Teradici PCoIP Connection Manager and Security Gateway

The *PCoIP Connection Manager* and the *PCoIP Security Gateway* are components of Teradici Cloud Access Software, and are deployed together as a set. Multiple instances of the Connection Manager and Security Gateway can be deployed to handle mixed LAN and WAN access points or for scaling large systems.

> 🔥 **Components in this release**
>
> The PCoIP Connection Manager and Security Gateway 22.01 is a combined release containing:
>
> • PCoIP Connection Manager 22.01
>
> • PCoIP Security Gateway 22.01

About the PCoIP Connection Manager

The *PCoIP Connection Manager* enables connections between PCoIP clients and PCoIP agents installed on remote desktops. It uses a required third-party connection broker to authenticate users, query available desktops and applications, and then establish a PCoIP connection between the client and the selected desktop.

About the PCoIP Security Gateway

The *PCoIP Security Gateway* enables WAN users to securely access their remote desktops via the Internet without a VPN connection.

> ✏️ **Note**
>
> The PCoIP Security Gateway is not required for LAN access.

Establishing a PCoIP Connection With the Connection Manager and Security Gateway

The diagram shown next illustrates a brokered connection to the PCoIP host machine using the PCoIP Connection Manager and the PCoIP Security Gateway.

> ⚠️ **Caution: A dedicated server is strongly recommended**
>
> Since the PCoIP Connection Manager is a component that handles authentication data for users connecting to virtual desktops, Teradici strongly recommends installing the PCoIP Connection Manager and PCoIP Security Gateway on a dedicated server that is accessible only by authorized system administrators according to your organization's security policy.

Deployment Scenarios

Depending on your deployment scenario, you can install the PCoIP Connection Manager with the PCoIP Security Gateway disabled.

- **All your desktops are on a LAN (internal access only)**: you may only need to install one PCoIP Connection Manager. Since the PCoIP Security Gateway isn't required for LAN connections, you can optionally disable it.

- **All your desktops are on a WAN**: Install one PCoIP Connection Manager, leaving the Security Gateway enabled. The Connection Manager will handle PCoIP Connection establishment and the Security Gateway will secure the PCoIP session across the public internet.

- **Your desktops are on both a LAN and WAN**: Teradici recommends installing at least two groups of connection managers; one for internal access with the PCoIP Security Gateway disabled, and one for external access with the PCoIP Security Gateway enabled. You can set up the DNS so that internal and external users are routed to the appropriate connection manager.

- **If you are exceeding the** system specifications **or have high availability requirements**: If you serve a large number of desktops, or require high availability, install additional connection managers and implement load balancing.

What's New in This Release

- **Support for IPv6 in Online Environments**: The PCoIP Connection Manager and PCoIP Security Gateway now supports IPv6 connections (pure IPv6, or mixed IPv4/IPv6) when connected to the public internet. Offline (dark site) deployments do not support IPv6 connections yet.

- **Support for Rocky Linux 8**: The PCoIP Connection Manager and PCoIP Security Gateway now supports RHEL 8 and Rocky Linux 8.

- **Modernized application architecture**: The application and its deployment tooling have been upgraded, and system dependencies, installation procedures, and application options have changed significantly.

  If you are coming from earlier versions, please carefully review this documentation.

System Requirements

The minimum system requirements for a PCoIP Connection Manager and PCoIP Security Gateway are:

- 2 or more CPUs or vCPUs, 2.5 GHz or higher

- 4 GB of RAM

- 4 GB of swap space

- 10 GB of free disk space

Supported operating systems:

- RHEL 8

- Rocky Linux 8

If the connection broker is configured to identify resources by host name, then DNS must be available in PCoIP Connection Manager and the PCoIP Broker.

Installation Prerequisites

The PCoIP Connection Manager and PCoIP Security Gateway depends on the following packages:

- **Docker 20.10.0** or higher

Project dependencies must be installed on the production machine *before* installing the PCoIP Connection Manager and PCoIP Security Gateway.

> ⚠ **Caution: Dependencies in offline environments**
>
> If your deployment will be running in an environment that is not connected to the public internet (a *dark site*), you must download the package dependencies, transfer them to the production machine, and install them before installing the PCoIP Connection Manager and PCoIP Security Gateway.

PCoIP Connection Manager and PCoIP Security Gateway Performance Limits

The following statistics represent the performance limits of the PCoIP Connection Manager and PCoIP Security Gateway with a *minimum* system configuration. You can exceed these limits, unless indicated, with more powerful systems.

PCoIP Connection Manager Limits

Session Establishment Limits

Based on the minimum connection manager system requirements, the PCoIP Connection Manager can establish the following number of sessions:

- 40 simultaneous *in-process* session establishment sequences
- Up to 400 simultaneous client communications

PCoIP Security Gateway Limits

Session Limits

Each PCoIP Security Gateway supports a maximum of **5,000** simultaneous sessions. You can lower this limit by changing the `MaxConnections` setting in **/opt/teradici/pcoipcm_data/ data/SecurityGateway.conf**. If you need to support more than 5,000 simultaneous sessions, deploy additional PCoIP Connection Manager and PCoIP Security Gateways behind a load balancer.

Bandwidth Limits

When using the PCoIP Connection Manager and Security Gateway there are certain session establishment and session bandwidth limits when dealing with external connections.

The following table outlines the RAM, vCPU and correlated estimated bandwidth support for all combined concurrent sessions:

vCPUs	RAM	Estimated Bandwidth
2vCPU	7.5 GB RAM	~ 365 Mbit/s
4vCPU	15 GB RAM	~ 830 Mbit/s
8vCPU	30 GB RAM	~ 1100 Mbit/s

✏️ **Estimated Bandwidth**

These are estimated bandwidth levels. The bandwidth can vary based on the host, OS, CSP, etc.

1100 Mbit/s is approximately the maximum bandwidth that can be achieved. Additional gains may be possible with larger sizing.

System Planning

Before deploying the PCoIP Connection Manager and PCoIP Security Gateway, ensure you understand the PCoIP session establishment process and how load balancers and firewalls fit in.

Session Establishment

Here's the sequence of events involved in establishing a PCoIP session in a typical brokered scenario. In this example, the PCoIP client is outisde the firewall, so the PCoIP Security Gateway is enabled to secure the connection and to proxy authorized traffic.

1. A user provides a server name and address to their PCoIP client, which passes the data to the **PCoIP Connection Manager** (this can be relayed through a load balancer, as shown here).

2. The *Connection Manager* communicates with the **Connection Broker** to authenticate the user and to obtain the list of desktops the user is entitled to use.

3. The *Connection Broker* passes the list of desktops back to the the **PCoIP Client**.

4. The user selects a desktop from the client UI, and their choice is passed back to the **PCoIP Connection Manager**.

5. The *PCoIP Connection Manager* prepares the **PCoIP Security Gateway** and the requested desktop's **PCoIP Agent**.

6. The *PCoIP Agent* acquires a session license from a licensing service (either the **PCoIP Cloud Licensing Service** or the a local **PCoIP License Server**).

7. The PCoIP session is established. The **PCoIP Client** now communicates directly with the selected desktop using the PCoIP Protocol.

> ✏️ **Note: PCoIP Security Gateway in LAN systems**
>
> The PCoIP Security Gateway secures PCoIP communications through the firewall. In systems where PCoIP clients are on the WAN, PCoIP traffic is relayed through the PCoIP Security Gateway. When the entire PCoIP system is on your company LAN, the PCoIP Security Gateway is unnecessary and the PCoIP Client and PCoIP agent communicate directly.

