ORACLE
Cloud Infrastructure

Running Graphical Applications Securely on Oracle Cloud Infrastructure

ORACLE WHITE PAPER | AUGUST 2018

ORACLE

Disclaimer

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Revision History

The following revisions have been made to this white paper since its initial publication:

Date	Revision
August 20, 2018	Initial publication

You can find the most recent versions of the Oracle Cloud Infrastructure white papers at https://cloud.oracle.com/iaas/technical-resources.

Table of Contents

Introduction

Some Linux applications and installers require a graphical interface instead of a simple command line. Some reasons for this requirement are the complexity of the installation or application, requirements for data entry, or to provide a better user experience. By default, Oracle Cloud Infrastructure Linux instances are used by entering the command line via an SSH session. To get a graphical application to run, the typical process is to install VNC, open the TCP port in the security list, and start running. When run in this way, however, VNC can present a security problem, particularly when run over a direct connection to the instance via the internet.

So, what's the solution? We need to establish a secure connection to the instance, and then run the application over this secure connection. We also want to preserve the security stance we have established within Oracle Cloud Infrastructure: using asymmetric encryption of communication and authentication, without specific passwords on user accounts. Any solution we create must maintain security while providing an easy way to run the application.

However, the problem is not simply running graphical applications. Although some applications can be run in a "standalone" manner (they don't require a specific window manager), others require a fully realized desktop environment to run. So we are really trying to solve two different problems:

- **Running individual applications that don't require a full desktop experience:** This method should be the default because it provides the least amount of exposure and limits the amount of software to install. For this method, the graphical application must have a specific executable and not require desktop resources.
- **Running a full desktop environment:** This method requires the installation of a large amount of software and requires the specific configuration to be fully secure. Consider using this method only when the application requires a full desktop environment, such as GNOME or KDE.

The good news is these two scenarios are not mutually exclusive, but there are different steps to set up both the client and the instance to make graphical applications run securely.

The Solution

This section provides steps for achieving both scenarios: running individual applications that don't require a full desktop experience, and running a full desktop environment.

Scenario 1: Running Individual Applications

Scenario 1 is the ability to run individual applications without the need for a desktop environment. To accomplish this, you use the first graphical environment: X11.

Although X11 sometimes gets a bad rap, it can be run securely when combined with an SSH tunnel—which is what you are setting up. The advantages of using X11 over an SSH tunnel to run individual applications is that little setup is required and you don't need to have a desktop environment running in the background of your compute instance. You can just start and stop the application as needed.

This section provides all the steps that you need to implement scenario 1.

Set Up the Oracle Cloud Infrastructure Instance

1. Log in to the instance.
2. Configure SSHD to *not* use localhost for X11:
   A. Open `/etc/ssh/sshd_config` in your favorite editor.
   B. Search for the line that has `X11UseLocalhost yes` (it's commented out).
   C. Remove the comment from the beginning of the line.
   D. Change the `yes` to `no`.
   E. Save the file.
   F. Restart SSHD: `sudo systemctl restart sshd`
3. Install xauth: `sudo yum -y install xauth`
4. Install xterm (used to verify X configuration): `sudo yum -y install xterm`
5. Log out of the instance.

The instance is now ready. However, to use graphical applications, you must set up an X server. How you do this on your client depends on whether you have a Mac or a PC.

Set Up the Client: PC

To set up the client on a PC, we can use one of two pieces of software. The one selected for this paper was the one I found that is most common in a *very* informal survey of PC users.

Having said that, following are the two providers of X servers that I have used:

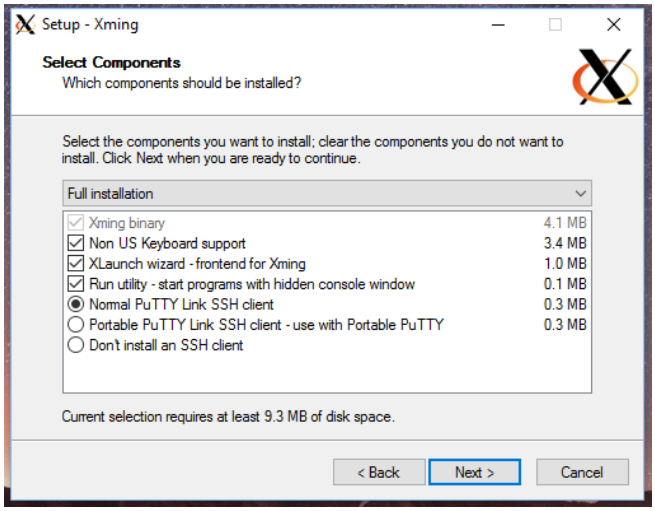
- Xming (https://sourceforge.net/projects/xming/)
- Cygwin/X (https://x.cygwin.com/)

This document uses Xming. Many of the steps are similar for other X servers, so translation should be fairly straightforward.

