Tutorial 10: Add-On - Using Linux and Teradici for a Workstation

Estimated Time to Complete: 1 hour, 30 minutes

Review of Studio in the Cloud

The first 7 of our Studio in the Cloud tutorials helped you create a fully cloud-based studio that leverages the power, scale and convenience of AWS. This basic setup introduced you to virtual workstations, cloud storage and cloud rendering and how they can be combined to move creative work into the cloud. In these add-on tutorials, we have been providing instructions for different ways you can leverage the flexibility and depth of AWS to customize your studio in different ways.

Overview of this Tutorial

In our first add-on tutorial, Tutorial 8: Upgrading to Teradici for Desktop Streaming, we showed you how to add desktop streaming with Teradici to your existing Windows-based artist workstations. This tutorial will show you how to create a Linux-based artist workstation and connect to it using Teradici Cloud Access Software. We'll install necessary applications, including Blender and Deadline, and create an AMI (Amazon Machine Image) and launch template that you can use to start up as many additional Linux workstations as you need.

Teradici Cloud Access Software

Because you cannot connect to a Linux instance using Remote Desktop, we will be using Teradici Cloud Access Software instead. Similar to Remote Desktop, Teradici streams pixels and audio from a virtual machine in the cloud to your local desktop and sends keyboard and mouse information back. However, it also has a few advantages over Remote Desktop:

- High-performance remote visualization with true 4:4:4 color accuracy and lossless image quality
- Delivers graphics intensive workloads to any endpoint (including PCoIP clients for Windows, Mac, Chrome OS, Android, iOS and zero clients)
- Highly performant across variable network conditions

Linux vs. Windows Workstations

If you already completed our other add-on tutorial, Tutorial 9: Using a Linux Instance for a Render Scheduler, you know that one of the motivations behind using Linux over Windows is cost savings. Your artist workstations will be running whenever your artists are working, so switching them to Linux can help cut down on those hourly costs. For the same type of instance, the hourly cost of just the instance by itself can be up to 50% cheaper for Linux vs. Windows.

In Tutorial 8, where we added Teradici to your existing Windows-workstations, we walked you through installing Teradici manually. We did this so that you would not have to re-install any software from previous tutorials. However, in this case, since we'll be switching from Windows to Linux, we'll need to re-install software anyway. As a result, for this tutorial we'll be using the Teradici Linux AMI from the AWS Marketplace. Since this AMI already has Teradici installed, it will save us setup time so that you and your artists can get working more quickly. Instead of requiring a separate license, the cost of using Teradici is built into the hourly pricing for the AMI. You pay an extra $0.50/hour on top of the normal hourly price for your instance.

Even taking the added hourly cost of the Teradici AMI into account, you should still see a cost savings of at least 10% by switching your Windows workstations to Linux workstations with Teradici. Rather than pay by the hour for Teradici software, it is also possible to purchase an annual subscription. For more information, please see the Teradici Linux AMI Subscription page.

Prerequisites

Complete the Previous Tutorials

This add-on tutorial assumes that you've already completed at least Tutorials 1-4 in our previous Studio in the Cloud tutorial series. If you are new to Studio in the Cloud, you can find the first tutorial here: Tutorial 1: Getting Started with AWS Virtual Workstations. If you have already completed the Studio in the Cloud tutorials, then you're all set to continue with the steps below.

Check Your G Instance Quota

You likely already checked your G instance quota at least once when you started the Studio in the Cloud tutorials. You may even have needed to submit a quota increase request when you started Tutorial 1 and another when you started launching more Windows G4 or G3 instances for your artists in Tutorial 7. In any case, it's a good idea to check your quota again, to make sure that you have enough quota to cover the new G4/G3 instance you will be launching in this tutorial. For instructions on checking your quota value and requesting an increase, please see the instructions in the Appendix for Tutorial 1.

Note: If you currently have any G4/G3 instances currently running, you will need to make sure that your **Applied quota value** is at least:

(16 x (total number of G4/G3s running + 1))

Each of the g4dn.4xlarge or g3.4xlarge instances that are used for workstations uses 16 vCPUs of quota, so you want to make sure your quota covers all the G4/G3s you have running, plus one more for this tutorial. For example, if you currently have 2 G4/G3s running, your Applied quota value will need to be at least (16 x (2 +1)) or 48, in order to complete this tutorial.

Quota increase requests can take up to 48 hours to process, so if you need to submit an increase request, it's best to do that as soon as possible.

Make Sure Your Render Scheduler Is Running

If you've been using your Studio in the Cloud, then most likely your Render Scheduler instance is already running and ready to go. However, if you stopped it, navigate to the EC2 dashboard and start it now. Once it's running, make sure to login as Administrator and start up the **Deadline**

Monitor, Deadline Remote Connection Server (RCS), and Deadline Pulse. We'll be running a test at the end of this tutorial to make sure that we can submit a render job from our new Linux workstation.

Create a DHCP Options Set

If you completed Tutorial 6, then you already created a DHCP options set that enables a Linux instance to connect to the Directory Service for user authentication. However, if you haven't completed Tutorial 6, then you will need to create a DHCP options set before continuing. Please follow the instructions in Tutorial 6 to Create DHCP Settings and Attach Your VPC to the DHCP Options Set.

Add Marketplace Permissions to Your User, If Needed

Since we'll be using a Teradici AMI from the AWS Marketplace as the starting point for our Linux workstation, we'll need to make sure that your user has permissions to use Marketplace AMIs. If you completed Tutorial 9: Using a Linux Instance for a Render Scheduler, you should already have the necessary permissions. If not, then please follow the instructions at the bottom of page 2 of Tutorial 9 to add them now: Add marketplace permissions to your user.

Download the Teradici Client

You may have already done this in Tutorial 8, if not follow the instructions at the bottom of page 8 of Tutorial 8: Download client to your local machine.

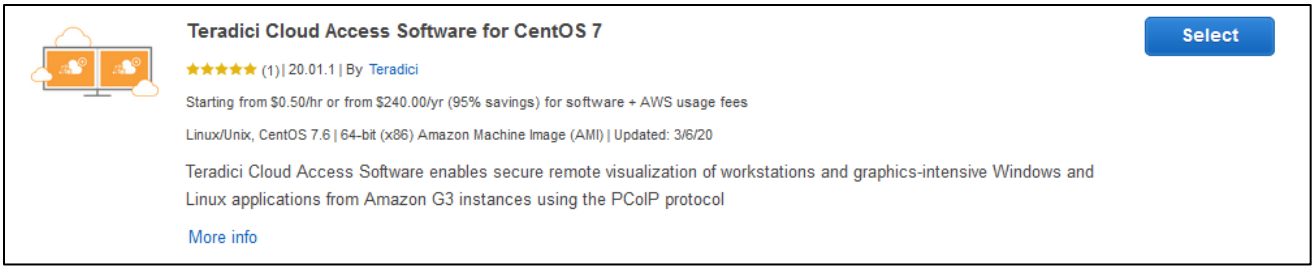
Locate Your Important Information Cheat Sheet

If you completed the other tutorials, then you probably already filled out the Important Information Cheat Sheet. We'll be referring back to some of this information in this tutorial, so you should locate your filled out cheat sheet. If you never filled out the cheat sheet, you should download it now as we'll be entering some new notes on it during this tutorial. Note: Because the add-on tutorials are optional, you will be entering any information from this tutorial into the "Notes" section at the bottom of the cheat sheet.

