Juniper Networks | Engineering Simplicity

Paragon Automation Troubleshooting Guide

Published
2023-10-26

RELEASE
23.2

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

Paragon Automation Troubleshooting Guide
23.2

YEAR 2000 NOTICE

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

END USER LICENSE AGREEMENT

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

Table of Contents

About This Guide | vi

1 Troubleshoot Paragon Automation Installation

Resolve Configuration File Merge Conflicts | 2

Troubleshoot Backup and Restore Issues | 3

Troubleshoot RabbitMQ Cluster Failure | 4

Use Log Files to Debug Issues | 5

- Debug Deployment Script Failure | 5
- Debug Installation-Related Issues | 6

Troubleshoot Issues with Ceph and Rook | 7

- Troubleshoot OSD Creation Failure | 7
- Debug Disk Formatting Issues | 7
- Troubleshoot Ceph OSD Failure | 8
- Debug Issues with Rook and Ceph Pods | 10

Troubleshoot Air-Gap Installation Failure | 12

Renew kubeadm-managed Certificates Manually | 13

Recover the Device Controller Group | 16

Create Controller Playbook Manually | 17

Troubleshoot Using the `paragon` CLI Utility | 18

- Understand `paragon` CLI Utility Commands | 18
- List of paragon CLI Utility Commands | 20

2 Debug Paragon Automation Components

User Service Accounts for Debugging | 38

3 Troubleshoot Paragon Pathfinder

General Troubleshooting Techniques | 40

LSP Controller Statuses Overview | 44

LSP Stuck in PENDING or PCC_PENDING State | 48

The Operational State of the LSP Is Down | 51

LSPs Missing from the Network Information Table | 53

Topology Not Displayed in the GUI | 57

Changes Not Reflected in the GUI | 63

Topology Displayed in the GUI Is Incorrect | 68

PCS Out of Sync with Toposerver | 70

PCCs Are Not PCEP-Enabled | 72

4 Troubleshoot Paragon Insights

Debug No-Data | 75

Diagnose Device Reachability Issues | 77

Diagnose Ingest Connectivity Issues | 78

Self-Test Basic Functionality | 79

Identify Kubernetes Cluster Health Issues | 81

Troubleshoot Common Paragon Insights Issues | 82

- Paragon Insights Is Not Listed as a Component of the Paragon Automation Icon on the GUI | 83
- Cannot View Paragon Insights-Related GUI Pages | 83
- License Key Format Is Invalid | 83
- System Log Messages Appear Twice | 84
- Field Data Queried at 60s When Trigger Frequency Is Not Set | 85
- Unable to Recover Data after TSDB Node Fails | 85

5 Troubleshoot Paragon Planner

Unable to Connect to the Paragon Planner Server | 87

Links Missing in the Topology Map | 88

Unable to View Any Archived Networks | 90

About This Guide

Use this guide to understand the most common problems that you might encounter while installing Paragon Automation, and while using the Paragon Pathfinder, Paragon Insights, and Paragon Planner components in the Paragon Automation suite.

RELATED DOCUMENTATION

- Paragon Automation Installation Guide
- Paragon Automation User Guide

PART 1

Troubleshoot Paragon Automation Installation

Resolve Configuration File Merge Conflicts | 2

Troubleshoot Backup and Restore Issues | 3

Troubleshoot RabbitMQ Cluster Failure | 4

Use Log Files to Debug Issues | 5

Troubleshoot Issues with Ceph and Rook | 7

Troubleshoot Air-Gap Installation Failure | 12

Renew kubeadm-managed Certificates Manually | 13

Recover the Device Controller Group | 16

Create Controller Playbook Manually | 17

Troubleshoot Using the `paragon` CLI Utility | 18

Resolve Configuration File Merge Conflicts

IN THIS SECTION

- Problem | 2
- Solution | 2

Problem

Merge conflicts arise when Paragon Automation merges files that the `init` script creates with the existing configuration files.

Solution

If you update an existing installation using the same ***config-dir*** directory that you used for the installation, you may see a conflict. The template configuration files that the `init` script creates are merged with the existing configuration files. This merge action might create a conflict that you must resolve. The `init` script prompts you about how to resolve the conflict.

When prompted, select one of the following options to resolve the conflict:

- C (Default option)—You can retain the existing configuration file and discard the new template file.
- n—You can discard the existing configuration file and reinitialize the template file.
- m—You can merge the files manually. Conflicting sections are marked with lines starting with "<<<<<<<<", "||||||||", "========", and ">>>>>>>>". You must edit the file and remove the merge markers before you proceed with the update.
- d—You can view the differences between the files before you decide how to resolve the conflict.

RELATED DOCUMENTATION

Troubleshoot Paragon Automation Installation

Troubleshoot Backup and Restore Issues

IN THIS SECTION

- Problem | 3
- Solution | 3

Problem

Restoring a configuration from a previously backed-up configuration folder fails.

Solution

When you destroy an existing cluster and redeploy a software image on the same cluster nodes, the restore operation may fail. Restoring the configuration from a previously backed-up configuration folder might cause this failure.

The restore operation fails because the mount path for the backed-up configuration is now changed. The persistent volume is deleted when you destroy an existing cluster. When you redeploy a new image, the persistent volume gets re-created in one of the cluster nodes wherever space is available. However, it is not necessarily created in the same node as it was present in previously. As a result, the restore operation fails.

To prevent this backup and restore issue:

1. Determine the mount path of the new persistent volume.
2. Copy the contents of the previous persistent volume's mount path to the new path.
3. Retry the restore operation.

RELATED DOCUMENTATION

Troubleshoot Paragon Automation Installation

Troubleshoot RabbitMQ Cluster Failure

IN THIS SECTION

- Problem | 4
- Solution | 4

Problem

The RabbitMQ cluster fails as a result of a power outage.

Solution

If the RabbitMQ cluster fails as a result of a power outage, the RabbitMQ message bus may not restart properly. To identify the status of the RabbitMQ message bus, run the `kubectl get po -n northstar -l app=rabbitmq` command. This command should show three pods with their status as `Running`. For example:

```
$ kubectl get po -n northstar -l app=rabbitmq
NAME READY STATUS RESTARTS AGE
rabbitmq-0 1/1 Running 0 10m
rabbitmq-1 1/1 Running 0 10m
rabbitmq-2 1/1 Running 0 9m37s
```

If the status of one or more pods is `Error`, use the following recovery procedure:

1. Delete RabbitMQ.

   ```
   kubectl delete po -n northstar -l app=rabbitmq
   ```

2. Check the status of the pods.

   ```
   kubectl get po -n northstar -l app=rabbitmq.
   ```

   Repeat `kubectl delete po -n northstar -l app=rabbitmq` until the status of all pods is `Running`.

3. Restart the Paragon Pathfinder application.

```
kubectl rollout restart deploy -n northstar
```

RELATED DOCUMENTATION

Troubleshoot Paragon Automation Installation

Use Log Files to Debug Issues

IN THIS SECTION

- Debug Deployment Script Failure | 5
- Debug Installation-Related Issues | 6

These topics provide instructions for debugging issues by reviewing log files.

Debug Deployment Script Failure

Problem

The deployment script fails.

Solution

If the deployment script fails, you must check the installation log files in the ***config-dir*** directory. By default, the ***config-dir*** directory stores six zipped log files. The current log file is saved as **log**, and the previous log files are saved as **log.1** through **log.5**. Every time you run the deployment script (`deploy`), the current log is saved, and the oldest one is discarded.

You typically find error messages at the end of a log file. View the error message, and fix the configuration.

Debug Installation-Related Issues

Problem

You cannot identify installation-related issues.

Solution

You use the Grafana UI, an open-source data visualization tool, to create and to view charts, graphs, and other visuals to help organize and understand data. You can create dashboards to monitor the status of devices, and you can also query data and view the results from the UI. You can then view error messages, identify issues, and fix the configuration. Grafana UI renders data from Paragon Automation time series database (TSDB). For more information, see Grafana Documentation.

To view logs in the Grafana application:

1. Use one of the following methods to access Grafana:
   - Use the virtual IP (VIP) address of the ingress controller: Open a browser and enter **https://** ***vip-of-ingress-controller-or-hostname-of-main-web-application*****/cluster-logs** in the URL field.
   - Use the Logs page: In the Paragon Automation UI, click **Monitoring > Logs** in the left-nav bar.
2. Enter the `grafana_admin_user` username and the `grafana_admin_password` password that you configured in the **config.yml** file during installation. The default username is **admin**.

   If you do not configure the `grafana_admin_password` password, the installer generates a random password. You can retrieve the password using the following command:

   ```
   # kubectl get secret -n kube-system grafana -o jsonpath={..grafana-password} | base64 -d
   ```
3. Click **Home** at the top left corner of the page.
4. Click **Paragon Logs** to view the logs. If it's not already visible, search for and click **Paragon Logs**.
5. (Optional) For instructions on how to create queries, see Query and Transform Data.

SEE ALSO

> *Troubleshoot Paragon Automation Installation*

Troubleshoot Issues with Ceph and Rook

IN THIS SECTION

- Troubleshoot OSD Creation Failure | 7
- Debug Disk Formatting Issues | 7
- Troubleshoot Ceph OSD Failure | 8
- Debug Issues with Rook and Ceph Pods | 10

These topics provide instructions for troubleshooting issues with Ceph and Rook.

Troubleshoot OSD Creation Failure

Problem

Installation fails because object storage daemons (OSDs) are not created.

Solution

A common reason for installation failure is that the object storage daemons (OSDs) are not created. An OSD configures the storage on a cluster node. OSDs might not be created because of non-availability of disk resources, in the form of either insufficient resources or incorrectly partitioned disk space. To prevent installation failing due to insufficient disk space, ensure that the nodes have sufficient unformatted disk space available.

Debug Disk Formatting Issues

Problem

Installation fails when a disk needs to be formatted.

Solution

Ensure sufficient unformatted disk space.

You must examine the logs of the `rook-ceph-osd-prepare-hostname-*` jobs to determine if you need to reformat the disk or partition. To reformat the disk or partition, and restart Rook:

1. Use one of the following methods to reformat an existing disk or partition:

   > ⚠ **CAUTION**: These commands completely reformat the disk or partitions that you are using, and you will lose all data on them.

   - If you have a block storage device that should have been used for Ceph, but wasn't used because it was in an unusable state, you can reformat the disk completely.

     ```
     $ sgdisk -zap /dev/disk
     $ dd if=/dev/zero of=/dev/disk bs=1M count=100
     ```

   - If you have a disk partition that should have been used for Ceph, you can clear the data on the partition completely.

     ```
     $ wipefs -a -f /dev/partition
     $ dd if=/dev/zero of=/dev/partition bs=1M count=100
     ```

2. Restart Rook to save the changes and reattempt the OSD creation process.

   ```
   $ kubectl rollout restart deploy -n rook-ceph rook-ceph-operator
   ```

Troubleshoot Ceph OSD Failure

Problem

The Ceph OSD fails.

Solution

You must identify the failed OSD and remove it. You can then reformat or replace the disk partially or completely.

To troubleshoot disk failure:

1. Run the following command to check the status of Rook and Ceph pods installed in the `rook-ceph` namespace.

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

2. If a `rook-ceph-osd-*` pod is in the `Error` or `CrashLoopBackoff` state, then you must repair the disk.

   Follow these steps to repair the disk:

   a. Stop `rook-ceph-operator`.

      ```
      # kubectl scale deploy -n rook-ceph rook-ceph-operator --replicas=0
      ```

   b. Remove the failing OSD processes.

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

   c. Connect to the toolbox.

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

   d. Identify the failing OSD.

      ```
      # ceph osd status
      ```

   e. Mark the failed OSD out.

      ```
      [root@rook-ceph-tools-/]# ceph osd out 5
      marked out osd.5.
      [root@rook-ceph-tools-/]# ceph osd status
      ID  HOST           USED  AVAIL  WR OPS  WR DATA  RD OPS  RD DATA  STATE
       0  10.xx.xx.210  4856M  75.2G      0        0       0        0   exists,up
       1  10.xx.xx.215  2986M  77.0G      0        0       1       89   exists,up
       2  10.xx.xx.98   3243M  76.8G      0        0       1       15   exists,up
       3  10.xx.xx.195  4945M  75.1G      0        0       0        0   exists,up
       4  10.xx.xx.170  5053M  75.0G      0        0       0        0   exists,up
       5  10.xx.xx.197     0      0       0        0       0        0   exists
      ```

   f. Remove the failed OSD.

      ```
      # ceph osd purge number --yes-i-really-mean-it
      ```

   g. Connect to the node that hosted the failed OSD, and do one of the following:

      - Replace the hard disk in case of a hardware failure.

- Reformat the disk completely.

```
$ sgdisk -zap /dev/disk
$ dd if=/dev/zero of=/dev/disk bs=1M count=100
```

- Reformat the partition completely.

```
$ wipefs -a -f /dev/partition
$ dd if=/dev/zero of=/dev/partition bs=1M count=100
```

h. Restart `rook-ceph-operator`.

```
# kubectl scale deploy -n rook-ceph rook-ceph-operator --replicas=1
```

i. Monitor the OSD pods.

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

If the OSD does not recover, use the same procedure to remove the OSD, and then remove the disk or delete the partition before restarting `rook-ceph-operator`.

SEE ALSO

> *Troubleshoot Paragon Automation Installation*

Debug Issues with Rook and Ceph Pods

Problem

Installation can fail when Rook and Ceph pods are in the `error` state,

Solution

An underpowered hardware can cause a Rook and Ceph pods error. A Rook and Ceph pods error can also cause installation to fail. To solve most issues with Rook and Ceph pods, ensure that the installed pods are in the `running` state. To confirm this, you must:

1. Run the following command to check the status of Rook and Ceph pods installed in the `rook-ceph` namespace.

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

2. Ensure that the following pods are in the `running` state:

   - `rook-ceph-mon-*`—Typically, three monitor pods
   - `rook-ceph-mgr-*`—One manager pod
   - `rook-ceph-osd-*`—Three or more OSD pods
   - `rook-ceph-mds-cephfs-*`—Metadata servers
   - `rook-ceph-rgw-object-store-*`—ObjectStore gateway
   - `rook-ceph-tools*`—For additional debugging options

   To connect to the toolbox, use this command:

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

   For more information on additional common commands in the toolbox, see Table 1 on page 11.

   Table 1: Additional Commands

Command	Description
`# ceph status`	View cluster status.
`# ceph osd status`	View summary of OSD map.
`# ceph osd df`	View details of disk usage (global and per pool).
`# ceph osd utilization`	View OSD utilization.
`# ceph osd pool stats`	View disk pool usage.
`# ceph osd tree`	View OSD tree.
`# ceph pg stat`	View pg status and performance.

