Mastering Linux, part 14

In the latest instalment of this Workshop series, *Jarrod Spiga* shows you how to configure a Linux PC so it can run in X Windows from a remote machine.

JARROD SPIGA

Jarrod Spiga is an infrastructure and network engineer who specialises in getting Linux and Windows systems happily coexisting with each other. He currently holds CCNA and numerous MCP certifications, and dabbles in Web application development in his spare time.

BONUS DVD SOFTWARE

- PDFs of every instalment of the Mastering Linux series.

SKILL LEVEL

- Advanced

REQUIREMENTS

- An installation of Linux on one system on a network; another PC on another system on the same network (Fedora Core 4 and Red Hat Enterprise Linux ES 3.0 instances were used in the writing of this article).

TIME TO COMPLETE

- 3 hours

IN THIS SERIES

- **Part 13 – December '05** Filesystems and interactive scripts.
- **Part 14 – January '06** Remote X Windows.
- **Part 15 – February '06** Security and system protection.

Unix has been a multiuser decentralised computing platform since it was invented in 1969. In those days, it wasn't uncommon to see Unix running on large mainframes with hundreds of people using dumb serial terminals to concurrently log in and use the high-powered computing system. In part 7 (*APC* June 2005, page 116 — also available on this month's DVD), it was shown how you could remotely use the SSH (secure shell) protocol to bring up additional terminal sessions on your Linux system, in much the same way as the serial terminals of old.

In the GUI world, true multiuser features have only recently crept into Microsoft Windows operating systems (e.g., Terminal Services and Remote Desktop). Even so, Microsoft's implementation of this function is kind of clumsy —additional users log on to virtual consoles, and a bitmap of the virtual screen is sent to the remote client.

THE X WINDOWS PROTOCOL

On the other hand, the X Windows system was designed and built on the same network-centric ideas that govern the command-line console. Specifically, an X server running on any system communicates with various client programs (called X Clients), which can be running on the local system or on a remote system. The local X server is responsible for drawing the image on the screen and taking input from the keyboard and mouse.

In a remote-access context, this allows the application to run on the remote system (the X Client). The data from the remote system is sent to the local system (the X server), which handles the I/O.

The communications protocol used to transfer data between the X Client and X server is transparent to most users — it supports the use of different architectures and operating systems, and is capable of being encrypted between endpoints. For more details on the protocol, see **http://en.wikipedia.org/wiki/X_Window_System_protocols_and_architecture**.

FIREWALL CONSIDERATIONS

In today's security-conscious network environment, it's not uncommon to see at least basic packet-filter firewalls in place on most computer systems. 1 Fedora installs with a simple default firewall configuration that blocks almost all remote access attempts to your Linux system. This firewall configuration needs to be modified in order to enable connection to your system from remote locations.

A basic configuration tool ships as standard with early Fedora versions and allows you to quickly open up the firewall to accept connections for basic services. This tool is accessible by navigating to GNOME > System Settings > Security Level. It will allow you to open up SSH access, which is required for the first of the three remote access methods detailed here. However, it isn't flexible enough to open up the ports required by the other two methods.

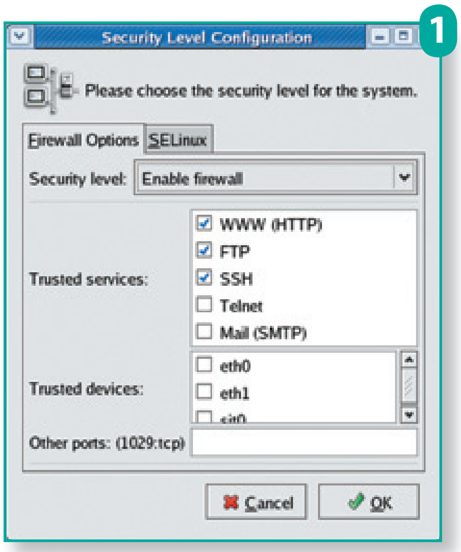
A more advanced version of this tool is the Red Hat Security Level Configuration Tool. It's almost identical to the version included with Fedora Core 1 and 2, but it allows you to specify additional ports to let through the firewall. Users of Fedora Core 3 or later will already have this advanced tool installed. Newer versions of the tool should be able to be found on the **www.rpmfind.net** Web site — simply search for "redhat-config-security".

To install the upgraded tool, download the redhat-config-security-1.2.11-1.i386.rpm file from the Web site. From within Nautilus, you can install the package by right-clicking on it and selecting the Install option. From the command line, the package can be installed by entering:

```
rpm –Uvh ~/redhat-config-security.rpm
```

Once the advanced tool is installed, launch it by selecting GNOME > System Settings > Security Level. If you'd like to enable inbound

1



Please choose the security level for the system.

