upgrade | 81

Ubuntu Kernel Upgrade

This addendum to the Paragon Insights (formerly HealthBot) Installation Guide is for those who need to upgrade the kernel version of their Ubuntu 16.04 server to 4.4.19 or later to meet the requirements for Paragon Insights installation. Kernel upgrade is required for both single and multinode installation. This document assumes that you are familiar with *apt*, the software packaging system on Ubuntu.

For those of you who would prefer to watch the commands get executed:



Video: [Ubuntu Kernel Upgrade](#)

The kernel upgrade process involves the following steps:

- Confirm current kernel version
- Update the *apt* repositories on the server.
- Upgrade existing software packages to the latest versions.
- Find and install latest kernel
- Reboot to load new kernel
- Verify that the system is running on the new kernel

To perform a kernel upgrade, you must be logged in to your server as the root user or be able to obtain root privileges using the `sudo -s` command. The commands in the rest of the procedure assume that you are logged in as root or have issued the `sudo -s` command.

To confirm the existing kernel version *apt* repositories:

-

```
root@server# uname -msr
Linux 4.4.0-178-generic x86_64
```

Since the existing kernel is below the required version, we must upgrade the kernel on this server.

Begin the kernel upgrade procedure by updating the *apt* repositories.

To update *apt* repositories:

- ```
root@server# apt update
root@pete:/home/probbins# apt update
Get:1 http://repo.juniper.net/Ubuntu/stable/JNPR/xenial jnpr InRelease [3,166 B]
Hit:2 http://us.archive.ubuntu.com/ubuntu xenial InRelease
```



```

Get:3 http://security.ubuntu.com/ubuntu xenial-security InRelease [109 kB]
Get:5 http://us.archive.ubuntu.com/ubuntu xenial-updates InRelease [109 kB]
Hit:4 https://packages.cloud.google.com/apt kubernetes-xenial InRelease
Get:6 http://security.ubuntu.com/ubuntu xenial-security/main amd64 Packages [863 kB]
Get:7 http://us.archive.ubuntu.com/ubuntu xenial-backports InRelease [107 kB]
Hit:8 https://download.docker.com/linux/ubuntu xenial InRelease
Get:9 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 Packages [1,140 kB]
Get:10 http://security.ubuntu.com/ubuntu xenial-security/main i386 Packages [659 kB]
Get:11 http://us.archive.ubuntu.com/ubuntu xenial-updates/main i386 Packages [918 kB]
Get:12 http://security.ubuntu.com/ubuntu xenial-security/main Translation-en [324 kB]
Get:13 http://us.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 Packages [797 kB]
Get:14 http://security.ubuntu.com/ubuntu xenial-security/universe amd64 Packages [491 kB]
Get:15 http://us.archive.ubuntu.com/ubuntu xenial-updates/universe i386 Packages [721 kB]
Get:16 http://security.ubuntu.com/ubuntu xenial-security/universe i386 Packages [422 kB]
Get:17 http://security.ubuntu.com/ubuntu xenial-security/universe Translation-en [201 kB]
Get:18 http://security.ubuntu.com/ubuntu xenial-security/multiverse amd64 Packages [6,092 B]
Get:19 http://security.ubuntu.com/ubuntu xenial-security/multiverse i386 Packages [6,248 B]
Fetched 6,878 kB in 15s (437 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
48 packages can be upgraded. Run 'apt list --upgradable' to see them.
W: Target Packages (stable/binary-amd64/Packages) is configured multiple times in /etc/apt/
sources.list:55 and /etc/apt/sources.list.d/download_docker_com_linux_ubuntu.list:1
W: Target Packages (stable/binary-all/Packages) is configured multiple times in /etc/apt/
sources.list:55 and /etc/apt/sources.list.d/download_docker_com_linux_ubuntu.list:1
W: Target Translations (stable/i18n/Translation-en_US) is configured multiple times
in /etc/apt/sources.list:55 and /etc/apt/sources.list.d/
download_docker_com_linux_ubuntu.list:1
W: Target Translations (stable/i18n/Translation-en) is configured multiple times in /etc/apt/
sources.list:55 and /etc/apt/sources.list.d/download_docker_com_linux_ubuntu.list:1
N: Skipping acquire of configured file 'stable/binary-i386/Packages' as repository 'https://
download.docker.com/linux/ubuntu xenial InRelease' doesn't support architecture 'i386'
W: Target Packages (stable/binary-amd64/Packages) is configured multiple times in /etc/apt/
sources.list:55 and /etc/apt/sources.list.d/download_docker_com_linux_ubuntu.list:1
W: Target Packages (stable/binary-all/Packages) is configured multiple times in /etc/apt/
sources.list:55 and /etc/apt/sources.list.d/download_docker_com_linux_ubuntu.list:1
W: Target Translations (stable/i18n/Translation-en_US) is configured multiple times
in /etc/apt/sources.list:55 and /etc/apt/sources.list.d/
download_docker_com_linux_ubuntu.list:1
W: Target Translations (stable/i18n/Translation-en) is configured multiple times in /etc/apt/
sources.list:55 and /etc/apt/sources.list.d/download_docker_com_linux_ubuntu.list:1

```

If you examine the output above, you'll see that 48 packages are eligible for upgrade.

