LINUX AND MP3 FOR ARCHIVING

Tom Hallewell
AJ Janitschek
Radio Free Asia
2025 M. Street NW
Washington DC 20036

ABSTRACT

When every on-air second counts, archiving your programming can be formidable if you choose the wrong system. The combined use of a Linux server and MP3 audio files burned to CD-R provides an inexpensive, reliable and long-term method of archiving and documenting audio programs.

Radio Free Asia[1] broadcasts 34 hours of programming per day in 9 languages over four transmission channels. While we're unaware of any legal requirement to archive a copy of every program that goes to air, the broadcasting business dictates prudence. Although your station is probably not required to keep complete copies of all programs, it surely can't hurt, especially if you can do it cheaply. In this litigious age, nearly every broadcaster will be brought to task for something said on-air. An MP3 archive is a nice way to cover yourself in the event of a lawsuit or even a simple enquiry from one of your everyday listeners: "Didn't you just say you were giving away $1-million to the first caller?"

At Radio Free Asia, we used to archive all of our audio as WAV files onto 4-Gigabyte DAT tapes. It was not only expensive ($8 per DAT x 7 tapes per week x 4 weeks per month = $56 versus $12.60 for our current CD-R system), but it was also terribly labor intensive. The programs were compressed, so you had to restore a segment of tape before you could listen to it. Plus, the DAT tapes (around 28 per month) took up a great deal of space, so we had a lot of capital tied up into buying and storing hundreds of tapes. And it was too inconvenient and time-consuming for anyone to ever really want to hunt down and listen to anyway! And, believe it or not, this method was far more convenient, compact and economical than archiving on standard audiotape. With our current system, we can store 100 hours of archive programming on a single CD-R and make it available to anyone who has an MP3 player on their PC.

LINUX AND MP3

What is this stuff?

To start, you need to understand a bit about Linux and MP3 audio files.

Linux[2] is a UNIX-like operating system that was designed to provide PC users a free or very low-cost operating system comparable to UNIX systems. Linux has a great reputation as a very efficient and fast-performing system that easily works on older 386 PCs (of course, a screaming P-III with a 1.2 GHz processor is cool too). Unlike Windows and other proprietary systems, the source code for Linux is publicly open and continuously grows in strength and capabilities due to the contributions of Linux users and

[1] Radio Free Asia: www.techweb.rfa.org and www.rfa.org

[2] The Linux Documentation Project: www.linuxdoc.org

programmers around the world. The Linux kernel (the central part of the operating system) was developed by Linus Torvalds at the University of Helsinki in Finland. Linux is named after him. Simply put, load Linux on a server with several Gigs of hard drive space and you have the basic beginnings of an audio archive system that will make your Station Manager and your budget happy.

Linux is a perfect platform for several reasons. First, it is the only *completely free* server-class operating system in the world. By server-class, we mean you can serve Internet applications such as web hosting, e-mail and a variety of other services, which cost you a bundle on any other platform. An astonishing number of applications and utilities have been written for use under Linux and are also available around the 'Net' for free. Many of these, such as Sendmail[3] and Apache[4], are industry standards even though they come to the consumer at no cost.

Linux is extremely malleable. You can write powerful scripts in anything from C to Perl[5] to simple but powerful "shell scripts". At present, Radio Free Asia has a marked preference for a language called Python[6], but we're non-denominational; use what works for you. We're mostly interested in the results, not the process of writing code, so use whatever gets the job done the fastest so you can get your archiving started!

The MP3 audio format (MPEG-1 Layer-3, also known as MPEG-3 or MP3)[7] should need no introduction. It has gained notoriety on the Internet as the music piracy medium of choice but can also be used to legally distribute and archive audio. MP3 is a standard audio technology and format for compressing a sound sequence into a very small file (about one-twelfth the size of the original file) while preserving the original sound quality when it is played back. In a way, the term "compression" is misleading. Nothing is actually compressed in this process so much as thrown away. It's a set of tradeoffs that the person doing the job decides best suit his or her quality and delivery needs. One very convenient characteristic of MP3 is that, unlike most other formats, the compression rate is highly variable, permitting you to select an optimum balance between audio quality and file size. The greater the compression, the smaller the file, but the smaller the file, the worse the file sounds. One of the first things you'll want to do is to determine how low a bit rate you can live with. Encode your audio to MP3 using different sample rates and determine the lowest acceptable bit rate. You can gain a lot of bang for the buck if you decide you can live with mono rather than stereo. With mono, you can basically halve the amount of space it would take to archive the same program in stereo.