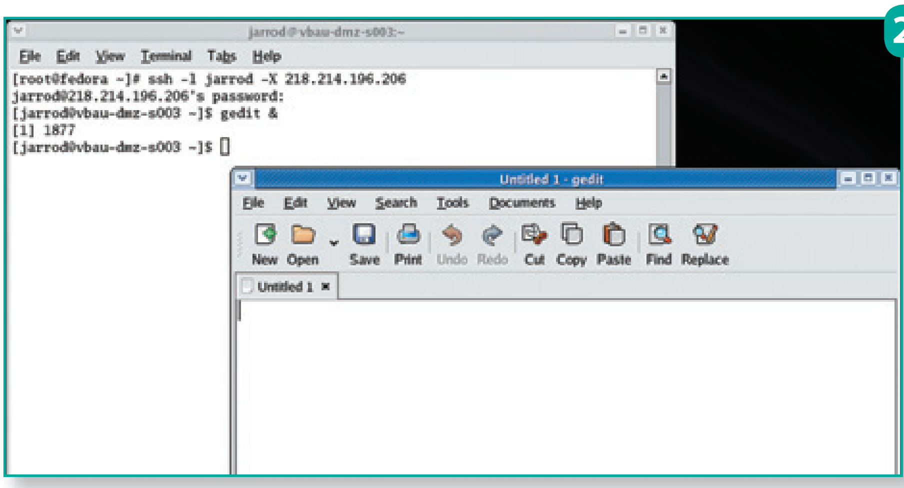
Security level: Enable firewall

Trusted services: [✓] WWW (HTTP) [✓] FTP [✓] SSH [ ] Telnet [ ] Mail (SMTP)

Trusted devices: [ ] eth0 [ ] eth1 [ ] sit0

Other ports: (1029:tcp)

Wall of fire: The Security Level Configuration tool allows you to make rudimentary changes to the default firewall configuration.

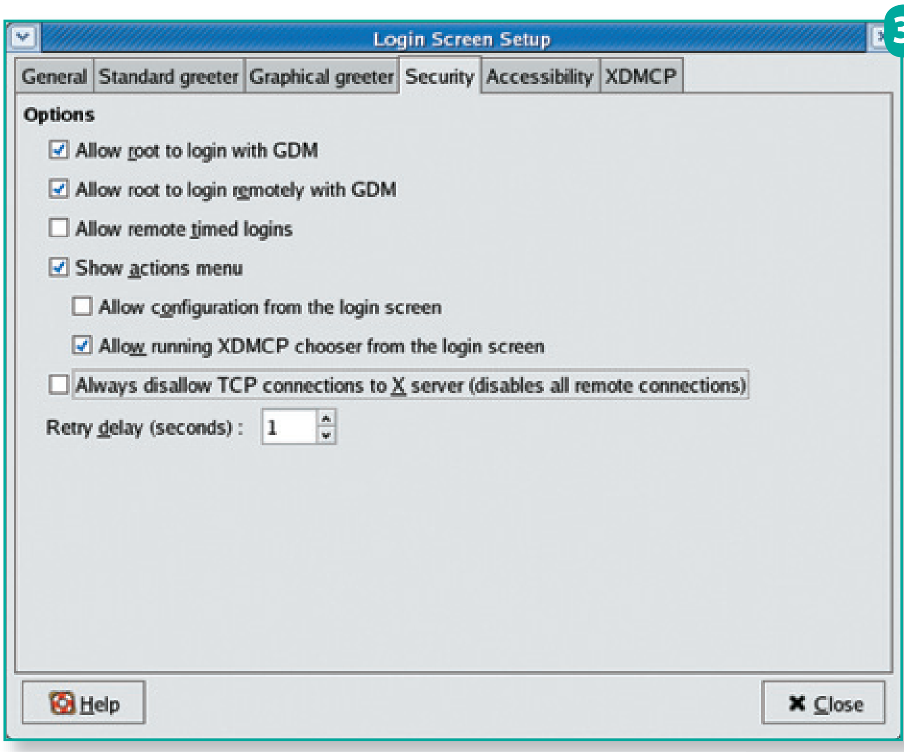


```
[root@fedora ~]# ssh -l jarrod -X 218.214.196.206
jarrod@218.214.196.206's password:
[jarrod@vbau-dmz-s003 ~]$ gedit &
[1] 1877
[jarrod@vbau-dmz-s003 ~]$
```

jarrod@vbau-dmz-s003:~
File Edit View Terminal Tabs Help
Untitled 1 - gedit
File Edit View Search Tools Documents Help
New Open Save Print Undo Redo Cut Copy Paste Find Replace
Untitled 1

2

Remote SSH Applications: Using SSH sessions to launch remote X Windows applications is the quickest and most convenient remote access method.