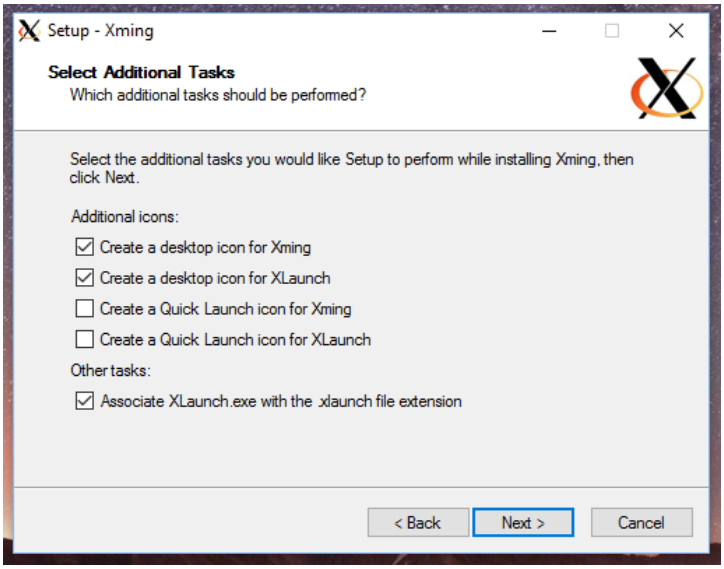
Note: You still need to use SSH to connect to the instance and establish the specific tunnel for the X11 traffic. PuTTY (www.putty.org) seems to be the most common SSH client, but any SSH client that supports X11 tunneling *should* work. If you are using a client other than PuTTY, consult the documentation regarding the setup of X11 forwarding.

1. Download Xming and run the resulting installer. Keep all defaults, except as indicated in the following steps:

   A. When the installer asks about the SSH client, select **Normal PuTTY Link SSH client**.



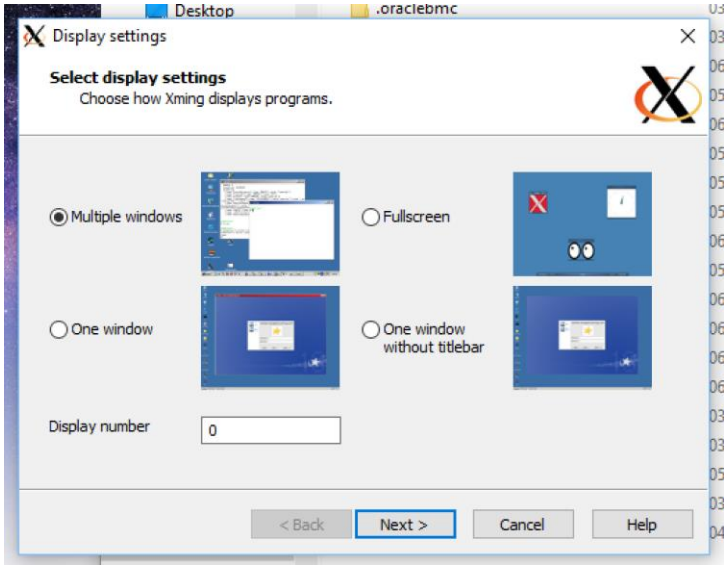
B. *(Optional)* To create desktop icons for both XLaunch and Xming, select these options. Ensure that the **Associate XLaunch.exe with the .launch file extension** check box is selected.



