VPNFilter EXIF to C2 mechanism analysed

securelist.com/vpnfilter-exif-to-c2-mechanism-analysed/85721/

Authors

GReAT

On May 23 2018, our colleagues from Cisco Talos published their excellent analysis of VPNFilter, an IoT / router malware which exhibits some worrying characteristics.

Some of the things which stand out about VPNFilter are:

- It has a redundant, multi-stage command and control mechanism which uses three different channels to receive information
- It has a multi-stage architecture, in which some of the more complex functionality runs only in the memory of the infected devices
- It contains a destructive payload which is capable of rendering the infected devices unbootable
- It uses a broken (or incorrect) RC4 implementation which has been observed before with the BlackEnergy malware

- Stage 2 command and control can be executed over TOR, meaning it will be hard to notice for someone checking the network traffic

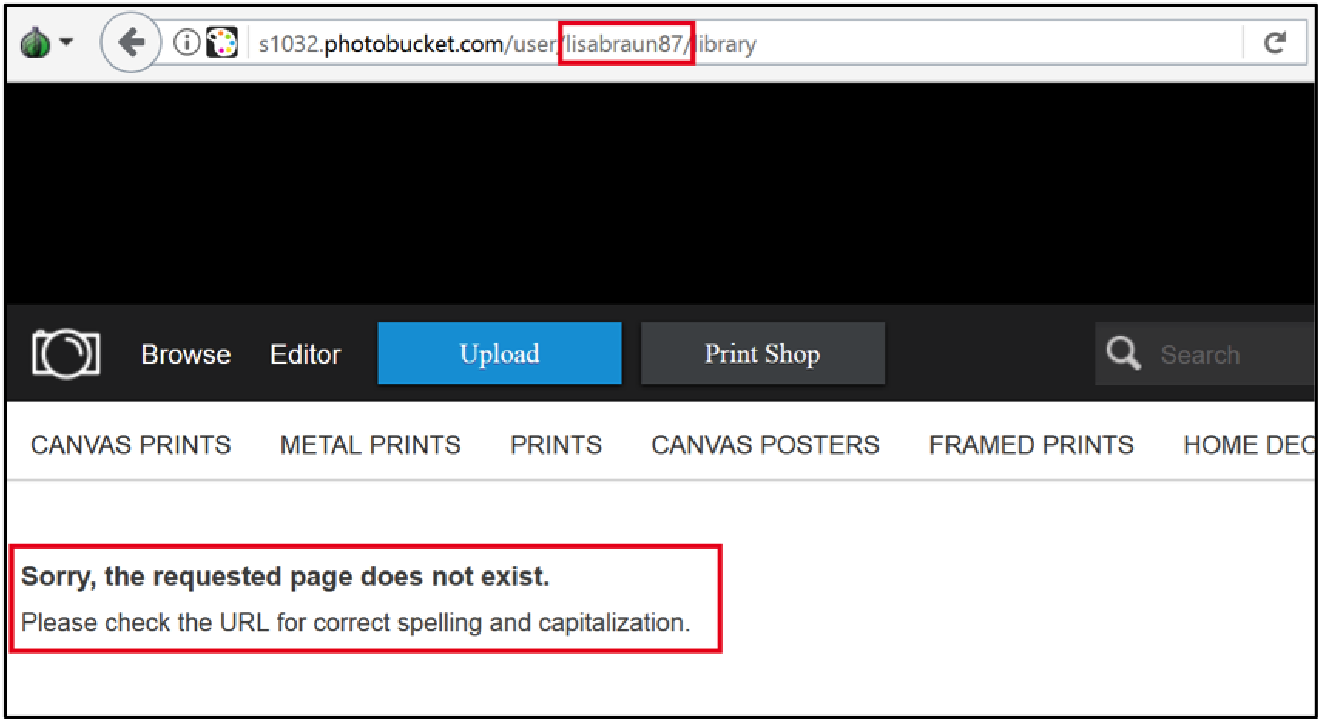
We’ve decided to look a bit into the C&C mechanism for the persistent malware payload. As described in the Talos blog, this mechanism has several stages:

- First, the malware tries to visit a number of gallery pages hosted on photobucket[.]com and fetches the first image from the page.
- If this fails, the malware tries fetching an image file from a hardcoded domain, toknowall[.]com. This C2 domain is currently sinkholed by the FBI.
- If that fails as well, the malware goes into a passive backdoor mode, in which it processes network traffic on the infected device waiting for the attacker’s commands.

