The LBX Mini−HOWTO

Table of Contents

The LBX Mini−HOWTO......1
Paul D. Smith, psmith@baynetworks.com......1
1. Introduction......1
2. What's The Status Of LBX?......1
3. Who Can Benefit From LBX?......1
4. Who Doesn't Need LBX?......1
5. How Does LBX Work?......1
6. What Do I Need To Use LBX?......1
7. What Don't I Need To Use LBX?......1
8. How Do I Start LBX?......1
9. Problems......1
10. Documentation......1
11. Alternatives......1
1. Introduction......1
2. What's The Status Of LBX?......2
3. Who Can Benefit From LBX?......2
4. Who Doesn't Need LBX?......2
5. How Does LBX Work?......2
6. What Do I Need To Use LBX?......3
7. What Don't I Need To Use LBX?......3
8. How Do I Start LBX?......4
9. Problems......4
10. Documentation......5
11. Alternatives......5
11.1 dxpc − The Differential X Protocol Compressor......5
Advantages......6
Disadvantages......6
Where Can I Get dxpc?......6
11.2 Ssh (Secure Shell)......6
11.3 Which Is Better?......6

The LBX Mini−HOWTO

Paul D. Smith, `psmith@baynetworks.com`

v1.04, 11 December 1997

LBX (Low Bandwidth X) is an X server extension which performs compression on the X protocol. It is meant to be used in conjunction with X applications and an X server which are separated by a slow network connection, to improve display and response time.

---

1. Introduction

2. What's The Status Of LBX?

3. Who Can Benefit From LBX?

4. Who Doesn't Need LBX?

5. How Does LBX Work?

6. What Do I Need To Use LBX?

7. What Don't I Need To Use LBX?

8. How Do I Start LBX?

9. Problems

10. Documentation

11. Alternatives

- 11.1 dxpc − The Differential X Protocol Compressor
- 11.2 Ssh (Secure Shell)
- 11.3 Which Is Better?

---

1. Introduction

Low−Bandwidth X (LBX) attempts to recognize that in this day and age, not everyone will be a fast LAN hop or two away from the system that they are running their applications on.

The X protocol can generate an extraordinary amount of traffic, especially for simple−seeming things such as creating new windows. As anyone who has tried to use X over a dial−in modem at 28.8 or even higher can attest, creating new X windows can involve an excruciating wait.

LBX is fundamentally a compression and caching scheme designed to minimize the amount of X traffic generated between two systems.

---

2. What's The Status Of LBX?

As of the X Consortium's release of X11R6.3 in December, 1996, LBX is a full extension to the X protocol. For XFree86 folks, that's XFree86 version 3.3.

---

3. Who Can Benefit From LBX?

If you use a modem to dial into a service provider, then run X applications on remote machines with their DISPLAYs set to your local machine (or vice versa), LBX will speed up that connection. Also if you set DISPLAYs from systems across WANs (other countries, for example) or other slow links, LBX can help.

---

4. Who Doesn't Need LBX?

LBX is useless, of course, if you're only running applications locally, or if you're not running X at all.

Also, if you're running on a fast LAN, LBX won't be much help. Some people say "if LBX cuts down on network traffic, wouldn't it be good to use even on fast LANs?" It might be, if your goal is to reduce network traffic. But if your goal is to get better response time LBX probably isn't what you want. Although it does introduce caching and compression, that comes at a cost on both ends (extra memory for caching, and extra CPU for decompression). If your link is fairly speedy LBX will probably result in an overall slowdown.

---

5. How Does LBX Work?

LBX works by introducing a *proxy server* at the client side, which performs caching and compression. The X server knows that the client is using a proxy server, and decompresses accordingly.

Here's a normal setup for remote X clients. In our discussion, LOCAL is always the workstation sitting in front of you, whose monitor you're looking at, and REMOTE is the remote workstation, where the actual application is running.

```
    REMOTE                                LOCAL
+-----+                                                  +-----+
| APP |-\          Network          +-----------+  |     |\
+-----+  \--------------------------->| X SERVER |=>|     ||
+-----+  /       (X Protocol)        +-----------+  +-----+\
```

```
| APP |-/                                          /_____//
+-----+
```

When using LBX, a proxy server (`lbxproxy`) is introduced on the remote side, and the applications talk to that process instead of directly to the LOCAL server. That process then performs the caching and compression of X requests and forwards them. It looks like this:

```
    REMOTE                                          LOCAL
                                                              +------+
+-----+  +--------+           Network          +----------+  |      |\
| APP |->| PROXY |----------------------------->| X SERVER |=>|      ||
+-----+  +--------+       (LBX/X Protocol)      +----------+  +------+\
+-----+   /                                                  /_____//
| APP |--/
+-----+
```

Details on exactly what caching and compression LBX does is beyond the scope of this document.

6. What Do I Need To Use LBX?

You need an X server on your LOCAL system which has the LBX extension compiled in. Unless you explicitly told it not to when building it, X11R6.3 servers automatically enable LBX. Also, all XFree86 3.3 servers have LBX enabled by default.

You can use the `xdpyinfo` command to see if your server has the LBX extension: run `xdpyinfo` and look at the list just under "number of extensions"; you should see "LBX" listed there.

Next, you need to get an `lbxproxy` program compiled for the REMOTE system. This is the tricky part. If the remote system is not the same type as your local system, the `lbxproxy` on your local system will do you no good, of course.

There is unfortunately no "broken out" distribution of `lbxproxy`, so you will have to either (a) get and build most, if not all, of X11R6.3 for the remote system, or (b) find someplace to get a pre−compiled `lbxproxy` binary for your system. The latter is much simpler of course.

