OzTiVo — Toys, Tools, Hacks

Warren Toomey, Dennis Boylan, Michael Edwards, Keith Wilkinson

OzTiVo

http://minnie.tuhs.org/TiVo

ABSTRACT

The TiVo disk-based video recorder is not only one of the most innovative of consumer products, but also one of the most hackable consumer products. Designed for the U.S market, TiVo Inc. has never supported the TiVo in Australia. This is the story of how several hackers in Australia and elsewhere reverse-engineered the TiVo's operation, how they modified it for Australian conditions, and how they have been able to make the device friendly to non-hackers.

1. Introduction

What is it that makes the TiVo disk-based video recorder so innovative? From a user's point of view, the TiVo radically changes your TV viewing pattern. You simply select a list of programs to record by name, and let the box record the programs whenever they come on. There is no fussing with TV guides, getting times wrong, changing cassettes or running out of tape. And when you get home at night, you simply look through the list of recordings and watch whatever you feel like. Most TiVo owners watch less TV (they skip ads), but they get more "quality" TV time.

At the same time, the TiVo would have to be a hacker's dream box. The hardware provides a separate MPEG encoder and decoder, so that programs can be recorded and played back simultaneously[1]. Hard disks can be upgraded with industry-standard ATA drives to increase recording capacity. The software is based on a Linux 2.1 PowerPC kernel, with a set of user-mode processes that control recordings, playback, and the management of a multimedia database (TV guide, actor names, program categories etc.). To top it off, the TiVo even provides a nice serial port that allows you to connect to a Bash shell over PPP.

In fact, one of the dangers of the TiVo is the "dog house syndrome": once a hacker breaks their TiVo, their partners and children will send them to the dog house until the TiVo is working again. There have been threats of divorces, but so far none have been realised.

However, the TiVo as it comes from the manufacturer is almost useless for Australian conditions. The U.S model only works for the NTSC broadcast system, and although the U.K model does support PAL, it cannot decode the FM stereo modulation used on Australian analogue TV transmissions. If this wasn't enough, the lack of any TV guide information for Australia quickly turns the TiVo into a disk-based VCR: you get to fuss with start/stop recording times, and curse when your program gets moved to a different timeslot or runs overtime.

All of these deficiencies have been overcome by some quite Herculean efforts by a bunch of devoted hackers in Australia and elsewhere. Some of our modifications have been hardware based, some in software. We have constructed a support infrastructure to let our TiVos "phone home" and collect TV guide data, and we have also built a users' support infrastructure to allow us humans to help each other out. None of this would have been possible without Open Source tools, and the willingness to make our home-grown TiVo hacking tools Open Source.

2. The Pioneering Hacks

The early TiVo hacks for Australian conditions were done by Andrew Tridgell, Paul Mackerras, Bob Edwards and others in Canberra. The first order of the day was to get PAL video into the box. This required research on the video and MPEG chips used in the TiVo and how to reprogram them. Many of the data sheets were hard to obtain or impossible to obtain without signing non-disclosure agreements. The solution here was to perform random register operations on the chips until the TiVo logs indicated a successful video lock; fortunately this technique was scriptable.

At this point the box was able to record PAL composite video input, but not display it. The same technique of fiddling register values will eventually set the box to PAL output, but how do you script the operation to stop when the TV is finally showing a picture? The lateral solution devised by tridge was to feed the TiVo's video

1. Yes, even the same program. Start watching your favourite movie at 9pm and miss the ads.

output back into its input: when the video output is working, the input will have a proper signal to record and the logs will indicate a signal lock.

With the register programming worked out, the TiVo software had to be modified to go into "PAL" mode at boot time. Initially this was a binary patch to the kernel module *fpga7114.o*, which worked but was very fiddly. The second version of the patch used the much more useful technique of installing a bypass on function calls inside binary kernel modules; fortunately, Paul Mackerras was the PowerPC Linux kernel maintainer at the time. A completely new kernel module, *Palmod*, was created to do this.

The TiVo was now usable in a sense, in that it could record from a PAL composite video source like a VCR or cable set-top box. However, there was still no TV guide data, no easy way to control the channels on the external box and the built-in tuner did not work.