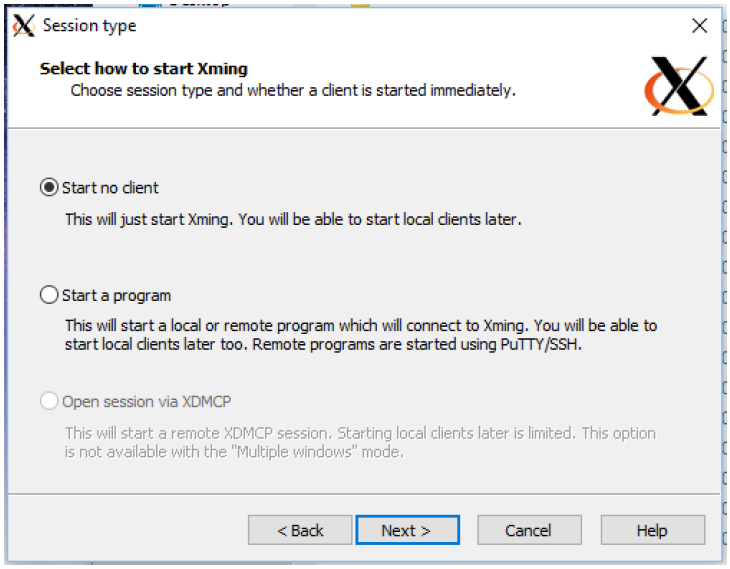
2. After Xming is installed, run the XLaunch application.

A series of settings pages appears.

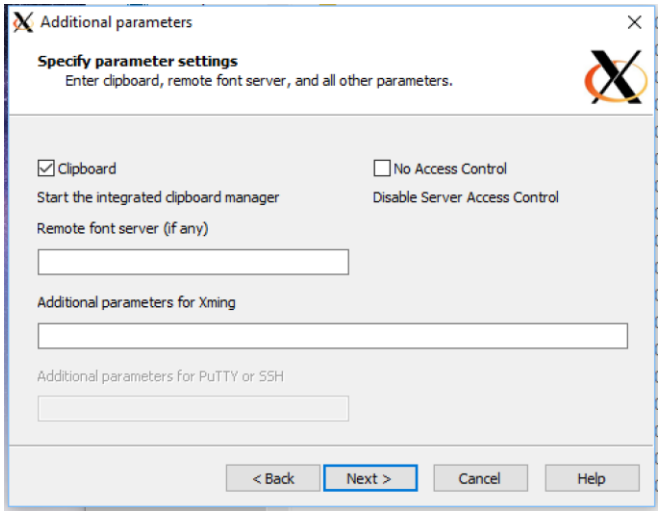
3. For display settings, select the **Multiple Windows** option and then click **Next**.



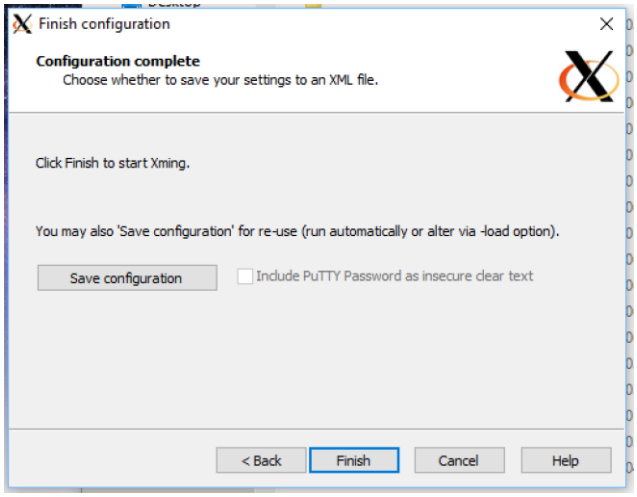
4. For the session type, select the **Start no client** option.



5. For more parameters, select the **Clipboard** check box.



6. On the **Finish configuration** page, click the **Save configuration** button, and save the configuration in the **C:\Program Files (x86)\Xming** directory. Then click **Finish**.

