Water Orthrus New Campaigns Deliver Rootkit and Phishing Modules

By By: Jaromir Horejsi, Joseph C Chen May 15, 2023 Read time: 9 min (2419 words)

Published: 2023-05-15 · Archived: 2026-04-05 14:44:04 UTC

Malware

Water Orthrus has been active recently with two new campaigns. CopperStealth uses a rootkit to install malware on infected systems, while CopperPhish steals credit card information. This blog will provide the structure of the campaign and how they work.

Since 2021, we have been tracking the activities of a threat actor we called Water Orthrus, which distributed CopperStealer malware via pay-per-install (PPI) networks. The threat actor has upgraded and modified the malware multiple times for different purposes, such as injecting network advertisements, acquiring personal information, and stealing cryptocurrency. We believe that they are associated with the threat campaign reported as "Scranosopen on a new tab" in 2019.

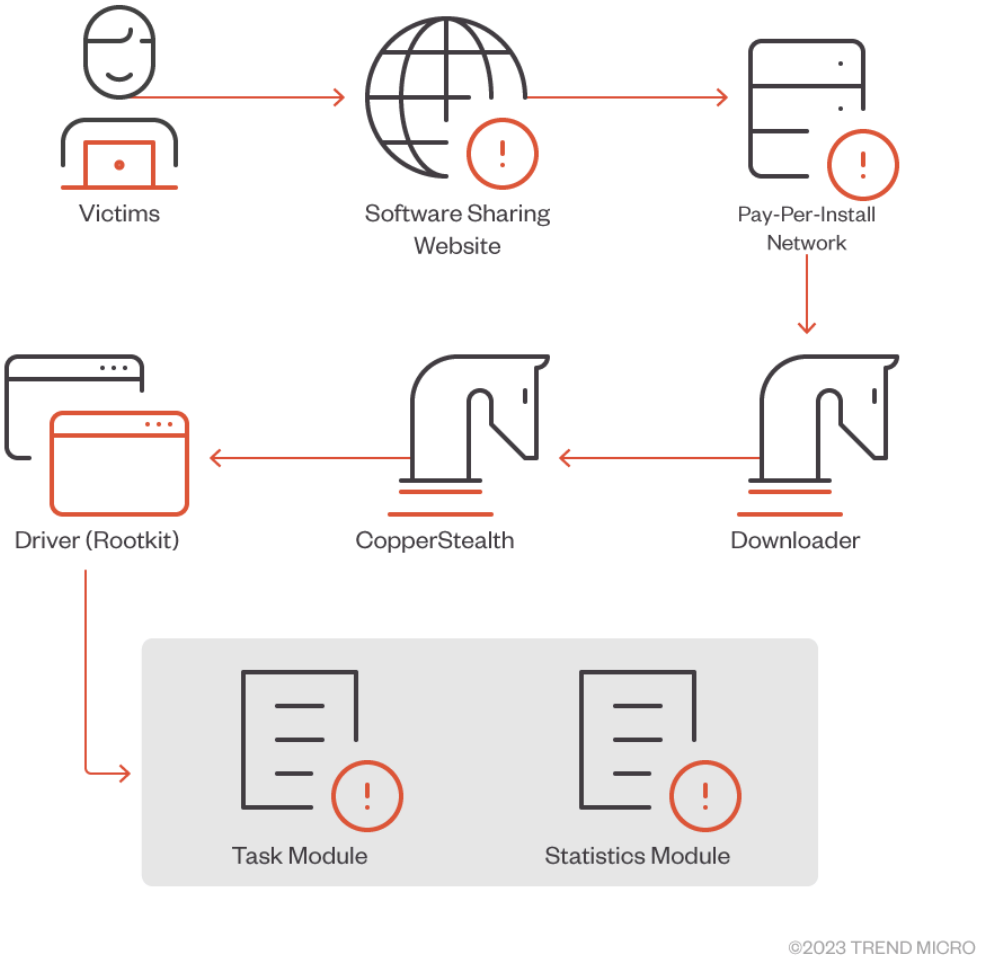

Figure 1. CopperStealth infection chain

In March 2023, we observed two campaigns delivering new malware that we named CopperStealth and CopperPhish. Both malware have characteristics that are similar to those of CopperStealer and are likely developed by the same author, leading us to believe that these campaigns are likely Water Orthrus' new activities.