For the first two scenarios in which the malware successfully receives an image file, a C2 extraction subroutine is called which converts the image EXIF coordinates into an IPv4 address. This is used as an easy way to avoid using DNS lookups to reach the C&C. Of course, in case this fails, the malware will indeed lookup the hardcoded domain (toknowall[.]com). It may be worth pointing that in the past, the BlackEnergy APT devs have shown a preference for using IP addresses for C&C instead of hardcoded domain names, which can be easily sinkholed.

To analyse the EXIF processing mechanism, we looked into the sample **5f358afee76f2a74b1a3443c6012b27b**, mentioned in the Talos blog. The sample is an i386 ELF binary and is about 280KB in size.

Unfortunately for researchers, it appears that the photobucket.com galleries used by the malware have been deleted, so the malware cannot use the first C2 mechanism anymore. For instance:



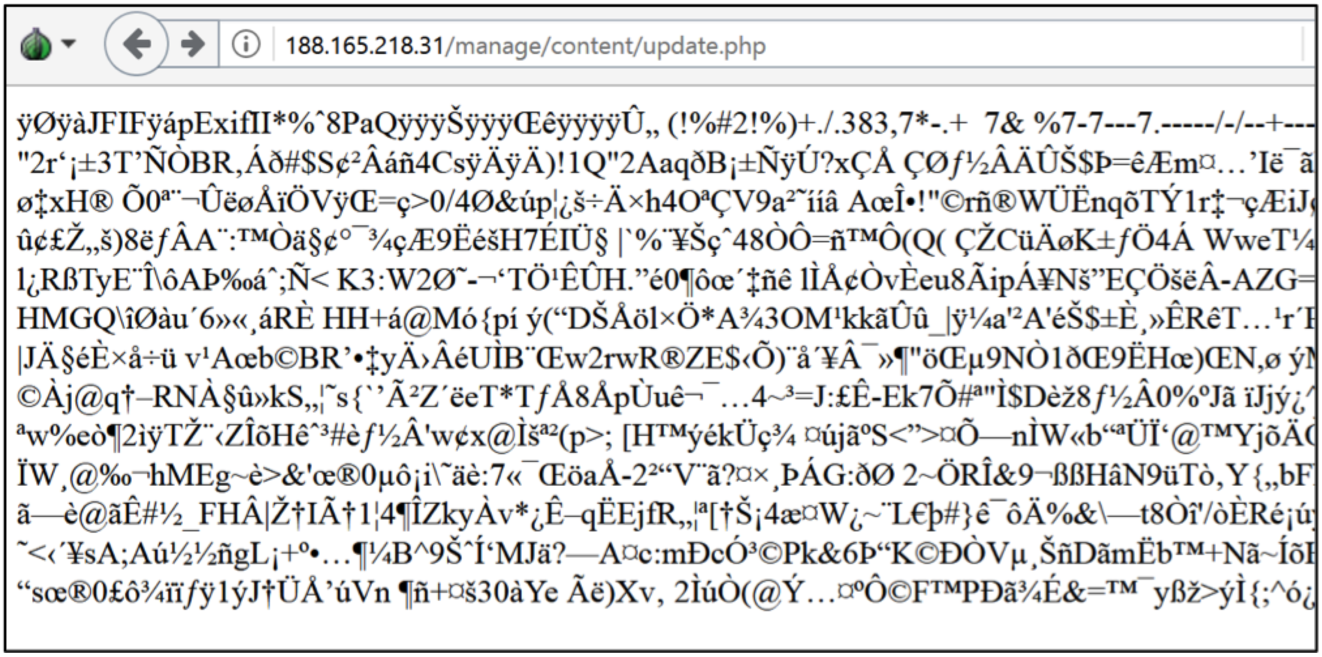
Sorry, the requested page does not exist.

Please check the URL for correct spelling and capitalization.