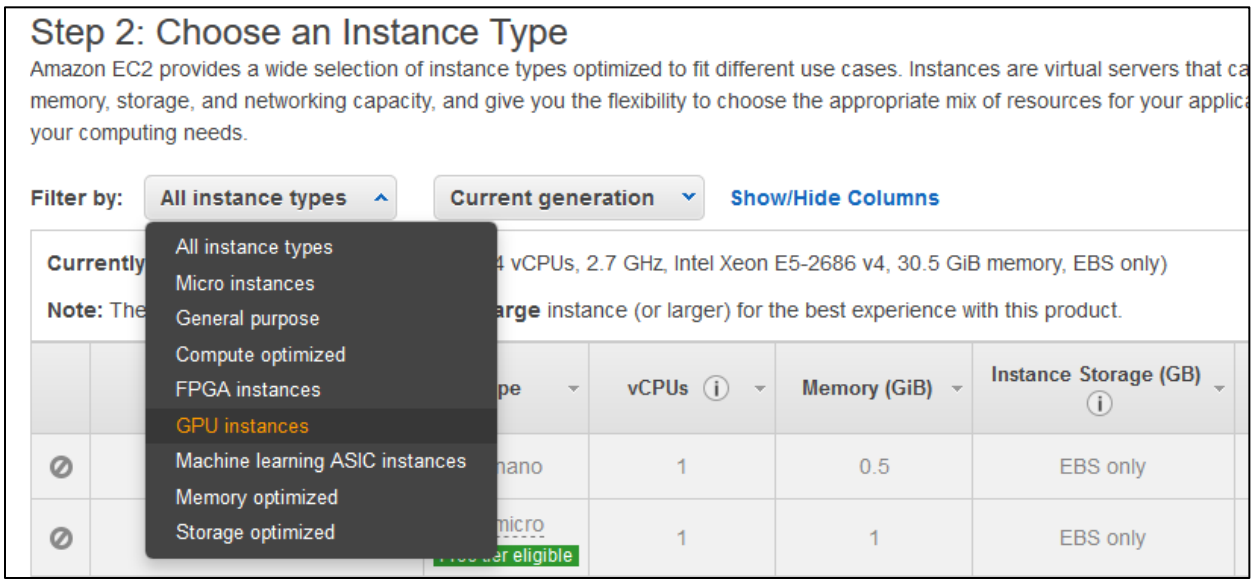
Launch an Instance with the Teradici Marketplace AMI

Once you've completed all the prerequisites above, you're ready to launch an instance with the Marketplace AMI.

- From the AWS Console go to Services→ EC2 and click on Running Instances to see a list of your running instances
- Click Launch Instance
- For Step 1: Choose an Amazon Machine Image (AMI), enter Teradici in the search box and hit <enter>
- On the left, below the search box, click on AWS Marketplace to list only AMIs that exist in the AWS Marketplace
- In the list, find the entry for Teradici Cloud Access Software for CentOS 7 and click the Select button next to it



- A pricing list for the various instance types will come up, showing you what the costs will be.
- This page also has links to more information about the Teradici software as well as usage instructions, but don't worry, we'll walk you through all the steps below!
- Click **Continue**
- For **Step 2: Choose an Instance Type**, open the **All instance types** drop down menu and select **GPU instances**



- Next, choose a **g4dn.4xlarge** instance type and click **Next: Configure Instance Details**. Note: If you don't see the g4dn.4xlarge available, you can use a g3.4xlarge instance instead.
- For **Step 3: Configure Instance Details** set the following:
  - Set **Network** to your VPC from the other tutorials (e.g., My-Studio-VPC)
  - Set **Subnet** to **Public Subnet A**
  - Set **Auto-assign Public IP** to **Enable**
  - Set **IAM role** to **EC2DomainJoin**
  - Check the checkbox next to **Enable CloudWatch detailed monitoring**
  - Click **Next: Add Storage**
- For **Step 4: Add Storage:**

    - Set storage to **150 GiB**
    - Click **Next: Add Tags**
- For **Step 5: Add Tags**:
    - Add a tag with Key = **Name** and Value = **Workstation_Linux**
    - Add a tag with Key = **Studio** and Value = **<name of your studio>** (e.g., My-Studio)
    - Click **Next: Configure Security Group**
- For **Step 6: Configure Security Group:**
    - This page is already pre-populated with a security group that was generated by the AWS Marketplace for Teradici Linux connections, so we're going to re-use that
    - Leave **Create a new security group** selected
    - Change the **Security group name** to **<name of your studio>-Linux-Teradici-SG** (e.g., My-Studio-Linux-Teradici-SG)
    - Leave the **Description** as-is
    - You'll see a warning at the bottom of the screen recommending that you limit access to known IP addresses

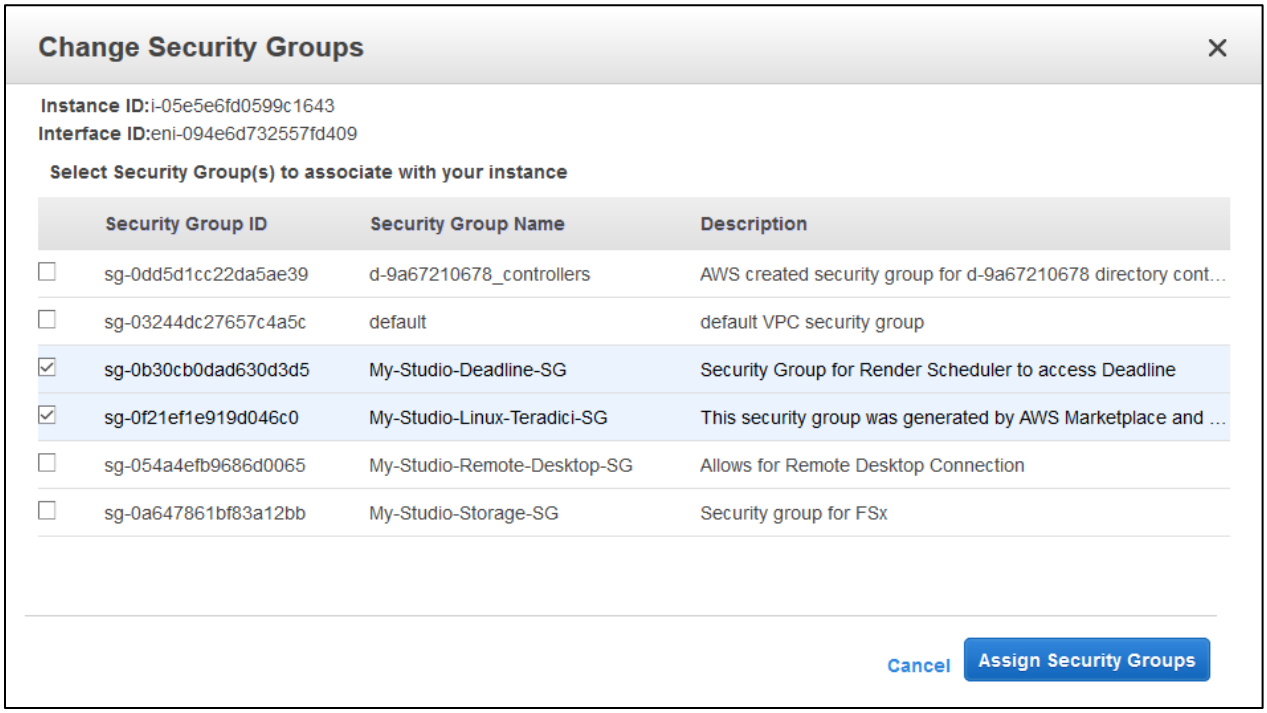
      ⚠ *Note:* As in earlier tutorials, we are initially opening up this security group to inbound traffic from any IP address. However, if you limited your source IP addresses during your Studio in the Cloud setup, you'll want to do the same here.

      And if you will be accessing your instances over the public Internet, then after initial testing we recommend that you look into an AWS Direct Connect connection and/or VPN so that your connection will be private and secure. You may use third party VPN software, but AWS also provides a robust set of VPN solutions called AWS VPN. More information about how to connect to your VPC using VPN can be found here.
    - Click **Review and Launch**
- Click **Launch** to launch your instance
- Choose the **key pair** that you created when you originally set up your Studio in the Cloud and then click **Launch Instances**. *If you don't remember the name of the key pair you created, you can find it at the bottom of the Tutorial 1 section of the Important Information Cheat Sheet.*

Add the Deadline Security Group to the Instance

We already opened the ports that we need to connect to our new Linux instance with Teradici, but before we continue, we should also add the Deadline security group that we created in Tutorial 4 as well. We'll need that in order to setup Deadline on our Linux workstation later in this tutorial.