To upgrade existing software packages, including kernel upgrades:

- ```

root@server# apt upgrade -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
The following packages were automatically installed and are no longer required:
  linux-headers-4.4.0-174 linux-headers-4.4.0-174-generic linux-image-4.4.0-174-generic
  linux-modules-4.4.0-174-generic linux-modules-extra-4.4.0-174-generic
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  linux-headers-4.4.0-178 linux-headers-4.4.0-178-generic linux-image-4.4.0-178-generic
  linux-modules-4.4.0-178-generic linux-modules-extra-4.4.0-178-generic
The following packages will be upgraded:
  apport distro-info-data git git-man kubeadm kubectl kubelet libcups2 libgd3 libglib2.0-0
  libglib2.0-data libicu55
  libldap-2.4-2 libpam-systemd libprocps4 libpulse0 libpython2.7-minimal libpython2.7-stdlib
  libpython3.5
  libpython3.5-dev libpython3.5-minimal libpython3.5-stdlib libsystemd0 libudev1 linux-base
  linux-firmware
  linux-generic linux-headers-generic linux-image-generic linux-libc-dev procps python2.7
  python2.7-minimal
  python3-apport python3-distupgrade python3-problem-report python3.5 python3.5-dev python3.5-
  minimal sosreport
  systemd systemd-sysv ubuntu-release-upgrader-core udev vim vim-common vim-runtime vim-tiny
48 upgraded, 5 newly installed, 0 to remove and 0 not upgraded.
Need to get 228 MB of archives.
After this operation, 311 MB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 systemd-sysv amd64
229-4ubuntu21.28 [12.3 kB]
Get:2 http://security.ubuntu.com/ubuntu xenial-security/main amd64 libpulse0 amd64
1:8.0-0ubuntu3.12 [253 kB]
Get:3 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libpam-systemd amd64
229-4ubuntu21.28 [115 kB]
Get:7 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libsystemd0 amd64
229-4ubuntu21.28 [206 kB]
Get:4 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 kubelet amd64
1.18.2-00 [19.5 MB]
Get:8 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 systemd amd64
229-4ubuntu21.28 [3,639 kB]

```

```

Get:5 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 kubect1 amd64
1.18.2-00 [8,825 kB]
Get:6 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 kubeadm amd64
1.18.2-00 [8,162 kB]
Get:9 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 udev amd64
229-4ubuntu21.28 [993 kB]
Get:10 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libudev1 amd64
229-4ubuntu21.28 [54.8 kB]
Get:11 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libprocps4 amd64
2:3.3.10-4ubuntu2.5 [32.9 kB]
Get:12 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 procps amd64
2:3.3.10-4ubuntu2.5 [222 kB]
Get:13 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 python3.5-dev amd64
3.5.2-2ubuntu0~16.04.10 [413 kB]
Get:14 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libpython3.5-dev amd64
3.5.2-2ubuntu0~16.04.10 [37.3 MB]
Get:15 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libpython3.5 amd64
3.5.2-2ubuntu0~16.04.10 [1,360 kB]
Get:16 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 python3.5 amd64
3.5.2-2ubuntu0~16.04.10 [165 kB]
Get:17 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libpython3.5-stdlib
amd64 3.5.2-2ubuntu0~16.04.10 [2,135 kB]
Get:18 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 python3.5-minimal amd64
3.5.2-2ubuntu0~16.04.10 [1,597 kB]
Get:19 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libpython3.5-minimal
amd64 3.5.2-2ubuntu0~16.04.10 [525 kB]
Get:20 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libglib2.0-0 amd64
2.48.2-0ubuntu4.6 [1,120 kB]
Get:21 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 python2.7 amd64
2.7.12-1ubuntu0~16.04.11 [224 kB]
Get:22 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libpython2.7-stdlib
amd64 2.7.12-1ubuntu0~16.04.11 [1,884 kB]
Get:23 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 python2.7-minimal amd64
2.7.12-1ubuntu0~16.04.11 [1,261 kB]
Get:24 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libpython2.7-minimal
amd64 2.7.12-1ubuntu0~16.04.11 [338 kB]
Get:25 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 distro-info-data all
0.28ubuntu0.14 [4,674 B]
Get:26 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 linux-base all
4.5ubuntu1.1~16.04.1 [18.1 kB]
Get:27 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 vim amd64
2:7.4.1689-3ubuntu1.4 [1,036 kB]
Get:28 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 vim-tiny amd64

