Teradici PCoIP Graphics Agent for Linux 21.10 Beta

This guide is intended for administrators who are deploying the Graphics Agent for Linux as part of the Teradici Cloud Access Software. It assumes thorough knowledge of Linux conventions and networking concepts, including firewall configuration.

> **Warning: This is a beta release**
>
> **Graphics Agent for Linux 21.10 is a beta release**, provided for user testing and feedback. It may contain bugs and may be unstable. Features in this beta release may not be present in the stable release when it is issued.
>
> **Beta releases should not be used in production systems.**

About the Graphics Agent for Linux 21.10 Beta

The Graphics Agent for Linux is part of Teradici Cloud Access Software. It enables Teradici customers to deliver GPU-powered physical and virtual workstations to end users via remote clients.

Users include high-end knowledge workers, graphic designers, artists, and CAD/CAM designers.

A deployed Graphics Agent for Linux requires these components:

- **A host machine** which provides the desktop to remote clients. The host must be a virtual machine in a data center or in the cloud. See System Requirements for more information.
- **The agent software** installed on the host machine.
- **A GPU**. The PCoIP Graphics Agent requires an NVIDIA GRID card.

  The GPU can be dedicated to the workstation, or shared among virtual workstations by a hypervisor. See System Requirements for more information.

© 2021 Teradici

Where to Find Information about Other Components

BETA

This guide describes the Graphics Agent for Linux.

For complete information about all of the components used in PCoIP ecosystems, including architectural diagrams and deployment suggestions, see one of the following documents:

Cloud Access Software architectures and descriptions:

- Teradici All Access Architecture Guide

For more information about PCoIP clients, see one of the following:

- Teradici PCoIP Software Client 21.10 for Windows Administrators' Guide
- Teradici PCoIP Software Client 21.10 for macOS Administrators' Guide
- Teradici PCoIP Software Client for 21.10 Linux Administrators' Guide
- Tera2 PCoIP Zero Client 21.10 Administrators' Guide

For information about Cloud Access licensing, see our Licensing FAQ. Most PCoIP systems use PCoIP Cloud Licensing. For systems using a local PCoIP License server instead, refer to the following guides:

- Teradici PCoIP License Server Administrators' Guide for ***Online Environments***
- Teradici PCoIP License Server Administrators' Guide for ***Offline Environments***

© 2021 Teradici

What's New in This Release

Release 21.10 of the Graphics Agent for Linux includes:

- Bug fixes and security updates.
- Additional documentation will be added soon!

BETA

System Requirements

The Graphics Agent for Linux depends on the following system capacities and capabilities:

Compatible Host Instance Types

Environment	Supported instance types
VMware ESXi (6.0+)	VMware Hardware Version 11
KVM (5.0+)	QEMU/KVM
Physical machines	Linux physical machine [1]
AWS EC2	• G3 • G4
Microsoft Azure (Generation 1 VMs)	• NV-series • NVv3-Series
Google Cloud Platform	Any instance with a supported GPU

Host Instance Requirements

Global instance requirements	
Operating Systems	• Ubuntu 18.04 LTS • RHEL/CentOS 7.7, 7.8, 7.9, 8.3, 8.4 ***(RHEL 8.x support requires agent 21.07.2 or higher)***

© 2021 Teradici

Global instance requirements	
Remote Host Memory	At least *2GB* of RAM is required on the host desktop. The agent should have at least *512MB* of available memory.
Remote Host GPU Memory	At least 1GB *per 4K UHD display*
Remote Host CPUs	At least 2 CPUs are required on the host desktop. Processors must support Streaming SIMD Extensions (SSE) 4.2. To use PCoIP Ultra, processors must support the AVX2 instruction set.
Network Ports	The following ports must be open on the host desktop: • TCP 443 • TCP 4172 • UDP 4172 • TCP 60443
Storage	At least 100MB for installation and 100MB for logging are recommended.
User	Cannot be root. You must create a user account for PCoIP connections.

Supported GPUs

Instance	GPU	Driver	Driver download (Ubuntu)	Driver download (RHEL)
VMware ESXi	GPUs as listed in NVIDIA's GRID release notes.	GRID 12.2 (460.73.01)+	Ubuntu Instructions	RHEL Instructions
KVM	GPUs as listed in NVIDIA's GRID release notes.	GRID 12.2 (460.73.01)+	Not supported	RHEL Instructions

BETA

Instance	GPU	Driver	Driver download (Ubuntu)	Driver download (RHEL)
Physical PC	• Tesla (any) • Quadro (k2000+)	Use NVIDIA's recommended driver.	Ubuntu Instructions	RHEL Instructions
AWS EC2 G3	NVIDIA M60	GRID 12.2 (460.73.01)+	Ubuntu Instructions	RHEL Instructions
AWS EC2 G4dn	NVIDIA T4	GRID 12.2 (460.73.01)+	Ubuntu Instructions	RHEL Instructions
Microsoft Azure	NVIDIA M60	GRID 12.2 (460.73.01)+	Ubuntu Instructions	RHEL Instructions
Google Cloud Platform	NVIDIA P4, P100, T4	GRID 12.2 (460.73.01)+	Ubuntu Instructions	RHEL Instructions

Info: Display limitations on Microsoft Azure

Graphics Agents running on Microsoft Azure may be limited to a single 2560x1600 display, depending on the state of NVIDIA GRID licensing.

Note: NVIDIA Tesla cards have license limitations

Systems using NVIDIA Tesla-based GPUs, including M60, M10, and M6, are subject to licensing restrictions which may further reduce the available number of monitors or their available resolutions. For systems using Tesla-based GPUs, only licensed displays and resolutions will use GRID functionality. For more information, see NVIDIA's Grid Licensing User Guide

1. Installation on physical machines requires some additional configuration. See the extra installation instructions for Ubuntu and RHEL/CentOS.

BETA

Audio Support

The Graphics Agent for Linux supports audio input and output between the host and the client. Audio can be enabled or disabled and audio bandwidth can be throttled by configuring the agent.

BETA

Supported Displays

The Graphics Agent for Linux, in combination with a GPU, supports a maximum of four displays on the PCoIP client and a maximum resolution of 4K UHD (3840×2160).

> **Important: Display acceleration requires a supported GPU**
>
> The Graphics Agent for Linux must be deployed with one or more supported GPUs. GPUs may have their own licensing requirements.

PCoIP agents support any of these monitor configurations:

- Vertical line
- Horizontal line
- Box display

Monitors can be used in any standard rotation (0°, 90°, 180°, or 270°). Any monitor can be the primary.

The Graphics Agent for Linux, in combination with one or more properly configured *and licensed* GPUs, provides the following benefits:

- **Display resolutions**: The Graphics Agent for Linux can provide any resolution a client asks for up to 4K UHD. The practical number of displays available at any given resolution is determined by the GPU, host system hardware, GPU profile licensing, and the connecting client's version.
- **Hot-pluggable displays from Tera2 PCoIP Zero Clients**: Displays may be added and removed as needed during an active PCoIP session when using a Tera2 PCoIP Zero Client.
- **3D application support**: Full-screen 3D applications are supported.

> **Note: Using multiple high-resolution displays**
>
> Systems with multiple high-resolution displays, such as quad 4K UHD topologies, require powerful system infrastructure. Be sure to use a system with sufficient bandwidth, client and host capabilities, and an appropriate number of licensed GPUs to support your required display topology.

© 2021 Teradici

Important: Attaching monitors to the host machine in not supported

PCoIP client supports a maximum of four displays. Attaching extra monitors to the host machine will conflict with client display topologies.

BETA

PCoIP Ultra

The Graphics Agent for Linux provides support for PCoIP Ultra, the latest protocol enhancements from Teradici. PCoIP Ultra is optimized for truly lossless support with bit-exact color accuracy and preservation of content detail at the highest frame rates.

PCoIP Ultra protocol enhancements propels our industry-recognized performance into the future of remote computing, with faster, more interactive experience for users of remote workstations working with high-resolution content.

PCoIP Ultra enhancements are disabled by default. You must enable them manually.

PCoIP Ultra is appropriate for specific use cases

For most users, the default PCoIP protocol will provide the best possible experience. Carefully review the recommended use cases in the next section to determine whether you should enable it.

For additional detail on PCoIP Ultra technical requirements for various use cases and troubleshooting steps, refer to KB 2109: PCoIP Ultra Troubleshooting.

When to Enable PCoIP Ultra

PCoIP Ultra is apporpriate for users with the following requirements:

Auto Offload: Achieves the best balance between color accuracy and network efficiency. This setting is appropriate for work-from-home or WAN content creators who require optimized delivery of high resolution content, including video playback, while still achieving build-to-lossless color accuracy.

CPU Offload: Provides efficient scaling across multicore CPUs, leveraging AVX2 instruction sets. Appropriate for users that require CPU-optimized delivery of 4K UHD, high-framerate video playback and build-to-lossless color accuracy. It is also useful when GPU encoding resources must be reserved for video encoding applications, typically in LAN environments.

GPU Offload: PCoIP encoding is always offloaded to a GPU. Appropriate for users who demand the highest possible CPU efficiency.

© 2021 Teradici

For ***all other scenarios***, Teradici recommends that you leave PCoIP Ultra disabled.

Requirements

To take advantage of PCoIP Ultra, you need:

- A **PCoIP agent** (any type), 21.10 or later
- A **PCoIP Software Client** (any type), 21.10 or later

> **PCoIP Tera2 Zero Clients do not support PCoIP Ultra**
>
> PCoIP Ultra is only available through PCoIP Software Clients.

- **CPU offload** requires CPU support for the AVX2 instruction set on both the agent and client machines.
- **GPU offload** requires an NVIDIA graphics card that supports NVENC on the agent machine.

Enabling PCoIP Ultra

To enable PCoIP Ultra features, edit the **`pcoip-agent.conf`** file and set the `pcoip.ultra` configuration setting as required:

- 1 To turn on ***PCoIP Ultra CPU Offload***. CPU offload requires CPU support for the AVX2 instruction set on both the remote host and client. The PCoIP Zero client is not supported. CPU offload is recommended for 4K UHD resolutions with video playback requirements of 30 fps (or more), and the highest possible image quality and color accuracy.
- 2 To turn on ***PCoIP Ultra GPU Offload*** - GPU offload requires an NVIDIA graphics card on the remote host capable of NVENC. GPU Offload is recommended when the CPU impact of pixel encoding should be minimized.
- 3 To use ***PCoIP Ultra Auto Offload*** This setting allows PCoIP to automatically switch between CPU and GPU Offload modes; CPU offload is used by default to provide the best image fidelity, and GPU offload is used during periods of high display activity to provide improved frame rates and bandwidth optimization.

BETA

This setting is only effective if the remote host and client endpoints are capable of both CPU and GPU Offload.

All PCoIP Ultra settings take effect on the next PCoIP session. No configuration is required on the PCoIP Software Client.

Turning PCoIP Ultra off

To disable PCoIP Ultra and use the default PCoIP experience instead, set `pcoip.ultra` to 0.

Setting configuration values

If you don't know how to set PCoIP agent configuration values, refer to Configuring the Graphics Agent for Linux.

BETA

Printing Support

The PCoIP Graphics Agent for Linux does not support local printing on remote clients.

Local, network, and cloud printers are supported in various ways:

- Linux hosts can print to any printer on the host machine's local area network.
- If your host host workstation has access to the Internet, cloud-based printing is supported through cloud-printing services such as Google Cloud Print and HP Mobile Printing.

BETA

USB Support

The Graphics Agent for Linux provides support for USB devices and certain Wacom tablets attached to PCoIP clients.

USB bridging must be explicitly installed

When installing the Graphics Agent for Linux, you must explicitly enable support for USB bridging by installing the required USB dependencies yourself. Refer to the installation steps for Ubuntu and RHEL/CentOS for the required commands. If the required USB dependencies are not installed, the Graphics Agent for Linux will be incapable of bridging USB devices.

This requirement does not affect support for keyboards, and mice or other pointer devices. It does not affect Wacom tablet support.

If the required USB packages are installed, USB bridging support is enabled by default. Administrators can disable or configure USB behavior by changing configuration options.

Keyboards, mice, and other pointer devices are managed by PCoIP clients, and are always allowed.

Xbox One Controller Support

The PCoIP Graphics Agent for Linux supports Xbox One controllers when attached to PCoIP Zero Clients.

Supported by PCoIP Zero Clients only

This feature is supported only by PCoIP Zero Clients. It is not currently supported by PCoIP Software Clients.

The following Xbox One controllers are supported:

- Xbox One 2015
- Xbox One
- Xbox One S
- Xbox One Bt
- Xbox One Elite

© 2021 Teradici

Wacom Tablet Support

The Graphics Agent for Linux supports Wacom tablets in two configurations: *bridged*, where peripheral data is sent to the desktop for processing, and *locally terminated*, where peripheral data is processed locally at the PCoIP client.

Locally Terminated Wacom Tablets

Locally-terminated tablets have greatly improved responsiveness, and tolerate higher-latency (including 25ms and higher) networks.

For the best experience and most complete device support, use the latest available PCoIP agent, PCoIP software client, and PCoIP Zero Client firmware. To find out when support was added for individual Wacom device, refer to the release notes for your client.

> **Caution: Using Wacom Local Termination on Ubuntu Cloud Hosts**
>
> Cloud-based Ubuntu hosts may fail to properly handle locally terminated Wacom tablets. When this occurs, pressure sensitivity and other advanced features will not work properly. To correct this issue, follow this procedure.

The following Wacom tablet models have been tested and are supported with local termination mode:

PCoIP client support for *locally terminated* Wacom tablets and the Graphics Agent for Linux

	PCoIP Tera2 Zero Client *6.2.0+, except as noted*	PCoIP Software Client for Windows	PCoIP Software Client for macOS	PCoIP Software Client for Linux
Intuos Pro Small *PTH-460*	–	✔	✔	✔
Intuos Pro Medium *PTH-660*	✔	✔	✔	✔

© 2021 Teradici

	PCoIP Tera2 Zero Client *6.2.0+, except as noted*	PCoIP Software Client for Windows	PCoIP Software Client for macOS	PCoIP Software Client for Linux
Intuos Pro Large *PTH-860*	✔	✔	✔	✔
Cintiq 22HD *DTK-2200*	✔[1]	✔	–	✔
Cintiq 22 *DTK-2260*	–	✔	✔	✔
Cintiq Pro 24 *DTK-2420*	✔[1]	✔	–	✔
Cintiq 22HDT - Pen & Touch *DTH-2200*	✔[1]	–	–	–
Cintiq Pro 24 - Pen & Touch *DTH-2420*	✔[1]	✔	✔	✔
Cintiq 32 Pro - Pen & Touch *DTH-3220*	✔[2]	✔	✔	✔

Important: Touch is not supported

Touch features of Wacom devices are not supported with local termination.

Other Wacom tablets may work, but have not been tested and should not be used in production environments.

BETA

Bridged Wacom Tablets

Bridged Wacom tablets are supported only in low-latency environments. Tablets in network environments with greater than 25ms latency will show reduced responsiveness and are not recommended.

The following Wacom tablet models have been tested and are supported with bridged mode:

PCoIP client support for *bridged* Wacom tablets and the Graphics Agent for Linux

	PCoIP Tera2 Zero Client	PCoIP Software Client for Windows	PCoIP Software Client for macOS	PCoIP Software Client for Linux
Intuos Pro Small *PTH-460*	✔	✔	✔	✔
Intuos Pro Medium *PTH-660*	✔	✔	✔	✔
Intuos Pro Large *PTH-860*	✔	✔	✔	✔
Cintiq 22HD *DTK-2200*	✔	✔	✔	✔
Cintiq Pro 24 *DTK-2420*	✔	✔	✔	✔
Cintiq 22HDT - Pen & Touch *DTH-2200*	✔ *Ubuntu only* [3]	✔ *Ubuntu only* [3]	✔ *Ubuntu only* [3]	✔ *Ubuntu only* [3]
Cintiq Pro 24 - Pen & Touch *DTH-2420*	✔	✔	✔	✔

BETA

	PCoIP Tera2 Zero Client	PCoIP Software Client for Windows	PCoIP Software Client for macOS	PCoIP Software Client for Linux
Cintiq 32 Pro - Pen & Touch *DTH-3220*	✔	✔	✔	✔

Other Wacom tablets may work, but have not been tested.