- Go to **Services→ EC2** and click on **Running Instances** to see a list of your running instances.
- Select **Workstaion_Linux**
- Choose **Actions → Networking → Change Security Groups**
- Add **My-Studio-Deadline-SG**

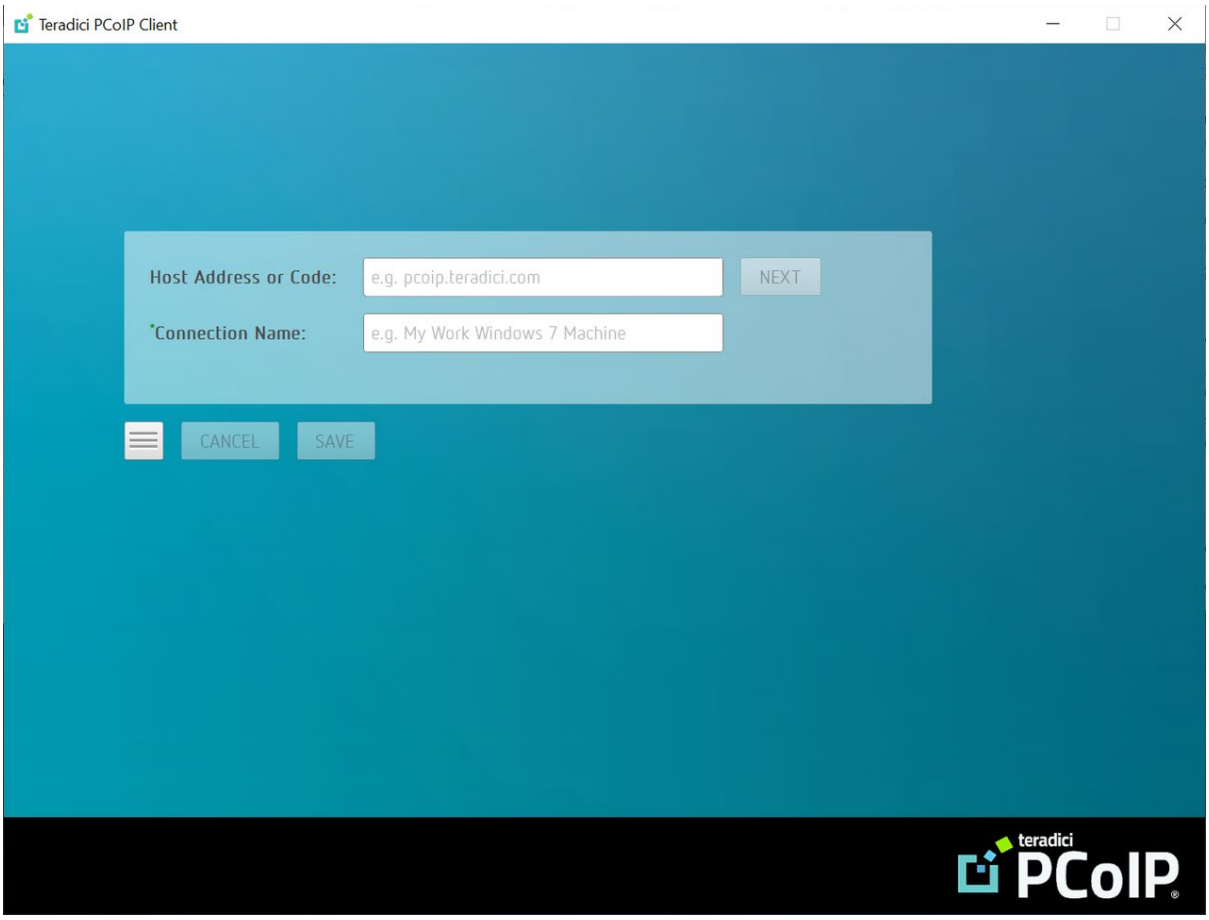


- *Now is a good time to note the Security Group ID of the Linux Teradici security group(e.g., My-Studio-Linux-Teradici-SG) that you created when launching your instance. You should enter its name and ID in the "Notes" section of the Important Information Cheat Sheet that you used for the other tutorials.*
- Click **Assign Security Groups**

Connect to Instance with Teradici

Since we used the Teradici Marketplace AMI to launch your Linux instance, it should already setup to connect to using the Teradici client software on your local machine.

- Once the status of your Workstation_Linux instance has changed to **2/2 checks passed**, it is ready for you to connect
- Launch the Teradici client software on your local machine by double-clicking the **desktop shortcut** or by going to **Start→Teradici→PCoIP Client** (Windows)

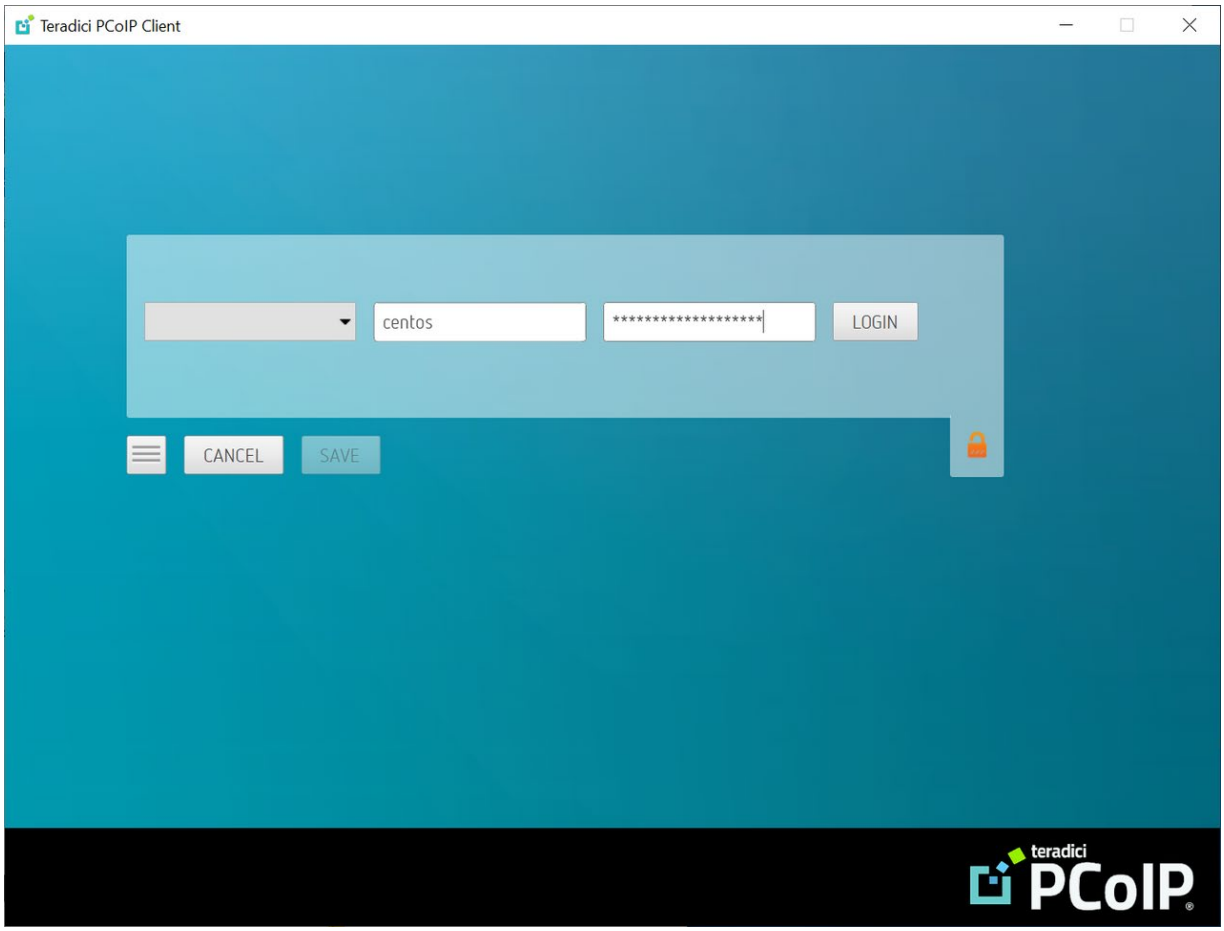


- In the **Host Address or Code** field, enter the **IPv4 Public IP** of your workstation instance.
    - Note: You can locate the IPv4 Public IP of your instance by going to the list of running instances in the AWS Console, selecting the instance and looking in the Description tab
- Click **Next**
- If you get a popup that says "Cannot verify your connection", click **Connect Insecurely** to continue.
    - Note: This connection is seen as insecure by Windows, but because the PCoIP protocol used by Teradici is inherently secure, your connection is still completely safe. From a Teradici support page:

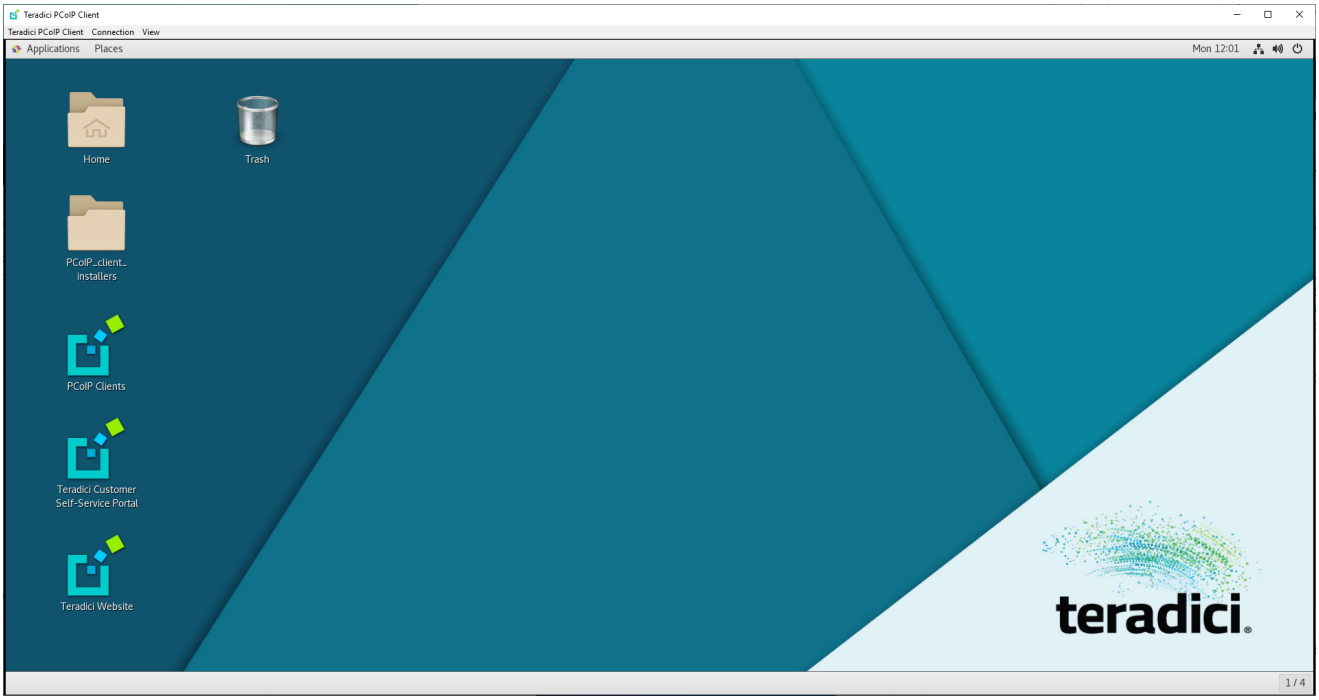
> **Note: About insecure connections**
>
> Windows sees this connection as insecure because the PCoIP Agent uses a self-signed certificate and not one signed by a trusted certificate authority. **The PCoIP session is inherently secure, so connecting this way is safe.** If you would like to avoid this step in the future, you can create your own certificate and install the appropriate files on the remote machine and your PCoIP clients.

    - For links to more information about Teradici security, please see the Appendix

- For **Username** enter **centos**
- For **Password**, use the **Instance ID** for your instance
    - You can find the Instance ID of your instance in the Description tab like we did for the IPv4 Public IP above

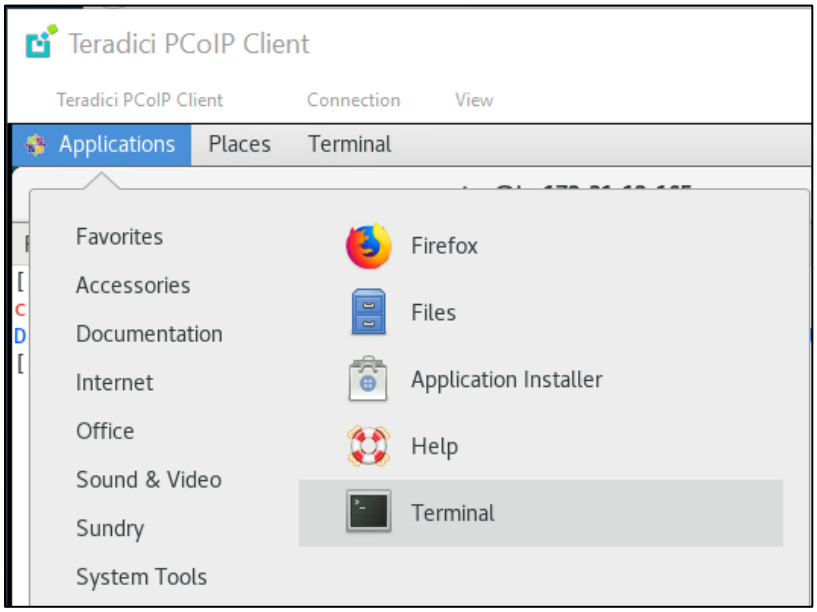


- Click **Login** to connect to your instance
- Once your Teradici session connects, you'll see the desktop for your Linux workstation. Note: The first time you connect, you may see a black screen for a minute or two, but don't worry, the session should eventually connect and you will see a desktop similar to the one below.



Connect to Active Directory

- Launch a **Terminal** by using the **Applications** menu at the top left of the desktop. You can find Terminal under Favorites.



- First we need to update the Linux instance with the latest software. In the **Terminal** run the command below:

```
sudo yum -y update
```

- Note: This initial update can take several minutes and may appear to freeze on some cleanup steps. But don't worry, just let it run for a few minutes and it should finish successfully.

- Next we'll install some specific tools to make it possible to connect to Active Directory:

```
sudo yum -y install sssd realmd krb5-workstation samba-common-tools
```

- After that, we'll run a command to actually join your Active Directory:

```
sudo realm join -U Admin@mystudio.com mystudio.com --verbose
```

- You should receive a message that says: **Successfully discovered: mystudio.com** (or your DNS name, if different)
- Type in the password for your Active Directory and you will see a message that says: **Successfully enrolled machine in realm**
- In preparation for the next step, it is recommended to perform a mkdir in /mnt/ to create a directory for the FSx mount.

```
sudo mkdir /mnt/studio
```

Mount FSx File System

Now we'll mount your FSx file system so that your Linux workstation has access to shared storage.

- First run this command to install some utilities:

```
sudo yum -y install cifs-utils
```

- Next run the command below and enter the Active Directory Admin password when prompted. Make sure to replace MYSTUDIO with your **Active Directory's DNS Name** and type it in all capital letters.

```
kinit Admin@MYSTUDIO.COM
```

- Finally, run the command below to mount your FSx drive. Again, make sure you update the red text to match your **Active Directory DNS Name** and the **FSx DNS Name**. *You can find this information under Tutorial 2 and Tutorial 3 on your cheat sheet.*
- Note: Please check the direction of the slashes when you enter the FSx DNS Name since they are "\" when on Windows, but "/" when on Linux.

```
sudo mount -t cifs -o user=Admin@MYSTUDIO.COM,cruid=$(id -u),uid=$(id -
u),sec=krb5,file_mode=0777,dir_mode=0777 //fs-
0c0b4fab1db1b9a0e.mystudio.com/share /mnt/studio -o vers=3.0
```

Again, please note which text is capitalized. These commands ARE case sensitive.

- To test and see if it mounted correctly, do an **ls** of the directory

```
ls /mnt/studio
```

- If you see a few folders returning back, then you've connected correctly!

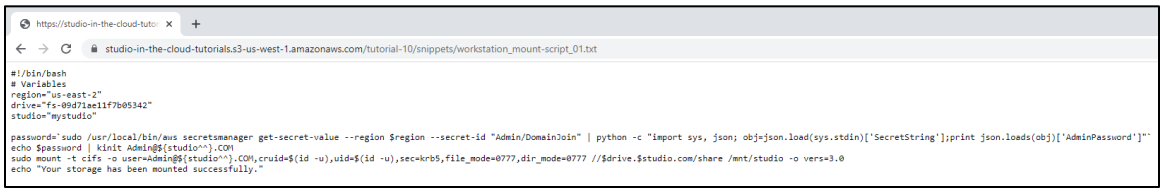
Create a Script to Automatically Mount FSx

We want to make sure that every time our Linux workstation restarts that the FSx file system is automatically re-mounted. To do that, we're going to create a script that can be run on instance startup. If you completed Tutorial 9, we did something similar there for the Linux render scheduler.