```

```

2:7.4.1689-3ubuntu1.4 [446 kB]
Get:29 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 vim-runtime all
2:7.4.1689-3ubuntu1.4 [5,169 kB]
Get:30 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 vim-common amd64
2:7.4.1689-3ubuntu1.4 [103 kB]
Get:31 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libglib2.0-data all
2.48.2-0ubuntu4.6 [131 kB]
Get:32 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libicu55 amd64
55.1-7ubuntu0.5 [7,650 kB]
Get:33 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libldap-2.4-2 amd64
2.4.42+dfsg-2ubuntu3.8 [159 kB]
Get:34 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 ubuntu-release-upgrader-
core all 1:16.04.30 [30.5 kB]
Get:35 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 python3-distupgrade all
1:16.04.30 [104 kB]
Get:36 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 python3-problem-report
all 2.20.1-0ubuntu2.23 [10.5 kB]
Get:37 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 python3-apport all
2.20.1-0ubuntu2.23 [80.1 kB]
Get:38 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 apport all
2.20.1-0ubuntu2.23 [121 kB]
Testing for an existing GRUB menu.lst file ... found: /boot/grub/menu.lst
Searching for splash image ... none found, skipping ...
done

```

```

user@server# apt install --install-recommends linux-generic-hwe-16.04
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  linux-headers-4.4.0-174 linux-headers-4.4.0-174-generic linux-image-4.4.0-174-generic
  linux-modules-4.4.0-174-generic linux-modules-extra-4.4.0-174-generic
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  linux-headers-4.15.0-99 linux-headers-4.15.0-99-generic linux-headers-generic-hwe-16.04
  linux-image-4.15.0-99-generic linux-image-generic-hwe-16.04 linux-modules-4.15.0-99-generic
  linux-modules-extra-4.15.0-99-generic thermald
Suggested packages:
  fdutils linux-hwe-tools
The following NEW packages will be installed:
  linux-generic-hwe-16.04 linux-headers-4.15.0-99 linux-headers-4.15.0-99-generic linux-

```

```

headers-generic-hwe-16.04
Found kernel: /vmlinuz-4.13.0-26-generic
Found kernel: /vmlinuz-4.4.19-040419-generic
Found kernel: /vmlinuz-4.4.0-178-generic
Found kernel: /vmlinuz-4.4.0-174-generic
Replacing config file /run/grub/menu.lst with new version
Updating /boot/grub/menu.lst ... done

/etc/kernel/postinst.d/zz-update-grub:
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-4.15.0-99-generic
Found initrd image: /boot/initrd.img-4.15.0-99-generic
Found initrd image: /boot/initrd.img-4.4.19-040419-generic
Found linux image: /boot/vmlinuz-4.4.0-178-generic
Found initrd image: /boot/initrd.img-4.4.0-178-generic
Found linux image: /boot/vmlinuz-4.4.0-174-generic
Found initrd image: /boot/initrd.img-4.4.0-174-generic
done
Processing triggers for dbus (1.10.6-1ubuntu3.5) ...
Processing triggers for systemd (229-4ubuntu21.28) ...
Processing triggers for ureadahead (0.100.0-19.1) ...

```

As you can see from the output above, multiple kernels were available on this server. The `apt install` command with the `-install-recommends` flag found the available kernels and prepared the *grub* menu so that you can choose which kernel to boot from at boot time.

The new kernel is now available but not running. To load the new kernel, you must reboot the server.

- `user@server# reboot`

After reboot, confirm the running kernel.

- `user@server# uname -msr`
Linux 4.15.0-99-generic x86_64

RELATED DOCUMENTATION

| [CentOS Kernel Upgrade](#) | 81

CentOS Kernel Upgrade

This addendum to the Paragon Insights (formerly HealthBot) Installation Guide is for those who need to upgrade the kernel version of their CentOS 7 server to 4.4.19 to meet the requirements for Paragon Insights installation. Kernel upgrade is required for both single and multinode installation. With Paragon Insights Release 4.0.0, for CentOS 8 server and RHEL 8, kernel upgrade is not required. This document assumes that you are familiar with *yum* and *rpm*, the software package management systems on CentOS and Red Hat, as well as the *Grub* boot manager.

For those who prefer to watch the commands as they are executed:



Video: [CentOS Kernel Upgrade](#)

The kernel upgrade process involves the following steps:

- Confirm the current kernel version
- Update existing software packages to the latest versions
- Install GPG key for *EIRepo* software repository
- Install the *EIRepo* software repository
- Confirm the latest kernel in *EIRepo* software repository
- Install the latest kernel
- Reboot to load new kernel
- Verify that the system is running on the new kernel
- Edit *Grub* configuration
- Install *Grub* configuration
- Reboot

To perform a kernel upgrade, you must be logged in to your server as the root user or be able to obtain root privileges using the `sudo su` command. The commands in the rest of the procedure assume that you are logged in as root or have issued the `sudo -s` command.

-

Confirm the Current Kernel Version

```
root@server# uname -msr
Linux 3.10.0-1062.18.1.el7.x86_64 x86_64
```

Since the existing kernel is below the required version, we must upgrade the kernel on this server.

Begin the kernel upgrade procedure by updating the *yum* repositories and installed software packages.

