JUNIPER NETWORKS | Engineering Simplicity

Paragon Active Assurance Installation Guide

RELEASE
4.3

Table of Contents

Introduction

Paragon Active Assurance Control Center can either be hosted by Juniper Networks or be installed at the customer site. This document describes how to perform an on-site installation of Control Center, to be operated and maintained by the customer's own staff.

Control Center is delivered as an application that is installed on an existing Ubuntu OS. (For precise OS requirements, see the section "Operating System Requirements" on page 5 .)

Prerequisites and Preparations

IN THIS SECTION

Hardware Requirements

- Control Center in production handling up to 2000 Test Agents and up to 20,000 active streams:
  - 8 vCPUs

  - 40 GB RAM

  - 2000 IOPS

  - 1.5 TB disk space

- Control Center in production handling up to 8000 Test Agents and up to 100,000 active streams:
  - 36 vCPUs

  - 72 GB RAM

- 15,000 IOPS

- 7.5 TB disk space

- Lab trials of a Control Center with up to 20 Test Agents and up to 1000 active streams:

  - 2 vCPUs

  - 8 GB RAM

  - 100 IOPS

  - 30 GB disk space

Deployments larger than the limits indicated for each case will require multiple Control Centers.

Disk Usage and Data Retention Periods (RRDs)

Below is some data on disk space taken up when testing various services. The total disk space required for a service is calculated by multiplying with the number of streams.

Service tested	Disk usage, standard resolution	Disk usage, high resolution
TWAMP	3.6 MB	180 MB
Bidirectional UDP	2 x 1.5 MB = 3.0 MB	2 x 110 MB = 220 MB

The measurement data resolutions "standard" and "high" are defined according to the tables that follow. As detailed below, each is composed of a set of progressively lower resolutions for progressively longer time spans. The total number of data points is that stored in the time series database for each metric from a stream for a time period of either one year (standard resolution) or five years (high resolution). When data points become older than that, they are purged from the database.

Standard resolution

Data points are stored for one year.

The total number of data points is 13,566 (the sum of all entries in the **Number of data points** column).

This is the default resolution in Paragon Active Assurance.

Time span	Data point resolution	Number of data points	Percentage of data points

Last 12 hours	10 s	4,320	31.8%
Last 2 days	1 min	2,880	21.2%
Last week	5 min	2,016	14.9%
Last month	20 min	2,160	15.9%
Last year	4 h	2,190	16.1%

High resolution

Data points are stored for five years.

The total number of data points is 963,385.

Time span	Data point resolution	Number of data points	Percentage of data points
Last 90 days	10 s	777,600	80.7%
Last 1.5 years	5 min	157,680	16.4%
Last 3 years	1 h	26,280	2.7%
Last 5 years	1 day = 24 h	1,825	0.2%

Disk Usage and Data Retention Periods (TimescaleDB)

The table below shows the number of data points and retention periods for TimescaleDB rollups used for monitor data. Data for tests are not rolled up, nor are they automatically deleted over time.

Granularity	Default retention period	Number of data points
10 s (raw)	3 days	25,920
1 min	7 days	10,080
5 min	30 days	8,640

(Continued)

Granularity	Default retention period	Number of data points
30 min	180 days	8,640
1 h	365 days	8,760

Required Communication Ports

When configuring a firewall, traffic on the following ports needs to be allowed to and from Control Center.

- Inbound:
  - TCP port 443 (HTTPS): Web interface

  - TCP port 80 (HTTP): Web interface (used by Speedtest, redirects other URLs to HTTPS)

  - TCP port 6000 (default): Encrypted OpenVPN connection for Test Agent Appliances

  - TCP port 6800: Encrypted WebSocket connection for Test Agent Applications

- Outbound:
  - TCP port 25 (SMTP): Mail delivery

  - UDP port 162 (SNMP): Sending SNMP traps for alarms

  - UDP port 123 (NTP): Time synchronization

Advanced: As mentioned above, the communication between the Test Agent Appliance and Control Center is done over an OpenVPN tunnel on port 6000. If you want to firewall this OpenVPN connection running on the `tun0` tunnel interface, you must allow these ports:

- Inbound:
  - TCP port 4334: General interaction with Test Agents

  - HTTP port 80: Test Agent firmware update requests

- Outbound:
  - TCP port 4334: General interaction with Test Agents

Operating System Requirements

- Ubuntu Server 22.04 LTS

A major release of Control Center will be "locked" to a version of the base OS, so a patch release of Control Center will not support a newer base OS than the original version.

Installing Required OS and Software

> **NOTE**: Please note that this only describes a "fresh install". For upgrades, please refer to the *Upgrade Guide*.

1. **Install a clean Ubuntu 22.04 server.**

   - The system user name does not matter, *except* that the name "netrounds" is not allowed since PostgreSQL creates a user with that name (as described in).

   - Install only standard components (do not change the default selection).

   - The following disk partitioning is recommended, especially for snapshot backups (but it is up to you as a user to decide):

     - Recommended partitioning for lab setup:

       - `/`: Whole disk, ext4.

     - Recommended partitioning for production setup:

       - `/`: 10% of disk space, ext4.

       - `/var`: 10% of disk space, ext4.

       - `/var/lib/netrounds/rrd` (this includes TimescaleDB if you make use of that technology): 80% of disk space, ext4.

     - No encryption

   - Set the time zone to UTC, for example as follows:

     ```
     sudo timedatectl set-timezone Etc/UTC
     ```

- Set all locales to `en_US.UTF-8`.