- From the **Terminal** run

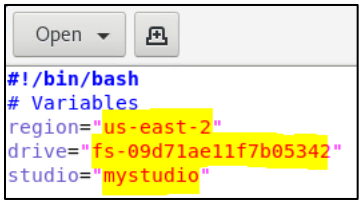
```
sudo gedit /usr/bin/workstation.sh
```

- <shift>+click on the image below to open a new browser tab with the text that needs to be entered into the gedit window:



workstation_mount-script_01.txt - <shift>+click on the image above to open the text file in a new tab

- Cut and paste the text from the browser tab into the **gedit** window
- IMPORTANT: You will need to update the items highlighted below in yellow with specific information for your studio:



- Update the value in quotes to the right of **region** with the **region for your studio**. *You can find your region on your cheat sheet.*
- Update the value in quotes to the right of **drive** with your **FSx File System ID.** *This can also be found on the cheat sheet.*
- Update the value in quotes to the right of **studio** with the **Directory NetBios name** for your studio (e.g., mystudio). *Refer to your cheat sheet for this as well.*
- Your workstation.sh file should look something like this when you're done:

- Save the file and exit the editor
- Next run the command below to add execute permissions to your script:

```
sudo chmod +x /usr/bin/workstation.sh
```

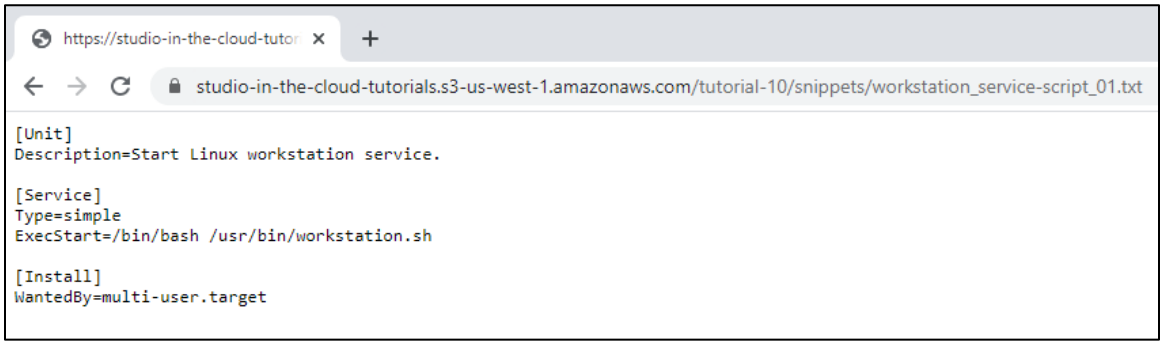
Create a Service to Run the Script

Next, you're going to create a service that will execute this script every time the instance is started. The nice thing about working this way is that we can modify this file to add more options any time we want and it will be executed again whenever the system is restarted. It's similar to the user data settings we have on the instance, but in this case it's much more controllable.

- Create the service file using **gedit**

```
sudo gedit /etc/systemd/system/workstation.service
```

- <shift>+click on the image below to open a new browser tab with the text that needs to be entered into the gedit window:



workstation_service-script_01.txt - <shift>+click on the image above to open the text file in a new tab

- Cut and paste the text from the browser tab into the **gedit** window
- Your workstation.sh file should look something like this when you're done:

- Save the file and exit the editor
- Next we'll start the service.

```
sudo systemctl start workstation
```

- And then make sure will start automatically on boot.

```
sudo systemctl enable workstation
```

Install the AWS Commandline Interface

Our workstation.sh startup script that we created above makes use of the AWS Commandline Interface (AWS CLI) to run some commands that retrieve your Active Directory's Admin user's password. But you're the Teradici Marketplace Linux AMI doesn't come with the AWS CLI installed by default, so we need to install it now.

- Run this command in the **Terminal** to download the zip file for the AWS CLI:

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
"awscliv2.zip"
```

- Next run the following to unzip the file:

```
unzip awscliv2.zip
```

- Finally, run this to install the AWS CLI:

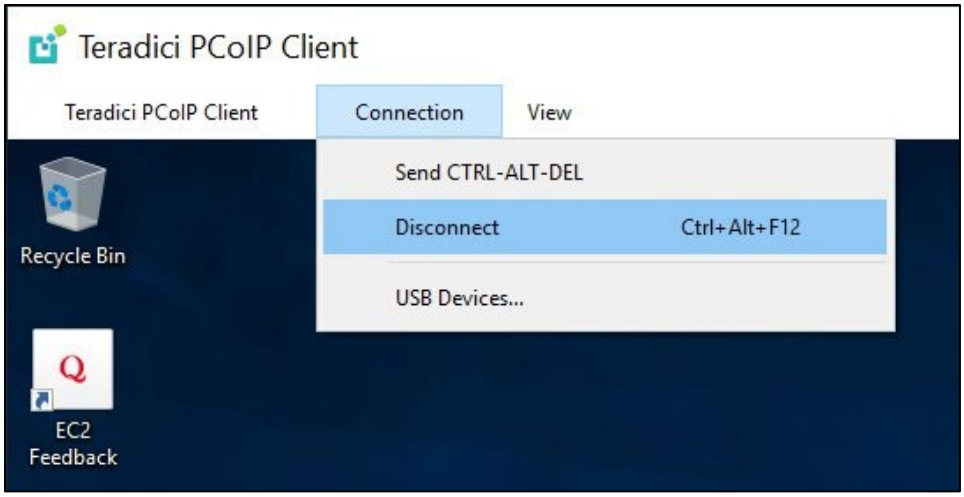
```
sudo ./aws/install
```

    - For more details, please see Installing the AWS CLI version 2 on Linux

Disconnect Your Teradici Session

To test that the service you created in the last step is working, we're going to stop and start your Linux workstation instance. But before we do that, we should disconnect our Teradici session.

- From the Teradici client's menu bar, open the **Connection** menu, then select **Disconnect**



Check That the Service Is Working

If this has all been set up correctly, you should be able to stop your instance, start it again, and then connect to it and see that the FSx file system is already mounted.

- Go **EC2** → **Instances**
- Select the **Workstation_Linux** instance
- Choose **Actions** → **Instance State** → **Stop**
- After the instance has stopped, choose **Actions** → **Instance State** → **Start**
- Once the instance has spun up and is ready to use, copy the **IPv4 Public IP**
    - Note: The public IP address changes every time an instance stops and starts
- Open the Teradici client on your local machine and enter the new public IP address
- Log in using username **centos** and the **Instance ID** for your instance that you used earlier
- Once logged in, open a **Terminal** and run:

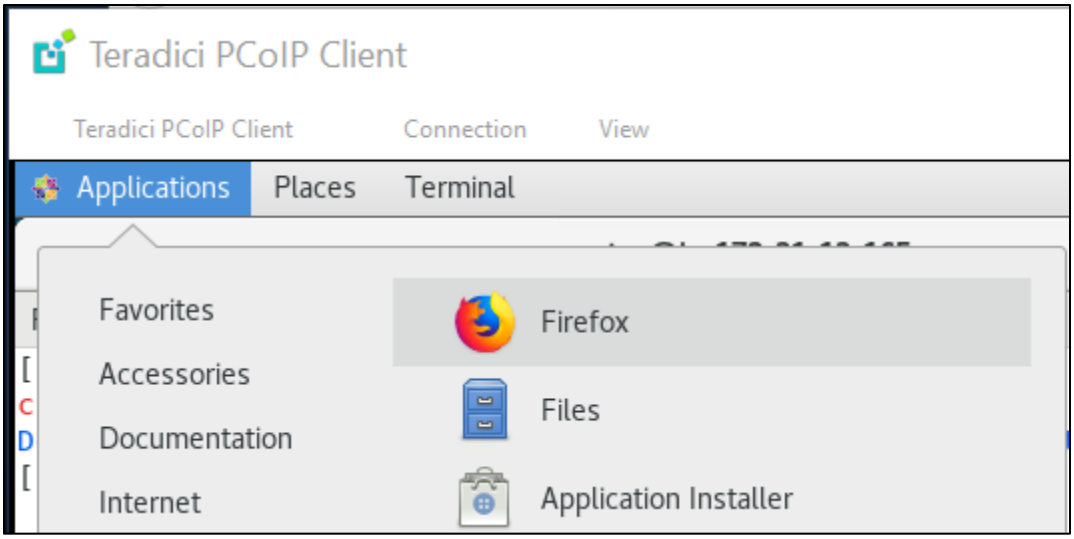
```
ls /mnt/studio
```

- If all goes well, you should see the directories on your FSx file system

Download Blender Installer

If you already downloaded the Blender installer for Linux in Tutorial 5, you can skip to the next step in this tutorial: Install Blender. If not, please follow these steps to download the installer:

- Launch Firefox by using the **Applications** menu at the top left of the desktop. You can find Firefox under Favorites.



- In Firefox, navigate to https://www.blender.org/download/
- Click **Download Blender 2.82a** (the latest version at the time of publication). The version number you see may be different.
- When the window pops up asking what you would like to do with **blender-2.82a-linux64.tar.xz**, choose **Save File** and click **OK**.
- Back in the **Terminal**, create a blender folder in /mnt/studio/installers

```
mkdir /mnt/studio/installers/blender
```

- Once the download is complete navigate to your **Downloads** folder.