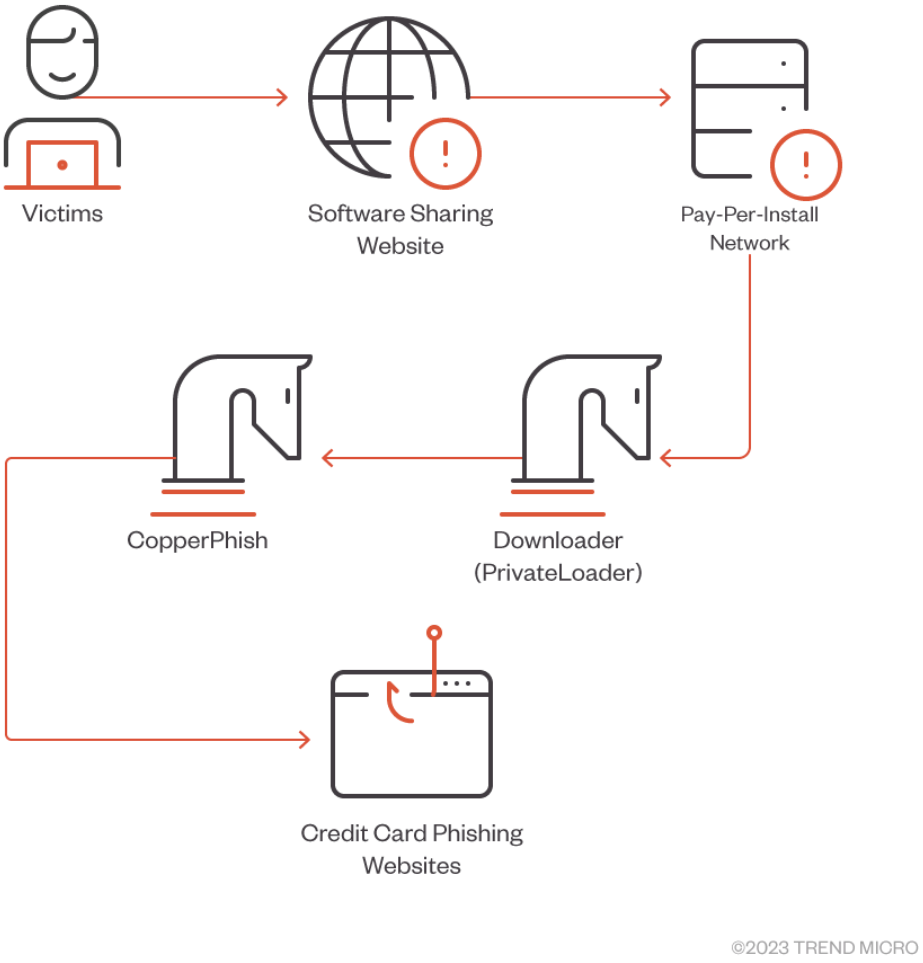

Figure 2. CopperPhish infection chain

This blog post discusses our analysis of CopperStealth's and CopperPhish's infection chains, and how they are similar to Water Orthrus.

CopperStealth campaign

The first campaign distributed CopperStealth on March 8, 2023, delivering the malware via installers provided on a popular Chinese software sharing website. It disguised the malware as free software and targeted the country's users.

CopperStealth's infection chain involves dropping and loading a rootkit, which later injects its payload into explorer.exe and another system process. These payloads are responsible for downloading and running additional tasks. The rootkit also blocks access to blocklisted registry keys and prevents certain executables and drivers from running.

A sample of the installer (SHA-256: *8a21eae144a23fffd35f8714964ff316caaa37fe464e8bbc143f4485119b5575*) was distributed via a popular Chinese software download website. It was previously reportedopen on a new tab to

provide malicious installers for infecting malware. The installer contains numerous encoded URLs; when the installer is run these URLs will be decoded and the files located at the said URLs will be downloaded and run on the affected system. One of these files was a dropper that we identified as CopperStealth.

CopperStealth (after decrypting the custom crypter) contains an export function called "HelloWorld," which has been around since earlier versions of CopperStealer. The dropper then checks if the system architecture is 32-bit or 64-bit, which it uses as a basis to read the corresponding driver from its resources.

```
strcat_ex(szDriverFilePath, "\\system32\\drivers\\");
szMiddlePartOfHash = (char *)deref_string(v12);
strcat_ex(szDriverFilePath, szMiddlePartOfHash);
strcat_ex(szDriverFilePath, ".sys");
Block = 0;
nNumberOfBytesToWrite = 0;
if ( bIs64bit )
  lpType = "CUSTOM1";
else
  lpType = "CUSTOM2";
```

Figure 3. CopperStealth's code for selecting 32-bit or 64-bit drivers

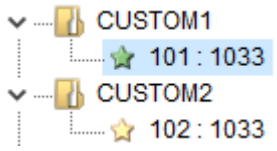

Figure 4. CopperStealth's resources contain two versions of the driver

```
hSCObject = CreateServiceA(
              hSCManager,
              lpServiceName,
              DisplayName,
              0xF01FFu,
              SERVICE_KERNEL_DRIVER,
              SERVICE_SYSTEM_START,
              SERVICE_ERROR_NORMAL,
              lpBinaryPathName,
              0,
              0,
              0,
              0,
              0);
```

Figure 5. The new driver service created by CopperStealth

Rootkit

The rootkit is registered as a file system filter driver. In the main entry point (DriverEntryopen on a new tab function), the driver has specific handlers for IRP_MJ_CREATEopen on a new tab, IRP_MJ_READopen on a new tab, and IRP_MJ_SHUTDOWNopen on a new tab.