There are some other basics about MP3 audio that need mentioning. A standard audio CD is formatted as a 44.1 kHz, 16-bit stereo WAV file. An average hour of audio takes about 600 Megabytes of space on a CD. The average CD-R holds 650 Meg (you're right, 700 Meg CD-R's are available too).

To satisfy the many needs of our company, Radio Free Asia encodes three sets of archives on three different servers and retains them for a limited time:

1. "Short-term high quality" is recorded at 48 kbps/32 kHz, 16 bit mono. We consider this to be our broadcast quality

[3] Sendmail: www.sendmail.org
[4] Apache Web Server: www.apache.org
[5] www.perl.org
[6] Python Scripting Language: www.python.org
[7] General MP3 Technical Reference Page: showcase.netins.net/web/phdss/mp3/mp3_faqs.html

archive. Since Radio Free Asia is an 'all short-wave' international broadcaster, 44.1 kHz or 48 kHz wouldn't serve us any better. 60-minutes of audio equals approximately 20 Megabytes. The MP3 files reside on the server for 14-21 days until an automated script deletes them for space conservation.

2. "Long-term archive of lesser quality" is recorded at 16 kbps/16 kHz, 16-bit mono. 60-minutes of audio equals 7-Megabytes or approximately 1% the size of a standard, high quality WAV on commercially available audio CD's. The MP3 files reside on the server for up to 45 days before they are copied to CD-R then manually deleted from the server. This remains a manual process to help safeguard the files when a CD-R burn is unsuccessful.

3. "Audio for web" is our lowest quality archive and is recorded at 14 kbps/12 kHz, 16-bit mono. 60-minutes of audio equals 5.9-Megabytes. We keep the audio quality low in order to keep the entire file size low. They are run through a signal processor to take some highs out and add a bit of compression to keep the levels more consistent. This lower bit rate translates into quicker download times for our listeners overseas, especially for those in countries with very limited bandwidth. The MP3 files remain on the server for approximately 60 days before they are deleted so the web server space can be used for newer files.

MONEY TALKS

How much does it cost to make CD-R's every month? First let's review some basic information again about our monthly broadcasts:

- Radio Free Asia broadcasts 34 hours of programming daily; that's an average of 1020 hours of audio to archive every month.

Now that we know how much material we have to archive, let's look at the costs for CD-R media:

- Bulk CD-R's, each in a jewel case, cost around $1 each.

- Bulk CD-R's, stacked on spindles, cost around $.60 each.

- Clamshell cases, when bought in lots of at least 1000, cost around $.24 each.

Radio Free Asia usually uses 15 CD-R's to archive a full month's worth of programming. We could combine some of our language service programming together on one CD-R, but we choose to leave the different languages separate. For example, the monthly total of 60-hours of Burmese audio stays on one CD-R while we put 60-hours of Laotian audio on it's own separate CD-R.

Radio Free Asia dropped the use of CD-R jewel cases for the use of clamshell cases. Two factors weighed heavily in this move the extra cost of CD-R's in a jewel case versus a CD-R sold on a spindle, and the extra space a jewel case takes versus a clamshell case. We literally cut our used storage space by 75% when we switched to the clamshell cases. This means that we've greatly extended the life of our existing archive storage space. We also saved manpower by eliminating the need for jewel case inserts that we printed on a standard inkjet printer, then cut and inserted into the cases by hand; it was a tedious job that everyone was happy to see go away at last. Clamshell cases come in a variety of colors

too. This could be helpful to you when classifying your archives.