To update *yum* repositories and installed software packages:

- **Update Existing Software Packages to the Latest Versions**

```
root@server# yum -y update
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirror.sfo12.us.leaseweb.net
 * elrepo: linux-mirrors.fnal.gov
 * extras: mirror.fileplanet.com
 * updates: mirror.sfo12.us.leaseweb.net
Resolving Dependencies
--> Running transaction check
---> Package bind-export-libs.x86_64 32:9.11.4-16.P2.el7_8.2 will be updated
---> Package bind-export-libs.x86_64 32:9.11.4-16.P2.el7_8.3 will be an update
---> Package bind-libs-lite.x86_64 32:9.11.4-16.P2.el7_8.2 will be updated
---> Package bind-libs-lite.x86_64 32:9.11.4-16.P2.el7_8.3 will be an update
---> Package bind-license.noarch 32:9.11.4-16.P2.el7_8.2 will be updated
---> Package bind-license.noarch 32:9.11.4-16.P2.el7_8.3 will be an update
---> Package binutils.x86_64 0:2.27-43.base.el7 will be updated
---> Package binutils.x86_64 0:2.27-43.base.el7_8.1 will be an update
---> Package containerd.io.x86_64 0:1.2.13-3.1.el7 will be updated
---> Package containerd.io.x86_64 0:1.2.13-3.2.el7 will be an update
---> Package device-mapper.x86_64 7:1.02.164-7.el7_8.1 will be updated
---> Package device-mapper.x86_64 7:1.02.164-7.el7_8.2 will be an update
---> Package device-mapper-event.x86_64 7:1.02.164-7.el7_8.1 will be updated
---> Package device-mapper-event.x86_64 7:1.02.164-7.el7_8.2 will be an update
---> Package device-mapper-event-libs.x86_64 7:1.02.164-7.el7_8.1 will be updated
---> Package device-mapper-event-libs.x86_64 7:1.02.164-7.el7_8.2 will be an update
---> Package device-mapper-libs.x86_64 7:1.02.164-7.el7_8.1 will be updated
---> Package device-mapper-libs.x86_64 7:1.02.164-7.el7_8.2 will be an update
---> Package kernel.x86_64 0:3.10.0-1127.8.2.el7 will be installed
---> Package kernel-tools.x86_64 0:3.10.0-1127.el7 will be updated
```

```

---> Package kernel-tools.x86_64 0:3.10.0-1127.8.2.el7 will be an update
---> Package kernel-tools-libs.x86_64 0:3.10.0-1127.el7 will be updated
---> Package kernel-tools-libs.x86_64 0:3.10.0-1127.8.2.el7 will be an update
---> Package lvm2.x86_64 7:2.02.186-7.el7_8.1 will be updated
---> Package lvm2.x86_64 7:2.02.186-7.el7_8.2 will be an update
---> Package lvm2-libs.x86_64 7:2.02.186-7.el7_8.1 will be updated
---> Package lvm2-libs.x86_64 7:2.02.186-7.el7_8.2 will be an update
---> Package python-perf.x86_64 0:3.10.0-1127.el7 will be updated
---> Package python-perf.x86_64 0:3.10.0-1127.8.2.el7 will be an update
---> Package systemd.x86_64 0:219-73.el7_8.5 will be updated
---> Package systemd.x86_64 0:219-73.el7_8.6 will be an update
---> Package systemd-libs.x86_64 0:219-73.el7_8.5 will be updated
---> Package systemd-libs.x86_64 0:219-73.el7_8.6 will be an update
---> Package systemd-sysv.x86_64 0:219-73.el7_8.5 will be updated
---> Package systemd-sysv.x86_64 0:219-73.el7_8.6 will be an update
---> Package yum-plugin-fastestmirror.noarch 0:1.1.31-53.el7 will be updated
---> Package yum-plugin-fastestmirror.noarch 0:1.1.31-54.el7_8 will be an update
---> Package yum-utils.noarch 0:1.1.31-53.el7 will be updated
---> Package yum-utils.noarch 0:1.1.31-54.el7_8 will be an update
--> Finished Dependency Resolution

```

Dependencies Resolved

```

=====
=====
Package                Arch          Version
Repository             Size
=====
=====
Installing:
kernel                  x86_64        3.10.0-1127.8.2.el7
updates                50 M
Updating:
bind-export-libs        x86_64        32:9.11.4-16.P2.el7_8.3
updates                1.1 M
bind-libs-lite          x86_64        32:9.11.4-16.P2.el7_8.3
updates                1.1 M
bind-license            noarch        32:9.11.4-16.P2.el7_8.3
updates                89 k
binutils                x86_64        2.27-43.base.el7_8.1
updates                5.9 M
containerd.io           x86_64        1.2.13-3.2.el7
stable                 25 M
docker-ce-