When a shutdown request is received, the rootkit driver copies itself to a newly generated random driver. This is done with a few registry modifications, such as changing the driver name in the *HKLM\SYSTEM\CurrentControlSet\Services* registry and inserting a "PendingFileRenameOperations" registry

value in *HKLM\SYSTEM\CurrentControlSet\Control\SessionManager* to automatically move the file upon rebootopen on a new tab.

As for IRP_MJ_READ, this monitors any attempts to read the files on filesystem. If any attempt to read the current rootkit's SYS file or executable file in *\windows\temp* folder by one of the blocklisted processes (such as *\360saf*, *\kingsoft antivirus*, *\360SD*, *\Windows Defender*) is detected, the driver will result in a STATUS_ACCESS_DENIED error message, preventing access to those files.

The rootkit then registers a registry callback routineopen on a new tab, which is called every time a thread performs an operation in the registry. In this case, the routine monitors attempts to access rootkit's registry path in *HKLM\SYSTEM\CurrentControlSet\Services\<drivername>*. Any attempt to delete the registry key (such as RegNtDeleteKeyopen on a new tab OR RegNtDeleteValueKeyopen on a new tab) will result in STATUS_ACCESS_DENIED, blocking the operation.

An attempt to query the registry key (RegNtQueryKeyopen on a new tab OR RegNtQueryValueKeyopen on a new tab) while any of the blocklisted processes (*360safe.exe*, *360sd.exe*, *QQPCTray.exe*, and *kxetray.exe*) is running will also result in a STATUS_ACCESS_DENIED message.

```
else if ( Argument1 == RegNtQueryKey || Argument1 == RegNtQueryValueKey )
{
  unicode_string_copy(&v5, 0, *Argument2_registryobject);
  if ( unicode_string_contains(&v5, wcs_services_selfName) )
  {
    if ( are_blocklisted_processes_running() )
      v3 = STATUS_ACCESS_DENIED;
  }
```

Figure 6. Rootkit's code to block querying it's (rootkit's) own registry key, if any of the blocklisted processes is running

The rootkit also sets an image load notification routineopen on a new tab, which is called whenever a new image (i.e., an executable/binary file, like .EXE, .DLL, and .SYS files) is loaded into memory. If an "image" name corresponds to the hardcoded filename *(\Fix\fixMBR.exe*), then a DLL library (embedded inside the driver) is mapped and run into the newly created process. The 64-bit version of the rootkit contains a 32-bit and a 64-bit version of the DLL library.

```
if ( peMachine == IMAGE_FILE_MACHINE_I386 )
{
  inject_into_32bit_process(v6, PEProcess);
}
else if ( peMachine == IMAGE_FILE_MACHINE_IA64 || peMachine == IMAGE_FILE_MACHINE_AMD64 )
{
  inject_into_64bit_process(v6, PEProcess);
}
```

Figure 7. Rootkit's code that terminates newly executed processes by injected DLL

```
BOOL __stdcall DllMain(HINSTANCE hinstDLL, DWORD fdwReason, LPVOID lpvReserved)
{
  if ( fdwReason == DLL_PROCESS_ATTACH )
    ExitProcess(0);
  return 1;
}
```

Figure 8. Termination DLL exits process

If the newly loaded process is a driver, then the rootkit searches for a few blocklisted byte sequences. If any of those are found, then the driver's entry point is patched to return STATUS_ACCESS_DENIED.

```
return_status_access_denied proc near   ; DATA XREF: patch_entrypoint+17↑o
                mov     eax, STATUS_ACCESS_DENIED
                retn    8
return_status_access_denied endp
```

Figure 9. Rootkit replaces blocklisted drivers' entrypoints to return error code

The blocklisted drivers contain one of the following byte sequences:

- Beijing Huorong Network Technology Co., Ltd.
- Beijing Kingsoft Security Software Co., Ltd.
- Beijing Qihu Technology Co., Ltd.
- HuoRongBoRui (Beijing) Technology Co., Ltd.
- Qihoo 360 Software (Beijing) Co., Ltd.

Lastly, the rootkit starts a payload injection thread. In the samples we analyzed, we saw two types of payloads: statistics module and task module. This thread enumerates all running processes (SystemProcessInformationopen on a new tab) and looks for *explorer.exe* and another process with SYSTEM integrity having "assign primary token privilegeopen on a new tab" and "increase quota privilegeopen on a new tab." It also increments the *HKLM\SOFTWARE\Microsoft\recount* registry value, which holds the number of system restarts since the machine became infected. If the restart recount value is greater than three, the rootkit decrypts (AES cipher) and unpacks (7-Zip) embedded statistics in the DLL module, then it patches its statistics URL address inside the binary (placeholder starting with "cnzz_url"), and finally injects the patched module into *explorer.exe*.