The monthly cost of material is extremely small compared to our previous archiving system that incorporated data DATs. It's important to note here that since the cost of CD-R media is low, don't skimp on spending a few extra cents to get quality media. Our vendor is always asking our opinion of the media we bought. They even took back what remained of a 500 CD-R shipment when the media had an unusually high failure rate. We received another full shipment of quality, replacement CD-R stock. You can bet we have continued to shop with that vendor. Here are our monthly budgetary numbers:

- Average of 15 CD-R's per month to archive on complete set.

- Each CD-R costs $.60. Available in spindles of 50 or 100.

15 x .60 = $9 per month for CD-R stock

- Use 15 CD clamshell cases per month.

- Each clamshell case costs $.24 when bought in bulk of 1000 or more.

15 x .24 = $3.60 per mo. for clamshell cases

Total per month: $9 + $.3.60 = **$12.60**

ABOUT INSTALLATION

Hardware requirements

Although Linux has a low enough overhead to run on a 386, you will need a little bit of muscle if you want to do anything else with the PC. Ripping the archive CD-R's is particularly intensive. We currently use a dual 200 MHz Pentium PC with 128 MB of RAM and it just does the job. Keep in mind, though, that we have up to 4 simultaneous streams writing at once. You may only have one stream to archive, in which case a 300 MHz single P2 should be more than up to the task.

You will want to have Serial Support compiled into the kernel and install Python, a super-powerful, yet easy to learn scripting language.

There are several services you will have to have running on your Linux machine before you can get serious about archiving.

- **Serial Support**[8] must be compiled into the Linux kernel or your serial ports won't work. This is the default installation on most commercial Linux versions.

- **Python**-Radio Free Asia's code is written in Python, an easy-to-learn, yet very powerful scripting language. You will need to have Python installed on your Linux machine in order for our scripts to work. It is installed by default in Red Hat, otherwise your mileage may vary.

- **Time-synch**: you have to have atomic time. Install Network Time Protocol (NTP)[9] and connect to a reliable time-server.

- **Web Server**-If you intend to have a web interface, you'll need to install and configure a web server such as Apache or Zope[10].

[8] Linux Serial Port How-To: www.linuxdoc.org/HOWTO/Serial-HOWTO.html
[9] NTP-Network Time Protocol: www.eecis.udel.edu/~ntp/
[10] Zope: www.zope.org

- **Security** is a serious consideration! We've been bitten many times through naiveté, poor planning and worse configuration. Secure your machine. You'll never regret the effort!

You'll need a good-sized, reliable hard drive, or better yet, a RAID array to store your archive on. Decide how many days you want to archive for and multiply that times the file size times the number of files per day, then add at least 25 percent for fat. You always take up more space than you anticipate!

Some sort of MP3 encoder[11] is, of course, central to creating an MP3 archive. There are a number of different hardware encoders available from various vendors. Select one that has input that is compatible with your audio output. Some units have analog inputs while others use digital; the best ones allow both. Choose an encoder that offers you plenty of different bit and sample rate permutations. Many encoders come equipped with a built-in Network Interface Card (NIC), so you can plug them straight into your LAN or WAN if you want to network the unit for remote access. Others have a serial output, or both NIC and serial output. Also take the footprint into consideration; do you prefer a rack mount unit or a desktop unit? Processors are getting fast enough now that you might even want to research real-time software encoding using an MP3 encoder on a PC running Linux. Unfortunately, this fascinating project is outside the scope of this article.

Depending on the type of MP3 Encoder you selected, you will now need to make some decisions about how to get your audio into the Linux machine. If you have a NIC interface and a reliable network, you might want to consider using http to stream the audio from the encoder across your LAN. This may cause a lot of unwanted traffic on your network, but if, like most broadcasters, you only need to archive one stream of audio, then an extra 10-20 megabytes of data an hour probably won't make that much difference. However, we would generally recommend that you go with serial input. This is a no-brainer if you only have one feed to archive. Just plug the cable straight into your PC's serial port. If you have more than one audio source, then you will probably have to install a multi-serial port. Cyclades, Rocket and Stallion all manufacture multi-serial ports with Linux drivers. They can be tricky so watch your pin-outs!

You will also need a CD-burning station somewhere. At RFA, we burn the archive CD's right on the archive server, which saves a lot of needless shuffling of files across the network. We recommend using a SCSI CD recorder instead of IDE. Also, there is a very reliable free CD burning software package for Linux called X-CD Roast[12] with an intuitive graphic interface. We highly recommend it.