  - One way to do this is to manually edit the file `/etc/default/locale`. Example:

    ```
    LANG=en_US.UTF-8
    LC_ALL=en_US.UTF-8
    LANGUAGE=en_US.UTF-8
    ```

  - Make sure the following line is NOT commented out in the file `/etc/locale.gen`:

    ```
    en_US.UTF-8 UTF-8
    ```

  - Regenerate the locale files to make sure selected language is available:

    ```
    sudo apt-get install locales
    sudo locale-gen
    ```

2. **Install NTP:**

   - First disable `timedatectl`:

     ```
     sudo timedatectl set-ntp no
     ```

   - Disable `systemd-timesyncd`.

     To disable the `systemd-timesyncd` service:

     a. Run the following command to check whether the `systemd-timesyncd` service (minimalistic service to synchronize local time with NTP servers) is running or not.

     ```
     systemctl status systemd-timesyncd
     ```
     If the service is activated, proceed to Step b.

     If the response is *Failed to disable unit: Unit file systemd-timesyncd.service does not exist*, then proceed with NTP installation.

     b. Run the following commands:

     ```
     systemctl disable systemd-timesyncd
     ```

     ```
     systemctl stop systemd-timesyncd
     ```

- Now you can run the NTP installation:

```
sudo apt-get update
sudo apt-get install ntp
```

- Make sure that the configured NTP servers are reachable:

```
ntpq -np
```

The "reach" value should normally be "all ones" expressed in octal.

(In the output, the "reach" value for the NTP servers is an octal value indicating the outcome of the last eight NTP transactions. If all eight were successful, the value will be octal 377 [= binary 0b11111111]. However, when you have just installed NTP, it is likely that fewer than eight NTP transactions have occurred, so that the value will be smaller: one of 1, 3, 7, 17, 37, 77, or 177 if all transactions were successful.)

3. **Install PostgreSQL,** set up a user for Control Center, and create databases:

```
sudo apt-get update

sudo apt-get install postgresql
sudo -u postgres psql -c "CREATE ROLE netrounds WITH ENCRYPTED PASSWORD 'netrounds' SUPERUSER
LOGIN;"

sudo -u postgres psql -c "CREATE DATABASE netrounds OWNER netrounds ENCODING 'UTF8' TEMPLATE
'template0';"
```

Using an external PostgreSQL server is not recommended.

4. **Install and configure an email server.**

- Control Center will send emails to users:

  - when they are invited to an account,

  - when sending email alarms (i.e. if email rather than SNMP is used for this purpose), and

  - when sending periodic reports.

- Run the command

```
sudo apt-get install postfix
```

- For a simple setup where postfix can send directly to the destination email server, you can set **General type of mail configuration** to "Internet Site", and **System mail name** can usually be left as-is. Otherwise, postfix needs to be configured according to the environment. For guidance, refer to the official Ubuntu documentation at ubuntu.com/server/docs/mail-postfix.

Downloading Control Center and Test Agent Repositories

These are the requisite files:

- Control Center

```
paa-control-center_4.3.0.15.tar.gz
```

- Test Agent repositories

```
paa-test-agent_4.3.0.16_all.deb
paa-test-agent-application_4.3.0.16_all.deb
```

- Plugin package

```
paa-test-agent-plugins_4.3.0.24_all.deb
```

This software is available at support.juniper.net/support/downloads (search for "Paragon Active Assurance").

Also click the **Checksums** link for each file and note down the SHA256 checksum. These will be used to verify the integrity of the downloaded files (as detailed in the chapter).

Installing Control Center and Related Tasks

IN THIS SECTION

Standard Procedure

1. **Install Control Center**.

```
export CC_BUILD=4.3.0.15

# Compute the SHA256 checksum for the tar file and verify that it is equal to the SHA256
# checksum provided on the download page
sha256sum paa-control-center_${CC_BUILD}.tar.gz

# Unpack the tarball
tar -xzf paa-control-center_${CC_BUILD}.tar.gz

# Make sure packages are up to date
sudo apt-get update

# Start the installation
sudo apt-get install ./paa-control-center_${CC_BUILD}/*.deb
```

An error "Download is performed unsandboxed as root ..." may occur during this installation. This is a harmless warning which can be ignored.

2. **Run the database migration.**

NOTE: This is a sensitive command, and care should be taken when executing it on a remote machine. In such a scenario it is strongly recommended that you use a program like `screen` (generally installed by default on popular Linux distributions) or `tmux` (run `sudo apt-get install`

> `tmux` to install) so that the migrate command will continue running even if the ssh session breaks.

```
sudo ncc migrate
```

The `ncc migrate` command takes considerable time to execute (many minutes). It should print the following (details omitted below):

```
Migrating database...
Operations to perform:
  <...>
Synchronizing apps without migrations:
  <...>
Running migrations:
  <...>
Processing models <...>:
<...>
Migrating plugin service database...
<...>
Creating cache table...
<...>
Syncing test scripts...
<Updating script ...>
Pre-creating kafka topics...
<...>
```

When the migration has finished, restart all Paragon Active Assurance services:

```
sudo ncc services restart
```

3. **Install the Test Agent repositories and plugins.**

The plugins are used by Test Agent Applications.

```
export TA_APPLIANCE_BUILD=4.3.0.16
export TA_APPLICATION_BUILD=4.3.0.16
export PLUGIN_BUILD=4.3.0.24


# Compute SHA256 checksums for the repositories and verify that they match the
```

```
# SHA256 checksums provided on the download page
sha256sum paa-test-agent_${TA_APPLIANCE_BUILD}_all.deb
sha256sum paa-test-agent-application_${TA_APPLICATION_BUILD}_all.deb
sha256sum paa-test-agent-plugins_${PLUGIN_BUILD}_all.deb

# Start the installation
sudo apt-get install ./paa-test-agent_${TA_APPLIANCE_BUILD}_all.deb
sudo apt-get install ./paa-test-agent-application_${TA_APPLICATION_BUILD}_all.deb
sudo apt-get install ./paa-test-agent-plugins_${PLUGIN_BUILD}_all.deb
```

NOTE: A message similar to the following may appear during plugin installation. It can safely be ignored.

```
W: APT had planned for dpkg to do more than it reported back (0 vs 4).
Affected packages: paa-test-agent-plugins:amd64
```

4. In the file `/etc/netrounds/netrounds.conf`, assign a URL to Control Center by setting `SITE_URL` appropriately. This URL is displayed for example in emails and reports.