Troubleshoot Air-Gap Installation Failure

IN THIS SECTION

- Problem | 12
- Solution | 13

Problem

The air-gap installation fails with the following error:

```
TASK [kubernetes/master : Activate etcd backup cronjob]
*********************************************************************
fatal: [192.xx.xx.2]: FAILED! => changed=true
  cmd:
  - kubectl
  - apply
  - -f
  - /etc/kubernetes/etcd-backup.yaml
  delta: '0:00:00.197012'
  end: '2022-09-13 13:46:31.220256'
  msg: non-zero return code
  rc: 1
  start: '2022-09-13 13:46:31.023244'
  stderr: The connection to the server 192.xx.xx.2:6443 was refused - did you specify the right
host or port?
  stderr_lines: <omitted>
  stdout: ''
  stdout_lines: <omitted>
```

Solution

The air-gap installation as well as the `kube-apiserver` fail because you do not have an existing `/etc/resolv.conf` file.

To create a new file, you must run the `#touch /etc/resolv.conf` command as the root user, and then redeploy the Paragon Automation cluster.

RELATED DOCUMENTATION

Troubleshoot Paragon Automation Installation

Renew kubeadm-managed Certificates Manually

IN THIS SECTION

- Problem | 13
- Solution | 13

Problem

kubeadm-managed certificates expire in one year after deployment. When the certificates expire, pods fail to come up and display bad certificate errors in the log.

Solution

The Paragon Automation Kubernetes cluster uses self generated kubeadm-managed certificates. These certificates expire in one year after deployment unless the Kubernetes version is upgraded or the certificates are manually renewed.

Follow these steps to manually renew certificates:

1. Check the current certificates-expiration date by using the `kubeadm certs check-expiration` command on each primary node of your cluster.

```
    root@primary1-node:~# kubeadm certs check-expiration
    [check-expiration] Reading configuration from the cluster...
    [check-expiration] FYI: You can look at this config file with 'kubectl -n kube-system get
cm kubeadm-config -o yaml'

    CERTIFICATE                EXPIRES                  RESIDUAL TIME   CERTIFICATE
AUTHORITY   EXTERNALLY MANAGED
    admin.conf                 Dec 13, 2023 13:20 UTC
328d                                    no
    apiserver                  Dec 13, 2023 13:20 UTC   328d
ca                      no
    apiserver-etcd-client      Dec 13, 2023 13:20 UTC   328d            etcd-
ca                 no
    apiserver-kubelet-client   Dec 13, 2023 13:20 UTC   328d
ca                      no
    controller-manager.conf    Dec 13, 2023 13:20 UTC
328d                                    no
    etcd-healthcheck-client    Dec 13, 2023 13:20 UTC   328d            etcd-
ca                 no
    etcd-peer                  Dec 13, 2023 13:20 UTC   328d            etcd-
ca                 no
    etcd-server                Dec 13, 2023 13:20 UTC   328d            etcd-
ca                 no
    front-proxy-client         Dec 13, 2023 13:20 UTC   328d            front-proxy-
ca          no
    scheduler.conf             Dec 13, 2023 13:20 UTC
328d                                    no

    CERTIFICATE AUTHORITY   EXPIRES                  RESIDUAL TIME   EXTERNALLY MANAGED
    ca                      Nov 27, 2032 21:31 UTC   9y              no
    etcd-ca                 Nov 27, 2032 21:31 UTC   9y              no
    front-proxy-ca          Nov 27, 2032 21:31 UTC   9y              no
```

2. To renew the certificates, use the `kubeadm certs renew all` command on each primary node of your Kubernetes cluster.

```
        root@primary1-node:~# kubeadm certs renew all
        [renew] Reading configuration from the cluster...
```

```
        [renew] FYI: You can look at this config file with 'kubectl -n kube-system get cm
kubeadm-config -o yaml'

        certificate embedded in the kubeconfig file for the admin to use and for kubeadm
itself renewed
        certificate for serving the Kubernetes API renewed
        certificate the apiserver uses to access etcd renewed
        certificate for the API server to connect to kubelet renewed
        certificate embedded in the kubeconfig file for the controller manager to use renewed
        certificate for liveness probes to healthcheck etcd renewed
        certificate for etcd nodes to communicate with each other renewed
        certificate for serving etcd renewed
        certificate for the front proxy client renewed
        certificate embedded in the kubeconfig file for the scheduler manager to use renewed

        Done renewing certificates. You must restart the kube-apiserver, kube-controller-
manager, kube-scheduler and etcd, so that they can use the new certificates.
```

3. Recheck the expiration date using the `kubeadm certs check-expiration` command on each primary node of your cluster.

```
        root@primary1-node:~# kubeadm certs check-expiration
        [check-expiration] Reading configuration from the cluster...
        [check-expiration] FYI: You can look at this config file with 'kubectl -n kube-system
get cm kubeadm-config -o yaml'

        CERTIFICATE                EXPIRES                  RESIDUAL TIME   CERTIFICATE
AUTHORITY   EXTERNALLY MANAGED
        admin.conf                 Jan 18, 2024 21:40 UTC
364d                                    no
        apiserver                  Jan 18, 2024 21:40 UTC   364d
ca                      no
        apiserver-etcd-client      Jan 18, 2024 21:40 UTC   364d            etcd-
ca                 no
        apiserver-kubelet-client   Jan 18, 2024 21:40 UTC   364d
ca                      no
        controller-manager.conf    Jan 18, 2024 21:40 UTC
364d                                    no
        etcd-healthcheck-client    Jan 18, 2024 21:40 UTC   364d            etcd-
ca                 no
        etcd-peer                  Jan 18, 2024 21:40 UTC   364d            etcd-
```

```
ca                no
        etcd-server               Jan 18, 2024 21:40 UTC   364d            etcd-
ca                no
        front-proxy-client        Jan 18, 2024 21:40 UTC   364d            front-proxy-
ca          no
        scheduler.conf            Jan 18, 2024 21:40 UTC
364d                                    no

        CERTIFICATE AUTHORITY   EXPIRES                  RESIDUAL TIME   EXTERNALLY MANAGED
        ca                      Nov 27, 2032 21:31 UTC   9y              no
        etcd-ca                 Nov 27, 2032 21:31 UTC   9y              no
        front-proxy-ca          Nov 27, 2032 21:31 UTC   9y              no
```

4. Restart the following pods from any one of the primary nodes to use the new certificates.

```
        root@primary1-node:~# kubectl delete pod -n kube-system -l component=kube-apiserver
        root@primary1-node:~# kubectl delete pod -n kube-system -l component=kube-scheduler
        root@primary1-node:~# kubectl delete pod -n kube-system -l component=kube-controller-
manager
        root@primary1-node:~# kubectl delete pod -n kube-system -l component=etcd
```

Recover the Device Controller Group

IN THIS SECTION

- Problem | 16
- Solution | 17

Problem

The default device controller group is deleted.

Solution

To recover the deleted controller group, perform the following steps:

1. Log in to any one of the primary nodes.
2. Run the following commands:

```
# kubectl delete jobs/ns-hbinit -n northstar

# kubectl apply -f /etc/kubernetes/po/hbinit/kube-cfg.yml
```

RELATED DOCUMENTATION

Troubleshoot Paragon Automation Installation

Create Controller Playbook Manually

IN THIS SECTION

- Problem | 17
- Solution | 17

Problem

A controller playbook is not created by default during Paragon Automaton installation.

Solution

A controller playbook defines the rules for collecting analytics data. When a controller playbook is not created during Paragon Automation installation, you must create the controller playbook manually. To do this:

1. Log in to one of the primary nodes.
2. Run the following commands:

```
kubectl delete jobs/ns-hbinit -n northstar
kubectl apply -f /etc/kubernetes/po/hbinit/kube-cfg.yml
```

Troubleshoot Using the paragon CLI Utility

SUMMARY

Read the following topic to understand how to use the paragon command CLI utility.

IN THIS SECTION

- Understand paragon CLI Utility Commands | 18
- List of paragon CLI Utility Commands | 20

Understand paragon CLI Utility Commands

You can use the paragon command CLI utility to run commands on pods running in the system. The paragon commands are a set of intuitive commands to enable you to analyze, query, and troubleshoot your cluster. To execute the commands, log in to any of the primary nodes. The output of some of the commands is color-coded because, for some commands, the paragon command utility executes the kubecolor commands instead of kubectl, kubecolor color codes your kubectl command output. See Figure 1 on page 20 for an example output.

To view the entire set of commands help options available, use one of the following commands:

```
root@primary-node:~# paragon ?
root@primary-node:~# paragon --help
root@primary-node:~# paragon -h
```

You can view help options at any command level (not only at top level). For example:

```
root@primary-node:~# paragon insights cli ?

   paragon insights cli alerta =>  Gets into the CLI of paragon insights alerta pod.
   paragon insights cli byoi =>  Gets into the CLI of byoi plugin.Usage : --byoi <BYOI plugin
name>.
   paragon insights cli configserver =>  Gets into the CLI of paragon insights config-server pod.
   paragon insights cli grafana =>  Gets into the CLI of paragon insights grafana pod.
   paragon insights cli influxdb =>  Gets into the CLI of paragon insights InfluxDB pod.Use
Argument: --influx <influxdb-nodeip> to specify the node ip ,else the command will use first
influx node as default.Eg: --influx influxdb-172-16-18-21
   paragon insights cli mgd =>  Gets into the CLI of paragon insights mgd pod.
```

You can use the tab option to view possible auto-completion options for the commands. To see top-level command auto-completion, type `paragon` and press tab. For example:

```
root@primary-node:~# paragon
ambassador   describe   get   pathfinder   set   common   ems   insights   rookceph
```

To view the underlying command that a paragon command runs, use the echo or `-e` option. For example:

```
root@primary-node:~# paragon -e get nodes all

  >>>> command: kubecolor --force-colors get nodes
```

To execute a paragon command as well as view the underlying command that it runs, use the debug or `-d` option. For example:

```
root@primary-node:~# paragon -d get nodes all

  >>>> command: kubecolor --force-colors get nodes

NAME           STATUS   ROLES                       AGE   VERSION
ix-pgn-pr-01   Ready    control-plane,etcd,master   17d   v1.26.6+rke2r1
ix-pgn-pr-02   Ready    control-plane,etcd,master   17d   v1.26.6+rke2r1
ix-pgn-pr-03   Ready    control-plane,etcd,master   17d   v1.26.6+rke2r1
ix-pgn-wo-01   Ready    <none>                      17d   v1.26.6+rke2r1
```

To view the entire list of `paragon` commands and the corresponding underlying commands that they run, use:

```
root@primary-node:~# paragon --mapped
```

Figure 1: Example `paragon` command output

```
root@ix-pgn-pr-01:~# paragon get nodes all
NAME           STATUS   ROLES                       AGE   VERSION
ix-pgn-pr-01   Ready    control-plane,etcd,master   17d   v1.26.6+rke2r1
ix-pgn-pr-02   Ready    control-plane,etcd,master   17d   v1.26.6+rke2r1
ix-pgn-pr-03   Ready    control-plane,etcd,master   17d   v1.26.6+rke2r1
ix-pgn-wo-01   Ready    <none>                      17d   v1.26.6+rke2r1
root@ix-pgn-pr-01:~# paragon -e get nodes all

  >>>> command: kubecolor --force-colors get nodes

root@ix-pgn-pr-01:~# paragon -d get nodes all

  >>>> command: kubecolor --force-colors get nodes

NAME           STATUS   ROLES                       AGE   VERSION
ix-pgn-pr-01   Ready    control-plane,etcd,master   17d   v1.26.6+rke2r1
ix-pgn-pr-02   Ready    control-plane,etcd,master   17d   v1.26.6+rke2r1
ix-pgn-pr-03   Ready    control-plane,etcd,master   17d   v1.26.6+rke2r1
ix-pgn-wo-01   Ready    <none>                      17d   v1.26.6+rke2r1
```

Follow the instructions with regards to specific usage criteria such as arguments or prerequisites, if any, in the help section of each command. Some commands need mandatory arguments. For instance, the `paragon insights logs devicegroup analytical` command needs the argument `--dg` *`devicegroup-name-with subgroup`*. For example:

`paragon insights logs devicegroup analytical --dg` *`controller-0`*

Some commands have prerequisites. For instance, prior to using the `paragon insights get playbooks` command, you must set the username and password by using the `paragon set username --cred` *`username`* and `paragon set password --cred` *`password`* commands.

List of paragon CLI Utility Commands

The complete set of commands are listed in Table 2 on page 21.

Table 2: paragon CLI Utility

Command	Description
`paragon ambassador get emissary`	Shows Paragon ambassador emissary pods.
`paragon ambassador get pods`	Shows all Paragon ambassador pods.
`paragon ambassador get services`	Shows all Paragon ambassador services.
`paragon common postgres roles`	Helps to find the Postgres roles.
`paragon describe node`	Shows the description of a particular node in the cluster. Use the `--node` *`node-ip`* argument. Example: `paragon describe node --node` *`172.16.x.221`* You can use the `paragon get nodes all` command to get the node IP address.
`paragon ems get devicemanager`	Shows the device manager Paragon ems pods.
`paragon ems get jobmanager`	Shows the job manager Paragon EMS pods.
`paragon ems get pods`	Shows all Paragon EMS pods.
`paragon ems get services`	Shows all Paragon EMS services.
`paragon ems logs devicemanager`	Shows the logs of Paragon EMS device manager pods. Use the `--type follow` argument to get live streaming logs.
`paragon ems logs jobmanager`	Shows the logs of paragon ems job manager pod. Use the `--type follow` argument to get live streaming logs.

Table 2: paragon CLI Utility *(Continued)*

Command	Description
`paragon get namespaces`	Shows all namespaces available in Paragon.
`paragon get nodes all`	Shows a list of all nodes in the cluster.
`paragon get nodes diskpressure`	Validates if kubelet has any disk pressure. Use the `--node` *node_ip/node_name* argument. Example: `paragon get nodes diskpressure --node` *172.16.x.221*
`paragon get nodes memorypressure`	Validates if kubelet has sufficient memory. Use the `--node` *node_ip/node_name* argument. Example: `paragon get nodes memorypressure --node` *172.16.x.221*
`paragon get nodes networkunavailable`	Checks for issues with calico and the network. Use the `--node` *node_ip/node_name* argument. Example: `paragon get nodes networkunavailable --node` *davinci-primary*
`paragon get nodes notready`	Shows list of all nodes that is not ready in the cluster.
`paragon get nodes pidpressure`	Validates if kubelet has sufficient PID available. Use the `--node` *node_ip/node_name* argument. Example: `paragon get nodes pidpressure --node` *davinci-worker1*
`paragon get nodes ready`	Shows list of all nodes that is ready in the cluster.