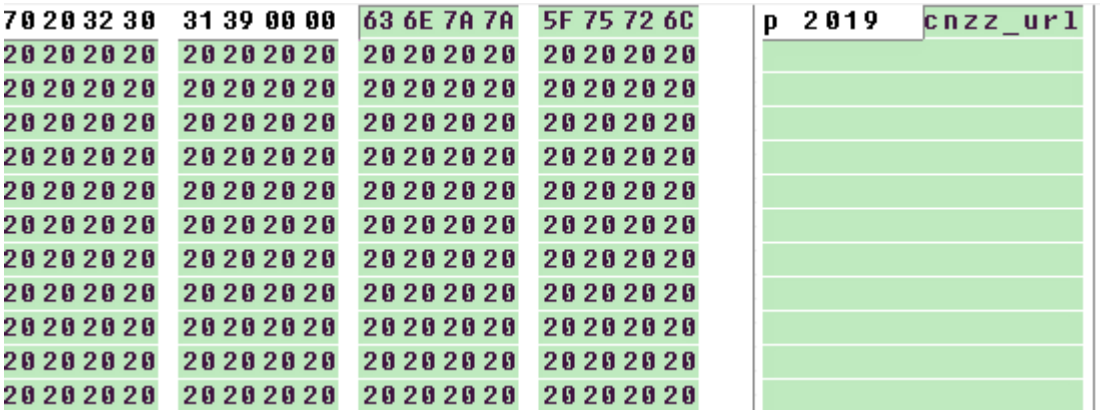

Figure 10. Placeholder inside the statistics module, which the rootkit replaces with its statistics URL

After successful injection into *explorer.exe*, it makes a GET request to the task URL to get the task command to be performed. Once the GET request is done by the driver, then the response to the GET request is then inserted (a place holder starting with “searching_magic_url”) into the task module, which, like the statistics module, is embedded in rootkit. It must be AES-encrypted, and then 7-Zip unpacked.And then the task module is injected into the previously found process with system integrity.

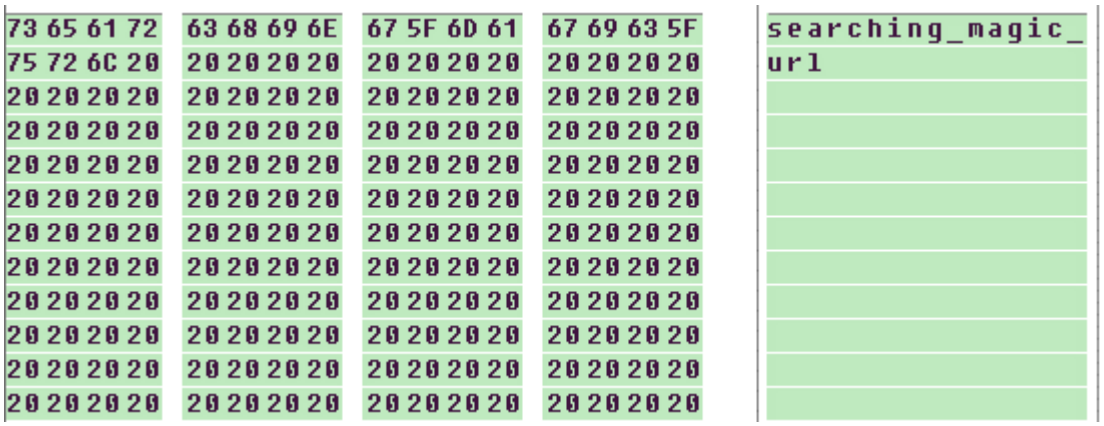

Figure 11. Placeholder inside the task module, which the rootkit replaces with a response to the task URL

Statistics module

The statistics module is called cnzzopen on a new tab (the largest Chinese internet statistics analysis service), though it has nothing to do with this organization. Each time it is run it increases the counter in *HKCU\Software\Microsoft\count_a0b1c2d3*. It checks the internet connection by trying to access *hxxp://www.msftconnecttest.com/connecttest.txt*, and once verified it will get the computer’s unique machine ID and report the statistics to *hxxp://cnzz_url&m=<machine ID>* then exits the module.

In some cases, cnzz_url contains keyword “tongji” (tǒng jì, 統open on a new tab計open on a new tab), which translates to “statistics.”

Task module

The task module is called curl, even though it has nothing to do with curlopen on a new tab (command line tool for transferring data with URLs). The module then proceeds with base64 decoding and DES decrypting (key=”taskhost”, IV=”winlogon”, which is the same encryption as described in previous blog posts) of the task command. The decrypted task is in JSONopen on a new tab format and has the following keys:

Name	Type	Explanation
name	string	File name created in TEMP directory
onlyone	bool	Run only once
exclude360	bool	Exclude machines where 360tray is running
reboot_count	int	The task should be run every n-th reboot of the system

url	string	URL with the file to download and execute

Table 1. List of implemented key names and value types in task query response (in JSON format)

If payload is expected to run only once (with onlyone [sic] key has value = true), then the hash of the URL is stored in *HKLM\Software\Microsoft\<hash>*. The actual count of runs is taken from *HKEY_USERS\<SID>\Software\Microsoft\count_a0b1c2d3*, which is set by the statistics module.