1. Local termination for Cintiq 22HD, 22HDT, 24P, and 24PT requires Tera2 Zero Client firmware 6.5.0 or higher.
2. Local termination for Cintiq Pro 32PT requires Tera2 Zero Client firmware 20.04 or higher.
3. Launching a PCoIP session with a ***bridged*** Cintiq 22HDT (DTH-2200) and a ***RHEL/CentOS host*** can cause the remote system to disconnect and become unresponsive. This issue does not occur when bridging to Ubuntu hosts.

BETA

PCoIP Graphics Agent for Linux Installation Guide

BETA

Installation instructions are provided here for Ubuntu and RHEL/CentOS distributions of Linux:

- Instructions for Ubuntu installations
- Instructions for RHEL or CentOS installations

Installing the PCoIP Graphics Agent for Linux on Ubuntu

Before you proceed with installation, a few prerequisites must be met.

Prerequisites

These instructions assume you have already built the remote desktop machine, and that the machine meets the agent's requirements.

Important: A desktop environment is required

Before proceeding, install a desktop environment of your choice. Kubuntu distributions are bundled with KDE; you can install KDE from other distributions by using this command:

```
sudo apt install kubuntu-desktop
```

To install Mate Desktop, use this command:

```
sudo apt install ubuntu-mate-desktop
```

These commands are provided as a convenience; there is no requirement for KDE or Mate Desktop. Any desktop environment will work.

The Graphics Agent for Linux has additional requirements for GPU acceleration. Make sure your machine is configured with a supported card before you start (we'll install the correct driver as part of this process).

A few other things to confirm before proceeding:

- SSH must be enabled.
- You must have a license registration code for the agent instance from Teradici (as part of a Teradici Cloud Access subscription).
- The desktop machine requires the following ports to be open: TCP 443, TCP 60443, TCP 4172, and UDP 4172.

© 2021 Teradici

- Your GPUs must be licensed with their manufacturers (if required).
- You must have super user (root) privileges and be able to issue `sudo` commands.
- If you are using a PCoIP Local License Server, PCoIP Local License Server, you'll need to know it's URL and port numbers.

Important: Protect your license registration code

The license registration code you receive from Teradici is unique to your organization, and should be protected as you would any sensitive data.

Be careful that you do not inadvertently expose your registration code in forums or other public areas by pasting log messages without redacting sensitive information.

BETA

Installation Overview

Once your prerequisites are in place, you can proceed with installation. Here's a brief overview of the process:

1. Connect to the machine using SSH.
2. Install the appropriate GPU drivers for your system.
3. Install the PCoIP Agent on a PCoIP Agent.
4. If required, configure the agent software.
5. Disconnect the SSH session.
6. Connect to the desktop using a PCoIP client.

If you're ready to start, connect to your machine with an SSH client and proceed to Installing GPU Drivers.

Installing GPU Drivers on Linux machines

Before installing the PCoIP Agent, you need to install the correct driver for your configured GPUs. SSH into your machine and then proceed with the installation instructions below.

The correct driver versions for release 21.10 are shown below. Be sure to install the specified driver, and ***not the latest available version***. Teradici qualifies Graphics Agent support against the driver versions listed here.

Documentation for download and installation of these drivers is provided by their manufacturers. As a convenience, we've provided the current locations of relevant third-party documentation here. While we do our best to keep these links up to date, Teradici does not control these resources or their locations and it's possible that they could break.

Install NVIDIA GRID Drivers on AWS Instances

Instance Type	Supported GPUs	Supported NVIDIA GRID driver version	Installation instructions
EC2 G3	NVIDIA M60	GRID 12.2 (460.73.01)+	Download the driver from Amazon using the ***AWS CLI or SDKs***, as documented by Amazon.
EC2 G4dn	NVIDIA T4	GRID 12.2 (460.73.01)+	Download the driver from Amazon using the ***AWS CLI or SDKs***, as documented by Amazon. G4 instances must also include NVIDIA licensing, as documented by NVIDIA.

© 2021 Teradici

BETA

Install NVIDIA Drivers on Physical PCs

Instance Type	Supported GPUs	Supported NVIDIA driver version	Installation instructions
Physical PC	• Tesla (any) • Quadro (k2000+)	Use NVIDIA's recommended driver.	Download the driver directly from NVIDIA and follow their instructions to install. You may only have access to this driver if you have purchased a corresponding card.

Install NVIDIA GRID Drivers on Azure NV-series Instances

Instance Type	Supported GPUs	Supported NVIDIA GRID driver version	Installation instructions
Azure NV-series	NVIDIA M60	GRID 12.2 (460.73.01)+	Microsoft Azure download

Install NVIDIA GRID Drivers on Google Cloud Instances

Instance Type	Supported GPUs	Supported NVIDIA GRID driver version	Installation instructions
Google Cloud instances	NVIDIA P4, P100, T4	GRID 12.2 (460.73.01)+	Google Cloud Documentation

Install NVIDIA GRID Drivers on Other Instance Types

Instance Type	Supported GPUs	Supported NVIDIA GRID driver version	Installation instructions
All other instance types	GPUs as listed in NVIDIA's GRID release notes.	GRID 12.2 (460.73.01)+	Instructions shown below.

© 2021 Teradici

BETA

To install NVIDIA driver for all other instances, including non-cloud instances:

1. Install tools required to install the NVIDIA drivers:

   ```
   sudo apt install build-essential
   ```

2. Disable the Nouveau video driver (two commands):

   ```
   echo 'blacklist nouveau' | sudo tee -a /etc/modprobe.d/blacklist.conf
   ```

3. As a super user (sudo), edit the file `/etc/default/grub`. Append the following to the GRUB_CMDLINE_LINUX entry:

   ```
   rd.driver.blacklist=nouveau nouveau.modeset=0
   ```

   For example:

   ```
   GRUB_CMDLINE_LINUX="crashkernel=auto console=ttyS0,38400n8
   rd.driver.blacklist=nouveau nouveau.modeset=0"
   ```

   Generate a new grub configuration to include the above changes, then reboot:

   ```
   sudo grub2-mkconfig -o /boot/grub2/grub.cfg
   sudo reboot
   ```

4. Install the NVIDIA driver:

   ```
   sudo ./NVIDIA-Linux-x86_64-xxx.xx-grid.run
   ```

   Where `xxx.xx` is the NVIDIA driver version shown in Linux System Requirements.

5. Respond to the installer prompts as follows:
   - Accept the EULA
   - Say ***yes*** to installing 32-bit binaries
   - Say ***no*** to modifying the x.org file
6. Verify the NVIDIA installation:

© 2021 Teradici

```
nvidia-smi
```

If the installation was successful, you will see your video card in the response.

Installing the PCoIP Agent

Once the NVIDIA drivers are present, you can install the Graphics Agent for Linux.

BETA

Installing the Graphics Agent for Linux on Ubuntu

Important: Required ports will be automatically opened

The Graphics Agent for Linux installer will add firewall exceptions for the following required PCoIP ports during installation: TCP 443, TCP 4172, UDP 4172, and TCP 60443.

To install the PCoIP Graphics Agent for Linux software:

1. Download and install the Teradici pcoip-agent-beta repository, via the shell script provided here.

   Warning: This installs the beta repository

   This will install the *beta* repository for the Graphics Agent for Linux. The beta repo runs ahead of current stable releases, and may contain bugs or incomplete features. We welcome feedback from beta users. Beta releases should not be used in production systems.

2. **Optionally** install USB dependencies, if you intend to support USB devices other than keyboards, mice, and pointer devices. ***If you skip this step, USB redirection will be completely disabled and bridged USB devices will not work.***

   ```
   sudo apt install usb-vhci-dkms
   ```

3. Install the PCoIP Graphics Agent for Linux:

   ```
   sudo apt update
   sudo apt install pcoip-agent-graphics
   ```

4. Note your machine's local IP address. Clients connecting directly to the host workstation will need this number to connect.
5. Enter the license registration code you received from Teradici.

© 2021 Teradici

BETA

> **Note: These instructions are for Cloud Licensing**
>
> These instructions assume you are using Teradici Cloud Licensing to activate your PCoIP session licenses. If you are using the Teradici License Server instead, see Licensing the Graphics Agent for Linux.

For unproxied internet connections, type:

```
pcoip-register-host --registration-code=<XXXXXX@YYY-YYYY-YYY>
```

For proxied internet conections, type:

```
pcoip-register-host --registration-code=<XXXXXX@YYY-YYYY-YYY> --proxy-
server=<serverURL> --proxy-port=<port>
```

6. Reboot the desktop.

> **Installing on stock Ubuntu 18.04 installations**
>
> When connecting to the Graphics Agent for Linux installed on a stock Ubuntu 18.04 installation, users may not be able to establish PCoIP sessions. You can work around this issue by setting the `pcoip.desktop_session` value in `pcoip-agent.conf`:
>
> 1. Open `/etc/pcoip-agent/pcoip-agent.conf` in a text editor
> 2. Add the following line:
>
>    ```
>    pcoip.desktop_session = ubuntu
>    ```
>
> 3. Save and close the editor.

> **Note: Desktop user interfaces will only be available using PCoIP**
>
> Once installed and running, the PCoIP Graphics Agent for Linux takes over the graphics subsystem which is then unavailable to hypervisors. You can only view the graphical user interface when connecting with a PCoIP client.
>
> For example, you cannot view an ESXi virtual machine console through VSphere; you must connect to the machine using PCoIP.

© 2021 Teradici

Installing the Graphics Agent for Linux on a Physical PC

When installing the PCoIP Agent on a physical machine with a Quadro card, additional steps are required. **If you installed the Graphics Agent for Linux on a virtual machine, you can ignore these steps.**

> **Note: Install the agent first**
>
> You must install the Graphics Agent for Linux as described above before proceeding.

> **Important: Some GPUs are automatically configured**
>
> The Graphics Agent for Linux can use the following NVIDIA GPUs automatically:
>
> - K2000
> - K2200
> - K4000
> - K4200
> - K5200
> - M4000
> - P2000
> - P4000
> - P5000
> - P6000
> - RTX4000
>
> If you are unable to start a session using any of these GPUs, follow the procedure described next.

These steps are only required when installing on a physical, non-virtualized machine with a Quadro card:

1. Attempt to start a PCoIP session. **If the connection attempt succeeds, you can stop here**. Your GPU will work without further configuration. **If the attempt fails**, continue on; we will use the log information generated by the failed attempt in the next steps.
2. Look in `/var/log/Xorg.100.log` for lines like this:

BETA

```
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-0): connected
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-0): Internal DisplayPort
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-0): 1440.0 MHz maximum
pixel clock
[   293.834] (--) NVIDIA(GPU-0):
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-1): connected
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-1): Internal TMDS
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-1): 165.0 MHz maximum
pixel clock
[   293.834] (--) NVIDIA(GPU-0):
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-2): connected
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-2): Internal DisplayPort
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-2): 1440.0 MHz maximum
pixel clock
[   293.835] (--) NVIDIA(GPU-0):
[   293.835] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-3): connected
[   293.835] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-3): Internal TMDS
[   293.835] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-3): 165.0 MHz maximum
pixel clock
[   293.835] (--) NVIDIA(GPU-0):
[   293.835] (--) NVIDIA(GPU-0): DFP-4: disconnected
[   293.835] (--) NVIDIA(GPU-0): DFP-4: Internal DisplayPort
[   293.835] (--) NVIDIA(GPU-0): DFP-4: 1440.0 MHz maximum pixel clock
[   293.835] (--) NVIDIA(GPU-0):
[   293.835] (--) NVIDIA(GPU-0): DFP-5: disconnected
[   293.835] (--) NVIDIA(GPU-0): DFP-5: Internal TMDS
[   293.835] (--) NVIDIA(GPU-0): DFP-5: 165.0 MHz maximum pixel clock
[   293.835] (--) NVIDIA(GPU-0):
[   293.835] (--) NVIDIA(GPU-0): DFP-6: disconnected
[   293.835] (--) NVIDIA(GPU-0): DFP-6: Internal DisplayPort
[   293.835] (--) NVIDIA(GPU-0): DFP-6: 1440.0 MHz maximum pixel clock
[   293.835] (--) NVIDIA(GPU-0):
[   293.835] (--) NVIDIA(GPU-0): DFP-7: disconnected
[   293.835] (--) NVIDIA(GPU-0): DFP-7: Internal TMDS
[   293.835] (--) NVIDIA(GPU-0): DFP-7: 165.0 MHz maximum pixel clock
```

3. Note the names of the outputs with the highest maximum pixel clocks (as many as four will be shown). In the above example, you would note `DFP-0`, `DFP-2`, `DFP-4`, and `DFP-6`.

> **Note: Systems with fewer than four displays (Quadro K series)**
>
> If there are fewer than four outputs in your system, note them all. K-series cards only support two enabled outputs at a time, but may show more.

4. Create a file called `/etc/X11/xorg.conf.d/10-pcoip.conf`:

```
touch /etc/X11/xorg.conf.d/10-pcoip.conf
```

5. Open `/etc/X11/xorg.conf.d/10-pcoip.conf` in a text editor.
6. Paste the following text into the file:

```
Section "Screen"
Identifier "dummy_screen"
Device "dummy_videocard"
Option "UseDisplayDevice" ""
Option "ConnectedMonitor" ""
Option "Monitor-" "Monitor0"
Option "Monitor-" "Monitor1"
Option "Monitor-" "Monitor2"
Option "Monitor-" "Monitor3"
Monitor "Monitor0"
EndSection
```

7. Populate the `ConnectedMonitor` and `UseDisplayDevice` settings with the display IDs you noted earlier, as a comma-separated list. Using our example (your values may be different):

```
Option "UseDisplayDevice" "DFP-0,DFP-2,DFP-4,DFP-6"
Option "ConnectedMonitor" "DFP-0,DFP-2,DFP-4,DFP-6"
```

8. Next, map these output values to your expected displays. Modify the monitor options to correspond to the displays in the previous step. Each output should be mapped to a monitor identifier from one of the `Monitor` sections in `10-pcoip.conf`.

   Continuing our example, we would map the four outputs to four monitors like this (yours may be different; for example, you may only have two outputs):

```
Option "Monitor-DFP-0" "Monitor0"
Option "Monitor-DFP-2" "Monitor1"
Option "Monitor-DFP-4" "Monitor2"
Option "Monitor-DFP-6" "Monitor3"
```

9. Save and close `10-pcoip.conf`.
10. Attempt another PCoIP session. If the session is still unsuccessful, and you have a Quadro K series card, repeat steps 7 and 8 with a different output combination, e.g. if you tried

© 2021 Teradici

`DFP-3,DFP-4`, try `DFP-1,DFP-2`. Due to driver differences certain output combinations may not work.

11. To prevent the NVIDIA driver from changing the kernel mode setting, add the `nomodeset` parameter to the GRUB configuration:
    a. Open **`/etc/default/grub`** in a text editor.
    b. Add `nomodeset` to `GRUB_CMDLINE_LINUX_DEFAULT`.
    c. Save and close the file.
    d. Rebuild the GRUB configuration:

    ```
    sudo update-grub
    ```

    e. Reboot the machine.
12. Attempt a PCoIP connection again. The connection should succeed.

    If the session does not start, make sure that the displays are correctly configured.

Once you've installed the software, you can configure it, register licenses, or connect to it.