```
cd ~/Downloads
```

- Move the installers to **/mnt/studio/installers/blender**

```
mv blender-*tar.xz /mnt/studio/installers/blender
```

Install Blender

- In the **Terminal**, navigate to the **installers** directory on your FSx file system.

```
cd /mnt/studio/installers/blender
```

- Next, you should unzip the contents of the installer zip file to your Linux workstation:

```
sudo tar -xvf blender-*tar.xz -C /usr/local/
```

- Finally, create an executable so you can run Blender by simply typing blender. Note: Please make sure to substitute "2.82a" with the version number that you downloaded.

```
sudo ln -s /usr/local/blender-2.82a-linux64/blender /usr/local/bin/blender
```

Install Additional Applications

There are two additional applications we would recommend installing, **DJV** and **Krita**:

- **DJV** (http://djv.sourceforge.net/) - Professional media review software for VFX, animation, and film production.
- **Krita** (https://krita.org/) - Professional open source painting program.

For each of these, we recommend creating a folder inside **/mnt/studio/installers** (e.g., /mnt/studio/installers/krita, /mnt/studio/installers/djv), downloading the installer files from the appropriate websites, and copying them to the installers location

Install the Deadline Client

Now we'll install the Deadline Client. Again, it's very similar to what we've done in the past.

- In the **Terminal**, run this command to install some extra libraries that are needed by Deadline:

```
sudo yum -y install lsb
```

- Next, navigate to the Thinkbox installers directory on your FSx file system

```
cd /mnt/studio/installers/thinkbox
```

- If you haven't done Tutorial 6, you will need to unzip the Deadline linux installers. If you have done Tutorial 6, you can skip to the next step.

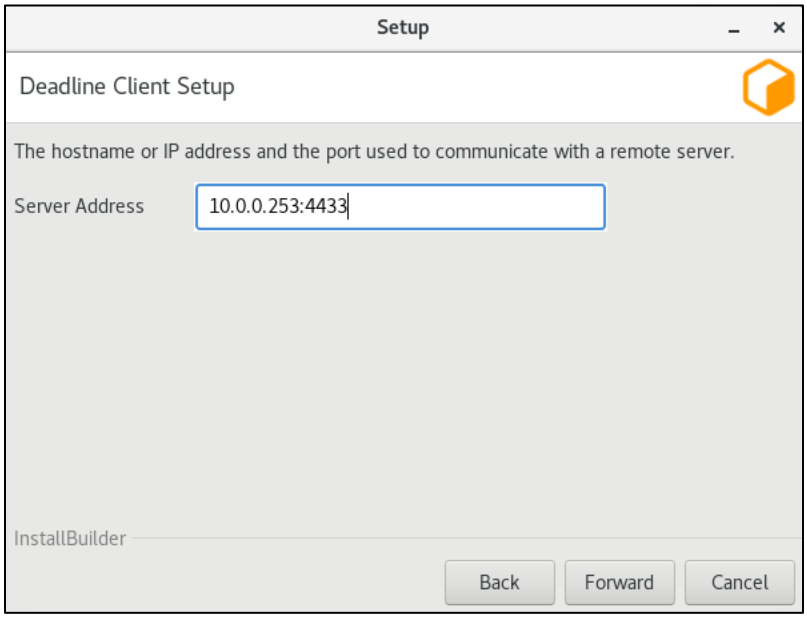
```
sudo tar -xvf Deadline-10*-linux-installers.tar
```

- To install the Deadline Client run this command.

```
sudo ./DeadlineClient-10*-linux-x64-installer.run
```

- Give these answers to the prompts:
    - Click **Forward** when the Welcome message appears
    - Accept the **License Agreement** and click **Forward**
    - Leave **Installation Directory** at the default and click **Forward**
    - Leave **Remote Connection Server unchecked** and click **Forward**
        - Note: We are leaving this unchecked because the render scheduler is the only instance that needs this installed.
    - Set **Connection Type** to **Remote Server** and click **Forward**

- Put the **Private IP address** of the Render Scheduler machine into **Server Address**, and point it at port **4433**
    - You should have already written down the Private IP for your render scheduler on the cheat sheet during Tutorial 4. It should look something like (10.0.0.219)
    - Enter the private IP with a port number of 4433 into the server address field (e.g., 10.0.0.219:4433) Note: This is important, because the default port is 8080 and that won't work.



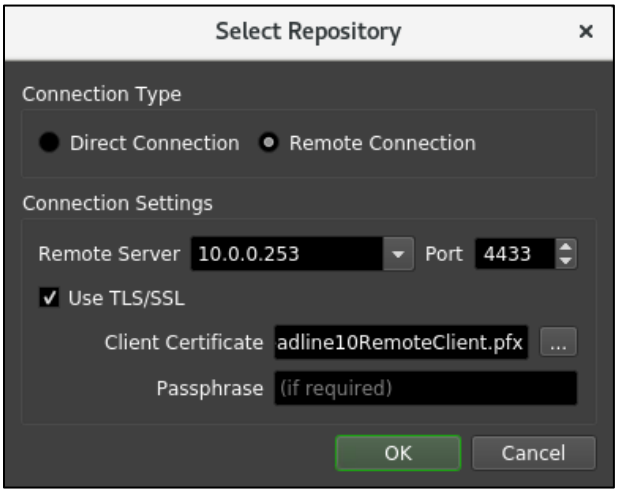
    - Click **Forward** to continue
- Set the **Client Certificate** to **/mnt/studio/app_env/thinkbox/DeadlineCertificates/Deadline10RemoteClient.pfx**
- Leave the **Certificate Password** blank and click **Forward**
- Set **License Mode** to **Usage Based** and click **Forward**
- Uncheck **Launch Worker When Launcher Starts** and click **Forward**
    - Note: We are turning this off because we are currently setting up one of our workstation machines. We don't want render workers to launch on these instances.
- Click **Forward** one more time to proceed the install
- Finally, click **Finish** to exit the installer

Connect to the Deadline Repository

- In the **Terminal**, launch the Deadline Monitor

```
/opt/Thinkbox/Deadline10/bin/deadlinemonitor &
```

- You'll see a popup window that says a new account for 'centos' has been created. Click **OK** to continue.
- If you get an error saying it can't connect to the Repository make sure you have these options selected in the prompt. Note: Your IP address will be different. Also make sure you have the correct port (4433) selected.



- If it still doesn't work, double-check that your Render Scheduler instance is running, you're logged in as Administrator, and **Deadline Monitor, Deadline RCS,** and **Deadline Pulse** are all running.
- If it works correctly, you will be able to connect to the Deadline Monitor.

Install Deadline Blender Submitter

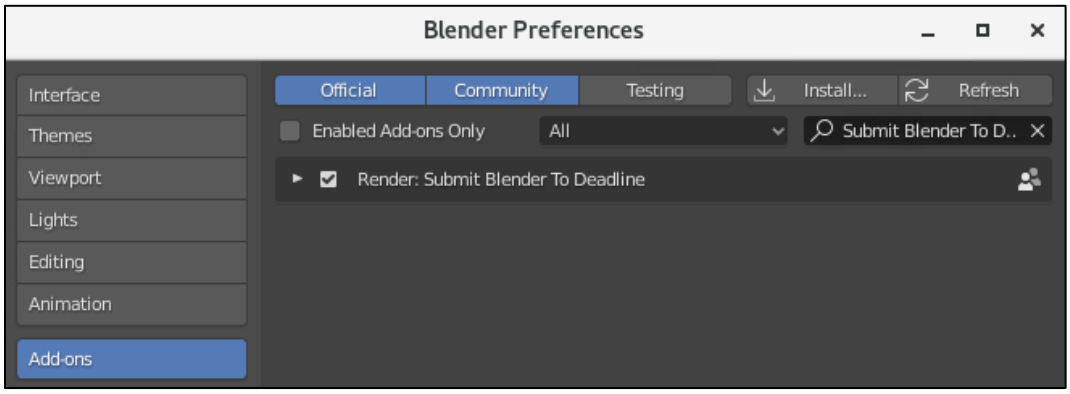
In preparation for working with your render farm, we recommend you install the Deadline Submitter add-on inside of Blender.

- Open a new **Terminal** window
    - Note: It is important to open a new Terminal window here instead of using your existing one. Otherwise some environment variables will not be set correctly.
- From the new Terminal, run the following command to launch **Blender**