```


device-mapper	x86_64	7:1.02.164-7.el7_8.2
updates 295 k		
device-mapper-event	x86_64	7:1.02.164-7.el7_8.2
updates 191 k		
device-mapper-event-libs	x86_64	7:1.02.164-7.el7_8.2
updates 190 k		
device-mapper-libs	x86_64	7:1.02.164-7.el7_8.2
updates 324 k		
kernel-tools	x86_64	3.10.0-1127.8.2.el7
updates 8.0 M		
kernel-tools-libs	x86_64	3.10.0-1127.8.2.el7
updates 8.0 M		
lvm2	x86_64	7:2.02.186-7.el7_8.2
updates 1.3 M		
lvm2-libs	x86_64	7:2.02.186-7.el7_8.2
updates 1.1 M		
python-perf	x86_64	3.10.0-1127.8.2.el7
updates 8.0 M		
systemd	x86_64	219-73.el7_8.6
updates 5.1 M		
systemd-libs	x86_64	219-73.el7_8.6
updates 416 k		
systemd-sysv	x86_64	219-73.el7_8.6
updates 94 k		
yum-plugin-fastestmirror	noarch	1.1.31-54.el7_8
updates 34 k		
yum-utils	noarch	1.1.31-54.el7_8
updates 122 k		

Transaction Summary

```
=====
=====
```

Install 1 Package

Upgrade 19 Packages

Total download size: 117 M

Downloading packages:

Delta RPMs disabled because /usr/bin/applydeltarpm not installed.

(1/20): bind-license-9.11.4-16.P2.el7_8.3.noarch.rpm | 89

kB 00:00:00

(2/20): bind-libs-lite-9.11.4-16.P2.el7_8.3.x86_64.rpm | 1.1

MB 00:00:00

(3/20): bind-export-libs-9.11.4-16.P2.el7_8.3.x86_64.rpm | 1.1

```

MB 00:00:01
(4/20): device-mapper-1.02.164-7.el7_8.2.x86_64.rpm | 295
kB 00:00:00
(5/20): device-mapper-event-1.02.164-7.el7_8.2.x86_64.rpm | 191
kB 00:00:00
(6/20): device-mapper-event-libs-1.02.164-7.el7_8.2.x86_64.rpm | 190
kB 00:00:00
(7/20): device-mapper-libs-1.02.164-7.el7_8.2.x86_64.rpm | 324
kB 00:00:00
(8/20): containerd.io-1.2.13-3.2.el7.x86_64.rpm | 25
MB 00:00:02
(9/20): binutils-2.27-43.base.el7_8.1.x86_64.rpm | 5.9
MB 00:00:08
(10/20): kernel-tools-3.10.0-1127.8.2.el7.x86_64.rpm | 8.0
MB 00:00:08
(11/20): kernel-tools-libs-3.10.0-1127.8.2.el7.x86_64.rpm | 8.0
MB 00:00:08
(12/20): lvm2-2.02.186-7.el7_8.2.x86_64.rpm | 1.3
MB 00:00:01
(13/20): systemd-libs-219-73.el7_8.6.x86_64.rpm | 416
kB 00:00:00
(14/20): systemd-sysv-219-73.el7_8.6.x86_64.rpm | 94
kB 00:00:00
(15/20): yum-plugin-fastestmirror-1.1.31-54.el7_8.noarch.rpm | 34
kB 00:00:00
(16/20): yum-utils-1.1.31-54.el7_8.noarch.rpm | 122
kB 00:00:00
(17/20): lvm2-libs-2.02.186-7.el7_8.2.x86_64.rpm | 1.1
MB 00:00:03
(18/20): systemd-219-73.el7_8.6.x86_64.rpm | 5.1
MB 00:00:07
(19/20): kernel-3.10.0-1127.8.2.el7.x86_64.rpm | 50
MB 00:00:36
(20/20): python-perf-3.10.0-1127.8.2.el7.x86_64.rpm | 8.0
MB 00:00:27