Table 2: paragon CLI Utility *(Continued)*

Command	Description
`paragon get nodes taint`	Shows list of all taint on the nodes.
`paragon get pods healthy`	Shows all the healthy Paragon pods.
`paragon get pods unhealthy`	Shows all the unhealthy Paragon pods.
`paragon get services exposed`	Shows all the Paragon services that are exposed.
`paragon insights cli alerta`	Logs in to the CLI of the Paragon Insights alerta pod.
`paragon insights cli byoi`	Logs in to the CLI of the BYOI plug-in. Use the `--byoi` *`BYOI plugin name`* argument.
`paragon insights cli configserver`	Logs in to the CLI of Paragon Insights config-server pod.
`paragon insights cli grafana`	Logs in to the CLI of Paragon Insights grafana pod.
`paragon insights cli influxdb`	Logs in to the CLI of Paragon Insights influxdb pod. Use the `--influx` *`influxdb-nodeip`* argument to specify the node IP If not, the command will use the first influxdb node as the default node. Example: `paragon insights cli influxdb --influx` *`influxdb-172.16.x.21`*
`paragon insights cli mgd`	Logs in to the CLI of Paragon Insights mgd pod.
`paragon insights describe alerta`	Describes the Paragon Insights alerta pod.
`paragon insights describe api`	Describes the Paragon Insights REST API pod.

Table 2: paragon CLI Utility *(Continued)*

Command	Description
`paragon insights describe configserver`	Describes the Paragon Insights config-server pod.
`paragon insights describe grafana`	Describes the Paragon Insights grafana pod.
`paragon insights describe influxdb`	Describes the Paragon Insights influxdb pod. Use the `--influx influxdb-nodeip` argument to specify the node IP. If not, the command will use the first influxdb node as the default node. Example: `paragon insights describe influxdb --influx` *`influxdb-172.16.x.21`*
`paragon insights describe mgd`	Describes the Paragon Insights mgd pod.
`paragon insights get alerta`	Shows the Paragon Insights alerta pod.
`paragon insights get api`	Shows the Paragon Insights REST API pod.
`paragon insights get configserver`	Shows the Paragon Insights config-server pod.
`paragon insights get devicegroups`	Shows all the Paragon Insights device groups. The default username is `admin`. To modify the username, run the `paragon set user --cred` *`username`*> command. As a prerequisite, run the `paragon set password --cred` *`password`* command to set the Paragon (UI host) password.

Table 2: paragon CLI Utility *(Continued)*

Command	Description
`paragon insights get devices`	Shows all Paragon Insights devices. The default username is `admin`. To modify the username, run the `paragon set user --cred` *`username`* command. As a prerequisite, run the `paragon set password --cred` *`password`* command to set the Paragon (UI host) password.
`paragon insights get grafana`	Shows the Paragon Insights grafana pod.
`paragon insights get influxdb`	Shows the Paragon Insights influxdb pod.
`paragon insights get ingest`	Shows the Paragon Insights network telemetry ingestion pods.
`paragon insights get mgd`	Shows the Paragon Insights mgd pod.
`paragon insights get playbooks`	Shows all Paragon Insights playbooks. The default username is `admin`. To modify the username, run the `paragon set user --cred` *`username`* command. As a prerequisite, run the `paragon set password --cred` *`password`* command to set the Paragon (UI host) password.
`paragon insights get pods`	Shows all the Paragon Insights pods.
`paragon insights get services`	Shows all the Paragon Insights services.
`paragon insights logs alerta`	Shows the logs of the Paragon Insights alerta pod.

Table 2: paragon CLI Utility *(Continued)*

Command	Description
`paragon insights logs api`	Shows the logs of the Paragon Insights rest api pod.
`paragon insights logs byoi`	Shows the logs of the Paragon Insights BYOI plug-in. Use the `--byoi` *`BYOI plugin name`* argument.
`paragon insights logs configserver`	Shows the logs of the Paragon Insights config-server pod.
`paragon insights logs devicegroup analytical`	Shows the logs of the Paragon Insights device group for service analytical engine. Use the `--dg` *`device Group name with subgroup`* argument. Example: `paragon insights logs devicegroup analytical --dg` *`controller-0`* In the example, *controller* is the devicegroup name and *0* is the subgroup.
`paragon insights logs devicegroup itsdb`	Shows the logs of the Paragon Insights device group for service itsdb. Use the `--dg` *`device Group name with subgroup`* argument. Example: `paragon insights logs devicegroup itsdb --dg` *`controller-0`* In the example, *controller* is the devicegroup name and *0* is the subgroup.

Table 2: paragon CLI Utility *(Continued)*

Command	Description
`paragon insights logs devicegroup jtimon`	Shows the logs of the Paragon Insights device group for service jtimon. Use the `--dg` *`device Group name with subgroup`* argument. Example: `paragon insights logs devicegroup jtimon --dg` *`controller-0`* In the example, *controller* is the devicegroup name and *0* is the subgroup.
`paragon insights logs devicegroup native`	Shows the logs of the Paragon Insights device group for service jti native. Use the `--dg` *`device Group name with subgroup`* argument. Example: `paragon insights logs devicegroup native --dg` *`controller-0`* In the example, *controller* is the devicegroup name and *0* is the subgroup.
`paragon insights logs devicegroup syslog`	Shows the logs of the Paragon Insights device group for service syslog. Use the `--dg` *`device Group name with subgroup`* argument. Example: `paragon insights logs devicegroup syslog --dg` *`controller-0`* In the example, *controller* is the devicegroup name and *0* is the subgroup.
`paragon insights logs grafana`	Shows the logs of the Paragon Insights Grafana pod.

Table 2: paragon CLI Utility *(Continued)*

Command	Description
`paragon insights logs influxdb`	Shows the logs of the Paragon Insights influxdb pod. Use the `--influx influxdb-nodeip` argument to specify the node IP. If not, the command will use the first influxdb node as the default node. Example: `paragon insights logs influxdb --influx` *`influxdb-172.16.x.21`*
`paragon insights logs mgd`	Shows the logs of the Paragon Insights mgd pod.
`paragon pathfinder cli bmp`	Logs in to the CLI of the Paragon Pathfinder BMP container.
`paragon pathfinder cli configserver`	Logs in to the CLI of the Paragon Pathfinder ns-configserver container.
`paragon pathfinder cli crpd`	Logs in to the CLI of the Paragon Pathfinder cRPD container.
`paragon pathfinder cli debugutils`	Logs in to the CLI of the Paragon Pathfinder debugutils container.
`paragon pathfinder cli netconf`	Logs in to the CLI of the Paragon Pathfinder netconf container.
`paragon pathfinder cli pceserver`	Logs in to the CLI of the Paragon Pathfinder ns-pceserver container (PCEP) services.
`paragon pathfinder cli pcserver`	Logs in to the CLI of the Paragon Pathfinder ns-pcserver (PCS) container.
`paragon pathfinder cli pcviewer`	Logs in to the CLI of the Paragon Pathfinder ns-pcsviewer (Paragon Planner Desktop Application) container.

Table 2: paragon CLI Utility *(Continued)*

Command	Description
`paragon pathfinder cli scheduler`	Gets into the CLI of paragon pathfinder scheduler container.
`paragon pathfinder cli toposerver`	Logs into the CLI of the Paragon Pathfinder ns-toposerver (Topology service) container.
`paragon pathfinder cli web`	Logs into the CLI of the Paragon Pathfinder ns-web container.
`paragon pathfinder debug bgpls config`	Debugs the Paragon Pathfinder cRPD routing-options configuration related to BGP-LS.
`paragon pathfinder debug bgpls routes`	Debugs the Paragon Pathfinder cRPD routes related to BGP-LS.
`paragon pathfinder debug genjvisiondata help`	Shows Paragon Pathfinder debugutils genjvisiondata help.
`paragon pathfinder debug genjvisiondata params`	Shows Paragon Pathfinder debugutils genjvisiondata params.
`paragon pathfinder debug lsp`	Logs in to the Paragon Pathfinder PCEP CLI for debugging.
`paragon pathfinder debug postgres status`	Shows the Kubernetes cluster Postgres status.
`paragon pathfinder debug rabbitmq status`	Shows the rabbitmqctl cluster status.
`paragon pathfinder debug snoop amqp`	Runs Paragon Pathfinder debugutils pod to snoop and decode data exchanged between AMQP.
`paragon pathfinder debug snoop help`	Shows Paragon Pathfinder debugutils snoop help.

Table 2: paragon CLI Utility *(Continued)*

Command	Description
`paragon pathfinder debug snoop postgres`	Runs Paragon Pathfinder debugutils pod to snoop and decode data exchanged between Postgres.
`paragon pathfinder debug snoop redis link`	Runs Paragon Pathfinder debugutils pod to snoop and decode data exchanged between Redis link.
`paragon pathfinder debug snoop redis lsp`	Runs Paragon Pathfinder debugutils pod to snoop and decode data exchanged between Redis lsp.
`paragon pathfinder debug snoop redis node`	Runs Paragon Pathfinder debugutils pod to snoop and decode data exchanged between redis nodes.
`paragon pathfinder debug topoutil help`	Shows Paragon Pathfinder debugutils topo_util help.
`paragon pathfinder debug topoutil safemode deactivate`	Shows Paragon Pathfinder debugutils topo_util tool to deactivate safe mode.
`paragon pathfinder debug topoutil topo refresh`	Runs Paragon Pathfinder debugutils topo_util tool to refresh the current topology.
`paragon pathfinder debug topoutil topo save`	Runs Paragon Pathfinder debugutils topo_util tool to save the current topology snapshot.
`paragon pathfinder describe bmp`	Describes Paragon Pathfinder pod including cRPD and BMP containers.
`paragon pathfinder describe configserver`	Describes Paragon Pathfinder pod including config-server container.
`paragon pathfinder describe debugutils`	Describes Paragon Pathfinder pod including debugutils container.

Table 2: paragon CLI Utility *(Continued)*

Command	Description
`paragon pathfinder describe netconf`	Describes Paragon Pathfinder pod including ns-netconfd container.
`paragon pathfinder describe pceserver`	Describes Paragon Pathfinder pod including ns-pceserver container (PCEP services).
`paragon pathfinder describe pcserver`	Describes Paragon Pathfinder pod including ns-pcserver container (PCS).
`paragon pathfinder describe pcviewer`	Describes paragon pathfinder pod including ns-pcsviewer container (Paragon Planner Desktop Application).
`paragon pathfinder describe scheduler`	Describes Paragon Pathfinder pod including scheduler container.
`paragon pathfinder describe toposerver`	Describes Paragon Pathfinder pod including ns-toposerver (Topology service) container.
`paragon pathfinder describe web`	Describes Paragon Pathfinder pod including web container.
`paragon pathfinder get bmp`	Shows Paragon Pathfinder pod including cRPD and BMP containers.
`paragon pathfinder get configserver`	Shows Paragon Pathfinder pod including ns-configserver and syslog containers.
`paragon pathfinder get debugutils`	Shows Paragon Pathfinder pod including debugutils container.
`paragon pathfinder get netconf`	Shows Paragon Pathfinder pod associated with the netconf process.

Table 2: paragon CLI Utility *(Continued)*

Command	Description
`paragon pathfinder get pceserver`	Shows Paragon Pathfinder pod including ns-pceserver container (PCEP services).
`paragon pathfinder get pcserver`	Shows Paragon Pathfinder pod including ns-pcserver container (PCS).
`paragon pathfinder get pcviewer`	Shows Paragon Pathfinder pod including ns-pcsviewer container (Paragon Planner Desktop Application).
`paragon pathfinder get pods`	Shows all Paragon Pathfinder pods.
`paragon pathfinder get scheduler`	Shows Paragon Pathfinder pod associated with the scheduler process.
`paragon pathfinder get services`	Shows all Paragon Pathfinder services.
`paragon pathfinder get toposerver`	Shows Paragon Pathfinder pod including ns-toposerver container (Topology service).
`paragon pathfinder get web`	Shows Paragon Pathfinder pod associated with the ns-web process.
`paragon pathfinder logs bmp container bmp`	Shows the logs of Paragon Pathfinder bmp pods bmp container. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs bmp container crpd`	Shows the logs of Paragon Pathfinder bmp pods cRPD container. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs bmp container syslog`	Shows the logs of Paragon Pathfinder bmp pods syslog container. Use the `--type follow` argument to get live streaming logs.

Table 2: paragon CLI Utility *(Continued)*

Command	Description
`paragon pathfinder logs configserver container nsconfigserver`	Shows the logs of Paragon Pathfinder configserver pods ns-configserver container. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs configserver container syslog`	Shows the logs of Paragon Pathfinder configserver pods syslog container. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs netconf container nsnetconfd`	Shows the logs of Paragon Pathfinder netconf pods ns-netconfd container. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs netconf container syslog`	Shows the logs of Paragon Pathfinder netconf pods syslog container. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs pceserver container nspceserver`	Shows the logs of Paragon Pathfinder pceserver pods ns-pceserver container. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs pceserver container syslog`	Shows the logs of Paragon Pathfinder pceserver pods syslog container. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs pceserver syslog filtered`	Shows processed logs of Paragon Pathfinder pceserver pods syslog container fetching only timestamp, level, and message. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs pcserver container nspcserver`	Shows the logs of Paragon Pathfinder pcserver pods ns-pcserver container. Use the `--type follow` argument to get live streaming logs.

Table 2: paragon CLI Utility *(Continued)*

Command	Description
`paragon pathfinder logs pcserver container syslog`	Shows the logs of Paragon Pathfinder pcserver pods syslog container. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs pcserver syslog filtered`	Shows processed logs of Paragon Pathfinder pceserver pods syslog container fetching only with timestamp, level, and message. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs pcviewer container nspcviewer`	Shows the logs of Paragon Pathfinder pcviewer pods ns-pcviewer container. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs pcviewer container syslog`	Shows the logs of Paragon Pathfinder pcviewer pods syslog container. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs toposerver container nstopodbinit`	Shows the logs of Paragon Pathfinder toposerver pods ns-topo-dbinit container. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs toposerver container nstopodbinitcache`	Shows the logs of Paragon Pathfinder toposerver pods ns-topo-dbinit-cache container. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs toposerver container nstoposerver`	Shows the logs of Paragon Pathfinder toposerver pods ns-toposerver container. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs toposerver container syslog`	Shows the logs of Paragon Pathfinder toposerver pods syslog container. Use the `--type follow` argument to get live streaming logs.