```
blender &
```

- Go **Edit → Preferences...** (or **File→Preferences** - for Blender 2.79 & earlier).
- Click on **Add-Ons** in the left panel.
- Click **Install...**
- Navigate to **/mnt/studio/installers/thinkbox/submission/Blender/Client**
- Choose **DeadlineBlenderClient.py .**

- Click **Install Add-on.**
- Click the checkbox next to **Render: Submit Blender to Deadline** add-on.



- Close your preferences window.

Test Submitting to Deadline

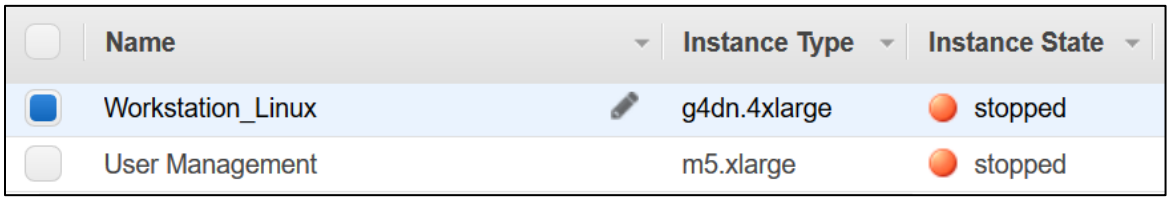
- **Save** your current scene (this is a requirement to submit to Deadline).
- Choose **Render→Submit To Deadline.**
- The **Submit Blender Job To Deadline** window will pop up.
- Change the **Output File** path to a location of your choosing.
- Change the Frame list from 1-250 to **1.**
- Hit **Submit** and then close the submit window
- If it's not already running, open the **Deadline Monitor.**
- Your job will be sitting in the queue.
- If you haven't yet completed Tutorial 6, then your job may fail or produce errors, but don't worry about that for now, all we wanted to do is test that our job was submitted to your Render Scheduler. After completing this tutorial, you can move on to Tutorial 6: Setting Up a Linux Farm Worker and Spot Fleet Request.

Create a Linux Workstation AMI

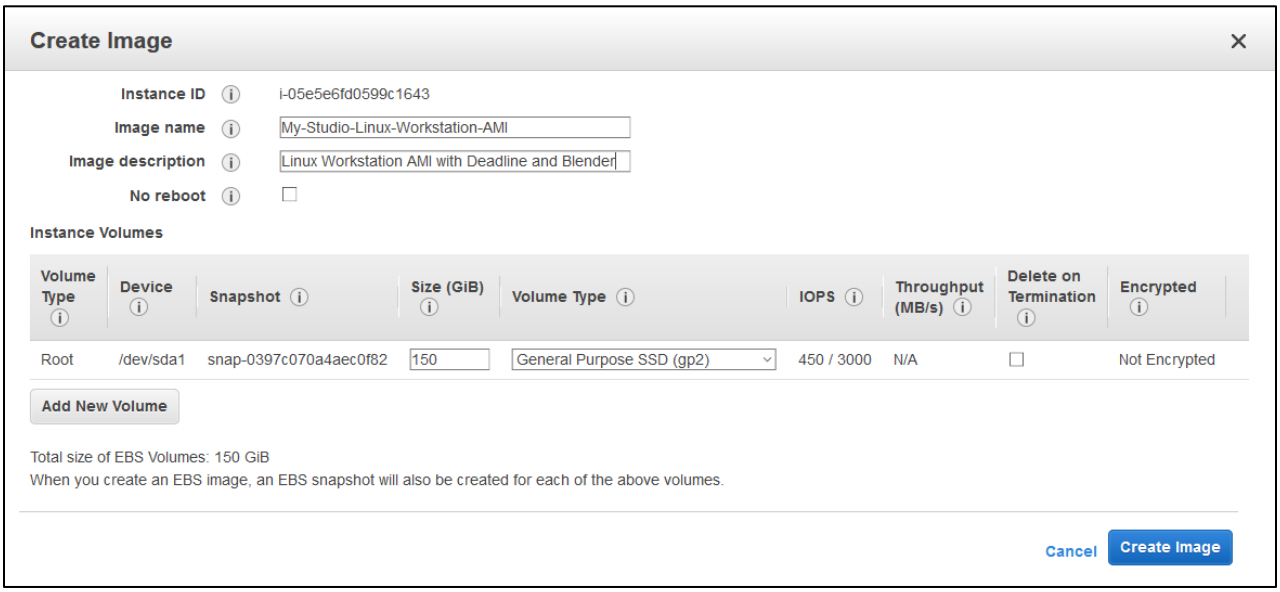
Now that you've set up your Linux workstation and installed applications, you'll want an easy way to launch another workstation just like this one without having to go through all the steps again. We already did something similar in Tutorial 3 when we created an AMI and launch template for your User Management instance.

- **Disconnect** from your Teradici session
- Next, in the **AWS Console**, go to **Services→EC2**
- Click on **Instances**

- Right click on your **Linux_Workstation** instance and choose **Instance State→Stop**
- Wait for the **Instance State** to change to **stopped**



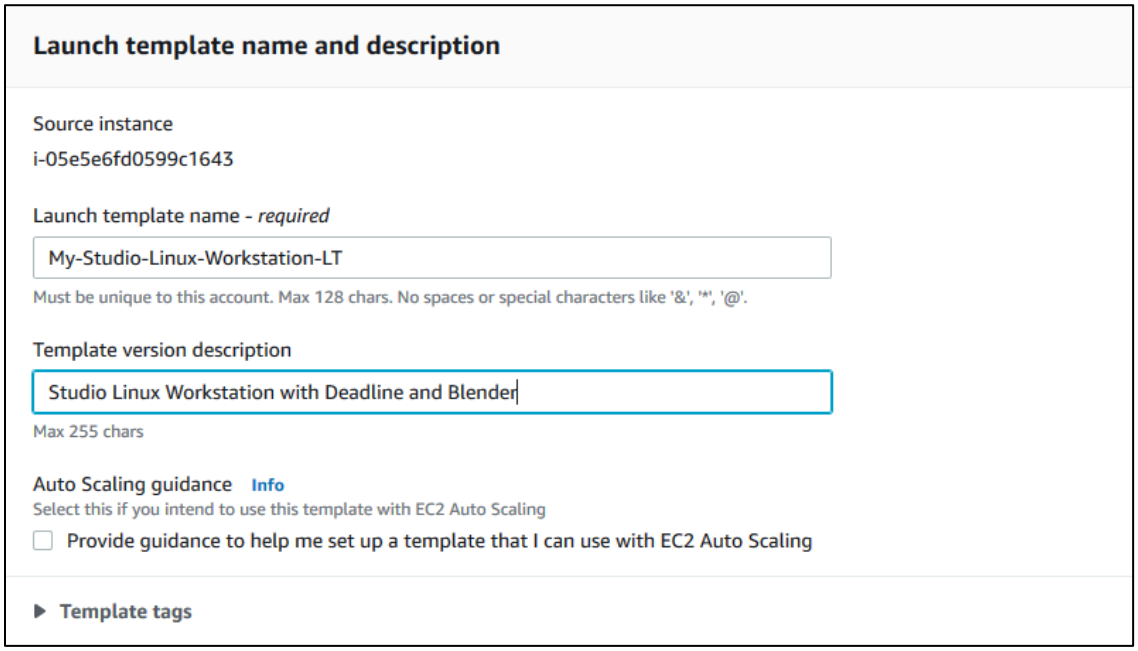
- Right-click on the instance and choose **Image → Create Image**
- Give it an appropriate name (e.g., My-Studio-Linux-Workstation-AMI).
- Give your workstation AMI a description if you want.
- Increase its storage if necessary.



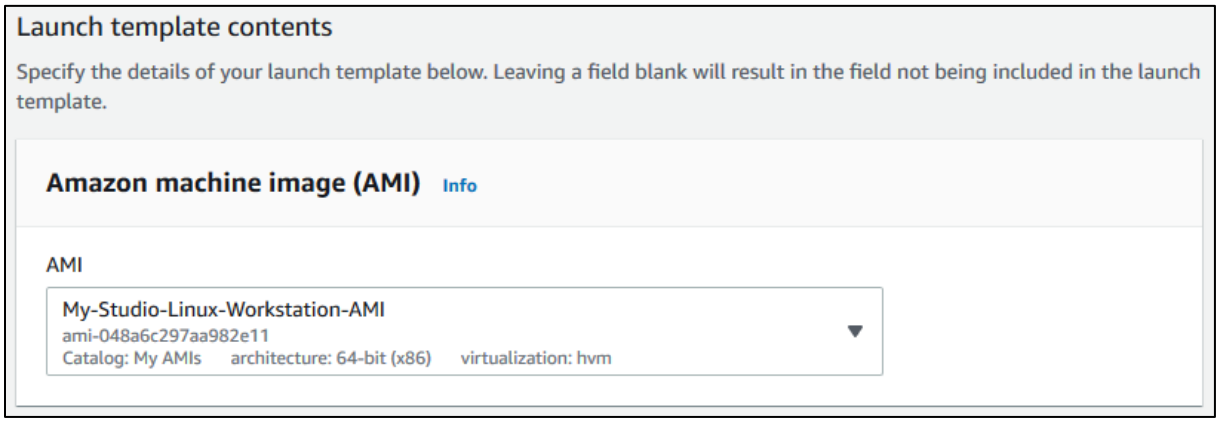
- Click **Create Image.**
- Wait 5 to 10 minutes for the new AMI to become available
    - To see if your AMI is ready, click on **AMIs** in the left panel. The AMI will need to finish creating before you can create a Launch Template with it.
    - While you're waiting, you can also add Tags to your AMI. Again, we recommend creating at least a **Studio** tag and **Name** tag.
    - *You may also want to enter the AMI ID and name in the Notes section of the Important Information Cheet Sheet.*

Create Linux Workstation Launch Template

- Once your new AMI is listed as available, go to **Services** → **EC2** and click on Running Instances.
- Right mouse click on your **Workstation_Linux** instance and choose **Create Template From Instance.**
- Name your launch template (e.g., My-Studio-Linux-Workstation-LT)
- Give it a description if you want.

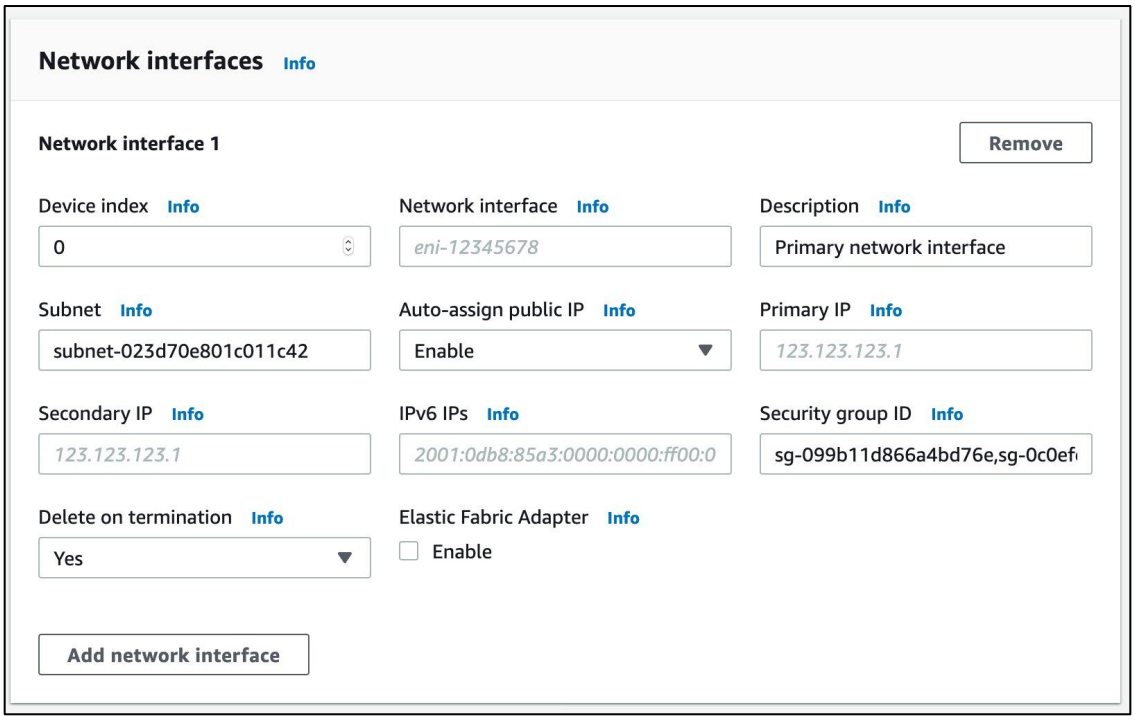


- You will need to change the **AMI ID** to point to the one that you just created.
    - In the **AMI dropdown**, scroll down to the **My AMIs** section and then select the Workstation AMI (e.g., My-Studio-Linux-Workstation-AMI) that you just made.



- Under **Network interfaces,** set **Auto-assign public IP** to **Enable,** otherwise you will not be able to connect to the instances you launch.

