Small Memory mini-HOWTO

Todd Burgess

tburgess@uoguelph.ca

2000-12-12

Revision History

Revision 0.1 2000-12-01 Revised by: tb

Describes how to run Linux on a system with a small amount of memory.

1. Introduction

Assuming buying more memory is out of the question there are many things you can do to tighten up memory usage in Linux.

Many Linux distributions out of the box are quite bloated from a memory perspective. They run more services and offer more features than most of us will ever need. By removing many of these services you can free up several megabytes of real memory.

My own system is a 486DX2-66 with 12MB of physical memory and 12MB of swap space. It has run Linux for the last 3 years quite happily, and hopefully it will run Linux for several more years. :)

2. Linux Kernel

All the Linux kernels which come with distributions are quite bloated and contain more features than any of us will ever need or use. If you have not re-compiled your own kernel, it is highly recommended that you do so. How to re-compile a kernel is beyond the scope of this document, but many excellent Linux books and guides cover this subject in intimate detail.

If you do re-compile your kernel, remember to put in no more features than you need. For instance: how many of you include PLIP support in your kernel? How many of you who include it actually use it? Smaller kernels require less time to load, use less memory, and use less CPU cycles.

Another thing is modules. I personally do not use them because I found them to be a cumbersome. If you use them and like them then they can help to relieve "kernel bloat."

3. Virtual Consoles

VCs are a great way to free up memory. Most Linux distributions run about 6 of them out of the box. On average running 6 VCs requires about 4MB of memory. Removing a couple of them can free up a couple MBs of memory. Most users can get away with running only 3 or 4 VCs. How many you choose to remove is a matter personal preference. Just remember that the fewer you run, the more memory your applications will have to run.

The file which outlines how many VCs get loaded is `/etc/inittab`. In order to remove VCs:

1. Load `/etc/inittab` in a text editor.
2. Look for a line which looks like the following line (the key feature being a line which starts with c1):

   ```
   c1:12345:respawn:/sbin/getty tty1 38400
        linux
   ```

   Start at the highest number (i.e. c6) and comment it out by inserting a '#' in the first row. Repeat this step as many times as needed. Remember every line you comment out is one less VC running.

3. Re-boot the system for your changes to take effect.

4. Daemons

Many Linux distributions run daemons most of us will never use. Most of these daemons are loaded by scripts. Where these scripts are and what they are called depends on your Linux distribution. Slackware set-up scripts are buried in `/etc/rc.d/rc.*`.

Before you proceed, a knowledge of Unix shell script programming would be a definite asset. However, if you have no experience writing Unix shell scripts, what follows is probably the quickest introduction to shell script programming ever written.

Take the following shell script:

```
#!/bin/sh echo "hello world"
   #echo "good bye cruel world"
```

The previous code will echo the string "hello world". Shell scripts must contain the the line

```
"#!/bin/sh"
```

at the very top line. After that every line is executed as if you had typed it at the keyboard (think of shell scripts as nothing more then glorified keyboard macros).

Lines which begin with a '#' are said to be commented out because they do not get executed by the shell. Most start-up scripts when they load daemons look like:

```
if somecondition
          do something
      fi
```

What you want to do is comment out every line starting with the

```
if
```

statement and ending with the

```
fi
```

statement.

If you want to find where a daemon is loaded, search the start-up scripts for the name of the daemon. If I wanted to find where inetd is loaded in Slackware I would do the following:

```
$ cd /edt/rc/d $grep -n inetd rd.*
```

4.1. inetd

inetd allows people to do things like **telnet**, **ftp**, and send **talk** requests to your machine. If you never use your system as a server or need to access any of its services remotely you can remove inetd.

4.2. lpd

lpd is used to print files on your printer using the **lpr** command. If you never print on your Linux box you can remove lpd. If, however, you own a HP Deskjet ™ printer and would like to print, I highly recommend the package I put together called dj-printcap which is available at:

ftp://sunsite.unc.edu/pub/Linux/system/Printing/dj-printcap.tar.gz
(ftp://ftp.redhat.com/pub/redhat/redhat-4.2/i386/RedHat/RPMS/dhcpcd-0.6-2.i386.rpm)

4.3. nfsd and mountd

These two daemons are used to run an NFS server. If you never use your Linux system as an NFS server you can safely remove these two daemons.

4.4. portmap

The portmap daemon is used to handle RPC services. If you do not run an NFS server or any other RPC programs you can remove portmap.

4.5. sendmail

sendmail is another daemon which requires a fair bit of memory. If you never use your Linux box as a relay for sending e-mail or you never receive mail on your Linux box, you can probably remove sendmail. If you do send e-mail from your Linux box most e-mail clients can be set-up to send e-mail from another mail server.

4.6. others

There may be other daemons your system starts up which you do not need. Remove what you feel you have to. Two daemons which you must run are syslogd and klogd.

5. Conclusions

The previous discussion illustrates the steps I took to tighten up my memory usage on my Linux box. Hopefully I have provided you with some insight into what you can do with your Linux box to conserve memory. Good luck and happy hacking!