To run X applications (such as xclock, emacs, totalview, etc) on the X client host (for example, a NAS system such as pfe*25*, etc.) and display them back to an X server (such as your local system), the simplest way is to use SSH X11 Forwarding.

If you are using a NAS-supported workstation or compute server, X11 Forwarding is already set up for you. The following command activates SSH X11 Forwarding automatically:

your_local_system% ssh *hostname*.nas.nasa.gov

Most modern SSH client software support this option (for example, Cygwin, TeraTerm, PuTTY, Unix, and Linux). To use SSH X11 Forwarding, the SSH server-side daemon (sshd) configuration file must contain the entry:

X11Forwarding yes

to support the forwarding capability. All NAS-supported hosts (including all workstations and compute servers) honor this setting in the default sshd_config file (for High End Computing systems and Linux systems, it is /etc/ssh/sshd_config; for Mac systems, it is /etc/sshd_config) For other non-NAS machines, ask your system administration to set this in the sshd_config file.

In addition to setting X11Forwarding yes on the SSH server side (for example, a NAS machine), it is recommended that ForwardX11 yes is also set in the ssh_config file on the SSH client host (for example, your local system). By default, NAS and HEC system configurations will enable these settings for both client and server. If ForwardX11 yes is not set in the ssh_config file by the system administrator of your local system, you can set it in your ~/.ssh/config file or use the -X option of SSH.

Other parameters related to the performance of X11 Forwarding are handled by the NAS-recommended ssh_config file. If you are on a NAS-supported system, no action is needed in setting these parameters yourself. If your local system is not supported by NAS and you would like to get configuration ideas, you can look at /etc/ssh/ssh_config on any NAS high-end computing system, such as pfe[*20-22,25,26*].

Example

To run an X11-based application, for example, xclock, on pfe*25*.nas.nasa.gov and have it displayed on your local system, do the following:

your_local_system% ssh pfe*25*.nas.nasa.gov
```
-------------------------------------------------------

    * * *  W A R N I N G     W A R N I N G  * * *

              U.S. GOVERNMENT COMPUTER
 If not authorized to access this system, disconnect NOW.
....
-------------------------------------------------------
                  ** PFE25 **
....
```

pfe*25*% xclock

To run an X11-based application on a NAS-supported desktop, follow the example below:

your_local_system% ssh sfe1.nas.nasa.gov
sfe1% ssh *a_nas_desktop*
a_nas_desktop% xclock

If ForwardX11 yes is not set in either the ssh_config file or the ~/.ssh/config file of your local system, use:

your_local_system% ssh -X *hostname*.nas.nasa.gov

WARNING: The SSH daemon sets the DISPLAY environment variable by itself. DO NOT RESET it to point to display zero (for example, setenv DISPLAY *your_local_system*:0), otherwise SSH X11 Forwarding will not work.
WARNING: For each new login, the .Xauthority file gets updated. If you are over your quota, this file cannot be updated and you get the error:

```
/usr/X11R6/bin/xauth:  error in locking authority file
/u/username/.Xauthority
```

X11 Forwarding will not work if you receive this error.

Virtual Network Computing (VNC) software provides a way to reduce X11 overhead on high-latency networks such as the Internet. In practical terms, once a VNC session is underway, latencies are on the order of seconds rather than minutes. VNC can make remote X11 applications useful instead of being tedious and non-productive.

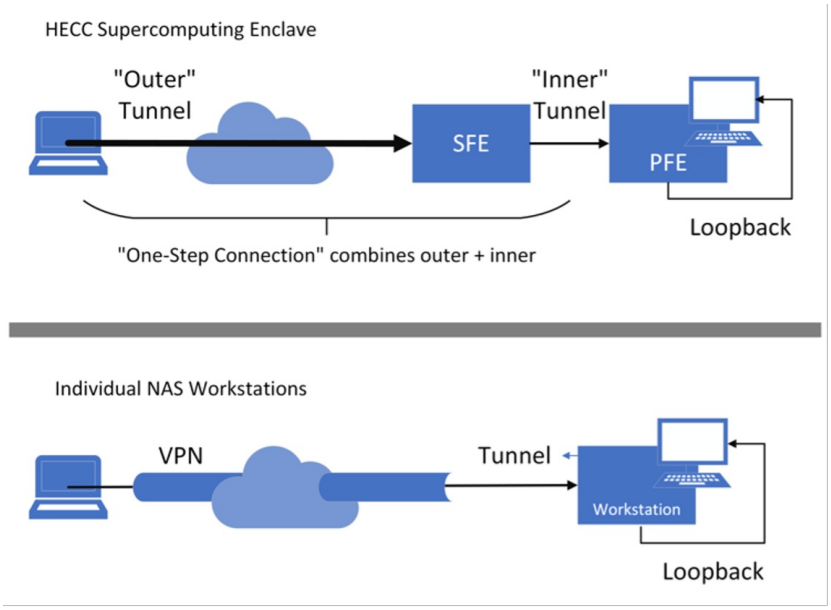
The principle of operation involves a host server process (for example, Xvnc) that communicates with X11 applications running a NAS high-end compute sysetem or a NAS workstation. The host server process transmits images and image updates using a low-overhead protocol to the remote system's viewer client.