Restart all Paragon Active Assurance services for this change to take effect:

```
sudo ncc services restart
```

This concludes the installation of Control Center. Before you can log in to the system you also need to go through the chapter .

Installing ConfD

This installation is needed only if you want to use the NETCONF & YANG API to communicate with Control Center. For full details on this topic, see the document *NETCONF & YANG API Orchestration Guide*.

ConfD (a product from Tail-f) is used as an intermediary between the Paragon Active Assurance system and NETCONF. ConfD connects Paragon Active Assurance configuration and operational data to the NETCONF & YANG API.

ConfD must be installed after Control Center has been installed.

Proceed as follows:

1. Get the Paragon Active Assurance NETCONF & YANG tarball:

```
export CC_BUILD=4.3.0.15
paa-netconf-yang_${CC_BUILD}.tar.gz
```

2. Compute the SHA256 checksum for the tar file and verify that it matches the SHA256 checksum provided on the download page:

```
sha256sum paa-netconf-yang_${CC_BUILD}.tar.gz
```

3. Unpack the tarball:

```
tar -xzf paa-netconf-yang_${CC_BUILD}.tar.gz
```

4. Install the `paa-netconf-yang` package:

```
sudo apt-get install ./paa-netconf-yang_${CC_BUILD}/*.deb
```

Some configuration is needed once you have set up your Paragon Active Assurance account; see the page.

Getting Started with Paragon Active Assurance

1. **Create a user** in Control Center by running this command:

```
ncc user-create <email> --password
```

Example: `jane.doe@example.com`

2. **Create an account** in Paragon Active Assurance with the user just created as owner:

```
ncc account-create --owner <email> --name "<full name of product account>" <short name of
product account>
```

Example: Full name = `"Example Corporation"`, short name = `example`.

The full name should be enclosed in double quotes. If it is not, any spaces in the name must be escaped with "\".

In the short name, only the following characters are allowed: a-z, 0-9, "-", and "_".

> **NOTE**: The account names are not the same as the owner's or some other user's personal name. The short account name will be used (for example) in the Control Center URL, while the full account name is what will normally be displayed to the user.

The `ncc account-create` command should normally be run only once, since by default you will have only one account in Paragon Active Assurance. The user specified as owner automatically gets admin permissions on the account.

3. **Activate licenses.**

Control Center will not be fully operational until valid licenses have been activated. Follow these steps to activate licenses:

- Generate a UUID string with the command

```
ncc license license-request
```

  You will need this UUID later on.

- Log in to the Juniper EMS Portal at license.juniper.net/licensemanage/ with the credentials you have received from Juniper.

- In the **My Product Licenses** view, expand **My Entitlements**.

- You will typically have a set of entitlements. The contents of the set may vary, but will include some of the following:

  - Control Center (generic)

  - Control Center with a given feature tier (Standard, Advanced, or Premium) and a specified number of tenants (= accounts), also encompassing Test Agents

  - A given number of *streams* entitling you to use product features

  - A set of entitlements for each of a number of Test Agents, licensed separately.

- For each entitlement that you want to activate (it is possible to activate only a subset), do the following:

    - Click the **Activate** button on the right.

    - In the dialog that appears, under **Software Version**, select **3.0 and Above**.

    - The field **Universal Unique ID (UUID)** appears. Enter the UUID string you generated in Control Center.

    - Check the box **I Agree with Terms & Conditions**.

    - Click the **Activate** button at the bottom of the screen.

- A license key will now be generated. Download it and save it as a plain-text file with a descriptive name, for example, `cc_license.txt` for the Control Center license.

You will thus end up with a set of license files in plain-text format.

> **NOTE**: It is vital that you enter the UUID exactly as received from Juniper, using the standard UUID format in lowercase with hyphens: for example, `0a1b2c3d-4e5f-6789-a0b1-c2d3e4f5678`.

> **NOTE**: If the activation of a license failed for some reason, for example because you mistyped the UUID, you can click the **Revoke** button in the **My Product Licenses** view (under **My Activations**) to revoke the license. You then need to start over with the `ncc license license-request` command as described above.

- Now activate the licenses in Control Center. You can do this for all license files at once as follows:

```
cat [list license files here] > all-licenses.txt
ncc license activate all-licenses.txt
```

  Alternatively, run the command `ncc license activate` for each license file in turn.

  This step is necessary in order to give the correct permissions to the account you just created. By default the account will be granted the full range of permissions granted by your licenses.

- *(Normally not needed:)* If you are using a *proxy* together with Control Center, you need to enter the following additional line in the file `/etc/environment`:

```
"no_proxy=localhost, 127.0.0.1"
```

  This is necessary for the license manager to work correctly, so that the product licenses can be activated.

4. You are now ready to log in to Control Center with a user and account that you have created.

   - **Open the URL** `SITE_URL` in your browser and log in with the credentials specified when creating the user.

   > **NOTE**: By default, Control Center including the REST API uses self-signed SSL certificates. The web browser will issue a warning about this. It is strongly advised that you obtain real certificates (signed by a certificate authority). See also the chapter Service Configuration, section "SSL Certificate Configuration" on page 19 .

5. An account in Control Center is like a tenant that multiple users can have access to. **Creating additional users** can be done either from the command line or in the Control Center web GUI:

- To create new users from the command line, repeat the command

```
ncc user-create <email> --password
```

with different email addresses as desired.