CopperPhish campaign

In April 2023, we noticed another campaign distributing CopperPhish. This campaign was not geofenced and delivered the malware via PPI networks behind free anonymous file sharing websites. CopperPhish is an interesting phishing kit, which uses two different processes for persistence: credential verification and confirmation code to ensure that valid credentials are phished before the phishing kits considers its job successful and exits.

CopperPhish's infection chain starts with downloaders like PrivateLoader (SHA-256: 48211c6f957c2ad024441be3fc32aecd7c317dfc92523b0a675c0cfec86ffdd9). Visitors will be redirected to a download page designed by the PPI network after clicking on its advertisements, which pretended to be a download link. The downloaded file is PrivateLoader, which downloads and runs many different malware.

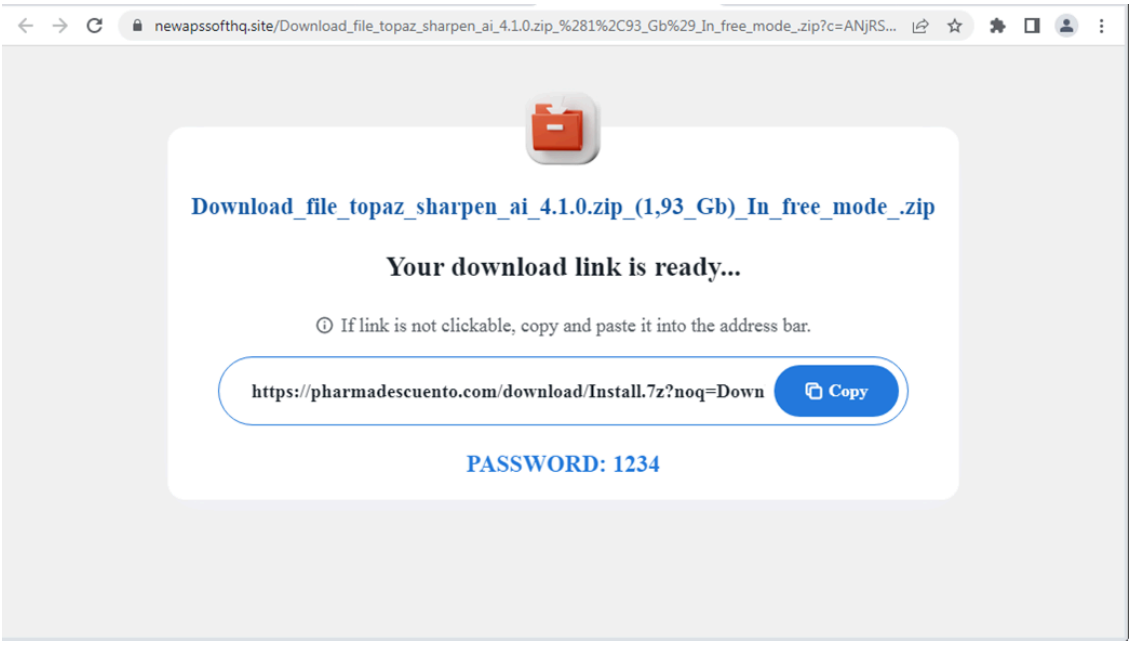

Figure 12. The download page of the pay-per-install network redirected from file sharing websites.

The downloader downloads and starts a new dropper we identified as CopperPhish. It decrypts and loads the second stage. This stage is responsible for dropping and starting the main payload. The main payload is again obfuscated with the same crypter used by CopperStealer, and then the second stage of the main payload drops a few files (PNG image with logo, PNG image with QR code, and HTML page with phishing URL) into *%APPDATA%\Roaming\Microsoft*.

Figure 13. Dropped phishing files by CopperPhish

These files are stored in the main payload's second stage's resources. These HTML pages are localized to fifty different languages. Each language has its own HTML file, but since PNG images do not contain any text, they remain the same for all language versions.

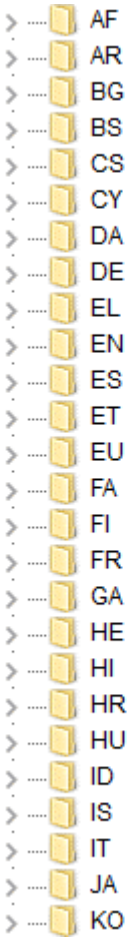

Figure 14. Phishing webpages in different language variants, stored in resources

The main malware then starts the two threads — the persistence thread is responsible for starting a rundll32 process and injecting a simple program with a browser window (written in Visual Basic) in it.

```
strcat_s(Buffer, 0x104u, "\\rundll32.exe");
while ( 1 )
{
  if ( !FindWindowA("ThunderRT6FormDC", "Operators Network Error Check") )
    inject_vb_webbrowser_process(Buffer);
  Sleep(0x2710u);
}
```

Figure 15. The thread responsible for starting processes with a web browser window