Load Balancing

You can use load balancers in front of multiple connection managers and security gateways to distribute system load to optimize performance. The load balancer must support the following:

- HTTPS
- Sticky sessions by the jsessionid

During session establishment, the PCoIP Connection Manager retrieves the PCoIP Security Gateway's public IP address and passes it to the client. After the session is established, the client uses the provided IP address to communicate directly with the PCoIP Security Gateway.

> 🔥 **Important: The PCoIP Security Gateway's public IP address must be set during installation**
>
> If the public IP address is configured to point to the *load balancer* instead of the *PCoIP Security Gateway*, the load balancer may direct the client to a PCoIP Security Gateway on the wrong server. If this happens, the client will not be able to establish a session.
>
> The IP address is set using the `--external-pcoip-ip` flag during installation.

> 🔥 **Public IP Address**
>
> The machine with the PCoIP Connection Manager and Security Gateway on it must have a public IP address if it is directly accessed from WAN.

To see how load balancers fit into firewall configurations, refer to Configuring Firewalls.

Configuring Firewalls

If there is a firewall on the PCoIP Connection Manager server, ensure ports for PCoIP traffic are open so that users can access their desktop. The illustration shown next shows the default port numbers.

Firewall recommendations for establishing a PCoIP Session

Source	Port	Destination	Port	Description
PCoIP Client	*	PCoIP Connection Manager	TCP: 443	PCoIP broker protocol (HTTPS)
PCoIP Connection Manager	*	Connection broker	TCP: 443	PCoIP broker protocol (HTTPS)
PCoIP Connection Manager	*	PCoIP Agent	TCP: 60443	PCoIP agent protocol
PCoIP Client	*	PCoIP Security Gateway	UDP: 4172	PCoIP user data

Source	Port	Destination	Port	Description
PCoIP Client	*	PCoIP Security Gateway	TCP: 4172	PCoIP control information
PCoIP Security Gateway	*	PCoIP Agent	TCP: 4172	PCoIP control information
PCoIP Security Gateway	UDP: 55000	PCoIP Agent	UDP: 4172	PCoIP user data. *When deploying a desktop with a PCoIP agent, only port 4172 needs to be open.*

Inbound Connections

Ensure these ports are open for inbound connections:

Port	Purpose
443 TCP	Used by clients to connect to the PCoIP Connection Manager
4172 TCP/UDP	Used by authorized clients to connect to the PCoIP Security Gateway

Instructions for opening these ports are included in the installation procedures.

Note that RHEL 8 and Rocky Linux 8 permit all outbound traffic by default.

> 🔥 **Important: Other required services may need open outbound ports**
>
> If the PCoIP Connection Manager is on a network behind a firewall that blocks outbound connections, ensure that the required ports for other required operating system services are open. Teradici recommends that DHCP, DNS, and NTP are active for PCoIP Connection Manager operation.

Configuring Docker Network

The docker network environment for the PCoIP Connection Manager and the PCoIP Security Gateway is defined in `/opt/teradici/pcoipcm_data/docker-compose.yaml` under the key `networks`. By default, it is assigned to `10.101.0.0/24`.

```
cmdeployment:
ipam:
    config:
    - subnet: 10.101.0.0/24
```

If your company network CIDR overlaps `10.101.0.0/24`, change the default network range in `/opt/teradici/pcoipcm_data/docker-compose.yaml` to resolve the conflict. Addresses from any of the following CIDR classes can be used:

```
Class A: 10.0.0.0 to 10.255.255.255.
Class B: 172.16.0.0 to 172.31.255.255.
Class C: 192.168.0.0 to 192.168.255.255.
```

for example:

```
cmdeployment:
  ipam:
    config:
    - subnet: 172.16.0.0/24
```

After editing `docker-compose.yaml`, run the following command to apply your changes:

```
sudo pcoip-cmsg-setup configure --compose-file /opt/teradici/pcoipcm_data/docker-compose.yaml
```

Installing for Online Environments

The following sections outline how to install the Modern Connection Manager and Security Gateway 22.01.

Before You Begin

Before you proceed with installation, note the following:

- **Docker must be installed** before you begin. For instructions, see About Docker.

- Make sure ports TCP:80, TCP:443, TCP:4172, and UDP:4172 are open:

```
firewall-cmd  --add-port 80/tcp
firewall-cmd  --add-port 443/tcp
firewall-cmd  --add-port 4172/tcp
firewall-cmd  --add-port 4172/udp
```

- If you will be using IPv6, set up the required port forwarding rules:

```
# Add port forwarding rules
firewall-cmd  --add-forward-port=port=443:proto=tcp:toport=8443
firewall-cmd  --add-forward-port=port=80:proto=tcp:toport=8080
firewall-cmd  --add-rich-rule='rule family=ipv6 forward-port protocol=tcp
port=443 to-port=8443'
firewall-cmd  --add-rich-rule='rule family=ipv6 forward-port protocol=tcp
port=80  to-port=8080'

# Make the new settings persistent
firewall-cmd --runtime-to-permanent
```

- If your environment has `podman` or `buildah` installed, uninstall them before proceeding.

```
sudo dnf erase podman buildah -y
```

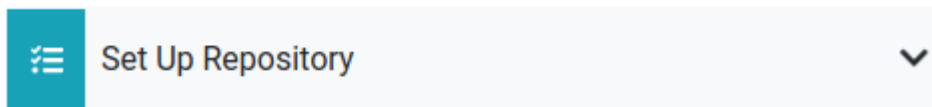# Install PCoIP Modern Connection Manager and PCoIP Security Gateway

1. On the machine that will host the PCoIP Connection Manager and PCoIP Security Gateway, open a browser and go to the PCoIP Connection Manager and PCoIP Security Gateway download page.

2. Click **Downloads and scripts**:

   → Downloads and scripts
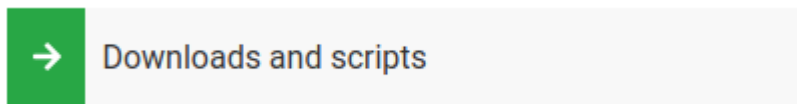
   If you see a login button instead, click it to log into the site and then proceed.

3. Accept the End User License Agreement, then click **Set Up Repository**:

   ≔ Set Up Repository ⌄

   .

   The window will expand and show the setup scripts for each supported operating system. Copy the command for your system to the clipboard.

4. Open a console window and paste in the command you copied in the previous step. You may need to press `Enter` to execute it.

   The command fetches a configuration script from our servers and runs it locally, setting up and configuring the repository on the local machine.

5. Install the PCoIP Connection Manager and PCoIP Security Gateway package:

   ```
   sudo dnf install pcoip-cmsg-setup
   ```

6. After the package is installed locally, run the `pcoip-cmsg-setup install` command with the required flags to complete installation.

   ```
   sudo pcoip-cmsg-setup install <installation_flags>
   ```

> 🔥 **Important: Required installation flags**
>
> There are a number of options and settings available. You can invoke the `install` command with the `--help` flag to list them:
>
> ```
> pcoip-cmsg-setup install --help
> ```
>
> They are also listed in the next section.
>
> The `install` command will prompt you for required parameters that have not been supplied via flags.

Installation Flags and Options

The following flags can be used to provide values at the command line. Flags that are required are identified in the description.

