How to Install and Configure VNC on Ubuntu 18.04

By finid and Brian Hogan
Updated May 7, 2020 871.9k

Introduction

Virtual Network Computing, or VNC, is a connection system that allows you to use your keyboard and mouse to interact with a graphical desktop environment on a remote server. It makes managing files, software, and settings on a remote server easier for users who are not yet comfortable with the command line.

In this guide, you'll set up a VNC server on an Ubuntu 18.04 server and connect to it securely through an SSH tunnel. You'll use TightVNC, a fast and lightweight remote control package. This choice will ensure that our VNC connection will be smooth and stable even on slower internet connections.

Prerequisites

To complete this tutorial, you'll need:

- One Ubuntu 18.04 server set up by following the Ubuntu 18.04 initial server setup guide, including a sudo non-root user and a firewall.
- A local computer with a VNC client installed that supports VNC connections over SSH tunnels.
  - On Winows, you can use TightVNC, RealVNC, or UltraVNC.
  - On macOS, you can use the built-in Screen Sharing program, or can use a cross-platform app like RealVNC.
  - On Linux, you can choose from many options, including `vinagre`, `krdc`, RealVNC, or TightVNC.

Step 1 — Installing the Desktop Environment and VNC Server

By default, an Ubuntu 18.04 server does not come with a graphical desktop environment or a VNC server installed, so we'll begin by installing those. Specifically, we will install packages for the latest Xfce desktop environment and the TightVNC package available in the official Ubuntu repository.

On your server, update your list of packages:

```
$ sudo apt update
```

Now install the Xfce desktop environment on your server:

```
$ sudo apt install xfce4 xfce4-goodies
```

Once that installation completes, install the TightVNC server:

```
$ sudo apt install tightvncserver
```

To complete the VNC server's initial configuration after installation, use the `vncserver` command to set up a secure password and create the initial configuration files:

```
$ vncserver
```

You'll be prompted to enter and verify a password to access your machine remotely:

```
Output
You will require a password to access your desktops.

Password:
Verify:
```

The password must be between six and eight characters long. Passwords more than 8 characters will be truncated automatically.

Once you verify the password, you'll have the option to create a a view-only password. Users who log in with the view-only password will not be able to control the VNC instance with their mouse or keyboard. This is a helpful option if you want to demonstrate something to other people using your VNC server, but this isn't required.

The process then creates the necessary default configuration files and connection information for the server:

```
Output
Would you like to enter a view-only password (y/n)? n
xauth:  file /home/sammy/.Xauthority does not exist

New 'X' desktop is your_hostname:1

Creating default startup script /home/sammy/.vnc/xstartup
Starting applications specified in /home/sammy/.vnc/xstartup
Log file is /home/sammy/.vnc/your_hostname:1.log
```

Now let's configure the VNC server.

Step 2 — Configuring the VNC Server

The VNC server needs to know which commands to execute when it starts up. Specifically, VNC needs to know which graphical desktop it should connect to.

These commands are located in a configuration file called `xstartup` in the `.vnc` folder under your home directory. The startup script was created when you ran the `vncserver` in the previous step, but we'll create our own to launch the Xfce desktop.

When VNC is first set up, it launches a default server instance on port `5901`. This port is called a *display port*, and is referred to by VNC as `:1`. VNC can launch multiple instances on other display ports, like `:2`, `:3`, and so on.

Because we are going to be changing how the VNC server is configured, first stop the VNC server instance that is running on port `5901` with the following command:

```
$ vncserver -kill :1
```

The output should look like this, although you'll see a different PID:

```
Output
Killing Xtightvnc process ID 17648
```

Before you modify the `xstartup` file, back up the original:

```
$ mv ~/.vnc/xstartup ~/.vnc/xstartup.bak
```

Now create a new `xstartup` file and open it in your text editor:

```
$ nano ~/.vnc/xstartup
```

Commands in this file are executed automatically whenever you start or restart the VNC server. We need VNC to start our desktop environment if it's not already started. Add these commands to the file:

~/.vnc/xstartup

```
#!/bin/bash
xrdb $HOME/.Xresources
startxfce4 &
```

The first command in the file, `xrdb $HOME/.Xresources`, tells VNC's GUI framework to read the server user's `.Xresources` file. `.Xresources` is where a user can make changes to certain settings of the graphical desktop, like terminal colors, cursor themes, and font rendering. The second command tells the server to launch Xfce, which is where you will find all of the graphical software that you need to comfortably manage your server.