To grant a user permissions on your Paragon Active Assurance account, use the command

```
ncc user-permission <email> <short name of product account> <permission>
```

where `<permission>` is one of "none", "read", "write", or "admin".

- To invite new users and grant them permissions via the Control Center web GUI, go to **Account** > **Permissions**.

For details on what a user is allowed to do at each permission level, see the in-app help under "Setting up your account" > "Administering users and permissions".

6. If you have a license allowing **multiple accounts**, you can create further accounts by using the command

```
ncc account-create --owner <email> --name "<full name of product account>" <short name of product account>
```

as detailed "above" on page 12 . If the admin of such an account is to be the same as for the first account, simply assign the same user as `owner`; there is then no need to create an additional user.

7. If you have installed **ConfD**, you need to configure users as follows:

- Add permissions to the ConfD user in the Paragon Active Assurance account that will be managed by ConfD. This can be done either in the Control Center web GUI or from the command line. In the latter case, give the following command:

```
ncc user-permission confd@netrounds.com <account> admin
```

- To enable access from NETCONF, a user and password need to be set up. Run this command:

```
sudo /opt/netrounds-confd/ncc-netconf user create --username=<user name> --password=<password>
```

Here, `<user name>` and `<password>` are unrelated to your Control Center user name and password. A simple expedient is to use "confd" for both.

8. Please turn to the in-app help under "Test Agents" for instructions on how to download, install, register, and configure Test Agents:

- Download and installation on various platforms (including virtualization platforms): "Test Agents" > "Installation"

> **NOTE**: For installing Test Agent Applications on Juniper routers, please turn to the Junos® OS Evolved Software Installation and Upgrade Guide. This document is found in TechLibrary under "Junos OS Evolved". Follow the links provided in the in-app help.

- Registration: "Test Agents" > "Configuring from Test Agent local console" > "Registration"

- Configuration:

  - From the Test Agent local console: "Test Agents" > "Configuring from Test Agent local console"

  - From the Paragon Active Assurance GUI: "Test Agents" > "Configuring from Paragon Active Assurance GUI"

You access the in-app help by clicking the question mark at top right in the Paragon Active Assurance GUI.

Service Configuration

IN THIS SECTION

Services are configured by editing various configuration files, as detailed below. Go through this chapter and configure your settings as appropriate.

Summary of relevant configuration files:

- `/etc/apache2/sites-available/netrounds-ssl.conf`

- `/etc/apache2/sites-available/netrounds.conf`

- `/etc/environment`

- `/etc/netrounds/consolidated.yaml`

- `/etc/netrounds/metrics.yaml`

- `/etc/netrounds/netrounds.conf`

- `/etc/netrounds/plugin.yaml`

- `/etc/netrounds/probe-connect.conf`

- `/etc/netrounds/restol.conf`

- `/etc/netrounds/test-agent-gateway.yaml`

- `/etc/netrounds/timescaledb.conf`

- `/etc/openvpn/netrounds.conf`

Main Settings File

- `/etc/netrounds/netrounds.conf`

This file has inline documentation and examples for all supported settings. The `SITE_URL` setting is one that *always needs to be modified* to get the correct URL to Control Center, for example in emails and reports.

Summary of settings in this file:

- Unique, secret string used for cryptographic operations

- Control Center web server URL

- User time zone; the default is UTC

- Sender name in outgoing emails

- Contact email address shown to users

- Settings for sending email (backend, host, and more)

- Logging configuration (for details see the section "Logging" on page 24)

- Maximum length of log tags

- Criteria for automatic updating of Test Agent software

- Storage location for time series data

- Storage location for OpenVPN certificates and keys used to authenticate Test Agents

- Number of tasks from the background task queue that can be processed in parallel

SSL Certificate Configuration

- `/etc/apache2/sites-available/netrounds-ssl.conf`

This Apache configuration file contains the following SSL certificate settings, with default values as shown:

```
SSLCertificateFile      "/etc/ssl/certs/ssl-cert-snakeoil.pem"
SSLCertificateKeyFile   "/etc/ssl/private/ssl-cert-snakeoil.key"
```

For exhaustive information on this topic, please consult Apache documentation.

- `/etc/netrounds/test-agent-gateway.yaml`

This configuration file contains SSL certificate settings for the Test Agent Application Gateway, which is used by Test Agent Applications to connect to Control Center.

```
# Test Agent Application config file
# Please run the command below to see available settings:
#   /usr/bin/test-agent-gateway-service --help
```

```
# SSL certificates used by the web server. Defaults to snakeoil.
ssl-cert: /etc/ssl/certs/ssl-cert-snakeoil.pem
ssl-key: /etc/ssl/private/ssl-cert-snakeoil.key
```

By default snakeoil SSL certificates are used in all cases, as seen in the code snippets above. These are created from the `ssl-cert` package which is preinstalled in Ubuntu. However, to ensure an encrypted and secure connection in a production environment, you are strongly advised to obtain proper, signed SSL certificates instead.

Apache

- `/etc/apache2/sites-available/netrounds-ssl.conf`

- `/etc/apache2/sites-available/netrounds.conf`

These files hold Apache settings.

For exhaustive information on this topic, please consult Apache documentation.

> **NOTE**: It is strongly discouraged to change the Apache configuration files unless you are fully aware of the consequences. Inappropriate changes may break Paragon Active Assurance functionality.

TimescaleDB Configuration

How to configure TimescaleDB is described in the document *Querying Metrics in TimescaleDB*.

Plugin Service Database Configuration

The plugin service database is configured in

- `/etc/netrounds/plugin.yaml`

Configuration of OpenVPN Keys

The location of the OpenVPN keys is configured in

- `/etc/openvpn/netrounds.conf`

Restart OpenVPN for any changes to take effect.