You can compress your files up to 25 percent more if you put some kind of processor to condition your output. Boost the low mids, notch out some of the high mids; 1-5 kHz is particularly ugly!! We use the TC Electronics[13] Finalizers, but even a 5 band graphic EQ or compressor/limiter or both would be a big help.

[11] LAME Project-Open Source MP3 Encoder: www.sulaco.org/mp3

[12] X-CD-Roast CD Burning Package: www.xcdroast.org

[13] TC Electronics: www.tcelectronics.com

Software

Once your PC is running Linux and you think you have your serial ports working and audio data going to them from the MP3 encoder, it's time to start testing your ports. A really neat feature of Unix-like operating systems is that they treat devices just like files. This may not sound that neat to you, but the implication is that if you have data going into a serial port, all you have to do is pipe that data into a file. You could say something like "read the output of the serial port /dev/ttySxx and put it into file xMP3", which translates to:

```
cat /dev/ttySxx > xMP3
```

in Linux. This command should get you some audio into a file. If not, you need to work on your serial ports.

Assuming you have succeeded in getting the encoder to write to a file, all you need to do now is get a script. We're going to cheat and use the Python code that Radio Free Asia developed and made available at **www.techweb.rfa.org/mp3archiving** but feel free to modify the code to suit your needs. One of the beautiful things about open source code is that you can tweak it any way you need.

- You can create a schedule for jobs on your server with a Unix utility called "/usr/bin/crontab". *Crontab* is a Linux command that sets up a series of automated tasks. Crontab is used to create a table or list of commands, each of which is executed by Linux at a specified time. One easy application to program with Crontab is an automated system for archiving on-air programs that checks for new recordings every 30-minutes.
- Crontab can be set to give a time, port and duration. And call "aaMain" (Fig. 1)

```
# Cronfile
# min hr day mon day-of-week   aaMain     service  line
duration-in-minutes

0 10 * * *  /usr/local/bin/aaMain   VIE    3      57
```

Fig 1-sample crontab file

This is Crontab's way of saying that at 0 minutes of the 10th hour every day of every week, run the shell script aaMain, passing the arguments VIE (program name), 3 (serial line in), 57 (duration). As you can see from this example, understanding and editing a Crontab is so easy, with a little coaching even a non-Unix person can work with it on a regular basis.

- "aaMain" passes the Crontab information , and calls the big script. (Fig. 2)

- aaSerialIn opens a file at the time specified by Crontab, listens at a "port" and writes that data to the file for the duration set in Crontab, then closes the file and cleans up the disk cache. (Fig. 3 on next page)

```
#!/bin/bash

if [ -f /usr/local/sbin/aaSerialIn.py ]; then
nice -n -20 /usr/local/sbin/aaSerialIn.py $*  &
fi
sleep 5
```

Fig. 2-sample aaMain script

- aaSerialIn opens a file at the time specified by Crontab, listens at a "port" and writes that data to the file for the duration set in Crontab, then closes the file and cleans up the disk cache. (Fig. 3 on next page)

Remember you have to delete your files at some point too, because you can't write to a full disk! We have an automated script for this. You want to keep at least 5 percent of your disk free. You can run another cronjob to expire the archives as you write them.

Process

You need to come up with a procedure to make it all stick. Someone has to burn and label the CD's, although if you're "only" doing 24-hours a day and one language, this could easily be automated using Crontab.

Radio Free Asia's Master Control staff checks every audio file for their shift at the end of their day. To reduce network traffic, each MP3 is copied to a local drive on a PC so we do not decode or read the files over our network. Then the MP3 is opened and evaluated for look for any waveform inconsistencies. A supervisor oversees every CD-R burn and verifies that each CD-R is 100% accurate. Beyond that, the supervisor verifies that all programs are included on a CD-R. Normal delivery of the monthly archive CD's is 5-10 business days after the end of the month. Once the files are successfully copied to CD-R, the files are deleted from the archive server.