Security and Firewalls

In the NAS environment VNC traffic is carried by an SSH and/or VPN tunnel, similar to the way SSH is used to tunnel X11 traffic. Using these tunnels provides security because SSH and VPN encrypt tunnel traffic in both directions. If you are already using SSH or VPN, then VNC traffic will travel to/from NAS systems over current connections and through current firewalls. There is no need for any additional communication updates or authorizations.

Generally, your network traffic to the VNC server, whether it is a front end for a NAS high-end compute system or a NAS workstation (for example, "linux*NNN*"), will follow a path like this:

Note: A PFE is used as an example of a front-end for NAS high-end compute systems. It can be replaced by other NAS high-end compute front-ends such as afe[01,02], cfe[01,02], cghfe[01,02], and so on.



Where is the VNC Software?

For the various front ends of NAS high-end compute systems and NAS workstations that run on Linux, all of the necessary VNC software files are installed in /usr/bin.

You do not need to run an X11 server on your local system because in the VNC environment, all of the X11 work is done on the NAS high-end compute front ends or NAS workstation. However, you do need a VNC client viewer on your local system. The client might already be installed in many Linux distributions and on recent versions of MacOS; if it is not installed on your system, you will need to download the client.

If you have a Windows desktop system, you can download free VNC clients from the following websites:

- Real VNC
- Tight VNC
- UVNC
- TigerVNC

Ask your local system administrator for help to install the VNC client software.

Steps to Establish a VNC Session

In the following steps, pfe25 is used as an example; you can substitute another pfe25 with or NAS high-end compute front end or NAS workstation.

Note: Although there are other ways to establish a VNC session, this method is convenient as it does not require you to manually find an available display number to use.

Before You Begin

VNC is much easier to use if you set up SSH Passthrough on your local system. If you are accessing a NAS high-end compute front end, you should consider setting up the One Step Connection Using Public Key and Passthrough method. You do not need to enable SSH X11 forwarding, but you will need to have set up the ability to connect via SSH from the secure front ends (SFEs) to internal NAS systems using SSH keypairs.

Known Issue: Make sure you do not have a MATLAB, Tecplot, or FieldView module or a Conda environment loaded when you invoke vncserver. Once the VNC session is established, you can load the module.

Step 1: Connect to the PFE

Once SSH Passthrough is set up properly, you can establish a SSH connection from your local system to pfe25:

your_local_system% ssh pfe25
pfe25%

If you are attempting to access a NAS workstation from a NAS-supported laptop, then the system will already be reachable from your VPN connection:

your_local_system% ssh linux123.nas.nasa.gov
linux123%

If you are attempting to access a NAS workstation from any other (non-NAS) computer, you will need to utilize the Two-Step Connection using Public Key and SecurID Password method. This will entail setting up the "outer" and "inner" SSH tunnels yourself. If you would like help with this step, please contact NAS User Services for support.

Step 2: Run the vncserver Command on pfe25

vncserver is a script that starts/stops/kills the actual VNC server, Xvnc.

The first time you invoke vncserver on a server, you will be prompted to create a password for VNC that is up to 8 characters in length. (If you create a longer password, it will be truncated to 8 characters.) This password is encrypted and saved in the $HOME/.vnc/passwd file on the server. Once this is done, you will not be prompted for a password on the server when you invoke vncserver for subsequent VNC connections.

Run vncserver as follows:

pfe25% vncserver -localhost

You will require a password to access your desktops.

