Linux XDMCP HOWTO

Thomas Chao

tomchao@lucent.com

Revision History

Revision v1.2	15 March 2002	Revised by: tc
Adding more info for Red Hat 7.2, Mandrake 8.1 and Slackware 8.0 Linux configuration and SSH X11 Forwarding.		
Revision v1.1	20 March 2001	Revised by: tc
Revision and adding RH 7.0.		
Revision v1.0	01 November 2000	Revised by: tc
Initial revision and release.		

This HOWTO describes how you can use the combination of X Display Manager (xdm, kdm and gdm) and XDMCP (X Display Manager Control Protocol) to provide the mechanism for an X−Terminal and a platform of cheap Remote X Apps solution. This document will be focusing on how to setup connection using XDMCP.

Table of Contents

1. Introduction

XDMCP stands for "X Display Manager Control Protocol" and is a network protocol. It provides a mechanism for X−Server to emulates the X−Terminal to run on your PC (or MAC). This allows the X−Server to run on one or multiple X Window based applications that resides on a host machine. The X−Terminal can be displayed with an individual windows or multiple windows, based on your X client software capabilities.

Some of us who uses Linux (like me) are looking for the best parts of Linux. Among them is the ability to re−use old systems (like 486 and Pentium, Pentium II CPUs) as a X−Terminal (with the Win32 apps; like Hummingbird's Exceed, X−Win32 or X−ThinPro. For MAC, try eXodus) to run Linux X solution from any PC remotely. It is somehow very surprising that there aren't many documents on the Internet which guide you step by step on how to set this up. This is how I come up with this document as a way to share my experience with all users. Essentially, by using X and XDMCP, you can create a good, non−expansive solution of a X−environment.

1.1. Disclaimer

No liability for the contents of this documents can be accepted. Use the concepts, examples and other content at your own risk. As this is a new edition of this document, there may be errors and inaccuracies, that may of course be damaging to your system. Proceed with caution, and although this is highly unlikely, the author(s) do not take any responsibility for that.

All copyrights are held by their by their respective owners, unless specifically noted otherwise. Use of a term in this document should not be regarded as affecting the validity of any trademark or service mark.

Naming of particular products or brands should not be seen as endorsements.

You are strongly recommended to take a backup of your system before major installation and backups at regular intervals.

1.2. Feedback

Feedback is most certainly welcome for this document. Without your submissions and input, this document wouldn't exist. Please send your additions, comments and criticisms to the following email address : `<tomchao@lucent.com>`.

2. The Procedure

This section details the procedures for setting up X−Terminal using XDMCP.

2.1. Before you begin, some backgrounds

Before you begin, it is better to have a basic understanding of how this works. (More details are at the Resources below and LDP HOWTO page)

The X server is usually started from the X Display Manager program (xdm, kdm and gdm. This document will use gdm as an example). It provides a nice and consistent interfaces for general users (X−based login, starting up a window manager, clock, etc.). X Display Manager manages a collection of X displays, which may be on the local host or remote servers.

When xdm runs, it is usually run as a local copy of X, also xdm can listen for requests from remote hosts over a network. For kdm (which comes with the KDE desktop), it is a replacement of xdm and configures the same way, except its files are in `/etc/X11/kdm`. The gdm (Gnome Display Manager) is a re−implementation of the xdm program. gdm has similar functions to xdm and kdm, but was written from scratch and does not contain any original XDM / X Consortium code.

In the case of xdm, it offers display management in two different ways. It can manage X servers running on the local machine and specified in X−servers, and it can manage remote X−servers (typically X−terminals) using XDMCP (the XDM Control Protocol) as specified in the Xaccess file. (Courtesy of xdm man page).

Other good references for the similar setup can be found in the following documents:

- The XDM and Xterminal mini−HOWTO, by Kevin Taylor
- Linux Remote X Apps mini HOWTO A very good reference for Remote X in both theoretical and practical view. By Vincent Zweije
- The Xterminal mini−HOWTO, by Scot W. Stevenson

2.2. Security Reminder

Using XDMCP is inherently insecure, therefore, most of the distributions shipped as it's XDMCP default turned off. If you must use XDMCP, be sure to use it only in a trusted networks, such as corporate network within a firewall. Unfortunately, XDMCP uses UDP, not TCP, therefore, it is not natively able to use it with SSH. To secure the connection with SSH, the technique is called X11 TCP/IP Port Forwarding. Check this Why Port Forwarding? site and the Resources area for additional HOW−TO information. If you would like to experiment this, I have added a new section below to show you the basic idea of how it works, and I am leaving the more advanced way of running it to other experts and/or HOWTOs.