Login Screen Setup

General | Standard greeter | Graphical greeter | Security | Accessibility | XDMCP

Options

- [✓] Allow root to login with GDM
- [✓] Allow root to login remotely with GDM
- [ ] Allow remote timed logins
- [✓] Show actions menu
  - [ ] Allow configuration from the login screen
  - [✓] Allow running XDMCP chooser from the login screen
- [ ] Always disallow TCP connections to X server (disables all remote connections)

Retry delay (seconds) : 1

3

Xhost preparation: TCP connections to the X server need to be enabled on the local system for xhost-based remote access to work.

SSH on your system (as per the first remote access method listed below), simply tick the SSH box in the "Allow incoming" section of the tool.

Lastly, if you're planning on connecting to a Linux system running on a home LAN from elsewhere on the Internet, don't forget to open up the necessary ports on your broadband router and to allow port forwarding to your Linux system.

LAUNCHING REMOTELY USING SSH

Part 7 explained how you could connect to a remote Linux system and bring up a command prompt via the Secure Shell (SSH) protocol.

After logging on to a remote system via SSH, you can start X Windows applications remotely and view their output locally. ❷ To do so, you simply need to know the name and location of the X Windows application that you want to launch on the remote system.

When launching X Windows applications in this way, it's a good idea to use the ampersand character to launch the application in the background. That way, you can continue using your SSH session to perform other functions or load other applications remotely. It's also important to remember the X switch. Without it, X forwarding will not be enabled over your SSH session and you'll get a Gtk warning that says "cannot open display".

In the event that you only want to launch a single remote application, you can pass the full path as another argument to the ssh command as follows:

```
ssh –l <username> -X <hostname>
<command>
```

where <username> is your username on the remote system, <hostname> is the hostname, fully qualified domain name (FQDN) or IP address of the system that you're connecting to, and <command> is the full path to the command that you wish to run on the remote system.

MANUALLY LAUNCHING REMOTE X WINDOWS APPS

Launching remote applications via SSH is the simplest way to do things, but if you can't SSH to the remote system, there's an alternative. You can use the xhost tool on the local system in conjunction with adjusting the DISPLAY environment variable on the remote system. This method requires a fair bit of initial configuration and can be prone to security issues if not configured correctly.

The xhost tool is responsible for adjusting the security settings on the local display system so as to allow remote applications to be displayed. The DISPLAY environment variable is what is used to tell the remote application which system's display should be used for the program's output.

ADJUSTING SECURITY SETTINGS

By default, Fedora systems are not configured to allow remote applications to be displayed on the local system. While this is good security practice, these settings will prevent you from manually setting up X Windows to operate over a network. By the same token, be sure that you adjust the following settings carefully — the more access you allow, the more vulnerable your system will become. Time to don our tin-foil hats!

❸ The first security setting that needs to be changed is to allow remote connections to the login screen of your system. To change this setting, click GNOME > System Settings > Login Screen. Click on the Security tab and uncheck the Always disallow TCP connections to X server option. When you click on the Close button, you will be prompted to restart your system.

After your reboot, check your firewall configuration — you need to allow TCP connections on port 6000 in order to allow remote applications to communicate with your X server. Ideally, you'll want to configure your firewall to only allow connections from the systems on which you specifically want to run applications. More detailed information on firewall configurationwill appear in next month's Mastering Linux Workshop.

THE XHOST WITH THE MOST

The xhost tool allows you to specify which hosts can be used for the running of remote applications. Allowing a host access is as simple as passing the hostname or IP address of the remote system as an argument to the `xhost` command, as follows:

```
xhost <hostname>
```

where <hostname> is the hostname, FQDN or IP address of the system on which you want to run remote applications. You can remove a remote host from this access list by repeating the above command with a minus (-) sign preceding the host name (without a space). To remove all remote hosts from this access list, simply enter the following command:

```
xhost –
```

CHANGE THE ENVIRONMENT

The last step is to change the DISPLAY environment on the remote host so that the applications are displayed on your local system. This requires physical access to the remote system, or access via Telnet or SSH. Another word of caution though: connecting to a remote system via Telnet is not secure, since data sent between systems is sent in plain text, and is able to be intercepted and "listened" to by malicious users.