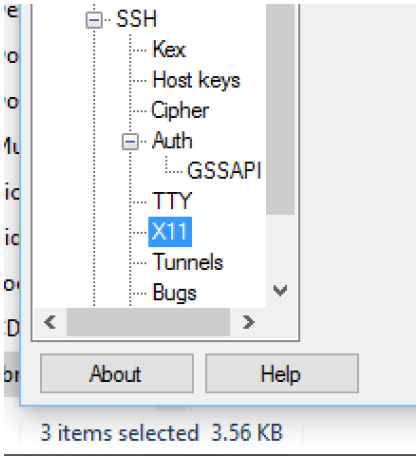


Configure the PuTTY SSH Session

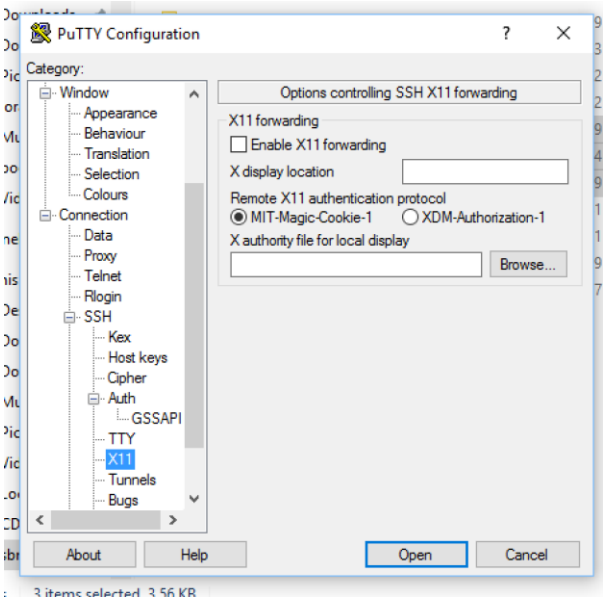
If you have set up an SSH session in PuTTY, edit the session to add X11 forwarding to the session configuration. If you don't save this configuration, you have to do it every time you run a graphical application.

1. Open PuTTY.

2. In the far-left column, expand **SSH** and select **X11**.



3. In the options pane, select the **Enable X11 forwarding** check box.



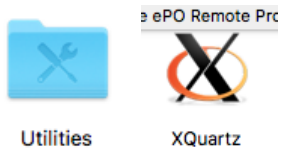
4. Stop here. Do *not* open the session. If you are saving the configuration, save the configuration. Skip to the “Run the Graphical Applications” section.

Set Up the Client: Mac

As with the PC setup, the Mac setup involves installing an X server. In this case, you use XQuartz.

Go to http://www.xquartz.org, download the `.dmg` file, and install it. You can use all the default values.

After XQuartz is installed, open the **Utilities** folder inside Finder and double-click the **XQuartz** icon.



A terminal window opens. You can use this session for the SSH connection, but any terminal window works after you have started XQuartz.

No specific SSH setup is required for Mac. SSH is built into the operating system. Simply type `ssh` in a terminal window.

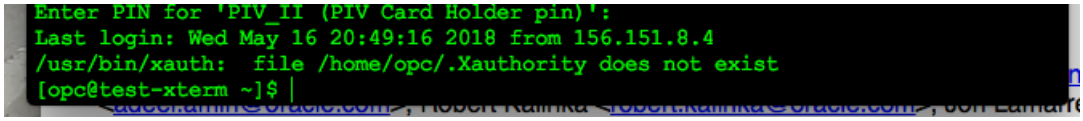
After your client is set up, you can start running individual applications.

Run the Graphical Applications

After both the instance and client are set up, you can run your application. In general, you must perform the following steps:

1. Start the X server on the laptop or desktop:
   - For Mac users, double-click the **XQuartz** icon.
   - For PC users, double-click the **Xming** icon.
2. Use SSH to connect to the instance.
   - For PC users, ensure that your PuTTY session is configured as indicated in the previous setup section.
   - For Mac users, use the following command:
     `ssh -XC opc@<public_IP_address>`

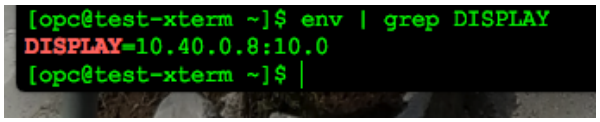
Note: The first time that you connect to the instance, you get a warning that looks like the following one:



This warning is normal and does not affect functionality. You can safely ignore it.

3. Ensure that the DISPLAY variable has been set by running `env | grep DISPLAY`.

   You should see a result similar to the following one (shown on a Mac using iTerm2 for the session):



4. Run `xterm &`.

   A new shell session opens on your local desktop and verifies that your X server is running correctly.

   The xterm session is put into the background to allow other applications to start in the same session.

5. Run your application. We recommend running graphical applications in the background by using `&` after the end of the full command line. This allows multiple graphical applications to run simultaneously.

Congratulations! Start as many applications as you like, but remember to set them in the background. To close an application, end it like you would on your desktop.

Tip: If the application doesn't end, you can use the xkill tool provided in the X11 stack. This tool ends the process associated with any window you click on. Run xkill in your SSH session and then click on the window you want to end.

Scenario 2: Running a Full Desktop Environment

If running individual applications is not possible, you can set up the full desktop experience. Although this might seem like a huge undertaking, it's fairly straightforward. Understand, however,

that you are adding processes that will run in the background, taking up resources. So if you have a smaller instance type, you *might* notice some changes in how the instance performs.