2.3. The System I use

I have tested the setup running a GNOME (gdm), as well as KDE (kdm) on Red Hat 6.0, 6.2 and Red Hat 7.x (up to 7.2). (Thanks to Peter van Eerten in Netherlands who provides info regarding Slakware 8.0 setup.

Many others also provide me info regarding different distributions. I would like to thank them as well). The other I have tried on are Caldera eDesktop 2.4, which is similar to RH's setup. I have also test it on current Mandrake version (V8.1) without a problem. I have not had a chance to test it on other Linux flavors like Debian and Slackware (One Slackware user told me it works the same way as mentioned in this document). However, the setup should be similar and should works fine. If you have successfully setup one other than the Red Hat, Caldera and Mandrake platform, please share it with me. I will add them into this document.

My server hardware is an IBM PC clone running an Intel Pentium II 500 MHz with 256 MB memory and 20 GB ATA−66 Hard Drive. (I found out that my old Pentium 100 MHz PC runs this just fine). I use a 3COM 10/100 Fast Ethernet (3C509B) NIC with an ATAPI 32X CD−ROM and an IOMEGA ZIP drive. I have also test it on my Toshiba Tecra 8100 laptop connecting using my Lucent/Agere Orinico Wireless LAN card (80211.b).

2.4. Remote piece

I use the Hummingbird Exceed 6.x (with Service Pack), Exceed 7.x and have tested them on Windows 98 SE, Windows NT 4.0 and Windows 2000 Pro. I found out that another popular choice are X−Win32 and X−ThinPro. However, there are many open−source apps as well as commercial one available.

2.5. Server Preparation

In RH 7.x, you need to setup DNS lookup, in order for some networking function to work properly (such as **telnet**). If you are in a small isolated environment (like home or small office, etc.) that do not have access to a public DNS Server, then add entry of the working DNS Server name(s) (such as your ISP's) in the `resolv.conf` file or you can add the host name of all workstations in your local `host` table.

To prepare your X−Server for XDMCP session, you need to make sure the following are properly installed:

1. Install your Linux OS. In my case, I installed Red Hat 7.2 (Custom Installation). If you plan to use SSH Port Forwarding, you need to compile SSH with your kernel. Also, RH 7.x comes with firewall installed as default. You will encounter problem, if you do not add firewall rules or temporary disable it for setting up XDMCP. I will not cover the firewall rules here in details, since this is not the focus of this document. I will share only how to make it works first and you can fine−tune it yourself.

   To show your firewall rules, use the command **ipchains −L** to list your default rule sets. To temporary disable it, use this command **ipchains −F** to flush the rules (Don't worry, it will restore by re−loading or re−boot). One user, Ryan Sheidow, shared with me that by adding this rule, you can do it without disable your firewall and can allow yourself to access the X−Server (you can try for yourself).

   ```
   ipchains −A input −p udp −i $extint −−dport 177 −j DENY
   ```

   For more firewall details, check the IP Masquerade HOWTO page.

   One other easy way is to add rules that only accept certain IP address(es) from your trusted workstations. This is how I use it myself.

Linux Kernel 2.4x shipped with new firewall app called **iptables**. Please feel free to experiment it. Again, I will not cover it here.

2. Setup your Networking. To test it out, **ping**, **ftp** and **telnet** are good commands to use to determine if your network works. RH 7.2 do not have **telnet** daemon turn on by default. Remember to enable it, if you prefer to use it for your test. One other thing is to remember firewall rules are there. Add your own rules or temporary disable it (as mentioned above) to make these commands work.
3. Setup X. Do *not* setup with a resolution higher than what the remote users are able to use for their display. Test the X−Server by typing either **startx** or **telinit 5**. Make sure X is running properly.
4. Creates the necessary user accounts (and associated groups) for user who will access via the X−Terminal.