Table 2: paragon CLI Utility *(Continued)*

Command	Description
`paragon pathfinder logs toposerver syslog filtered`	Shows processed logs of Paragon Pathfinder toposerver pods syslog container fetching only with timestamp, level, and message. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs web container nsweb`	Shows the logs of Paragon Pathfinder web pods ns-web container. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs web container nswebdbinit`	Shows the logs of Paragon Pathfinder web pods ns-web-dbinit container. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder logs web container syslog`	Shows the logs of Paragon Pathfinder web pods syslog container. Use the `--type follow` argument to get live streaming logs.
`paragon pathfinder rabbitmq geoha status`	Shows the federation status (from rabbitmq-0 instance). GeoHa status is only available for a dual cluster setup.
`paragon rookceph ceph osddf`	Reports Rook and Ceph OSD file system disk space usage.
`paragon rookceph ceph osdpoolstats`	Shows Rook and Ceph OSD pool statistics.
`paragon rookceph ceph osdstatus`	Shows Rook and Ceph OSD status.
`paragon rookceph ceph osdtree`	Shows Rook and Ceph OSD tree.
`paragon rookceph ceph osdutilization`	Shows Rook and Ceph OSD utilization.
`paragon rookceph ceph pgstat`	Shows Rook and Ceph pg status.

Table 2: paragon CLI Utility *(Continued)*

Command	Description
`paragon rookceph ceph status`	Shows Rook and Ceph status.
`paragon rookceph cli toolbox`	Logs in to the CLI of Rook and Ceph toolbox pod.
`paragon rookceph get pods`	Shows Rook and Ceph pods.
`paragon rookceph get services`	Shows Rook and Ceph services.
`paragon rookceph radosgw get period`	This is RADOS gateway user administration utility which gets the period info.
`paragon rookceph radosgw synch status`	This is RADOS gateway user administration utility which gets the metadata sync status.
`paragon set password`	Sets the Paragon (UI host) password for REST calls authentication. Use this mandatory one-time set password command to set the password using the `--cred` *`password`* argument. Example: `paragon set password --cred` *`AdminXYX!`*
`paragon set username`	Sets the Paragon (UI host) username for Rest calls authentication. The default username is `admin`. Use the `--cred` *`username`* argument to set a different username. Example: `paragon set username --cred` *`newadmin`*

PART 2

Debug Paragon Automation Components

User Service Accounts for Debugging | 38

User Service Accounts for Debugging

Paragon Pathfinder, telemetry manager, and base platform applications internally use Paragon Insights for telemetry collection. To facilitate debugging of configuration issues associated with these applications, three user service accounts are created, by default, during Paragon Automation installation. The scope of these service accounts is limited to debugging the corresponding application only. The service accounts details are listed in the following table.

Table 3: Service Account Details

Application Name and Scope	Account Username	Account Default Password
Paragon Pathfinder (northstar)	hb-northstar-admin	Admin123!
Telemetry manager (tm)	hb-tm-admin	
Base platform (ems-dmon)	hb-ems-dmon	

For example, to debug issues with Paragon Pathfinder, log in to the Paragon Automation GUI using *hb-northstar-admin* as the username and *Admin123!* as the password. We recommend that you change the login credentials upon first use, for security reasons.

You must use these accounts solely for debugging purposes. Do not use these accounts for day-to-day operations or for modifying any configuration.

PART 3

Troubleshoot Paragon Pathfinder

General Troubleshooting Techniques | 40

LSP Controller Statuses Overview | 44

LSP Stuck in PENDING or PCC_PENDING State | 48

The Operational State of the LSP Is Down | 51

LSPs Missing from the Network Information Table | 53

Topology Not Displayed in the GUI | 57

Changes Not Reflected in the GUI | 63

Topology Displayed in the GUI Is Incorrect | 68

PCS Out of Sync with Toposerver | 70

PCCs Are Not PCEP-Enabled | 72

General Troubleshooting Techniques

If you encounter a problem while using Paragon Pathfinder, you can use the following general troubleshooting techniques to identify the cause of the problem:

- Before you begin any troubleshooting investigation, confirm that all pods associated with the system processes are running.

  To confirm whether a pod is running, do the following:

  1. Use your server credentials to log in to the primary node of Paragon Automation.
  2. From the primary node, use the following command to confirm whether the pods associated with the system processes are running:

     ```
     kubectl -n northstar get pod
     ```

     The output displays the list of pods and their details. For each pod that is running, the Status column corresponding to that pod displays the word RUNNING. Pods that have stopped running then restart automatically.

- If a pod is not in the Running state, view the events associated with the pod by using the following command. The events help you identify the problem:

  ```
  kubectl -n northstar describe pod pod-name
  ```

  In the command, *pod-name* is the name of the pod for which you want to view the events.

- Confirm whether a system process is running by logging in to the pod associated with the process and view the logs. See the "View Logs" on page 42 for details.

- Run CLI commands—You can run CLI commands (ping, traceroute, and vendor-specific command sets) on the following network elements without manually logging in to the devices. You can perform this task from the Diagnostics page in the GUI.

  NOTE: You must be able to connect to the devices through SSH; otherwise, you will not be able to run CLI commands.

  - Nodes—For details about running CLI commands on nodes, see *About the Node Tab* in the Paragon Automation User Guide.

- Links—For details about running CLI commands on links, see *About the Link Tab* in the Paragon Automation User Guide.
- Tunnels—For details about running CLI commands on tunnels, see *About the Tunnel Tab* in the Paragon Automation User Guide.
- Interfaces—For details about running CLI commands on interfaces, see *About the Interface Tab* in the Paragon Automation User Guide.

The CLI commands for nodes, links, tunnels, and interfaces are present in the corresponding configuration files located in their respective pods. You can modify these configuration files by using a text editor of your choice.

For example, the CLI commands configuration file (**clicommands.json**) contains vendor-specific command sets. By default, the commands in the file are organized into categories such as Accounting Info, Chassis, or File Operation. The same categories appear in the Diagnostics list when you run CLI commands from the GUI. These categories are vendor-specific. When you select a node on which to run CLI commands, Paragon Pathfinder collects the vendor type of the node. Paragon Pathfinder then displays the categories and command sets that are applicable. You can modify the category names and the organization of commands within the categories. You can also add commands. The same is true for the ping and traceroute categories.

The following example shows the structure of entries in the **clicommands.json** file:

```
{
   "cmd": "show config | display inheritance|no-more",
   "displayName": "Show configuration",
   "cmdId": 380
},
{
   "cmd": "show config logical (LR) | display inheritance|no-more",
   "displayName": "Show configuration (LR)",
   "cmdId": 1571
},
```

To modify the configuration files, do the following:

1. Log in to the primary node of Paragon Automation by using your server credentials.
2. From the primary node, obtain the name of the pod associated with the ns-web process by using the following command:

.

```
kubectl -n northstar get pod | grep ns-web
```

The log file for the ns-web process contains log entries for the REST API requests from the GUI

3. Log in to the pod associated with the ns-web process by using the following command:

```
kubectl -n northstar exec -it ns-web-pod-name -c ns-web -- bash
```

In the command, *ns-web-pod-name* is the name of the pod that you obtained in "Step 2" on page 41.

4. View the list of configuration files present in the pod by using the following command:

```
ls /opt/pcs/db/diagnostic/
```

The output displays the ping (`pingcommands.json`), traceroute (`traceroutecommands.json`), and CLI (`clicommands.json`) configuration files.

- View logs—You can view logs for all Paragon Automation components on the Logs page (**Monitoring > Logs**) in the GUI.

  Alternatively, you can view the logs by using the CLI.

  The following example explains the procedure to view the logs associated with the Config Server process by using the CLI. You can use the same procedure to view the logs for other processes.

  1. Log in to the primary node of Paragon Automation by using your server credentials.
  2. From the primary node, obtain the name of the pod associated with the Config Server process by using the following command:

```
kubectl -n northstar get pod | grep configserver
```

  3. View the logs associated with the Config Server process by using the following command:

```
kubectl -n northstar logs -f ns-configserver-pod-name -c syslog
```

  In the command, *ns-configserver-pod-name* is the name of the pod that you obtained in "Step 2" on page 42.

The output displays the logs associated with the Config Server process.

Table 1 on page 43 provides information about the most commonly used log files for troubleshooting Paragon Pathfinder.

Table 4: Commonly Used Paragon Pathfinder Log Files

Log File	Description
ns-pceserver	The log file contains log entries related to the Path Computation Element (PCE) server. The PCE server maintains the Path Computation Element Protocol (PCEP) session. The logs contain information about communication between the Path Computation Client (PCC) and the PCE in both directions.
ns-pcserver	The log file contains log entries related to the Path Computation Server (PCS). The PCS is responsible for path computation. The logs include events that the PCS receives from the Toposerver, including provisioning orders. The logs also contain notifications of communication errors and issues that prevent the PCS from starting properly.
ns-toposerver	The log file contains log entries that are related to the topology server (also called a Toposerver in Paragon Pathfinder). The Toposerver is responsible for maintaining the topology. The logs contain a record of the events between the PCS and the Toposerver, the Toposerver and the BGP Monitoring Protocol (BMP), and the Toposerver and the PCE server.
ns-web	The log file contains log entries related to the REST API requests from the Paragon Pathfinder GUI.

- View historical events for links and tunnels—On the Events page in the GUI, you can view historical events associated with links and tunnels. A graph with a timeline view displays the events for a selected time range. The Events table at the bottom of the page displays the details of these events.

  NOTE: The events displayed on the Events page pertain only to external communication to and from the PCE. Most of the communications internal to the PCE are captured only in the log files.

- To access the Events page for links, navigate to the Link tab in the network information table (**Network > Topology**). Select the link for which you want to view the events, and select **View > Link Events**. The Events page appears. For more information, see *About the Link Tab* in the Paragon Automation User Guide.
- To access the Events page for tunnels, navigate to the Tunnel tab in the network information table (**Network > Topology**). Select the tunnel for which you want to view the events, and select **View > Events**. The Events page appears. For more information, see *About the Tunnel Tab* in the Paragon Automation User Guide.

If you encounter a problem with the network that Paragon Pathfinder controls, you can run CLI commands for various elements in the network to identify the cause of the problem. See "Run CLI Commands" on page 40 for details.

LSP Controller Statuses Overview

A label-switched path (LSP) passes through various provisioning states before it is successfully provisioned in the network. You can view these provisioning states in the **Controller Status** column on the Tunnel tab of the network information table (**Network > Topology**).

Table 5 on page 44 lists the various LSP controller statuses and their meanings.

Table 5: LSP Controller Statuses

This Controller Status	Indicates That
FAILED	The controller has failed to provision the LSP.
PENDING	The Path Computation Server (PCS) has sent a provisioning order for the LSP to the Path Computation Element (PCE) server. The PCS is awaiting a response from the PCE server.
PCC_PENDING	The PCE server has sent an LSP provisioning order to the Path Computation Client (PCC). The PCS is awaiting a response from the PCC.
NETCONF_PENDING	The PCS has sent an LSP provisioning order to the netconfd process. The PCS is awaiting a response from the netconfd process.

Table 5: LSP Controller Statuses *(Continued)*

This Controller Status	Indicates That
PRPD_PENDING	The PCS has sent an LSP provisioning order to the programmable routing protocol process (daemon) [PRPD] client to provision a BGP route. The PCS is awaiting a response from the PRPD client.
SCHEDULED_DELETE	The PCS has scheduled the LSP to be deleted; the PCS will send the deletion provisioning order to the PCC.
SCHEDULED_DISCONNECT	The PCS has scheduled the LSP to be disconnected. The LSP will be moved to Shutdown status. Therefore, the LSP is retained in the Pathfinder database and the Persist state is associated with it. The LSP is not used in Constrained Shortest Path First (CSPF) calculations.
NoRoute_Rescheduled	The PCS hasn't found a path for the LSP. The PCS will scan the LSPs periodically and will try to find a path for the LSP that is not routed. The PCS will then schedule the reprovisioning of the LSP.
FRR_DETOUR_Rescheduled	The PCS has detoured the LSP and rescheduled the LSP's re-provisioning.
Provision_Rescheduled	The PCS has scheduled the LSP to be provisioned.
Maint_NotHandled	The LSP is not a part of the ongoing maintenance event, because Pathfinder does not control the LSP.
Maint_Rerouted	The PCS has rerouted the LSP due to maintenance.
Callsetup_Scheduled	The PCS must provision the LSP when an event that triggers LSP provisioning starts.
Disconnect_Scheduled	The PCS must disconnect the LSP when the event that triggers LSP provisioning ends.
No path found	The PCS was unable to find a path for the LSP.

Table 5: LSP Controller Statuses *(Continued)*

This Controller Status	Indicates That
Path found on down LSP	The PCS has found a path for the LSP, but the PCE server has reported that the LSP is down.
Path include loops	The segment routing-LSP has one or more loops.
Maint_NotReroute_DivPathUp	The PCS doesn't reroute the LSP due to a maintenance event, because a standby path is already up and running.
Maint_NotReroute_NodeDown	The PCE doesn't reroute the LSP because the maintenance event is for the endpoints of the LSP.
PLANNED_LSP	The LSP must be provisioned but is not in the provisioning queue yet.
PLANNED_DISCONNECT	The LSP must be disconnected but is not in the provisioning queue yet.
PLANNED_DELETE	The LSP must be deleted but is not in the provisioning queue yet.
Candidate_ReOptimization	The PCS has selected the LSP as a candidate for reoptimization.
Activated (used_by_primary)	The secondary path for the LSP is activated.
Time_Expired	The scheduled window for the LSP has expired.
PCEP_Capability_not_supported	The device may not support PCEP. If the device supports PCEP, it may not be configured, may be disabled, or may be misconfigured on the device.
De-activated	The controller has deactivated the secondary LSP.

Table 5: LSP Controller Statuses *(Continued)*

This Controller Status	Indicates That

NS_ERR_NCC_NOT_FOUND	The controller is unable to use the NETCONF Connection Client (NCC) to establish a NETCONF connection with the device to create a NETCONF-based LSP.
	
	Workaround:
	
	Log in to the NETCONF pod and view the logs to troubleshoot the issue. See "View Logs" on page 42 for the procedure to view logs:
	
	• To solve a synchronization issue, you can re-synchronize the network model from the Pathfinder page (**Configuration > Network Settings > Pathfinder Settings > Advanced Settings**) in the GUI. See "Synchronize the Network Model" on page 64 for details.
	
	• If the NETCONF log doesn't print a message, restart the NETCONF pod on the Kubernetes server by using one of the following commands:
	
	  • `kubectl delete pod netconf-pod-name -n northstar`
	
	  • `kubectl rollout restart deployment ns-netconfd -n northstar`

SR LSP provisioning requires LSP stateful SR capability	The Junos OS device is not configured with the segment routing capability. Configure the following command on the Junos OS device through the CLI to provision the segment routing LSP: `set protocols pcep pce name spring-capability`

To decide the corrective actions that you must perform to solve problems related to the provisioning of LSPs, refer to the log files. You can view the log files either from the Logs page (**Monitoring > Logs**) in the GUI or by manually logging in to the respective pods by using the CLI. To find the procedure for viewing logs from the CLI, see "General Troubleshooting Techniques" on page 40.

