TimeSys Linux Install HOWTO

Trevor Harmon

<trevor@vocaro.com>

2005−04−05

Revision History

Revision 1.0	2005−04−05	Revised by: TH
first official release		

This document is a quick−start guide for installing TimeSys Linux on a typical desktop workstation.

Table of Contents

1. Introduction

1.1. Background

TimeSys Linux is a derivative of Linux created by TimeSys Corporation. It includes a fully preemptible kernel, a constant−time scheduler, fully schedulable interrupt handlers, fully schedulable soft−IRQs, and reduced interrupt disable times. These features reduce latency to a point where TimeSys Linux can be used as a Real−Time Operating System (RTOS).

TimeSys Linux comes in four flavors:

- *TimeSys Linux/GPL* The basic TimeSys Linux kernel; offers full preemption at the kernel level, prioritized interrupt handlers, and so on.
- *TimeSys Linux/Real−time* Makes Linux a true RTOS by adding priority inheritance and a POSIX−based high−resolution timer API.
- *TimeSys Linux/CPU* Adds support for CPU reservation, which gives a thread, process, or process group exclusive use of the CPU.
- *TimeSys Linux/Net* Adds support for network bandwidth reservation, guaranteeing that your thread or process will get the bandwidth it requires, regardless of network activity in other processes.

This document deals with the first variant, GPL, as it is the only one available for free. The GPL version is unsupported by TimeSys Corporation (unless you purchase a support contract, of course), and thus the documentation for it is a bit lacking.

To be fair, the documentation for installing TimeSys Linux onto an embedded board and cross−compiling code for it is fairly good. The problem is that TimeSys charges for the toolchains necessary for cross−compiling, and even then, many first−time users may not wish to start compiling for a target board right away. They may just want to try out TimeSys Linux, or they may not even have an embedded board to begin with. TimeSys Corporation's documentation does not help these users.

In this mini−HOWTO, I try to rectify this situation somewhat by explaining how to install TimeSys Linux/GPL onto a typical "beige box" desktop workstation. Although this is clearly not the intended platform for TimeSys Linux, installing the kernel onto a plain−vanilla desktop is an easy way to get started quickly and play around with some real−time applications, such as the RTSJ Reference Implementation.

1.2. Copyright and License

1.3. Disclaimer

No liability for the contents of this document can be accepted. Use the concepts, examples and information at your own risk. There may be errors and inaccuracies that could be damaging to your system. Proceed with caution, and although this is highly unlikely, the author does not take any responsibility.

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.

1.4. Feedback

Feedback is welcome for this document. Send your additions, comments, and criticisms to <`trevor@vocaro.com`>.

2. Requirements

For this HOWTO, I assume that you have the following:

- Red Hat Linux 9 [1]
- TimeSys Linux 4.1 Build 155 for the generic Pentium [2]
- Pentium−class computer capable of running Red Hat Linux 9

3. Install the packages

Your first task is to install Red Hat Linux 9 on the desktop computer that will be running TimeSys Linux. If you already have a computer with RHL9 on it (or even just a boot partition that has it), you can skip this step, but a clean installation is always best. Be sure to choose the *Workstation* installation option so that all of the development packages get installed.

Next, make sure that the `kernel-source` package is installed. You can do this by typing:

```
# rpm -q kernel-source
```

If it's not installed, retrieve it by whatever means you prefer, such as downloading the binary RPM from Red Hat's FTP site, then install it with:

```
# rpm -Uvh kernel-source-2.4.20-8.i386.rpm
```

Copy the TimeSys Linux distribution archive to the computer running RHL9 and extract it. Switch to the `TimeSys-x86bsp` directory that was created and run **./install** as root. This will put all of the TimeSys Linux files into `/opt/timesys`.

4. Prepare the source directories

Before compiling the TimeSys Linux kernel, you need to set up the source directories so that the kernel knows about all the drivers and things on your system. To do so, issue the following commands:

```
# cd /usr/src
        # mv linux linux.old   # Only if the linux directory already exists
        # mv linux-2.4 linux-2.4.old
        # ln -s /opt/timesys/linux/4.1/x86bsp/src/2.4.21-timesys-4.1/kernel linux
        # ln -s /opt/timesys/linux/4.1/x86bsp/src/2.4.21-timesys-4.1/kernel linux-2.4
```

Note that in the above commands, you may need to change *x86bsp* and the version numbers as needed for the TimeSys Linux package that you downloaded.

5. Configure the kernel

You are now ready to compile the TimeSys Linux kernel. Begin by executing these commands:

```
# cd linux
        # make menuconfig
```

Note that you cannot skip the above step and edit the `.config` file yourself. Running the configuration utility and saving your changes generates files that are necessary for compiling the kernel.

If you had performed a clean install of Red Hat Linux 9, then most likely you don't need to change any of the configuration options that appear. However, if you have any strange hardware or just want to perform a "sanity check", you should walk through the menus and enable whatever drivers and options you need. For instance, you may need to add Ethernet drivers for whatever network card you have. If you are installing to a laptop, don't forget to enable *General Setup* --> *PCMCIA/CardBus Support* before going to the *Network device support* page so that you can see the PCMCIA Ethernet drivers. If you have a USB keyboard or mouse, be sure USB devices are enabled. You should also make sure that *Block Devices* --> *Loopback device support* is enabled. And it wouldn't hurt to check other likely prerequisites, such as *Networking options* --> *IP: DHCP support*.

When you are satisfied with the configuration, exit and save.

6. Compile the kernel

The next step is to compile TimeSys Linux:

```
# make dep
        # make bzImage [3]
        # make modules
        # make modules_install
        # make install
```

The **make install** should have put the kernel into `/boot` and even added a *TimeSys* entry into GRUB for you. If you want the new kernel to boot by default, edit `/etc/grub.conf` and change the *default* line to the appropriate zero−based index corresponding to the TimeSys kernel entry. (This will probably be 0).

7. Prepare for rebooting

At this point, the kernel is prepped and ready, but if you reboot now, your system won't come back up due to device file system errors. The problem is that TimeSys Linux depends on the *devfs* file system.

To solve this problem, install devfsd from Richard Gooch's site. Extract the tar file, then copy it to `/usr/src/redhat/SOURCES/`. Switch to the directory where the file extracted to, then run:

```
# rpmbuild -ba rpm.spec [4]
```

This should build the *devfs* package and place it in `/usr/src/redhat/RPMS/i386/`. You can then install this RPM by typing:

```
# rpm -Uvh /usr/src/redhat/RPMS/i386/devfsd-1.3.25-1.i386.rpm
```

Normally, you would now need to add the line **/sbin/devfsd /dev** into `/etc/rc.d/rc.sysinit`, but Red Hat Linux 9 should already have done this for you during installation.

8. Rebooting

Congratulations! You're done! You can now reboot into the TimeSys Linux/GPL kernel.

9. Further Information

Here are some web sites related to TimeSys Linux that you may find useful:

- TimeSys Linux/GPL on SourceForge an independent project that hosts TimeSys Linux/GPL
- Review of TimeSys Linux/RT by Linux Journal a brief look at the RT version (not GPL) of TimeSys Linux

Notes

[1] Other Linux distributions are compatible with TimeSys Linux, but I find that RHL9 is the most TimeSys−friendly.

[2] Other versions should work, of course, but this is the one I tested. Note, too, that TimeSys may occasionally update its kernel with security fixes, so obtaining the most recent build is recommended.

[3] If you get errors about not finding **i586−linux−gcc**, edit the `Makefile` and comment out the line about CROSS_COMPILE.

[4] If your system can't find **rpmbuild**, you may need to install the `rpm−build` package first.