```

```

-----
-----
Total 2.6 MB/s | 117

```

```

MB 00:00:44

```

```

Running transaction check

```

```

Running transaction test

```

```

Transaction test succeeded

```

```

Running transaction

```

Updating	: systemd-libs-219-73.el7_8.6.x86_64	1/39
Updating	: systemd-219-73.el7_8.6.x86_64	2/39
Updating	: 7:device-mapper-libs-1.02.164-7.el7_8.2.x86_64	3/39
Updating	: 7:device-mapper-1.02.164-7.el7_8.2.x86_64	4/39
Updating	: 7:device-mapper-event-libs-1.02.164-7.el7_8.2.x86_64	5/39
Updating	: 7:device-mapper-event-1.02.164-7.el7_8.2.x86_64	6/39
Updating	: 7:lvm2-libs-2.02.186-7.el7_8.2.x86_64	7/39
Updating	: kernel-tools-libs-3.10.0-1127.8.2.el7.x86_64	8/39
Updating	: 32:bind-license-9.11.4-16.P2.el7_8.3.noarch	9/39
Updating	: 32:bind-libs-lite-9.11.4-16.P2.el7_8.3.x86_64	10/39
Updating	: kernel-tools-3.10.0-1127.8.2.el7.x86_64	11/39
Updating	: 7:lvm2-2.02.186-7.el7_8.2.x86_64	12/39
Updating	: containerd.io-1.2.13-3.2.el7.x86_64	13/39
Updating	: systemd-sysv-219-73.el7_8.6.x86_64	14/39
Updating	: 32:bind-export-libs-9.11.4-16.P2.el7_8.3.x86_64	15/39
Updating	: python-perf-3.10.0-1127.8.2.el7.x86_64	16/39
Updating	: yum-utils-1.1.31-54.el7_8.noarch	17/39
Installing	: kernel-3.10.0-1127.8.2.el7.x86_64	18/39
Updating	: binutils-2.27-43.base.el7_8.1.x86_64	19/39
Updating	: yum-plugin-fastestmirror-1.1.31-54.el7_8.noarch	20/39
Cleanup	: systemd-sysv-219-73.el7_8.5.x86_64	21/39
Cleanup	: containerd.io-1.2.13-3.1.el7.x86_64	22/39
Cleanup	: yum-utils-1.1.31-53.el7.noarch	23/39
Cleanup	: yum-plugin-fastestmirror-1.1.31-53.el7.noarch	24/39
Cleanup	: 7:lvm2-2.02.186-7.el7_8.1.x86_64	25/39
Cleanup	: 7:lvm2-libs-2.02.186-7.el7_8.1.x86_64	26/39
Cleanup	: 7:device-mapper-event-1.02.164-7.el7_8.1.x86_64	27/39
Cleanup	: 7:device-mapper-event-libs-1.02.164-7.el7_8.1.x86_64	28/39
Cleanup	: 7:device-mapper-libs-1.02.164-7.el7_8.1.x86_64	29/39
Cleanup	: 7:device-mapper-1.02.164-7.el7_8.1.x86_64	30/39
Cleanup	: systemd-219-73.el7_8.5.x86_64	31/39
Cleanup	: kernel-tools-3.10.0-1127.el7.x86_64	32/39
Cleanup	: 32:bind-libs-lite-9.11.4-16.P2.el7_8.2.x86_64	33/39
Cleanup	: 32:bind-license-9.11.4-16.P2.el7_8.2.noarch	34/39
Cleanup	: kernel-tools-libs-3.10.0-1127.el7.x86_64	35/39
Cleanup	: systemd-libs-219-73.el7_8.5.x86_64	36/39
Cleanup	: 32:bind-export-libs-9.11.4-16.P2.el7_8.2.x86_64	37/39
Cleanup	: python-perf-3.10.0-1127.el7.x86_64	38/39
Cleanup	: binutils-2.27-43.base.el7.x86_64	39/39
Verifying	: containerd.io-1.2.13-3.2.el7.x86_64	1/39
Verifying	: 32:bind-license-9.11.4-16.P2.el7_8.3.noarch	2/39
Verifying	: yum-plugin-fastestmirror-1.1.31-54.el7_8.noarch	3/39
Verifying	: 7:device-mapper-1.02.164-7.el7_8.2.x86_64	4/39

```

Verifying : 32:bind-libs-lite-9.11.4-16.P2.el7_8.3.x86_64 5/39
Verifying : kernel-tools-3.10.0-1127.8.2.el7.x86_64 6/39
Verifying : systemd-sysv-219-73.el7_8.6.x86_64 7/39
Verifying : 7:device-mapper-event-1.02.164-7.el7_8.2.x86_64 8/39
Verifying : systemd-libs-219-73.el7_8.6.x86_64 9/39
Verifying : binutils-2.27-43.base.el7_8.1.x86_64 10/39
Verifying : systemd-219-73.el7_8.6.x86_64 11/39
Verifying : kernel-3.10.0-1127.8.2.el7.x86_64 12/39
Verifying : 7:lvm2-libs-2.02.186-7.el7_8.2.x86_64 13/39
Verifying : kernel-tools-libs-3.10.0-1127.8.2.el7.x86_64 14/39
Verifying : 7:lvm2-2.02.186-7.el7_8.2.x86_64 15/39
Verifying : 7:device-mapper-libs-1.02.164-7.el7_8.2.x86_64 16/39
Verifying : yum-utils-1.1.31-54.el7_8.noarch 17/39
Verifying : python-perf-3.10.0-1127.8.2.el7.x86_64 18/39
Verifying : 32:bind-export-libs-9.11.4-16.P2.el7_8.3.x86_64 19/39
Verifying : 7:device-mapper-event-libs-1.02.164-7.el7_8.2.x86_64 20/39
Verifying : 7:device-mapper-event-1.02.164-7.el7_8.1.x86_64 21/39
Verifying : systemd-219-73.el7_8.5.x86_64 22/39
Verifying : 32:bind-libs-lite-9.11.4-16.P2.el7_8.2.x86_64 23/39
Verifying : 7:device-mapper-event-libs-1.02.164-7.el7_8.1.x86_64 24/39
Verifying : 7:device-mapper-libs-1.02.164-7.el7_8.1.x86_64 25/39
Verifying : 32:bind-export-libs-9.11.4-16.P2.el7_8.2.x86_64 26/39
Verifying : 32:bind-license-9.11.4-16.P2.el7_8.2.noarch 27/39
Verifying : binutils-2.27-43.base.el7.x86_64 28/39
Verifying : 7:device-mapper-1.02.164-7.el7_8.1.x86_64 29/39
Verifying : kernel-tools-3.10.0-1127.el7.x86_64 30/39
Verifying : kernel-tools-libs-3.10.0-1127.el7.x86_64 31/39
Verifying : 7:lvm2-2.02.186-7.el7_8.1.x86_64 32/39
Verifying : systemd-sysv-219-73.el7_8.5.x86_64 33/39
Verifying : yum-utils-1.1.31-53.el7.noarch 34/39
Verifying : python-perf-3.10.0-1127.el7.x86_64 35/39
Verifying : 7:lvm2-libs-2.02.186-7.el7_8.1.x86_64 36/39
Verifying : containerd.io-1.2.13-3.1.el7.x86_64 37/39
Verifying : systemd-libs-219-73.el7_8.5.x86_64 38/39
Verifying : yum-plugin-fastestmirror-1.1.31-53.el7.noarch 39/39