LSP Stuck in PENDING or PCC_PENDING State

IN THIS SECTION

- Problem | 48
- Solution | 48

Problem

The Controller Status of the LSP may display a pending status. This error may persist even after the topology server (Toposerver) marks the nodes as PCEP-enabled and you provision the label-switched path (LSP). The Controller Status (as seen in the Tunnel tab of the network information table) displays the word PENDING or PCC_PENDING.

Solution

Through the provisioning process, an LSP has various provisioning states that indicate the status of provisioning. The Controller Status column on the Tunnel tab of the network information table displays these provisioning states. The most common LSP states are **PENDING** and **PCC_PENDING**.

The following sequence describes what happens behind the scenes:

1. To provision an LSP, the Path Computation Server (PCS) computes a path that satisfies all the requirements for the LSP. The PCS then sends a provisioning order to the Path Computation Element (PCE) server.

   Following is an example of the log message that appears in the PCS log while this process is taking place:

NOTE: The logs that you view may differ from the logs in this document. The logs change based on the installed version of Paragon Automation.

```
Apr Apr 25 10:06:44.798336 user-PCS PCServer  [->TopoServer] push lsp configlet, action=ADD
Apr 25 10:06:44.798341 user-PCS PCServer  {#012"lsps":[#012{"request-
id":928380025,"name":"JTAC","from":"10.0.0.102",
 "to":"10.0.0.104","pcc":"172.25.158.66","bandwidth":"100000","metric":0,"local-
protection":false,"type":"primary",
"association-group-id":0,"path-attributes":{"admin-group":{"exclude":0,"include-all":0,
"include-any":0},"setup-priority":
7,"reservation-priority":7,"ero":[{"ipv4-address":"10.102.105.2"},{"ipv4-
address":"10.105.107.2"}, {"ipv4-address":
"10.114.117.1"}]}}#012]#012}
Apr 25 10:06:44.802500 user-PCS PCServer  provisioning order sent, status = SUCCESS
Apr 25 10:06:44.802519 user-PCS PCServer  [->TopoServer] Save LSP action, id=928380025
event=Provisioning Order(ADD) sent request_id=928380025
Apr 25 10:06:44.802534 user-PCS PCServer   lsp action=ADD JTAC@10.0.0.102 path=
controller_state=PENDING
```

2. The LSP's controller status is PENDING. This status means that the PCS has sent the provisioning order to the PCE server, but the PCS has not received an acknowledgment yet. If an LSP remains in the PENDING state, it suggests that the problem lies with the PCE server.

   You can identify the problem by viewing the logs for the PCE server.

   To view the logs for the PCE server:

   a. Use your server credentials to log in to the primary node of Paragon Automation.

   b. From the primary node, use the following command to obtain the name of the pod in which the PCE server process runs:

   ```
   kubectl -n northstar get pod | grep pceserver
   ```

   c. Log in to the PCE server pod by using the following command:

   ```
   kubectl -n northstar exec -it ns-pceserver-pod-name -c ns-pceserver -- bash
   ```

In the command, *ns-pceserver-pod-name* is the name of the pod that you obtained in "Step b" on page 49.

d. View the logs associated with the PCE server by using the following command:

```
kubectl -n northstar logs -f ns-pceserver-pod-name -c syslog
```

The following is a sample PCE server log entry:

```
2016-04-25 10:06:45.196263(27897)[EVENT]: 172.25.158.66:JTAC UPD RCVD FROM PCC, ack
928380025
2016-04-25 10:06:45.196517(27897)[EVENT]: 172.25.158.66:JTAC ADD SENT TO   PCS 928380025,
UP
```

e. If the PCE server logs indicate a problem with the PCE server, you can manually restart the PCE server pod by using the following command:

```
kubectl -n northstar rollout restart deployment pce-server-pod-name
```

If the logs do not contain the information that you are looking for, you can also run a variety of `show` commands on the PCE server to obtain the information. Just as with Junos OS syntax, you can enter `show ?` to see the `show` command options.

3. The PCE server performs the following actions after it receives the provisioning order:

   - The PCE server forwards the order to the Path Computation Client (PCC).
   - The PCE server sends an acknowledgment back to the PCS.

4. The LSP controller status changes to PCC_PENDING, indicating that the PCE server received the provisioning order and forwarded it to the PCC, but the PCC has not yet responded. If an LSP is stuck in the PCC_PENDING state, it suggests that the problem lies with the PCC. Check the configuration on the PCC. For information on how to check the configuration on the PCC, refer to your vendor's documentation for the PCC.

5. If the PCC receives the provisioning order successfully, it sends a response to the PCE server, which in turn forwards the response to the PCS. When the PCS receives this response, the PCS clears the LSP controller status completely. A clear status indicates that the PCS has provisioned the LSP and that no action is pending from the PCE server or PCC. The operational status for the tunnel then becomes the indicator for the condition of the tunnel. You can view the operational status in the **Op**

Status column on the Tunnel tab of the network information table. The following is a sample log entry that the PCS generates:

```
Apr 25 10:06:45.203909 user-PCS PCServer  [<-TopoServer] JTAC@10.0.0.102, LSP event=(0)CREATE
request_id=928380025 tunnel_id=9513 lsp_id=1 report_type=ACK
```

The Operational State of the LSP Is Down

IN THIS SECTION

- Problem | 51
- Solution | 51

Problem

The Path Computation Server (PCS) has sent a provisioning order for the label-switched path (LSP) to the Path Computation Element (PCE) server. The PCS has received an acknowledgment from the PCE server. In addition, the PCS clears the controller status. However, the LSP still is not up (is not operational).

Solution

Check whether the LSP's operational status in the **Op** Status column on the Tunnel tab of the network information table is marked as Down. The LSP's operational status can be Down if the live and planned utilizations are different from each other. If the operational status of the LSP is marked as Down, the Path Computation Client (PCC) cannot signal the LSP.

- Live utilization—The routers in the network use this utilization to signal a path. This utilization value is learned from the traffic engineering database through the BGP Monitoring Protocol (BMP).

  The following is a sample log entry of the Path Computation Server (PCS). In particular, note the reservable bandwidth (`reservable_bw`) entries that advertise the RSVP live utilization on the link:

NOTE: The logs that you view may differ from the logs in this document. The logs change based on the installed version of Paragon Automation.

```
Apr 25 10:10:11.475686 user-PCS PCServer  [<-TopoServer] LINK UPDATE:
ID=L10.105.107.1_10.105.107.2 status=UP nodeA=0110.0000.0105 nodeZ=0110.0000.0107
protocols=(260)ISIS2,MPLS
Apr 25 10:10:11.475690 user-PCS PCServer    [A->Z] ID=L10.105.107.1_10.105.107.2 IP
address=10.105.107.1 bw=10000000000 max_rsvp_bw=10000000000 te_metric=10 color=0
reservable_bw={9599699968 8599699456 7599699456 7599699456 7599699456 7599699456 7599699456
7099599360 }
Apr 25 10:10:11.475694 user-PCS PCServer    [Z->A] ID=L10.105.107.1_10.105.107.2 IP
address=10.105.107.2 bw=10000000000 max_rsvp_bw=10000000000 te_metric=10 color=0
reservable_bw={10000000000 10000000000 10000000000 8999999488 7899999232 7899999232
7899999232 7899999232 }
```

- Planned utilization—The controller uses this type of utilization for path computation. The controller computes this utilization value through the Path Computation Element Protocol (PCEP). The controller computes this value when the router advertises the LSP and communicates to the controller the bandwidth and path the LSP must use.

  Following is a sample log entry that the PCS generates. In particular, note the bandwidth (`bw`) and record route object (`RRO`) entries that advertise the RSVP planned utilization on the link:

```
Apr 25 10:06:45.208021 ns-PCS PCServer  [<-TopoServer] routing_key = ns_lsp_link_key
Apr 25 10:06:45.208034 ns-PCS PCServer  [<-TopoServer] JTAC@10.0.0.102, LSP event=(2)UPDATE
request_id=0 tunnel_id=9513 lsp_id=1 report_type=STATE_CHANGE
Apr 25 10:06:45.208039 ns-PCS PCServer   JTAC@10.0.0.102, lsp add/update event
lsp_state=ACTIVE admin_state=UP, delegated=true
Apr 25 10:06:45.208042 ns-PCS PCServer   from=10.0.0.102 to=10.0.0.104
Apr 25 10:06:45.208046 ns-PCS PCServer   primary path
Apr 25 10:06:45.208049 ns-PCS PCServer   association.group_id=128 association_type=1
Apr 25 10:06:45.208052 ns-PCS PCServer    priority=7/7 bw=100000 metric=30
Apr 25 10:06:45.208056 ns-PCS PCServer    admin group bits exclude=0 include_any=0
include_all=0
Apr 25 10:06:45.208059 ns-PCS PCServer    PCE initiated
Apr 25 10:06:45.208062 ns-PCS PCServer
ERO=0110.0000.0102--10.102.105.2--10.105.107.2--10.114.117.1
Apr 25 10:06:45.208065 ns-PCS PCServer
```

```
RRO=0110.0000.0102--10.102.105.2--10.105.107.2--10.114.117.1
Apr 25 10:06:45.208068 ns-PCS PCServer    samepath, state changed
```

The two utilization values can be different from each other. The difference might interfere with the successful computation or signaling of the path. For example:

- If the planned utilization is higher than the live utilization, a path computation issue could arise. From the perspective of the PCS, it might seem like no bandwidth is available for the new path. Therefore, the PCS cannot compute the path. However, sufficient bandwidth might be available to compute the path.
- If the planned utilization is lower than the live utilization, the PCC does not signal the path. This is because, from the perspective of the PCC, it might seem like no bandwidth is available to signal the new path.

To view utilization in the GUI, navigate to the Topology (**Network > Topology**) page. In the Link tab of the network information table, the **RSVP Live Util AZ** and **RSVP Live Util ZA** columns indicate the RSVP live utilization. The **RSVP Util AZ** and **RSVP Util ZA** columns indicate the RSVP planned utilization. The topology map reflects the live utilization from the PCC and the planned utilization that the Controller computes based on planned properties.

You can compare the entries in these columns to check whether any mismatch exists between the live and the planned utilizations. To solve any mismatch, you can try re-synchronizing the network model from the Pathfinder page (**Configuration > Network Settings > Pathfinder Settings > Advanced Settings**) in the GUI. See "Synchronize the Network Model" on page 64 for more information about re-synchronizing the network model.

LSPs Missing from the Network Information Table

IN THIS SECTION

- Problem | 54
- Solution | 54

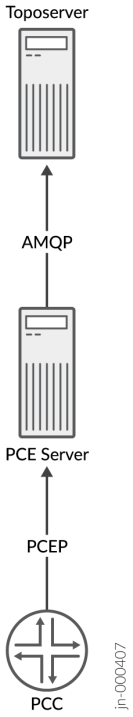
Problem

One or more label-switched paths (LSPs) are missing from the Tunnel tab in the network information table.

Solution

The topology map on the Topology page is populated from the data in the network information table. If one or more LSPs lack corresponding entries in the table, the paths cannot be highlighted in the topology map. To troubleshoot the problem, you must understand the flow of information from the Path Computation Client (PCC) to the Topology server (also called Toposerver), as illustrated in Figure 2 on page 54. The LSPs are added to the network information table only if this flow of information is successful. The flow begins with the configuration at the PCC. The PCC sends an **LSP Update** message to the Path Computation Element (PCE) server through a PCEP session. The PCE server sends this message to the Toposerver through an Advanced Message Queuing Protocol (AMQP) connection.

Figure 2: LSP Information Flow



To check these connections, do the following:

1. Using your server credentials, log in to the primary node of Paragon Automation.
2. From the primary node, use the following command to obtain the name of the pod in which the PCE server process runs:

```
kubectl -n northstar get pod | grep pce
```

3. Use the following command to log in to the PCE server pod:

```
kubectl -n northstar exec -it ns-pceserver-pod-name -c ns-pceserver -- bash
```

In the command, *ns-pceserver-pod-name* is the name of the pod that you obtained in "Step 2" on page 55.

4. Use the following command to view the logs associated with the PCE server:

```
kubectl -n northstar logs -f ns-pceserver-pod-name -c syslog
```

In the command, *ns-pceserver-pod-name* is the name of the pod that you obtained in "Step 2" on page 55. The log prints a message every 15 seconds when it detects that its connection with the PCE server has been lost or was never successfully established. In that case, you can re-synchronize the network model from the Pathfinder page (**Configuration > Network Settings > Pathfinder Settings > Advanced Settings**) in the GUI. See "Synchronize the Network Model" on page 64 for more information on re-synchronizing the network model.