HSTS Configuration

- `/etc/netrounds/netrounds.conf`

HTTP Strict Transport Security (HSTS) is a web security policy mechanism which helps to protect websites against protocol downgrade attacks and cookie hijacking. It allows web servers to declare that web browsers (or other complying user agents) should only interact with it using secure HTTPS connections, and never via the insecure HTTP protocol.

A server implements an HSTS policy by supplying a header (Strict-Transport-Security) over an HTTPS connection. The header age is set to one hour.

> **NOTE**: By default HSTS is disabled in Paragon Active Assurance since the Speedtest page uses HTTP for performance reasons. Uncomment the line below to enable HSTS if you are not using Speedtest.

```
# STRICT_TRANSPORT_SECURITY_HEADER = "max-age=3600; includeSubDomains"
```

Another way to allow enabling of HSTS in Control Center is to host Speedtest on a separate web server, as explained in the document *Creating a Custom Speedtest Web Page*.

Configuring the Lifetime of REST API Tokens

The lifetime of REST API tokens is limited and is 10 years by default. This is governed by the parameter `REST_TOKEN_LIFETIME` in the file `/etc/netrounds/netrounds.conf`.

If you intend to use the REST API, you might need to change the value of this parameter to whatever is required in your case.

> **NOTE**: In connection with an upgrade, you need to set the desired lifetime value for existing tokens prior to running the `ncc migrate` command.

Limiting the REST API Rate

You can apply REST API rate throttling in the file

- `/etc/netrounds/restol.conf`

  - The settings `RATE_LIMIT_ENABLED` and `RATE_LIMIT_DEFAULT` are used for throttling REST API requests. To enable throttling, set `RATE_LIMIT_ENABLED=True`.

  - Then configure the `RATE_LIMIT_DEFAULT` setting to indicate the maximum frequency of API requests from the same IP address. For example, `RATE_LIMIT_DEFAULT=30/second` will allow up to 30 requests per second from each IP address. The rate limit can be set per second, minute, hour, or day. It is also possible to set multiple limits, as in `RATE_LIMIT_DEFAULT=30/second,60/minute`.

  - To disable throttling, set `RATE_LIMIT_ENABLED=False`.

Configuring Password Strength

The *default* password strength requirements for Control Center user passwords are as follows: Each password must contain

- at least 8 characters in total

- at least one digit

- at least one uppercase letter

- at least one lowercase letter.

It is possible to switch to a set of *stricter* requirements. According to these, each password must contain

- at least 12 characters in total

- at least one digit

- at least one uppercase letter

- at least one lowercase letter

- at least one special character

and must also not have any character occurring twice in a row.

To enforce the stricter requirements, make the following setting in the file `/etc/netrounds/netrounds.conf`:

```
USER_PASSWORD_STRENGTH='strong'
```

(You need to add the variable `USER_PASSWORD_STRENGTH` itself as it is by default not present in the configuration file.) This setting applies globally to all new users created in Control Center.

Orchestration with NETCONF & YANG

How to integrate Paragon Active Assurance with a network service orchestrator via the Control Center NETCONF/YANG API is described in the document *NETCONF & YANG API Orchestration Guide*. That document also explains how to install ConfD, which is used as an intermediary between the Paragon Active Assurance system and NETCONF. Please note that ConfD is not included in the regular installation described in the present installation guide.

Troubleshooting

IN THIS SECTION

Logging

For troubleshooting it is generally helpful to check the following logs:

- `/var/log/apache/netrounds_access.log`: All HTTP requests made to the Control Center web interface.

- `/var/log/apache/netrounds_error.log`: Errors reported by Apache for HTTP requests towards the Control Center web GUI. Console output from the Control Center back-end is also in this file; by default, all logging is done by the console.

- `/var/log/syslog`: General system log.

- `/var/log/mail.log`: Email server log.

- `dmesg`: Kernel log

It may also help to turn on additional logging in `/etc/netrounds/netrounds.conf` by changing the line

```
LOGGING['handlers']['console']['level'] = 'ERROR'
```

to

```
LOGGING['handlers']['console']['level'] = 'DEBUG'
```

Another way to show logging for the Paragon Active Assurance callexecuter is to use the `journalctl` utility:

```
journalctl -u netrounds-callexecuter
```

Read more about `journalctl` here: www.digitalocean.com/community/tutorials/how-to-use-journalctl-to-view-and-manipulate-systemd-logs.

A more detailed list of logs is found in the Operations Guide, chapter *Monitoring the System*.

Root Shell Option in Test Agent Local Console

You can perform additional troubleshooting under the guidance of Paragon Active Assurance by logging into the root shell of a Test Agent. This is done from the Test Agent local admin console.

How to access the local console is explained in the in-app help under "Test Agents" > "Configuring Test Agents from the local console".

- Navigate to **Utilities** and select **Root shell**.

- Log in with password `onlyfordebugaccess`.

The root shell prompt now appears.

The root password can be changed within the root shell using the `passwd` command. Alternatively, this can be done using the **Change root password** option in the local console.

> ⚠ **WARNING**: Any changes made to the Test Agent in the root shell may cause the Test Agent to malfunction and/or void the warranty. When in doubt, always consult Juniper staff before proceeding.

The subsections that follow deal with specific problems.

Problem: Applying a license fails

Suggested actions:

- First try restarting the licensing service:

```
sudo systemctl restart netrounds-license-daemon
```

- If that does not help, you can wipe all licenses and reapply them as described in the Operations Guide, chapter *Deleting Licenses from the System*.

Problem: Services cannot be accessed

Suggested actions:

- Check service status with the following commands:

```
sudo systemctl status "netrounds-*" openvpn@netrounds
sudo systemctl status apache2 openvpn@netrounds
```

- Check that the host name resolves to the correct IP, for instance using `dig` or `ping`

- Check listening ports with

```
netstat -lan
```

- Check network traffic with

```
tcpdump
```

Problem: No data collected

- *Possible cause:* NTP time sync problem. This will introduce a time offset in the server-generated result views, so that it may take a while until any results appear in these views.