With these galleries unavailable, the malware tries to reach the hardcoded domain toknowall[.]com.
While looking at the pDNS history for this domain, we noticed that it resolved to an IP addresses in France, at OVH, between Jan and Feb 2018:

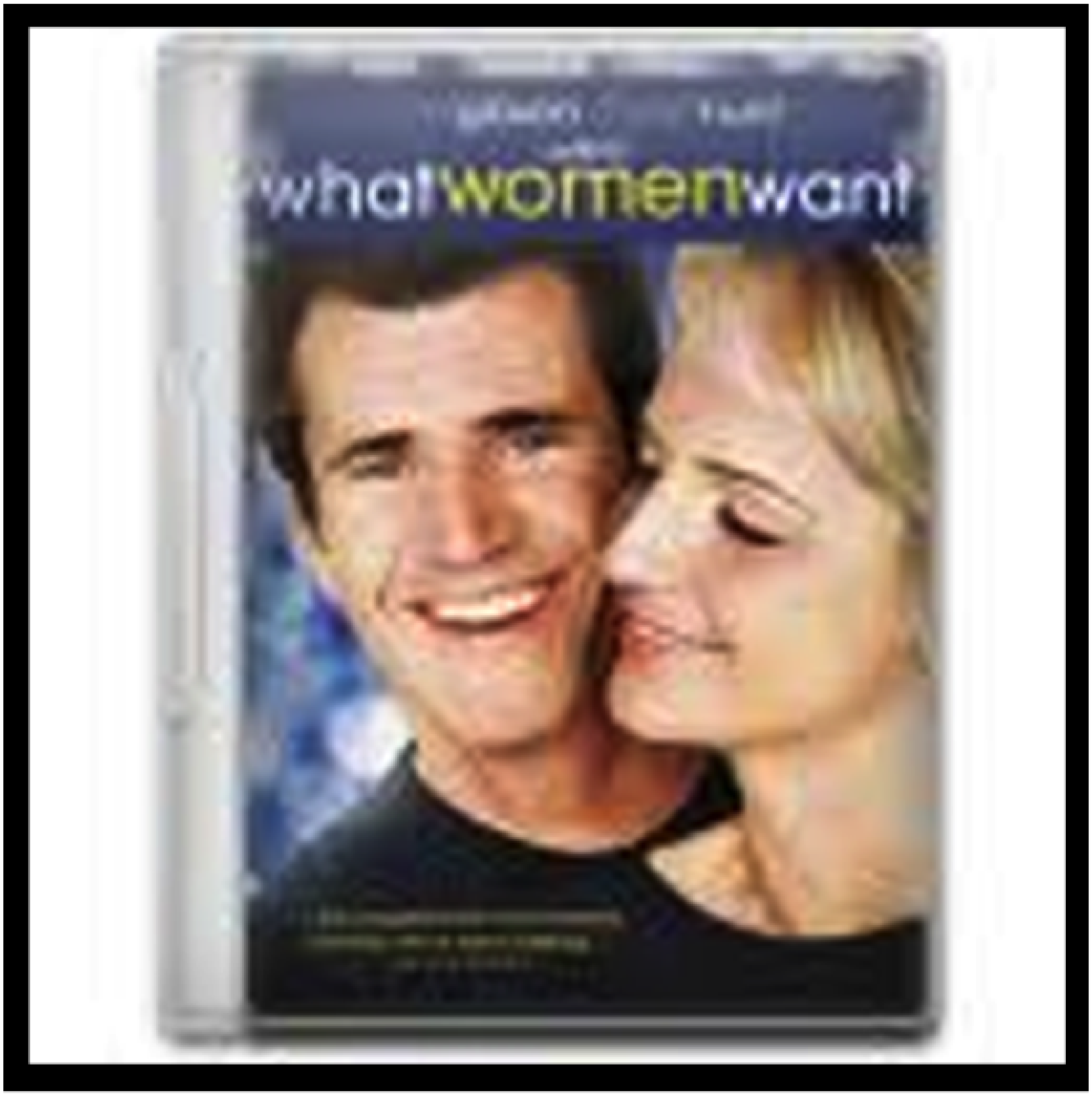
toknowall.com	188.165.218.31	FR	**2018-01-23 11:23:14**	**2018-02-06 05:50:32**