Fixing the tuner required the NTSC tuner to be removed from the TiVo and be replaced by a PAL tuner. These are not easy to find in small quantities, and initially the PAL tuners were scavenged from PC video tuner cards. Bob Edwards did most of the hardware hacking here. The register programming for the new tuner was added to Palmod, which also caught the NTSC tuner channel change commands and converted them into the correct PAL tuner channel change commands.

Another useful hardware hack was the design of the "TiVonet" card. The TiVo designers had helpfully left an edge connector on the motherboard for possible future expansion, but the signals on the connector were not documented. The Canberra hackers were able to reverse-engineer the pinouts of the edge connector, and design a converter board which would allow an ISA bus NE2000 network card to be installed. Finally, users could telnet to their TiVos instead of using the 9,600 bps serial port.

3. Palmod Support for Other Tuners & Countries

As the number of Australian TiVo users increased, so did the demand for more functionality from the TiVo. The original Palmod module only supported a single PAL tuner type from Philips, and it didn't cover all of the Australian channels. Nor did it take advantage of the many features that the TiVo hardware had to offer. The original Palmod, however, did provide a wonderful platform to build on.

The 2.5.1 Palmod provided support for all Australian channels and thus made the TiVo available to users in all states. Picture quality into the TiVo was improved by enabling the S-Video input, and the ability to specify command line parameters was added. Initially this last feature only enabled the customisation of the picture width, brightness, contrast and colour but there was more to come.

Palmod weaves its magic in two ways: firstly by manipulating register values of the hardware components and secondly by replacing some of the original function calls in various modules with its own code. This second method relies on knowing the offsets to the function calls in each module, which was not a problem as they were defined in the modules' symbol table and easily found.

When writing version 3.0 of the proprietary TiVo software, the TiVo programmers must have decided that they had made things a little too easy for the hackers in the past. One of the most notable changes in version 3.0 of the TiVo software was that most of the offsets in the symbol tables were gone, and along with it was the working Palmod. Fortunately this was only a short-term setback and enough offsets were found to restore Palmod to a going concern although, even to this day, the full original functionality has never been achieved.

At around this time, TiVo users were in search of an alternate replacement tuner. Eventually a Samsung unit was successfully installed and a new, separate Palmod was born. Another development was the addition of a second audio input, which allowed the connection of two set-top boxes (one via S-Video and one via composite video) to the TiVo. Palmod was again modified make it all work.

Of course, a PAL TiVo isn't only useful in Australia. There are lots of countries that use PAL, and TiVos can be found in many of them. As a result, more flavours of Palmod were created to suit local requirements. Even in Australia there were two versions in use, supporting either the Philips or Samsung tuner. Keeping the separate modules in synch was problematic, so a unified Palmod was created along with a configuration utility for managing the growing list of command line parameters.

Although nearing the end of its development for the Series 1 TiVos, Palmod may one day be born again for Series 2. Much work will need to be done on TiVo Series 2, as PAL is not supported natively and the documentation for the new Broadcom hardware is not readily available. And of course there is also the soon to be released High Definition TiVo.

4. Getting TV Guide Data

Without a TV guide, the TiVo is next to useless. TiVo Inc. doesn't support TiVos in Australia, of course, so there is no official way of obtaining TV guide data.

TiVo Inc. distributes data from their "mothership" out to TiVos phoning home in what is known as 'slice' format: this is effectively a binary-encoded set of data records with efficient representation of integers and text. Initially, U.S users were able to capture real TiVo slices and begin the process of reverse-engineering their format, including the schema for the data records. The earliest and still the most common slice encoding and decoding tools are Andrew Tridgell's *writeguide* and *readguide*.

Once the syntax of the slice files had been worked out, the next step was to determine the semantics of the files: what records were required, what was valid and what was invalid, and the relationship between records within a single slice.

Chris Yeoh and Andrew Tridgell were the first to produce a tool to create TV guide slices for the TiVo: their *guidekit*. This was written in Perl using a MySQL database backend, and used the *writeguide* tool to do the slice encoding. The *guidekit* obtained weekly program data from the Sofcom and Foxtel websites.