Boolean values should be provided as either `true` or `false`, lowercased, as in this example:

```
--example-flag=true
```

Flag	Type	Description
`--accept-policies`	Boolean	Automatically accepts the EULA and Privacy Policy. **Required.**
`--broker-url`	String	The URL of the PCoIP Broker, specified either as a https://: or https://: or https://[]:. **Required.**
`--ca-cert`	String	The full path and filename of the custom Certificate Authority's public certificate to be used in the PCoIP Connection Manager and PCoIP Security Gateway. **Required if** `--self-signed` **is not used.**
`--compose-file`	String	Specify the full path to a local docker-compose file.
`--darksite-bundle-path`	string	The path of darksite install bundle to be used for darksite installation
`--docker-password`	String	Password to login to private registry.

© 2021 Teradici   16

Flag	Type	Description
`--docker-registry`	String	Specifies the Teradici source for Cloud Access Connector images to be install from. **Debugging only**: This is intended to be used for debugging purposes and should not be used without guidance from Teradici support. Using this flag incorrectly can result in failed installations.
`--docker-username`	String	Username to login to private registry.
`--enable-collaboration`	Boolean	Allow multiple PCoIP clients to collaborate on a PCoIP agent. (default true)
`--enable-ipv6`	Boolean	Enables IPv6 connections (default false). To enable IPv6 use `--enable-ipv6=true`. To disable IPv6 use `--enable-ipv6=false`, or omit this flag.
`--external-pcoip-ip`	StringArray	Sets the public IP address of Security Gateway. If `--enable-ipv6` is true, this option may be used twice (once for IPv4 and once for IPv6). **Required if PCoIP Security Gateway is enabled**
`--enable-security-gateway`	Boolean	Enable and use the PCoIP Security Gateway (default true).
`--help`		Lists all available flags.
`--host-address`	stringArray	Sets the host FQDN/IP address. The option may be used twice (once for the IP address and once for the FQDN)
`--ignore-disk-req`	Boolean	Ignore the check for the minimum disk space requirement.
`--license-server-url`	String	The address of the locally installed PCoIP License Server. Example: `https://<license-server-address>:<port>`
`--self-signed`	Boolean	Automatically generate self-signed SSL cert and key for testing purposes. If specified, `--ssl-key` and `--ssl-cert` options are ignored.

Flag	Type	Description
`--ssl-cert`	String	The full path and filename of the SSL certificate to be used in the PCoIP Connection Manager and PCoIP Security Gateway. **Required if** `--self-signed` **is not used.**
`--ssl-key`	String	The full path and filename of the SSL key to be used in the PCoIP Connection Manager and PCoIP Security Gateway. **Required if** `--self-signed` **is not used.**

About Docker

The PCoIP Connection Manager and PCoIP Security Gateway depends on Docker 20.10.0 or higher, which must be installed on the machine before you install the PCoIP Connection Manager and PCoIP Security Gateway.

If you have not installed Docker yet, install it now.

If you are not sure if Docker is installed, or are not sure what Docker version you have, verify your Docker version first.

Verifying Docker Version

To verify your Docker installation and version:

1. SSH into the machine.
2. Open a console window and run the following command:

```
sudo docker -v
```

   - If Docker is **not** installed, this command will produce an error. Installation instructions are provided in the next section.
   - If you see a version number that is **lower** than 20.10.0, you must uninstall Docker and then re-install the supported version. Instructions for uninstalling and installing are provided in the next section.

- If you see a version number that is equal to or higher than 20.10.0, you have a compatible version of Docker already installed and can skip to PCoIP Connection Manager and PCoIP Security Gateway installation.

Uninstalling Docker

You'll only need to do this if you have an unsupported version of Docker already on the machine. If you haven't installed Docker yet, skip this section.

To uninstall Docker:

1. SSH into the machine.

2. Open a console window and run the following command:

   ```
   sudo dnf remove docker docker-client docker-client-latest docker-common
   docker-latest docker-latest-logrotate docker-logrotate docker-engine docker-
   ce docker-ce-cli containerd.io runc
   ```

3. When uninstalling is complete, proceed to Installing Docker.

Installing Docker

To install Docker:

If you do not have Docker installed, or if the Docker version is too low, install it using the following procedure:

1. SSH into the machine that will host the PCoIP Connection Manager and PCoIP Security Gateway.

2. Open a console window, and run the following command. This will remove the `podman` and `buildah` packages if they are installed (these packages conflict with Docker):

   ```
   sudo dnf remove podman buildah
   ```

3. Run the following commands in the same console window. Note that if you copy and paste these commands into the console, you may need to press `Enter` again to execute the last command:

```
sudo dnf install -y dnf-utils
sudo dnf config-manager --add-repo https://download.docker.com/linux/centos/
docker-ce.repo
sudo dnf install docker-ce docker-ce-cli containerd.io
```

4. Confirm installation:

```
sudo docker -v
```

Installing for Offline Environments

If the PCoIP Connection Manager and PCoIP Security Gateway machine does not have a connection to the public internet, you must create a temporary internet-connected machine to download the package files and dependencies and then transfer them to the production machine.

For information on package dependencies, see System Requirements.

Before You Begin

Before you proceed with installation, note the following:

- **Docker must be installed** before you begin. If you are able to open a temporary internet connection to your machine, you can use the instructions below. If you are not able to open a temporary internet connection, install Docker on the production machine using any acceptable method.
- The PCoIP Security Gateway is installed as part of this process. IPv6 connections are not supported in offline deployments.
- If your connection broker is configured to identify resources by host name, then DNS must be available and configured as follows:
    - Host names must be resolvable from the PCoIP Connection Manager server.
    - Host names must be resolvable from the PCoIP broker.
- If your environment has `podman` or `buildah` installed, uninstall them before proceeding.

    ```
    sudo dnf erase podman buildah -y
    ```

## Creating the Installation Bundle

First, you'll download the package and dependencies to a temporary machine, create an installation bundle, and then transfer the bundle to the production machine for installation.

To create the offline installation bundle:

1. Create a disposable internet-connected machine that is identical to the production machine. To compare the OS information, open a console on each machine and run the following command on both:
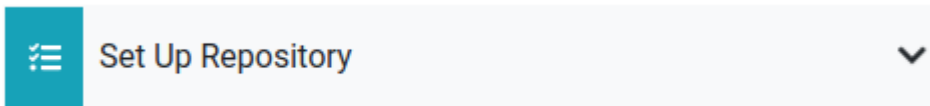
   ```
   cat /etc/os-release
   ```

2. On the temporary machine, open a browser and go to the PCoIP Connection Manager and PCoIP Security Gateway download page.

3. Click **Downloads and scripts**:
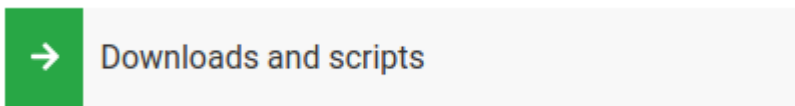
   

   If you see a login button instead, click it to log into the site and then proceed.

4. Accept the End User License Agreement, then click **Set Up Repository**:

   
   .

   The window will expand and show the setup scripts for each supported operating system. Copy the command for your system to the clipboard.

5. Open a console window and paste in the command you copied in the previous step. You may need to press Enter to execute it.

   The command fetches a configuration script from our servers and runs it locally, setting up and configuring the repository on the local machine.

6. Install **pcoip-cmsg-setup**

   ```
   sudo dnf install pcoip-cmsg-setup
   ```

7. Find and note the rpm name for the setup package. We will use this name when creating the offline bundle next.

   ```
   sudo dnf info pcoip-cmsg-setup
   ```

   The rpm name will similar to this: `pcoip-cmsg-setup-<version>-<release>` .