To ensure that the VNC server will be able to use this new startup file properly, we'll need to make it executable.

```
$ sudo chmod +x ~/.vnc/xstartup
```

Now, restart the VNC server.

```
$ vncserver
```

You'll see output similar to this:

```
Output
New 'X' desktop is your_hostname:1

Starting applications specified in /home/sammy/.vnc/xstartup
Log file is /home/sammy/.vnc/your_hostname:1.log
```

With the configuration in place, let's connect to the server from our local machine.

Step 3 — Connecting the VNC Desktop Securely

VNC itself doesn't use secure protocols when connecting. We'll use an SSH tunnel to connect securely to our server, and then tell our VNC client to use that tunnel rather than making a direct connection.

Create an SSH connection on your local computer that securely forwards to the `localhost` connection for VNC. You can do this via the terminal on Linux or macOS with the following command:

```
$ ssh -L 5901:127.0.0.1:5901 -C -N -l sammy your_server_ip
```

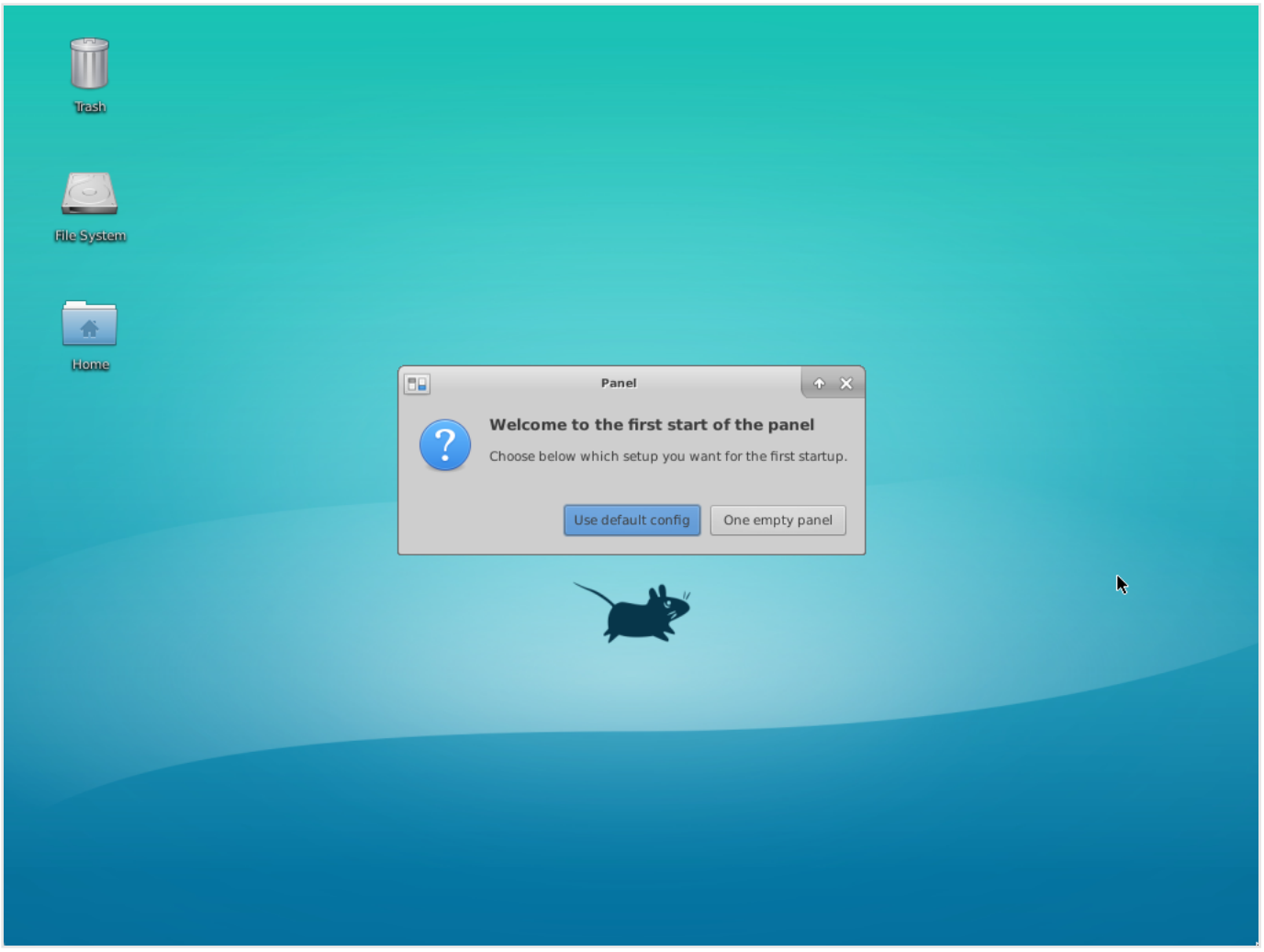
The `-L` switch specifies the port bindings. In this case we're binding port `5901` of the remote connection to port `5901` on your local machine. The `-C` switch enables compression, while the `-N` switch tells `ssh` that we don't want to execute a remote command. The `-l` switch specifies the remote login name.