2.6. Steps to Complete the Procedures

These are steps I used to setup the X−server for accepting XDMCP requests:

1. For RH 6.2, modify `/etc/rc.d/init.d/xfs` and make the following changes. Change all (this is where the Font Server port):

   ```
   daemon xfs −droppriv −daemon −port −1
   ```

   to:

   ```
   daemon xfs −droppriv −daemon −port 7100
   ```

   In Mandrake 7.2, the port is already set to 7100. Also, in RH 7.x, you do not need to do this, since by default, it is, for security enhancement, not listening to TCP port any longer! If you need to setup default font server to use, do it in /etc/X11/fs/config and add the setting there. Different Linux distribution may put the xfs in different folder under /etc/rc.d. You may search for it if that's the case.

2. Modify `/etc/X11/xdm/xdm−config` and make the following change. Be default (in most Linux distributions), this line is set, so that it is not listening to XDMCP connection. This is for security reason. For Caldera using kdm, this file is at `/etc/X11/kdm`. Find this line:

   ```
   DisplayManager.requestPort:     0
   ```

   and comment it out as:

   ```
   ! DisplayManager.requestPort:     0
   ```

   Remember, this does not affects gdm. For gdm setup, it is in the following section.

3. In `/etc/X11/xdm/Xaccess`, change this. (this allow all hosts to connect). For Caldera using kdm, this file is at `/etc/X11/kdm`. Set the security to 644 (chmod 644):

   ```
   #*     # any host can get a login window
   ```

   to:

   ```
   *      # any host can get a login window
   ```

xdm usually run as a local copy of X and can listen for requests from remote hosts over a network. xdm reads its configuration files `/etc/X11/xdm/xdm−config` for all configuration and log files that xdm uses. For kdm, it is a replacement of xdm and configures the same way, except its files are in `/etc/X11/kdm` for Caldera. It is worth noting that the **Xsession** file is what runs your environment.

The gdm (Gnome Display Manager) is a re−implementation of the well known xdm. gdm has similar functions to xdm and kdm, gdm is the Gnome Display Manager, and its configuration files are found in `/etc/X11/gdm/gdm.conf`. The `gdm.conf` file contains sets of variables and many options for gdm, and the Sessions directory contains a script for each session option; each script calls `/etc/X11/xdm/Xsession` with the appropriate option.

The above setup is in a Broadcast mode, which will list all the X−Server that are listening and willing to manage your X connection. If you only want to allow certain connections, use the **CHOOSER** section in this same file. An example can be found in the Resources.

4. I use the gdm as default and use gdm login window to switch between KDE and GNOME. For gdm, edit `/etc/X11/gdm/gdm.conf`. This activates XDMCP, causing it to listen to the request. (For kdm, if you are using KDE2, edit `/usr/share/config/kdm/kdmrc` or `/opt/kde2/share/config/kdm/kdmrc` for Slackware version). Change this:

```
[xdmcp]
Enable=0
```

to:

```
Enable=1
```

Make sure "**Port=177**" is at the end of this block. For Caldera using kdm, modify this file `/usr/share/config/kdm/kdmrc`.

5. Now edit `/etc/inittab` and change the following line:

```
id:3:initdefault:
```

to:

```
id:5:initdefault:
```

Before changing this line, you can use the **telinit** command (or preferably **ssh** command) to test prior to modifying the line. Use either **telinit 3** to set to level 3, or **telinit 5** to set to level 5, graphics mode (you can issue this command on the second machine that telnets into this server).

6. Make sure the proper security of the file `/etc/X11/xdm/Xservers` is set to 444 (chmod 444).
7. Locate `/etc/X11/xdm/Xsetup_0` and **chmod 755** this file.
8. Edit the `XF86Config` file (if you are using XFree86 4.x, the file is `XF86Config−4`) at `/etc/X11` and change the line, if you are using RH Linux:

```
FontPath    "unix:-1"
```

to:

```
FontPath    "unix:7100"
```

9. (You **do not** have to make this change. You can keep the default setting, but this is what I use. If you are not sure, leave this alone.) Add this line to the end of `/etc/inittab`:

```
x:5:respawn:/usr/bin/gdm
```

You are now ready to run a test.

