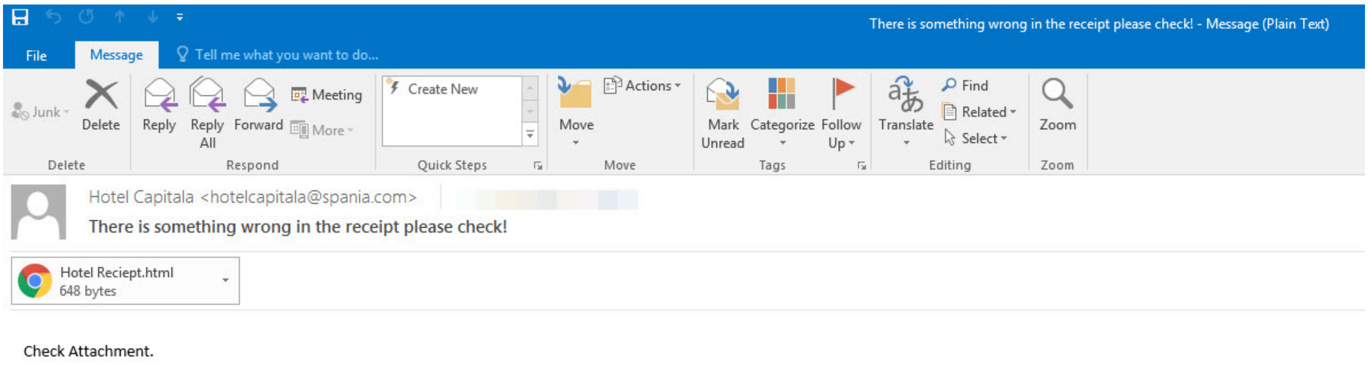
You Can Run, But You Can’t Hide

written by Mert SARICA | 1 September 2021

In the past, there was a threat actor, when the barbers were fleas, and the horses were jesters. This threat actor had sent an email to top-level employees of the institutions he targeted, with an HTML file attached. When this HTML file was opened, and the link address (https://go0gle-drive[.]blogspot[.]com) followed, the targeted person was directed to an address on the mega.nz file storage and sharing site (https://mega[.]nz/file/axlmBSxR). If this file was downloaded and run, the threat actor could remotely control the targeted system, making all kinds of mischief, including recording audio, video and keystrokes. According to legend, some network-based sand pool systems could not analyze the link address contained in this HTML file sent by the threat actor.



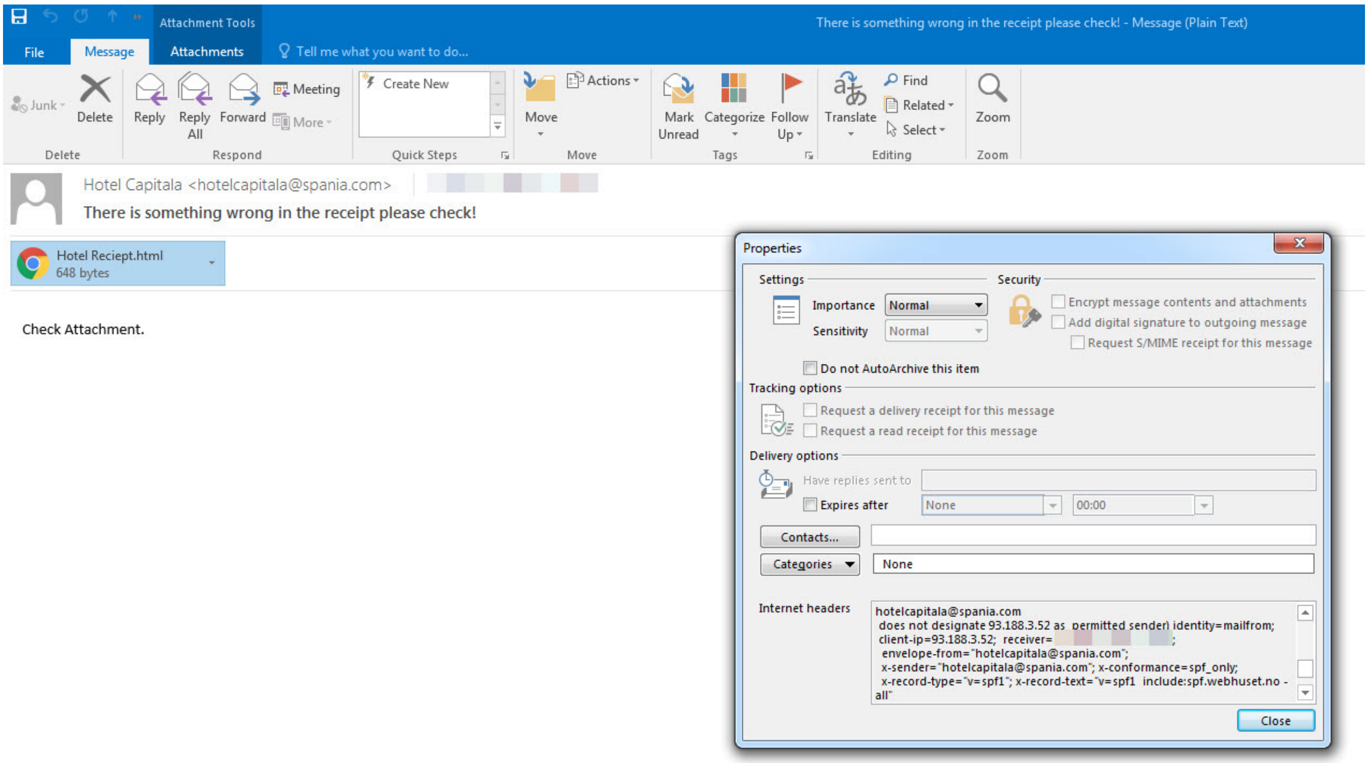
There is something wrong in the receipt please check! - Message (Plain Text)

Hotel Capitala <hotelcapitala@spania.com> [redacted]

There is something wrong in the receipt please check!

Hotel Reciept.html
648 bytes

Check Attachment.



There is something wrong in the receipt please check! - Message (Plain Text)

Hotel Capitala <hotelcapitala@spania.com> [redacted]

There is something wrong in the receipt please check!

Hotel Reciept.html
648 bytes

Check Attachment.

Properties

Settings
Importance: Normal
Sensitivity: Normal

Security
□ Encrypt message contents and attachments
□ Add digital signature to outgoing message
□ Request S/MIME receipt for this message

□ Do not AutoArchive this item

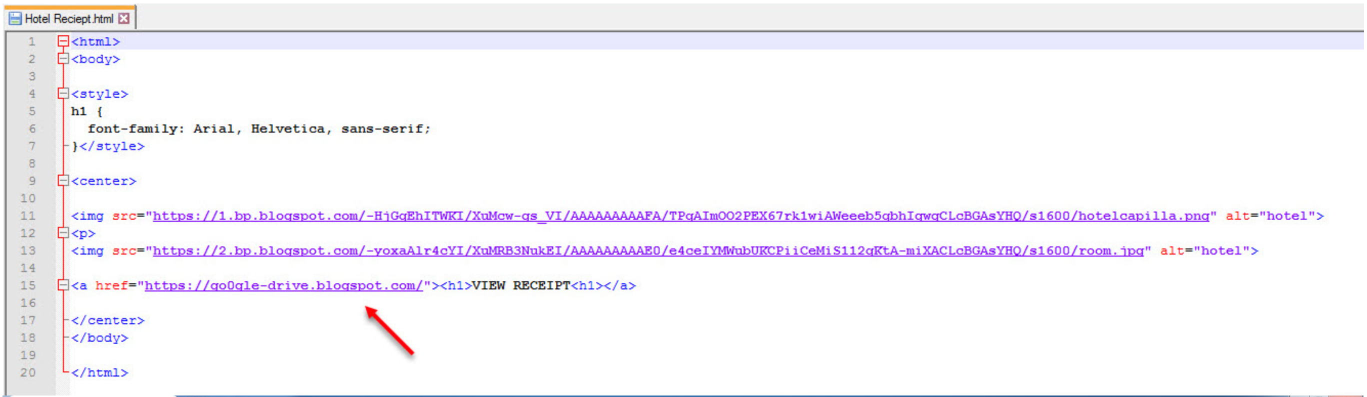
Tracking options
□ Request a delivery receipt for this message
□ Request a read receipt for this message

Delivery options
Have replies sent to
□ Expires after: None 00:00

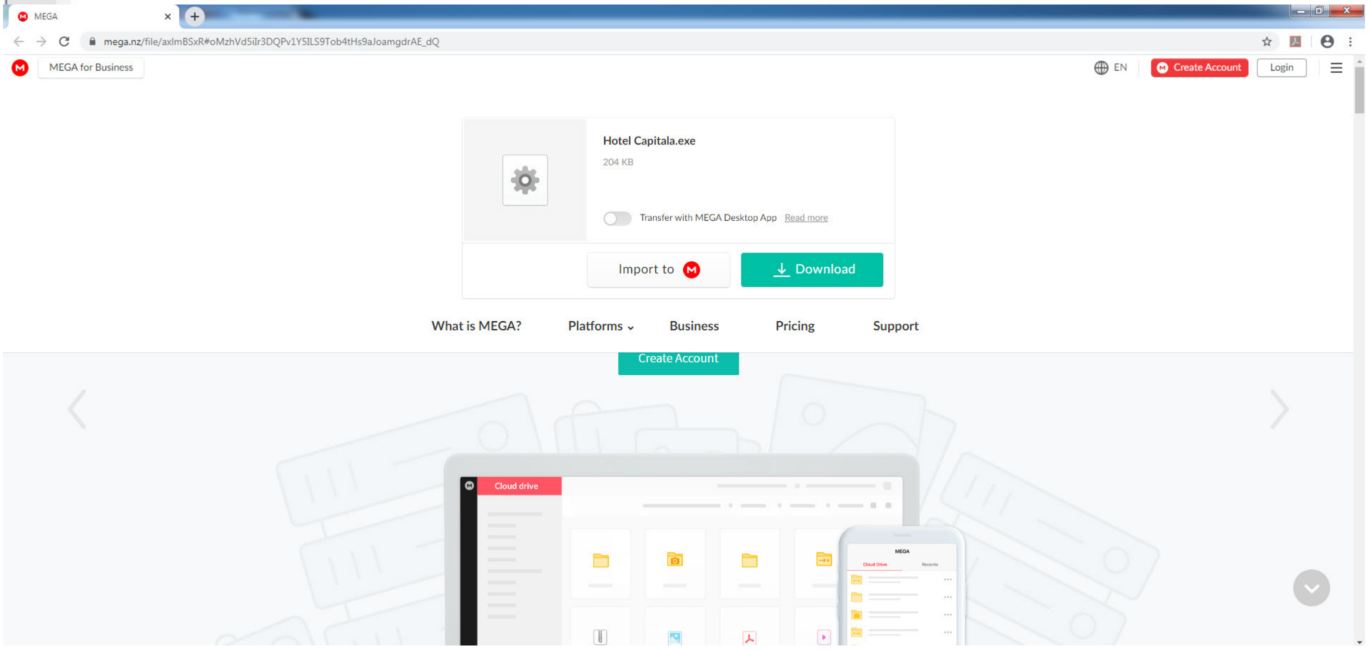
Contacts...
Categories: None

Internet headers:
```
hotelcapitala@spania.com
 does not designate 93.188.3.52 as  permitted sender) identity=mailfrom;
 client-ip=93.188.3.52;  receiver=[redacted];
  envelope-from="hotelcapitala@spania.com";
  x-sender="hotelcapitala@spania.com"; x-conformance=spf_only;
  x-record-type="v=spf1"; x-record-text="v=spf1  include:spf.webhuset.no -
all"
```

Close


Hotel Reciept.html

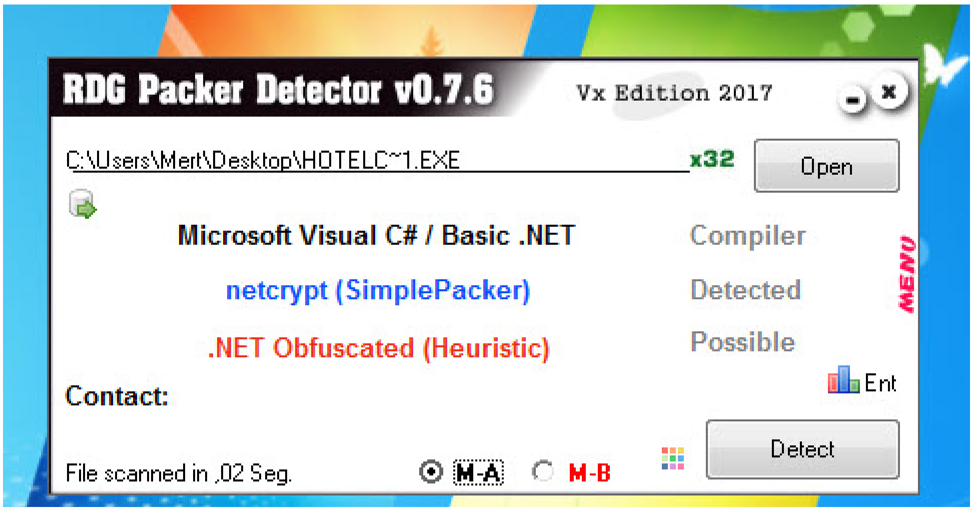
```html
<html>
<body>

<style>
h1 {
  font-family: Arial, Helvetica, sans-serif;
}</style>

<center>

<img src="https://1.bp.blogspot.com/-HjGqEhITWKI/XuMcw-qs_VI/AAAAAAAAAFA/TPqAImOO2PEX67rk1wiAWeeeb5qbhIqwqCLcBGAsYHQ/s1600/hotelcapilla.png" alt="hotel">
<p>
<img src="https://2.bp.blogspot.com/-yoxaAlr4cYI/XuMRB3NukEI/AAAAAAAAAE0/e4ceIYMWubUKCPiiCeMiS112qKtA-miXACLcBGAsYHQ/s1600/room.jpg" alt="hotel">

<a href="https://go0gle-drive.blogspot.com/"><h1>VIEW RECEIPT<h1></a>

</center>
</body>

</html>
```



Hotel Capitala.exe
204 KB
Transfer with MEGA Desktop App Read more

When an institution faces a scenario like the one described above, even if the attack attempt is not successful, it should still handle the matter with great care because this may be an indication of a precursor earthquake, and a sign of a bigger one to come. Therefore, it is important to investigate whether the attack was targeted (Spear Phishing), organized (APT), or just a part of a general campaign targeting a large number of users. It may not always be possible to find answers to these questions, but through analysis, an idea may be gained. In this writing, I will attempt to find answers to these questions.