In the following example log, the connection between the Toposerver and the PCE server is marked as Down:

NOTE: The logs that you view may differ from the logs in this document. The logs change based on the installed version of Paragon Automation.

```
Toposerver log:
Apr 22 16:21:35.016721 user-PCS TopoServer  Warning, did not receive the PCE beacon within 15
seconds, marking it as down. Last up: Fri Apr 22 16:21:05 2016
Apr 22 16:21:35.016901 user-PCS TopoServer  [->PCS] PCE Down: Warning, did not receive the
PCE beacon within 15 seconds, marking it as down. Last up: Fri Apr 22 16:21:05 2016
Apr 22 16:21:50.030592 user-PCS TopoServer  Warning, did not receive the PCE beacon within 15
seconds, marking it as down. Last up: Fri Apr 22 16:21:05 2016
```

```
Apr 22 16:21:50.031268 user-PCS TopoServer  [->PCS] PCE Down: Warning, did not receive the
PCE beacon within 15 seconds, marking it as down. Last up: Fri Apr 22 16:21:05 2016
```

5. Install the netstat utility in the `pcep_server` container by using the following command; ensure that your system has Internet access:

   NOTE: The netstat utility is not available by default.
   Run the following commands before you install netstat utility:

   ```
   kubectl -n northstar exec -it ns-pceserver-pod-name -c ns-pceserver -- bash
   ```

   ```
   apt-get update
   ```

   ```
   apt-get install net-tools
   ```

6. Use the following command to verify that the PCEP session between the PCC and the PCE server is established through port 4189:

   ```
   netstat -na | grep 4189
   ```

   Following is sample output:

   ```
   tcp        0      0 0.0.0.0:4189            0.0.0.0:*               LISTEN
   tcp        0      0 172.25.152.42:4189      172.25.155.50:59143     ESTABLISHED
   tcp        0      0 172.25.152.42:4189      172.25.155.48:65083     ESTABLISHED
   ```

7. Even if the session is established, it does not necessarily mean that any data was transferred. From the PCE server pod, use the following command to check whether the PCE server discovered any LSPs from the PCC through PCEP:

   ```
   pcep_cli
   ```

   ```
   show lsp all list
   ```

Following is sample output:

```
2016-04-22 17:09:39.696061(19661)[DEBUG]: pcc_lsp_table.begin:
2016-04-22 17:09:39.696101(19661)[DEBUG]: pcc-id:1033771436/172.25.158.61, state: 0

2016-04-22 17:09:39.696112(19661)[DEBUG]: START of LSP-NAME-TABLE
…
2016-04-22 17:09:39.705358(19661)[DEBUG]: Summary pcc_lsp_table:
2016-04-22 17:09:39.705366(19661)[DEBUG]:   Summary LSP name tabl:
2016-04-22 17:09:39.705375(19661)[DEBUG]:     client_id:1033771436/172.25.158.61, state:0,num
LSPs:13
2016-04-22 17:09:39.705388(19661)[DEBUG]:     client_id:1100880300/172.25.158.65, state:0,num
LSPs:6
2016-04-22 17:09:39.705399(19661)[DEBUG]:     client_id:1117657516/172.25.158.66, state:0,num
LSPs:23
2016-04-22 17:09:39.705410(19661)[DEBUG]:     client_id:1134434732/172.25.158.67, state:0,num
LSPs:4
2016-04-22 17:09:39.705420(19661)[DEBUG]:   Summary LSP id table:
2016-04-22 17:09:39.705429(19661)[DEBUG]:     client_id:1033771436/172.25.158.61, state:0,
num LSPs:13
2016-04-22 17:09:39.705440(19661)[DEBUG]:     client_id:1100880300/172.25.158.65, state:0,
num LSPs:6
2016-04-22 17:09:39.705451(19661)[DEBUG]:     client_id:1117657516/172.25.158.66, state:0,
num LSPs:23
2016-04-22 17:09:39.705461(19661)[DEBUG]:     client_id:1134434732/172.25.158.67, state:0,
num LSPs:4
```

In the far right column of the output, you can view the number of LSPs that the PCE server learned. If this number is 0, the PCE server did not receive any LSP information. In that case, check the configuration on the PCC. For information about how to check the configuration on the PCC, see your vendor's documentation for the PCC.

Topology Not Displayed in the GUI

IN THIS SECTION

Problem | 58

Solution | 58

Problem

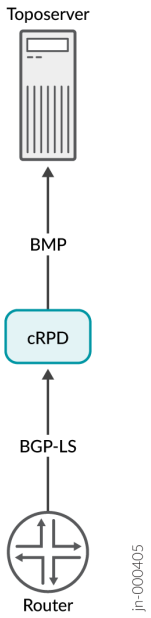
The Topology page (**Network > Topology**) does not display the topology.

Solution

The topology appears on the Topology page only if the topology server (Toposerver) receives information from the router.If the topology does not appear on the Topology page, it is likely that the flow of information from the router to the Toposerver was interrupted. To resolve the problem, you must identify where this flow was interrupted and prevent further interruption. Figure 3 on page 59 illustrates the flow of information from the router to the Toposerver.

The topology originates at the routers. For Paragon Pathfinder to receive the topology, you must establish a BGP-LS session from one of the routers in the network to the containerized routing protocol process (daemon) [cRPD]. Ensure that a BGP Monitoring Protocol (BMP) session is already established between cRPD and the Toposerver.

Figure 3: Topology Information Flow



To check these connections, do the following:

1. To verify that the Toposerver successfully established a BMP connection with cRPD, do the following:

   a. Using your server credentials, log in to the primary node of Paragon Automation.

   b. From the primary node, use the following command to obtain the name of the pod associated with the Toposerver process:

   ```
   kubectl -n northstar get pod | grep topo
   ```

   c. Use the following command to log in to the Toposerver pod:

   ```
   kubectl -n northstar exec -it ns-toposerver-pod-name -c ns-toposerver -- bash
   ```

   In the command, *ns-toposerver-pod-name* is the name of the pod that you obtained in "Step b" on page 59.

d. Use the following command to verify that the Toposerver successfully established a BMP connection with cRPD through port 1002:

```
netstat -na | grep :1002
```

The following example output indicates that the connection was successfully established:

```
tcp        0      0 172.16.16.1:55752           172.16.16.2:1002            ESTABLISHED
```

2. BMP is configured automatically on cRPD during the installation of Paragon Automation to enable the topology export. To confirm that the configuration is present on cRPD, do the following:

a. From the primary node, log in to cRPD by using the following command:

```
pf-crpd
```

b. Use the following command to confirm whether BMP is configured:

```
show configuration routing-options
```

If you see output similar to the following, it indicates that BMP is configured:

```
root@bmp-pod-name> show configuration routing-options
autonomous-system 64512
bmp {
    local-address local-ip-address;
    local-port port-number;
    connection-mode passive;
    monitor enable;
    station localhost {
        station-address ip-address;
     }
}
static {
     route 0.0.0.0/0 next-hop next-hop-ip-address;
}
```

If the BMP configuration is missing, configure BMP. See the Junos OS documentation for details.

3. Use the Junos OS `show` commands to confirm whether the BGP-LS session between cRPD and the router is ACTIVE. If the session is not ACTIVE, the router cannot send the topology information to cRPD.

4. From cRPD, use the following command to verify whether the lsdist.0 routing table has any entries:

```
show route table lsdist.0 hidden
```

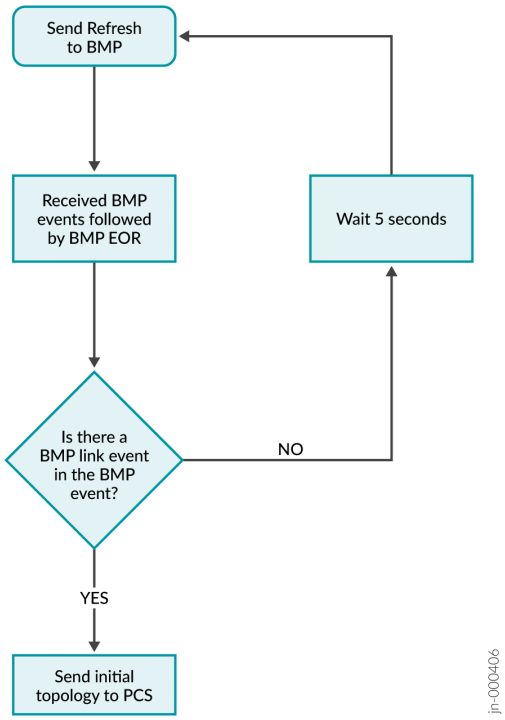
   If you see output similar to the following, it indicates that the lsdist.0 routing table has entries:

```
lsdist.0: 54 destinations, 54 routes (0 active, 0 holddown, 54 hidden)
```

   Ensure that at least one link exists in the lsdist.0 routing table. The Toposerver can create an initial topology only if it receives at least one BMP link event. A network that consists of a single node with no interior gateway protocol (IGP) adjacency with other nodes will not enable the Toposerver to create a topology. For example, in a lab environment, the network may not have IGP adjacency established with other nodes.

   Figure 3 on page 59 illustrates the Toposerver's logical process for creating the initial topology.

Figure 4: Logical Process for Initial Topology Creation



If the Toposerver cannot create an initial topology, it generates a log entry similar to the following example:

NOTE: The logs that you view may differ from the logs in this document. The logs change based on the installed version of Paragon Automation.

```
Dec 9 16:03:57.788514 fe-cluster-03 TopoServer Did not send the topology because no links
were found.
```

If the Toposerver cannot create an initial topology, you must re-synchronize the network model from the Pathfinder page (**Configuration > Network Settings > Pathfinder Settings > Advanced Settings**) in the GUI to obtain the topology. See "Synchronize the Network Model" on page 64 for details about re-synchronizing the network model.

Changes Not Reflected in the GUI

IN THIS SECTION

- Problem | 63
- Solution | 63
- Synchronize the Network Model | 64
- Reset the Network Model | 65

Problem

The changes that you made in the live network do not appear in the GUI.

Solution

Ideally, the network model in the Paragon Pathfinder database is in sync with the live network. However, the network model might become out of sync with the live network for various reasons. For example, when the network model audit has unresolved discrepancies, the node ISO network entity title (NET) address changes, the node ID changes, and so on. To solve the problem, you can either synchronize the network model or reset the network model from the Pathfinder page (**Configuration > Network Settings > Pathfinder Settings > Advanced Settings**):

NOTE: You can synchronize and reset the network model from the GUI only if your user role has the permissions to perform the Sync and Reset actions, respectively.

WARNING: Perform the Reset action only with the supervision of Juniper Networks Technical Assistance Center (JTAC). This action erases the network data model and the data provided through the Add or Modify actions in the network information table. Therefore, reserve this action for a lab setting (not a production setting).

- Synchronize the network model—Perform this action if you don't want to lose user planning data when you synchronize the network model with the live network. The Path Computation Server (PCS) still requests a topology synchronization, but the Toposerver does not delete the existing elements.
- Reset the network model—Perform this action only if the sync operation does not resolve the discrepancies in the network model and you want to start over from scratch with your topology.

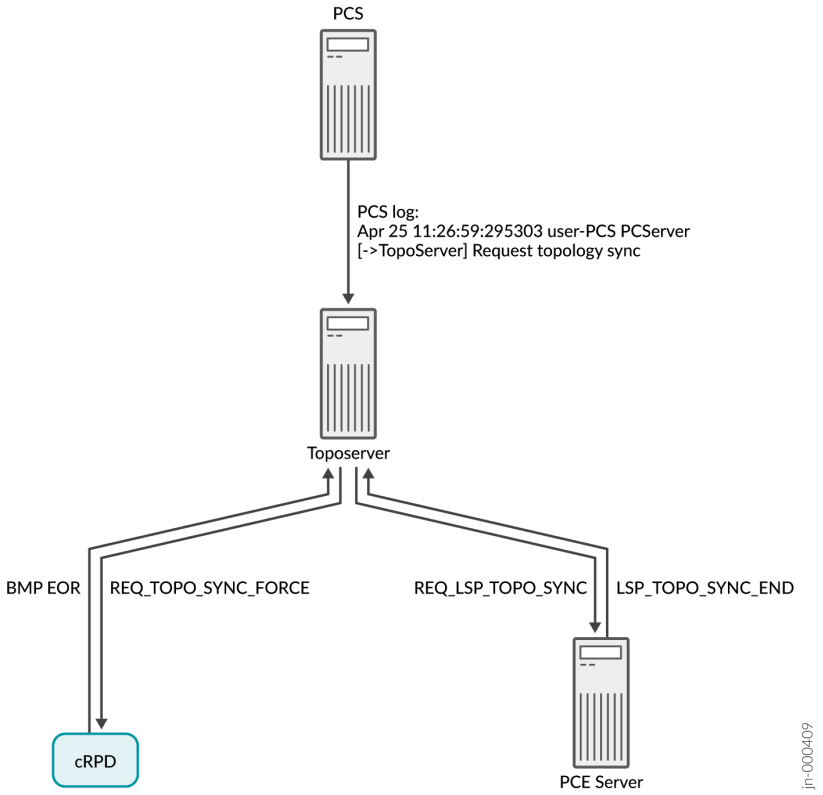
Synchronize the Network Model

When you synchronize the network model:

1. Information associated with the network model remains intact. This includes information such as the information associated with the nodes, links, label-switched paths (LSPs), interfaces, shared risk link groups (SRLG), and user-defined parameters. Nothing is purged from the database.
2. The network model is repopulated with live data learned from topology acquisition.

Figure 5 on page 65 illustrates the flow from the PCS to the containerized routing protocol process (daemon) [cRPD] and Path Computation Element (PCE) server. It also illustrates the updates coming back to the Toposerver.

Figure 5: Synchronization Request and Model Updates Using Sync Network Model



Reset the Network Model

If you execute the Reset Network Model operation, you will lose changes that you've made to the database. In a multi-user environment, one user might reset the network model without the knowledge of the other users.

When you reset the network model:

1. When you request a reset, the request goes from the PCS to the Toposerver.

   Following is a sample of the PCS log that indicates that a topology reset is requested:

NOTE: The logs that you view may differ from the logs in this document. The logs change based on the installed version of Paragon Automation.

```
Apr 25 10:54:50.385008 user-PCS PCServer  [->TopoServer] Request topology reset
```

2. Information associated with the network model (nodes, links, LSPs, interfaces, SRLGs, and user-defined parameters) is purged from the database.

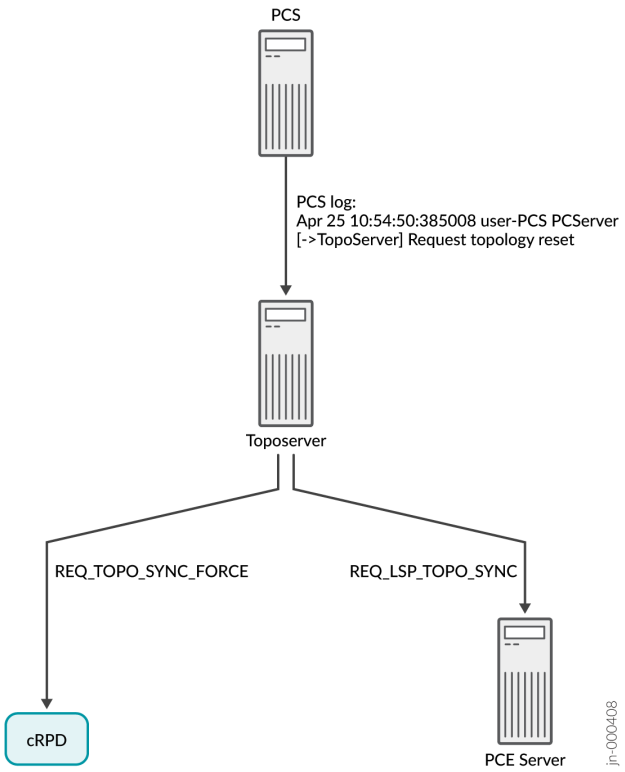
   NOTE: Device profiles are not purged from the database.

   The following sample of the Toposerver log indicates that the database elements are being removed:

```
Apr 25 10:54:50.386912 user-PCS TopoServer  Truncating pcs.links...
Apr 25 10:54:50.469722 user-PCS TopoServer  Truncating pcs.nodes...
Apr 25 10:54:50.517501 user-PCS TopoServer  Truncating pcs.lsps...
Apr 25 10:54:50.753705 user-PCS TopoServer  Truncating pcs.interfaces...
Apr 25 10:54:50.806737 user-PCS TopoServer  Truncating pcs.facilities...
```

3. The Toposerver then requests a synchronization with both the cRPD (to retrieve the topology nodes and links) and the PCE server (to retrieve the LSPs). In this way, the Toposerver relearns the topology, but any user updates are missing. Figure 6 on page 67 illustrates the flow from the topology reset request to the request for synchronization with the cRPD and the PCE Server.