One other thing to know (that some users have asked) is how to display with **Willing to manage** message with load info As I know this is available in xdm by adding the following to the `/etc/X11/xdm/xdm-config`.

```
DisplayManager.willing:  su noboby -c /etc/X11/xdm/XWilling
```

and the XWilling script must exist. For gdm, add this line to the `/etc/X11/gdm/gdm.conf` in `[security]` section:

```
Willing=/etc/X11/gdm/Xwilling
```

2.7. Testing

To test if your XDMCP with X−Server is ready to accept connections, do these steps. I find it easier using the X−Server and another machine to test it:

1. Restart your display manager gdm (or xdm and I am assuming you are running level 5). If you are not sure how to do this, simply reboot your system (but this is really not necessary, if you know how to restart it using command line. That's the beauty of Linux, comparing to my Windows).
2. If you have not modify your firewall rules, you need to temporary disable it by using **ipchains −F**.
3. Make sure the Graphical login page comes up. Make sure the display resolution and mouse work. Log in from the console to see if the local access is OK. If OK, do not log off.
4. Setup Hummingbird Exceed to either query this machine (using the IP address or fully qualified DNS name) or set to use XDMCP−Broadcast and try to connect to the X server. You should see the X Session come up and the login screen appear.

3. X11 Forwarding using SSH

As I have explained earlier, using XDMCP to display X across Internet is basically a no−no, due to it's lack of encryption across the Internet. One way of enforce the traffic security is to use the SSH by the way of X11 forwarding. SSH (Secure Shell) is developed in 1995 by Tatu Ylonen to replace the insecure **telnet** and **ftp**. The first thing you need to know is that X11 forwarding using SSH is different from your regular, non−secure way of running X Window.

To start this setup, you need an additional piece of information. First, you must have your SSH package installed. In Linux, they are the OpenSSH packages. Check your distribution to decide what package you need to install (some installed it as standard packages). Secondly, you need a Windows SSH Client (other OS version, like MAC, are also available). I recommend PuTTY. It is a wonderful free SSH client and you can download them from this link. Download the document as well and read them carefully. The other good free SSH client are: Tera Term Pro + TTSSH: An SSH Extension to Tera Term, SSH Secure Shell Client by SSH.com (only free for non−commercial use). I will break down again into steps, so it is easy for you to follow.

1. Open up the command **putty.exe** by double−click it. It will brings up the interface. First, setup the connection info in Host Name (or use IP) field and select SSH (SSH is using port 22). In Connection Category, find the Connection tree. In SSH, expand it and you will see Tunnels window. Click "Enable X11 forwarding". It is setting the default to X display at "localhost:0". Now, go back to Session and save this session with a name you like. I normally use the Host Name to make me easily remember where I am connecting to.
2. In the example of Hummingbird Exceed, this is what you need to do. (For other X client, the setup is similar). Open up the Xconfig from your Exceed folder. In your "Screen Definition", change to "Multiple" Window mode and save it. Next, open up your "Communication" icon and set the Startup mode to "Passive".
3. Now you are done. To test it, first using PuTTY (or other SSH client) to connect to your server. The first time connection, it will ask you whether you want to cache the Security Key or not. (Yes is normal choice). Once log in is done, fire up your Exceed. It will stay in the background. Now you can execute any of your X application and it should forward the X application via SSH to your local screen. For example:

   ```
   $ xclock &
   ```

   We should now see the Xclock is running on your local screen.

Now you see the difference is that you do not see all your X Window. You are simply running X application one by one and forwarding via SSH to your local screen. Therefore, you need to know the command for running each X application. All the control are done via SSH client window. To me, the security is worth the slightly inconvenience!

For user running the X−Win32, you can use this link for your SSH + X11 forwarding setup.

4. Troubleshooting

- If X cannot come up and is broken:

  If X is broken and the connection fails, most of the time it has this error messages:

  ```
  _ FontTransSocketUNIXConnect: Can't connect: errno = 111
  failed to set default font path 'unix:-1'
  Fatal server error:
  could not open default font 'fixed'
  ```

  This is likely due to xfs not finding the correct port for the Font Server (again, if you are running RH 6.2). To resolve this, check steps 1 and 7 above. Make sure the configuration are pointing to (port) 7100 and make sure you have the following fonts installed (if not re−install the XFree86 font packages from your CD). Check the listing in `XF86Config` file (if you are using XFree86 4.x, the file is `XF86Config−4`) at `/etc/X11`:

  ```
  FontPath  "/usr/lib/X11/fonts/75dpi/"
  FontPath  "/usr/lib/X11/fonts/misc/"
  FontPath  "/usr/lib/X11/fonts/CID"
  FontPath  "/usr/lib/X11/fonts/Speedo"
  FontPath  "/usr/lib/X11/fonts/100dpi"
  ```

  Use the command **startx** (on local) to restart the X server (or use **telinit 5** to switch the run−level).