Initially, through static analysis, I saw that the file was developed and packaged with C#. When I ran the file on a virtual system and analyzed it dynamically, I discovered that the malware accessed an address on the Pastebin site. When I visited this web address, I saw that the page contained an IP address (193.161.193.99) and a port.



C:\Users\Mert\Desktop\HOTELC~1.EXE

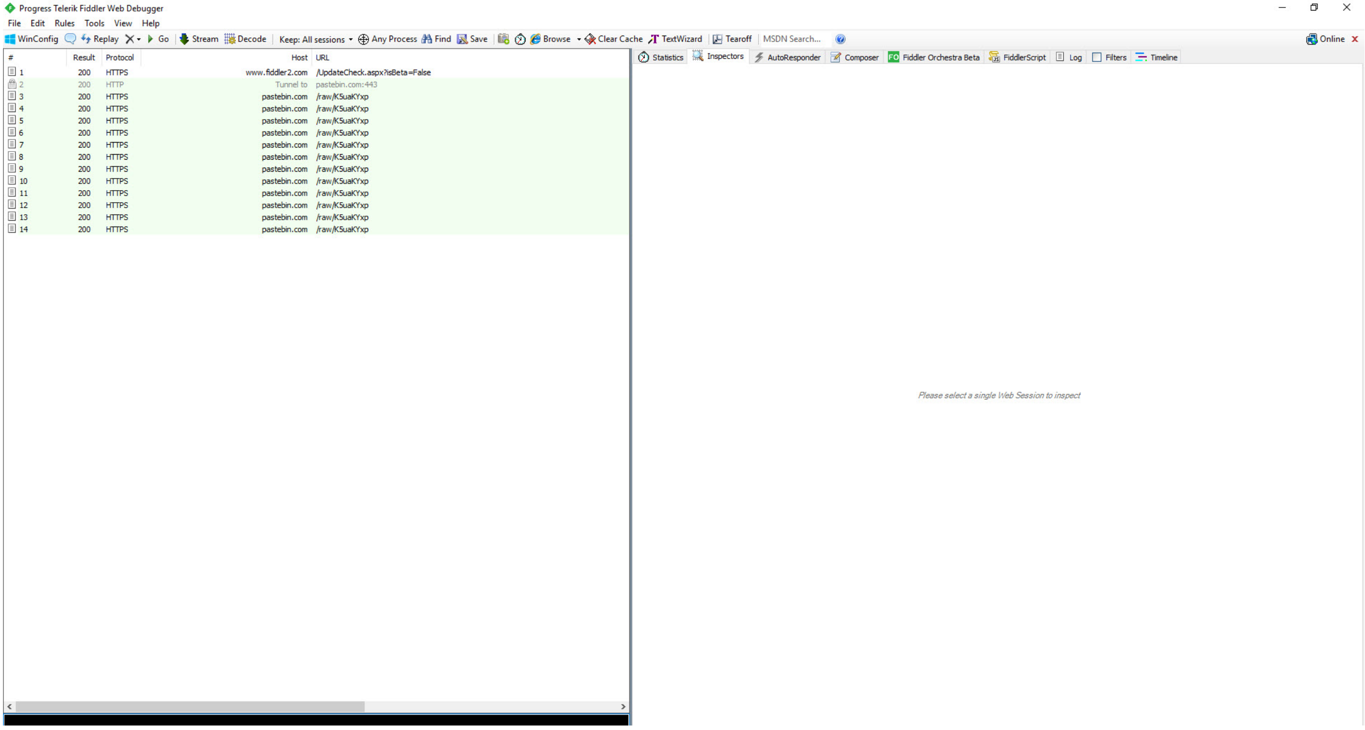
Microsoft Visual C# / Basic .NET — Compiler

netcrypt (SimplePacker) — Detected

.NET Obfuscated (Heuristic) — Possible

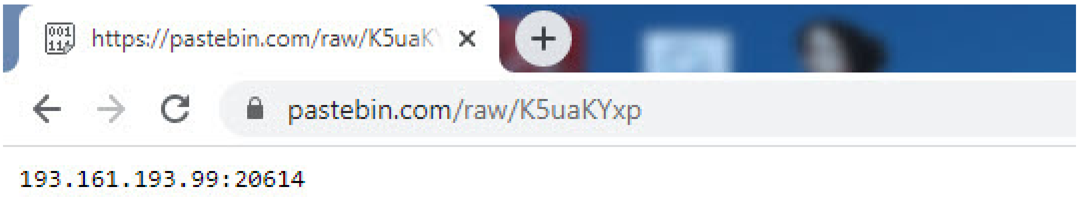
Contact:

File scanned in ,02 Seg.


Progress Telerik Fiddler Web Debugger
File Edit Rules Tools View Help
WinConfig Replay Go Stream Decode Keep: All sessions Any Process Find Save Browse Clear Cache TextWizard Tearoff MSDN Search... Online
Statistics Inspectors AutoResponder Composer Fiddler Orchestra Beta FiddlerScript Log Filters Timeline

#	Result	Protocol	Host	URL
1	200	HTTPS	www.fiddler2.com	/UpdateCheck.aspx?isBeta=False
2	200	HTTP	Tunnel to	pastebin.com:443
3	200	HTTPS	pastebin.com	/raw/K5uaKYxp
4	200	HTTPS	pastebin.com	/raw/K5uaKYxp
5	200	HTTPS	pastebin.com	/raw/K5uaKYxp
6	200	HTTPS	pastebin.com	/raw/K5uaKYxp
7	200	HTTPS	pastebin.com	/raw/K5uaKYxp
8	200	HTTPS	pastebin.com	/raw/K5uaKYxp
9	200	HTTPS	pastebin.com	/raw/K5uaKYxp
10	200	HTTPS	pastebin.com	/raw/K5uaKYxp
11	200	HTTPS	pastebin.com	/raw/K5uaKYxp
12	200	HTTPS	pastebin.com	/raw/K5uaKYxp
13	200	HTTPS	pastebin.com	/raw/K5uaKYxp
14	200	HTTPS	pastebin.com	/raw/K5uaKYxp

Please select a single Web Session to inspect

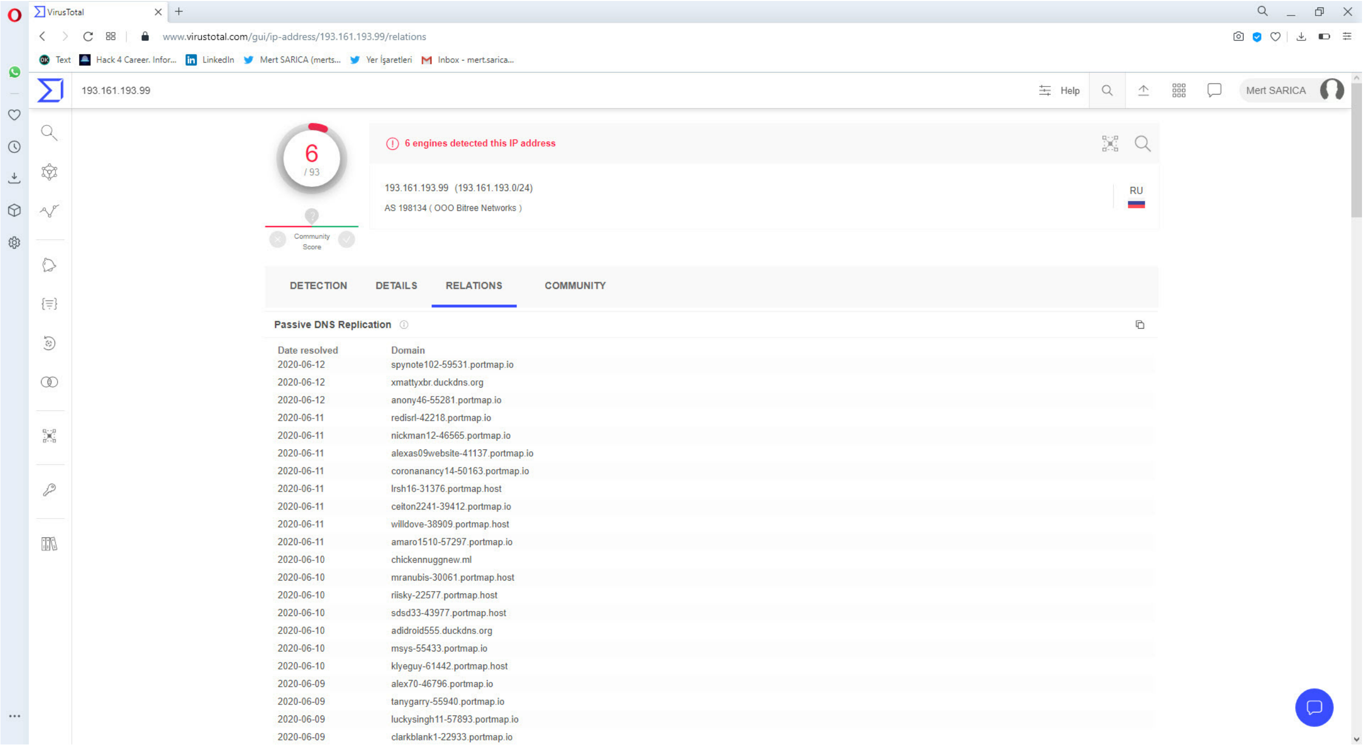


```
193.161.193.99:20614
```

Especially in APT attacks, the malware used is often specially developed by the threat actors and compiled just before the attack, so when it is uploaded to VirusTotal, it is usually detected under a general signature name such as (Backdoor, Trojan, etc). In such cases, it may be possible to use services like Intezer to search for which other malware the code of this malware was used and make comparisons, and thus gain information about the threat actor.

When I uploaded the malware to VirusTotal, I saw that it was not specifically matched with any other malware. When I searched on Intezer, unfortunately, I came up empty handed. (Generic Malware)

When I searched the IP address I obtained from the Pastebin.com page on VirusTotal, I found out that it belongs to the Portmap.io service which serves for redirecting ports.

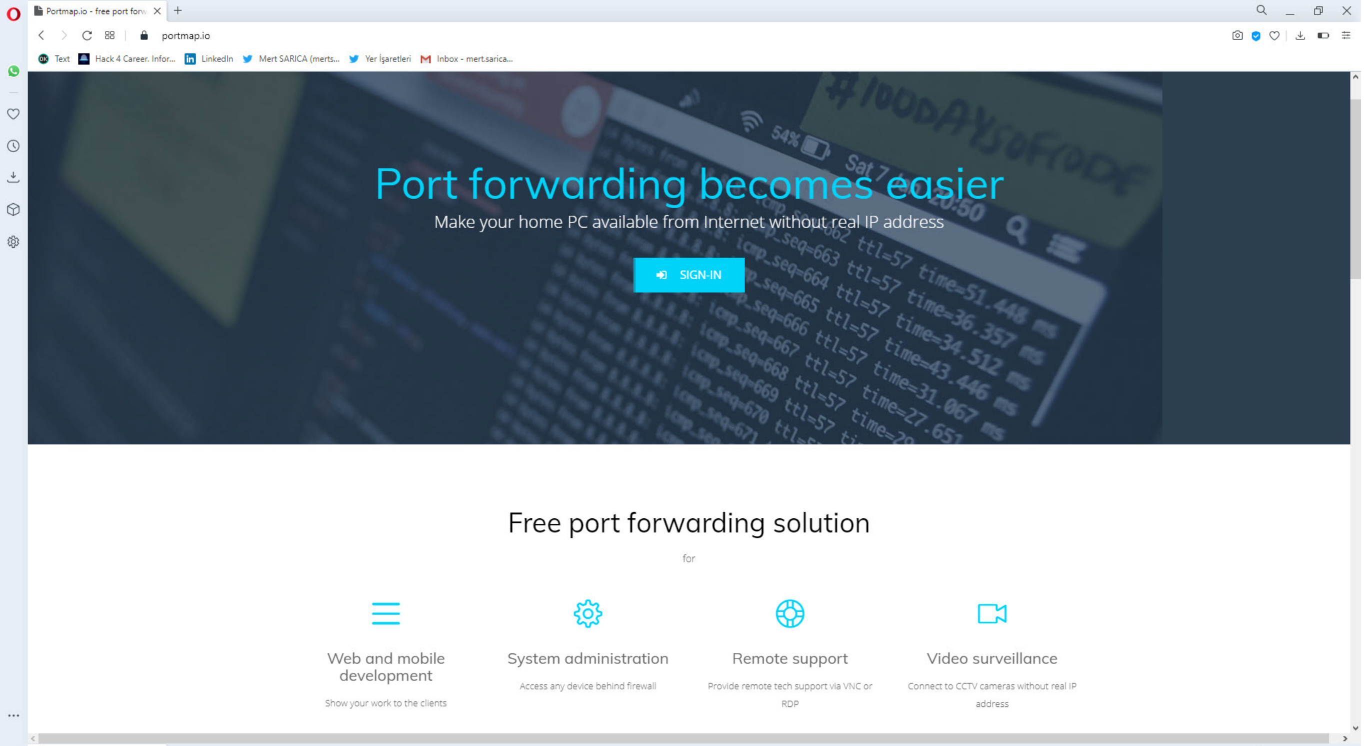


6 engines detected this IP address

193.161.193.99 (193.161.193.0/24)
AS 198134 (OOO Bitree Networks)