BETA

Licensing The Graphics Agent for Linux

The Graphics Agent for Linux must be assigned a valid PCoIP session license before it will work. Until you've registered it, you can't connect to the desktop using a PCoIP client.

You receive a registration code when you purchase a pool of licenses from Teradici. Each registration code can be used multiple times; each use consumes one license in its pool.

> **Note: Registration code format**
>
> Registration codes look like this: `ABCDEFGH12@AB12-C345-D67E-89FG`

PCoIP agent license registrations are managed automatically by Teradici's Cloud Licensing service. If necessary, you can manage them yourself, using your own locally-installed PCoIP license server instead.

If you need to purchase licenses, contact Teradici.

Troubleshooting Licensing Issues

If you're encountering problems with Teradici licensing, refer to Troubleshooting License Issues.

Using Teradici Cloud Licensing

To use Cloud Licensing, all you need to do is provide a registration code for each PCoIP agent in your deployment (the same registration code can be used multiple times).

To provide the registration code:

SSH into the agent machine, and invoke `pcoip-register-host` with the license registration code and proxy settings if required:

```
pcoip-register-host --registration-code=<registration-code> [--proxy-
server=<proxy-server-address>] [--proxy-port=<proxy-port-number>]
```

© 2021 Teradici

Whitelist network blocks for Teradici Cloud Licensing

If you are using Teradici Cloud Licensing, you will need to whitelist the following:

- teradici.flexnetoperations.com
- teradici.compliance.flexnetoperations.com

Alternatively, you can also ensure the following network blocks are whitelisted:

- **Production**: 64.14.29.0/24
- **Disaster Recovery**: 64.27.162.0/24

The following network blocks are not currently in use, but may also be used in the future:

- **Production**: 162.244.220.0/24
- **Disaster Recovery**: 162.244.222.0/24

Licensing PCoIP Agents With a Local License Server

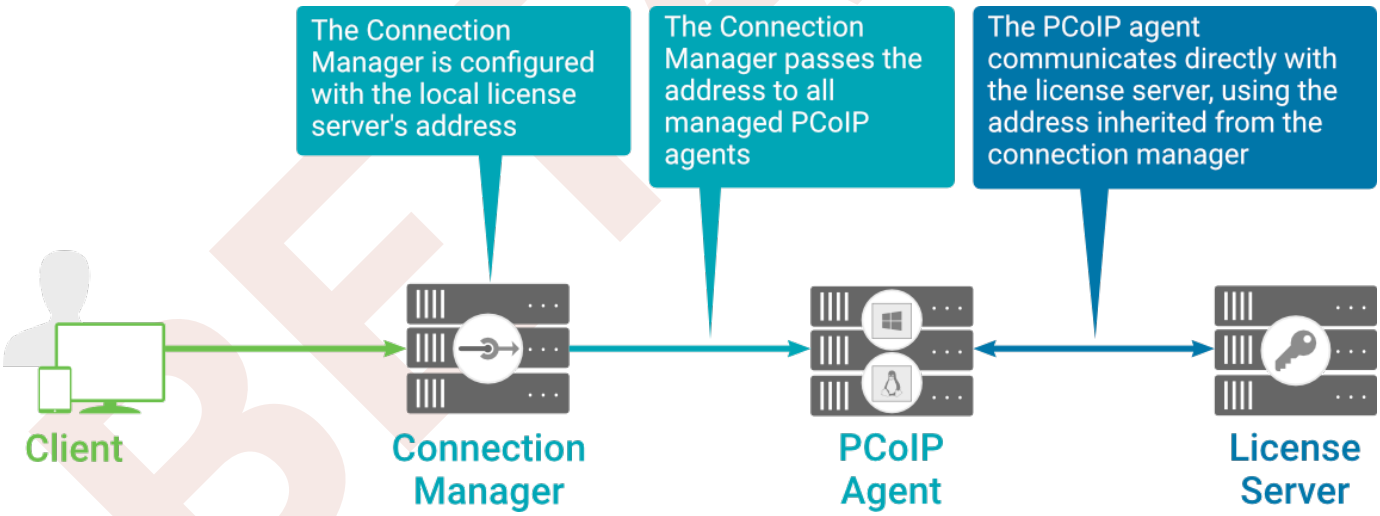
In deployments where PCoIP agents cannot access the internet, or where cloud-based licensing is not permitted or desired, a local PCoIP License Server can be used instead. The PCoIP License Server manages PCoIP session licenses within your private environment.

Configuring PCoIP agents to use a local license server is done in one of two ways, depending on whether your deployment uses a PCoIP Connection Manager, or whether your PCoIP clients connect directly to PCoIP agents.

Brokered Environment Licensing

In ***brokered*** deployments, the license server address is configured in the Connection Manager, which passes it through to its managed PCoIP agents.

BETA



Local license validation using a Graphics Agent for Linux and a brokered connection

When using a Connection Manager, the license server address is only configured once no matter how many PCoIP agents are behind the Connection Manager.

To set the License Server URL in the Connection Manager:

1. On the Connection Manager machine, use a text editor to open /etc/ConnectionManager.conf.
2. Set the `LicenseServerAddress` parameter with the address of your local license server:
   - `http://{license-server-address}:{port}/request`
3. Save and close the configuration file.
4. Restart the Connection Manager.

Verifying Your Brokered Licensing Configuration

To verify your system's licensing configuration, run `pcoip-validate-license` from the console on the Graphics Agent for Linux machine. The command will ping the license server and attempt to retrieve information on an available license:

```
pcoip-validate-license --license-server-url <license-server-address> [--proxy-server <proxy-server-address>] [--proxy-port <proxy-port-number]
```

Where `<license-server-address>` is the address of the license server to ping, formatted as `http://{license-server-address}:{port}/request`

If the license server is behind a proxy server, provide the proxy information via the `--proxy-server` and `--proxy-port` parameters.

© 2021 Teradici

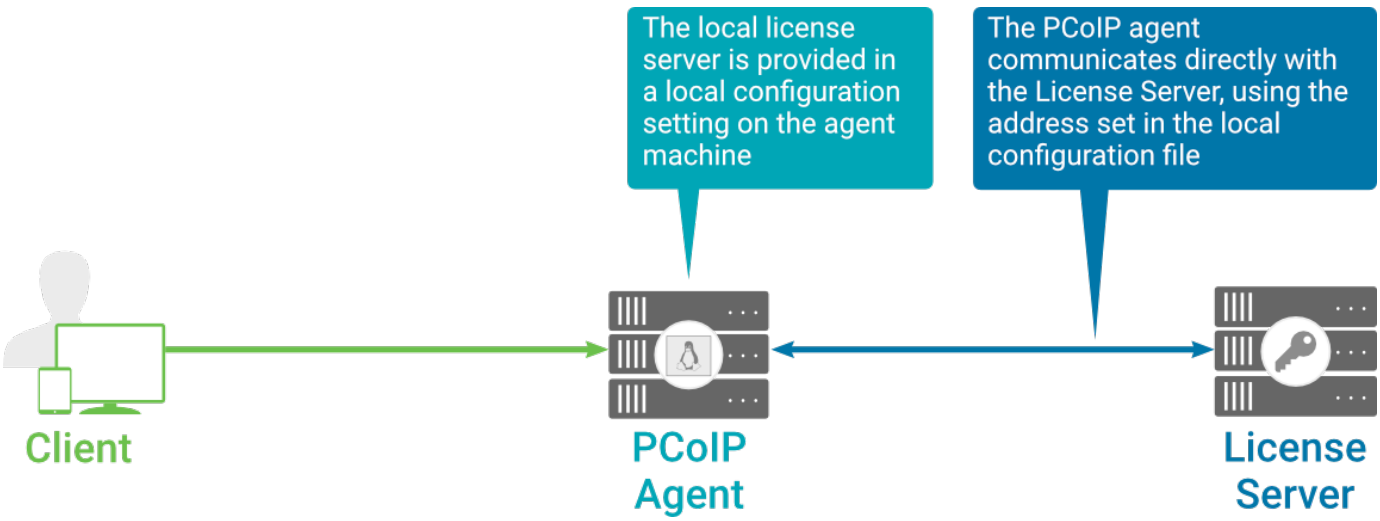
If successful, the response will show that a valid license was found on the license server, and its expiration date.

If the connection is unsuccessful, investigate the following possibilities:

- The license server address is incorrect, or formatted incorrectly.
- The license server is inaccessible.
- There are no available licenses on the license server. `pcoip-validate-license` will only return a positive response if there is at least one available session license.
- If you have only one license on the license server and run `pcoip-validate-license` from a PCoIP session, the command will fail because you are currently using the single license. In this scenario, disconnect your PCoIP session and try again from an SSH session instead.

Unbrokered Environment Licensing

In direct, or unbrokered, deployments, each PCoIP agent is configured with the license server address via a local agent setting. When a client initiates a new PCoIP session, the PCoIP agent uses its local configuration to communicate with the license server.



Local license validation using a Graphics Agent for Linux and a direct (unbrokered) connection

Each PCoIP agent in your environment must be individually configured with the license server's URL.

BETA

To configure the License Server URL on the Graphics Agent for Linux machine:

1. Using a text editor, open `/etc/pcoip-agent/pcoip-agent.conf`.
2. Add or modify the `pcoip.license_server_path` directive:

   ```
   pcoip.license_server_path = <license-server-address>
   ```

   Where `<license-server-address>` is the address of the license server, formatted as `http://{license-server-address}:{port}/request`.
3. If the license server is behind a proxy server, provide the proxy information using the `pcoip.license_proxy_server` and `pcoip.license_proxy_port` directives.
4. Save and close `pcoip-agent.conf`.

The changes will take effect on the next PCoIP session.

Verifying Your Unbrokered Licensing Configuration

To verify your system's licensing configuration, run `pcoip-validate-license` from the console on the Graphics Agent for Linux machine. The command will ping the license server and attempt to retrieve information on an available license:

```
pcoip-validate-license --license-server-url <license-server-address> [--proxy-
server <proxy-server-address>] [--proxy-port <proxy-port-number]
```

Where `<license-server-address>` is the address of the license server to ping, formatted as `http://{license-server-address}:{port}/request`

If the license server is behind a proxy server, provide the proxy information via the `--proxy-server` and `--proxy-port` parameters.

If successful, the response will show that a valid license was found on the license server, and its expiration date.

If the connection is unsuccessful, investigate the following possibilities:

- The license server address is incorrect, or formatted incorrectly.
- The license server is inaccessible.

BETA

- There are no available licenses on the license server. `pcoip-validate-license` will only return a positive response if there is at least one available session license.
- If you have only one license on the license server and run `pcoip-validate-license` from a PCoIP session, the command will fail because you are currently using the single license. In this scenario, disconnect your PCoIP session and try again from an SSH session instead.

BETA

Updating the Graphics Agent for Linux on Ubuntu

Updates to the Graphics Agent for Linux will be published on a regular basis. New stable builds will be produced approximately every three months.

To upgrade to the latest version, use the following three commands:

```
sudo apt update
sudo apt install pcoip-agent-graphics
sudo reboot
```

BETA

Uninstalling the Graphics Agent for Linux

You can remove the Graphics Agent for Linux from your system, or you can remove the repo config entirely.

Remove the Graphics Agent for Linux package

To remove the package, open a console window and run the following command:

```
sudo apt-get remove pcoip-agent-*
```

Remove the repo configuration

If you want to remove the repo configuration completely, you can do that as well. You'll need to do this if you are switching from one channel to another (for example, from beta to stable), before reconfiguring with the new repo:

```
rm /etc/apt/sources.list.d/teradici-pcoip-agent-beta
.list
apt-get clean
rm -rf /var/lib/apt/lists/*
apt-get update
```

© 2021 Teradici

Installing the PCoIP Graphics Agent for Linux on RHEL or CentOS

Before you proceed with installation, a few prerequisites must be met.

Prerequisites

These instructions assume you have already built the remote desktop machine, and that the machine meets the agent's requirements.

Before proceeding with Graphics Agent for Linux installation, install a desktop environment. To install a desktop environment in RHEL or CentOS, use the following command:

```
sudo yum groupinstall 'Server with GUI'
```

The Graphics Agent for Linux has additional requirements for GPU acceleration. Make sure your machine is configured with a supported card before you start (we'll install the correct driver as part of this process).

A few other things to confirm before proceeding:

- SSH must be enabled.
- You must have a license registration code for the agent instance from Teradici (as part of a Teradici Cloud Access subscription).
- The desktop machine requires the following ports to be open: TCP 443, TCP 60443, TCP 4172, and UDP 4172.
- Your GPUs must be licensed with their manufacturers (if required).
- You must have super user (root) privileges and be able to issue `sudo` commands.
- If you are using a PCoIP Local License Server, PCoIP Local License Server, you'll need to know it's URL and port numbers.

© 2021 Teradici

Important: Protect your license registration code

The license registration code you receive from Teradici is unique to your organization, and should be protected as you would any sensitive data.

Be careful that you do not inadvertently expose your registration code in forums or other public areas by pasting log messages without redacting sensitive information.

Installation Overview

Once your prerequisites are in place, you can proceed with installation. Here's a brief overview of the process:

1. Connect to the machine using SSH.
2. Install the appropriate GPU drivers for your system.
3. Install the PCoIP Agent.
4. If required, configure the agent software.
5. Disconnect the SSH session.
6. Connect to the desktop using a PCoIP client.

If you're ready to start, connect to your machine with an SSH client and proceed to Installing GPU Drivers.

BETA

Installing GPU Drivers on Linux machines

Before installing the PCoIP Agent, you need to install the correct driver for your configured GPUs. SSH into your machine and then proceed with the installation instructions below.

The correct driver versions for release 21.10 are shown below. Be sure to install the specified driver, and ***not the latest available version***. Teradici qualifies Graphics Agent support against the driver versions listed here.

Documentation for download and installation of these drivers is provided by their manufacturers. As a convenience, we've provided the current locations of relevant third-party documentation here. While we do our best to keep these links up to date, Teradici does not control these resources or their locations and it's possible that they could break.

Install `libglvnd-opengl` before proceeding

Before installing the NVIDIA drivers, install the `libglvnd-opengl` package:

```
sudo yum install libglvnd-opengl
```

Install NVIDIA GRID Drivers on AWS Instances

Instance Type	Supported GPUs	Supported NVIDIA GRID driver version	Installation instructions
EC2 G3	NVIDIA M60	GRID 12.2 (460.73.01)+	Download the driver from Amazon using the ***AWS CLI or SDKs***, as documented by Amazon.
EC2 G4dn	NVIDIA T4	GRID 12.2 (460.73.01)+	Download the driver from Amazon using the ***AWS CLI or SDKs***, as documented by Amazon. G4 instances must also include NVIDIA licensing, as documented by NVIDIA.

© 2021 Teradici

BETA

Install NVIDIA Drivers on Physical PCs

Instance Type	Supported GPUs	Supported NVIDIA driver version	Installation instructions
Physical PC	• Tesla (any) • Quadro (k2000+)	Use NVIDIA's recommended driver.	Download the driver directly from NVIDIA and follow their instructions to install. You may only have access to this driver if you have purchased a corresponding card.

Install NVIDIA GRID Drivers on Azure NV-series Instances

Instance Type	Supported GPUs	Supported NVIDIA GRID driver version	Installation instructions
Azure NV-series	NVIDIA M60	GRID 12.2 (460.73.01)+	Microsoft Azure download

Install NVIDIA GRID Drivers on Google Cloud Instances

Instance Type	Supported GPUs	Supported NVIDIA GRID driver version	Installation instructions
Google Cloud instances	NVIDIA P4, P100, T4	GRID 12.2 (460.73.01)+	Google Cloud Documentation

Install NVIDIA GRID Drivers on Other Instance Types

Instance Type	Supported GPUs	Supported NVIDIA GRID driver version	Installation instructions
All other instance types	GPUs as listed in NVIDIA's GRID release notes.	GRID 12.2 (460.73.01)+	Instructions shown below.