Password: <--- *type in a password of your choice*

Warning: password truncated to the length of 8.
Verify: <-- *retype your password*

New '*X*' desktop is pfe25:25

Creating default startup script /u/*username*/.vnc/xstartup
Starting applications specified in /u/*username*/.vnc/xstartup
Log file is /u/*username*/.vnc/pfe25:25.log

There are a few options to the vncserver command, such as :display (for setting the display number), -geometry (for setting the desktop width and height in pixel), etc. The -localhost option shown in the above example is a local security option that you should use all the time. It must appear as the last option or it won't get processed.

Similar to an X11 session, a VNC session uses a display number. If not supplied, the vncserver searches over the valid range from 0 to 99 and assigns the next free display number for your session. In the above example, a display number of 25 is assigned.

Step 3: Create a SSH Tunnel from Your Local System to the Server

The next step is to create a SSH tunnel from your local system to the server. This is done by first escaping into an SSH sub-shell and specifying a local client's port number and a server's port number to use. The default SSH escape characters are ~C ('tilde' character, then upper case 'C'). If you do not get the SSH prompt, repeat the ~C.

Note: The ~C characters should not appear on your terminal.

pfe25% ~C
ssh> -L 59xx:localhost:59*xx*
Forwarding port.

At the SSH prompt, provide a local client port and a remote server port. VNC by default uses TCP port 5900+xx. Thus, it is common to provide the value 59xx for both the local client port (the number before localhost) and server port (the number after localhost). The value for *xx* is obtained from the final output from the vncserver startup command. In the example shown in Step 2, a vncserver was started on pfe25:25, so in this scenario *xx* would have a value of 25. The port number would therefore be 5925.

Note that the client port number and the server port number do not need to be the same. Some may suggest using a very high client

port number such as 22222 or 33333 since high port numbers are less likely to be reserved for other purposes. For example:

```
pfe25% ~C
ssh> -L 22222:localhost:5925
Forwarding port.
```

The maximum number allowed for the client port is 65535. Avoid using the local port numbers 0-1024 (root privilege required), 5900 (for Mac systems, reserved for some Apple remote desktop products), and 6000-6063 (reserved for local X window server). Use the netstat -an command to check what local port numbers have been used:

```
your_local_system% netstat -an | less
tcp46      0      0  *.5900                 *.*                    LISTEN
tcp4       0      0  *.22                   *.*                    LISTEN
```

The above example shows local ports 5900 and 22 are in use and should be avoided.

Step 4: Start the VNC Viewer Application on Your Local System

- If your local system is a Mac and you have downloaded TurboVNC Viewer, launch it. For the VNC server, enter localhost:*display number* and click **Connect**.

  In the popup window, enter your VNC password in the **Password** field.

- If your local system is a Linux system, run:

  your_local_system% vncviewer localhost:*localportnumber*

  You should get a password prompt. Enter your VNC password that you created on the server.

  The *localportnumber* is the one you use in step 3. For example, if you choose 22222 as your local port, run:

  your_local_system% vncviewer localhost:22222

If everything goes well, the Xvnc server will send a X11 window manager display to your local system that will appear as an xterm in the viewer's window.

The default window manager is GNOME, and there are a couple other window managers to choose from in the /usr/bin directory, including FVWM, MWM, and IceWM. The GNOME window manager provides a GUI view of a user's files and includes a few useful tools.

To use a non-default manager, modify your $HOME/.vnc/xstartup file on the host where your start vncserver. For example:

```
#/usr/bin/gnome-session
/usr/bin/icewm
```

You can also change the size and position of the xterm in your viewer's desktop by changing the values in the following line of the $HOME/.vnc/xstartup file on the host where you start vncserver. For example:

```
xterm -geometry 80x24+10+10 -ls -title "$VNCDESKTOP Desktop" &
```

This specifies an xterm that is 80 characters wide, 24 characters high, at a position (10 pixels, 10 pixels) from the upper left corner of the VNC viewer's desktop.

TIP: The modifications to the xstartup file only take effect for a new VNC connection. You will need to stop the existing VNC server and start a new one.