Once you have a command line up on the remote system, change the DISPLAY environment variable as follows:

```
DISPLAY=<local host>:0
export DISPLAY
```

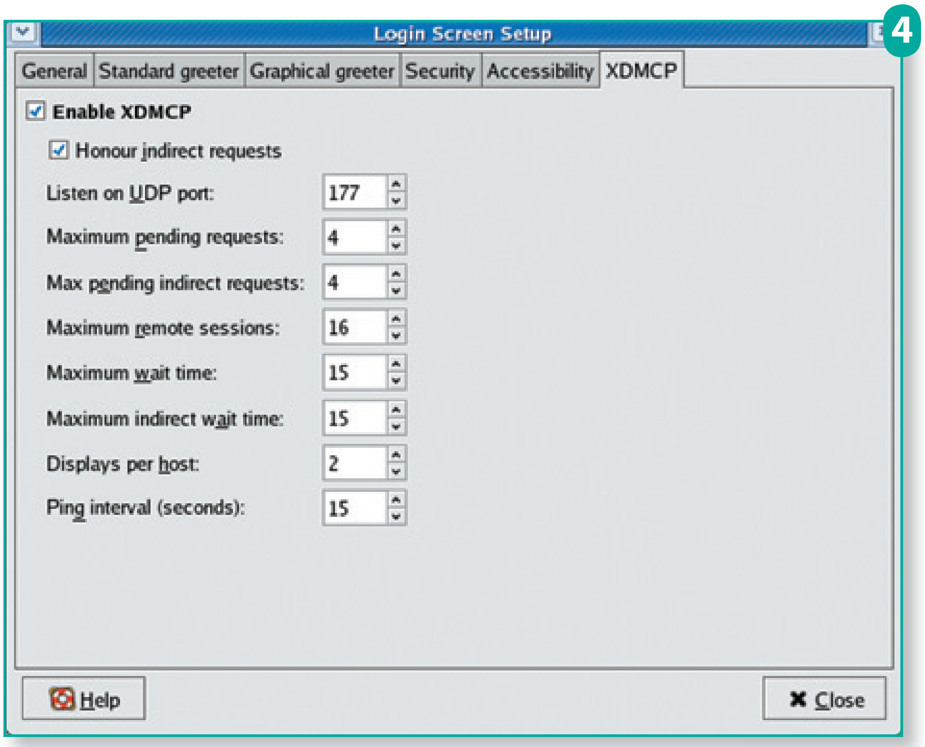
where <local host> is the hostname, FQDN or IP address of your local system. Any application that you run from this remote command line will be displayed on your local system.

To revert the DISPLAY variable so that applications launched from the remote system are once again displayed on the remote system, enter the following commands:

```
DISPLAY=:0.0
export DISPLAY
```

SETTING UP REMOTE X SESSIONS

Those familiar with the Remote Desktop Connection feature in Windows 2000, XP and 2003 will be accustomed to working with complete remote desktops. This is also possible with X, but you'll need to change a few security settings to enable the remote's functionality.


4

Login Screen Setup

General | Standard greeter | Graphical greeter | Security | Accessibility | XDMCP

☑ Enable XDMCP
☑ Honour indirect requests

Listen on UDP port:	177
Maximum pending requests:	4
Max pending indirect requests:	4
Maximum remote sessions:	16
Maximum wait time:	15
Maximum indirect wait time:	15
Displays per host:	2
Ping interval (seconds):	15

Familiar Remote Desktop: XDMCP should be enabled on a remote system in order to allow remote clients to connect to the X server.

The main setting you have to adjust is located under GNOME > System Settings > Login Screen. 4 This time, click on the XDMCP tab and tick the Enable XDMCP box. Advanced users (especially those with a good understanding of TCP/IP) may also want to configure some of the settings below this checkbox, but most of the default settings should suffice for most users.

You'll also need to ensure that your firewall is configured to allow inbound TCP and UDP connections on port 177 (or the port number that's listed next to the Listen on UDP port option under the XDMCP tab). If your firewall blocks these requests, you won't be able to access this system.

REMOTE X QUERIES

It's not possible to connect a remote X session from within an X session running on the local system. The reason for this lies in the way that X runs. On a standalone Linux PC, the X server runs in the background, and your X session connects to this server instance. A remote X session is started from the command line console on the local system and connects to the remote X server instance.

Most Linux installations (which have X Windows installed) boot directly to an X session connected to the local X server. The easiest way to connect to a remote X server is to switch to another virtual console running on your system and to establish another X session.

To switch to a different virtual console, you hold down the Ctrl and Alt keys simultaneously while pressing F1 through to F7, depending on which one of the seven available consoles you'd like to bring up. Your local X session should be running on console 7 (accessible via Ctrl+Alt+F7), while the other consoles should all be command-line driven.

Once you've switched to another console, log on to your local system and then execute the following command to attach that console to the remote X server:

```
XFree86 –query <hostname>
```

where <hostname> is the host name, FQDN or IP address of the system that you want to run remote applications on. apc

INSIDE INFO

● Red Hat Security Level Configuration Tool
ftp://195.220.108.108/linux/fedora/core/1/i386/os/Fedora/RPMS/redhat-config-securitylevel-1.2.11-1.i386.rpm

NEXT MONTH

● In next month's Mastering Linux Workshop, security is the focus – there'll be more detail about firewalls, and a demonstration of how to strengthen your system against attack.