8. Create the offline install bundle:

```
sudo pcoip-cmsg-setup create-darksite-bundle --pcoip-cmsg-rpm-path <rpm name>
```

...where `<rpm name>` is the name you noted in the previous step.

The process will create a tarball called `teradici-pcoip-cmsg-bundle.tar.gz`.

9. Transfer the installation bundle to the production machine using any acceptable method, such as a USB flash drive or SCP.

Once this process has completed successfully, you can dispose of the temporary machine.

Installing PCoIP Connection Manager and the PCoIP Security Gateway

Once you have created the installation bundle and transferred it to the production machine, you can install the software.

To install the PCoIP Connection Manager and the PCoIP Security Gateway:

1. SSH into the production machine.

2. Navigate to the directory where you placed the installer bundle.

3. Extract the bundle and move into the newly-created `teradici-pcoip-cmsg-bundle` directory:

```
tar xzvf teradici-pcoip-cmsg-bundle.tar.gz
```

```
cd teradici-pcoip-cmsg-bundle
```

4. Install the PCoIP Connection Manager. The command depends on whether Docker is already installed on the production machine (if you are unsure, use `docker -v` to check your Docker version):

   • **If Docker is already installed**:

   ```
   sudo dnf install --allowerasing pcoip-cmsg-setup*.rpm --disablerepo="*" -y
   ```

- If Docker is *not* installed:

```
sudo dnf install --allowerasing --nobest --skip-broken --disablerepo="*"
-y *.rpm
```

5. Start the docker daemon

```
sudo systemctl start docker
```

6. Move back up one directory level and then install the PCoIP Connection Manager and PCoIP Security Gateway:

```
cd ..
sudo pcoip-cmsg-setup install --darksite-bundle-path teradici-pcoip-cmsg-
bundle <installation_flags>
```

> 🔥 **Important: Required installation flags**
>
> There are a number of options and settings available. You can invoke the `install` command with the `--help` flag to list them:
>
> ```
> pcoip-cmsg-setup install --help
> ```
>
> They are also listed in the next section.
>
> The `install` command will prompt you for required parameters that have not been supplied via flags.

Installation Flags and Options

The following flags can be used to provide values at the command line. Flags that are required are identified in the description.

Boolean values should be provided as either `true` or `false`, lowercased, as in this example:

```
--example-flag=true
```

Flag	Type	Description
`--accept-policies`	Boolean	Automatically accepts the EULA and Privacy Policy. **Required.**
`--broker-url`	String	The URL of the PCoIP Broker, specified either as a https://: or https://: or https://[].. **Required**.
`--ca-cert`	String	The full path and filename of the custom Certificate Authority's public certificate to be used in the PCoIP Connection Manager and PCoIP Security Gateway. **Required if** `--self-signed` **is not used.**
`--compose-file`	String	Specify the full path to a local docker-compose file.
`--darksite-bundle-path`	string	The path of darksite install bundle to be used for darksite installation
`--docker-password`	String	Password to login to private registry.
`--docker-registry`	String	Specifies the Teradici source for Cloud Access Connector images to be install from. **Debugging only**: This is intended to be used for debugging purposes and should not be used without guidance from Teradici support. Using this flag incorrectly can result in failed installations.
`--docker-username`	String	Username to login to private registry.
`--enable-collaboration`	Boolean	Allow multiple PCoIP clients to collaborate on a PCoIP agent. (default true)
`--enable-ipv6`	Boolean	Enables IPv6 connections (default false). To enable IPv6 use `--enable-ipv6=true`. To disable IPv6 use `--enable-ipv6=false`, or omit this flag.
`--external-pcoip-ip`	StringArray	Sets the public IP address of Security Gateway. If `--enable-ipv6` is true, this option may be used twice (once for IPv4 and once for IPv6). **Required if PCoIP Security Gateway is enabled**

Flag	Type	Description
`--enable-security-gateway`	Boolean	Enable and use the PCoIP Security Gateway (default true).
`--help`		Lists all available flags.
`--host-address`	stringArray	Sets the host FQDN/IP address. The option may be used twice (once for the IP address and once for the FQDN)
`--ignore-disk-req`	Boolean	Ignore the check for the minimum disk space requirement.
`--license-server-url`	String	The address of the locally installed PCoIP License Server. Example: `https://<license-server-address>:<port>`
`--self-signed`	Boolean	Automatically generate self-signed SSL cert and key for testing purposes. If specified, `--ssl-key` and `--ssl-cert` options are ignored.
`--ssl-cert`	String	The full path and filename of the SSL certificate to be used in the PCoIP Connection Manager and PCoIP Security Gateway. Required if `--self-signed` is not used.
`--ssl-key`	String	The full path and filename of the SSL key to be used in the PCoIP Connection Manager and PCoIP Security Gateway. Required if `--self-signed` is not used.

Enabling or Disabling the PCoIP Security Gateway

By default, the PCoIP Security Gateway is enabled when the package is installed. This configuration is highly recommended for deployments where users will connect over the WAN. If your users are behind a firewall and do not access their desktops from the WAN, you may not need the PCoIP Security Gateway.

If you are sure that you do not need the PCoIP Security Gateway, reinstall the package using the `--enable-security-gateway=false` flag.

To reenable the PCoIP Security Gateway, reinstall the package using the default options.

About Docker

The PCoIP Connection Manager and PCoIP Security Gateway depends on Docker 20.10.0 or higher, which must be installed on the machine before you install the PCoIP Connection Manager and PCoIP Security Gateway.

If you have not installed Docker yet, install it now.

If you are not sure if Docker is installed, or are not sure what Docker version you have, verify your Docker version first.

Verifying Docker Version

To verify your Docker installation and version:

1. SSH into the machine.

2. Open a console window and run the following command:

```
sudo docker -v
```

- If Docker is *not* installed, this command will produce an error. Installation instructions are provided in the next section.

- If you see a version number that is *lower* than 20.10.0, you must uninstall Docker and then re-install the supported version. Instructions for uninstalling and installing are provided in the next section.

- If you see a version number that is equal to or higher than 20.10.0, you have a compatible version of Docker already installed and can skip to PCoIP Connection Manager and PCoIP Security Gateway installation.

Uninstalling Docker

You'll only need to do this if you have an unsupported version of Docker already on the machine. If you haven't installed Docker yet, skip this section.

To uninstall Docker:

1. SSH into the machine.

2. Open a console window and run the following command:

```
sudo dnf remove docker docker-client docker-client-latest docker-common
docker-latest docker-latest-logrotate docker-logrotate docker-engine docker-
ce docker-ce-cli containerd.io runc
```

3. When uninstalling is complete, proceed to Installing Docker.

Installing Docker

To Install Docker:

> 🔥 **Important: This method requires a temporary internet connection**
>
> The method described here requires you to open a temporary connection to the public internet, and then close it after installation. If you cannot open a connection to the internet, you must move the docker installer and dependencies onto the production machine manually instead.

1. Open a temporary internet connection on the production machine.

2. SSH into the machine that will host the PCoIP Connection Manager and PCoIP Security Gateway.

3. Open a console window, and run the following command. This will remove the `podman` and `buildah` packages if they are installed (these packages conflict with Docker):

```
sudo dnf remove podman buildah
```

4. Run the following commands in the same console window. Note that if you copy and paste these commands into the console, you may need to press `Enter` again to execute the last command:

```
sudo dnf install -y dnf-utils
sudo dnf config-manager --add-repo https://download.docker.com/linux/centos/
docker-ce.repo
sudo dnf install docker-ce docker-ce-cli containerd.io
```

5. Confirm installation:

```
sudo docker -v
```

6. After confirming installation, close the internet connection.

Updating the PCoIP Connection Manager and PCoIP Security Gateway

The PCoIP Connection Manager can be deployed with or without the PCoIP Security Gateway, depending on your environment. The procedure for updating this component is different for both scenarios.