Interestingly, when visiting this website’s C2 URL, we are presented with a JPG image, suggesting it is still an active C2:



```
ÿØÿàJFIFÿápExifII*%ˆ8PaQÿÿÿŠÿÿÿŒêÿÿÿÿÛ„ (!%#2!%)+./.383,7*-.+  7& %7-7---7.-----/-/--+---
"2r‘¡±3T’ÑÒBR‚Áð#$S¢²Ââñ4CsÿÄÿÄ)!1Q"2AaqðB¡±ÑÿÚ?xÇÅ ÇØƒ½ÂÄÛŠ$Þ=êÆm¤…’Iё¯ã
ø‡xH® Õ0ª¨¬ÛëøÅïÖVÿŒ=ç>0/4Ø&úp¦¿š÷Ä×h4OªÇV9a²˜ííâ AœÎ•!"©rñ®WÜËnqõTÝ1r‡¬çÆiJ
û¢£Ž„š)8ëƒÂA¨:™Òä§¢°¯¾çÆ9ËéšH7ÉÏÜ§ |`%¨¥Šçˆ48ÒÔ=ñ™Ô(Q( ÇŽCüÄøK±ƒÖ4Á WweT¼
l¿RßTyE¨Î\ôAÞ‰áˆ;Ñ< K3:W2Ø˜-¬‘TÖ¹ÊÛH.”é0¶ôœ´‡ñê lÌÅ¢ÒvÈeu8ÃipÁ¥Nš”EÇÖšëÂ-AZG=
HMGQ\îØàu´6»«¸áRÈ HH+á@Mó{pí ý(“DŠÅöl×Ö*A¾3OM¹kkãÛû_|ÿ¼a'²A'éŠ$±È¸»ÊRêT…¹r´I
|JÄ§éÈ×å÷ü v¹Aœb©BR’•‡yÄ›ÂéUÌB¨Œw2rwR®ZE$‹Õ)¨å´¥Â¯»¶"öŒµ9NÒ1ðŒ9ËHœ)ŒN‚ø ýM
©Àj@q†–RNÀ§û»kS„¦˜s{`’Ã²Z´ëeT*TƒÅ8ÅpÙuê¬¯…4~³=J:£Ê-Ek7Õ#ª"Ì$Dèž8ƒ½Â0%ºJã ïJjý¿
ªw%eò¶2ìÿTŽ¨‹ZÎõHêˆ³#èƒ½Â'w¢x@Ìšª²(p>; [H™ýékÜç¾ ¤újãºS<”>¤Õ—nÌW«b“ªÜÏ‘@™YjõÄ
ÏW¸@‰¬hMEg~è>&'œ®0µô¡i\˜äè:7«¯ŒöaÅ-2²“V¨ã?¤×¸ÞÁG:ðØ 2~ÖRÎ&9¬ßßHâN9üTò‚Y{„bF
ã—è@ãÊ#½_FHÂ|Ž†IÃ†1¦4¶ÎZkyÀv*¿Ê–qËEjfR„¦ª[†Š¡4æ¤W¿~¨L€þ#}ê¯ôÄ%&\—t8Òî'/òÈRé¡ú
˜<‹´¥sA;Aú½½ñgL¡+º•…¶¼B^9Šˆ Í‘MJä?—A¤c:mÐcÓ³©Pk&6Þ“K©ÐÒVµ¸ŠñDãmËb™+Nã~ÍõI
“sœ®0£ô¾ïïƒÿ1ýJ†ÜÅ’úVn ¶ñ+¤š30àYe Ãë)Xv, 2ÌúÒ(@Ý…¤ºÔ©F™PÐã¾É&=™¯yßž>ýÌ{;^ó¿
```

Here’s how it looks when viewed as an image:



When we look into the EXIF data for the picture, for instance using IrfanView, it looks as following:

Filename – update.jpg

- GPS information: –
- GPSLatitude – 97 30 -175 (97.451389)
- GPSLongitude – -118 140 -22 (-115.672778)

How to get the IP out of these? The subroutine which calculates the C2 IP from the Latitude and Longitude can be found at offset 0x08049160 in the sample.

As it turns out, VPNFilter implements an actual EXIF parser to get the required information.

First, it searches for a binary value 0xE1. This makes sense because the EXIF attribute information begins with a tag “0xFF 0xE1”. Then, it verifies that the tag is followed by a string “Exif”. This is the exact data that should appear in a correct header of the Exif tag:

Exif tag
FF E1 Exif tag
xx Length of field
45 78 69 66 00 ‘Exif’
00 Padding

The tag is followed by an additional header:

“Attribute information” header
49 49 (or 4D 4D) Byte order, ‘II’ for little endian (‘MM’ for big endian)
2A 00 Fixed value
xx xx Offset of the first IFD

The data following this header is supposed to be the actual “attribute information” that is organized in so-called IFDs (Image File Directory) that are data records of a specific format. Each IFD consists of the following data:

IFD record
xx xx IFD tag
xx xx Data type
xx xx xx xx Number of data records of the same data type
xx xx xx xx Offset of the actual data, from the beginning of the EXIF

The malware’s parser carefully traverses each record until it finds the one with a tag ’25 88′ (0x8825 little endian). This is the tag value for “GPS Info”. That IFD record is, in turn, a list of tagged IFD records that hold separate values for latitude, longitude, timestamp, speed, etc. In our case, the code is looking for the tags ‘2’ (latitude) and ‘4’ (longitude). The data for latitude and longitude are stored as three values in the “rational” format : two 32-bit values, the first is the enumerator and the second one is the denominator. Each of these three values corresponds to degrees, minutes and seconds, respectively.