© 2021 Teradici

To install NVIDIA driver for all other instances, including non-cloud instances:

1. Install tools required to install the NVIDIA drivers:

   ```
   sudo yum install kernel-devel
   ```

2. Disable the Nouveau video driver (two commands):

   ```
   echo 'blacklist nouveau' | sudo tee -a /etc/modprobe.d/blacklist.conf
   ```

3. As a super user (sudo), edit the file `/etc/default/grub`. Append the following to the GRUB_CMDLINE_LINUX entry:

   ```
   rd.driver.blacklist=nouveau nouveau.modeset=0
   ```

   For example:

   ```
   GRUB_CMDLINE_LINUX="crashkernel=auto console=ttyS0,38400n8
   rd.driver.blacklist=nouveau nouveau.modeset=0"
   ```

   Generate a new grub configuration to include the above changes, then reboot:

   ```
   sudo grub2-mkconfig -o /boot/grub2/grub.cfg
   sudo reboot
   ```

4. Install the NVIDIA driver:

   ```
   sudo ./NVIDIA-Linux-x86_64-xxx.xx-grid.run
   ```

   Where `xxx.xx` is the NVIDIA driver version shown in Linux System Requirements.

   Important: Azure installations require Hyper-V daemon installation

   If you are running on an Azure instance, you also need to install the Azure Hyper-V daemon:

   ```
   sudo yum install hyperv-daemons
   ```

BETA

5. Respond to the installer prompts as follows:
    - Accept the EULA
    - Say ***yes*** to installing 32-bit binaries
    - Say ***no*** to modifying the x.org file
6. Update `initramfs` using `dracut`:

    ```
    dracut -fv
    ```

7. Verify the NVIDIA installation:

    ```
    nvidia-smi
    ```

    If the installation was successful, you will see your video card in the response.

Installing the PCoIP Agent

Once the NVIDIA drivers are present, you can install the Graphics Agent for Linux.

BETA

Installing the Graphics Agent for Linux on RHEL or CentOS

BETA

Important: Required ports will be automatically opened

The Graphics Agent for Linux installer will add firewall exceptions for the following required PCoIP ports during installation: TCP 443, TCP 4172, UDP 4172, and TCP 60443.

To install the Graphics Agent for Linux software:

1. Download and install the Teradici pcoip-agent-beta repository, via the shell script provided here.

   Warning: This installs the beta repository

   This will install the *beta* repository for the Graphics Agent for Linux. The beta repo runs ahead of current stable releases, and may contain bugs or incomplete features. We welcome feedback from beta users. Beta releases should not be used in production systems.

2. Install the EPEL repository:

   ```
   sudo yum install epel-release
   ```

3. **Optionally** install USB dependencies, if you intend to support USB devices other than keyboards, mice, and pointer devices. *If you skip this step, USB redirection will be completely disabled and bridged USB devices will not work.*

   ```
   sudo yum install usb-vhci
   ```

4. Install the PCoIP Graphics Agent for Linux:

   ```
   sudo yum install pcoip-agent-graphics
   ```

5. Note your machine's local IP address. Clients connecting directly to the host workstation will need this number to connect.

© 2021 Teradici

6. Enter the license registration code you received from Teradici.

> **Note: These instructions are for Cloud Licensing**
>
> These instructions assume you are using Teradici Cloud Licensing to activate your PCoIP session licenses. If you are using the Teradici License Server instead, see Licensing the Graphics Agent for Linux.

For unproxied internet connections, type:

```
pcoip-register-host --registration-code=<XXXXXX@YYY-YYYY-YYY>
```

For proxied internet conections, type:

```
pcoip-register-host --registration-code=<XXXXXX@YYY-YYYY-YYY> --proxy-
server=<serverURL> --proxy-port=<port>
```

7. Reboot the desktop.

Installing the Graphics Agent for Linux on a Physical PC

When installing the PCoIP Agent on a physical machine with a Quadro card, additional steps are required. **If you installed the Graphics Agent for Linux on a virtual machine, you can ignore these steps.**

> **Note: Install the agent first**
>
> You must install the Graphics Agent for Linux as described above before proceeding.

BETA

BETA

Important: Some GPUs are automatically configured

The Graphics Agent for Linux can use the following NVIDIA GPUs automatically:

- K2000
- K2200
- K4000
- K4200
- K5200
- M4000
- P2000
- P4000
- P5000
- P6000
- RTX4000

If you are unable to start a session using any of these GPUs, follow the procedure described next.

These steps are only required when installing on a physical, non-virtualized machine with a Quadro card:

1. Attempt to start a PCoIP session. **If the connection attempt succeeds, you can stop here**. Your GPU will work without further configuration. **If the attempt fails**, continue on; we will use the log information generated by the failed attempt in the next steps.
2. Look in `/var/log/Xorg.100.log` for lines like this:

```
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-0): connected
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-0): Internal DisplayPort
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-0): 1440.0 MHz maximum
pixel clock
[   293.834] (--) NVIDIA(GPU-0):
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-1): connected
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-1): Internal TMDS
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-1): 165.0 MHz maximum
pixel clock
[   293.834] (--) NVIDIA(GPU-0):
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-2): connected
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-2): Internal DisplayPort
[   293.834] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-2): 1440.0 MHz maximum
pixel clock
```

© 2021 Teradici

```
[   293.835] (--) NVIDIA(GPU-0):
[   293.835] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-3): connected
[   293.835] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-3): Internal TMDS
[   293.835] (--) NVIDIA(GPU-0): LNX Linux XGA (DFP-3): 165.0 MHz maximum
pixel clock
[   293.835] (--) NVIDIA(GPU-0):
[   293.835] (--) NVIDIA(GPU-0): DFP-4: disconnected
[   293.835] (--) NVIDIA(GPU-0): DFP-4: Internal DisplayPort
[   293.835] (--) NVIDIA(GPU-0): DFP-4: 1440.0 MHz maximum pixel clock
[   293.835] (--) NVIDIA(GPU-0):
[   293.835] (--) NVIDIA(GPU-0): DFP-5: disconnected
[   293.835] (--) NVIDIA(GPU-0): DFP-5: Internal TMDS
[   293.835] (--) NVIDIA(GPU-0): DFP-5: 165.0 MHz maximum pixel clock
[   293.835] (--) NVIDIA(GPU-0):
[   293.835] (--) NVIDIA(GPU-0): DFP-6: disconnected
[   293.835] (--) NVIDIA(GPU-0): DFP-6: Internal DisplayPort
[   293.835] (--) NVIDIA(GPU-0): DFP-6: 1440.0 MHz maximum pixel clock
[   293.835] (--) NVIDIA(GPU-0):
[   293.835] (--) NVIDIA(GPU-0): DFP-7: disconnected
[   293.835] (--) NVIDIA(GPU-0): DFP-7: Internal TMDS
[   293.835] (--) NVIDIA(GPU-0): DFP-7: 165.0 MHz maximum pixel clock
```

3. Note the names of the outputs with the highest maximum pixel clocks (as many as four will be shown). In the above example, you would note `DFP-0`, `DFP-2`, `DFP-4`, and `DFP-6`.

> **Note: Systems with fewer than four displays (Quadro K series)**
>
> If there are fewer than four outputs in your system, note them all. K-series cards only support two enabled outputs at a time, but may show more.

4. Create a file called `/etc/X11/xorg.conf.d/10-pcoip.conf`:

```
touch /etc/X11/xorg.conf.d/10-pcoip.conf
```

5. Open `/etc/X11/xorg.conf.d/10-pcoip.conf` in a text editor.
6. Paste the following text into the file:

```
Section "Screen"
Identifier "dummy_screen"
Device "dummy_videocard"
Option "UseDisplayDevice" ""
Option "ConnectedMonitor" ""
Option "Monitor-" "Monitor0"
```

© 2021 Teradici

```
Option "Monitor-" "Monitor1"
Option "Monitor-" "Monitor2"
Option "Monitor-" "Monitor3"
Monitor "Monitor0"
EndSection
```

7. Populate the `ConnectedMonitor` and `UseDisplayDevice` settings with the display IDs you noted earlier, as a comma-separated list. Using our example (your values may be different):

```
Option "UseDisplayDevice" "DFP-0,DFP-2,DFP-4,DFP-6"
Option "ConnectedMonitor" "DFP-0,DFP-2,DFP-4,DFP-6"
```

8. Next, map these output values to your expected displays. Modify the monitor options to correspond to the displays in the previous step. Each output should be mapped to a monitor identifier from one of the `Monitor` sections in `10-pcoip.conf`.

   Continuing our example, we would map the four outputs to four monitors like this (yours may be different; for example, you may only have two outputs):

```
Option "Monitor-DFP-0" "Monitor0"
Option "Monitor-DFP-2" "Monitor1"
Option "Monitor-DFP-4" "Monitor2"
Option "Monitor-DFP-6" "Monitor3"
```

9. Save and close `10-pcoip.conf`.
10. Attempt another PCoIP session. If the session is still unsuccessful, and you have a Quadro K series card, repeat steps 7 and 8 with a different output combination, e.g. if you tried `DFP-3,DFP-4`, try `DFP-1,DFP-2`. Due to driver differences certain output combinations may not work.
11. To prevent the NVIDIA driver from changing the kernel mode setting, add the `nomodeset` parameter to the GRUB configuration:
    a. Open `/etc/default/grub` in a text editor.
    b. Add `nomodeset` to `GRUB_CMDLINE_LINUX_DEFAULT`.
    c. Save and close the file.
    d. Rebuild the GRUB configuration:

```
sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

BETA

e. Reboot the machine.

12. Attempt a PCoIP connection again. The connection should succeed.

    If the session does not start, make sure that the displays are correctly configured.

Once you've installed the software, you can configure it, register licenses, or connect to it.

> **Note: Desktop user interfaces will only be available using PCoIP**
>
> Once installed and running, the PCoIP Graphics Agent for Linux takes over the graphics subsystem which is then unavailable to hypervisors. You can only view the graphical user interface when connecting with a PCoIP client.
>
> For example, you cannot view an ESXi virtual machine console through VSphere; you must connect to the machine using PCoIP.

Installing the Graphics Agent for Linux in a Dark Site

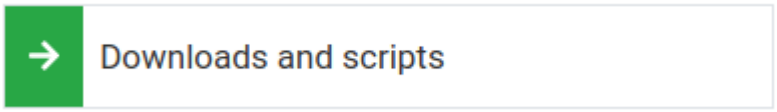
The Graphics Agent for Linux can be installed in an offline environment (with no connection to the public internet). To do this, you'll create a temporary internet-connected machine, download the agent RPM and its dependencies, then transfer the downloaded files to the permanent machine and install them.

To install the Graphics Agent for Linux in an offline environment:

> **Install GPU drivers before proceeding**
>
> The appropriate GPU driver must be downloaded, transferred, and installed before installing the Graphics Agent for Linux. The procedure for downloading GPU drivers varies by manufacturer and environment; a list of resources is located in Installing GPU Drivers on Linux Machines.

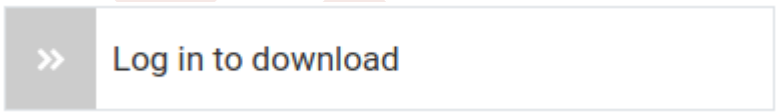
1. Create a temporary RHEL or CentOS machine with a connection to the internet and SSH into it.
2. On the temporary machine, install the Teradici repository:
   a. Using a browser, go to Teradici's documents and downloads site, and look in the sidebar for documentation and download links.
   b. Click the **Downloads and scripts** button.

BETA



Note: An account is required

If you are not logged in, you will see a log in prompt instead:



You can create an account when you click this button if you do not already have one.

c. Read and accept the Teradici End User License Agreement.

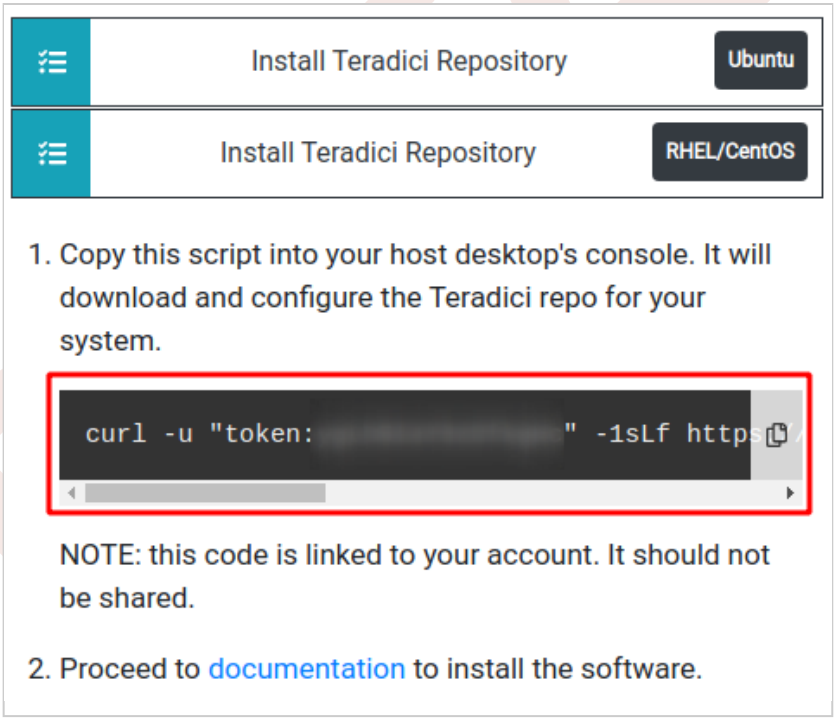
d. Click **Install Teradici Repository (RHEL/CentOS)** button

The container will expand to show a configuration script.

e. Copy the cURL script shown to your clipboard.

About the configuration script

This script automatically detects your operating system, and configures yum to download the appropriate packages. You can inspect the contents of the script by fetching the URL in the cURL command, without including `| sudo -E bash` .



3. On the temporary machine, paste the cURL command you copied in the previous step and press **Enter**. The script will set up the Teradici repo on your system.
4. Install the EPEL repository:

```
sudo yum install epel-release
```

5. **If you intend to support USB devices other than keyboards, mice, or pointers**, install usb-vhci dependencies:

```
sudo yum install usb-vhci
```

6. Update your yum cache:

```
sudo yum -y makecache
```

7. Download the `createrepo` utility.

```
sudo yum install -y yum-plugin-downloadonly yum-utils createrepo
```

8. Create two new directories. The first, **`agent-installroot`**, will be used to collect installation files and then deleted later. The second, **`agent-rpms`**, will receive the actual repo files and will be transferred to the offline machine.