Troubleshooting Corrupt Files

The world is not perfect. Sometimes we make mistakes or have problems with our MP3 recordings. Here is a list of some typical errors. At first, we had some regular occurrences of these, but with some sweat and some great brain-power, recording errors have practically vanished.

- Huffman Errors. MPEG encoder unlock is one way to cause Huffman errors due to digital clipping. Once a file encounters the MPEG unlock, the entire file is destroyed. In other words, as you listen to the file you hear a "Darth Vader" sort of chirping throughout the rest of the file. The only 'fix' we know of is to re-record the file, however our IT staff is constantly investigating other causes so the Master Control staff provides detailed error messages every time an error occurs.

- Header Errors. Bitrate errors occur when the MP3 encoders record a file with bad header information. In this case, the MP3 files playback at a faster rate than normal sounding like you hired a bunch of squirrels instead of humans. Another clue to the header or bitrate

errors is that the MP3 player shows bit-rate and sample frequencies that do not match the original encoder settings. This is usually fixed within seconds by editing the header information on the individual MP3 file.

```
#!/usr/bin/python

import select,signal,sys,time

AlrmHandled="AlrmHandled"
def alrmhandler(arg1,arg2):
    raise AlrmHandled,arg1
    return()

def runmain(fargs):
    signal.signal(signal.SIGALRM,alrmhandler)
    LANGUAGE = fargs[1]
    ARCHIVE = "/home/httpd/html/archive/"
    TIME_STAMP=time.strftime("%Y-%m%d-\
    %H%M",time.localtime(time.time()))

    FILE = LANGUAGE + '-' + TIME_STAMP + '.mp3'
    DISK_OUT = ARCHIVE + LANGUAGE + '/' + FILE
    diskout=open(DISK_OUT,'w')
    inport=int(fargs[2])-1          # Offset for serial port
    SERIAL_IN="/dev/ttyR" + `inport` # Specify serial port
    serialin=open(SERIAL_IN,'r')    # Open serial port

    delay=( 60*int(fargs[4]) ) + 5  # Kill after 60 minutes
    signal.alarm(delay)      # Set alarm to wake up and die
            while 1:
            try:
                    rdy = select.select([serialin],[],[])[0]
                    if serialin in rdy:
                            data=serialin.read(4096)
                            diskout.write(data)

            except AlrmHandled:
                    serialin.close()
                    diskout.close()
                    time.sleep(20)
                    sys.exit()

if __name__ == '__main__':
        runmain(sys.argv)
```

Fig. 3-Sample aaSerialIn.py Python script

- Dead-Air. Radio Free Asia insists that all archives accurately reflect the program as it aired. Editing any programming to produce a more suitable archive is taboo. Sections of dead air are left intact whether it's due to mechanical or human error, just as long as it represents what the listeners heard.

- Missing Programs. Although rarely, this does occur from time to time. You will most likely see this come in spurts due to network maintenance, bad routings or the disc on your server being full. Again the only fix at the present time is to re-dub the file.

Web archive

Radio Free Asia's system is also a great stepping stone into web archiving, as it is a simple matter to write a script to generate HTML code based on the MP3 files contained in an archive director. Radio Free Asia has made copyright-free sample scripts to this and other broadcast utilities available at no cost on our technical web site at **www.techweb.rfa.org** .

All you need is to read the contents of your archive directory and generate a file called index.html which writes the actual contents of the directory into a template you create. Run it at the time you start an archive file and one minute after so that you can catch any files that have been opened or closed. A sample script is available at Radio Free Asia's 'techweb' site mentioned a the bottom of the previous paragraph.

And the last word

You can combine Linux and an MP3 encoder system into a simple automated system that makes audio archiving a snap. (Fig. 4)

Archives are usually not meant for re-broadcast but for documentation. Keep the sampling/bit rates low and the size of your archive becomes more manageable. CD-R's take up very little space. CD-R's are inexpensive, especially when bought in bulk. Many options for storage exist beyond the standard CD case or 'crystal,' like clamshell cases or custom CD racks and cases. With the added benefit that a CD-R will not deteriorate like a tape, archiving your programs as MP3 files to CD-R is a perfect method to carry your network or station well into the next few decades. It also provides you with great flexibility in a changing world. You can easily change formats from MP3 to tomorrow’s dominant encoding standard, or from CD-R to DVD-R, simply by swapping out a single piece of hardware.