Set Up the Oracle Cloud Infrastructure Instance

1. Log in to the instance
2. Install libEGL: `sudo yum -y install mesa-libEGL`
3. Install libGL: `sudo yum -y install mesa-libGL`
4. Install the graphical desktop: `sudo yum -y groupinstall "Server with GUI"`

   Note: This installation takes a long time and requires approximately 2 GB of space on the root volume.

5. Install TigerVNC: `sudo yum -y install tigervnc-server`
6. Set a password for `opc` to access VNC by running `vncpasswd`.

   Note: The password is required by VNC. It's not used for the instance user account or for authentication to the instance. You disable that possibility in the next step.

7. Copy the template vncserver systemd script, indicating which port to access:
   `cp /lib/systemd/system/vncserver@.service /lib/systemd/system/vncserver@:1.service`
   - The `:1` in the file name indicates the offset from the standard VNC port (5900). In this case, clients would connect to 5901.
   - If other users need to be created, create multiple copies of the template and give them unique port numbers.
8. Open the `vncserver@:1` file with your favorite editor.
9. In the file, change all instances of `<USER>` to `opc`.

   If you are configuring for multiple users, in each subsequent file change `<USER>` to the appropriate username.
10. Find the line starting with `ExecStart`. Part of that line contains the string `/usr/bin/vncserver %i`. Change that string to `/usr/bin/vncserver -localhost %i`. This tells VNC to allow connections only via the loopback adapter.

This is important because it disables the ability to connect from any IP address on the instance other than the one found inside. Essentially, you have to be connected to the instance in some other way before you can connect to the VNC server. Repeat this step for any other users that you are configuring.

11. Start the graphical desktop: `sudo systemctl start graphical.target`
12. Start each VNC session: `sudo systemctl start vncserver@:1`

    Multiple users would substitute their port number, for example, `vncserver@:2` or `vncserver@:3`.

Steps 11 and 12 are effective only until the instance is rebooted. After a reboot, both the desktop and the VNC server are stopped and do not restart automatically. In general, this behavior is preferable; you are probably using the desktop environment only to set up something or perform configurations. When you want to use the desktop, just run the commands listed in steps 11 and 12.

However, if you want the desktop and the VNC server to start every time you reboot the instance, run the following commands one time to tell the instance to restart when the instance is rebooted:

```
sudo systemctl enable graphical.target
sudo systemctl enable vncserver@:1
```

Set Up the Client

As noted previously, the VNC server has been set up to accept only connections from within the instance itself. So how are you going to make that work? Before we can answer that question, you must install the VNC client on the local client (your laptop or desktop).

The setup for both Mac and PC is the same: install a VNC client on the OS. This example uses TigerVNC.

Go to http://www.tigervnc.org and download the *client* software only; the server is not necessary.

- The Windows client is listed as **vncviewer64*.exe**.
- The Mac client is contained within the **.dmg** file on at https://bintray.com/tigervnc/stable/tigervnc/1.8.0.

Install the software according to the instructions.

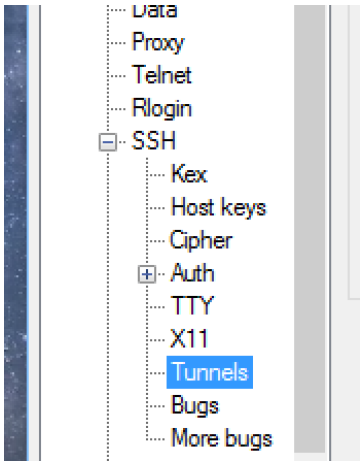
Now that you have the VNC software, you must get the laptop to talk to the instance by using SSH. As with Scenario 1, the client setup for SSH is different between Mac and PC.

Configure SSH

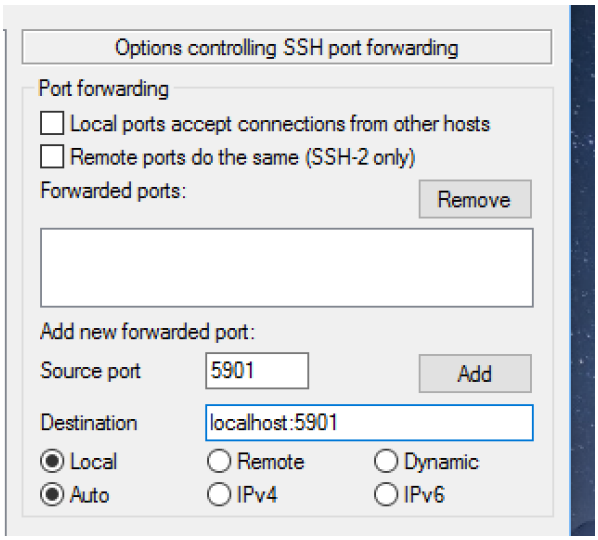
As in Scenario 1, there no specific SSH setup required for the Mac.