```
mkdir agent-installroot
mkdir agent-rpms
```

9. Download the repo files. In all the following examples, replace `<absolute_path_to>` with the absolute path to your working directory, and replace `<os_version>` with the major os version number (for example, RHEL/CentOS 7.x is `7`):
   - If you will support USB devices *and you downloaded usb-vhci in step 5*:

     ```
     sudo yum install -y --downloadonly --installroot=<absolute_path_to>/
     agent-installroot --releasever=<os_version> --
     downloaddir=<absolute_path_to>/agent-rpms pcoip-agent-graphics usb-vhci
     ```

   - If you will *only* allow keyboards, mice, and pointer devices to connect via USB, omit `usb-vhci`. If you use this method, USB devices other than keyboards, mice, and pointer devices will not work:

     ```
     sudo yum install -y --downloadonly --installroot=<absolute_path_to>/
     agent-installroot --releasever=<os_version> --
     downloaddir=<absolute_path_to>/agent-rpms pcoip-agent-graphics
     ```

10. Create a local repo in a new `agent-rpms` directory. This is the folder you will ultimately transfer to the offline machine.

    ```
    sudo createrepo --database <absolute_path_to>/agent-rpms
    ```

11. Delete the installroot folder, which is no longer needed:

    ```
    rm -rf <absolute_path_to>/agent-installroot
    ```

12. Transfer the `agent-rpms` directory and its contents to the offline machine using any acceptable method, such as a USB flash drive. Make a note of the absolute destination filepath on the offline machine; we'll need that next.
13. On the offline machine: Open `/etc/yum.repos.d/offline-agent.repo` in a text editor.
14. Paste in the following content, replacing `<absolute_path_to>` with the absolute path to the `agent-rpms` directory:

BETA

```
[offline-agent]
name=CentOS-$releasever - agent-rpms
baseurl=file:///<absolute_path_to>/agent-rpms
enabled=0
gpgcheck=0
```

15. Install the packages:

> **Your GPU driver must already be installed on the offline machine**
>
> The GPU driver should already be installed before you proceed with this step. If you have not already installed it, refer to Installing GPU Drivers on Linux Machines for more information.

- If you will support USB devices *and you downloaded usb-vhci in step 5*:

```
sudo yum --disablerepo=* --enablerepo=offline-agent install -y pcoip-agent-graphics usb-vhci
```

- If you will *only* allow keyboards, mice, and pointer devices to connect via USB, omit `usb-vhci` .

```
sudo yum --disablerepo=* --enablerepo=offline-agent install -y pcoip-agent-graphics
```

16. Reboot the offline machine:

```
sudo reboot
```

BETA

Licensing The Graphics Agent for Linux

The Graphics Agent for Linux must be assigned a valid PCoIP session license before it will work. Until you've registered it, you can't connect to the desktop using a PCoIP client.

You receive a registration code when you purchase a pool of licenses from Teradici. Each registration code can be used multiple times; each use consumes one license in its pool.

> **Note: Registration code format**
>
> Registration codes look like this: `ABCDEFGH12@AB12-C345-D67E-89FG`

PCoIP agent license registrations are managed automatically by Teradici's Cloud Licensing service. If necessary, you can manage them yourself, using your own locally-installed PCoIP license server instead.

If you need to purchase licenses, contact Teradici.

Troubleshooting Licensing Issues

If you're encountering problems with Teradici licensing, refer to Troubleshooting License Issues.

Using Teradici Cloud Licensing

To use Cloud Licensing, all you need to do is provide a registration code for each PCoIP agent in your deployment (the same registration code can be used multiple times).

To provide the registration code:

SSH into the agent machine, and invoke `pcoip-register-host` with the license registration code and proxy settings if required:

```
pcoip-register-host --registration-code=<registration-code> [--proxy-
server=<proxy-server-address>] [--proxy-port=<proxy-port-number>]
```

© 2021 Teradici

Whitelist network blocks for Teradici Cloud Licensing

If you are using Teradici Cloud Licensing, you will need to whitelist the following:

- teradici.flexnetoperations.com
- teradici.compliance.flexnetoperations.com

Alternatively, you can also ensure the following network blocks are whitelisted:

- **Production**: 64.14.29.0/24
- **Disaster Recovery**: 64.27.162.0/24

The following network blocks are not currently in use, but may also be used in the future:

- **Production**: 162.244.220.0/24
- **Disaster Recovery**: 162.244.222.0/24

Licensing PCoIP Agents With a Local License Server

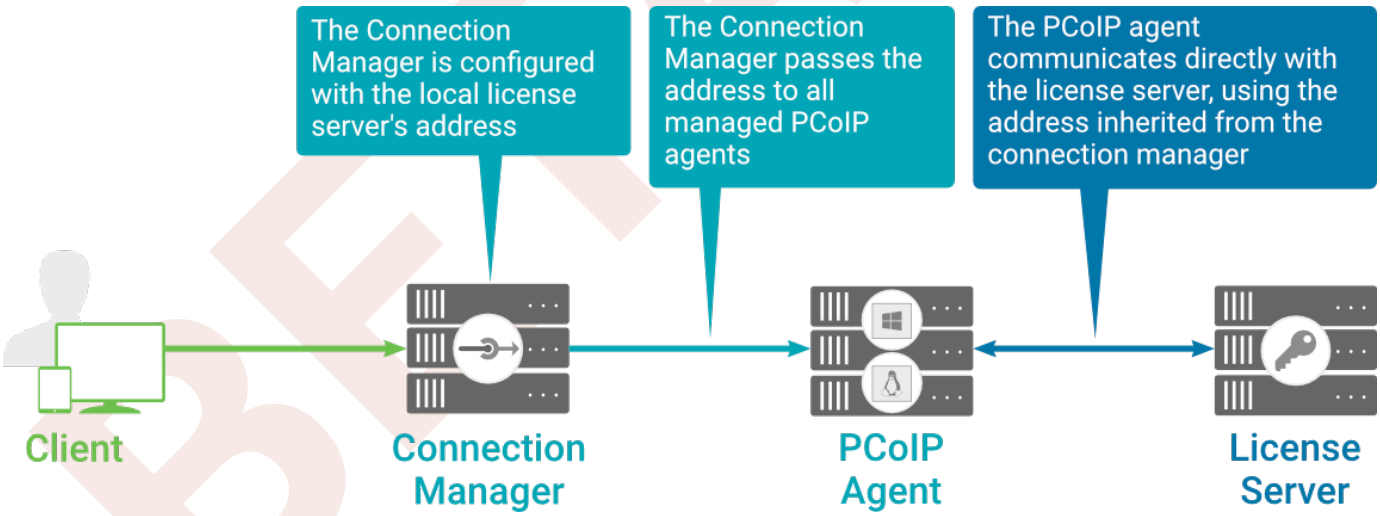
In deployments where PCoIP agents cannot access the internet, or where cloud-based licensing is not permitted or desired, a local PCoIP License Server can be used instead. The PCoIP License Server manages PCoIP session licenses within your private environment.

Configuring PCoIP agents to use a local license server is done in one of two ways, depending on whether your deployment uses a PCoIP Connection Manager, or whether your PCoIP clients connect directly to PCoIP agents.

Brokered Environment Licensing

In ***brokered*** deployments, the license server address is configured in the Connection Manager, which passes it through to its managed PCoIP agents.

BETA



Local license validation using a Graphics Agent for Linux and a brokered connection

When using a Connection Manager, the license server address is only configured once no matter how many PCoIP agents are behind the Connection Manager.

To set the License Server URL in the Connection Manager:

1. On the Connection Manager machine, use a text editor to open /etc/ConnectionManager.conf.
2. Set the `LicenseServerAddress` parameter with the address of your local license server:
   - `http://{license-server-address}:{port}/request`
3. Save and close the configuration file.
4. Restart the Connection Manager.

Verifying Your Brokered Licensing Configuration

To verify your system's licensing configuration, run `pcoip-validate-license` from the console on the Graphics Agent for Linux machine. The command will ping the license server and attempt to retrieve information on an available license:

```
pcoip-validate-license --license-server-url <license-server-address> [--proxy-server <proxy-server-address>] [--proxy-port <proxy-port-number]
```

Where `<license-server-address>` is the address of the license server to ping, formatted as `http://{license-server-address}:{port}/request`

If the license server is behind a proxy server, provide the proxy information via the `--proxy-server` and `--proxy-port` parameters.

© 2021 Teradici

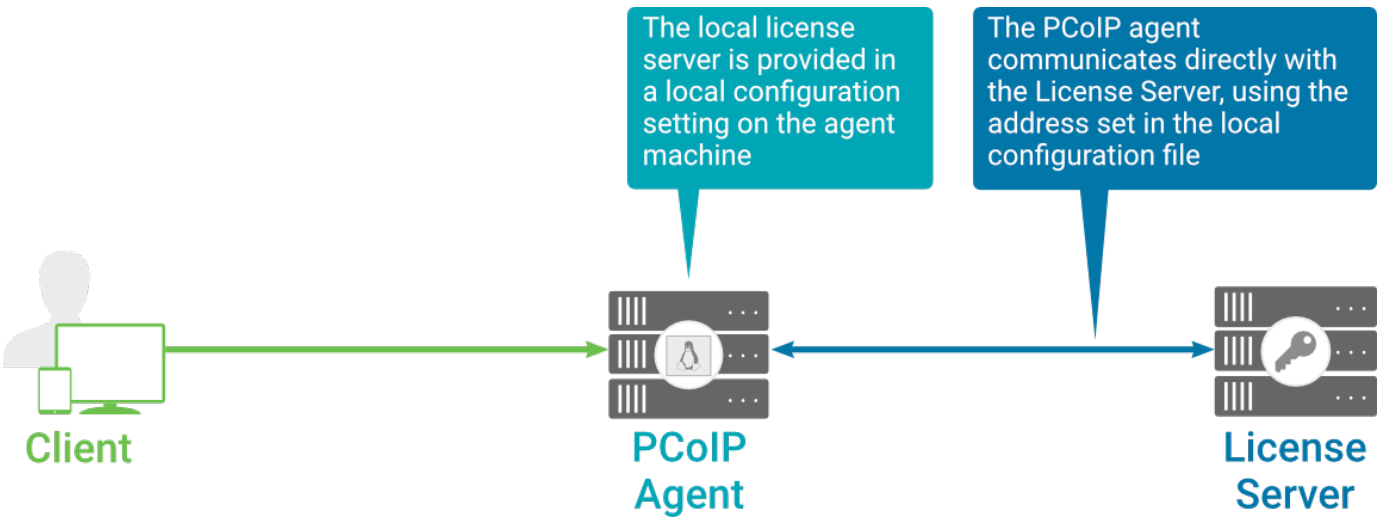
If successful, the response will show that a valid license was found on the license server, and its expiration date.

If the connection is unsuccessful, investigate the following possibilities:

- The license server address is incorrect, or formatted incorrectly.
- The license server is inaccessible.
- There are no available licenses on the license server. `pcoip-validate-license` will only return a positive response if there is at least one available session license.
- If you have only one license on the license server and run `pcoip-validate-license` from a PCoIP session, the command will fail because you are currently using the single license. In this scenario, disconnect your PCoIP session and try again from an SSH session instead.

Unbrokered Environment Licensing

In direct, or unbrokered, deployments, each PCoIP agent is configured with the license server address via a local agent setting. When a client initiates a new PCoIP session, the PCoIP agent uses its local configuration to communicate with the license server.



Local license validation using a Graphics Agent for Linux and a direct (unbrokered) connection

Each PCoIP agent in your environment must be individually configured with the license server's URL.

BETA

To configure the License Server URL on the Graphics Agent for Linux machine:

1. Using a text editor, open `/etc/pcoip-agent/pcoip-agent.conf`.
2. Add or modify the `pcoip.license_server_path` directive:

   ```
   pcoip.license_server_path = <license-server-address>
   ```

   Where `<license-server-address>` is the address of the license server, formatted as `http://{license-server-address}:{port}/request`.
3. If the license server is behind a proxy server, provide the proxy information using the `pcoip.license_proxy_server` and `pcoip.license_proxy_port` directives.
4. Save and close `pcoip-agent.conf`.

The changes will take effect on the next PCoIP session.

Verifying Your Unbrokered Licensing Configuration

To verify your system's licensing configuration, run `pcoip-validate-license` from the console on the Graphics Agent for Linux machine. The command will ping the license server and attempt to retrieve information on an available license:

```
pcoip-validate-license --license-server-url <license-server-address> [--proxy-
server <proxy-server-address>] [--proxy-port <proxy-port-number]
```

Where `<license-server-address>` is the address of the license server to ping, formatted as `http://{license-server-address}:{port}/request`

If the license server is behind a proxy server, provide the proxy information via the `--proxy-server` and `--proxy-port` parameters.

If successful, the response will show that a valid license was found on the license server, and its expiration date.

If the connection is unsuccessful, investigate the following possibilities:

- The license server address is incorrect, or formatted incorrectly.
- The license server is inaccessible.

BETA

- There are no available licenses on the license server. `pcoip-validate-license` will only return a positive response if there is at least one available session license.
- If you have only one license on the license server and run `pcoip-validate-license` from a PCoIP session, the command will fail because you are currently using the single license. In this scenario, disconnect your PCoIP session and try again from an SSH session instead.

BETA

Updating the Graphics Agent for Linux on RHEL or CentOS

Updates to the Graphics Agent for Linux will be published on a regular basis. New stable builds will be produced approximately every three months.

To upgrade to the latest version, use the following three commands:

```
sudo yum makecache
sudo yum update pcoip-agent-graphics
sudo reboot
```

BETA

Uninstalling the Graphics Agent for Linux

BETA

You can remove the Graphics Agent for Linux from your system, or you can remove the repo config entirely.

Remove the Graphics Agent for Linux package

To remove the package, open a console window and run the following command:

```
sudo yum remove pcoip-agent-*
```

Remove the repo configuration

If you want to remove the repo configuration completely, you can do that as well. You'll need to do this if you are switching from one channel to another (for example, from beta to stable), before reconfiguring with the new repo:

```
rm /etc/yum.repos.d/pcoip-agent-beta
.repo
rm /etc/yum.repos.d/pcoip-agent-beta
-source.repo
```

© 2021 Teradici

Configuring the PCoIP Agent

You can configure the PCoIP agent, and optimize PCoIP protocol behavior for local network conditions, by adjusting configuration directives found in **`/etc/pcoip-agent/pcoip-agent.conf`**.

You can find detailed information and descriptions about each setting in the next section. You can also consult the `man` pages for **`pcoip-agent.conf`**:

```
man pcoip-agent.conf
```

Applying Configuration Changes

To set or change a configuration value, add or modify directives in **`pcoip-agent.conf`**. Place one directive on each line, in this format:

```
directive.name = <value>
```

For example, to set the maximum frame rate to *60 frames per second*, set the maximum bandwidth to *900000 kilobits/second*, and the device bandwidth floor to *5000 kilobits/second*, you would set values in **`pcoip-agent.conf`** like this:

```
pcoip.maximum_frame_rate = 60
pcoip.max_link_rate = 900000
pcoip.device_bandwidth_floor = 5000
```

A complete list of configurable values is shown next in Configurable Settings.

Configurable Settings

The following settings can be configured on the Graphics Agent for Linux. Refer to Configuring the PCoIP agent to understand how to modify these settings.

© 2021 Teradici

Build-to-lossless

Directive	Options	Default
`pcoip.enable_build_to_lossless`	**0** (off), **1** (on)	Off

This setting takes effect immediately. Specifies whether to turn the build-to-lossless feature of the PCoIP protocol off or on; this feature is turned off by default.

When build-to-lossless is turned off images and other desktop content may never build to a lossless state. In network environments with constrained bandwidth, turning off build-to-lossless can provide bandwidth savings. Build-to-lossless is recommended for environments that require images and desktop content to be built to a lossless state.

Clipboard redirection

Directive	Options	Default
`pcoip.server_clipboard_state`	**0**—Disabled in both directions **1**—Enabled in both directions **2**—Enabled client to agent only **3**—Enabled agent to client only	—

This setting takes effect when you start the next session. Determines the direction in which clipboard redirection is allowed. You can select one of these values:

- Disabled in both directions
- Enabled in both directions (default setting)
- Enabled client to agent only (That is, allow copy and paste only from the client system to the host desktop.)
- Enabled agent to client only (That is, allow copy and paste only from the host desktop to the client system.)

Clipboard redirection is implemented as a virtual channel. If virtual channels are disabled, clipboard redirection does not function.

BETA

Connection addresses

Directive	Options	Default
`pcoip.connection_address`	string *(up to 511 characters)*	—
`pcoip.client_connection_address`	string *(up to 511 characters)*	—

This setting takes effect when you start the next session. Configuring this allows you to control the IPv4 or IPv6 address used by the agent or client in PCoIP sessions.

'Connection Address' controls the IP address used by the agent for the PCoIP session.

'Client Connection Address' controls the IP address the client is told to use when establishing the PCoIP session.

Please note that neither of these values should need to be set under normal circumstances.

Desktop environment

Directive	Options	Default
`pcoip.desktop_session`	string *(up to 511 characters)*	—

This setting takes effect when you start the next session. Choose the desktop environment that will be launched from */usr/share/xsessions/*.desktop*, defaults to "kde-plasma" if present, else the first session found alphabetically in */usr/share/xsessions*.

Note that this setting only takes effect after an existing desktop session ends (either due to a reboot or logging out).

BETA

Enable Disclaimer Authentication

Directive	Options	Default
`pcoip.enable_disclaimer_auth`	**0** (off), **1** (on)	Off

This setting takes effect when you start the next session. When this setting is enabled, users connecting via direct connect will be presented a disclaimer prior to user authentication. If the disclaimer is rejected, the user will not be able to connect.

Disclaimer files must be placed in ***/etc/pcoip-agent/disclaimers/*** and must be readable by the "pcoip" system user. Files must be named according to the locale, e.g. en_US.txt for en_US, ko_KR.txt for ko_KR, etc. If a file matching the negotiated locale is not present, en_US will be used as a fallback. If disclaimer text cannot be found, an blank disclaimer will be presented.

Enable/disable USB in the PCoIP session

Directive	Options	Default
`pcoip.enable_usb`	**0** (off), **1** (on)	On

This setting takes effect when you start the next session. Determines whether USB support is enabled in PCoIP sessions. When this setting is not configured, USB is enabled by default. By default all devices are supported unless restrictions are configured through the USB device rules setting.

Enable/disable audio in the PCoIP session

Directive	Options	Default
`pcoip.enable_audio`	**0** (off), **1** (on)	On

This setting takes effect when you start the next session. Determines whether audio is enabled in PCoIP sessions. Both endpoints must have audio enabled. When this setting is enabled, PCoIP

BETA

audio is allowed. When it is disabled, PCoIP audio is disabled. When this setting is not configured, audio is enabled by default.

Hide local cursor

Directive	Options	Default
`pcoip.disable_locally_rendered_cursor`	**0** (off), **1** (on)	Off

This setting takes effect immediately. When this setting is enabled the local cursor on the client will be hidden. This may resolve duplicate cursor issues if there is a host rendered cursor within the host environment but may also result in no visible cursor. With this setting enabled there may be delays in mouse movements due to network latency and video processing times. By default, this setting is disabled, meaning that local cursors will be used, providing the most responsive user experience.

Host key auto repeats

Directive	Options	Default
`pcoip.use_host_autorepeat`	**0** (off), **1** (on)	Off

This setting takes effect when you start the next session. Configuring this allows you to enable or disable host generated key auto repeats. When not configured or disabled, key auto repeats are driven by the client.

License server URL

Directive	Options	Default
`pcoip.license_server_path`	string *(up to 511 characters)*	—

This setting takes effect when you start the next session. This policy sets the license server path. Enter the license server path in 'https://address:port/request' or 'http://address:port/request' format.

© 2021 Teradici

Maximum PCoIP session bandwidth

Directive	Range	Increment	Default
`pcoip.max_link_rate`	104 – 900000	100	900000

This setting takes effect when you start the next session. Specifies the maximum bandwidth, in kilobits per second, in a PCoIP session. The bandwidth includes all imaging, audio, virtual channel, USB, and control PCoIP traffic.

Set this value based on the overall capacity of the link to which your endpoint is connected, taking into consideration the number of expected concurrent PCoIP sessions. For example, with a single user VDI configuration (e.g. a single PCoIP session) that connects through a 4Mbit/s Internet connection, set this value to 4Mbit (or 10% less than this value to leave some allowance for other network traffic).

Setting this value prevents the agent from attempting to transmit at a higher rate than the link capacity, which would cause excessive packet loss and a poorer user experience. This value is symmetric. It forces the client and agent to use the lower of the two values that are set on the client and agent side. For example, setting a 4Mbit/s maximum bandwidth forces the agent to transmit at a lower rate, even though the setting is configured on the client.

When this setting is disabled or not configured on an endpoint, the endpoint imposes no bandwidth constraints. When this setting is configured, the setting is used as the endpoint's maximum bandwidth constraint in kilobits per second.

The default value when this setting is not configured is 900000 kilobits per second.

This setting applies to the agent and client. If the two endpoints have different settings, the lower value is used.

BETA

PCoIP Security Certificate Settings

Directive	Options	Default
`pcoip.ssl_cert_type`	**1**—From certificate storage **2**—Generate a unique self-signed certificate **0**—From certificate storage if possible, otherwise generate	—
`pcoip.ssl_cert_min_key_length`	**1024**—1024 bits **2048**—2048 bits **3072**—3072 bits **4096**—4096 bits	—

This setting takes effect when you start the next session. A certificate is used to secure PCoIP related communications. The way PCoIP components choose a certificate is based on the certificate type and the key length. Without a certificate being generated or selected, a PCoIP Session cannot be established.

Depending on the value chosen for the option, 'How the PCoIP agent chooses the certificate...' and the availability of appropriate certificates, PCoIP components may acquire a CA signed certificate from certificate storage or generate an in-memory self-signed certificate.

In order for a CA signed certificate to be loadable by PCoIP components, it must be stored at ***/etc/pcoip-agent/ssl-certs*** in three .pem files, owned by the pcoip user, only readable by the owning user.

- pcoip-key.pem must contain an unlocked RSA key
- pcoip-cert.pem must contain a certificate that signs the key in pcoip.pem
- pcoip-cacert.pem must contain a CA certificate chain that validates the certificate in pcoip-cert.pem.

Note: Self-signed certificates are 3072 bits long.

Select a minimum key length (in bits) for a CA signed certificate. Longer length certificates will require more computing resources and may reduce performance, but will increase security. Shorter length certificates will provide better performance at the cost of lower security.

BETA

Note: Please refer to Teradici documentation for instructions on creating and deploying certificates.

PCoIP Security Settings

Directive	Options	Default
`pcoip.tls_security_mode`	**0**—Maximum Compatibility	—
`pcoip.tls_cipher_blacklist`	string *(up to 1023 characters)*	—
`pcoip.data_encryption_ciphers`	**6**—AES-256-GCM, AES-128-GCM (default, AES-256-GCM preferred) **4**—AES-256-GCM only **2**—AES-128-GCM only	—

This setting takes effect when you start the next session. Controls the cryptographic cipher suites and encryption ciphers used by PCoIP endpoints.

The endpoints negotiate the actual cryptographic cipher suites and encryption ciphers based on the settings configured here. Newer versions of TLS and stronger cipher suites will be preferred during negotiation between endpoints.

If this setting is not configured or disabled, the TLS Security Mode will be set to Maximum Compatibility, and the PCoIP Data Encryption Ciphers will be set to AES-256-GCM, AES-128-GCM.

TLS Security Mode

Maximum Compatibility offers TLS 1.2 and TLS 1.3, and a range of cipher suites including those that support Perfect Forward Secrecy (PFS) and SHA-1. Supported cipher suites:

- 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

BETA

- 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

Blacklisted Cipher Suites

Provides the ability to block specific cipher suites from being offered during negotiation. Must be entered as a semi-colon separated list of cipher suites.

PCoIP Data Encryption Ciphers

Encryption ciphers used for PCoIP UDP data encryption. "AES-256-GCM, AES-128-GCM" is the default setting. AES-256-GCM will get negotiated if the client supports it, otherwise, AES-128-GCM will get negotiated.

PCoIP USB allowed and unallowed device rules

Directive	Options	Default
`pcoip.usb_auth_table`	23XXXXXX 2203XXXX	23XXXXXX
`pcoip.usb_unauth_table`	2203XXXX	—

This setting takes effect when you start the next session. This setting only applies if the related setting to Enable/disable USB in the PCoIP session is enabled, and specifies the USB devices that are authorized and not authorized for PCoIP sessions. Only devices listed in the USB authorization table are permitted in PCoIP sessions, provided they are not subsequently excluded by an entry in the USB unauthorization table.

You can define a maximum of 10 USB authorization rules and a maximum of 10 USB unauthorization rules. Separate multiple rules with the vertical bar (|) character.

BETA

BETA

Each rule can be a combination of a Vendor ID (VID) and a Product ID (PID), or a rule can describe a class of USB devices. A class rule can allow or disallow an entire device class, a single subclass, or a protocol within a subclass.

The format of a combination VID/PID rule is 1xxxxyyyy, where xxxx is the VID in hexadecimal format and yyyy is the PID in hexadecimal format. For example, the rule to authorize or block a device with VID 0x1a2b and PID 0x3c4d is 11a2b3c4d.

For class rules, use one of the following formats:

Allow all USB Format: 23XXXXXX devices Example: 23XXXXXX

Allow USB Format: 22classXXXX devices with a Example: 22aaXXXX specific class ID

Allow a specific Format: 21class-subclassXX subclass Example: 21aabbXX

Allow a specific Format: 20class-subclass-protocol protocol Example: 20aabbcc

For example, the USB authorization string to allow USB HID (mouse and keyboard) devices (class ID 0x03) and mass storage devices (class ID 0x08) is 2203XXXX|2208XXXX. The USB unauthorization string to disallow USB Mass Storage devices (class ID 0x08) is 2208XXXX.

An empty USB authorization string means that no USB devices are authorized. An empty USB unauthorization string means that only USB devices in the authorization list are allowed.

If these settings are unconfigured, the default behavior is that all devices are allowed.

PCoIP Ultra

Directive	Options	Range	Increment	Default
`pcoip.ultra`	0—Disabled 1—CPU Offload 2—GPU Offload 3—Automatic Offload			Off
`pcoip.ultra_offload_mpps`		1 – 40	1	10

This setting takes effect when you start the next session. When this setting is disabled or not configured then PCoIP Ultra will not be used.

- PCoIP Ultra CPU Offload - these optimizations require CPU support for the AVX2 instruction set on both the remote host and client and are not compatible with the PCoIP Zero client. CPU Offload is recommended for 4K UHD resolutions with video playback requirements of 30 fps (or more) and highest image quality / color accuracy.
- PCoIP Ultra GPU Offload - these optimizations require an NVIDIA graphics card on the remote host capable of NVENC. GPU Offload is recommended when minimal CPU impact of pixel encoding is desired.
- PCoIP Ultra Auto Offload - enabling this setting allows PCoIP to automatically switch between CPU and GPU Offload modes; CPU Offload is used by default to provide the best image fidelity, GPU Offload is used during periods of high display activity to provide improved frame rates and bandwidth optimization. This setting is only effective if the remote host and client endpoints are capable of both CPU and GPU Offload.

The PCoIP Ultra Offload MPPS sets the Megapixels Per Second (MPPS) transition rate between PCoIP Ultra CPU Offload and PCoIP Ultra GPU Offload. Under Auto-Offload, PCoIP Ultra uses CPU Offload at lower pixel rates and switches to GPU Offload at the Offload MPPS. Increasing this value results in PCoIP Ultra transitioning to GPU Offload at a higher pixel rate and decreasing this value results in the transition at a lower pixel rate. The default PCoIP Ultra Offload MPPS is set to 10.

PCoIP event log verbosity

Directive	Range	Increment	Default
`pcoip.event_filter_mode`	0 – 3	1	2

This setting takes effect immediately. Configures the PCoIP event log verbosity ranging from 0 (least verbose) to 3 (most verbose).

BETA

PCoIP image quality levels

Directive	Options	Range	Increment	Default
`pcoip.minimum_image_quality`		30 – 100	10	40
`pcoip.maximum_initial_image_quality`		30 – 100	10	80
`pcoip.frame_rate_vs_quality_factor`		0 – 100	10	50
`pcoip.maximum_frame_rate`		0 – 120	1	—
`pcoip.yuv_chroma_subsampling`	**0**—4:4:4 **1**—4:2:0			—
`pcoip.use_client_img_settings`	**0** (off), **1** (on)			Off

This setting takes effect immediately. Controls how PCoIP renders images during periods of network congestion. The Minimum Image Quality, Maximum Initial Image Quality, and Maximum Frame Rate values interoperate to provide fine control in network-bandwidth constrained environments.

Use the Minimum Image Quality value to balance image quality and frame rate for limited-bandwidth scenarios. You can specify a value between 30 and 100. The default value is 40. A lower value allows higher frame-rates, but with a potentially lower quality display. A higher value provides higher image quality, but with potentially lower frame rates when network bandwidth is constrained. When network bandwidth is not constrained, PCoIP maintains maximum quality regardless of this value.

Use the Maximum Initial Image Quality value to reduce the network bandwidth peaks required by PCoIP by limiting the initial quality of the changed regions of the display image. You can specify a value between 30 and 100. The default value is 80. A lower value reduces the image quality of content changes and decreases peak bandwidth requirements. A higher value increases the image quality of content changes and increases peak bandwidth requirements. Unchanged regions of the image progressively build to a lossless (perfect) quality regardless of this value. A value of 80 or lower best utilizes the available bandwidth.

BETA

BETA

The Minimum Image Quality value cannot exceed the Maximum Initial Image Quality value.

Use the Frame Rate vs Image Quality value to favor image sharpness over smooth motion during a PCoIP session when network bandwidth is limited. Lower values favor smoothness, higher values favor sharpness of image.

Use the Maximum Frame Rate value to manage the average bandwidth consumed per user by limiting the number of screen updates per second. You can specify a value between 1 and 120 frames per second. The default value is 30 for PCoIP Standard Agent and 60 for PCoIP Graphics Agent. A higher value can use more bandwidth but provides less jitter, which allows smoother transitions in changing images such as video. A lower value uses less bandwidth but results in more jitter.

YUV chroma subsampling is set to 4:4:4 by default for maximum image quality. Setting YUV chroma subsampling to 4:2:0 is only supported in combination with PCoIP Ultra GPU optimization. This setting will enable chroma subsampling to further compress the imaging to reduce bandwidth usage at the cost of reduced color accuracy. Please note: 4:4:4 subsampling with PCoIP Ultra GPU optimization is GPU dependent and is not supported by all GPUs, in this case PCoIP will fallback to 4:2:0 subsampling. Please see our support site for further details.