```

Installed:

```
kernel.x86_64 0:3.10.0-1127.8.2.el7
```

Updated:

```

bind-export-libs.x86_64 32:9.11.4-16.P2.el7_8.3      bind-libs-lite.x86_64
32:9.11.4-16.P2.el7_8.3      bind-license.noarch 32:9.11.4-16.P2.el7_8.3
binutils.x86_64 0:2.27-43.base.el7_8.1

```

```

containerd.io.x86_64 0:1.2.13-3.2.el7          device-mapper.x86_64
7:1.02.164-7.el7_8.2          device-mapper-event.x86_64
7:1.02.164-7.el7_8.2          device-mapper-event-libs.x86_64 7:1.02.164-7.el7_8.2
device-mapper-libs.x86_64 7:1.02.164-7.el7_8.2          kernel-tools.x86_64
0:3.10.0-1127.8.2.el7          kernel-tools-libs.x86_64
0:3.10.0-1127.8.2.el7          lvm2.x86_64 7:2.02.186-7.el7_8.2
lvm2-libs.x86_64 7:2.02.186-7.el7_8.2          python-perf.x86_64
0:3.10.0-1127.8.2.el7          systemd.x86_64
0:219-73.el7_8.6          systemd-libs.x86_64 0:219-73.el7_8.6
systemd-sysv.x86_64 0:219-73.el7_8.6          yum-plugin-fastestmirror.noarch
0:1.1.31-54.el7_8          yum-utils.noarch 0:1.1.31-54.el7_8

Complete!

```

If you examine the output above, you'll see that 19 packages were upgraded and a new kernel with the same major release number (3) as the existing kernel was installed.

CentOS does not provide the latest available kernel versions in its software repository. Therefore, the *ElRepo* (*elrepo*) software repository is used. Additionally, CentOS requires all software to be signed; so you must install the *elrepo* GPG signature key before installing the *elrepo* repository.

Installing the *elrepo* GPG signature key requires a single `rpm` command which returns no output if successful:

- **Install GPG Key for ElRepo Software Repository**

```

root@server# rpm --import https://www.elrepo.org/RPM-GPG-KEY-elrepo.org

```

- **Install ElRepo Software Repository**

```

root@server# rpm -Uvh https://www.elrepo.org/elrepo-release-7.0-4.el7.elrepo.noarch.rpm
Retrieving https://www.elrepo.org/elrepo-release-7.0-4.el7.elrepo.noarch.rpm
Preparing...
##### [100%]
Updating/Installing...
  1:elrepo-release-7.0-4.el7.elrepo          ##### [100%]

```

- **Confirm the Latest Kernel in the ElRepo Repository**

```

root@server# yum list available --disablerepo='*' --enablerepo=elrepo-kern
Loaded plugins: fastestmirror

```

```
* elrepo-kernel: linux-mirrors.fnal.gov
Available Packages
kernel-lt.x86_64                4.4.219-1.el7.elrepo      elrepo-
kernel
kernel-lt-devel.x86_64          4.4.219-1.el7.elrepo      elrepo-
kernel
kernel-lt-doc.noarch            4.4.219-1.el7.elrepo      elrepo-
kernel
kernel-lt-headers.x86_64        4.4.219-1.el7.elrepo      elrepo-
kernel
kernel-lt-tools.x86_64          4.4.219-1.el7.elrepo      elrepo-
kernel
kernel-lt-tools-libs.x86_64     4.4.219-1.el7.elrepo      elrepo-
kernel
kernel-lt-tools-libs-devel.x86_64 4.4.219-1.el7.elrepo      elrepo-
kernel
kernel-ml.x86_64                5.6.6-1.el7.elrepo        elrepo-
kernel
kernel-ml-devel.x86_64          5.6.6-1.el7.elrepo        elrepo-
kernel
kernel-ml-doc.noarch            5.6.6-1.el7.elrepo        elrepo-
kernel
kernel-ml-headers.x86_64        5.6.6-1.el7.elrepo        elrepo-
kernel
kernel-ml-tools.x86_64          5.6.6-1.el7.elrepo        elrepo-
kernel
kernel-ml-tools-libs.x86_64     5.6.6-1.el7.elrepo        elrepo-
kernel
kernel-ml-tools-libs-devel.x86_64 5.6.6-1.el7.elrepo        elrepo-
kernel
perf.x86_64                    5.6.6-1.el7.elrepo        elrepo-
kernel
python-perf.x86_64              5.6.6-1.el7.elrepo        elrepo-
kernel
```