Remember to replace `sammy` and `your_server_ip` with the sudo non-root username and IP address of your server.

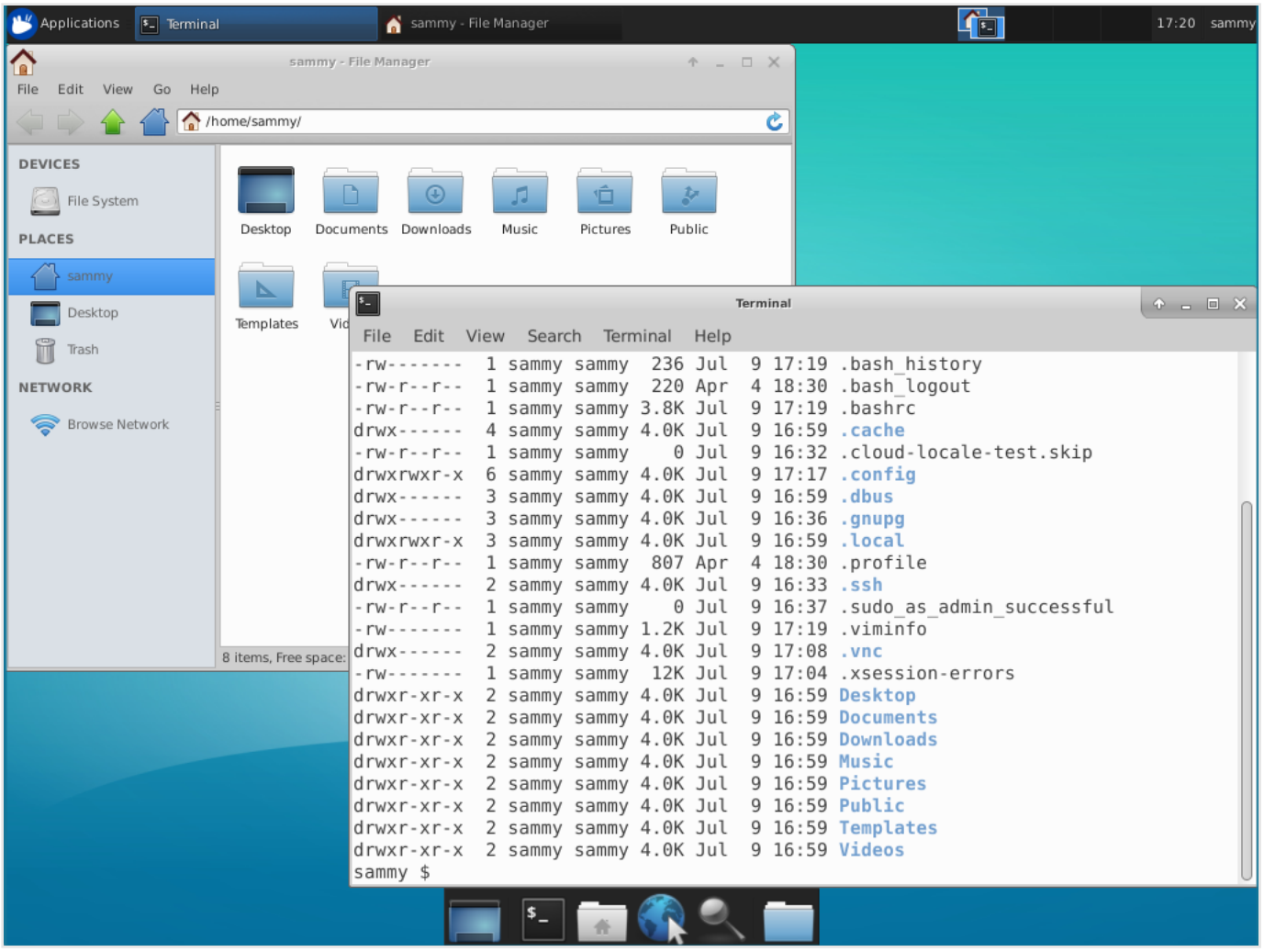
If you are using a graphical SSH client, like PuTTY, use `your_server_ip` as the connection IP, and set `localhost:5901` as a new forwarded port in the program's SSH tunnel settings.