Then, for each record of interest, the code extracts the enumerator part and produces a string of three integers (i.e. “97 30 4294967121” and “4294967178 140 4294967274″ that will be displayed by a typical EXIF parser as 1193143 deg 55′ 21.00″, 4296160226 deg 47′ 54.00”). Then, curiously enough, it uses sscanf() to convert these strings back to integers. **This may indicate that the GPS Info parser was taken from a third-party source file and used as-is.** The extracted integers are then used to produce an actual IP address. The pseudocode in C is as follows:

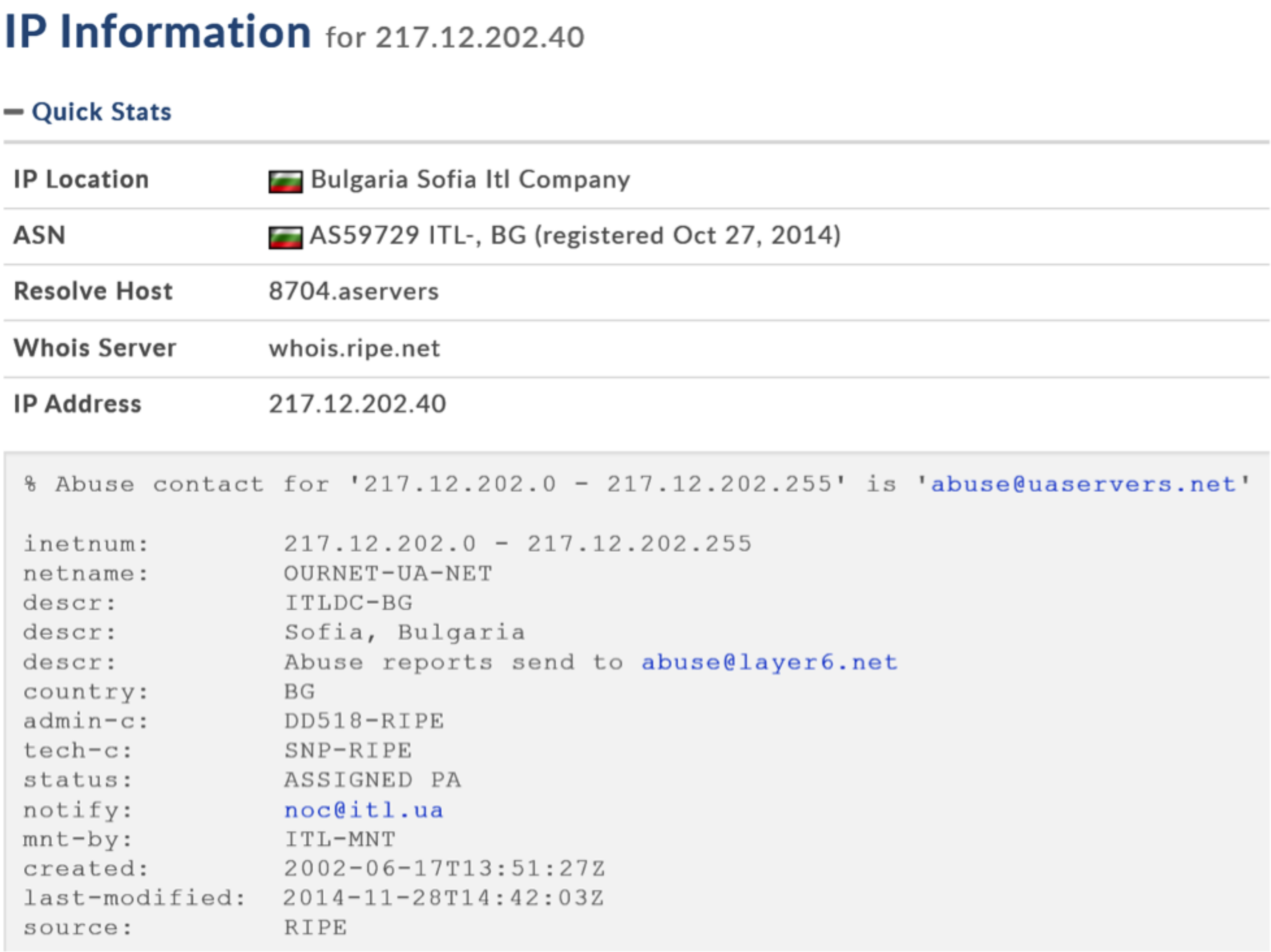
```
const char lat[] = "97 30 4294967121"; // from Exif data
const char lon[] = "4294967178 140 4294967274"; // from Exif data
int o1p1, o1p2, o2p1, o3p1, o3p2, o4p1;
uint8_t octets[4];
```

```
sscanf(lat, "%d %d %d", &o1p2, &o1p1, &o2p1);
sscanf(lon, "%d %d %d", &o3p2, &o3p1, &o4p1);
octets[0] = o1p1 + ( o1p2 + 0x5A );
octets[1] = o2p1 + ( o1p2 + 0x5A );
octets[2] = o3p1 + ( o3p2 + 0xB4 );
octets[3] = o4p1 + ( o3p2 + 0xB4 );
```

```
printf("%u.%u.%u.%u\n", octets[0], octets[1], octets[2], octets[3]);
```