The exit message thread connects to a named pipe and waits for a message containing a magic string to be received. If the message is received, then the main program uninstalls itself (and deletes all the dropped phishing files) and exits. This indicates that the confirmation code (more on this code later) entered by the victim was correct, which means that the phishing was successful and it can uninstall itself.

```
hPipe = CreateNamedPipeA(
          "\\\\.\\pipe\\microsoft_shutdown_network",
          0x40000003u,
          6u,
          0xFFu,
          0x1000u,
          0x1000u,
          0,
          &SecurityAttributes);
```

Figure 16. Opened pipe that waits for a message to close and uninstall CopperPhish

```
if ( strstr(Buffer, "MAGIC_START") )
{
  if ( strstr(Buffer, "MAGIC_OVER") )
    uninstall_self();
}
```

Figure 17. Code that uninstalls the phishing tool after receiving magic values

The web browser program loads an instance of an Internet Explorer objectopen on a new tab and uses the web browser control class (SHDocVwCtl) to control the behavior of the web browser object. It reads the valueopen on a new tab written by the victim of the input text field called “checkcode” (which will be explained later). If correct, it then sends the exit message to the main payload via the pipes mentioned earlier.

```
text "UTF-16LE", 'MAGIC_STARTHelloWorlMAGIC_OVER',0
```

Figure 18. Exit message sent to the main payload via pipe

The phishing webpage displayed by the web browser shows the content of the dropped page with the Microsoft logo and QR code. The window has no controls that can be used to minimize or close it. The victim could close the browser’s process in Task Manager or Process Explorer, but they would also need to terminate the main payload process, otherwise the browser process will happen again due to the persistence thread. Thus, to proceed, a confirmation code will be asked (which was stated after Figure 15), and this code will act as proof that the victim provided the correct details.

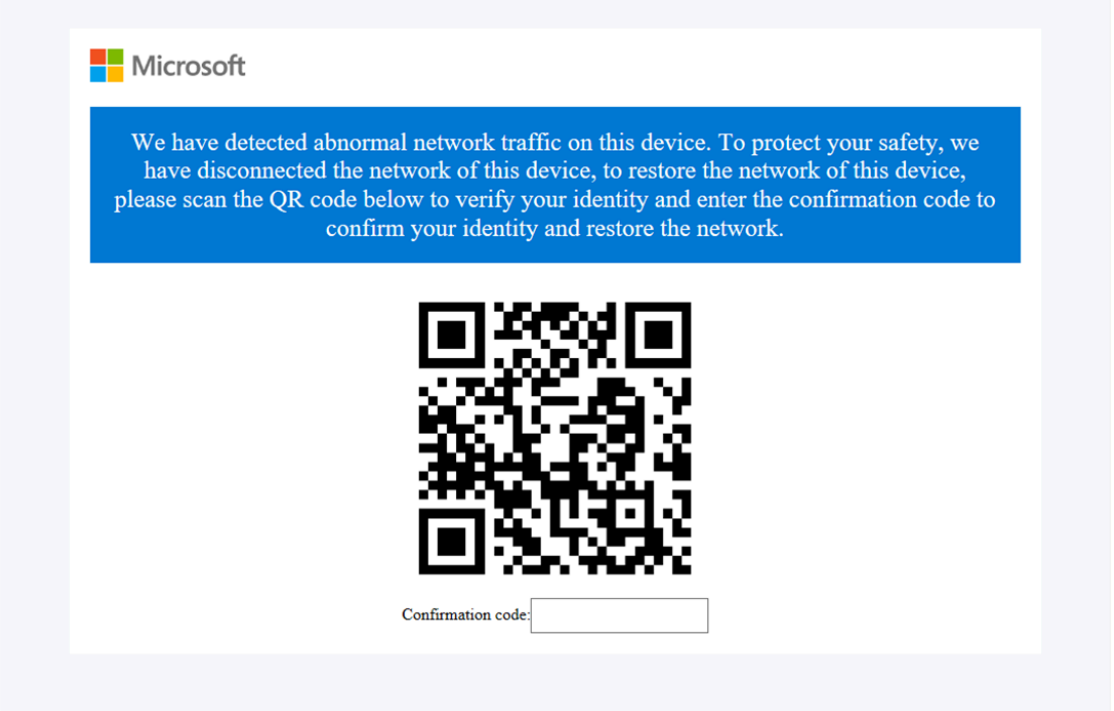

Figure 19. Initial phishing webpage displayed by CopperPhish

After scanning and opening the phishing URL, the victim is presented with a webpage asking to confirm the identity.