Passive DNS Replication

Date resolved	Domain
2020-06-12	spynote102-59531.portmap.io
2020-06-12	xmattyxbr.duckdns.org
2020-06-12	anony46-55281.portmap.io
2020-06-11	redisrl-42218.portmap.io
2020-06-11	nickman12-46565.portmap.io
2020-06-11	alexas09website-41137.portmap.io
2020-06-11	coronanancy14-50163.portmap.io
2020-06-11	lrsh16-31376.portmap.host
2020-06-11	ceiton2241-39412.portmap.io
2020-06-11	willdove-38909.portmap.host
2020-06-11	amaro1510-57297.portmap.io
2020-06-10	chickennuggnew.ml
2020-06-10	mranubis-30061.portmap.host
2020-06-10	riisky-22577.portmap.host
2020-06-10	sdsd33-43977.portmap.host
2020-06-10	adidroid555.duckdns.org
2020-06-10	msys-55433.portmap.io
2020-06-10	klyeguy-61442.portmap.host
2020-06-09	alex70-46796.portmap.io
2020-06-09	tanygarry-55940.portmap.io
2020-06-09	luckysingh11-57893.portmap.io
2020-06-09	clarkblank1-22933.portmap.io



Port forwarding becomes easier

Make your home PC available from Internet without real IP address

Free port forwarding solution

for

Web and mobile development

Show your work to the clients

System administration

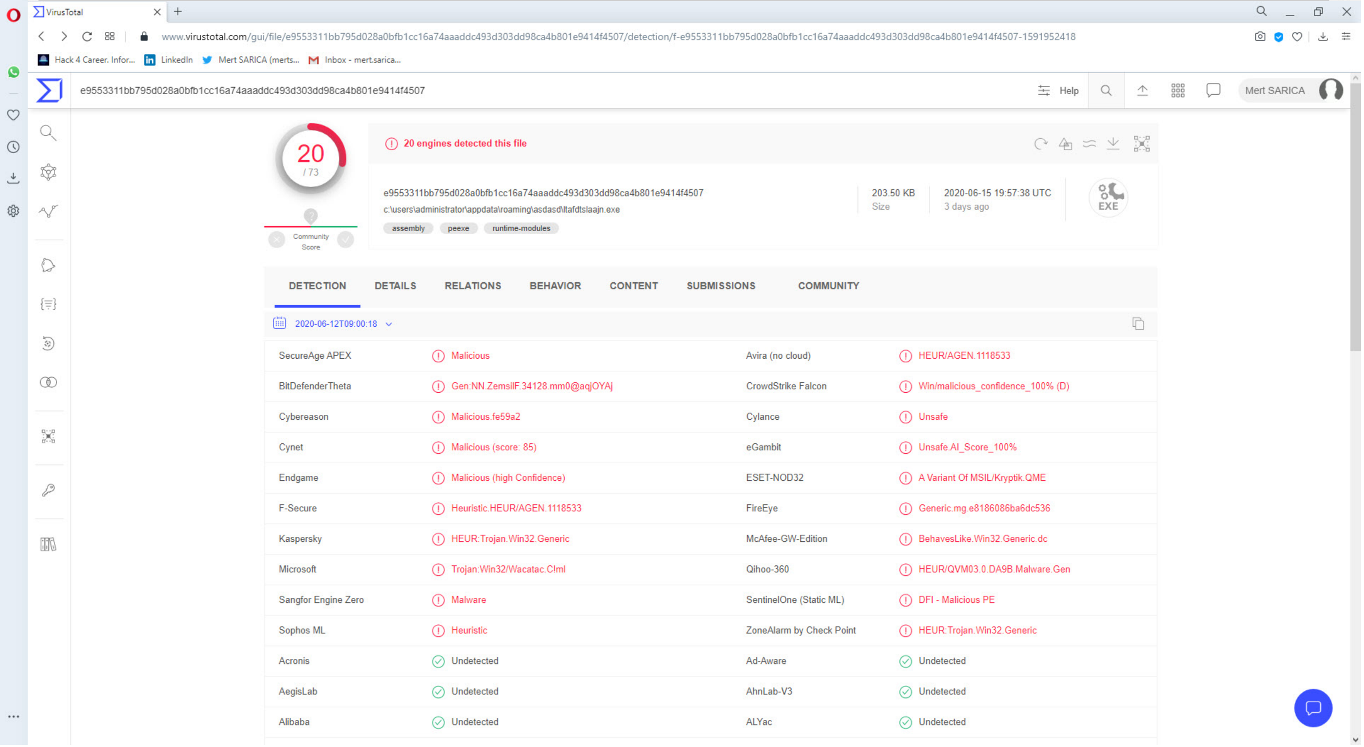
Access any device behind firewall

Remote support

Provide remote tech support via VNC or RDP

Video surveillance

Connect to CCTV cameras without real IP address



www.virustotal.com/gui/file/e9553311bb795d028a0bfb1cc16a74aaaddc493d303dd98ca4b801e9414f4507/detection/f-e9553311bb795d028a0bfb1cc16a74aaaddc493d303dd98ca4b801e9414f4507-1591952418

e9553311bb795d028a0bfb1cc16a74aaaddc493d303dd98ca4b801e9414f4507

20 / 73

20 engines detected this file

e9553311bb795d028a0bfb1cc16a74aaaddc493d303dd98ca4b801e9414f4507
c:\users\administrator\appdata\roaming\asdasd\ltafdtslaajn.exe
assembly peexe runtime-modules

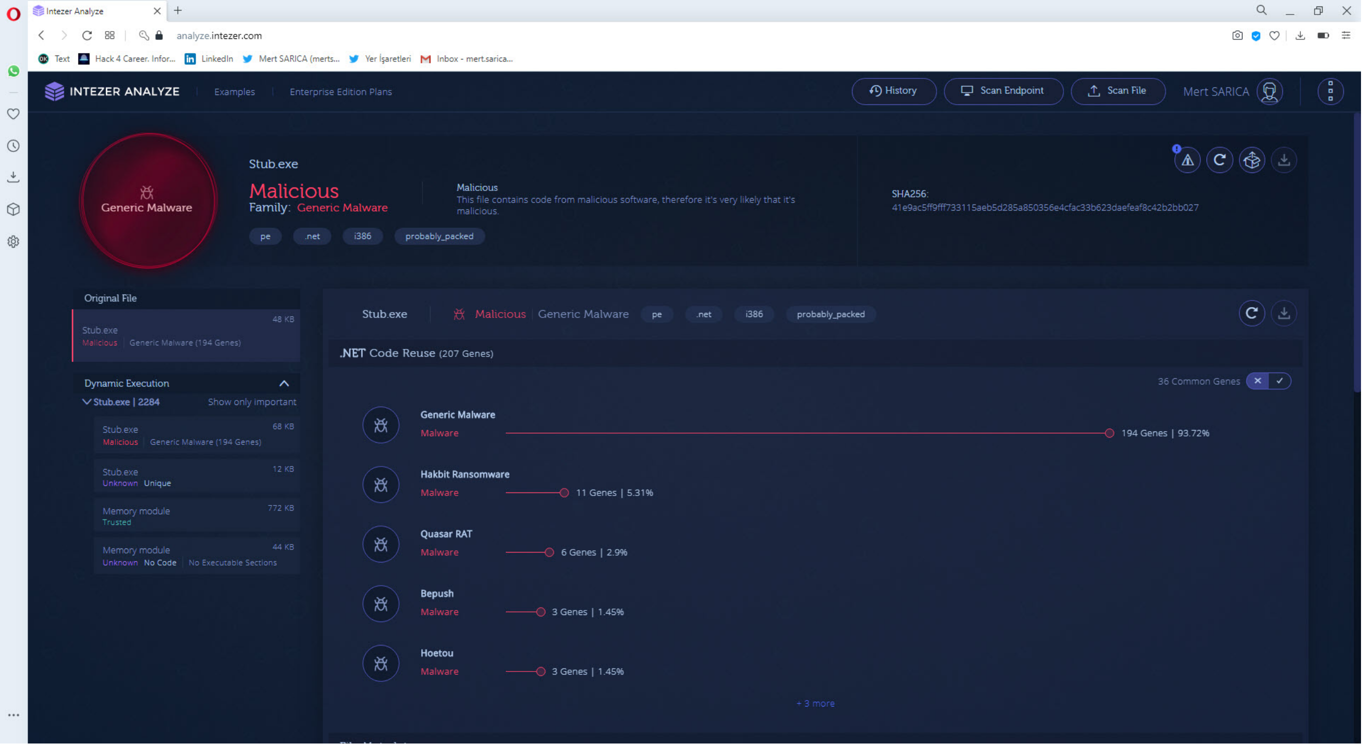
203.50 KB
Size

2020-06-15 19:57:38 UTC
3 days ago

DETECTION DETAILS RELATIONS BEHAVIOR CONTENT SUBMISSIONS COMMUNITY

2020-06-12T09:00:18

SecureAge APEX	Malicious	Avira (no cloud)	HEUR/AGEN.1118533
BitDefenderTheta	Gen:NN.ZemsilF.34128.mm0@aqjOYAj	CrowdStrike Falcon	Win/malicious_confidence_100% (D)
Cybereason	Malicious.fe59a2	Cylance	Unsafe
Cynet	Malicious (score: 85)	eGambit	Unsafe.AI_Score_100%
Endgame	Malicious (high Confidence)	ESET-NOD32	A Variant Of MSIL/Kryptik.QME
F-Secure	Heuristic.HEUR/AGEN.1118533	FireEye	Generic.mg.e8186086ba6dc536
Kaspersky	HEUR:Trojan.Win32.Generic	McAfee-GW-Edition	BehavesLike.Win32.Generic.dc
Microsoft	Trojan:Win32/Wacatac.C!ml	Qihoo-360	HEUR/QVM03.0.DA9B.Malware.Gen
Sangfor Engine Zero	Malware	SentinelOne (Static ML)	DFI - Malicious PE
Sophos ML	Heuristic	ZoneAlarm by Check Point	HEUR:Trojan.Win32.Generic
Acronis	✓ Undetected	Ad-Aware	✓ Undetected
AegisLab	✓ Undetected	AhnLab-V3	✓ Undetected
Alibaba	✓ Undetected	ALYac	✓ Undetected



Stub.exe

Malicious

Family: Generic Malware

pe | .net | i386 | probably_packed

Malicious

This file contains code from malicious software, therefore it's very likely that it's malicious.

SHA256:
41e9ac5ff9fff733115aeb5d285a850356e4cfac33b623daefeaf8c42b2bb027

Original File

Stub.exe 48 KB
Malicious | Generic Malware (194 Genes)

Dynamic Execution

Stub.exe | 2284 — Show only important

Stub.exe	Malicious	Generic Malware (194 Genes)	68 KB
Stub.exe	Unknown	Unique	12 KB
Memory module	Trusted		772 KB
Memory module	Unknown	No Code | No Executable Sections	44 KB

Stub.exe | Malicious | Generic Malware | pe | .net | i386 | probably_packed

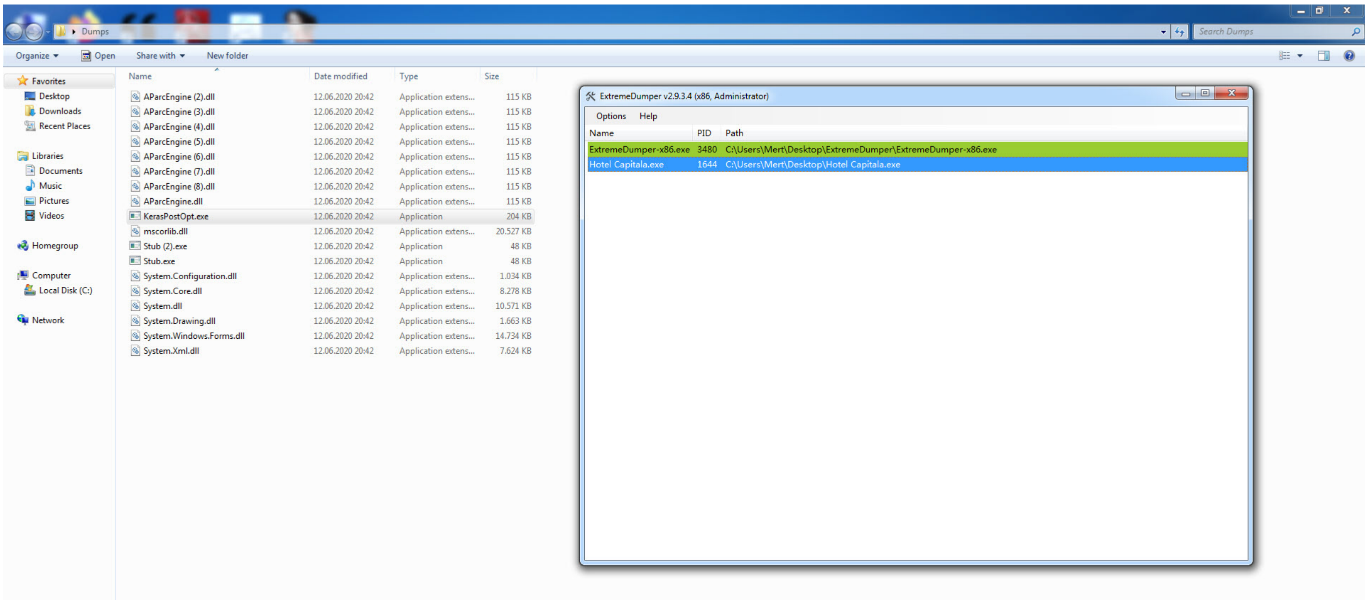
.NET Code Reuse (207 Genes)

36 Common Genes

Generic Malware	Malware	194 Genes | 93.72%
Hakbit Ransomware	Malware	11 Genes | 5.31%
Quasar RAT	Malware	6 Genes | 2.9%
Bepush	Malware	3 Genes | 1.45%
Hoetou	Malware	3 Genes | 1.45%

+ 3 more

As I continued my research to find out what the malware developed by the threat actor who tries to hide himself as much as possible, I reached the stage of dynamic code analysis and the dnSpy debugger that I used in my article titled OPSEC came to my aid. Before starting debugging with dnSpy, in order to find the main module that the packaged software hides in memory, when I ran the ExtremeDumper tool, the mother of evils, Stub.exe, emerged.