Once the tunnel is running, use a VNC client to connect to `localhost:5901`. You'll be prompted to authenticate using the password you set in Step 1.

Once you are connected, you'll see the default Xfce desktop. It should look something like this:



You can access files in your home directory with the file manager or from the command line, as seen here:



Press `CTRL+C` in your terminal to stop the SSH tunnel and return to your prompt. This will disconnect your VNC session as well.

Next let's set up our VNC server as a service.

Step 4 — Running VNC as a System Service

Next, we'll set up the VNC server as a systemd service so we can start, stop, and restart it as needed, like any other service. This will also ensure that VNC starts up when your server reboots.

First, create a new unit file called `/etc/systemd/system/vncserver@.service` using your favorite text editor:

```
$ sudo nano /etc/systemd/system/vncserver@.service
```

The `@` symbol at the end of the name will let us pass in an argument we can use in the service configuration. We'll use this to specify the VNC display port we want to use when we manage the service.

Add the following lines to the file. Be sure to change the value of **User**, **Group**, **WorkingDirectory**, and the username in the value of **PIDFILE** to match your username:

/etc/systemd/system/vncserver@.service

```
[Unit]
Description=Start TightVNC server at startup
After=syslog.target network.target

[Service]
Type=forking
User=sammy
Group=sammy
WorkingDirectory=/home/sammy

PIDFile=/home/sammy/.vnc/%H:%i.pid
ExecStartPre=-/usr/bin/vncserver -kill :%i > /dev/null 2>&1
ExecStart=/usr/bin/vncserver -depth 24 -geometry 1280x800 :%i
ExecStop=/usr/bin/vncserver -kill :%i

[Install]
WantedBy=multi-user.target
```

The `ExecStartPre` command stops VNC if it's already running. The `ExecStart` command starts VNC and sets the color depth to 24-bit color with a resolution of 1280x800. You can modify these startup options as well to meet your needs.

Save and close the file.

Next, make the system aware of the new unit file.

```
$ sudo systemctl daemon-reload
```

Enable the unit file.

```
$ sudo systemctl enable vncserver@1.service
```

The `1` following the `@` sign signifies which display number the service should appear over, in this case the default `:1` as was discussed in Step 2..

Stop the current instance of the VNC server if it's still running.

```
$ vncserver -kill :1
```

Then start it as you would start any other systemd service.

```
$ sudo systemctl start vncserver@1
```

You can verify that it started with this command:

```
$ sudo systemctl status vncserver@1
```

If it started correctly, the output should look like this:

```
Output
● vncserver@1.service - Start TightVNC server at startup
   Loaded: loaded (/etc/systemd/system/vncserver@.service; indirect; vendor preset: enabled)
   Active: active (running) since Mon 2018-07-09 18:13:53 UTC; 2min 14s ago
  Process: 22322 ExecStart=/usr/bin/vncserver -depth 24 -geometry 1280x800 :1 (code=exited, status=0/SUCCESS)
  Process: 22316 ExecStartPre=/usr/bin/vncserver -kill :1 > /dev/null 2>&1 (code=exited, status=0/SUCCESS)
 Main PID: 22330 (Xtightvnc)

...
```

Your VNC server will now be available when you reboot the machine.

Start your SSH tunnel again:

```
$ ssh -L 5901:127.0.0.1:5901 -C -N -l sammy your_server_ip
```

Then make a new connection using your VNC client software to `localhost:5901` to connect to your machine.

Conclusion

You now have a secured VNC server up and running on your Ubuntu 18.04 server. Now you'll be able to manage your files, software, and settings with an easy-to-use and familiar graphical interface, and you'll be able to run graphical software like web browsers remotely.