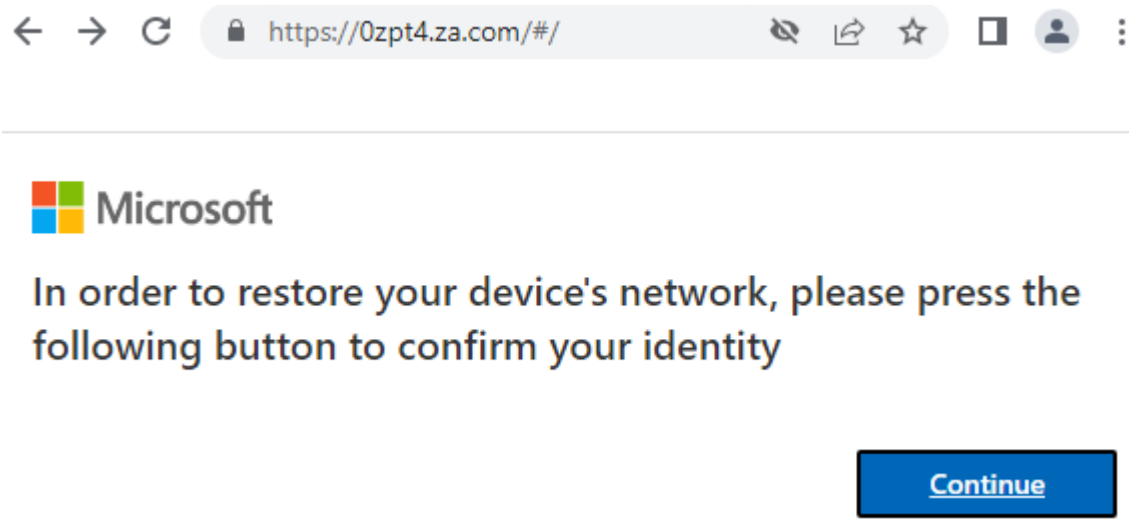

Figure 20. Phishing webpage asking to confirm the user’s identity

A follow-up webpage asks for various sensitive details, such as the user’s credit card number, its expiration date, and the CVV code.

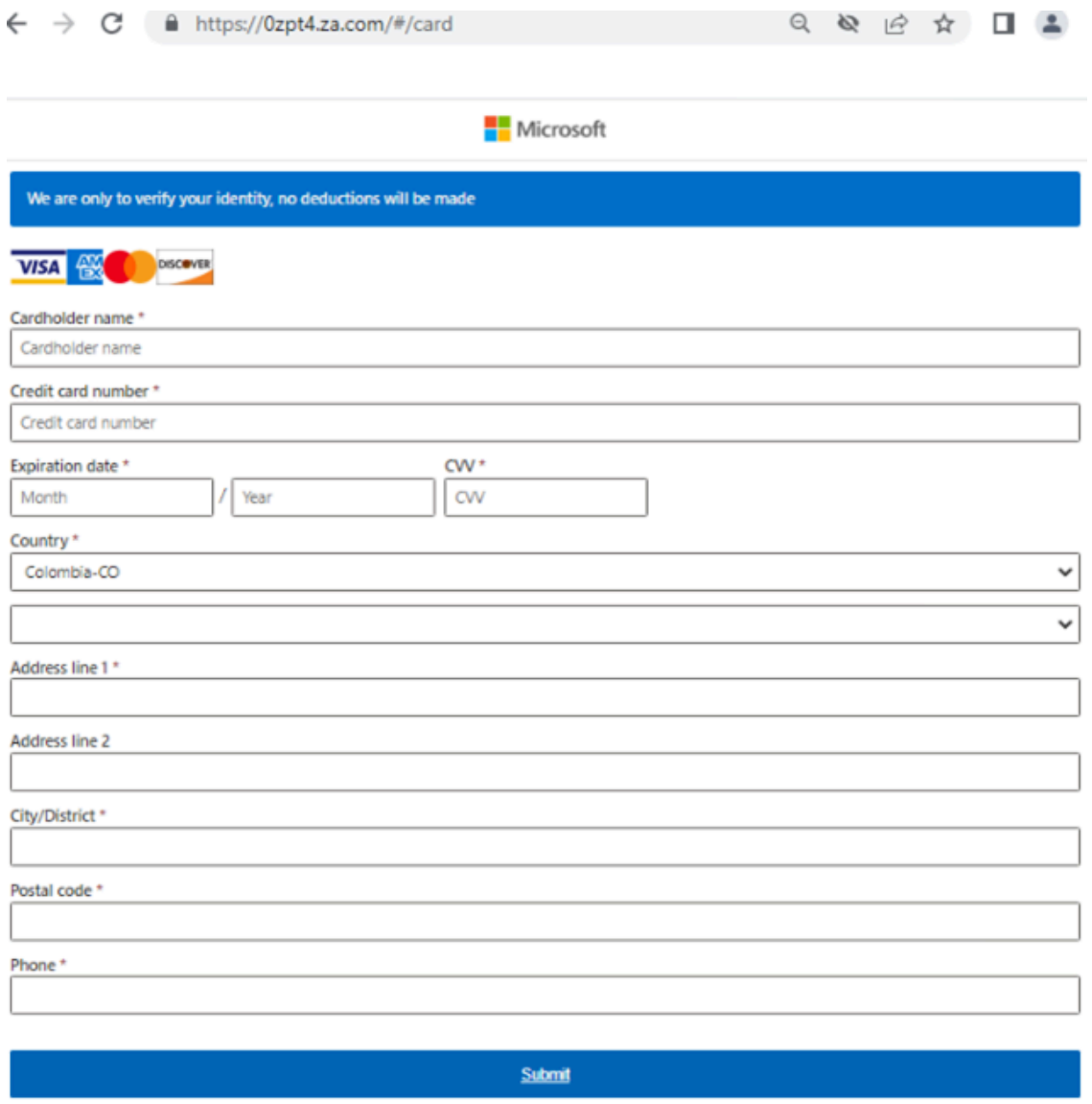

Figure 21. Phishing website asking for sensitive banking details

And after entering the sensitive details and passing checks (such as a credit card number validity check), the phishing kit displays a success message and shows the confirmation code, which closes the browser and uninstalls the phishing kit from the system. This is the “checkcode” mentioned previously.