Notice the names of the available kernels and tools in the output. The exact versions available will likely differ from what is shown in the example above. The `-lt` kernels (and tools) are stable, *long-term* kernels. The `-ml` kernels (and tools) are *mainline* release kernels with shorter support terms but more frequent releases.

BEST PRACTICE: We recommend using a `-lt` releases for stability.

After reboot, confirm the running kernel.

- **Install Latest Kernel**

```

root@server# yum --enablerepo=elrepo-kernel install kernel-lt
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirror.sfo12.us.leaseweb.net
 * elrepo: linux-mirrors.fnal.gov
 * elrepo-kernel: linux-mirrors.fnal.gov
 * extras: mirror.fileplanet.com
 * updates: mirror.sfo12.us.leaseweb.net
elrepo-kernel | 2.9 kB 00:00:00
elrepo-kernel/primary_db | 1.9 MB 00:00:00
Resolving Dependencies
--> Running transaction check
---> Package kernel-lt.x86_64 0:4.4.223-1.el7.elrepo will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=====
=
Package Arch Version
Repository Size
=====
=
Installing:
kernel-lt x86_64 4.4.223-1.el7.elrepo elrepo-kernel 39 M

Transaction Summary
=====
=
Install 1 Package

Total download size: 39 M
Installed size: 180 M
Is this ok [y/d/N]: y
Downloading packages:
kernel-lt-4.4.223-1.el7.elrepo.x86_64.rpm | 39 MB 00:00:08
Running transaction check
Running transaction test

```

```
Transaction test succeeded
Running transaction
  Installing : kernel-lt-4.4.223-1.el7.elrepo.x86_64      1/1
  Verifying  : kernel-lt-4.4.223-1.el7.elrepo.x86_64      1/1

Installed:
  kernel-lt.x86_64 0:4.4.223-1.el7.elrepo

Complete!
```

- **Reboot to Load New Kernel**

```
root@server# reboot
```

We do not show the output from the reboot command.

- **Verify New Kernel**

```
root@server# uname -msr
Linux 4.4.223-1.el7.elrepo.x86_64 x86_64
```

- **Edit Grub Configuration**

We don't know exactly how *Grub* is configured on your server. The following procedure is, therefore, generic. The goal is to have the boot loader (*grub*) use the new kernel whenever you reboot. Since the previous steps put the new kernel at the top of the list of kernels, in position 0 (zero); we configure *grub* to use the kernel listed in that position by default.

Use the text editor of your choice to edit the file: **/etc/default/grub**. (Editors include *vi*, *nano*, and *emacs*.)

Change the line that says:

```
GRUB_DEFAULT=<some-value>
```

To:

```
GRUB_DEFAULT=0
```

Save the file.

The edited grub configuration needs to be installed so that the server will use it at boot time.

- **Install Grub Configuration**

```
root@server# grub2-mkconfig -o /boot/grub2/grub.cfg
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-4.4.223-1.el7.elrepo.x86_64
Found initrd image: /boot/initramfs-4.4.223-1.el7.elrepo.x86_64.img
Found linux image: /boot/vmlinuz-3.10.0-1127.8.2.el7.x86_64
Found initrd image: /boot/initramfs-3.10.0-1127.8.2.el7.x86_64.img
Found linux image: /boot/vmlinuz-3.10.0-1127.el7.x86_64
Found initrd image: /boot/initramfs-3.10.0-1127.el7.x86_64.img
Found linux image: /boot/vmlinuz-3.10.0-1062.18.1.el7.x86_64
Found initrd image: /boot/initramfs-3.10.0-1062.18.1.el7.x86_64.img
Found linux image: /boot/vmlinuz-3.10.0-957.el7.x86_64
Found initrd image: /boot/initramfs-3.10.0-957.el7.x86_64.img
Found linux image: /boot/vmlinuz-0-rescue-cceb16d4ef4e4e84af1fa8bad95bacb8
Found initrd image: /boot/initramfs-0-rescue-cceb16d4ef4e4e84af1fa8bad95bacb8.img
done
```

The new kernel is now operational. The last step is to confirm that grub uses the new kernel at reboot.

- root@server# **reboot**

RELATED DOCUMENTATION

| [Ubuntu Kernel Upgrade](#) | 75