- *Possible cause:* No free disk space.

Problem: Data collection is slow, or data loss occurs

- *Possible cause:* Tests and monitors are queuing up so that data collection is delayed.

Run the command

```
ncc status
```

and check the value of `scheduled_call_latency`, which indicates the length of the queue. See the Operations Guide, chapter , section "Control Center" for further information and a suggested remedy.

Problem: ncc migrate command fails

- *Possible cause:* Zookeeper has gone down.

If the `ncc migrate` command fails with an error message saying "Unable to create Kafka topics", this could be due to Control Center being unable to start the Zookeeper server process. Kafka relies on Zookeeper and requires it to be running in order to operate correctly.

There is a known issue where certain Zookeeper files are corrupted, which causes a Java EOF exception. To check the logs for this, run the following command

```
sudo tail -n 100 /var/log/syslog
```

and look for `ERROR` entries related to Zookeeper:

```
2021-02-20 18:55:13,302 - ERROR [main:ZooKeeperServerMain@64] - Unexpected exception, exiting
abnormally
java.io.EOFException
```

To fix this, delete all files of size zero that are located in the `/var/lib/zookeeper/version-2` folder. First identify these files:

```
/var/lib/zookeeper/version-2# du -sh *
0K      log.1
8.0K    log.1dd
```

```
8.0K    log.210
...
```

Then delete the corrupted file or files:

```
sudo rm /var/lib/zookeeper/version-2/log.1
```

Finally, restart Zookeeper:

```
sudo systemctl restart zookeeper
```

Problem: Services fail

- *Possible cause:* Kafka has gone down. This in turn may be due to Zookeeper issues, or to the system having run out of memory.

If this happens, services relying on Kafka will also fail and try to reboot until Kafka comes back online.

For example, if the netrounds-ta3-compat service fails, Test Agent Applications will be unable to register to Control Center.

You can check if Kafka is running with the following command:

```
sudo systemctl list-units --type=service | grep kafka
```

If Kafka has gone offline, then checking logs with

```
journalctl -u netrounds-test-agent-gateway.service
```

will return entries like the following:

```
Oct 17 11:17:05 example.com test-agent-gateway-service[23009]: {"level":"info","service":"test-
agent-gateway-
service","host":"example.com","src":"core","time":"2020-10-17T11:17:05.429473575Z","caller":"/app
/cmd/server/
Oct 17 11:17:06 example.com systemd[1]: netrounds-test-agent-gateway.service: Main process
exited, code=exited, status=1/FAILURE
```

```
Oct 17 11:17:06 example.com systemd[1]: netrounds-test-agent-gateway.service: Unit entered
failed state.
Oct 17 11:17:06 example.com systemd[1]: netrounds-test-agent-gateway.service: Failed with result
'exit-code'.
Oct 17 11:17:07 example.com systemd[1]: netrounds-test-agent-gateway.service: Service hold-off
time over, scheduling restart.
Oct 17 11:17:07 example.com systemd[1]: Stopped Netrounds TA3 Connection Service.
```

Problem: Kafka error "Too many open files"

- *Detailed description:* Kafka is repeatedly restarting and crashing after some seconds have passed. In the journal log, Kafka displays a stacktrace with the error: `Too many open files`.

- *Cause:* The default setting governing how many file descriptors Kafka is allowed to use is too low. The allowed number will be sufficient in the beginning but may become too small after some time.

To resolve:

- Log in to Control Center and run

```
sudo systemctl edit kafka.service
```

- An editor will open. Insert the following lines:

```
[Service]
LimitNOFILE=65536
```

  Save the file and exit.

Kafka should now successfully start and work correctly.

Problem: A Test Agent has successfully registered but does not come online (status icon remains red)

- *Possible cause:* The network does not allow the encrypted VPN connection to be established. Please check the configuration and logs of any firewall between the Test Agent and Control Center.

Detailed description:

The registration is done over HTTPS, while the connection setup after registration is done using OpenVPN on the same port. This can sometimes cause the network to allow the registration, but not the connection attempt.

- *Possible cause:* The clocks on the Test Agent and in Control Center are not in sync. Please check that both parties have NTP correctly configured, that their clocks are in sync with the NTP server, and that the NTP server clock in turn is also in sync.

Detailed description:

NOTE: This description and the remedy that follows are applicable only when Test Agents are not coming online during a *fresh installation* of Control Center, and the time sync has been confirmed to be incorrect.

If the Test Agent clock is behind, then after the Test Agent registers, the TLS certificate signed by Control Center for the Test Agent will be invalid from the Test Agent's point of view until the Test Agent's clock has reached the time of signing according to the Control Center clock. Until that point, the Test Agent will not accept the certificate and will remain offline.

The TLS certificates are generated at the time of installing Control Center. If the clocks were not correct at this point, the certificates must be regenerated. Note that this also requires re-registration of all Test Agents. Follow these steps:

1. Make sure that the Control Center clock is in sync:

```
ntpq -np
```

2. Remove the old certificates:

```
rm /var/lib/netrounds/openvpn/*
```

3. Generate new certificates:

```
dpkg-reconfigure paa-test-agent-login
```

4. Register each Test Agent again under a different name.

5. Restart Control Center services:

```
sudo ncc services restart
```

Problem: Disk is running out due to incremental backups of TimescaleDB data

Detailed description:

Disk space is running out due to files being continuously created under

```
/var/lib/netrounds/rrd/timescaledb/pgbackrest/repo/archive/paa-metrics/
```

This is due to TimescaleDB incremental backups being turned on by default.