> 🔥 **Important: Powering off the PCoIP Connection Manager and PCoIP Security Gateway**
>
> When you power off the existing PCoIP Connection Manager and PCoIP Security Gateway, any PCoIP sessions that are active and using the security gateway will be dropped and will need to be re-established.

> ✏️ **Note: Load Balancer**
>
> If you have a load balancer in front of a group of PCoIP Connection Manager and PCoIP Security Gateway virtual machines, you can configure the load balancer to stop sending new connections to the PCoIP Connection Manager and PCoIP Security Gateway while you are updating.

Updating an Offline Installation

If your deployment is offline (dark site), use this procedure.

1. Create a new installation bundle .

2. Transfer the installation bundle to the production machine using any acceptable method, such as a USB flash drive or SCP.

3. Extract the bundle and move into the newly-created teradici-pcoip-cmsg-bundle directory:

```
tar xzvf teradici-pcoip-cmsg-bundle.tar.gz
cd teradici-pcoip-cmsg-bundle
```

4. Install the PCoIP Connection Manager

```
sudo dnf install --allowerasing pcoip-cmsg-setup*.rpm --disablerepo="*" -y
```

5. Move back up one directory level and then install the PCoIP Connection Manager and PCoIP Security Gateway:

```
cd ..
sudo pcoip-cmsg-setup install --darksite-bundle-path teradici-pcoip-cmsg-bundle <installation_flags>
```

Updating an Online Installation

To upgrade a PCoIP Connection Manager and PCoIP Security Gateway that can reach the public internet:

> **Important: Installation flags are required**
>
> If installation flags are absent, or are different from the original installation, the configuration on the new machine will be different.

1. Update the package:

```
dnf upgrade pcoip-cmsg-setup -y
```

2. Reinstall the package:

```
pcoip-cmsg-setup install <installation_flags>
```

To downgrade to an earlier version:

1. Downgrade the package:

```
dnf downgrade pcoip-cmsg-setup -y
```

2. Reinstall the package:

```
pcoip-cmsg-setup install <installation_flags>
```

Uninstalling PCoIP Connection Manager and PCoIP Security Gateway

If you want to remove the PCoIP Connection Manager and PCoIP Security Gateway completely from the production machine, open a console and run the following commands:

1. Close out running Docker containers:

```
sudo docker stack rm pcoipcm
sudo docker swarm leave --force
```

2. Remove Docker images

```
sudo docker rmi -f $(sudo docker images --format "{{.ID}} {{.Repository}}" |
grep -E */pcoip-cm | awk '{ print $1 }')
sudo docker rmi -f $(sudo docker images --format "{{.ID}} {{.Repository}}" |
grep -E */sg | awk '{ print $1 }')
```

3. Remove the setup files and repository information:

```
sudo dnf remove pcoip-cmsg-setup
sudo rm -f /etc/yum.repos.d/teradici-pcoip-cmsg.repo
```

4. Clean up Teradici files and directories:

```
sudo rm -rf /opt/teradici
sudo rm -rf /var/log/Teradici
```

5. Optionally remove Docker, if it will no longer be needed:

```
sudo docker system prune -f -a # remove all unused images
sudo systemctl stop docker # stop Docker
sudo systemctl disable docker # Prevent Docker from running on reboot
sudo dnf remove docker-ce docker-ce-cli containerd.io # uninstall Docker
Engine
```

6. Optionally remove the Docker repository:

```
sudo rm -f /etc/yum.repos.d/docker-ce.repo
```

Configuring the PCoIP Conenction Manager and PCoIP Security Gateway

You can configure the PCoIP Connection Manager and Security Gateway using the `pcoip-cmsg-setup configure` command.

The general syntax is:

```
sudo pcoip-cmsg-setup configure <flags>
```

For example, to specify a broker url, you would open a console window and enter the following:

```
sudo pcoip-cmsg-setup configure --broker-url https://<example>
```

Configuration Flags and Options

The following flags can be used to provide values at the command line.

Flag	Type	Description
`--broker-url`	String	The URL of the PCoIP Broker, specified either as a https://: or https://: or https://[]:. **Required**.
`--clear-host-address`	Boolean	Clears the host address.
`--ca-cert`	String	The full path and filename of the custom Certificate Authority's public certificate to be used in the PCoIP Connection Manager and PCoIP Security Gateway. **Required if** `--self-signed` **is not used.**
`--clear-trusted-license`	Boolean	Clears trusted license certificate and key.
`--compose-file`	String	Specify the full path to a local docker-compose file.

Flag	Type	Description
`--docker-password`	String	Password to login to private registry.
`--docker-registry`	String	Specifies the Teradici source for Cloud Access Connector images to be install from. **Debugging only**: This is intended to be used for debugging purposes and should not be used without guidance from Teradici support. Using this flag incorrectly can result in failed installations.
`--docker-username`	String	Username to login to private registry.
`--enable-collaboration`	Boolean	Allow multiple PCoIP clients to collaborate on a PCoIP agent. (default true)
`--external-pcoip-ip`	StringArray	Sets the public IP addresses of VM which hosts Security Gateway. This option can be used twice, once for IPv4 and once for IPv6 (if using). **Required if PCoIP Security Gateway is enabled**.
`--help`		Display configuration help.
`--host-address`	stringArray	Sets the host FQDN/IP address. The option may be used twice (once for the IP address and once for the FQDN)
`--license-server-url`	String	The address of the locally installed PCoIP License Server. Example: `https://<license-server-address>:<port>`
`--ssl-cert`	String	The full path and filename of the SSL certificate to be used in the PCoIP Connection Manager and PCoIP Security Gateway. **Required if** `--self-signed` **is not used**.
`--ssl-key`	String	The full path and filename of the SSL key to be used in the PCoIP Connection Manager and PCoIP Security Gateway. **Required if** `--self-signed` **is not used**.
`--trusted-license-cert`	String	Trusted Customer License certificate path. Defaults to `/opt/teradici/pcoipcm_data/certs/tcl-cert.crt`).

Flag	Type	Description
`--trusted-license-cert-key`	String	Trusted Customer License certificate key path. Defaults to `/opt/teradici/pcoipcm_data/certs/tcl-cert.key`.

Security Considerations

All certificate files must be in base64-encoded PEM format.

> ⚡ **Follow your organisation's security policy**
>
> For all security and certficate procedures, ensure you follow your organisation's security policy.

Creating, Installing, and Managing Certificates

In order to establish secure TLS connections with clients, certificates must be configured for the PCoIP Connection Manager and the PCoIP Security Gateway. If the required certificate files are not present or they are improperly configured, clients will not be able to connect and users will not be able to establish PCoIP sessions.

Only certificates with RSA private keys having at least 2,048-bit length are supported. RSA private keys having at least 3,072-bit length are recommended. Certificates with DSA private keys are not supported. Certificates that include an MD5-based digital signature algorithm are not supported.

Both the PCoIP Connection Manager and PCoIP Security Gateway support wildcard certificates which can be used on multiple PCoIP Connection Manager and PCoIP Security Gateway servers.