Two drawbacks of *guidekit* were its documentation and license conditions[2], both summarised in the README file, which is given in full below:

> TiVo guidekit
>
> DO NOT DISTRIBUTE THIS KIT TO ANYONE.
>
> basic instructions:

While waiting the few weeks it took to obtain a copy of *guidekit*, Warren Toomey wrote his own TV guide fetcher and converter in Perl. Initially this stopgap tool fetched a single channel's worth of data at a time and was given the uninspiring title of *fetch_one*. After two major rewrites it is now called *Wktivoguide*, and it is now providing the majority of Australian TiVo users with their weekly TV guide data.

Both *guidekit* and *Wktivoguide* provide guide data in slice format. The variety of information that can be placed in each slice is quite immense:

- Station-id, start time and duration for each program.
- Program's title, description of the program, if it is episodic, title of the episode, identity of the series that this program belongs to, episode number in that series.
- Genre of the program (Drama, Movie, Documentary etc.), TV rating (G, PG, MA etc.), movie star rating.
- List of actors in the program, director of the program, flags for captioning, repeat, letterbox, violence, offensive language, sexual themes, nudity etc.

In the U.S and the U.K, TiVo users subscribe to TiVo Inc's service and they pay to be able to download this wealth of program information each week. In Australia, such a rich source of guide information is not available, and we need to use some clever techniques to infer some of the missing information.

Wktivoguide starts the process by downloading what program data it can from several web sources. The first difficulty is to parse this data. To begin with, only the start time of each program is given: the end time must be inferred from the start time of the next program. Each "day" of data from the web sources actually starts at 6am and goes to (around) 6am the next morning, so two "days" of information need to be scanned to obtain data for one real day.

One significant annoyance is the great variation of episode title data and actor data. Some web sources provide the episode title data separate from the program's title, but most keep both the episode's title and the list of actors in the program's description. We apply several heuristics to separate this data. Similarly, certain phrases are trimmed from the beginning of a program's title (e.g. "Special:", "Movie:", "Sport:") in order to get to the real program title.

One important piece of information is a program's genre. The TiVo tracks program genres and uses these to record programs on your behalf. For example, if you watch enough documentaries, the TiVo will begin to record other documentaries for you to watch[3]. Unfortunately, many web sources do not provide this information. To overcome this problem, *Wktivoguide* keeps its own program database which includes the genre of the program. This database can be updated on-line by the OzTiVo user community, and this allows the weekly program slices to have accurate and up to date genre information.

2. The license conditions have subsequently been relaxed.

3. How aggressive the TiVo is with recording extra programs can be controlled via the 'Thumbs Up' and 'Thumbs Down' buttons on the remote control.

With the raw web data fetched and parsed, *Wktivoguide* begins the process of augmenting this with the data from the program database and other databases. We use the movie ratings and directors list from the IMDB movie database to add star ratings and directors names to the data.

Once the raw data is augmented, the weekly program slices can be generated. At present, 45 different program slices are made each week. This is necessary as each TiVo will only fetch data based on its location and the input sources it uses. The matrix of Australian states versus {Antenna, Cable, Satellite, Digital, plus combinations of these} is large. Fortunately it takes about a minute to generate each program slice. Once generated, the program slices are moved into a download area where they can be retrieved by the Australian TiVos phoning home.

5. Emulating the TiVo Mothership

With the TiVo hacked for Australian broadcast standards, it was now a usable device. However, it still was a hackers-only gizmo, as owners had to telnet in and perform some weekly care and feeding: updating the clock, manually downloading new TV guide data, even creating new channel lineups by hand and editing the database to set channel numbers. To make the TiVo a consumer-friendly device, we needed a way to emulate the service that TiVo Inc. provides to their real customers.

Fortunately, the real TiVo service is done using HTTP, with TiVos making a daily PPP-over-modem call to "phone home" and collect new configuration and TV guide data. Emulating this service just required a large amount of reverse-engineering.