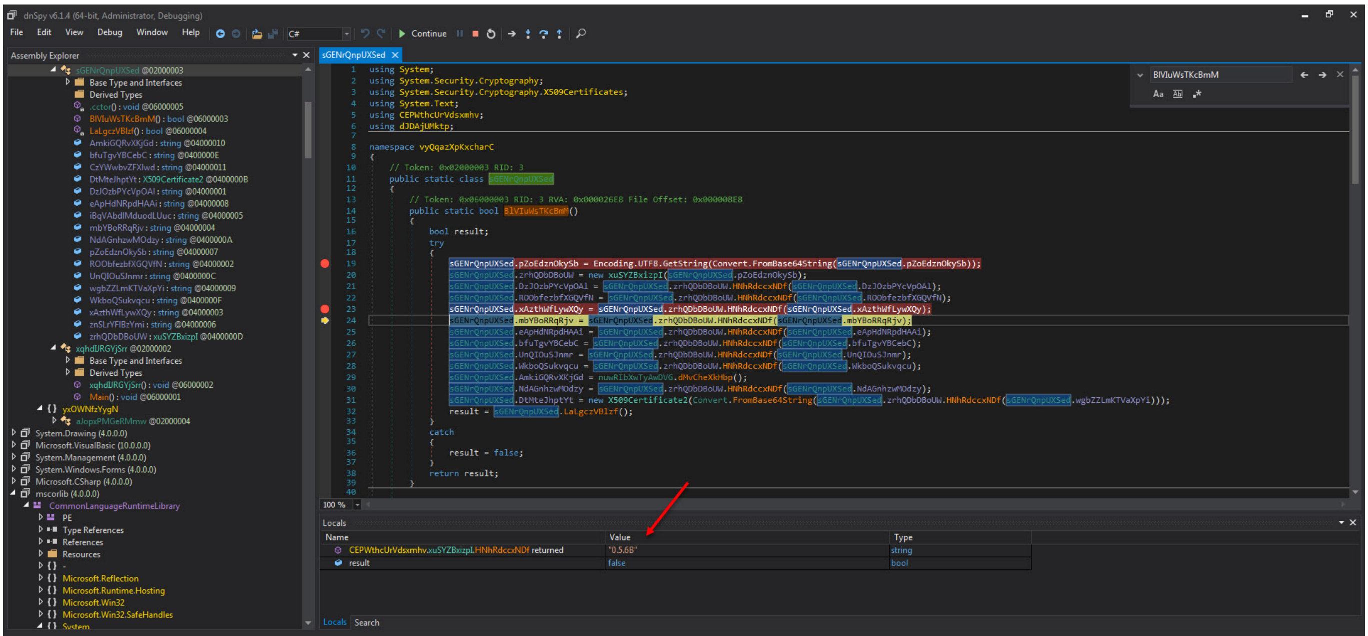


Name	Date modified	Type	Size
AParcEngine (2).dll	12.06.2020 20:42	Application extens...	115 KB
AParcEngine (3).dll	12.06.2020 20:42	Application extens...	115 KB
AParcEngine (4).dll	12.06.2020 20:42	Application extens...	115 KB
AParcEngine (5).dll	12.06.2020 20:42	Application extens...	115 KB
AParcEngine (6).dll	12.06.2020 20:42	Application extens...	115 KB
AParcEngine (7).dll	12.06.2020 20:42	Application extens...	115 KB
AParcEngine (8).dll	12.06.2020 20:42	Application extens...	115 KB
AParcEngine.dll	12.06.2020 20:42	Application extens...	115 KB
KerasPostOpt.exe	12.06.2020 20:42	Application	204 KB
mscorlib.dll	12.06.2020 20:42	Application extens...	20.527 KB
Stub (2).exe	12.06.2020 20:42	Application	48 KB
Stub.exe	12.06.2020 20:42	Application	48 KB
System.Configuration.dll	12.06.2020 20:42	Application extens...	1.034 KB
System.Core.dll	12.06.2020 20:42	Application extens...	8.278 KB
System.dll	12.06.2020 20:42	Application extens...	10.571 KB
System.Drawing.dll	12.06.2020 20:42	Application extens...	1.663 KB
System.Windows.Forms.dll	12.06.2020 20:42	Application extens...	14.734 KB
System.Xml.dll	12.06.2020 20:42	Application extens...	7.624 KB

ExtremeDumper v2.9.3.4 (x86, Administrator)

Name	PID	Path
ExtremeDumper-x86.exe	3480	C:\Users\Mert\Desktop\ExtremeDumper\ExtremeDumper-x86.exe
Hotel Capitala.exe	1644	C:\Users\Mert\Desktop\Hotel Capitala.exe

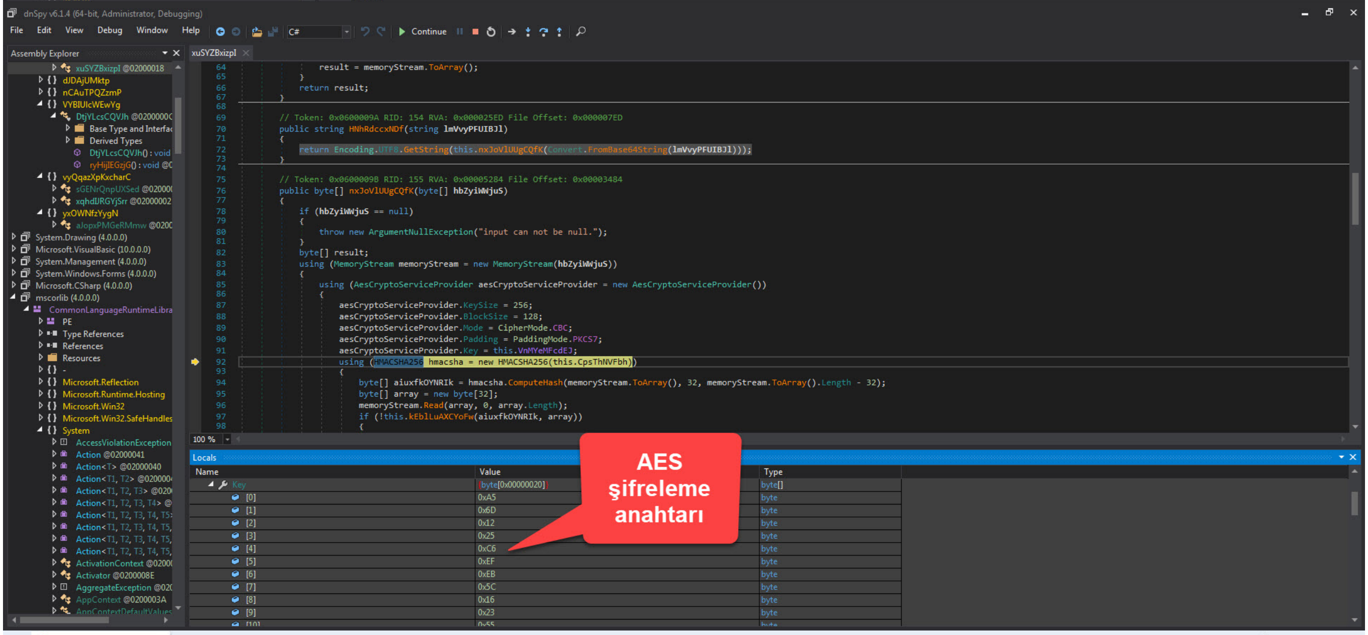
As I analyzed the Stub.exe program step by step with dnSpy, at one point, I noticed that it was encrypted with AES and the decrypted value of 0.5.6B caught my attention. When I searched this value on Google with the keywords “rat 0.5.6B,” guess what came up? The open-source AsyncRAT! :)



```
sGENrQnpUXSed @02000003
    Base Type and Interfaces
    Derived Types
    .cctor() : void @06000005
    BlVIuWsTKcBmM() : bool @06000003
    LaLgczVBlzf() : bool @06000004
    AmkiGQRvXKjGd : string @04000010
    bfuTgvYBCebC : string @0400000E
    CzYWwbvZFXlwd : string @04000011
    DtMteJhptYt : X509Certificate2 @0400000B
    DzJOzbPYcVpOAl : string @04000001
    eApHdNRpdHAAi : string @04000008
    iBqVAbdlMduodLUuc : string @04000005
    mbYBoRRqRjv : string @04000004
    NdAGnhzwMOdzy : string @0400000A
    pZoEdznOkySb : string @04000007
    ROObfezbfXGQVfN : string @04000002
    UnQIOuSJnmr : string @0400000C
    wgbZZLmKTVaXpYi : string @04000009
    WkboQSukvqcu : string @0400000F
    xAzthWfLywXQy : string @04000003
    znSLrYFIBzYmi : string @04000006
    zrhQDbDBoUW : xuSYZBxizpI @0400000D
xqhdIJRGYjSrr @02000002
    Base Type and Interfaces
    Derived Types
    xqhdIJRGYjSrr() : void @06000002
    Main() : void @06000001
{ } yxOWNfzYygN
    aJopxPMGeRMmw @02000004
System.Drawing (4.0.0.0)
Microsoft.VisualBasic (10.0.0.0)
System.Management (4.0.0.0)
System.Windows.Forms (4.0.0.0)
Microsoft.CSharp (4.0.0.0)
mscorlib (4.0.0.0)
    CommonLanguageRuntimeLibrary
        PE
        Type References
        References
        Resources
        { } -
        { } Microsoft.Reflection
        { } Microsoft.Runtime.Hosting
        { } Microsoft.Win32
        { } Microsoft.Win32.SafeHandles
        { } System
```

```csharp
using System;
using System.Security.Cryptography;
using System.Security.Cryptography.X509Certificates;
using System.Text;
using CEPWthcUrVdsxmhv;
using dJDAjUMktp;

namespace vyQqazXpKxcharC
{
    // Token: 0x02000003 RID: 3
    public static class sGENrQnpUXSed
    {
        // Token: 0x06000003 RID: 3 RVA: 0x000026E8 File Offset: 0x000008E8
        public static bool BlVIuWsTKcBmM()
        {
            bool result;
            try
            {
                sGENrQnpUXSed.pZoEdznOkySb = Encoding.UTF8.GetString(Convert.FromBase64String(sGENrQnpUXSed.pZoEdznOkySb));
                sGENrQnpUXSed.zrhQDbDBoUW = new xuSYZBxizpI(sGENrQnpUXSed.pZoEdznOkySb);
                sGENrQnpUXSed.DzJOzbPYcVpOAl = sGENrQnpUXSed.zrhQDbDBoUW.HNhRdccxNDf(sGENrQnpUXSed.DzJOzbPYcVpOAl);
                sGENrQnpUXSed.ROObfezbfXGQVfN = sGENrQnpUXSed.zrhQDbDBoUW.HNhRdccxNDf(sGENrQnpUXSed.ROObfezbfXGQVfN);
                sGENrQnpUXSed.xAzthWfLywXQy = sGENrQnpUXSed.zrhQDbDBoUW.HNhRdccxNDf(sGENrQnpUXSed.xAzthWfLywXQy);
                sGENrQnpUXSed.mbYBoRRqRjv = sGENrQnpUXSed.zrhQDbDBoUW.HNhRdccxNDf(sGENrQnpUXSed.mbYBoRRqRjv);
                sGENrQnpUXSed.eApHdNRpdHAAi = sGENrQnpUXSed.zrhQDbDBoUW.HNhRdccxNDf(sGENrQnpUXSed.eApHdNRpdHAAi);
                sGENrQnpUXSed.bfuTgvYBCebC = sGENrQnpUXSed.zrhQDbDBoUW.HNhRdccxNDf(sGENrQnpUXSed.bfuTgvYBCebC);
                sGENrQnpUXSed.UnQIOuSJnmr = sGENrQnpUXSed.zrhQDbDBoUW.HNhRdccxNDf(sGENrQnpUXSed.UnQIOuSJnmr);
                sGENrQnpUXSed.WkboQSukvqcu = sGENrQnpUXSed.zrhQDbDBoUW.HNhRdccxNDf(sGENrQnpUXSed.WkboQSukvqcu);
                sGENrQnpUXSed.AmkiGQRvXKjGd = nuwRIbXwTyAwDVG.dMvCheXkHbp();
                sGENrQnpUXSed.NdAGnhzwMOdzy = sGENrQnpUXSed.zrhQDbDBoUW.HNhRdccxNDf(sGENrQnpUXSed.NdAGnhzwMOdzy);
                sGENrQnpUXSed.DtMteJhptYt = new X509Certificate2(Convert.FromBase64String(sGENrQnpUXSed.zrhQDbDBoUW.HNhRdccxNDf(sGENrQnpUXSed.wgbZZLmKTVaXpYi)));
                result = sGENrQnpUXSed.LaLgczVBlzf();
            }
            catch
            {
                result = false;
            }
            return result;
        }
```