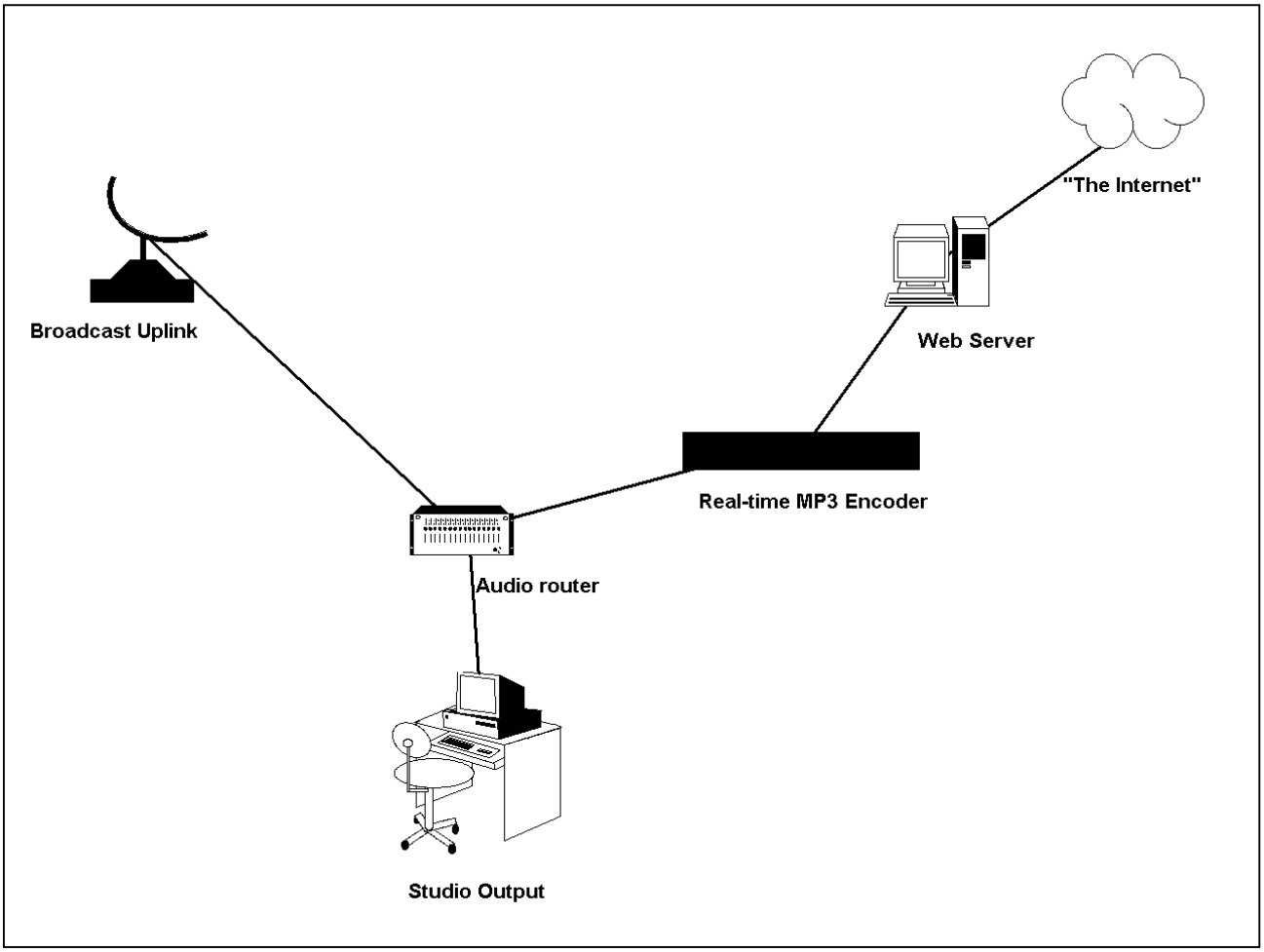

Fig. 4-Basic view of the archiving process