Figure 6: Reset Model Request



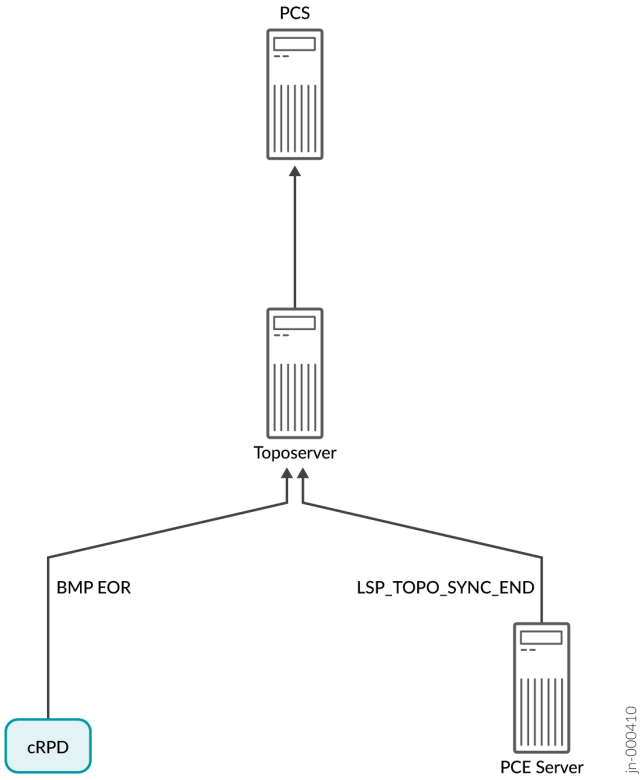
4. Upon receiving the synchronization requests, cRPD and the PCE server return topology updates that reflect the current live network.

   The following sample of the PCS log indicates that the topology updates are being added to the database:

```
Apr 25 10:54:52.237882 user-PCS PCServer  [<-TopoServer] Update Topology
Apr 25 10:54:52.237894 user-PCS PCServer  [<-TopoServer] Update Topology Persisted Nodes (0)
Apr 25 10:54:52.238957 user-PCS PCServer  [<-TopoServer] Update Topology Live Nodes (7)
Apr 25 10:54:52.242336 user-PCS PCServer  [<-TopoServer] Update Topology Persisted Links (0)
Apr 25 10:54:52.242372 user-PCS PCServer  [<-TopoServer] Update Topology live Links (10)
Apr 25 10:54:52.242556 user-PCS PCServer  [<-TopoServer] Update Topology Persisted Facilities
(1)
Apr 25 10:54:52.242674 user-PCS PCServer  [<-TopoServer] Update Topology Persisted LSPs (0)
Apr 25 10:54:52.279716 user-PCS PCServer  [<-TopoServer] Update Topology Live LSPs (47)
Apr 25 10:54:52.279765 user-PCS PCServer  [<-TopoServer] Update Topology Finished
```

Figure 7 on page 68 illustrates the return of topology updates from the cRPD and the PCE Server to the Toposerver and the PCS.

Figure 7: Model Updates by Using Reset Network Model



Topology Displayed in the GUI Is Incorrect

IN THIS SECTION

- Problem | 69
- Solution | 69

Problem

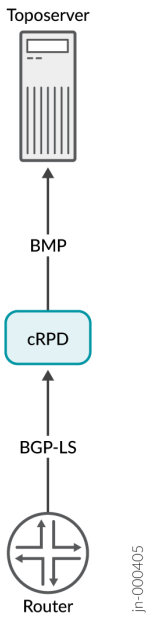
The topology displayed on the Topology page (**Network > Topology**) in the GUI is incorrect. An incorrect topology can pertain to a variety of problems such as a missing link, a link that is up but marked as Down in the topology, and so on.

Solution

The topology displayed on the Topology page is automatically populated based on the data from the topology server (Toposerver). The Toposerver correlates the unidirectional link (interface) information that it receives from the routers into bidirectional links by matching source and destination IPv4 and IPv6 link identifiers from BMP link events. The topology displayed on the Topology page can be incorrect if the router has advertised the links incorrectly or if the Toposerver has correlated the links incorrectly.

Figure 8 on page 69 illustrates the flow of information from the router to the Toposerver.

Figure 8: Topology Information Flow



To check whether the router has advertised the links correctly, do the following:

1. Use the following command to log in to the router through SSH:

```
ssh username@router-ip-address
```

2. Use the following command to view the lsdist.0 routing table for the link:

```
show route table lsdist.0 | match link-ip-address
```

The router advertises each unidirectional link as two separate entries, one for each endpoint. The local IP address in the first entry must match the remote IP address in the second entry (and vice versa).

If the advertisement doesn't exist, the link will not appear in the topology. If the advertisement exists but is incorrect, the Toposerver cannot correlate the links correctly. To solve this problem, contact Juniper Networks Technical Assistance Center (JTAC).

If the router has advertised the links correctly but the topology appears incorrectly, it is likely that the Toposerver has correlated the links incorrectly. To solve this problem, trigger a re-advertisement of all the links by re-synchronizing the topology from the Pathfinder page (**Configuration > Network Settings > Pathfinder Settings > Advanced Settings > Sync Network Model**). See "Synchronize the Network Model" on page 64 for details about this action.

PCS Out of Sync with Toposerver

IN THIS SECTION

- Problem | 70
- Solution | 71

Problem

The Path Computation Server (PCS) is out of sync with the topology server (Toposerver); therefore, these two components have a different view of the network.

Solution

The PCS can be out of sync with the Toposerver for a variety of reasons. For example, the messaging bus that these components use for communication might have a problem, the server might be busy or overloaded, and so on. Therefore, each component has a different view of the network.

NOTE: The following example provides information about solving the problem when the PCS and Toposerver disagree on the state of the label-switched paths (LSPs). You can also perform these steps to resolve similar synchronization issues, such as with the nodes.

To restore synchronization, you must delete and re-create the Path Computation Element (PCE) server pod by using the following procedure:

CAUTION: Be aware that this procedure:

- Kills the Path Computation Element Protocol (PCEP) sessions for all Path Computation Clients (PCCs), not just the one with which there is a problem.
- Results in the loss of all user data, which then needs to be repopulated.
- Has an impact on the production system due to the re-synchronization.

1. Use your server credentials to log in to the primary node of Paragon Automation.
2. From the primary node, delete the deployment of the PCE server so that the PCE server pod is removed.

```
kubectl delete -f /etc/kubernetes/po/pcserver/kube.cfo.yml
```

3. Wait for 10 seconds to allow the PCC to remove all lingering LSPs. Then, use the following command to re-create the PCE server pod:

```
kubectl create -f /etc/kubernetes/po/pcserver/kube.cfo.yml
```

4. Use the following command to re-create the Toposerver pod:

```
kubectl create -f /etc/kubernetes/po/toposerver/kube.cfo.yml
```

NOTE: Alternatively, you can contact the Juniper Networks Technical Assistance Center (JTAC) for help with a Reset action that erases all data. This action is for a lab environment (not production). We strongly recommend that you perform the Reset action only under the supervision of JTAC.

The Reset action consists of restarting the Toposerver by resetting the network model from the GUI (**Configuration > Network Settings > Pathfinder Settings> Advanced Settings > Reset Network Model**). The Reset action erases the data in the network model and the data provided through the Add or Modify actions in the network information table. Therefore, the Reset action is typically appropriate for a lab rather than a production environment. See "Reset the Network Model" on page 65 for more information about this operation.

The synchronization is now restored.

PCCs Are Not PCEP-Enabled

IN THIS SECTION

- Problem | 72
- Solution | 72

Problem

Path Computation Clients (PCCs) in the topology are not Path Computation Element Protocol (PCEP)-enabled.

Solution

The topology server (Toposerver) associates the PCEP sessions with the PCCs in the topology to mark the PCCs as PCEP-enabled. Ideally, the Toposerver recognizes the IP address of the PCC that establishes the PCEP session. The association is not successful if the Toposerver does not recognize that IP address automatically from the traffic engineering database. For example, the Toposerver will not receive the IP

address from the traffic engineering database if the PCC establishes a PCEP session through the management IP address.

When the PCC successfully establishes a PCEP session, the PCC sends a **PCC_SYNC_COMPLETE** message to the Toposerver. This message indicates to the controller that synchronization is complete. Following is a sample log entry that the Toposerver generates. The log entry displays the **PCC_SYNC_COMPLETE** message and the PCEP IP address, which the controller may or may not recognize:

> **NOTE**: The logs that you view may differ from the logs in this document. The logs change based on the installed version of Paragon Automation.

```
Dec  9 17:12:11.610225 fe-cluster-03 TopoServer  NSTopo::updateNode (PCCNodeEvent) ip:
172.25.155.26 pcc_ip: 172.25.155.26 evt_type: PCC_SYNC_COMPLETE
Dec  9 17:12:11.610230 fe-cluster-03 TopoServer  Adding PCEP flag to pcep_ip: 172.25.155.26
node_id: 0880.0000.0026 router_ID: 10.0.0.26 protocols: 4
Dec  9 17:12:11.610232 fe-cluster-03 TopoServer  Setting live pcep_ip: 172.25.155.26 for
router_ID: 88.0.0.26
```

To solve the problem, you must:

- Manually enter the unrecognized IP address of the device on the Add Devices page **(Configuration > Devices >+**).
- Ensure that at least one LSP originates from the PCC. If an LSP originates from the PCC, the Toposerver can associate the PCEP session with that PCC in the traffic engineering database.

After the problem is solved, the Toposerver adds the PCEP IP address to the PCC attributes. Following is a sample PCS log. This log indicates that the Toposerver has added the PCEP IP address to the PCC attributes:

```
Dec  9 17:12:11.611392 fe-cluster-03 PCServer  [<-TopoServer] routing_key = ns_node_update_key
Dec  9 17:12:11.611394 fe-cluster-03 PCServer  [<-TopoServer] NODE UPDATE(Live):
ID=0880.0000.0026 protocols=(20)ISIS2,PCEP status=UNKNOWN hostname=skynet_26 router_ID=88.0.0.26
iso=0880.0000.0026 isis_area=490001 AS=41 mgmt_ip=172.25.155.26 source=NTAD Hostname=skynet_26
pcep_ip=172.25.155.26
```

PART 4

Troubleshoot Paragon Insights

Debug No-Data | 75

Diagnose Device Reachability Issues | 77

Diagnose Ingest Connectivity Issues | 78

Self-Test Basic Functionality | 79

Identify Kubernetes Cluster Health Issues | 81

Troubleshoot Common Paragon Insights Issues | 82

Debug No-Data

IN THIS SECTION

- Problem | 75
- Solution | 75

Problem

You are unable to determine why a device or rule displays a “no-data” status on the **Configuration** > **Devices** page and the **Configuration** > **Rules** page. You are unable to debug this error.

Solution

You can use the debug no-data tool to determine why a device or rule displays a “no-data” status. The tool goes through a step-by-step process to determine at which stage incoming data gets dropped or blocked.

The step-by-step process is as follows:

1. Verifies that all common and device group-related services are up and running.
2. Tests connectivity to the device by using ping and SSH.
3. Verifies that configured ingest sessions are established.

   This verification is limited to OpenConfig, iAgent, and SNMP-based ingests.
4. Verifies whether the ingest is receiving any raw data from the devices.
5. (Within rules) Verifies that the fields process sensor-related information and that this information is populated in the database.
6. (Within rules) Verifies that the trigger settings work as intended and that trigger status information is populated in the database.
7. Checks for API timeouts that might be affecting the GUI.

To debug the no-data status of a device or rule:

1. Access the debug no-data tool by navigating to either of the following pages:

   - **Monitoring** > **Network Health**

     On this page, click any tile that shows "no-data."

   - **Configuration** > **Data Ingest** > **Diagnostics**

   The **Diagnostics** page appears.

2. Click **No Data** to view the **No-Data Settings** tabbed page.

3. From the **No-Data Settings** tabbed page, configure the following settings:

   a. Select a device group from the **Device Group** drop-down list.

   b. Select a device from the **Device** drop-down list.

   c. Select a topic from the **Topic** drop-down list.

   d. Select one or more rules from the **Rule(s)** drop-down list.

4. Click **Test** to run the debug no-data test.

   The test results appear after a few minutes.

   The color coding for the test results is as follows:

   - Green status = pass
   - Yellow status = error (unable to test)
   - Red status = fail (test failed)
   - Yellow or red status includes a message with details about the issue.

   High-level information about the results of the test is also displayed.

RELATED DOCUMENTATION

Use the No-Data Tool

Diagnose Device Reachability Issues

IN THIS SECTION

- Problem | 77
- Solution | 77

Problem

You cannot easily determine whether the devices you added to your network are up and reachable.

Solution

You can use the device reachability tool to diagnose device reachability issues during onboarding and any time after the onboarding process. The device reachability tool can verify connectivity to a device by using ping and SSH.

To run the device reachability test:

1. Navigate to the **Configuration** > **Data Ingest** > **Diagnostics** page.
2. On the **Diagnostics** page, click **Reachability** to view the **Reachability Test** tabbed page.
3. From the **Reachability Test** tabbed page, configure the following settings:
   a. Select a device group from the **Device Group** drop-down list.
   b. Select a device from the **Device** drop-down list.
4. Click **Test** to run the device reachability test.

   The test results of the ping test and the SSH test appear after a few minutes.

   The color coding for the test results is as follows:

   - Green status = pass
   - Yellow status = error (unable to test)

- Red status = fail (test failed)
- Yellow or red status includes a message with details about the issue.

High-level information about the results of the test is also displayed.

RELATED DOCUMENTATION

Use the Reachability Test

Diagnose Ingest Connectivity Issues

IN THIS SECTION

Problem | 78
Solution | 78

Problem

You cannot identify ingest connectivity with a network device.

Solution

The ingest connectivity tool can verify ingest methods such as OpenConfig, iAgent, and SNMP, where Paragon Insights initiates the connection with the device. The tool helps you identify a missing configuration on the network device. The tool also helps you choose appropriate playbooks and rules (that use sensors) that are compatible with the supported ingest methods.

To run the ingest connectivity test:

1. Navigate to the **Configuration** > **Data Ingest** > **Diagnostics** page.
2. On the **Diagnostics** page, click **Ingest** to view the **Ingest Test Settings** tabbed page.