The `lbxproxy` is simply a single executable. There are no configuration files, resource files, etc. associated with it.

7. What Don't I Need To Use LBX?

The REMOTE system **does not** need a new X server (as always, the REMOTE system doesn't need *any* X server running).

The application you want to run **does not** need to be linked with any special version of X, or any special libraries; I regularly use commercial X11R5 apps over LBX with no trouble.

You **do not** need root or other privileged access on the REMOTE system; the `lbxproxy` process runs under your normal access permissions. Further, you can run it right from your home directory: it does not have to be installed anywhere.

8. How Do I Start LBX?

OK, here it is... after all that it's actually quite simple. Replace LOCAL and REMOTE below with the hostnames of your local workstation and remote system, respectively (don't get them mixed up!)

On LOCAL:

1. Start your X server.
2. Tell your X server that the remote system is allowed access. Using the host−list method, type `xhost +REMOTE`. If you use `xauth` you may need to do more than this; see the *xauth(1)* man page for more information. You should consult the Remote X Apps Mini−HOWTO if you're not familiar with remote X access permission setup.

On REMOTE:

1. Start `lbxproxy` and tell it to forward to the LOCAL X server, like this:
   ```
   $ lbxproxy −display LOCAL:0 :1 &
   ```
   This tells `lbxproxy` to use display `:1` on the REMOTE system; if that system has >1 display already you can use `:2` or whatever instead.
2. Set your DISPLAY environment variable to point to the display that `lbxproxy` is providing, instead of the normal display:
   ```
   $ DISPLAY=:1
   $ export DISPLAY
   ```
   Or, if you use csh or clones:
   ```
   % setenv DISPLAY :1
   ```
3. If you're using `xauth` you will need to ensure that your cookie is available locally. See the Remote X Apps Mini−HOWTO for more information on this.
4. Start your X applications!

That's it; all X apps that are started up pointing to `:1` will use LBX. Of course, there's no reason you couldn't also start X apps pointing to `LOCAL:0` and have both running at the same time.

9. Problems

Here are some common problems:

Q)

`lbxproxy` exits with an "access denied" error.

A)

This means the LOCAL system isn't accepting connections from the REMOTE system due to permissions errors. See the Remote X Apps Mini−HOWTO for details on these issues.

As a simple trouble−shooting measure, try running a simple X app like `xclock` on REMOTE and have it display on the local system without using `lbxproxy`:

```
$ xclock −display LOCAL:0
```

If that doesn't work, it's `xhost` or some other basic X problem, not LBX.

10. Documentation

The only documentation available in a standard X distribution may be the *lbxproxy(1)* man page.

If you have access to the X source tree, then very interesting information on LBX is available there:

- `xc/doc/specs/Xext/lbx.mif` (Framemaker MIF)
- `xc/doc/hardcopy/Xext/lbx.PS.Z` (Compressed Postscript)
- `xc/doc/hardcopy/Xext/lbxTOC.html` (HTML)

More detailed discussion of specific LBX algorithms is available here:

- `xc/doc/specs/Xext/lbxalg.mif` (Framemaker MIF)
- `xc/doc/specs/Xext/lbxalg.PS.Z` (Compressed Postscript)

If you don't have access to the X11 source, you can obtain these files from the X Consortium's FTP site.

11. Alternatives

If you don't like `lbxproxy` for some reason: you're not satisfied with the performance, it doesn't work for you, you don't want to hassle with creating an lbxproxy for the remote host, or you simply are interested in trying other options, there is at least one other package for X protocol compression (anyone have others?)

11.1 dxpc − The Differential X Protocol Compressor

- Original Author: Brian Pane <brianp@cnet.com>
- Current Maintainer: Zachary Vonler <lightborn@mail.utexas.edu>

`dxpc` works in essentially the same way as LBX. However, to avoid having to implement an X extension and modify the X server code, `dxpc` uses **two** proxies: one that runs on the REMOTE host, like `lbxproxy`, and one that runs on the LOCAL host.

The REMOTE host proxy communicates between the X clients and the LOCAL host proxy, and the LOCAL host proxy communicates between the X server and the REMOTE host proxy.

So, to *both* the X clients and the X server, it looks like X protocol as usual.

Advantages

- Since it's a completely separate application that does not require any X internals, it's *much* simpler to compile and install.
- It's maintained separately, so you don't have to wait for the OSF to release new X versions for enhancements or fixes.
- It provides more and better compression information and statistics than `lbxproxy`.

Disadvantages

- It is not a standard part of X; you must obtain and build it separately.
- It is slightly more complex to set up, since it requires a LOCAL−side proxy as well as the REMOTE proxy.

Where Can I Get dxpc?

The source for dxpc is available at ftp.x.org.

There is a WWW homepage for dxpc that gives a lot of good information, including pointers to the dxpc mailing list, access to the source code, and a number of pre−built binaries for various platforms:

http://ccwf.cc.utexas.edu/~zvonler/dxpc/

11.2 Ssh (Secure Shell)

Ken Chase <lbxhowto@sizone.org> notes that `ssh` can be used for compression. Although its main purpose is to provide security, it also compresses the data it sends.

Thus, if you run X over a `ssh` link you will automatically obtain some amount of compression.

11.3 Which Is Better?

I don't know. Both LBX and `dxpc` are certainly better at raw compression than `ssh`. Of course, `ssh` provides the added advantage of security. And of course, there's no reason you can't use both `ssh` and one of the other two, to get good compression and security.

It shouldn't be hard to run some benchmarking against these options and get both subjective and statistical measurings of performance. But I haven't done this, and I don't know of anyone who has.