The implementation of the EXIF parser appears to be pretty generic. The fact that it correctly handles the byte order (swapping the data, if required) and traverses all EXIF records skipping them correctly, and that the GPS data is converted to a string and then back to integers most likely indicates that the code was reused from an EXIF-parsing library or toolkit.

For the values provided here, the code will produce the IP address “217.12.202.40” that is a known C&C of VPNFilter.



IP Information for 217.12.202.40

— Quick Stats

IP Location	Bulgaria Sofia Itl Company
ASN	AS59729 ITL-, BG (registered Oct 27, 2014)
Resolve Host	8704.aservers
Whois Server	whois.ripe.net
IP Address	217.12.202.40

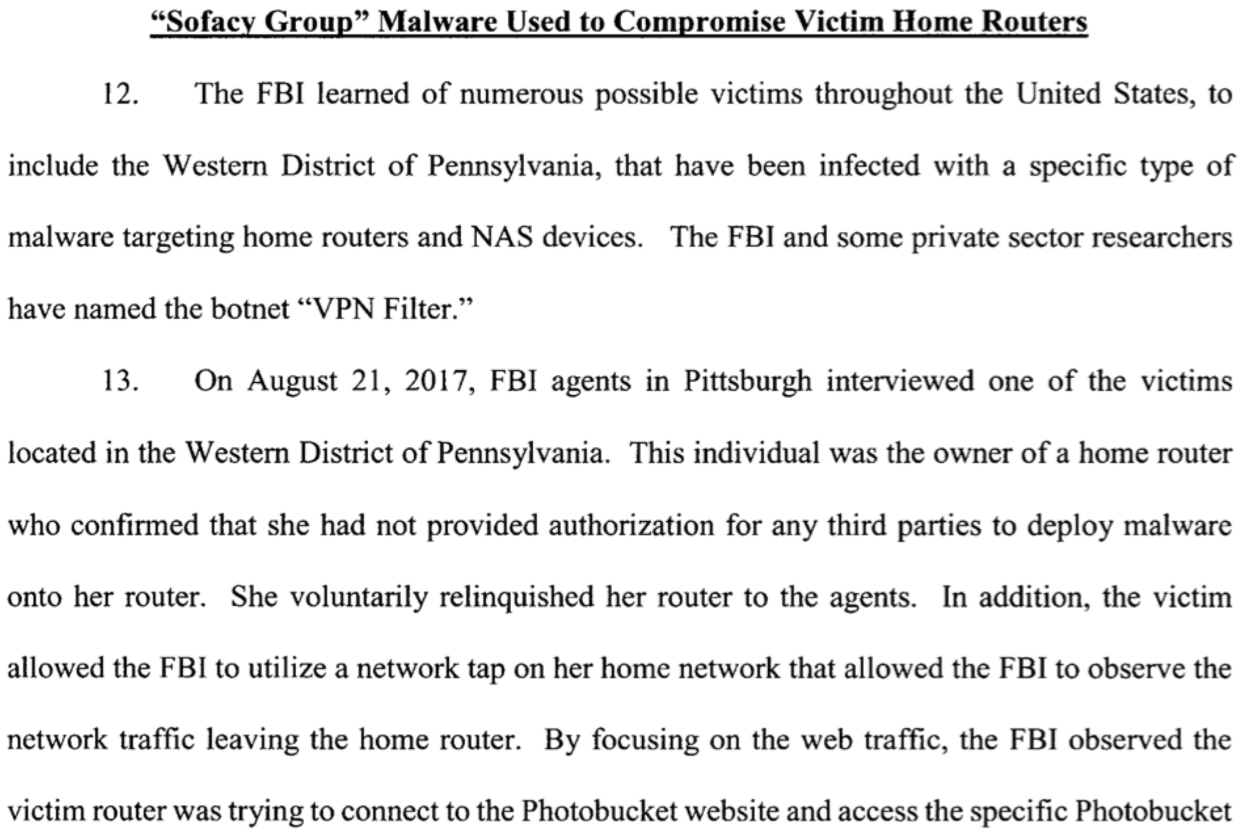
```
% Abuse contact for '217.12.202.0 - 217.12.202.255' is 'abuse@uaservers.net'

inetnum:        217.12.202.0 - 217.12.202.255
netname:        OURNET-UA-NET
descr:          ITLDC-BG
descr:          Sofia, Bulgaria
descr:          Abuse reports send to abuse@layer6.net
country:        BG
admin-c:        DD518-RIPE
tech-c:         SNP-RIPE
status:         ASSIGNED PA
notify:         noc@itl.ua
mnt-by:         ITL-MNT
created:        2002-06-17T13:51:27Z
last-modified:  2014-11-28T14:42:03Z
source:         RIPE
```

