Linux X Window System

Part 2 of 2

Lab

Section # O07

RETURN TO INDEX

Exercise Instructions With Hints

Configuring XFree86

__1. If you are in a runlevel that automatically starts X, switch to a runlevel that does not start X. If you started X manually, stop it.
» **# init 3**

__2. Make a backup copy of the XF86Config file.
» **# cp /etc/X11/XF86Config /etc/X11/XF86Config.bak**

__3. Try to configure your XF86Config file using redhat-config-xfree86 (Red Hat) or SaX (SuSE). If redhat-config-xfree86 or SaX yields a sufficiently good XF86Config file test this with startx), then make a backup of this file called XF86Config.redhat-config-xfree86 or XF86Config.sax2.
» redhat# **redhat-config-xfree86**
suse# **sax2**
» Answer all questions.
» **# startx**

__ 4. Try to configure your XF86Config file using XFree86 -configure. If XFree86 -configure yields a sufficiently good XF86Config file (test this with startx), then make a backup of this file called /etc/X11/XF86Config.XFree86.
» **# XFree86 -configure**
» Answer all questions.
» **# startx**

__ 5. Select the XF86Config file that worked the best for you and rename it to /etc/X11/XF86Config. Then, start X or switch to the runlevel that starts X for you.
» **# cp /etc/X11/XF86Config. something /etc/X11/XF86Config**
» **# init 5**

Running Applications

For an application to run in an X environment, it needs to be ready for it. Most applications that start with an "x" are. You can run them directly. Applications that are not X enabled, can be run from an "xterm" window. An xterm window emulates an ordinary terminal in an X environment.

__6. In X, start an xterm from one of the window manager's menus. In Linux, the xterm is sometimes also known as xterm, kterm or something else ending in -term. Try to run some commands from this window.

__ 7. There are several ways of starting applications within X. One is from the window

manager's menus, which we already did. Another is from the command line from an xterm window. Try that out: In the xterm window, run the command **xterm &**. You will see a new xterm window appear.
» **# xterm &**

__8. Another way of starting an application is from a real terminal window. Note that X started in virtual terminal number 7, so we can still access number 1 through 6. Not with Alt-F1 this time however, but with Ctrl-Alt-F1. Try that out. To switch back, use Alt-F7.
» **<Ctrl-Alt-F1>**
» **<Alt-F7>**

__9. Switch to a virtual window where you have an ordinary command line prompt. If necessary, log in as root.
» **<Ctrl-Alt-F1>**
» Login: **root**
» Password: **ibmlnx**

__10. Run xterm. You will see an error message: Can't open display. What happened is that the application wanted to contact the X-Server, but there was no information in the environment about which X-Server to contact. This information is normally stored in the $DISPLAY environment variable.
» **# xterm**

__11. Do an echo $DISPLAY in this window. Also do an echo $DISPLAY in an xterm window.
» **# echo $DISPLAY**
» **<Alt-F7>**
» **# echo $DISPLAY**

__12. Switch back to the true terminal and set the display variable with: **export DISPLAY=127.0.0.1:0.0.**. Now restart the xterm application. There should not be any error messages.
» **<Ctrl-Alt-F1>**
» **# export DISPLAY=127.0.0.1:0.0**
» **# xterm &**

__13. Switch to the X screen and note that you started an extra xterm window.
» **<Alt-F7>**

Running Applications Over a Network

Since applications use a TCP/IP connection to communicate to the X server, we can also run applications from another server.

__14. Walk over to the PC of a fellow student and ask him/her to open a terminal window for you.

__ 15. Set the display variable to point to your own screen. The command for that will look like this: **export DISPLAY=1.2.3.4:0.0**, where 1.2.3.4 is your own IP-address.
» $ **export DISPLAY=1.2.3.4:0.0**

__16. Now try to start an xterm. You should get an error message: could not open display. This is a safety feature of X: it does not automatically accept incoming connections.
» $ **xterm &**

__17. Go back to your own system and enter the command **xhost +** This will enable incoming connections.
» **# xhost +**

__18. Go back to the system of your fellow student and retry the **xterm** command. This time it should succeed.
» $ **xterm &**

__ 19. In your graphical screen you should see a new xterm. Try the **hostname** command in this screen to verify that the xterm application is actually running on the other system.
» **# hostname**

__ 20. Close the xterm and do an xhost -. Then try to open another xterm from your partners system, but this time use xauth authentication.
» **# xhost -**
» **# xauth extract xauthfile hostname:0.0**
» Transfer the xauthfile to the other machine, for instance using
» **# scp xauthfile root@otherhost:xauthfile**
» On the other host:
» **$xauth merge xauthfile**
» **$xterm -display hostname:0.0**

Running X-sessions over a network
You can not only run a single application over a network, but you can run your whole X-session over a network as well. In this case it's not only the application that is running on a remote system, but the window manager as well. In fact, the only program that needs to be running locally is your XFree86 server.
For this part of the exercise to work, you need to work with a partner team. You can only perform step 22 after your partner team has finished step 21.

__ 21. Make all necessary changes to the configuration file of your favorite login manager to enable remote logins. Then restart your login manager by switching to runlevel 3 and then to runlevel 5 again. To determine the display manager you' running:

» **# ps ax | grep dm**

For **xdm:**
» # **init 3**
» # **cd /etc/X11/xdm**
» # **vi Xaccess**
Uncomment the line which only has an '*' on it. (This is done already on SuSE.)
» # **vi xdm-config**
Comment out the last line that specifies " 0"
» # **init 5**
For **kdm**:
» # **init 3**
» redhat# **cd /etc/kde/kdm**
suse# **cd /etc/opt/kde3/share/config/kdm**
» # **vi kdmrc**
In the **[Xdmcp]** section, change **Enable=false** to **Enable=true**
» # **vi Xaccess**
Uncomment the line which only has an '' on it. (This is done already on SuSE.)
» # **init 5**
For **gdm:**
» # **init 3**
» redhat# **cd /etc/X11/gmd**
suse# **cd /etc/opt/gnome2/gdm**
» # **vi gdm.conf**
In the **[xdmcp]** section, change **Enable=false** to **Enable=true**
» # **init 5**

__22. Start a second X-server, this time telling X to get its login manager from your partner system.
» # **X -query <hostname> :1**

__ 23. Stop the second session and start it again, but do an indirect broadcast for a login manager. You should get a chooser which allows you to login to any system running a display manager on the network.
Note. There seems to be a problem with indirect queries and **kdm** in Red Hat 7.3 and up. At the time of this writing, this had not been resolved.
» # **X -indirect <hostname> :1**

END OF EXERCISE

RETURN TO INDEX