- If Exceed has no respond (in blank screen):

  In this case, most likely your xdm (or gdm, depending upon which is used in `/etc/inittab`) is not starting correctly. Issue the command: **ps −ef | grep gdm** (or **ps −ef | grep xdm** if xdm is used). Also, if your box has udp port turned on for XDMCP, you can type **netstat −l | grep xdmcp** and you should see this:

  ```
  udp     0     0   *:xdmcp          *:*
  ```

  If the process is not running, check the steps on the setup above (make sure there are no typos and that the correct path is given). Restart X using the command **telinit 5**. If the udp port is not there for XDMCP, do step 2 as above.

  Another possibilities are that your DNS setup is incorrect and/or firewall is enabled. An easy way to find out is simply **ping** or **telnet** your host and if the reply takes a long time, then that's DNS problem. If by using **telnet** and you got a `"Connection Refused"`, then this is a firewall problem (assuming that you have your telnet daemon turned on already)! Check the section above for details how to resolve this.

- PC Box with PPPoE (PPP over Ethernet):

  A user using PPPoE told me that if you have PPPoE, you might experience problem using XDMCP. After uninstall it, he then is able to get XDMCP working. I personally do not have the environment to test this, so you can test it yourself.

- Linux to Linux Display export:

If you are using another Linux with X, you do not need to use XDMCP to manage your display. You can actually export your display right in your X box. To do this, you must enable your access control to allow to make connection to the X Server. The common error you will get without doing so are:

```
xlib: Connection refused (error 111): unable to connect to X server
xlib: No such process (error 3): Server error
```

To resolve the problem, use the command below:

```
$ xhost +
$ export DISPLAY=(your local host IP):0.0
```

Always remember to enable access control by using the command "`xhost -`" again. One thing to remind you, you do not need this, if you are using PC as X−Terminal using XDMCP. This is only required when you have Linux to Linux or Linux to UNIX connection.

If you are using many Linux X boxes and you would like to setup the Chooser to pick from which X to login, you need to enable the following in the `/etc/X11/gdm/gdm.conf`:

```
[daemon]
Chooser=/usr/bin/gdmchooser --disable-sound --disable-crash-dialog
...
[xdmcp]
Enable=1
HonorIndirect=1
```

- I got a "Signal 11" error:

  The "Signal 11" error, also called "Segmentation Fault", can sometimes be a problem of your hardware and/or software. If you have this problem in bring up the X−server, you need to fix it before configuring XDMCP. Unfortunately, there is no simple way to fix the problem due to many possible causes. For details, please check this SIG 11 while compiling the Kernel.

5. XDMCP and GDM (Gnome Display Manager)

The following is taken from the Gnome Display Manager Reference Manual:

GDM also supports the X Display Manager Protocol (XDMCP) for managing remote displays. GDM listens to UDP port 177 and will respond to QUERY and BROADCAST_QUERY requests by sending a WILLING packet to the originator. GDM can also be configured to honor INDIRECT queries and present a host chooser to the remote display. GDM will remember the user's choice and forward subsequent requests to the chosen manager. GDM only supports the MIT−MAGIC−COOKIE−1 authentication system. Little is gained from the other schemes, and no effort has been made to implement them so far. Since it is fairly easy to do denial of service attacks on the XDMCP service, GDM incorporates a few features to guard against attacks. Please read the XDMCP reference section below for more information.

Even though GDM tries to outsmart potential attackers, it is still advised that you block UDP port 177 on your firewall unless you really need it. GDM guards against DoS attacks, but the X protocol is still inherently insecure and should only be used in controlled environments. Even though your display is protected by cookies the XEvents and thus the keystrokes typed when entering passwords will still go over the wire in clear text. It is trivial to capture these. You should also be aware that cookies, if placed on an NFS mounted directory, are prone to eavesdropping too.

6. Additional References

Some additional references on this subject include:

- Your local xdm man page.
- Your local gdm man page.
- Configuring XDM
- xdmcp/udp
- XDMCP Documentation (File download)
- Should you be running XDMCP?
- X Window System Terminals
- A second way of using XDM
- Accessing Xterms from Windows
- How to install X−Win32
- Taming the X Display Manager
- Why Port Forwarding?, Port Forwarding, SSH: X11 Forwarding
- GNOME Display Manager
- Through the X Window (for MAC)
- Securing Services on your system (Debian)
- Remote X using KDM (Caldera)

---

7. Authors

Current: Thomas Chao, Lucent Technologies

8. Copyright Information