For PC, if you have set up an SSH session in PuTTY for the instance, you need edit the session and add some configuration.

1. Open PuTTY.
2. In the far-left column, expand **SSH** and select **Tunnels**.

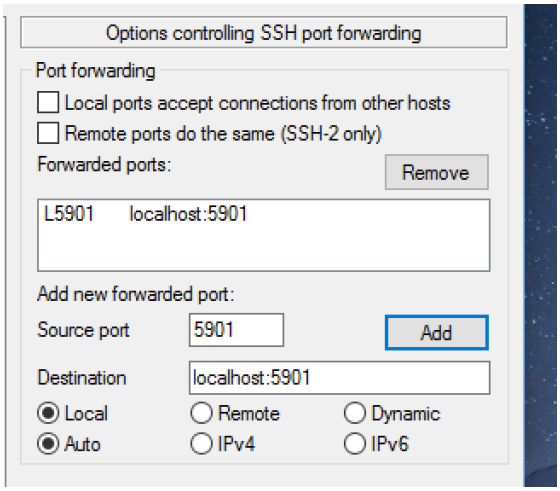


3. In the options pane, enter the source port of the VNC server that you configured earlier. For the **opc** user (assuming you selected **:1** as the VNC server for **opc**), this port is typically 5901. In the **Destination** field, enter **localhost:** followed by the server port (again, typically 5901).



4. Click **Add**.

   The **Forwarded ports** box should now have an entry similar to the one in the following screenshot:



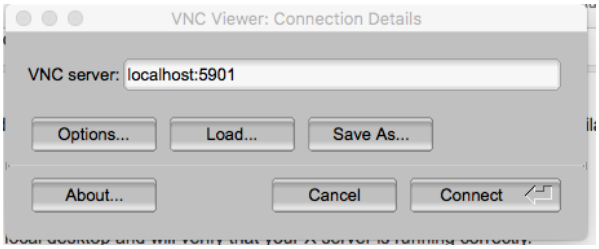
5. Save the configuration.

After the client is set up, you can start running applications in the full desktop environment.

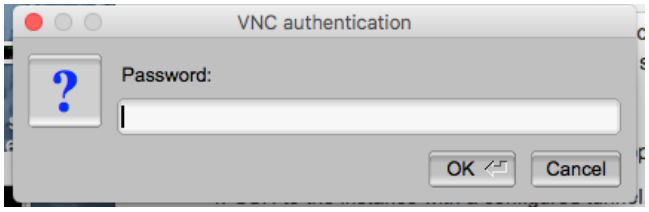
Run the Full Desktop Environment

You are now ready to connect to the desktop on the Oracle Cloud Infrastructure instance. To do so, you must perform some steps to open a secure connection:

1. Use SSH to connect to the instance with a configured tunnel.
   - For PC users, use the PuTTY session configuration created in the setup section.
   - For Mac users, use the following command: `ssh -L 5901:localhost:5901 opc@<public_IP_address>`
2. Start TigerVNC by clicking the icon on your laptop or desktop.
3. In the **VNC server** field, enter **localhost:5901**.



4. Click **Connect**.
5. In the password dialog box, enter the VNC password that you set in the setup section for the VNC server.



6. If this is the first time that you are connecting to the instance desktop, answer the setup questions that appear. After you complete the questions, your desktop is ready to use.

Conclusion

Applications have many different requirements. One of those requirements might be the ability to run in a graphical mode. However, the Oracle Cloud Infrastructure security model and instance configuration are built on executing applications on the command line, without the need for a graphical interface. To give you the ability to run graphical applications but still maintain strong security, we developed these procedures. You can choose the type of environment that you need to run (a single-application environment or a full desktop) based on the requirement for the particular application.

ORACLE

Oracle Corporation, World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065, USA

Worldwide Inquiries
Phone: +1.650.506.7000
Fax: +1.650.506.7200

Integrated Cloud Applications & Platform Services

Running Graphical Applications Securely on Oracle Cloud Infrastructure
August 2018
Author: Steve B. Nelson