The window manager's xterm is running on pfe25 itself. From this xterm, you can do tasks that you normally do on pfe25—for example, start an X application or ssh to other NAS systems. PBS jobs can also connect to a VNC session. Specifically, in the xterm in the viewer's window, submit an interactive PBS job with the -X option (upper case 'X') and do not reset the DISPLAY variable before starting an X application:

```
pfe25% qsub -X -I -lselect=1:ncpus=40:model=cas_ait,walltime=1:00:00
qsub: job 1030046.pbspl1.nas.nasa.gov ready
PBS> xclock
```

Note: Not all NAS high-end compute systems have xclock installed.

TIP: Your VNC session and the interactive PBS job will continue to be active even if you disconnect from the front end of a NAS high-end compute system where you started vncserver. Assuming the front end where you started vncserver is not down, you can reconnect to the same VNC session: simply ssh into the front end (pfe24 in this example) and repeat steps 3 and 4 with the same port number that you used before (5925 in this example). If the interactive PBS session has not reached its wall time limit, the PBS job will be there waiting.

Step 5: Shut Down the Server When You are Done with the VNC Session

On each VNC server, there are a limited number of VNC sockets available. At the end of a session, be sure to exit the VNC application on your local system so that others can use the sockets. In the terminal window where you started up VNC, use the following command to clean up a few temporary socket files vncserver had created.

```
pfe25% vncserver -kill :xx (supply the original display number)
```

For example:

```
pfe25% vncserver -kill :25
Killing Xvnc process ID 3435054
```

WARNING: Don't manually kill vncserver. Doing so will leave lock and socket files (for example, /tmp/.X11-unix/X25, $HOME/.vnc/pfe25:25.pid, etc.) on the server.

Troubleshooting

Black Screen

If you get a black screen on your VNC viewer, try the following methods to resolve the issue:

1. Check /tmp/.X11-unix for any existing VNC sessions, and clean them up by using the vncserver -kill :xx command, as described in Step 5 above.
2. If you normally load MATLAB, Tecplot, or other GUI application modules, unload them before you start vncserver. The LIBGL_ALWAYS_INDIRECT=y setting in these modules is known to cause the black screen.
3. If unloading the MATLAB and Tecplot modules does not solve the problem, use the icewm window manager instead of gnome-session in your .vnc/xstartup file.
4. Check to see if you currently have a Python or (Ana)conda environment activated. This can cause issues when loading graphical libraries and desktop environments. The telltale sign is usually a customized prompt (see the parenthesized prefix):

   ```
   (maestro38) linuxNNN%
   ```

   Or, there may be unusual elements in your path (tcsh) or PATH (bash/ksh/zsh) variable, like the /u/wk location below:

   ```
   (maestro38) linuxNNN% echo $path
   /u/wk/lwust/.local/bin:/usr/local/bin:/usr/bin:/bin
   ```

   In these cases, check any shell initialization files (such as ~/.bashrc, ~/.profile,~/.csh.login, etc.) for lines like conda activate, setenv path, source /path/to/virtual-env/bin/activate, etc. You might want to remove these — perhaps you can turn it into an alias, instead of sourcing it every time.

Locked GNOME Session

On NAS Linux workstations, if you stop using GNOME for some time over VNC and are required to enter a PIV Smartcard and PIN (which is not possible remotely), try using the following script in order to unlock that session:

```
#!/bin/bash



tmpfile=$(mktemp)

pgrep -u $USER -f /usr/bin/gnome-shell > $tmpfile

if [ $(wc -l $tmpfile | cut -f1 -d' ') -eq 1 ]; then

  # GNOME appears to be running

  gnomepid=$(cat $tmpfile)

  # https://ma.ttias.be/show-the-environment-variables-of-a-running-process-in-linux/

  displayline=$(cat /proc/$gnomepid/environ | tr '\0' '\n' | grep '^DISPLAY')

  display=$(echo "$displayline" | sed -r 's/^([^=]+)=(.*)$/\2/')

  # https://askubuntu.com/questions/341014/unlock-login-screen-using-command-line

  DISPLAY=$display dbus-send --session \

                        --dest=org.gnome.ScreenSaver \

                        --type=method_call \

                        --print-reply \

                        --reply-timeout=20000 \

                        /org/gnome/ScreenSaver \

                        org.gnome.ScreenSaver.SetActive \

                        boolean:false

elif [ $(wc -l $tmpfile | cut -f1 -d' ') -eq 0 ]; then
```

```
  echo "No gnome-shell processes are running" >/dev/stderr
else
  echo "Multiple gnome-shell processes are running" >/dev/stderr
fi
```