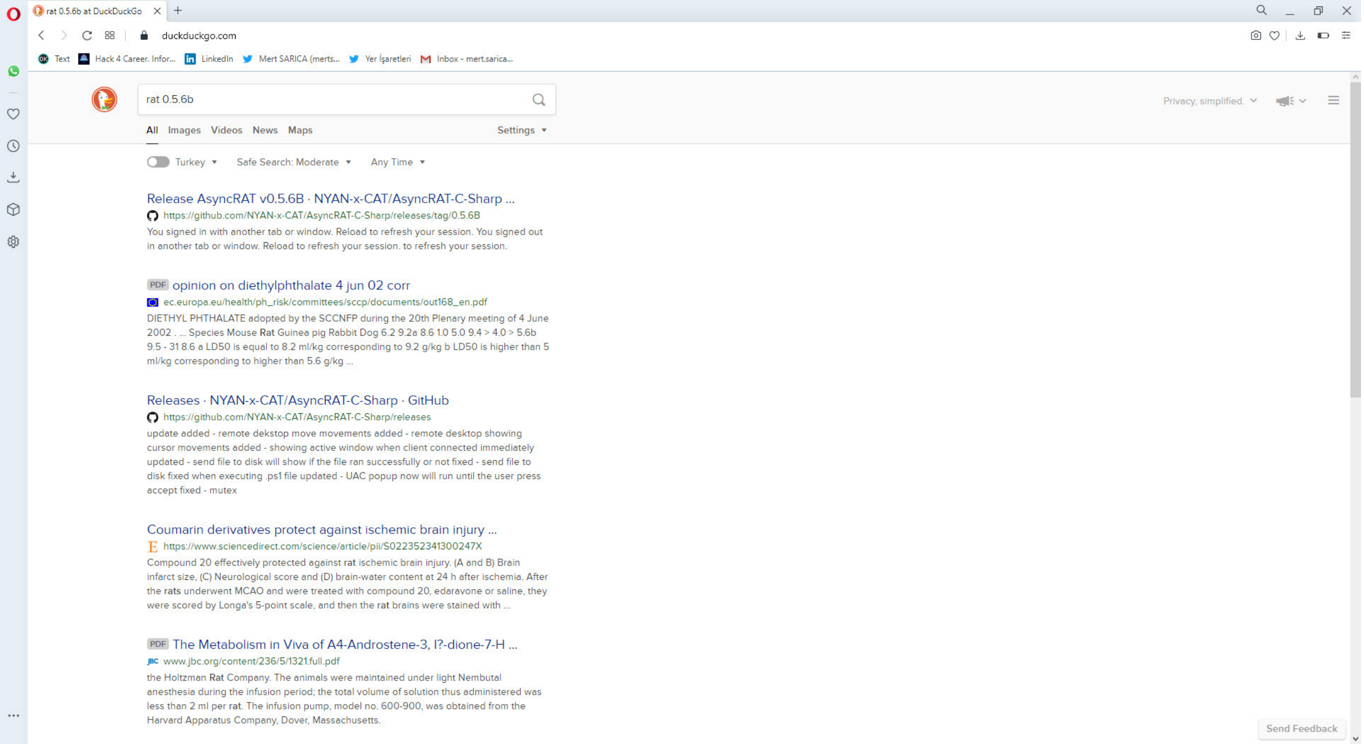
Name	Value	Type
CEPWthcUrVdsxmhv.xuSYZBxizpI.HNhRdccxNDf returned	"0.5.6B"	string
result	false	bool



```
xuSYZBxizpI @02000018
{ } dJDAjUMktp
{ } nCAuTPQZzmP
{ } VYBIUlcWEwYg
    DtjYLcsCQVJh @0200000C
        Base Type and Interfac
        Derived Types
        DtjYLcsCQVJh() : void
        ryHijIEGzjG() : void @0
{ } vyQqazXpKxcharC
    sGENrQnpUXSed @02000
    xqhdIJRGYjSrr @02000002
{ } yxOWNfzYygN
    aJopxPMGeRMmw @0200
System.Drawing (4.0.0.0)
Microsoft.VisualBasic (10.0.0.0)
System.Management (4.0.0.0)
System.Windows.Forms (4.0.0.0)
Microsoft.CSharp (4.0.0.0)
mscorlib (4.0.0.0)
    CommonLanguageRuntimeLibra
        PE
        Type References
        References
        Resources
        { } -
        { } Microsoft.Reflection
        { } Microsoft.Runtime.Hosting
        { } Microsoft.Win32
        { } Microsoft.Win32.SafeHandles
        { } System
            AccessViolationException
            Action @02000041
            Action<T> @02000040
            Action<T1, T2> @020000
            Action<T1, T2, T3> @020
            Action<T1, T2, T3, T4> @
            Action<T1, T2, T3, T4, T5
            Action<T1, T2, T3, T4, T5,
            Action<T1, T2, T3, T4, T5,
            Action<T1, T2, T3, T4, T5,
            ActivationContext @0200
            Activator @0200008E
            AggregateException @020
            AppContext @0200003A
            AppContextDefaultValues
```

```csharp
                    result = memoryStream.ToArray();
                }
                return result;
            }

            // Token: 0x0600009A RID: 154 RVA: 0x000025ED File Offset: 0x000007ED
            public string HNhRdccxNDf(string lmVvyPFUIBJl)
            {
                return Encoding.UTF8.GetString(this.nxJoVlUUgCQfK(Convert.FromBase64String(lmVvyPFUIBJl)));
            }

            // Token: 0x0600009B RID: 155 RVA: 0x00005284 File Offset: 0x00003484
            public byte[] nxJoVlUUgCQfK(byte[] hbZyiWWjuS)
            {
                if (hbZyiWWjuS == null)
                {
                    throw new ArgumentNullException("input can not be null.");
                }
                byte[] result;
                using (MemoryStream memoryStream = new MemoryStream(hbZyiWWjuS))
                {
                    using (AesCryptoServiceProvider aesCryptoServiceProvider = new AesCryptoServiceProvider())
                    {
                        aesCryptoServiceProvider.KeySize = 256;
                        aesCryptoServiceProvider.BlockSize = 128;
                        aesCryptoServiceProvider.Mode = CipherMode.CBC;
                        aesCryptoServiceProvider.Padding = PaddingMode.PKCS7;
                        aesCryptoServiceProvider.Key = this.VnMYeMFcdEJ;
                        using (HMACSHA256 hmacsha = new HMACSHA256(this.CpsThNVFbh))
                        {
                            byte[] aiuxfkOYNRIk = hmacsha.ComputeHash(memoryStream.ToArray(), 32, memoryStream.ToArray().Length - 32);
                            byte[] array = new byte[32];
                            memoryStream.Read(array, 0, array.Length);
                            if (!this.kEblLuAXCYoFw(aiuxfkOYNRIk, array))
                            {
```

Name	Value	Type
Key	{byte[0x00000020]}	byte[]
[0]	0xA5	byte
[1]	0x6D	byte
[2]	0x12	byte
[3]	0x25	byte
[4]	0xC6	byte
[5]	0xEF	byte
[6]	0xEB	byte
[7]	0x5C	byte
[8]	0x16	byte
[9]	0x23	byte
[10]	0x55	byte



rat 0.5.6b

Release AsyncRAT v0.5.6B · NYAN-x-CAT/AsyncRAT-C-Sharp ...
https://github.com/NYAN-x-CAT/AsyncRAT-C-Sharp/releases/tag/0.5.6B
You signed in with another tab or window. Reload to refresh your session. You signed out in another tab or window. Reload to refresh your session. to refresh your session.

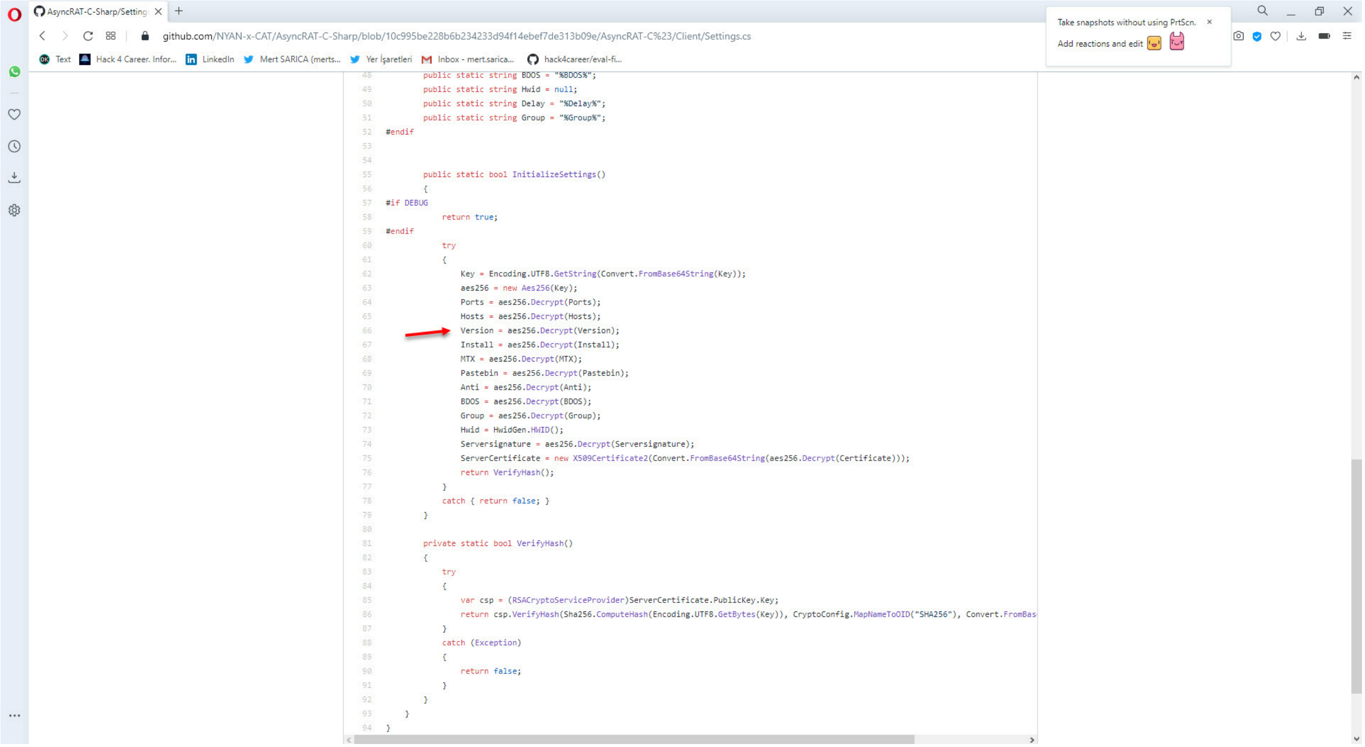
PDF **opinion on diethylphthalate 4 jun 02 corr**
ec.europa.eu/health/ph_risk/committees/sccp/documents/out168_en.pdf
DIETHYL PHTHALATE adopted by the SCCNFP during the 20th Plenary meeting of 4 June 2002 . ... Species Mouse **Rat** Guinea pig Rabbit Dog 6.2 9.2a 8.6 1.0 5.0 9.4 > 4.0 > 5.6b 9.5 - 31 8.6 a LD50 is equal to 8.2 ml/kg corresponding to 9.2 g/kg b LD50 is higher than 5 ml/kg corresponding to higher than 5.6 g/kg ...

Releases · NYAN-x-CAT/AsyncRAT-C-Sharp · GitHub
https://github.com/NYAN-x-CAT/AsyncRAT-C-Sharp/releases
update added - remote dekstop move movements added - remote desktop showing cursor movements added - showing active window when client connected immediately updated - send file to disk will show if the file ran successfully or not fixed - send file to disk fixed when executing .ps1 file updated - UAC popup now will run until the user press accept fixed - mutex

Coumarin derivatives protect against ischemic brain injury ...
https://www.sciencedirect.com/science/article/pii/S022352341300247X
Compound 20 effectively protected against **rat** ischemic brain injury. (A and B) Brain infarct size, (C) Neurological score and (D) brain-water content at 24 h after ischemia. After the **rats** underwent MCAO and were treated with compound 20, edaravone or saline, they were scored by Longa's 5-point scale, and then the **rat** brains were stained with ...

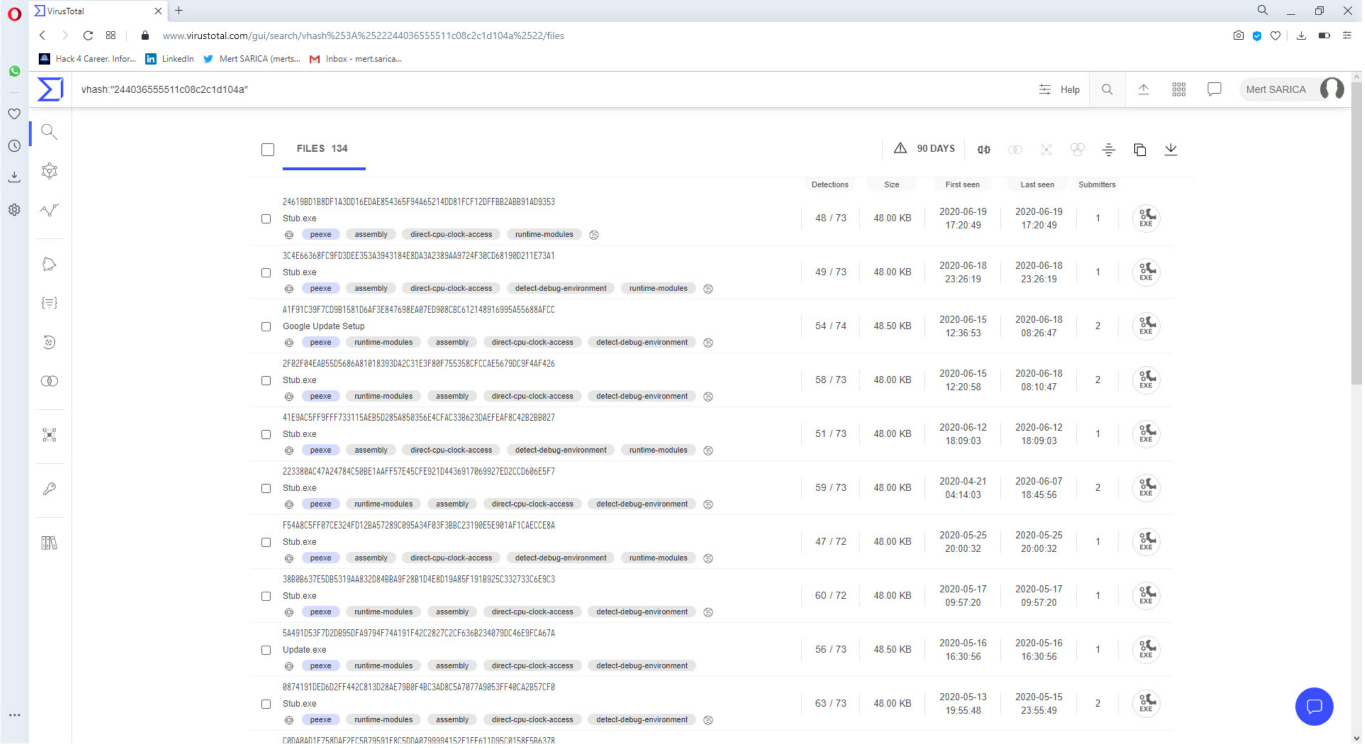
PDF **The Metabolism in Viva of A4-Androstene-3, l?-dione-7-H ...**
www.jbc.org/content/236/5/1321.full.pdf
the Holtzman **Rat** Company. The animals were maintained under light Nembutal anesthesia during the infusion period; the total volume of solution thus administered was less than 2 ml per **rat**. The infusion pump, model no. 600-900, was obtained from the Harvard Apparatus Company, Dover, Massachusetts.