To resolve:

- In the file `/var/lib/netrounds/rrd/timescaledb/data/postgresql.conf`, set

```
archive_mode = off
```

  to turn off these backups.

- Restart the `netrounds-timescaledb` service:

```
sudo ncc services restart netrounds-timescaledb
```

Problem: Error when accessing Rest API URL and TypeError from apache2 service logs

If you are unable to access the REST API URL:

1. Access the **apache2.log** file.

```
sudo journalctl -b -u apache2 > apache2.log
```

2. Check whether the apache2.log file has the following exception for **Traceback**:

```
Traceback (most recent call last):
File "/usr/lib/python3.10/dist-packages/restol/netrounds_restol.wsgi", line 9, in <module>
application = create_app(load_config())
File "/usr/lib/python3.10/dist-packages/restol/restol app.py", line 158, in create app
Limiter(
TypeError: Limiter.__init__() got multiple values for argument 'key_func'
```

3. Do one of the following:

   - If **Traceback** is not found in the **apache2.log** file, then file a ticket at support.juniper.net/support/requesting-support. You must attach the **apache2.log** file.

   - If **Traceback** is found in the **apache2.log** file"

     a. Open **/usr/lib/python3.10/dist-packages/restol/restol_app.py**.

     b. Replace the following:

```
 Limiter(
app,
default_limits=config['rate_limit_default'],
headers_enabled=True,
key_func=get_remote_address,
)
```

     with

```
 Limiter(
get_remote_address,
app=app,
default_limits=config['rate_limit_default'],
headers_enabled=True
)
```

     c. Save **/usr/lib/python3.10/dist-packages/restol/restol_app.py**

d. Download Ubuntu – Package Download Selection -- python3-limits_2.8.0-1_all.deb.

e. Install the downloaded file using the **sudo apt-get install /path/to/python3-limits_2.8.0-1_all.deb** command.

> **NOTE**: Ensure that you change /path to /prefix along with the path where **python3-limits deb package** was copied.

f. Restart Control Center services by running the **sudo ncc services restart** command.

Problem: Errors related to BF-CBC cipher in the openvpn log

This issue occurs if the old test agents are unable to connect to Control Center Release 4.2. In this case, you need to add the support for BF-CBC cipher in the **openvpn** log.

To add the support for BF-CBC cipher:

1. Access **/etc/openvpn/netrounds.conf**.

2. Add BF-CBC cipher to **/etc/openvpn/netrounds.conf**

```
data-ciphers AES-256-GCM:AES-128-GCM:BF-CBC
data-ciphers-fallback BF-CBC
```

Contacting Juniper Technical Support

To contact Juniper technical support, file a ticket at support.juniper.net/support/requesting-support. In your ticket, please upload the file generated from running the command

```
ncc generate-troubleshooting-report
```

The file will be located in your user's home directory (/home/<username>).

Problem: Issues Related to ncc Commands

You might see the following warning when you execute **ncc**-related commands from the Control Center CLI:

```
WARNINGS:

?: (axes.W003) You do not have 'axes.backends.AxesStandaloneBackend' or a subclass in your
settings.AUTHENTICATION_BACKENDS.

    HINT: AxesModelBackend was renamed to AxesStandaloneBackend in django-axes version 5.0.
```

To avoid this, in the **/etc/netrounds/netrounds.conf** file, ensure that you specify *'axes.backends.AxesBackend'* as the first item in the *AUTHENTICATION_BACKENDS*.

For example,

```
AUTHENTICATION_BACKENDS = (
    'axes.backends.AxesBackend',
    'netrounds.domain.backends.LDAPAuthBackend',
    'netrounds.domain.backends.AuthBackend',
)
```

LDAP Authentication

This chapter is relevant only if you wish to use an LDAP server to authenticate Paragon Active Assurance users.

Paragon Active Assurance supports the use of LDAP to manage and authenticate its users in a centralized way. The authentication is then done using a remote server instead of the local Control Center user database.

When a user attempts to log in to Control Center, the latter sends an authorization request to the LDAP server. Based on the response, Control Center grants or denies the user access to Paragon Active Assurance accounts as detailed in the response.

- If some account is defined twice in the mapping from LDAP to Control Center, the user will receive the higher permission of the two granted. Thus, if the permission is set to "read" in one list element and to "admin" in another, the user will receive admin permission for that account.

- If one permission mapping grants permission to one account and another mapping denies it, then the user will receive access to the account.

- Account permissions are synchronized with the LDAP server on each user login. If the user is granted additional permissions by the Paragon Active Assurance local admin, these are valid only until the next time the user logs in. Conversely, if the user's privileges are changed on the LDAP server, these will not come into effect until next login.

- If the user name entered at login matches the email address of an existing Control Center user, the login will proceed using that user rather than a new one being created. However, during server authentication, all user profile details except the password will be overwritten with what is stored on the server, insofar as these details have been defined. The user can still log in using the existing Control Center password. This means that select users can have the password set in Control Center, so that it is still possible to log in if the LDAP server goes down.

- If the user name entered at login does not exist in Control Center, the following happens:

  - The user's email address is read from the user `email` field. The name of this field can be changed using the setting `AUTH_LDAP_USER_ATTR_MAP`.

  - If the email address is valid, it will be entered as email address in the Control Center database as well. However, the user will still have to log in with his LDAP username.

  - If the email address is not valid, an email address will be created with the structure `username@LDAP_EMAIL_DOMAIN` and entered into the database. Edit the settings file to change this domain.

We will illustrate the above by reproducing a typical (OpenLDAP) server-side file with data preloaded into the LDAP database, and subsequently showing what corresponding configuration is necessary in Control Center.