Set the 'Use image settings from zero client' when you want to use the 'Minimum Image Quality', 'Maximum Initial Image Quality', 'Maximum Frame Rate', 'Disable Build to Lossless' values from the client instead of the host. Currently, only Zero Client Firmware 3.5 and above support these settings on the client side.

These image quality values apply to the soft host only and have no effect on a soft client.

When this setting is disabled or not configured, the default values are used.

PCoIP session MTU

Directive	Range	Increment	Default
`pcoip.mtu_size`	500 – 1500	1	1200

This setting takes effect when you start the next session. Specifies the Maximum Transmission Unit (MTU) size for UDP packets for a PCoIP session.

© 2021 Teradici

The MTU size includes IP and UDP packet headers. TCP uses the standard MTU discovery mechanism to set MTU and is not affected by this setting. The maximum MTU size is 1500 bytes. The minimum MTU size is 500 bytes. The default value is 1200 bytes.

Typically, you do not have to change the MTU size. Change this value if you have an unusual network setup that causes PCoIP packet fragmentation.

This setting applies to the agent and client. If the two endpoints have different MTU size settings, the lowest size is used.

If this setting is disabled or not configured, the client uses the default value in the negotiation with the agent.

PCoIP session audio bandwidth limit

Directive	Range	Increment	Default
`pcoip.audio_bandwidth_limit`	0 – 100000	1	500

This setting takes effect immediately. Specifies the maximum audio bandwidth that can be used for audio output (sound playback) from the virtual desktop to the client in a PCoIP session. Note that the network transport overhead can add an additional 20-40% bandwidth to this number.

Audio processing monitors the bandwidth needed for audio and selects the audio compression algorithm that provides the best quality possible, without exceeding the bandwidth limit:

- 256 kbit/s or higher - stereo, high-quality, compressed audio
- 48 kbit/s to 255 kbit/s - stereo audio ranging between FM radio quality down to AM radio quality
- 32 kbit/s to 47 kbit/s - monaural AM radio or phone call quality
- Below 32 kbit/s - results in no audio playback

If this setting is not configured, a default audio bandwidth limit of 256 kbit/s is configured to constrain the audio compression algorithm selected.

Note that zero clients on older firmware have less efficient audio compression algorithms that may require setting this limit higher to achieve the same audio quality or upgrading the firmware.

BETA

PCoIP session bandwidth floor

Directive	Range	Increment	Default
`pcoip.device_bandwidth_floor`	0 – 100000	1	—

This setting takes effect immediately. Specifies a lower limit, in kilobits per second, for the bandwidth that is reserved by the PCoIP session.

This setting configures the minimum expected bandwidth transmission rate for the endpoint. When you use this setting to reserve bandwidth for an endpoint, the session does not have to wait for bandwidth to become available, which improves session responsiveness.

Make sure that you do not over-subscribe the total reserved bandwidth for all endpoints. Make sure that the sum of bandwidth floors for all connections in your configuration does not exceed the network capability.

The default value is 0, which means that no minimum bandwidth is reserved. When this setting is disabled or not configured, no minimum bandwidth is reserved.

This setting applies to the agent and client, but the setting only affects the endpoint on which it is configured.

PCoIP statistics interval

Directive	Range	Increment	Default
`pcoip.server_statistics_interval_seconds`	0 – 65535	1	—

This setting takes effect immediately. Configuring this allows you to set an interval in seconds for logging performance statistics to the PCoIP server log. When not configured, logging is disabled by default.

BETA

PCoIP transport header

Directive	Options	Default
`pcoip.transport_session_priority`	1—High Priority 2—Medium Priority (default) 3—Low Priority 4—Undefined Priority	—

This setting takes effect when you start the next session. Configures the PCoIP transport header.

PCoIP transport header is a 32-bit long header which is added to all PCoIP UDP packets (only if the transport header is enabled/supported by both sides). PCoIP transport header allows network devices to make better prioritization/Qos decisions when dealing with network congestions. The transport header is enabled by default.

The transport session priority determines the PCoIP session priority reported in the PCoIP Transport Header. Network devices make better prioritization/Qos decisions based on the specified transport session priority. The transport session priority value is negotiated by the PCoIP agent and client. If agent has specified a transport session priority value (high, medium, or low), then the session uses the agent specified session priority. If only the client has specified a transport session priority (high, medium, or low), then the session uses the client specified session priority. If neither agent nor client has specified a transport session priority (or specified 'undefined priority'), then the session uses/defaults to the medium session priority.

PCoIP virtual channels

Directive	Options	Default
`pcoip.enable_vchan`	1—Enable all virtual channels other than those in the list 2—Disable all virtual channels other than those in the list	—
`pcoip.vchan_list`	string *(up to 255 characters)*	—

This setting takes effect when you start the next session. Specifies the virtual channels that can or cannot operate over a PCoIP session.

© 2021 Teradici

BETA

There are two modes of operation:

- Enable all virtual channels except for <list> (default setting)
- Disable all virtual channels except for <list>

When specifying which virtual channels to include or not include in the list, the following rules apply:

- An empty list is allowed
- Multiple virtual channel names in the list must be separated by the vertical bar (|) character. For example: channelA|channelB
- Vertical bar or backslash () characters in virtual channel names must be preceded by a backslash. For example: the channel name "awk|ward\channel" must be specified as "awk|ward\channel" (without the double quotes)
- A maximum of 15 virtual channels are allowed in a single PCoIP session

The virtual channel must be enabled on both agent and client for it to be used.

Proxy Access to a remote License Server

Directive	Options	Range	Increment	Default
`pcoip.license_proxy_server`	string *(up to 511 characters)*			—
`pcoip.license_proxy_port`		0 – 65535	1	—

This setting takes effect when you start the next session. If a proxy is required to access a local License Server or the Cloud License Server, enter those parameters here. These parameters are loaded only during agent startup.

Timezone redirection

Directive	Options	Default
`pcoip.enable_timezone_redirect`	**0** (off), **1** (on)	On

© 2021 Teradici

This setting takes effect when you start the next session. Configuring this allows you to enable or disable timezone redirection. When not configured, timezone redirection is enabled by default.

X server remote access

Directive	Options	Default
`pcoip.allow_x_remoting`	**0** (off), **1** (on)	Off

This setting takes effect when you restart the agent. Configuring this allows you to enable or disable remote access to the X server run by the PCoIP Agent. When not configured, remote access is disabled by default.

BETA

Making a Connection from a PCoIP Client

Once you've installed and configured your Graphics Agent for Linux, you're ready to accept incoming connections from remote ***PCoIP Clients***. PCoIP clients are remote endpoint devices available in as software or firmware and make secure PCoIP connections to the remote desktop through the installed Graphics Agent for Linux.

For more information about PCoIP client connectivity requirements and usage instructions, see the following documentation:

- Software clients:
  - Teradici PCoIP Software Client for Windows
  - Teradici PCoIP Software Client for macOS
  - Teradici PCoIP Software Client for Linux
- Mobile Clients:
  - Teradici PCoIP Mobile Client for iOS
  - Teradici PCoIP Mobile Client for Android
  - Teradici PCoIP Mobile Client for Chromebooks
- Zero clients:
  - Teradici Tera2 PCoIP Zero Client

PCoIP Agent Deployment and Client Connectivity Requirements

PCoIP clients can connect to your desktops hosted in proof-of-concept, cloud, or datacenter deployments. Requirements and network security levels will vary depending on your deployment type. See Supported PCoIP Architectures for each deployment's components and requirements.

© 2021 Teradici

BETA

Managing Client Connections

In most cases, PCoIP clients connect to PCoIP agents through a ***connection broker***. The broker is responsible for matching users to their available desktops, and then establishing the PCoIP session with their selected resource.

PCoIP agents do not need to be configured to use these brokering services. All relevant configuration is done at the broker, which then communicates with the agent.

Brokering Options

There are several ways you can manage client connections to remote desktops

Direct Connections

In direct connection scenarios—where a broker is not involved—the PCoIP agent acts as its own broker. In these cases, a client user will provide the IP address or FQDN of the agent machine to their client, and the connection is made securely with no intermediate step.

Teradici Cloud Access Manager

Teradici Cloud Access Manager is a cloud-based service available as part of Cloud Access Software that centrally manages PCoIP deployments. It enables highly scalable and cost-effective Cloud Access Software deployments by managing cloud compute costs and brokering PCoIP connections to remote Windows or Linux workstations.

Teradici PCoIP Connection Manager

The **Teradici PCoIP Connection Manager** is provided in a bundle with the **Teradici PCoIP Security Gateway**, and allows self-managed brokering services. For information about the Teradici PCoIP Connection Manager, including installation and configuration instructions, see the Connection Manager and Security Gateway documentation.

© 2021 Teradici

Third-party Connection Brokers

Teradici PCoIP agents also support third-party connection brokers. For a current list of brokering partners, see Teradici Technology Partners on Teradici's website.

BETA

Security Certificates in PCoIP Agents

PCoIP requires a certificate to establish a session. By default, PCoIP agents generate a self-signed certificate that secures the PCoIP session. Each component in the PCoIP system can generate these self-signed certificates, which will automatically work together without requiring any configuration.

You can, if needed, create and deploy your own custom certificates instead of relying on Teradici's self-signed certificates. This section explains how to create and implement custom certificates.

Using Custom Security Certificates

You can use OpenSSL, Microsoft Certification Authority, or a public certificate authority (CA) of your choice to create your certificates. If you are not using OpenSSL, consult your certificate authority's documentation for instructions on creating certificates in a Windows Certificate Store-compatible format.

The procedures is this section use OpenSSL to generate certificates that will satisfy most security scanner tools when the root signing certificate is known to them.

> **Caution: Certificates are stored in the Windows Certificate Store**
>
> Certificates are stored in the Windows certificate store. If you have old certificates that are stored on the host, they should be deleted to avoid conflicts or confusion.

Custom Certificate Guidelines

If you choose to use your own certificates, follow these general guidelines:

- Save your root CA signing certificate in a safe place for deployment to clients.
- Back up private and public keys to secure locations.
- Never store files created when generating keys or certificates on network drives without password protection.

© 2021 Teradici

- Once certificates have been deployed to the Windows certificate store, the files they came from are no longer needed and can be deleted.
- Standard automatic tools, such as Automatic Certificate Enrollment and Group Policy, can be used for deploying automatically generated certificates. Both Automatic Certificate Enrollment and Group Policies are implemented through Active Directory. See MSDN Active Directory documentation for more information.

Pre-session Encryption Algorithms

Connections are negotiated using the following supported RSA cipher suites:

- 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

Note: Minimum SSL version

These Max Compatibility security level cipher suites have a minimum required SSL version of TLS 1.0.

BETA

BETA

Custom Security Certificates

In order for a CA signed certificate to be loadable by PCoIP components, it must be stored in `/etc/pcoip-agent/ssl-certs` in three **.pem** files, owned by the pcoip user, and only readable by the owning user:

- **pcoip-key.pem** must contain an unlocked RSA key
- **pcoip-cert.pem** must contain a certificate that signs the key in pcoip.pem
- **pcoip-cacert.pem** must contain a CA certificate chain that validates the certificate in pcoip-cert.pem

Configure the Graphics Agent for Linux to use custom certificates

The Graphics Agent for Linux can be configured to look locally for certificates or to generate its own by setting the `pcoip.ssl_cert_type` directive in `pcoip-agent.conf`.

For more detailed information, see Configuring the Agent.

Select a Security Key Length

When the Graphics Agent for Linux is attempting to find a certificate in storage, the required key length can be set via the `pcoip.ssl_cert_min_key_length` directive in `pcoip-agent.conf`.

If the system cannot find a local certificate with the specified key length, it will either self-generate a certificate (if `pcoip.ssl_cert_type` is 0), or refuse the connection (if `pcoip.ssl_cert_type` is 1). This setting has no effect if `pcoip.ssl_cert_type` is set to 2.

For more detailed information, see Configuring the Agent.

© 2021 Teradici

Using Wacom Local Termination on Ubuntu Cloud Hosts

Cloud-based Ubuntu hosts will fail to properly identify Wacom tablets that have been locally-terminated at the PCoIP client. When this occurs, pressure sensitivity and other advanced features will not work properly.

To work around this issue, remove the default AWS, Microsoft Azure, or Google Cloud kernel and replace it with a generic kernel.

> **Note: Ubuntu cloud hosts only**
>
> This procedure applies only to Ubuntu hosts on AWS, Microsoft Azure, or Google Cloud Platform. All valid RHEL and non-cloud Ubuntu installations work as expected.

To enable local termination:

1. First, confirm that you need to replace the kernel. Open a console and enter the following command:

   ```
   uname -r
   ```

   If the response contains the word `generic` (for example, `4.15.0-66-generic`) then your kernel is already generic and you can skip this procedure.

   If the response ends in `aws`, `azure`, or `gcp`, note the version number and continue.

2. Find the available `linux-virtual` package for your distribution. In a console window, enter the following command:

   ```
   apt-cache policy linux-virtual
   ```

   In the response, note the candidate major version number. For example, if the candidate's number is `4.15.0.66.68`, then the major version number is `4`.

© 2021 Teradici

3. Compare the major versions of the ***installed*** kernel from step 1, and the ***candidate*** kernel in step 2:

   - **If the major versions for the installed and candidate kernels are the same**

     In a console window, enter the following command:

     ```
     sudo apt install linux-virtual
     sudo apt install linux-cloud-tools-virtual
     ```

   - **If the major versions for the installed and candidate kernels are *not* the same**

     - Retrieve the full list of available kernels:

       ```
       apt-cache policy linux-virtual*
       ```

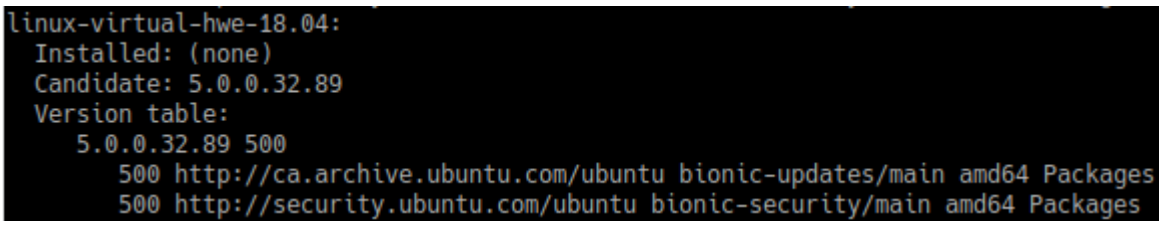
       Look through the output for the generic kernel version matching your installed kernel's major version.

     - Install the kernel and cloud tools packages for the correct version:

       ```
       sudo apt install linux-virtual-<version>
       sudo apt install linux-cloud-tools-virtual-<version>
       ```

       ...where `<version>` is the number reported in the output from `apt-cache policy linux-virtual*`.

       For example, if you needed to find a kernel with a major version of `5`, you would look through the output of `apt-cache policy linux-virtual*` and find a response similar to this one:

       The version is `hwe-18.04`. You will install that package and the corresponding cloud tools package:

       ```
       sudo apt install linux-virtual-hwe-18.04
       sudo apt install linux-cloud-tools-virtual-hwe-18.04
       ```

BETA

4. Purge the cloud-specific ubuntu image:
   - AWS:

     ```
     sudo apt purge linux*aws
     ```

   - Azure:

     ```
     sudo apt purge linux*azure
     ```

   - GCP

     ```
     sudo apt purge linux*gcp
     ```

5. When you see the *Abort kernel removal* message, respond with `No`.
6. Reboot the machine:

   ```
   sudo reboot
   ```

7. When the machine comes back up, reconnect and check that the generic kernel is in use:

   ```
   uname -r
   ```

   You should see a response ending in `-generic`.
8. Obtain the `uhid` driver by installing `linux-modules-extra` for your kernel version:

   ```
   sudo apt install linux-modules-extra-$(uname-r)
   ```

9. Reboot the machine:

   ```
   sudo reboot
   ```

10. When the machine comes back up, reconnect and check that the uhid driver is present:

    ```
    ls /dev/uhid
    ```

    You should see a response similar to `/dev/uhid`.

BETA

11. Install USB driver packages:

```
sudo apt install usb-vhci-dkms
```

12. Reboot the machine:

```
sudo reboot
```

13. When the machine comes back up, reconnect and check that the USB drivers are present:

```
lsmod | grep usb
```

You should see a response similar to this:

```
usb_vhci_iocifc        20480  3
usb_vhci_hcd           20480  1 usb_vhci_iocifc
```

> **Note: What if the response is empty?**
>
> If the output is empty, you may need to uninstall and reinstall the `vhci` package:
>
> ```
> sudo apt remove usb-vhci-dkms
> sudo reboot
> sudo apt install usb-vhci-dkms
> sudo reboot
> ```

14. Install the Wacom driver for your tablet.
15. Reboot the host machine.
16. If you have not installed the Graphics Agent for Linux, install it now.

BETA

Reference Information for VSphere Users

This section contains reference information useful for VSphere users configuring displays with the PCoIP Graphics Agent.

You can find information here for:

- Configuring multiple monitors for NVIDIA GRID vGPUs
- Configuring displays for NVIDIA GRID with pass-through

BETA

Configuring Multiple Monitors for NVIDIA GRID vGPU

This section describes how to add a shared NVIDIA GRID PCI device to your virtual machine and specify its GPU profile. The following instructions are only required for virtualized workstations using a supported NVIDIA GRID vGPU video card.

> **Note: Monitor limitations**
>
> The PCoIP Graphics Agent supports up to four displays. Your video card may limit the actual number of displays you can have.

Before you begin, ensure the following prerequisites are met:

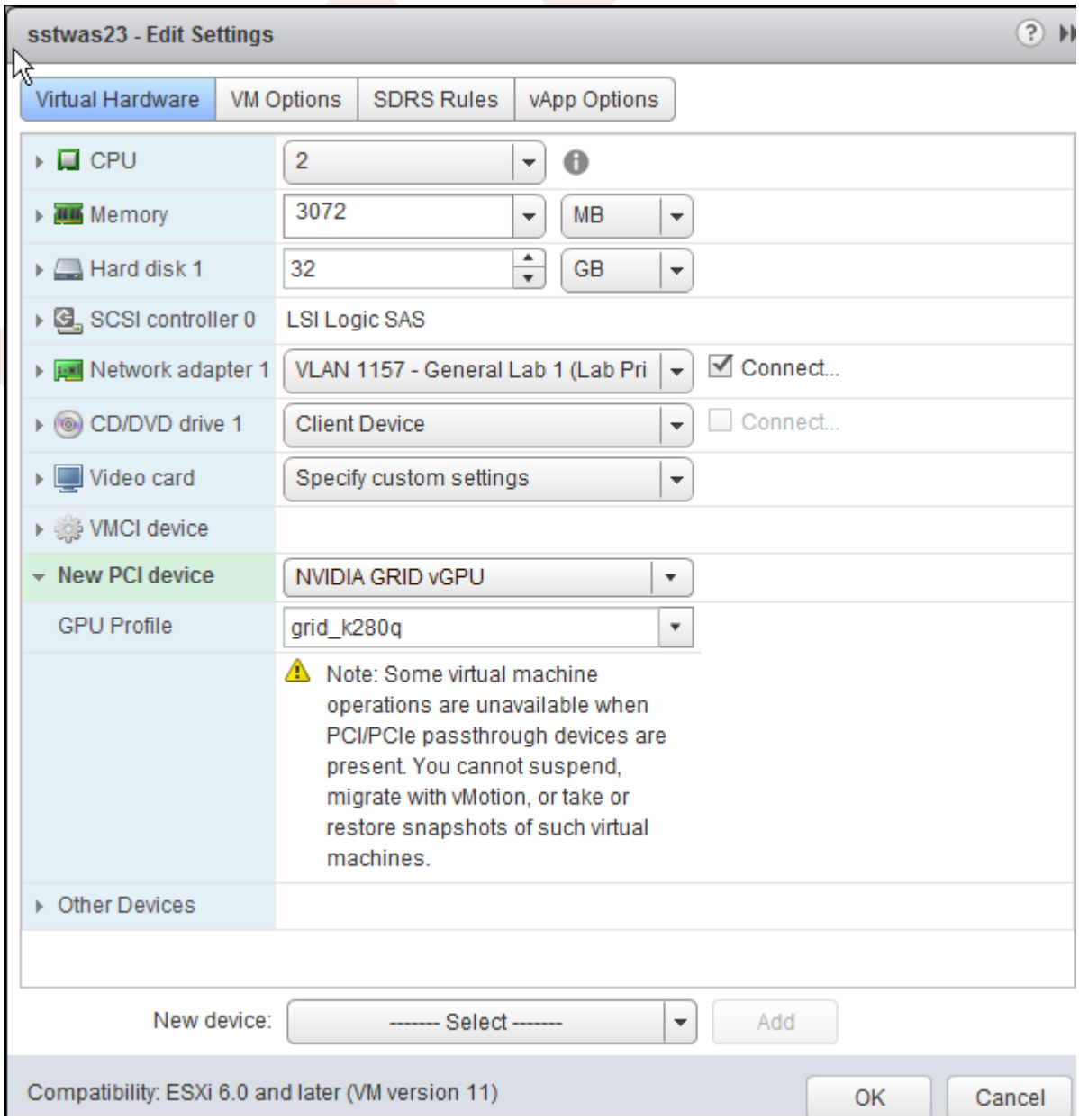
- You have installed the NVIDIA GRID graphics card and supported driver on the ESXi host.
- You have installed VMware Tools on your ESXi host and the VMware SVGA 3D driver on the virtual machine.
- You have local administrative permission to the workstation.
- You have disabled OS power management features on the workstation.

To add a shared PCI device and specify its GPU profile:

1. Using vSphere Web Client, right-click the virtual machine in the Navigator list and select **Power** > **Power Off**.

© 2021 Teradici

2. Right-click the virtual machine and select **Edit Settings**.



3. In the New device drop-down list, select **Shared PCI Device** and then click **Add**.
4. Click **Reserve all memory**.
5. In the GPU Profile drop-down list, select the profile for your card. This profile determines how many virtual display heads, or displays, are available for your card.
6. Click **OK**.
7. Start the virtual machine.

© 2021 Teradici

Configuring Displays for NVIDIA GRID with Pass-Through

BETA

This section describes how how to add an NVIDIA GRID PCI device to your ESXi host.

The following instructions are only required for virtualized workstations using a supported NVIDIA GRID video card with pass-through. For specific supported models, see System Requirements.

Before you begin, ensure the following prerequisites are met:

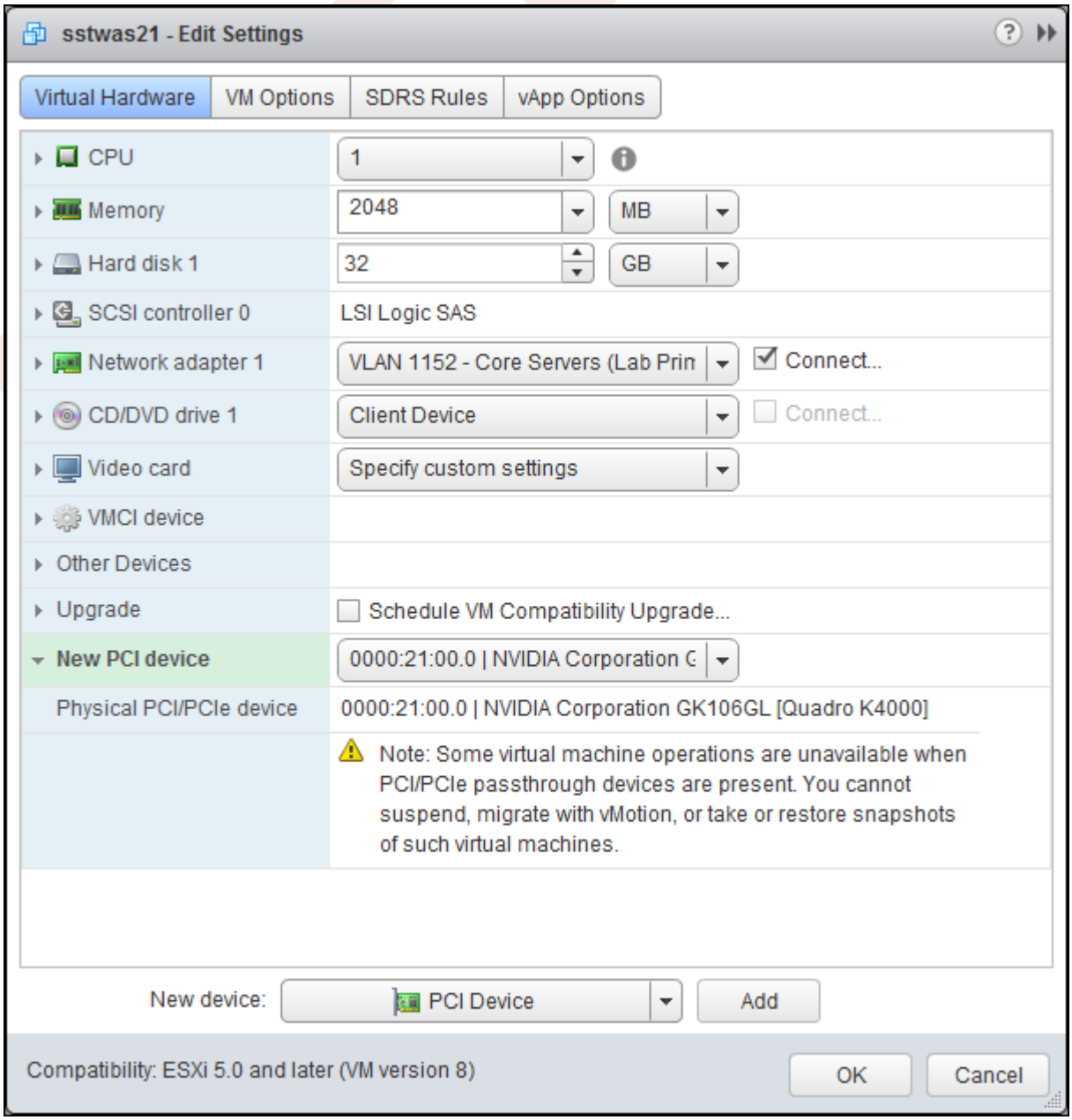
- You have installed the NVIDIA GRID graphics card.
- You have installed VMware Tools on your ESXi host and the VMware SVGA 3D driver on the virtual machine.
- You have local administrative permission to the workstation.
- You have disabled OS power management features on the workstation.

To add a PCI device:

1. Using vSphere Web Client, right-click the virtual machine in the Navigator list and select **Power** > **Power Off**.
2. Right-click the virtual machine and select **Edit Settings**.

© 2021 Teradici

3. In the *New device* drop-down list, select **PCI Device** and then click **Add**.



4. Click **OK**.

5. Start the virtual machine.

IPv6

The Graphics Agent for Linux now supports IPv6 addresses with direct (unbrokered) connections. No configuration is needed to switch between IPv4 and IPv6 modes.

If you need to support brokered IPv6 connections, contact Teradici.

BETA

Contacting Support

If you encounter any problems installing, configuring, or running the Graphics Agent for Linux, you can create a support ticket with Teradici.

Before creating a ticket, be prepared with the following:

- A detailed description of the problem
- Your agent version number (how do I find my version number?)
- A prepared support file

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 https://communities.teradici.com.

© 2021 Teradici

Finding the Graphics Agent for Linux Version Number

To find the agent's version number in Ubuntu:

```
dpkg -l "pcoip*"
```

To find the agent's version number in RHEL or CentOS:

```
rpm -qai "pcoip*"
```

The console will display a table of all registered PCoIP components and their version number, if they have one.

BETA

Creating a Technical Support File

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

To create a support file, type the following command as a super user:

```
sudo pcoip-support-bundler
```

The support file will be created and placed in your `/tmp` directory. A message will display containing the full system path to the generated file.

BETA

Performing Diagnostics

Each PCoIP component creates and updates a log file which records its activity as the system is used. Most troubleshooting within a PCoIP system begins by examining these log files and looking for error conditions or other indications that may explain why the system is not operating as expected.

Log files for the Graphics Agent for Linux and other Teradici PCoIP components are saved to specific directories.

> **Note: Bundling log files for support**
>
> When investigating issues with Teradici support, you may need to provide a support file which includes system log files. Instructions are provided here.

Locating Agent Log Files

Log files for the PCoIP agent are located in the following directories by default. If you changed your agent's location during installation, the log files will be in your custom location instead.

Component	Log file location
Agent	`/var/log/pcoip-agent/agent.log`
Arbiter	`/var/log/pcoip-agent/arbiter.log`
Session Launcher	`/var/log/pcoip-agent/session-launcher.log`
Server/User	`/var/log/pcoip-agent/server.<user>.log`

> **Note: Bundling log files for support**
>
> When investigating issues with Teradici support, you may need to provide a support file which includes system log files. Instructions are provided here.

© 2021 Teradici

Setting Log Verbosity

Each PCoIP component generates diagnostic log messages. The default log levels are recommended for use in a production deployment. When troubleshooting a particular problem, Teradici Support Services may recommend adjusting the PCoIP event log verbosity level to obtain more information from certain parts of the system.

> **Note: This is a global setting**
>
> The `pcoip.event_filter_mode` directive is a global setting, and affects the output levels of all PCoIP components.

To change the log verbosity level, set the `pcoip.event_filter_mode` directive in the **`pcoip-agent.conf`** file. See Configuring the PCoIP Agent for instructions.

Log rotation

Log files in Linux agents are managed by `logrotate`. To manage how log files are rotated, edit the following files:

- `/etc/logrotate.d/pcoip-*`
- `/usr/share/pcoip-agent/pcoip-server.logrotate`

Session Log IDs

At the start of each PCoIP session, a unique session ID is generated by the PCoIP Client and passed to all connected PCoIP components (including the Graphics Agent for Linux). Log messages generated by the agent are prefixed with this session ID, making it easy to identify. All log messages generated during a single session, by any PCoIP component, will be prefixed with the same session log ID in RFC-4122 format:

```
yyyy-mm-ddThh:mm:ss.ffffffZ xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx > …
```

For example:

BETA

```
2015-11-06T08:01:18.688879Z 4208fb66-e22a-11d1-a7d7-00a0c982c00d > …
```

Log messages that do not pertain to a specific session will show a string of zeroes in place of the session log ID number.

If a PCoIP component does not receive a session log ID from the PCoIP client, or receives an invalid value, it will generate a new session log ID and distribute it to the other components in the system.

BETA

Troubleshooting License Issues

Teradici includes a license validation utility that scans your local system and any connected physical or cloud-based license servers for active licenses, and informs you of when your license subscription expires. For more information, see FAQ - Licensing Cloud Access Software in our Knowledge Base.

To run the license validation tool, type:

```
pcoip-validate-license
```

For more detailed information on `pcoip-validate-license`, type:

```
man pcoip-validate-license
```

To list your licenses and their expiration status, type:

```
pcoip-list-licenses
```

For more detailed instructions on `pcoip-list-licenses`, type:

```
man pcoip-list-licenses
```

Tracking Usage Over Time

Teradici Local License Server users can use our open-source script, which displays the maximum Cloud Access Software license concurrent usage for a license server over time. For more information, refer to our Github page.

Teradici Cloud Licensing users can write a short script that runs `pcoip-list-licenses` periodically (for example, every 60 minutes) on any PCoIP agent machine to track license usage.

© 2021 Teradici

Frequently Asked Questions

Can I use a screensaver?

Yes. However, a blank, static screensaver will provide the most efficient CPU ane network bandwidth usage.

Why can't I use the maximum number of monitors or monitor resolution?

GPU profile licensing, GPUs, or clients may limit monitor resolution and monitor count, which will prevent you from utilizing the full agent specification. For more information, see supported displays.

Which graphics rendering APIs are supported?

The Graphics Agent captures and delivers the output from the GPU, which is responsible for providing rendering APIs like OpenGL. See the documentation for your GPU for more information about rendering APIs.

How quickly does a PCoIP agent complete a connection?

PCoIP agents can usually achieve a connection in 15 to 30 seconds. Teradici uses the statistical value Top Percentile (TP) to measure the time to establish a session:

- TP99: Ninety-nine percent of connections complete in under 30 seconds.
- TP50: Fifty percent of connections complete in under 15 seconds.

Why is my application not sending audio?

The PCoIP agent delivers audio over PCoIP connections by reassigning the system's default audio device. Only applications that use the system default audio device will send or receive audio over PCoIP; applications that are configured to use non-default devices will not work. If you don't hear audio from your application, make sure it is configured to use the system default audio device.

© 2021 Teradici

BETA

I'm using Teradici Cloud Licensing. What network blocks should I leave open?

If you are using Teradici Cloud Licensing, you will need to whitelist the following:

- teradici.flexnetoperations.com
- teradici.compliance.flexnetoperations.com

Alternatively, you can also ensure the following network blocks are whitelisted:

- Production: 64.14.29.0/24
- Disaster Recovery: 64.27.162.0/24

The following network blocks are not currently in use, but may also be used in the future:

- Production: 162.244.220.0/24
- Disaster Recovery: 162.244.222.0/24