After examining this project in detail on GitHub, I was able to confirm that the malware I analyzed is AsyncRAT by inferring it from similar code blocks.



```
        public static string BDOS = "%BDOS%";
        public static string Hwid = null;
        public static string Delay = "%Delay%";
        public static string Group = "%Group%";
#endif


        public static bool InitializeSettings()
        {
#if DEBUG
            return true;
#endif
            try
            {
                Key = Encoding.UTF8.GetString(Convert.FromBase64String(Key));
                aes256 = new Aes256(Key);
                Ports = aes256.Decrypt(Ports);
                Hosts = aes256.Decrypt(Hosts);
                Version = aes256.Decrypt(Version);
                Install = aes256.Decrypt(Install);
                MTX = aes256.Decrypt(MTX);
                Pastebin = aes256.Decrypt(Pastebin);
                Anti = aes256.Decrypt(Anti);
                BDOS = aes256.Decrypt(BDOS);
                Group = aes256.Decrypt(Group);
                Hwid = HwidGen.HWID();
                Serversignature = aes256.Decrypt(Serversignature);
                ServerCertificate = new X509Certificate2(Convert.FromBase64String(aes256.Decrypt(Certificate)));
                return VerifyHash();
            }
            catch { return false; }
        }

        private static bool VerifyHash()
        {
            try
            {
                var csp = (RSACryptoServiceProvider)ServerCertificate.PublicKey.Key;
                return csp.VerifyHash(Sha256.ComputeHash(Encoding.UTF8.GetBytes(Key)), CryptoConfig.MapNameToOID("SHA256"), Convert.FromBas
            }
            catch (Exception)
            {
                return false;
            }
        }
    }
}
```

Finally, when I searched for similar Stub.exe files with vhash on VirusTotal, I encountered many examples. As I wondered whether all these examples had the Pastebin address from the malware I analyzed, or were part of a common campaign, either I would have to examine the analysis report of each of more than 50 examples or find a very short and practical way which is suitable for lazy people. :) After starting to think in a cunning way, the idea of preparing a tool in Python that analyzes all these examples statically, first finds the AES encryption key and then extracts the configuration information came to my mind.

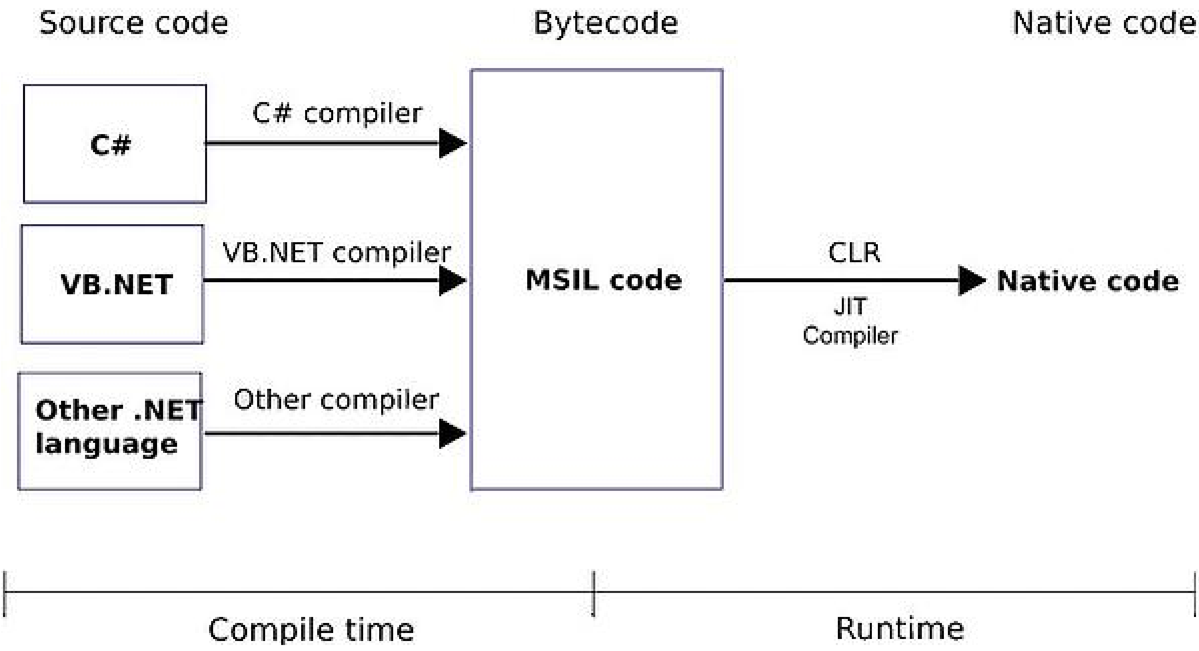


vhash:"244036555511c08c2c1d104a"

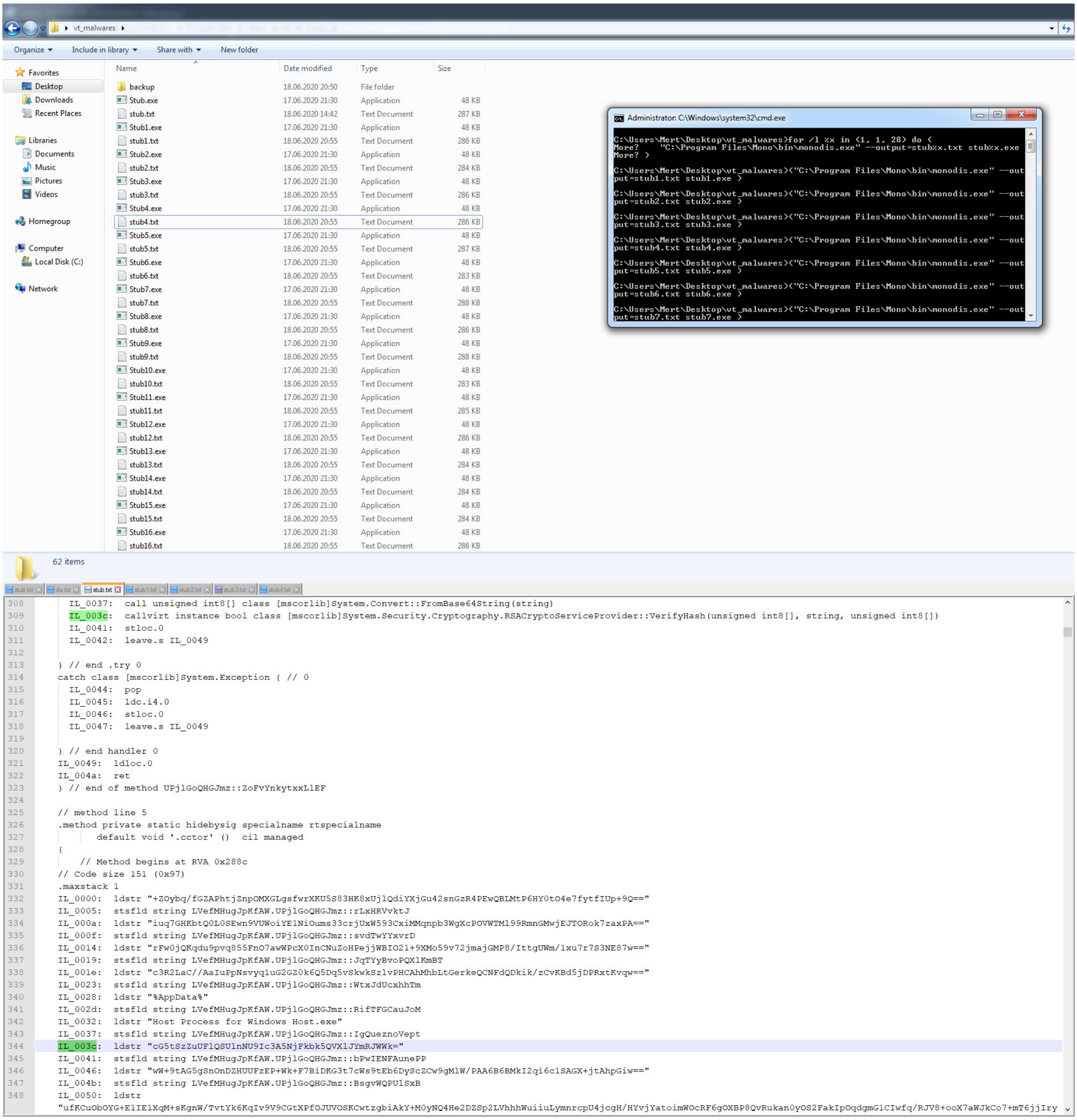
□ FILES 134

	Detections	Size	First seen	Last seen	Submitters
24619BD1B8DF1A3DD16EDAE854365F94A65214DD81FCF12DFFBB2ABB91AD9353 Stub.exe peexe assembly direct-cpu-clock-access runtime-modules	48 / 73	48.00 KB	2020-06-19 17:20:49	2020-06-19 17:20:49	1
3C4E66368FC9FD3DEE353A3943184E8DA3A2389AA9724F30CD68190D211E73A1 Stub.exe peexe assembly direct-cpu-clock-access detect-debug-environment runtime-modules	49 / 73	48.00 KB	2020-06-18 23:26:19	2020-06-18 23:26:19	1
A1F91C39F7CD9B1581D6AF3E847698EA07ED908CBC612148916995A55688AFCC Google Update Setup peexe runtime-modules assembly direct-cpu-clock-access detect-debug-environment	54 / 74	48.50 KB	2020-06-15 12:36:53	2020-06-18 08:26:47	2
2F02F04EAB55D5686A81018393DA2C31E3F80F755358CFCCAE5679DC9F4AF426 Stub.exe peexe runtime-modules assembly direct-cpu-clock-access detect-debug-environment	58 / 73	48.00 KB	2020-06-15 12:20:58	2020-06-18 08:10:47	2
41E9AC5FF9FFF733115AEB5D285A850356E4CFAC33B623DAEFEAF8C42B2BB027 Stub.exe peexe assembly direct-cpu-clock-access detect-debug-environment runtime-modules	51 / 73	48.00 KB	2020-06-12 18:09:03	2020-06-12 18:09:03	1
223380AC47A24784C50BE1AAFF57E45CFE921D4436917069927ED2CCD606E5F7 Stub.exe peexe runtime-modules assembly direct-cpu-clock-access detect-debug-environment	59 / 73	48.00 KB	2020-04-21 04:14:03	2020-06-07 18:45:56	2
F54A8C5FF07CE324FD12BA57289C095A34F03F3BBC23190E5E901AF1CAECCE8A Stub.exe peexe assembly direct-cpu-clock-access detect-debug-environment runtime-modules	47 / 72	48.00 KB	2020-05-25 20:00:32	2020-05-25 20:00:32	1
38B0B637E5DB5319AA832D84BBA9F28B1D4E8D19A85F191B925C332733C6E9C3 Stub.exe peexe runtime-modules assembly direct-cpu-clock-access detect-debug-environment	60 / 72	48.00 KB	2020-05-17 09:57:20	2020-05-17 09:57:20	1
5A491D53F7D2DB95DFA9794F74A191F42C2827C2CF636B234079DC46E9FCA67A Update.exe peexe runtime-modules assembly direct-cpu-clock-access detect-debug-environment	56 / 73	48.50 KB	2020-05-16 16:30:56	2020-05-16 16:30:56	1
0874191DED6D2FF442C813D28AE79B0F4BC3AD8C5A7077A9053FF40CA2B57CF0 Stub.exe peexe runtime-modules assembly direct-cpu-clock-access detect-debug-environment	63 / 73	48.00 KB	2020-05-13 19:55:48	2020-05-15 23:55:49	2
C0DA0AD1F758DAF2FC5B79591F8C5DDA0799994152F1FF611D95C0158F5B6378					

Of course, since the variable names are randomly generated in each program, I had to first find the AES key by using a static variable. Since we know that programs developed with .Net are compiled into bytecode (CIL/MSIL), I started to search for static values on bytecode.



For this, I decided to take advantage of the Mono Disassembler (monodis) tool, which is part of the famous Mono project. Using the monodis tool, I converted all Stub.exe examples to code, and I found out that the AES encryption key is always after the 0x288c value, and the IL_003c value. And using this information, I developed the AsyncRAT Configuration Extractor tool in Python. When I run the tool on all examples, I found that the information in the configuration of each one of them was different from the malware I analyzed, so I learned that the malware I analyzed was not a part of a common campaign.