If you are ready to replace your default self-signed certificates with your own signed certificates, proceed to Signed Certificates for Production.

> ⚡ **Ensure all certificate files follow your security policy**
>
> Protect the regenerated certificate and ensure all certificate files you use conform to your organization's security policy.

Default Certificate

The PCoIP Connection Manager and PCoIP Security Gateway installation script generates a self-signed certificate during installation to facilitate testing. **This should be replaced with your own certificate, signed by a trusted Certificate Authority (CA), when deploying a production system**.

By default, both the PCoIP Connection Manager and the PCoIP Security Gateway use the same private key and signed certificate; if your security policy requires it, each service can use its own key/certificate pair instead. If two sets of certificates are required, follow these procedures twice to generate two key/certificate pairs and configure the PCoIP Security Gateway appropriately.

> ✏️ **Copying certificates from a Window system to a Linux system**
>
> When copying certificates from a Windows system to a Linux system, line endings might be incorrect. Check that the certificate text is formatted correctly.

Signed Certificates for Production

Production systems should use your own certificates, signed by a trusted certificate authority (CA). The following sections describe the process of creating, signing, and installing certificates.

At a high level, the process is:

1. Generate a new private key and certificate signing request.

2. Submit the CSR to a trusted certificate authority (CA) for signing, either internal or third-party.

3. Verify and convert the resulting certificate files to the `.pem` format.

4. Install the certificates on the PCoIP Connection Manager and Security Gateway machine, restart both services, and inspect their log files to verify that the certificates are working and that all services have started.

5. Protect the certificate files and access.

> ⚡ **Danger: These instructions are examples**
>
> The following procedures are working examples. Before following them, you should be sure they conform to your organization's security policies. Modify these procedures as necessary.

> ℹ️ **These examples use openssl**
>
> The following procedures use openssl to create and manage certificates. If you use another tool, adapt these instructions accordingly.

Creating Certificate Files

First, generate a new private key and CSR (certificate signing request).

To generate a private key and CSR:

1. On the PCoIP Connection Manager server, open a command prompt.

2. Create a temporary directory to store the certificate and move into it.

   This example uses `~/certs`, which creates a `certs` directory under your home directory, but you can create it anywhere you like:

   ```
   mkdir ~/certs
   cd ~/certs
   ```

3. Generate a private key file and CSR according to your organization's security policy.

   This example creates an RSA 3072-bit private key and a CSR requesting a sha384 hash algorithm. The private key file is `private.key` and the CSR file is `server.csr`.

   ```
   openssl req -new -newkey rsa:3072 -sha384 -nodes –keyout private.key -out
   server.csr
   ```

   When running this command, you will be prompted for information to be displayed in the certificate.

Distinguished Name Field	Description	Example
Country Name	The two-letter ISO abbreviation for your country	CA for Canada
State or Province Name	The unabbreviated name of the state or province where your organization is legally located.	British Columbia
Locality Name	The city where your organization is legally located.	Burnaby

Distinguished Name Field	Description	Example
Organization Name	The full legal name of your organization. Cannot use < > ~ ! @ # $ % ^ * / \ ( ) ? . , &	Teradici Corporation
Organization Unit Name	Department of your organization. Cannot use < > ~ ! @ # $ % ^ * / \ ( ) ? . , &	Global Support Services
Common Name	The fully qualified domain name (FQDN) of your server. This must be an exact match or, in the case of a wild card, an asterisk (*) before the domain.	If your PCoIP Connection Manager address is teradiciplatform.teradici.com then the CSR must have the common name teradiciplatform.teradici.com. If you plan on having a wildcard certificate for use on multiple PCoIP Connection Manager servers, then prefix the domain with an asterisk (*). In this example: *.teradici.com.
Email Address	Leave blank	
A challenge password	Leave blank	
An optional company name	Leave blank	

You should now have two files in your `~/certs` folder; `private.key` and `server.csr`.

You can verify the details of the CSR request using the following command:

```
openssl req -noout -text -in ~/certs/server.csr
```

Obtaining the Signed Public Key Certificate

Next, use your CSR request to obtain a public signed certificate. Submit `server.csr` to a trusted CA following your organization's security policy. Follow the CA's instructions to obtain the public signed certificate.

If your CA offers the public signed certificate both with and without the certificate chain, download both. If they do not offer a certificate file including the certificate chain, refer to your CA's documentation on how to build it.

When you have received the signed files, copy them into your working directory (`~/certs`).

Verifying and Converting Certificate File Format

Before installing your certificate, you must verify that it's in the correct format and convert it to .

These instructions assume the following:

- You have copied the files received from the CA to `~/certs`.
- The public certificate signed by the CA *without* the certificate chain is named `certificate.crt`.
- The public certificate signed by the CA *with* the certificate chain (intermediary and root certificates) is named `CAcertificate.crt`.

To verify the certificate file format:

Verify the `certificate.crt` file:

```
openssl x509 -in certificate.crt -text -noout
```

- If you don't see any errors, change the file extension from `.crt` to `.pem`:

  ```
  mv certificate.crt certificate.pem
  ```

- If you DO see errors, open the certificate file in a text editor and verify the following:

  - There are no extra characters at the end of lines
  - The file starts with `-----BEGIN CERTIFICATE-----`
  - The file ends with `-----END CERTIFICATE-----`

  If the file doesn't begin and end with the required lines, it's in the wrong format. Convert it to PEM:

  ```
  openssl x509 –inform der –in certificate.crt –out certificate.pem
  ```

Verify the newly renamed file:

```
openssl x509 —in certificate.pem —text —noout
```

Repeat these steps for `CAcertificate.crt` (the certificate that includes the certificate chain).

When you are done, you should have two `.pem` files and one private key file in the `~/certs` directory:

File	Explanation
`private.key`	Contains the certificate's private key.
`certificate.pem`	Contains a public certificate signed by a CA without the certificate chain. This is presented to PCoIP clients when they connect to the PCoIP Connection Manager during authentication and resource allocation.
`CAcertificate.pem`	Contains the certificate chain, including any intermediate and root certificate. Self-signed certificates do not have any root or intermediate certificate.

> **Important: Back up your certificate and private key**
>
> Back up the private key and certificate in a secure location according to your organization's security policy.

Using a PCoIP License Server with the Connection Manager

Using a PCoIP License Server with the PCoIP Connection Manager

In most cases, PCoIP licenses are validated automatically using Teradici's Cloud Licensing Service. In deployments where PCoIP agents cannot reach the public internet, a PCoIP License Server can be used to handle license validation instead. PCoIP License servers can be hosted on-premises or in any public or private cloud environment.

To use the PCoIP Connection Manager with a PCoIP License Server, you must configure the PCoIP Connection Manager with the address of the license server and the address of the connection broker.