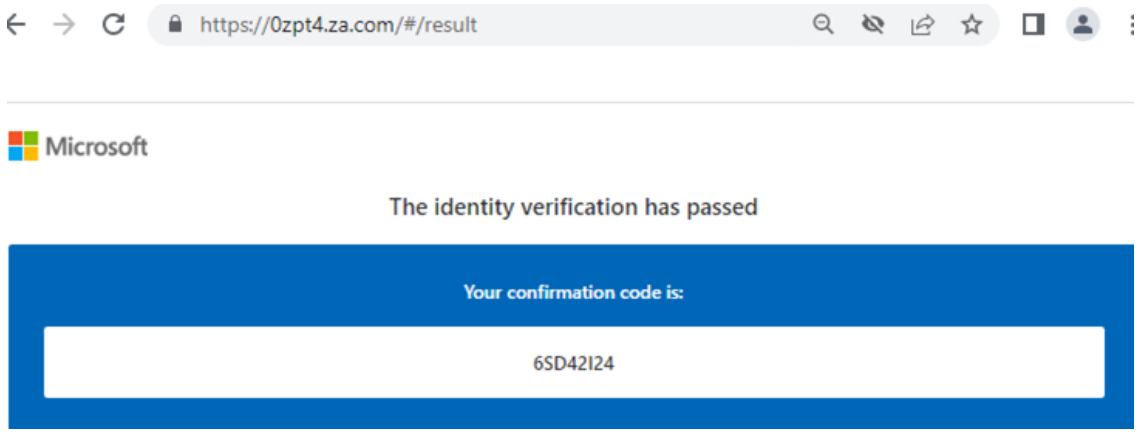

Figure 22. Confirmation code displayed after credentials were successfully phished

```
u = e => (e.preventDefault(), Y().fns.validateCardNumber(M.card) ? Y().fns.
validateCardExpiry(M.mm, M.yy) ? Y().fns.validateCardCVC(M.cvv) ? (M.loading = !0,
 void fetch("/index.php", {
```

Figure 23. Code in the phishing webpage to validate the user's credit card number, expiration date, and CVV

This is a simple but interesting approach. If the phishing page is displayed in a standard browser window or tab, the user can simply close the window or tab and continue working. In this case, two processes are used — one for persistence, and another for displaying the phishing window. This sort of persistence, though simple and easy to remove, can remain and display longer, which could annoy victims enough to force them to enter the credentials just to get rid of the phishing window.

The credential verification and confirmation code are two useful features that make this phishing kit more successful, as the victim cannot simply close the window or enter fake information just to get rid of the window.

Attribution

We classify both previously mentioned campaigns to be related to previously analyzed campaigns because of the following similarities:

1. The use of the same crypter, which encrypts the dropper stage (described in our previous blog posts).
2. The use of Data Encryption Standard (DES) with the same key ("taskhost") and initialization vector ("winlogon").
3. The use of the same name of the DLL export function (for later versions of CopperStealer).
4. The use of similar mutex naming conventions (previously: exist_sign_cps, exist_sign_task_Hello001, exist_sign__install_r3 now: exist_sign_redns, dl_exist_sign_cnzz, dl_exist_sign_sys).

Conclusion

We uncovered two new campaigns conducted by the Water Orthrus threat actor. The actor has not only refined their malware but has also tailored their attacks for different targets. The CopperStealth campaign, distributed to machines in China, focuses in installing the rootkit, which later delivers additional malware. Meanwhile, CopperPhish aims to phish credit card information and is distributed globally. It is likely that the actor has multiple objectives at the same time. Our findings also highlight the shift of Water Orthrus's interests, from personal information to cryptocurrency, and now targeting credit card information.

A proactive approach on security can help organizations protect their devices against these types of threats. Trend Micro Apex One™products employs a variety of threat detection capabilities, notably behavioral analysis that protects against malicious scripts, injection, ransomware, and memory and browser attacks related to fileless threats. Additionally, the Apex One Endpoint Sensorproducts provides context-aware endpoint detection and response (EDR) that monitors events and quickly examines what processes or events are triggering malicious activity.

IOCs

The full list of IOCs can be found here.

Tags

Source: https://www.trendmicro.com/en_us/research/23/e/water-orthrus-new-campaigns-deliver-rootkit-and-phishing-modules.html