Contents of `ldap.ldif`:

```
dn: dc=example,dc=com
        objectClass: organization
        objectClass: dcObject
        objectClass: top
        dc: example
        o: example.com

        dn: ou=users,dc=example,dc=com
        objectClass: top
        objectClass: groupOfNames
        cn: Users
        member: uid=jsmith,dc=example,dc=com
```

```
member: uid=jane.smith@example.com,dc=example,dc=com
ou: users

dn: uid=jsmith,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: extensibleObject
cn: John Smith
givenName: John
sn: Smith
c: GB
telephoneNumber: +44 20 7946 0296
mail: john.smith@example.com
uid: jsmith
userPassword: Password1

dn: uid=jane.smith@example.com,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: extensibleObject
cn: Jane Smith
givenName: Jane
sn: Smith
c: GB
telephoneNumber: +44 20 7946 0580
mail: jane.smith@example.com
uid: jane.smith@example.com
userPassword: Password1

dn: ou=admins,dc=example,dc=com
objectClass: top
objectClass: groupOfNames
cn: Administrators
member: uid=jdoe,dc=example,dc=com
member: uid=jane.doe@example.com,dc=example,dc=com
ou: admins

dn: uid=jdoe,dc=example,dc=com
objectClass: top
```

```
        objectClass: person
        objectClass: organizationalPerson
        objectClass: inetOrgPerson
        objectClass: extensibleObject
        cn: John Doe
        givenName: John
        sn: Doe
        c: GB
        telephoneNumber: +44 20 7946 0344
        mail: john.doe@example.com
        uid: jdoe
        userPassword: Password1

        dn: uid=jane.doe@example.com,dc=example,dc=com
        objectClass: top
        objectClass: person
        objectClass: organizationalPerson
        objectClass: inetOrgPerson
        objectClass: extensibleObject
        cn: Jane Doe
        givenName: Jane
        sn: Doe
        c: GB
        telephoneNumber: +44 20 7946 0113
        mail: jane.doe@example.com
        uid: jane.doe@example.com
        userPassword: Password1
```

This creates a total of four users, two with write privileges (`jsmith` and `jane.smith@example.com`) and two with admin privileges (`jdoe` and `jane.doe@example.com`). Note how two of the users have their user name as `uid`, while the other two have a `uid` consisting of an email address.

In order to enable LDAP authentication in Control Center, the following attributes have to be provided in the settings file `/etc/netrounds/netrounds.conf`:

```
# LDAP settings


        import ldap
        # NOTE: ActiveDirectoryGroupType is only needed if you are using Microsoft Active
Directory.
        from django_auth_ldap.config import LDAPSearch, GroupOfNamesType,
ActiveDirectoryGroupType
```

```
AUTH_LDAP_ENABLED = True
AUTH_LDAP_START_TLS = True
AUTH_LDAP_SERVER_URI = 'ldap://ldap.example.com'
AUTH_LDAP_USER_DN_TEMPLATE = 'uid=%(user)s,dc=example,dc=com'

# LDAP user attribute value to use during lookup on database side
AUTH_LDAP_USER_QUERY_FIELD = 'email'

# This is the domain that will be appended to the username when the one from
# LDAP is not a valid email
AUTH_LDAP_EMAIL_DOMAIN='example.com'

# Set up the basic group parameters
# For vanilla LDAP such as OpenLDAP, use this.
GROUP_OBJECTCLASS_NAME ='groupOfNames'
# Uncomment this if using Microsoft Active Directory
# GROUP_OBJECTCLASS_NAME  = 'group'

AUTH_LDAP_GROUP_SEARCH = LDAPSearch('dc=example,dc=com',
ldap.SCOPE_ONELEVEL,
'(objectClass=%s)' % GROUP_OBJECTCLASS_NAME
)

# LDAP group type used to indicate group membership for a user
AUTH_LDAP_GROUP_TYPE = GroupOfNamesType()
# Note: If you are using Microsoft Active Directory, use this instead:
# AUTH_LDAP_GROUP_TYPE = ActiveDirectoryGroupType()

# Mapping between LDAP groups and Control Center account/permission
AUTH_LDAP_ACCOUNT_GROUP_MAPPING = [
{
'dn': 'ou=users,dc=example,dc=com',
'accounts': [
{
'name': 'dev',
'permission': 'write'
}
]
},
{
'dn': 'ou=admins,dc=example,dc=com',
'accounts': [
```

```
{
'name': 'dev',
'permission': 'admin'
}
]
}
]


# Populate the Django user from the LDAP directory
AUTH_LDAP_USER_ATTR_MAP = {
'first_name': 'givenName',
'last_name': 'sn',
'email': 'mail',
'phone': 'telephoneNumber',
'country': 'c'
}


# LDAP connection options
AUTH_LDAP_CONNECTION_OPTIONS = {
# Limit on waiting for a network response in seconds. This is a timeout
# returned by poll/select following a connect in case of no activity.
# OpenLDAP specific.
ldap.OPT_NETWORK_TIMEOUT: 30,
# Timeout value for the synchronous API calls in seconds. OpenLDAP specific.
ldap.OPT_TIMEOUT: 30
}


# The following user must exist unless anonymous login is allowed
# Non-anonymous bind DN
# Note: It is mandatory for AUTH_LDAP_BIND_DN to start with a 'cn=' relative
# distinguished name. Using 'uid=' will not work in case OpenLDAP is used as server.
AUTH_LDAP_BIND_DN = 'cn=admin,dc=example,dc=com'
# Non-anonymous bind password
AUTH_LDAP_BIND_PASSWORD = 'Password1'


# Finally, in order to get LDAP working, the first list item must be
# added below
AUTHENTICATION_BACKENDS = (
'netrounds.domain.backends.LDAPAuthBackend',
'netrounds.domain.backends.AuthBackend',
)
```

Most of the above follows what is documented at https://django-auth-ldap.readthedocs.io/en/latest/reference.html#settings.