Nick Giannis and Dan Boardman of the *tivo_canada* Yahoo group created separate versions of the TiVo Service Emulator in April 2003. These were the first publicly available versions, but were limited in their functionality. From these humble beginnings, the *n4zmz-emu* came into existence. It took the best of both worlds and extended their functionality to its current state. The original versions were named *tivo-service-emulator*, which were supported by Nick and Dan. So as to not conflict with the existing names, the choice of using *n4zmz* (an amateur callsign) made the naming unique.

As to where Nick and Dan got their information to create their versions, it could have been from looking at the files on their TiVos (`/tvlib/tcl/tv/{*}.itcl`) or doing a packet capture of the conversations between their TiVos and the real TiVo mothership. Both of these methods are currently used to diagnose and extend the service emulator.

The TiVo mothership needs to perform a number of tasks.

- Capture service logs (mlog.cgi)
- Download phone information (TCD411.cgi)
- Download software information (HServer.cgi)
- Download lineup information (HServer.cgi)
- Download guide information (HServer.cgi)
- Download showcase[4] information (HServer.cgi)
- Download keys (keyserver.cgi)
- Capture usage information (acceptfile.cgi)

These tasks are performed by capturing information from the TiVo and sending back commands to the TiVo to perform. This would include downloading a file, setting internal database configurations, or executing a script. The files sent to the TiVo are protected by signatures if it is an executable, or SHA1 checksums if it is a data file. The TiVo mothership protects its files by requiring extra information that a proxy server would not normally pass in the HTTP headers.

The service emulator is controlled by a master configuration file (*tivo.conf*). This file contains the locations of all of the directories it needs and any specific parameters for an individual TiVo. The configuration is merged with the information from the TiVo to provide a flexible system of control. The software version on the TiVo is used to determine the correct format of the commands to send.

The task of capturing the service logs is used by the service emulator to know the last time a particular TiVo talked to it for guide information selection. It updates the statuslog with this information.

The task of downloading the phone information by the service emulator is based upon information received from the TiVo and the configuration in the master configuration file. The TiVo provides the current database information about what it uses to talk to the mothership. The service emulator takes this information and verifies it against the files in the headend[5] directory, along with the specific information in the master configuration file to

4. A showcase is a collection of menu entries, video and audio clips which are used as promotional material like commercials, movie previews or movie trailers. They usually correspond to future or current shows/movies.

determine what, if anything needs to be sent to the TiVo. This might be to send it new phone numbers to call, or a new toll free number to call, or disable toll free calling.

The task of downloading software information, like the phone information, is based upon information received from the TiVo and the configuration in the master configuration file. The TiVo provides the current database information about its software versions and the reason for the call. The service emulator takes this information and verifies it against the files in the headend directory and the master configuration file to decide what it must tell the TiVo to do.

The task of downloading lineup information, like all of the previous tasks, is controlled by the information provided by the TiVo, along with the master configuration file. The TiVo provides a number of location ids that it wants information on. The service emulator must take this information along with the type of unit (U.K/U.S), the signal type (Antenna etc.), and the last successful download time to determine which files the TiVo might need. In a perfect world, the latter pieces of information would not be needed. They are needed to get around bugs in the client software on the TiVo and to normalize the U.K values to the U.S model.

The task of downloading guide information is done by the TiVo providing a list of headends that it wants information for. The service emulator takes this list and searches the TV guide directory for matching names. The service emulator uses the file names to determine what days the file covers: it will not send a file to the TiVo if that date has passed, even if the modification date of the file is after the last time the TiVo called.

The task of downloading keys at the moment is a static function. The service emulator checks the keyring of the TiVo and sends the statically defined ElGamal keys to properly populate the keyring. This is the area that would need to be enhanced in order to get HMO[6] fully functioning on the series 2 TiVos.

5. A headend is a file that holds configuration information for a TiVo. It includes the TiVo's location, postcode, the list of stations in that area and how they can be received (Antenna, Cable, Satellite). When a TiVo is first switched on, the user walks through a number of on-screen menus which allows the TiVo to determine the correct headend for it.
6. HMO is a set of new functionality in the Series 2 TiVos that allows recordings to be shared between TiVos on a LAN.