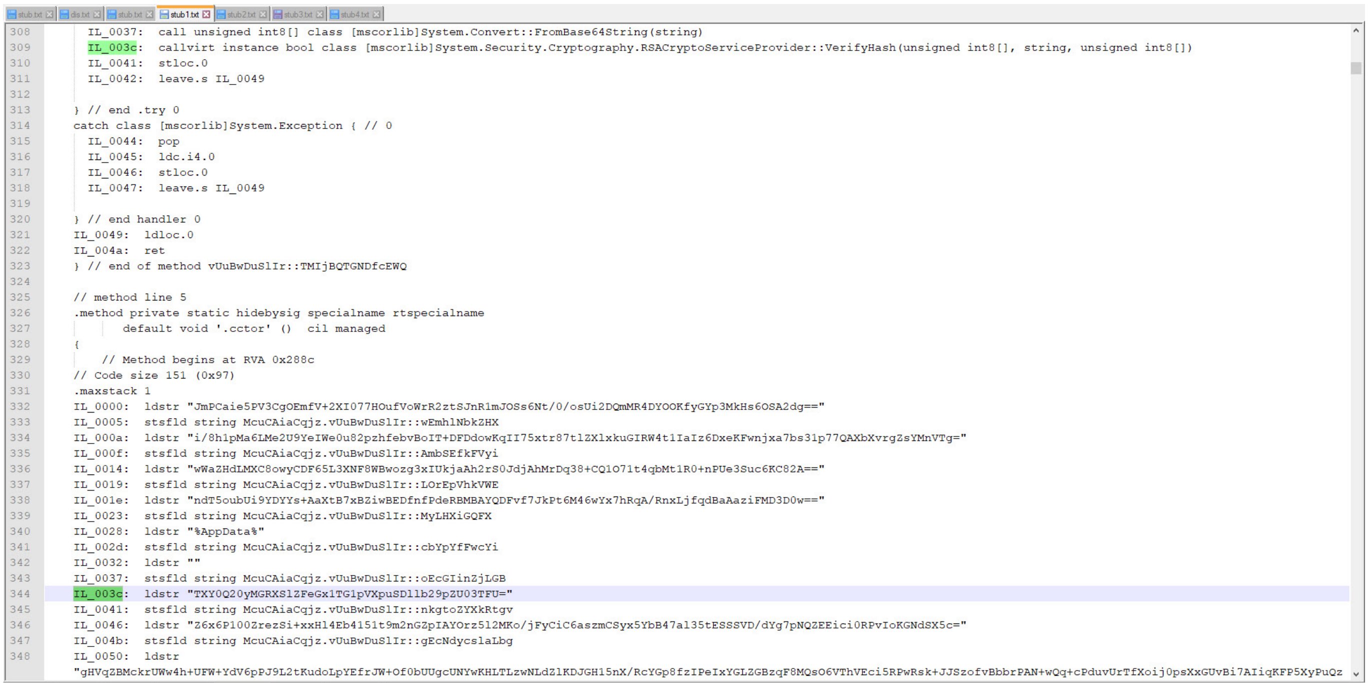


Name	Date modified	Type	Size
backup	18.06.2020 20:50	File folder	
Stub.exe	17.06.2020 21:30	Application	48 KB
stub.txt	18.06.2020 14:42	Text Document	287 KB
Stub1.exe	17.06.2020 21:30	Application	48 KB
stub1.txt	18.06.2020 20:55	Text Document	286 KB
Stub2.exe	17.06.2020 21:30	Application	48 KB
stub2.txt	18.06.2020 20:55	Text Document	284 KB
Stub3.exe	17.06.2020 21:30	Application	48 KB
stub3.txt	18.06.2020 20:55	Text Document	286 KB
Stub4.exe	17.06.2020 21:30	Application	48 KB
stub4.txt	18.06.2020 20:55	Text Document	286 KB
Stub5.exe	17.06.2020 21:30	Application	48 KB
stub5.txt	18.06.2020 20:55	Text Document	287 KB
Stub6.exe	17.06.2020 21:30	Application	48 KB
stub6.txt	18.06.2020 20:55	Text Document	283 KB
Stub7.exe	17.06.2020 21:30	Application	48 KB
stub7.txt	18.06.2020 20:55	Text Document	288 KB
Stub8.exe	17.06.2020 21:30	Application	48 KB
stub8.txt	18.06.2020 20:55	Text Document	286 KB
Stub9.exe	17.06.2020 21:30	Application	48 KB
stub9.txt	18.06.2020 20:55	Text Document	288 KB
Stub10.exe	17.06.2020 21:30	Application	48 KB
stub10.txt	18.06.2020 20:55	Text Document	283 KB
Stub11.exe	17.06.2020 21:30	Application	48 KB
stub11.txt	18.06.2020 20:55	Text Document	285 KB
Stub12.exe	17.06.2020 21:30	Application	48 KB
stub12.txt	18.06.2020 20:55	Text Document	286 KB
Stub13.exe	17.06.2020 21:30	Application	48 KB
stub13.txt	18.06.2020 20:55	Text Document	284 KB
Stub14.exe	17.06.2020 21:30	Application	48 KB
stub14.txt	18.06.2020 20:55	Text Document	284 KB
Stub15.exe	17.06.2020 21:30	Application	48 KB
stub15.txt	18.06.2020 20:55	Text Document	284 KB
Stub16.exe	17.06.2020 21:30	Application	48 KB
stub16.txt	18.06.2020 20:55	Text Document	286 KB

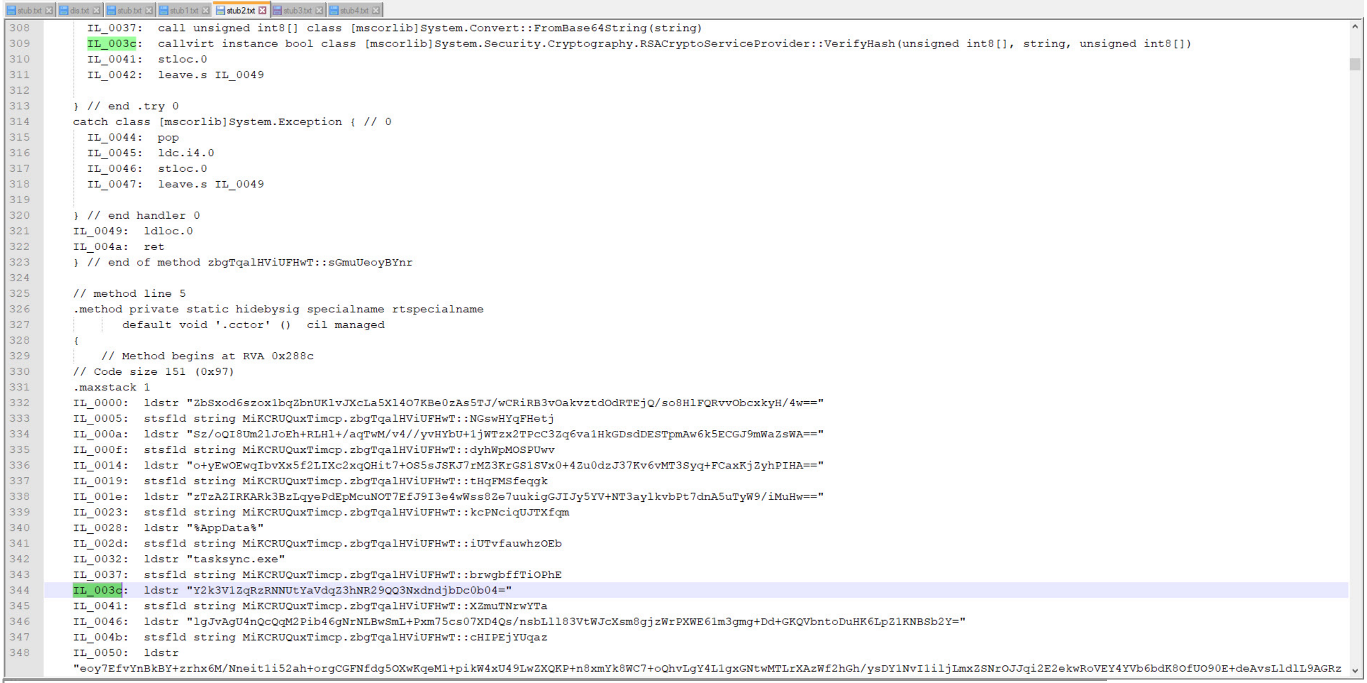
Administrator: C:\Windows\system32\cmd.exe

```
C:\Users\Mert\Desktop\vt_malwares>for /l %x in (1, 1, 28) do (
More?    "C:\Program Files\Mono\bin\monodis.exe" --output=stub%x.txt stub%x.exe
More? )

C:\Users\Mert\Desktop\vt_malwares>("C:\Program Files\Mono\bin\monodis.exe" --out
put=stub1.txt stub1.exe )

C:\Users\Mert\Desktop\vt_malwares>("C:\Program Files\Mono\bin\monodis.exe" --out
put=stub2.txt stub2.exe )

C:\Users\Mert\Desktop\vt_malwares>("C:\Program Files\Mono\bin\monodis.exe" --out
put=stub3.txt stub3.exe )

C:\Users\Mert\Desktop\vt_malwares>("C:\Program Files\Mono\bin\monodis.exe" --out
put=stub4.txt stub4.exe )

C:\Users\Mert\Desktop\vt_malwares>("C:\Program Files\Mono\bin\monodis.exe" --out
put=stub5.txt stub5.exe )

C:\Users\Mert\Desktop\vt_malwares>("C:\Program Files\Mono\bin\monodis.exe" --out
put=stub6.txt stub6.exe )

C:\Users\Mert\Desktop\vt_malwares>("C:\Program Files\Mono\bin\monodis.exe" --out
put=stub7.txt stub7.exe )
```

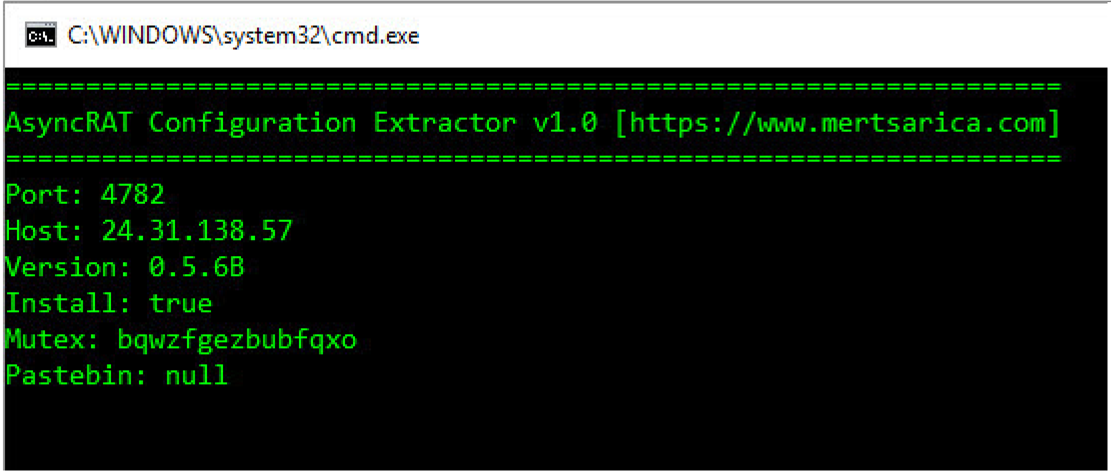
```
      IL_0037:  call unsigned int8[] class [mscorlib]System.Convert::FromBase64String(string)
      IL_003c:  callvirt instance bool class [mscorlib]System.Security.Cryptography.RSACryptoServiceProvider::VerifyHash(unsigned int8[], string, unsigned int8[])
      IL_0041:  stloc.0
      IL_0042:  leave.s IL_0049

    } // end .try 0
    catch class [mscorlib]System.Exception { // 0
      IL_0044:  pop
      IL_0045:  ldc.i4.0
      IL_0046:  stloc.0
      IL_0047:  leave.s IL_0049

    } // end handler 0
    IL_0049:  ldloc.0
    IL_004a:  ret
    } // end of method UPjlGoQHGJmz::ZoFvYnkytxxLlEF

    // method line 5
    .method private static hidebysig specialname rtspecialname
           default void '.cctor' ()  cil managed
    {
        // Method begins at RVA 0x288c
    // Code size 151 (0x97)
    .maxstack 1
    IL_0000:  ldstr "+ZOybq/fGZAPhtjZnpOMXGLgsfwrXKU5S83HK8xUjlQdiYXjGu42snGzR4PEwQBLMtP6HY0tO4e7fytfIUp+9Q=="
    IL_0005:  stsfld string LVefMHugJpKfAW.UPjlGoQHGJmz::rLxHRVvktJ
    IL_000a:  ldstr "iuq7GHKbtQ0L0SEwn9VUWoiYE1NiOums33crjUxW593CxiMMqnpb3WgXcPOVWTMl99RmnGMwjEJTORok7zaxPA=="
    IL_000f:  stsfld string LVefMHugJpKfAW.UPjlGoQHGJmz::svdTwYYxvrD
    IL_0014:  ldstr "rFw0jQKqdu9pvq855FnO7awWPcX0InCNuZoHPejjWBIO21+9XMo59v72jmajGMP8/IttgUWm/1xu7r7S3NE87w=="
    IL_0019:  stsfld string LVefMHugJpKfAW.UPjlGoQHGJmz::JqTYyBvoPQXlKmBT
    IL_001e:  ldstr "c3R2LaC//AaIuPpNsvyq1uG2GZ0k6Q5Dq5vSkwkSzlvPHCAhMhbLtGerkeQCNFdQDkik/zCvKBd5jDPRxtKvqw=="
    IL_0023:  stsfld string LVefMHugJpKfAW.UPjlGoQHGJmz::WtxJdUcxhhTm
    IL_0028:  ldstr "%AppData%"
    IL_002d:  stsfld string LVefMHugJpKfAW.UPjlGoQHGJmz::RifTFGCauJoM
    IL_0032:  ldstr "Host Process for Windows Host.exe"
    IL_0037:  stsfld string LVefMHugJpKfAW.UPjlGoQHGJmz::IgQueznoVept
    IL_003c:  ldstr "cG5tSzZuUFlQSUlnNU9Ic3A5NjFkbk5QVXlJYmRJWWk="
    IL_0041:  stsfld string LVefMHugJpKfAW.UPjlGoQHGJmz::bPwIENFAunePP
    IL_0046:  ldstr "wW+9tAG5gSnOnDZHUUFzEP+Wk+F7BiDKG3t7cWs9tEb6DyScZCw9gMlW/PAA6B6BMkI2qi6c1SAGX+jtAhpGiw=="
    IL_004b:  stsfld string LVefMHugJpKfAW.UPjlGoQHGJmz::BsgvWQPUlSxB
    IL_0050:  ldstr
    "ufKCuObOYG+E1IE1XqM+sKgnW/TvtYk6KqIv9V9CGtXPfOJUVOSKCwtzgbiAkY+M0yNQ4He2DZSp2LVhhhWuiiuLymnrcpU4jcgH/HYvjYatoimWOcRF6gOXBP8QvRukan0yOS2FakIpOqdgmG1CIwfq/RJV8+ooX7aWJkCo7+mT6jjIry
```