- Also under **Network interfaces**, check that the **Security Group IDs** for both your **Deadline Security Group** (e.g., My-Studio-Deadline-SG) and your **Linux Teradici Security Group** (e.g., My-Studio-Linux-Teradici-SG) are listed. They will be separated by a comma. *You can find the IDs for both of those security groups on your cheat sheet.*
    - If either security group is missing, add it now, being careful to put a comma between security group IDs



- Click **Create launch template.**

Launch a New Workstation

Now let's test the launch template!

- Go to **Services → EC2** and click on **Launch Templates** in the left panel.
- Select your Linux Workstation launch template (e.g., My-Studio-Linux-Workstation-LT).
- Choose **Actions → Launch instance from template.**
- Select the version (if this is your first time running through the tutorial you'll select version 1).
- Choose **Launch instance from template.**
- Go back to **Services → EC2,** click on **Running instances.**
- When your new instance is done initializing you can connect to it using the Teradici client again. This time, instead of logging with username "centos", you can login with an Active Directory username and password

    - Note: the syntax for logging in with an Active Direcory username and password is different with Teradici on Linux. Instead of logging in as <domain_name>\<user>(mystudio\jason), your username should be **<user>@domain_name** (e.g., jason@mystudio.com)
- Note: The Deadline Blender submitter needs to be installed separately for each user. If they follow the same instructions above, they should be all set. This is also the case for the Windows workstations that are created in Tutorial 5. We call this out in the sample workflow instructions in Tutorial 7, but wanted to mention it here as well.

Shut Down Notes

- Once you confirm that your new Workstation_Linux instance is running correctly, feel free to terminate the old Workstation_Linux instance that you set up at the beginning of this tutorial. It should currently be listed as stopped in the instance list.
- If you or your artists don't have an immediate need for the Linux workstation that is running, you should stop or terminate it as well. Since you created a launch template for your Linux Workstation, you can easily spin up a new one whenever you want.

Appendix

List of Studio in the Cloud Tutorials

Tutorial 1: Getting Started with AWS Virtual Workstations
Tutorial 2: Creating a VPC and Active Directory for Your Studio
Tutorial 3: Setting Up an FSx File System and User Accounts
Tutorial 4: Building a Render Scheduler with AWS Thinkbox Deadline
Tutorial 5: Installing Applications and Creating a Workstation AMI
Tutorial 6: Setting Up a Linux Farm Worker and Spot Fleet Request
Tutorial 7: Onboarding New Artists and Sample Workflow
Tutorial 8: Add-On - Upgrading to Teradici for Desktop Streaming
Tutorial 9: Add-On - Using a Linux Instance for a Render Scheduler
Tutorial 10: Add On – Using Linux and Teradici for a Workstation – *this page*

Links to AWS Documentation

- AWS Direct Connect
- AWS VPN
- VPN Connections to AWS VPC
- VPN Connections to AWS VPC
- Installing the AWS CLI version 2 on Linux

Links to Other Resources

- Teradici Cloud Access Software
- Getting Started Guide - Connecting with a PCoIP Client
- What is PCoIP Technology?
- Cloud Access Software Security Features
- PCoIP Software Client Security Modes
- Installing Certificates on PCoIP Client for Windows
- Teradici Graphics Agent Configuration Guide

Downloads

- Important Information Cheat Sheet