By providing all of the above functionality, the service emulator lets a TiVo perform these three functions:

- Guided Setup
- Test Call
- Daily Call

The Guided Setup occurs when a TiVo is first switched on. It consists of three parts. The first part determines the correct phone numbers to use (and is effectively a null operation here). The second part is the downloading of the current affiliation, IR database, logos and headends. The third part is the downloading of the guide data for the selected lineup. The Test Call is just a special case of the first phase of Guided Setup. It is used to verify that there is good network connectivity between the TiVo and the server.

The Daily Call checks the lineup for updates along with any new TV guide information. Along with the guide information, the Daily Call to the emulator sets a number of internal parameters on the TiVo (upload, keyserver, time server, service state).

6. Building a Turnkey TiVo System for Australia

Before the TiVo service emulator arrived, Australian TiVo users had import their TiVos from the U.S or U.K, and hand-hack the configuration to install the Palmod module, configure the channels and the video sources etc. While hacker types mostly enjoy this low-level configuration, it excluded many potential TiVo users who just wanted to record TV. After all, that's what a TiVo was built for!

Early on, when TiVo hacking here had just begun, several people released backup images of the TiVo software pre-configured for Australia (Andrew Palm, Warren Toomey, Andrew Tridgell). Users still had to restore the backup to a hard disk, requiring them to boot a Linux distro. The "Hinsdale HowTo" and a well designed Linux Boot CD helped to reduce the learning curve here.

With the pre-configured TiVo software installed, users still had to build their own headend and configure the channels & TiVo station-ids to suit their conditions. Even with appropriate HowTo documentation, most non-hacker types were put off by such obscure instructions as:

> *Load the backup image onto your TiVo harddrive, and then add some channels using "mkchannels"*
>
> *e.g. Bash # cd /var/hack/guide*

Bash # ./mkchannels source/setup oz_fta.txt City

Usage is mkchannels source/setup ChannelFile City

e.g. Bash # ./mkchannels 636950/18 oz_fta.txt MelbDig

will add the Melbourne Digital channels for you if your source is 636950/18

At the same time, several networking options were becoming available to TiVo users: the internal modem, PPP over the serial port, the TivoNet, TurboNet and AirNet network cards. What users wanted was a software image which they could load, which would bring up some form of network, which could "phone home" to the mothership emulator, and which would allow them to configure their TiVo the 'normal' way (i.e. via a TV screen, not a bash prompt).

Michael Edwards took up this challenge, and began with a virgin 3.0 TiVo disk image and began to modify it to load the Palmod, configure the appropriate network device, connect to the Australian service emulator (and not the real one!), and correctly set certain internal parameters after the Guided Setup had completed.

The task of constructing an image which would properly negotiate all of these steps was difficult, and was made even harder because there is no boot console: the serial port, for example, only becomes available once there is a user-mode process listening on the port.

One quirk of the U.S TiVo models is that they refuse to perform a Guided Setup if they are running in PAL mode. Of course, if the device is not in PAL mode, the user cannot see the on-screen menus to perform the Guided Setup, resulting in a Catch-22 situation. Michael was able to find a "not-exactly PAL" mode where the TiVo produce PAL output with occasional stutters, and where the TiVo is happy to perform the Guided Setup. This is now known as "Maintenance Mode".

The image produced for Australian conditions has also been enhanced with the drivers for all the TiVo network devices, and it contains a grab-bag of software which has been written to extend the functionality of the basic TiVo.

7. Open Source and Home-grown Tools

Given that the TiVo (as it arrives from the factory) has a Linux kernel, a serial port and a working shell, it was only a matter of time before users began to hack on the system. The TiVo is, however, an embedded system running proprietary applications, and initially there was a steep learning curve before new features and functions could be added by the user.

In terms of programming languages, the TiVo provides shell scripting and Tcl scripting. There is no C compiler, but fortunately this has been rectified with a GCC cross-compiler for the Linux x86 platform. Many of the early hacks were to add critical tools such as "strings", "less", text editors etc. Once networking functionality (PPP over the serial port, or Ethernet) was added, services such as cut-down Telnet and FTP daemons were also created.