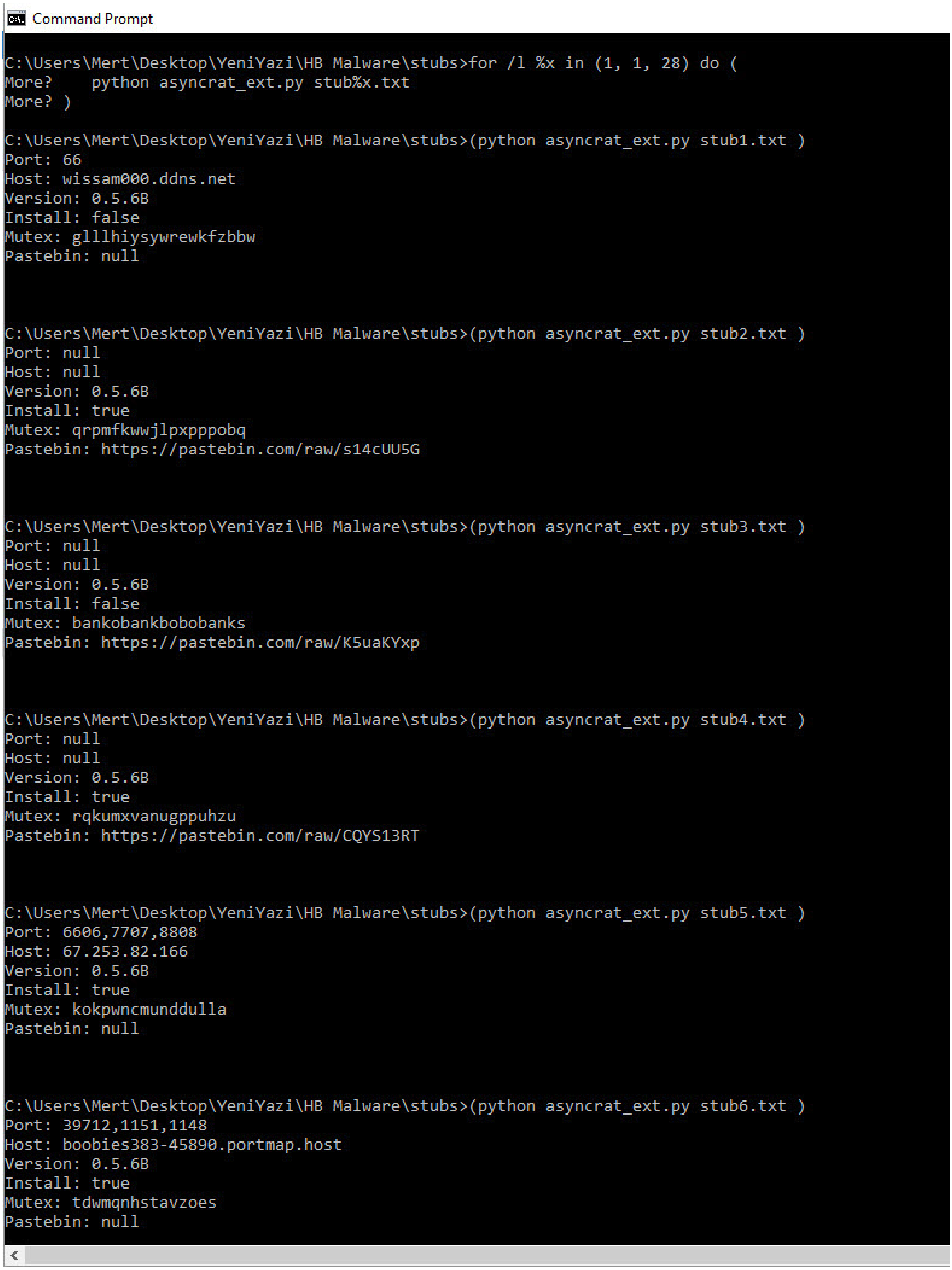
```
  IL_0037:  call unsigned int8[] class [mscorlib]System.Convert::FromBase64String(string)
  IL_003c:  callvirt instance bool class [mscorlib]System.Security.Cryptography.RSACryptoServiceProvider::VerifyHash(unsigned int8[], string, unsigned int8[])
  IL_0041:  stloc.0
  IL_0042:  leave.s IL_0049

} // end .try 0
catch class [mscorlib]System.Exception { // 0
  IL_0044:  pop
  IL_0045:  ldc.i4.0
  IL_0046:  stloc.0
  IL_0047:  leave.s IL_0049

} // end handler 0
IL_0049:  ldloc.0
IL_004a:  ret
} // end of method vUuBwDuSlIr::TMIjBQTGNDfcEWQ

// method line 5
.method private static hidebysig specialname rtspecialname
        default void '.cctor' ()  cil managed
{
    // Method begins at RVA 0x288c
// Code size 151 (0x97)
.maxstack 1
IL_0000:  ldstr "JmPCaie5PV3CgOEmfV+2XI077HOufVoWrR2ztSJnR1mJOSs6Nt/0/osUi2DQmMR4DYOOKfyGYp3MkHs6OSA2dg=="
IL_0005:  stsfld string McuCAiaCqjz.vUuBwDuSlIr::wEmhlNbkZHX
IL_000a:  ldstr "i/8h1pMa6LMe2U9YeIWe0u82pzhfebvBoIT+DFDdowKqII75xtr87tlZXlxkuGIRW4t1IaIz6DxeKFwnjxa7bs31p77QAXbXvrgZsYMnVTg="
IL_000f:  stsfld string McuCAiaCqjz.vUuBwDuSlIr::AmbSEfkFVyi
IL_0014:  ldstr "wWaZHdLMXC8owyCDF65L3XNF8WBwozg3xIUkjaAh2rS0JdjAhMrDq38+CQ1O71t4qbMt1R0+nPUe3Suc6KC82A=="
IL_0019:  stsfld string McuCAiaCqjz.vUuBwDuSlIr::LOrEpVhkVWE
IL_001e:  ldstr "ndT5oubUi9YDYYs+AaXtB7xBZiwBEDfnfPdeRBMBAYQDFvf7JkPt6M46wYx7hRqA/RnxLjfqdBaAaziFMD3D0w=="
IL_0023:  stsfld string McuCAiaCqjz.vUuBwDuSlIr::MyLHXiGQFX
IL_0028:  ldstr "%AppData%"
IL_002d:  stsfld string McuCAiaCqjz.vUuBwDuSlIr::cbYpYfFwcYi
IL_0032:  ldstr ""
IL_0037:  stsfld string McuCAiaCqjz.vUuBwDuSlIr::oEcGIinZjLGB
IL_003c:  ldstr "TXY0Q20yMGRXSlZFeGx1TG1pVXpuSDllb29pZU03TFU="
IL_0041:  stsfld string McuCAiaCqjz.vUuBwDuSlIr::nkgtoZYXkRtgv
IL_0046:  ldstr "Z6x6P100ZrezSi+xxHl4Eb4151t9m2nGZpIAYOrz512MKo/jFyCiC6aszmCSyx5YbB47al35tESSSVD/dYg7pNQZEEici0RPvIoKGNdSX5c="
IL_004b:  stsfld string McuCAiaCqjz.vUuBwDuSlIr::gEcNdycslaLbg
IL_0050:  ldstr
"gHVqZBMckrUWw4h+UFW+YdV6pPJ9L2tKudoLpYEfrJW+Of0bUUgcUNYwKHLTLzwNLdZlKDJGH15nX/RcYGp8fzIPeIxYGLZGBzqF8MQsO6VThVEci5RPwRsk+JJSzofvBbbrPAN+wQq+cPduvUrTfXoij0psXxGUvBi7AIiqKFP5XyPuQz
```



```
  IL_0037:  call unsigned int8[] class [mscorlib]System.Convert::FromBase64String(string)
  IL_003c:  callvirt instance bool class [mscorlib]System.Security.Cryptography.RSACryptoServiceProvider::VerifyHash(unsigned int8[], string, unsigned int8[])
  IL_0041:  stloc.0
  IL_0042:  leave.s IL_0049

} // end .try 0
catch class [mscorlib]System.Exception { // 0
  IL_0044:  pop
  IL_0045:  ldc.i4.0
  IL_0046:  stloc.0
  IL_0047:  leave.s IL_0049

} // end handler 0
IL_0049:  ldloc.0
IL_004a:  ret
} // end of method zbgTqalHViUFHwT::sGmuUeoyBYnr

// method line 5
.method private static hidebysig specialname rtspecialname
        default void '.cctor' ()  cil managed
{
    // Method begins at RVA 0x288c
// Code size 151 (0x97)
.maxstack 1
IL_0000:  ldstr "ZbSxod6szox1bqZbnUKlvJXcLa5Xl4O7KBe0zAs5TJ/wCRiRB3vOakvztdOdRTEjQ/so8HlFQRvvObcxkyH/4w=="
IL_0005:  stsfld string MiKCRUQuxTimcp.zbgTqalHViUFHwT::NGswHYqFHetj
IL_000a:  ldstr "Sz/oQI8Um2lJoEh+RLHl+/aqTwM/v4//yvHYbU+1jWTzx2TPcC3Zq6va1HkGDsdDESTpmAw6k5ECGJ9mWaZsWA=="
IL_000f:  stsfld string MiKCRUQuxTimcp.zbgTqalHViUFHwT::dyhWpMOSPUwv
IL_0014:  ldstr "o+yEwOEwqIbvXx5f2LIXc2xqQHit7+OS5sJSKJ7rMZ3KrGS1SVx0+4Zu0dzJ37Kv6vMT3Syq+FCaxKjZyhPIHA=="
IL_0019:  stsfld string MiKCRUQuxTimcp.zbgTqalHViUFHwT::tHqFMSfeqgk
IL_001e:  ldstr "zTzAZIRKARk3BzLqyePdEpMcuNOT7EfJ9I3e4wWss8Ze7uukigGJIJy5YV+NT3aylkvbPt7dnA5uTyW9/iMuHw=="
IL_0023:  stsfld string MiKCRUQuxTimcp.zbgTqalHViUFHwT::kcPNciqUJTXfqm
IL_0028:  ldstr "%AppData%"
IL_002d:  stsfld string MiKCRUQuxTimcp.zbgTqalHViUFHwT::iUTvfauwhzOEb
IL_0032:  ldstr "tasksync.exe"
IL_0037:  stsfld string MiKCRUQuxTimcp.zbgTqalHViUFHwT::brwgbffTiOPhE
IL_003c:  ldstr "Y2k3V1ZqRzRNNUtYaVdqZ3hNR29QQ3NxdndjbDc0b04="
IL_0041:  stsfld string MiKCRUQuxTimcp.zbgTqalHViUFHwT::XZmuTNrwYTa
IL_0046:  ldstr "lgJvAgU4nQcQqM2Pib46gNrNLBwSmL+Pxm75cs07XD4Qs/nsbLl183VtWJcXsm8gjzWrPXWE61m3gmg+Dd+GKQVbntoDuHK6LpZ1KNBSb2Y="
IL_004b:  stsfld string MiKCRUQuxTimcp.zbgTqalHViUFHwT::cHIPEjYUqaz
IL_0050:  ldstr
"eoy7EfvYnBkBY+zrhx6M/Nneit1i52ah+orgCGFNfdg5OXwKqeM1+pikW4xU49LwZXQKP+n8xmYk8WC7+oQhvLgY4L1gxGNtwMTLrXAzWf2hGh/ysDY1NvI1iljLmxZSNrOJJqi2E2ekwRoVEY4YVb6bdK8OfUO90E+deAvsLldlL9AGRz
```



```
============================================================
AsyncRAT Configuration Extractor v1.0 [https://www.mertsarica.com]
============================================================
Port: 4782
Host: 24.31.138.57
Version: 0.5.6B
Install: true
Mutex: bqwzfgezbubfqxo
Pastebin: null
```



```
C:\Users\Mert\Desktop\YeniYazi\HB Malware\stubs>for /l %x in (1, 1, 28) do (
More?    python asyncrat_ext.py stub%x.txt
More? )

C:\Users\Mert\Desktop\YeniYazi\HB Malware\stubs>(python asyncrat_ext.py stub1.txt )
Port: 66
Host: wissam000.ddns.net
Version: 0.5.6B
Install: false
Mutex: glllhiysywrewkfzbbw
Pastebin: null



C:\Users\Mert\Desktop\YeniYazi\HB Malware\stubs>(python asyncrat_ext.py stub2.txt )
Port: null
Host: null
Version: 0.5.6B
Install: true
Mutex: qrpmfkwwjlpxpppobq
Pastebin: https://pastebin.com/raw/s14cUU5G



C:\Users\Mert\Desktop\YeniYazi\HB Malware\stubs>(python asyncrat_ext.py stub3.txt )
Port: null
Host: null
Version: 0.5.6B
Install: false
Mutex: bankobankbobobanks
Pastebin: https://pastebin.com/raw/K5uaKYxp



C:\Users\Mert\Desktop\YeniYazi\HB Malware\stubs>(python asyncrat_ext.py stub4.txt )
Port: null
Host: null
Version: 0.5.6B
Install: true
Mutex: rqkumxvanugppuhzu
Pastebin: https://pastebin.com/raw/CQYS13RT



C:\Users\Mert\Desktop\YeniYazi\HB Malware\stubs>(python asyncrat_ext.py stub5.txt )
Port: 6606,7707,8808
Host: 67.253.82.166
Version: 0.5.6B
Install: true
Mutex: kokpwncmunddulla
Pastebin: null



C:\Users\Mert\Desktop\YeniYazi\HB Malware\stubs>(python asyncrat_ext.py stub6.txt )
Port: 39712,1151,1148
Host: boobies383-45890.portmap.host
Version: 0.5.6B
Install: true
Mutex: tdwmqnhstavzoes
Pastebin: null
```

In conclusion, after compiling and collecting all this information, it appears that while this cyber attack attempt is not an APT attack, it is part of a targeted attack (Spear Phishing). Especially in light of the increase in such targeted cyber attack attempts after the Covid-19 pandemic, I recommend that organizations and employees be very careful.

Hope to see you in the following articles.