Use the `install` and `configure` commands to configure the PCoIP License Server information:

```
pcoip-cmsg-setup install --license-server-url  https://<license-server-
address>:<port>
pcoip-cmsg-setup configure --license-server-url  https://<license-server-
address>:<port>
```

For more information about the PCoIP License Server, see the following guides:

- PCoIP License Server Administrators' Guide (Online deployments)

- PCoIP License Server Administrators' Guide (Offline deployments)

PCoIP Connection Manager and Security Gateway RPM Package Contents

The following table shows the files installed by the RPM packages:

Folder or file path	Type	Description
/sbin/pcoip-cmsg-setup	File	binary installation file
/usr/local/teradici/conf/docker-compose.yaml	File	docker-compose template file
/usr/local/teradici/conf/docker-compose-ipv6.yaml	File	docker-compose file for IPv6
/usr/local/teradici/conf/	Folder	configuration files templates
/usr/local/teradici/licenses/	Folder	license information files

After installation, the following files and folders are present:

Folder or file path	Type	Description
/var/log/Teradici/pcoip-cmsg-setup/	Folder	`pcoip-cmsg-setup` installation log files
/var/lib/docker/containers	Folder	All docker-related folders and files
/opt/teradici/pcoipcm_data/docker-compose.yaml	File	Template file used to generate docker containers
/opt/teradici/pcoipcm_data/certs/	Folder	Required certificates
/opt/teradici/pcoipcm_data/data/	Folder	
/opt/teradici/pcoipcm_data/data/SecurityGateway.conf	File	Security Gateway configuration file

Folder or file path	Type	Description
/opt/teradici/pcoipcm_data/data/config.properties	File	Connection Manager configuration file
/opt/teradici/pcoipcm_data/data/cmsg.env	File	Environment file that configures shared data for both Connection Manager and Security Gateway

TLS Cipher Suites

This page contains information about the TLS Cipher Suites used by the PCoIP Connection Manager and PCoIP Security Gateway, and instructions for restricting the full list to subsets if desired.

TLS Versions

THe PCoIP Connection Manager and PCoIP Security Gateway support TLS 1.2 and TLS 1.3.

PCoIP Connection Manager TLS Cipher Suites

The PCoIP Connection Manager supports the following cipher suites for the TLS connections from the PCoIP client, to the connection broker, and to the PCoIP Agent (in decreasing order of preference):

- TLS_AES_256_GCM_SHA384

- TLS_CHACHA20_POLY1305_SHA256

- TLS_AES_128_GCM_SHA256

- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256

- TLS_RSA_WITH_AES_256_GCM_SHA384

- TLS_RSA_WITH_AES_128_GCM_SHA256

PCoIP Security Gateway Supported TLS Cipher Suites

The PCoIP Security Gateway supports the following cipher suites for TLS connections, in decreasing order of preference:

- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256

- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384

- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256

- TLS_RSA_WITH_AES_256_GCM_SHA384

- TLS_RSA_WITH_AES_128_GCM_SHA256

- TLS_RSA_WITH_AES_256_CBC_SHA256

- TLS_RSA_WITH_AES_128_CBC_SHA256

- TLS_RSA_WITH_AES_256_CBC_SHA

- TLS_RSA_WITH_AES_128_CBC_SHA

Troubleshooting Connectivity Issues

Network Connectivity Problems

Connectivity issues are often caused by firewall misconfiguration. If you are unable to establish PCoIP connections, verify that sent packets are actually being received at the intended destination.

Useful tools for troubleshooting this type of issue are **ssldump** or **tcpdump** (for Linux), or **Wireshark** (for Windows).

The network connections between the following endpoints all need to be operational for a PCoIP session to be successful.

Connection	Port	Source
PCoIP Connection Manager	443 TCP	PCoIP Client
Connection Broker	443 (configurable) TCP	PCoIP Connection Manager
PCoIP Agent	60443 TCP	PCoIP Connection Manager
When Security Gateway is enabled		
PCoIP Security Gateway	4172 TCP/UDP	PCoIP Client
PCoIP Agent	4172 TCP/UDP	PCoIP Security Gateway
When Security Gateway is disabled (Direct Connection)		
PCoIP Agent	4172 UDP/TCP	PCoIP Client

Methods for testing communication between components on required ports are given next:

- PCoIP Client to PCoIP Connection Manager

- PCoIP Connection Manager to Connection Broker

- PCoIP Connection Manager to PCoIP Agent

- PCoIP Client to PCoIP Security Gateway

- PCoIP Security Gateway from PCoIP Client (UDP)

- PCoIP Agent from PCoIP Client (UDP)

Methods are also given for verifying the component availability:

- Verifying PCoIP Agent availability

- Verifying Connection Broker availability

- Verifying PCoIP Connection Manager Web Application Availability

Connectivity from PCoIP Client to PCoIP Connection Manager

This check looks for traffic between the PCoIP Client and the PCoIP Connection Manager on TLS port 443.

To verify connectivity from a PCoIP Client to PCoIP Connection Manager:

1. On the server hosting the PCoIP Connection Manager, start `ssldump`:

   ```
   sudo ssldump -i <interface> host <client-ip-address> port 443
   ```

2. From the client, connect to the PCoIP Connection Manager.

3. In the ssldump output, look for packets originating from the PCoIP Client.

Connectivity from PCoIP Connection Manager to Connection Broker

This check looks for traffic between the PCoIP Connection Manager and a broker on TLS port 443.

To verify connectivity from the PCoIP Connection Manager to a connection broker connectivity:

1. On the server hosting the connection broker, use **ssldump** or **Wireshark** to capture packets from the PCoIP Connection Manager on TLS port 443.

2. From the client, connect to the PCoIP Connection Manager.

3. Try to authenticate.

4. Verify from ssldump or Wireshark output that the connection broker is receiving data from the PCoIP Connection Manager.

Connectivity from PCoIP Connection Manager to PCoIP Agent

This check looks for traffic between the PCoIP Connection Manager and a PCoIP Agent on TLS port 60443.

To verify PCoIP Connection Manager to agent collectivity:

1. On the PCoIP agent machine, use ssldump or Wireshark to capture packets from the PCoIP Connection Manager on TLS port 60443.

2. From the client, connect to the PCoIP Connection Manager.

3. Try to authenticate and establish a session.

4. Select a resource and connect.

5. Verify from ssldump or Wireshark output that the PCoIP agent is receiving data from the PCoIP Connection Manager.

Connectivity from PCoIP Client to PCoIP Security Gateway

This check looks for traffic between the PCoIP client and the PCoIP Security Gateway on TLS port 4172.

To verify that the PCoIP Security Gateway server is receiving session initiation data from the PCoIP client:

1. On the server hosting the PCoIP Security Gateway, start ssldump:

```
sudo ssldump -i <interface> host [client-ip-address] and port 4172
```

2. From the client, connect to the PCoIP Connection Manager.

3. Try to authenticate and establish a session.

4. Select a resource and connect.

5. Verify from ssldump output that the PCoIP Security Gateway is receiving data from the client.

> ✏️ **Note: Firewall must allow traffic on UDP:4172**
>
> If the firewall is configured to enable TCP traffic over port 4172 but not UDP traffic, then the ssldump output will show packets but you won't be able to establish a PCoIP session.

Connectivity (UDP) to PCoIP Security Gateway from PCoIP Client

This check looks for UDP traffic between the PCoIP Security Gatway and the PCoIP client on TLS port 4172.

To verify that the PCoIP Security Gateway is receiving UDP traffic from the PCoIP client:

1. On the server hosting the PCoIP Security Gateway, start tcpdump:

   ```
   sudo tcpdump -i <interface> host [client-ip-address] and -n udp port 4172
   ```

2. From the client, connect to the PCoIP Connection Manager.

3. Try to authenticate and establish a session.

4. Select a resource and connect.

5. Verify from ssldump output that the PCoIP Security Gateway is receiving data from the client.

Connectivity (UDP) to PCoIP Agent from PCoIP Client

This check looks for UDP traffic between the PCoIP Security Gatway and the PCoIP client on TLS port 4172.