One of the hardest areas in the TiVo to hack has been the Multimedia File System (MFS). This is TiVo's proprietary filesystem, which provides both database functionality (to hold all details of the system state: the TV guide, programs to record, season passes etc.) and the efficient storage of audio/video streams. There are two approaches to using the MFS. One is to leverage from the existing Tcl scripts and libraries on the TiVo that access the MFS; the second is to reverse-engineer the MFS structure and to write C libraries to access it. Both approaches have been used successfully.

The Tcl approach to the MFS is best exemplified by the Tivoweb tool. This is a set of Tcl scripts and modules that run a web service on a user's TiVo. Using a web client, the user can browse the TV guide data, choose programs to record, delete recordings, browse & edit the MFS database and many other functions. Tivoweb is derived from an early TCL web server written by Stephen Rothwell, and is now maintained by dozens of hackers worldwide.

The C approach to the MFS is exemplified by the numerous programs written to extract recordings from the TiVo and to convert them into standard MPEG2 video streams. Nearly all of these tools (TyTool, TyStudio, MFS_FTP, Playitsam) have their origins in the "vplay" program written by tridge after reverse-engineering portions of the MFS structure. The video storage side of TiVo's MFS is now well-known, and recent tools allow recordings from one TiVo to be copied into a second TiVo.

Some TiVo hacks have even been done to add back functionality that was removed by TiVo. For example, all the libraries that deal with timezones were removed from the TiVo, and the main TiVo application was hard-coded to only have timezones for the U.S and the U.K. This has lead to several time-related issues in Australia. The solution has been to run the TiVo with no timezone offset (i.e. to set the clock to local time), and to "port back" the timezone libraries,

database files and applications. We now have a client that listens continuously for NTP broadcasts, and a Time client (TCP port 37) which can be run from the command line or from cron to update the local clock and to follow daylight saving changes.

8. OzTiVo — A Community of Users and Hackers

In the days of the pioneering hacks, most of the Australian TiVo owners knew each other and were skilled at hacking on an unknown box. Once the box had been opened up enough, the requisite skill level dropped so that an experienced Unix hacker (or someone willing to learn quickly) could configure a TiVo for Australian conditions.

This new plateau of usability brought many new Australian TiVo owners, and the old communication channels (private e-mail/face-to-face meetings) were not adequate to support them. In July 2002, Warren Toomey set up the OzTiVo mailing list to provide an avenue for new TiVo owners to ask questions and to obtain answers.

The OzTiVo mailing list has grown from about 20 subscribers to nearly 400 subscribers, with about 4 or 5 new subscribers per week. While the mailing list is heavily used and provides casual interaction between TiVo users, one of the main drawbacks that we have found is that the mailing list does not provide a "memory": the same questions are asked week after week by new subscribers.

To alleviate the "repetitive FAQ syndrome", a Wiki[7] for the OzTiVo community was set up by Andrew Palm. This holds answers to most of the frequently asked questions, several HowTo categories, and links to other TiVo resources on the Web. As new questions or new tips appear on the mailing list, subscribers are encouraged to add the information to the OzTiVo Wiki. The Wiki also provides access to the *Wktivoguide* program database so that it can be kept up to date.

While the Wiki was introduced to solve some of the mailing list problems, it has brought some problems of its own. The number of FAQs and HowTos has grown tremendously to the point where newcomers are overwhelmed by the amount of information available. At some point, the Wiki will need to be 'refactored' to have fewer documents that are better structured, and which guide the newcomer from one point to another. This has been done to some extent (e.g. the guide to setting up a TiVo in Australia), but more work here is required.

9. Conclusion

Hacking the TiVo wasn't easy, despite it being based around a Linux kernel. It has taken the combination of several devoted hackers, way too much time, and significant levels of devotion to make the TiVo suitable for Australian conditions. We have had to reverse-engineer much of the TiVo's operation, design hardware, write tools & programs to overcome its limitations, and construct support programs that emulate as much as possible the services provided to TiVo users in the U.S. and the U.K. Along the way, we have managed to build a community structure to support ourselves, and we have reduced the hacking skills required so that new TiVo adopters can mostly avoid a vicious and steep learning curve and go straight to what the TiVo was designed for: recording TV programs.

7. Wiki: a collaborative system where many authors can write, update and browse a collection of documents.