3. From the **Ingest Test Settings** tabbed page, configure the following settings:

   a. Select one or more sensor types from the **Sensor Type(s)** drop-down list.

   b. Select a device group from the **Device Group** drop-down list.

   c. Select a device from the **Device** drop-down list.

4. Click **Test** to run the ingest connectivity test.

   The test results appear after a few minutes.

   The color coding for the test results is as follows:

   - Green status = pass
   - Yellow status = error (unable to test)
   - Red status = fail (test failed)
   - Yellow or red status includes a message with details about the issue.

   High-level information about the results of the test is also displayed.

RELATED DOCUMENTATION

Use the Ingest Test Tool

Self-Test Basic Functionality

IN THIS SECTION

Problem | 80
Solution | 80

Problem

You have difficulty diagnosing problems after initial setup and validating basic functionality of Paragon Insights.

Solution

After you set up the basic functionality in Paragon Insights, it can be challenging to diagnose problems related to configuring devices, adding devices, and applying playbooks. When an issue occurs, there are many areas that you must investigate. The self-test tool essentially acts as a fully working setup, running entirely within the Paragon Insights system. When testing is complete, the tool provides a report.

The self-test tool:

- Provides a simulated device connected to Paragon Insights, so you can demo Paragon Insights without adding a real device or applying playbooks.
- Auto-configures a simulated device connected to Paragon Insights, thereby eliminating the complexity of adding devices, applying playbooks, and so on.

To run the self-test tool:

1. Navigate to the **Configuration** > **Data Ingest** > **Diagnostics** page.
2. On the **Diagnostics** page, click **Application** to view the **Self Test Settings** tabbed page.
3. From the **Self Test Settings** tabbed page, configure the following self-test settings:

   a. Select one or more sensor types from the **Sensor Type(s)** drop-down list.

   b. (Optional) Click the **Retain Test Topology** toggle to turn it on or off.

      - If you turn the **Retain Test Topology** toggle on (default), the self-test device and device groups are retained in the **Device** and **Device Group Configuration** pages, respectively.
      - If you turn the **Retain Test Topology** toggle off, the self-test resources are auto-deleted when the test is completed.

4. Click **Test** to run the self-test.

   The test results appear after a few minutes.

   The color coding for the test results is as follows:

   - Green status = pass

- Yellow status = error (unable to test)
- Red status = fail (test failed)
- Yellow or red status includes a message with details about the issue.

High-level information about the results of the test is also displayed.

RELATED DOCUMENTATION

Use the Self Test Tool

Identify Kubernetes Cluster Health Issues

IN THIS SECTION

Problem | 81
Solution | 81

Problem

You are unable to identify issues related to Kubernetes cluster health.

Solution

The third-party kube-state-metrics monitoring service generates metrics based on the current state of Kubernetes clusters. The kube-state-metrics service is a beta-only feature that runs as a cluster service and is installed automatically when you install Paragon Automation. Once this service is installed, you can enable this service to generate, monitor, and expose metrics of various objects within a Kubernetes cluster.

To enable kube-state-metrics from the Paragon Automation UI:

1. Create an unmanaged device.

The unmanaged device represents the cluster. The device hostname must be the kube-state-metrics service IP. For example, the hostname could be `kube-state-metrics.healthbot.svc.cluster.local`.

2. Add the device to a device group.
3. Create rules.
4. Apply playbooks to the device group.

After you have enabled kube-state-metrics, you can view metrics on pods, DaemonSets, deployments, persistent volume, endpoints, ingress, job, lease, and configmap objects that are part of a Kubernetes cluster. For more information, see *Understand kube-state-metrics Service*.

To view metrics:

1. Navigate to the **Monitoring** > **Network Health** page.
2. On the **Network Health** page, click the **Device Group** entity type, and select the new unmanaged device from the **Devices** drop-down list.

A tile view and table view of the results are displayed after a few minutes.

RELATED DOCUMENTATION

Understand kube-state-metrics Service

Troubleshoot Common Paragon Insights Issues

IN THIS SECTION

- Paragon Insights Is Not Listed as a Component of the Paragon Automation Icon on the GUI | **83**
- Cannot View Paragon Insights-Related GUI Pages | **83**
- License Key Format Is Invalid | **83**
- System Log Messages Appear Twice | **84**
- Field Data Queried at 60s When Trigger Frequency Is Not Set | **85**
- Unable to Recover Data after TSDB Node Fails | **85**

These topics provide instructions for troubleshooting common issues with Paragon Insights.

Paragon Insights Is Not Listed as a Component of the Paragon Automation Icon on the GUI

Problem

Paragon Insights is not listed as a component in the Paragon Automation icon in the GUI.

Solution

The Paragon Automation icon in the top-left corner of the GUI is updated depending on the license that you add. When you log in to the Paragon Automation GUI for the first time, the Paragon Automation icon appears without the names of the components below it. The GUI displays the name of a component only after you add a license for that component. For example, after you add a Paragon Insights license, the name **Insights** appears below the Paragon Automation icon.

For more information, see *Paragon Insights Licensing Overview*.

Cannot View Paragon Insights-Related GUI Pages

Problem

You are unable to view Paragon Insights-related GUI pages after logging in to the Paragon Automation GUI for the first time.

Solution

When you log in to the Paragon Automation GUI for the first time, the **Dashboard** page appears. You must navigate to **Administration** > **License Management** to add a Paragon Insights license. After you successfully add a license, you can see the Paragon Insights-related GUI pages. However, the availability to use features in Paragon Insights is based on the license that you purchased.

For more information, see *Paragon Insights Licensing Overview*.

License Key Format Is Invalid

Problem

Successfully added a Paragon Insights license, but the license key format is invalid.

Solution

When you navigate to the **License Management** page, a warning message may appear. This message asks you to upgrade to the new license key format. This message appears if you added a Paragon Insights license but the license key format is incompatible with the Paragon Automation software version that you use. You must generate a new license key. To generate a new license key in the compatible format:

1. Log in to the Juniper Agile Licensing Portal.
2. Revoke the existing license.

   A new license key is generated.

   > **NOTE**: After you download the newly generated license key file, you must remove eight lines starting from `Issue Date` through `License Key` from the file.

3. Activate the new license key that is compatible with the software version you're running. Select the software version during the activation process.

   For more information, see the FAQ section in theJuniper Agile Licensing Portal.

4. Add the new license key to the **License Management** page.

   See *Add a License*.

You now have access to the Paragon Insights-related pages on the Paragon Automation GUI. However, the availability of features depends on the license that you purchased. See Paragon Insights Licensing.

System Log Messages Appear Twice

Problem

After configuring a system log (syslog) source, system log messages appear twice.

Solution

When you configure a system log source from the **Configuration** > **Devices** page, you must ensure that you configure either the IP address or the hostname of the system log source, but not both. If you configure both, system log messages appear twice.

For more information, see *Edit Devices*.

Field Data Queried at 60s When Trigger Frequency Is Not Set

Problem

Field data is queried and evaluated every 60s even when trigger frequency is not set.

Solution

Setting a trigger frequency when you create a rule is optional. The sensor frequency is applied if you do not set a trigger frequency. Navigate to **Configuration** > **Rules** to add a trigger. For more information, see *Set Trigger Frequency*.

When you enter a trigger frequency, you can enter it as a multiple or as an offset of the sensor frequency. In this scenario, since the sensor frequency is *60s* (default value) and the trigger frequency is *1o* (default value), the trigger frequency is set to *60s* (1*60s).

Unable to Recover Data after TSDB Node Fails

Problem

You are unable to recover data after the TSDB node goes down.

Solution

When the TSDB node fails, you cannot recover the data if the replication factor is set to one. If the replication factor is set to one, only one copy of data is maintained within a TSDB group. You must replace the failed node with a new node and adjust the replication factor to match the number of new nodes. The last 1GB of data directed to the failed node will buffer continuously until the TSDB node is replaced or recovered.

You can manage TSDB settings from the **Configuration** > **Insights Settings** > **TSDB Settings** page. For more information, see *Configure TSDB Settings*.

PART 5

Troubleshoot Paragon Planner

Unable to Connect to the Paragon Planner Server | 87

Links Missing in the Topology Map | 88

Unable to View Any Archived Networks | 90

Unable to Connect to the Paragon Planner Server

IN THIS SECTION

- Problem | 87
- Solution | 87

Problem

You cannot connect to the Paragon Planner server from the Paragon Planner desktop application.

Solution

If you cannot connect to the Paragon Planner server, it is likely that your network firewall prevents access to TCP port 7000. TCP port 7000 must be open to enable connectivity to the IP address of the Paragon Automation web GUI.

To solve this problem, do the following:

1. Use TELNET to connect to port 7000 on the Paragon Planner server:

   ```
   telnet web-ui-ip-address 7000
   ```

2. If you cannot connect to port 7000, you can use the following sample command to set up SSH tunneling:

   ```
   ssh -L 8443: web-ui-ip-address: 443 -L 7000: web-ui-ip-address: 7000 root@jump-host-ip-address
   ```

   The following tasks are applicable only if SSH tunneling is set up:

3. Log in to the Paragon Automation GUI with your credentials.

4. On the left navigation menu, select **Planning > Paragon Planner Desktop**.

The Paragon Planner Desktop page appears.

5. Enter the memory to be allocated and click **OK**. The default value is 512 MB.
6. Click **Save File** when prompted.

   A Java Network Launch Protocol (JNLP) file is downloaded to your computer.
7. In the JNLP file, replace **IPaddress** in the first argument with **localhost**, as shown in the following example:

```
<argument>localhost</argument>
```

8. Save the JNLP file.
9. Double-click the JNLP file to launch the Paragon Planner desktop application.

   The login page for the Paragon Planner desktop application appears.

If the login page does not appear, contact the Juniper Networks Technical Assistance Center (JTAC).

Links Missing in the Topology Map

IN THIS SECTION

- Problem | 88
- Solution | 89

Problem

The topology map in the Paragon Planner desktop application GUI does not display any links for an archived network.

Solution

If the topology map does not display any links for an archived network, there can be two possible reasons: Either the backbone link (**bblink.***) file is missing from the system, or the file is present in the system but is empty.

The **bblink*** file contains the location, quantity, vendor, and attributes of the links found in the network. To check the status of the **bblink.*** file in the system, do the following:

1. Use your server credentials to log in to the primary node of Paragon Automation.
2. From the primary node, use the following command to obtain the name of the pod associated with the scheduler process (**dcscheduler**):

   ```
   root@primary-node:~# kubectl get pod -n northstar|grep dcscheduler
   ```

3. Use the following command to log in to the dcscheduler pod:

   ```
   root@primary-node:~# kubectl exec -it dc-scheduler-pod-name -c dcscheduler -n northstar --
   bash
   ```

   In the command,*dc-scheduler-pod-name*is the name of the pod that you obtained in "Step 2" on page 89.

4. Navigate to the directory (**/opt/northstar/data/archives/...**) that contains the files corresponding to the archived network.

   The output displays the list of files present in the directory.

   Following is an example output that displays the size of the **bblink.*** file:

   ```
   total 360
   drwxr-xr-x 1 root root     28 Aug 30 13:15 ./
   drwxr-xr-x 1 root root      1 Aug 30 13:15 ../
   drwxr-xr-x 1 root root      1 Aug 30 13:15 Log/
   drwxr-xr-x 1 root root     10 Aug 30 13:15 Report/
   -rw-r--r-- 1 root root    624 Aug 30 13:15 aclist.x
   -rw-r--r-- 1 root root    572 Aug 30 13:15 atconfig.x
   -rw-r--r-- 1 root root  24849 Aug 30 13:15 bblink.new
   ```

5. From the list of files displayed, check whether the **bblink.*** file is present and that the file size is greater than 0.

If the file is not present, or if the file is present but the file size is 0, contact the Juniper Networks Technical Assistance Center (JTAC).

Unable to View Any Archived Networks

IN THIS SECTION

- Problem | 90
- Solution | 90

Problem

You cannot view any existing archived networks on the Archives tab (**Network > Browser > Archives**) in the Paragon Planner desktop application GUI.

Solution

It is likely that the **/opt/northstar/data** directory has no space available or has less space available than is required to store an archived network.

1. Use your server credentials to log in to the primary node of Paragon Automation.
2. From the primary node, use the following command to check the capacity of the **/opt/northstar/data** directory:

```
root@primary-node:~# kubectl get pvc -n northstar
```

Following is an example output that displays the capacity of the **/opt/northstar/data** directory:

```
NAME      STATUS   VOLUME                                     CAPACITY   ACCESS MODES
STORAGECLASS   AGE
```

```
ns-data  Bound    pvc-f24d9e79-5bbe-4f12-ace3-aa4b3fd420b9   6Gi        RWX            rook-
cephfs    48d
```

3. (Optional) From the primary node, use the following command to check the capacity and usage of the Ceph storage space:

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

```
[root@rook-ceph-tools-]# ceph status
```

Following is an example output:

```
data:
 volumes:  1/1 healthy
 pools:    11 pools, 177 pgs
 objects:  14.20k objects, 14GiB
 usage:    29GiB used, 71 GiB/100 GiB avail
 pgs:      14198/42594 objects degraded(33.333%)
           120 active+undersized+degraded
           57 active+undersized
```

This information is useful to understand the amount of storage allocation that you can increase for the **/opt/northstar/data** directory.

4. Use the following command to obtain the name of the pod associated with the scheduler process (that is, the **dcscheduler** pod):

```
root@primary-node:~# kubectl get pod -n northstar|grep dcscheduler
```

5. Use the following command to log in to the dcscheduler pod:

```
root@primary-node:~# kubectl exec -it dc-scheduler-pod-name -c dcscheduler -n northstar --
bash
```

In the command, *dc-scheduler-pod-name* is the name of the pod that you obtained in "Step 4" on page 91.

6. From the dcscheduler pod, use the following command to check the amount of free space available in the **/opt/northstar/data** directory:

```
root@dc-scheduler-pod-name:# df -h /opt/northstar/data
```

Following is an example output:

```
Filesystem       Size  Used   Avail   Use%  Mounted on
Filesystem name  58G   14G    45G     24%   /opt/northstar/data
```

If no free space is available in the directory, the **Use%** column in the output displays **0%**.

7. If the percentage of space used is closer to the capacity of the directory, you can use one of the the following commands to make space in the directory:

- To remove all directories created more than 24 hours ago (the recent directories might be in use):

```
root@dc-scheduler-pod-name:# ls -lrt
```

- To remove a specific directory:

```
root@dc-scheduler-pod-name:# \rm -r directory-name
```

You can now check whether you can view any archived networks in the GUI. If you still cannot view any archived networks, contact the Juniper Networks Technical Assistance Center (JTAC).