It should be noted that this IP is included in Cisco Talos' IOCs list as a known C&C. Currently, it appears to be down.

What's next?

Perhaps the most interesting question is who is behind VPNFilter. In their Affidavit for sinkholing the malware C2, FBI suggests it is related to Sofacy:



"Sofacy Group" Malware Used to Compromise Victim Home Routers

12. The FBI learned of numerous possible victims throughout the United States, to include the Western District of Pennsylvania, that have been infected with a specific type of malware targeting home routers and NAS devices. The FBI and some private sector researchers have named the botnet "VPN Filter."

13. On August 21, 2017, FBI agents in Pittsburgh interviewed one of the victims located in the Western District of Pennsylvania. This individual was the owner of a home router who confirmed that she had not provided authorization for any third parties to deploy malware onto her router. She voluntarily relinquished her router to the agents. In addition, the victim allowed the FBI to utilize a network tap on her home network that allowed the FBI to observe the network traffic leaving the home router. By focusing on the web traffic, the FBI observed the victim router was trying to connect to the Photobucket website and access the specific Photobucket

Interestingly, the same Affidavit contains the following phrase: *"Sofacy Group, also known as apt28,* ***sandworm****, x-agent, pawn storm, fancy bear and sednit"*. This would suggest that Sandworm, also known as BlackEnergy APT, is regarded as subgroup of Sofacy by the FBI. Most threat intel companies have held these groups separate before, although their activity is known to have overlapped in several cases.

Perhaps the most interesting technical detail, which Cisco Talos points in their blog linking VPNFilter to BlackEnergy, is the usage of a flawed RC4 algorithm.The RC4 key scheduling algorithm implementation from these is missing the typical "swap" at the end of the loop. While rare, this mistake or perhaps optimization from BlackEnergy, has been spotted by researchers and described publicly going as far back as 2010. For instance, Joe Stewart's excellent analysis of Blackenergy2 explains this peculiarity.

So, is VPNFilter related to BlackEnergy? If we are to consider only the RC4 key scheduling implementation alone, we can say there is only a low confidence link. However, it should be noted that BlackEnergy is known to have deployed router malware going back as far as 2014, which we described in our blogpost: “BE2 custom plugins, router abuse, and target profiles“. We continue to look for other similarities which could support this theory.

- APT
- BlackEnergy
- Internet of Things
- Malware Descriptions
- Router
- Targeted attacks

Authors

GReAT

VPNFilter EXIF to C2 mechanism analysed

Your email address will not be published. Required fields are marked *