To verify that the PCoIP server is receiving UDP traffic from the PCoIP client:

1. On the server hosting the PCoIP server, start tcpdump:

   ```
   sudo tcpdump -i <interface> host [server-ip-address] and -n udp port 4172
   ```

2. From the client, connect to the PCoIP Connection Manager.

3. Try to authenticate and establish a session.

4. Select a resource and connect.

5. Verify from ssldump output that the PCoIP server is receiving data from the client.

Verifying PCoIP Agent Availability

Ensure your DNS is configured correctly, then verify you can establish and maintain a connection to the agent.

For each virtual desktop host in your deployment or RDS farm, verify that you can establish TLS connections from the server hosting the PCoIP Connection Manager to the PCoIP agent listening on ports 4172 and 60443:

```
openssl s_client -connect <host-ip-address>:4172
openssl s_client -connect <host-ip-address>:60443
```

Verifying Connection Broker Availability

If you are using a connection broker and the firewall is configured correctly, then verify you can establish a TLS connection from the server hosting the PCoIP Connection Manager to the connection broker listening on port 443:

```
openssl s_client -connect <broker-ip-address>:443
```

Verifying PCoIP Connection Manager and Security Gateway Status

To verify the PCoIP Connection Manager and its components, SSH into the PCoIP Connection Manager machine.

- List the running services in Docker: to verify that the `pcoipcm_cm` and `pcoipcm_sg` services are running:

  ```
  docker service ls
  ```

  You should see the `pcoipcm_cm` and `pcoipcm_sg` services in the response, similar to the following example:

  ```
  ID         NAME        MODE        REPLICAS
  IMAGE                                        PORTS
  34iefjidf  pcoipcm_cm  replicated     1/1      cloudaccessmanager.azurecr.io/
  pcoip-cm:5-release
  ```

```
54ibvbbdt  pcoipcm_sg  replicated      1/1      cloudaccessmanager.azurecr.io/
sg:3-release
```

Verifying PCoIP Connection Manager Web Application Status:

1. Establish a TLS connection using the openssl `s_client` command:

```
openssl s_client -connect 127.0.0.1:443
```

2. When the SSL connection is established, copy and paste the following text to issue a dummy HTTP POST command:

```
POST /pcoip-broker/xml HTTP/1.1
Host: localhost
Content-type: text/xml; charset=UTF-8
Content-Length: 39
<?xml version="1.0" encoding="UTF-8"?>
```

- If the PCoIP Connection Manager is operational, you will receive an XML response containing an `<error-resp>` element.
- If the PCoIP Connection Manager **is not** operational, check the logs of the PCoIP Connection Manager and PCoIP Security Gateway containers:

```
docker logs <CONTAINER_ID>
```

Troubleshooting Certificate Errors

> ✏️ **Error messages may be caused by different issues**
>
> Errors discussed here might not be caused by certificate problems.

Error messages

Failed to get broker / agent response due to: read ECONNRESET

If the broker or agent certificate key bit length is less than 2048, the PCoIP Connection Manager may not be able to establish a connection. Ensure that all components are using certificates of 2048 bits or more.

Troubleshooting Error Messages

Some common PCoIP client error messages and their possible causes are listed here.

Error occurred while communicating with broker

Possible cause	Resolution
The connection broker may be down or unreachable	Ensure the broker server is up and the broker service is running
	Ensure the broker server is resolvable by DNS

Timeout occurred while communicating with broker

Possible cause	Resolution
The port which the connection broker listens on may be blocked	Ensure the port which the broker server listen on is not blocked by firewall
	Ensure the broker server is funcational

Error occurred while communicating with agent

Possible cause	Resolution
The PCoIP agent may be down or or unreachable.	Ensure the host is up and the agent service is running.
	Ensure the host is resolvable by DNS

Timeout occurred while communicating with agent

Possible causes:

Possible cause	Resolution
The port which the PCoIP agent linsten on may be blocked.	Ensure the port of the PCoIP agent listen on is not blocked by firewall

PCoIP Connection Manager and Security Gateway Log Files

Each PCoIP component logs its activities and stores the log files locally. Troubleshooting behavior problems usually begins with an examination of PCoIP log files for error conditions or other system health indicators.

All PCoIP components use an identical, session-specific ID in their respective log files, allowing you to separate individual sessions and aggregate messages from multiple components within a session. The session ID is a 36-character hexadecimal string.

Log Maintenance

Logs and log rotation for both the PCoIP Connection Manager and PCoIP Security Gateway are managed automatically by Docker.

Sensitive Information in Logs

Sensitive information such as passwords, session cookies, and other session data that can potentially be used to gain unauthorized access is either obscured or not logged. Non-sensitive, unique session identifiers such as user names and IP addresses are logged as these often help with troubleshooting.

Log File Locations

Docker stores logs from its containers in `/var/lib/docker/containers` . You can check logs based on the CONTAINER_ID for the PCoIP Connection Manager and PCoIP Security Gateway.

If the PCoIP Connection Manager and PCoIP Security Gateway is running, you can also use the following command to check logs:

```
docker logs <CMSG_CONTAINER_ID>
```

...where `<CMSG_CONTAINER_ID>` is the container ID for the PCoIP Connection Manager and PCoIP Security Gateway.

Log Verbosity

PCoIP logs can capture log messages at several different verbosity levels, ranging from highly verbose informational messages to error-only reporting.

Teradici recommends using the default verbosity log level in production deployments. When troubleshooting a problem, Teradici might recommend changing the log level for specific components to obtain more information from parts of the system.

> 🔥 **Security Gateway log levels cannot be changed**
>
> The log levels for the Security Gateway are not configurable.

> ✏️ **Note: Increasing verbosity will reduce history**
>
> Increasing log verbosity will generate more and larger log files, which will then reach the system limits and be aged out more quickly.

Changing the PCoIP Connection Manager Log Level

The available log levels for the PCoIP Connection Manager, from most verbose to least verbose, are:

- debug
- info
- warn
- error

To configure the log level of the PCoIP Connection Manager:

1. Open `/opt/teradici/pcoipcm_data/docker-compose.yaml` in a text editor.

2. Add or modify the `LOG_LEVEL` value as an environment variable under the Connection Manager service:

```
LOG_LEVEL = <log level value>
```

Where `<log level value>` is one of `debug`, `info`, `warn`, or `error`.

3. Use the `config` command to apply changes:

```
pcoip-cmsg-setup config --compose-file /opt/teradici/pcoipcm_data/docker-compose.yaml
```

Contacting Support

If you encounter any problems installing, configuring, or running the PCoIP Connection Manager and PCoIP Security Gateway, you can create a support ticket with Teradici.

Before creating a ticket, be prepared with the following:

- A detailed description of the problem

- Your PCoIP Connection Manager and PCoIP Security Gateway version number. You can find this by opening a console window and running the command:

  ```
  pcoip-cmsg-setup --version
  ```

- A support bundle, which contains log files and other diagnostic information we can use to help solve the problem. See Generating a Support Bundle for more information.

The Teradici Community Forum

The PCoIP Community Forum enables users to have conversations with other IT professionals to learn how they resolved issues, find answers to common questions, have peer group discussions on various topics, and access the Teradici PCoIP Technical Support Service team. Teradici staff are heavily involved in the forums.

To visit the Teradici community, go to the Teradici Knowledge Center.

Generating a Support Bundle

Teradici may request a support bundle from your system in order to troubleshoot and diagnose PCoIP issues. The support file is an archive containing logs and other diagnostic data that can help support diagnose your problem.

To generate the support bundle:

1. Open a console window

2. Run the following command:

```
pcoip-cmsg-setup diagnose --support-bundle
```