StripedFly: Perennially flying under the radar

securelist.com/stripedfly-perennially-flying-under-the-radar/110903

Authors

- Expert Sergey Belov
- Expert Vilen Kamalov
- Expert Sergey Lozhkin

Introduction

It’s just another cryptocurrency miner… Nobody would even suspect the mining malware was merely a mask, masquerading behind an intricate modular framework that supports both Linux and Windows. It comes equipped with a built-in TOR network tunnel for communication with command servers, along with update and delivery functionality through trusted services such as GitLab, GitHub, and Bitbucket, all using custom encrypted archives. The amount of effort that went into creating the framework is truly remarkable, and its disclosure was quite astonishing.

How it started

In 2022, we came across two unexpected detections within the WININIT.EXE process of an older code which was earlier observed in Equation malware. Subsequent analysis revealed earlier instances of suspicious code dating back to 2017. During that time, it had

effectively evaded analysis and had previously been misclassified as a cryptocurrency miner. However, while it was in fact serving that purpose, that wasn't its main objective.

We decided to conduct a comprehensive analysis of the collected samples with the sole objective of resolving any uncertainties. What we discovered was completely unexpected; the cryptocurrency miner was just one component of a much larger entity. This malware employed a custom EternalBlue SMBv1 exploit to infiltrate its victims' systems. Importantly, our investigation, which considered binary timestamps, indicated that this exploit was created prior to April 2017. It is worth noting that the EternalBlue exploit was publicly disclosed by the Shadow Brokers group on April 14, 2017.

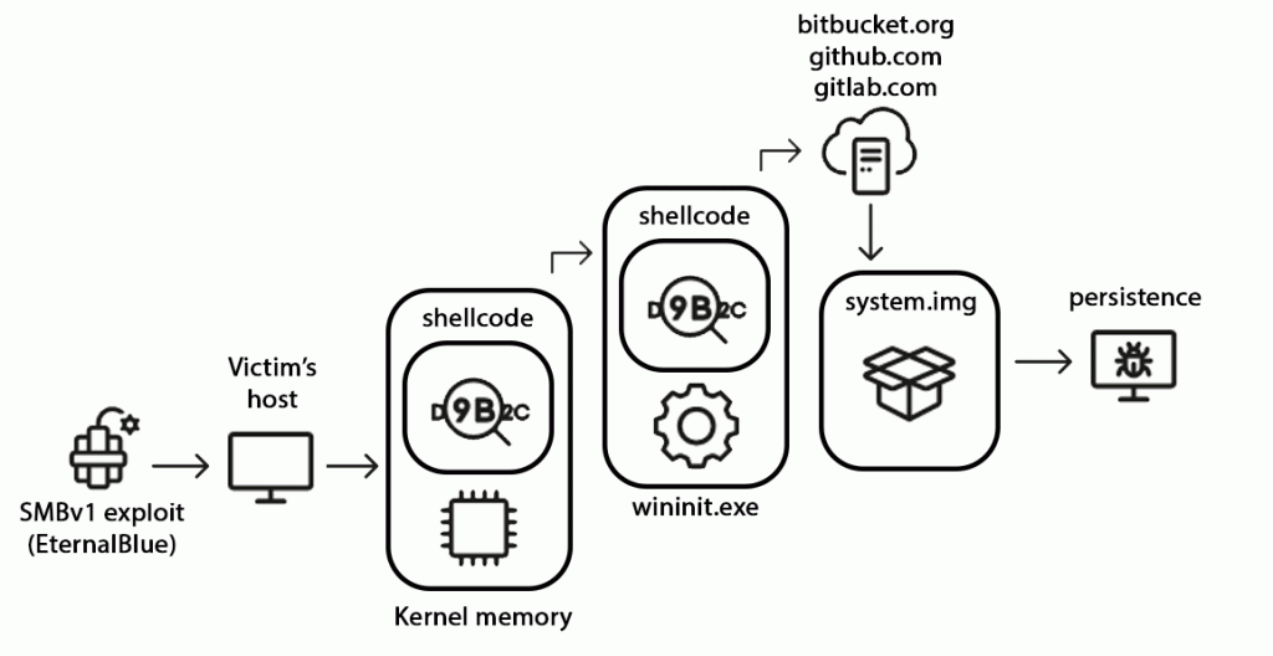
What set this particular worm apart from other malware that used EternalBlue was its distinctive propagation pattern. It spread quietly, allowing it to avoid detection by most security solutions. With the completion of our extensive private investigation report, this article now provides a concise overview of our findings.

The infection

The first detected shellcode was located within the **WININIT.EXE** process, which has the ability to download binary files from **bitbucket[.]org** and execute PowerShell scripts. At the time of the initial detections, the infection vector was unknown. However, as our investigation progressed, we discovered an SMBv1 exploit that was remarkably similar to EternalBlue.

The kernel shellcode, delivered via an exploit, injects an additional shellcode into the user space. The payload is then deployed, which includes a framework with plugin-like expandable functionality along with an extremely lightweight TOR network client. Once this process is completed, the gates are permanently sealed, and the malware proceeds to disable the SMBv1 protocol on the infected system.

The worming functionality attempts to propagate within the local network, relying not only on the exploit but also on the SSH protocol, using keys found on the victim's machine.



Infection flow on Windows host

Persistence

For its persistence malware employs various methods. Its behavior depends on the availability of the PowerShell interpreter and the privileges granted to the process. Typically, the malware would be running with administrative privileges when installed via the exploit, and with user-level privileges when delivered via the Cygwin SSH server.

If the PowerShell is not present, the malware generates a hidden file with MZ-PE loader with a randomized name located in **%**APPDATA**%** directory. This loader includes a copy of **system.img** and is subsequently registered with a GUID-like name within the Windows registry key **Software\Microsoft\Windows\CurrentVersion\Run**.

If PowerShell is installed, it will behave differently based on whether administrative access is available. If administrative rights are present, its ether executes a PowerShell script that creates two task scheduler entries with GUID-like names and with different triggers. The action of these tasks is run of PowerShell loader script.

If administrative rights are unavailable, the PowerShell script loader, is surrounded thousands of spaces in the beginning and in the end, is placed in the **HKCU\Software\Microsoft\Windows\CurrentVersion\Applets** registry key, which is then registered in the **HKCU\Software\Microsoft\Windows\CurrentVersion\Run** key with the following command:

```
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -W Hidden -
Command "Invoke-Expression((Get-ItemProperty -Path
'HKCU:\Software\Microsoft\Windows\CurrentVersion\Applets').
'%RegValNameFromPrevStep%')".
```

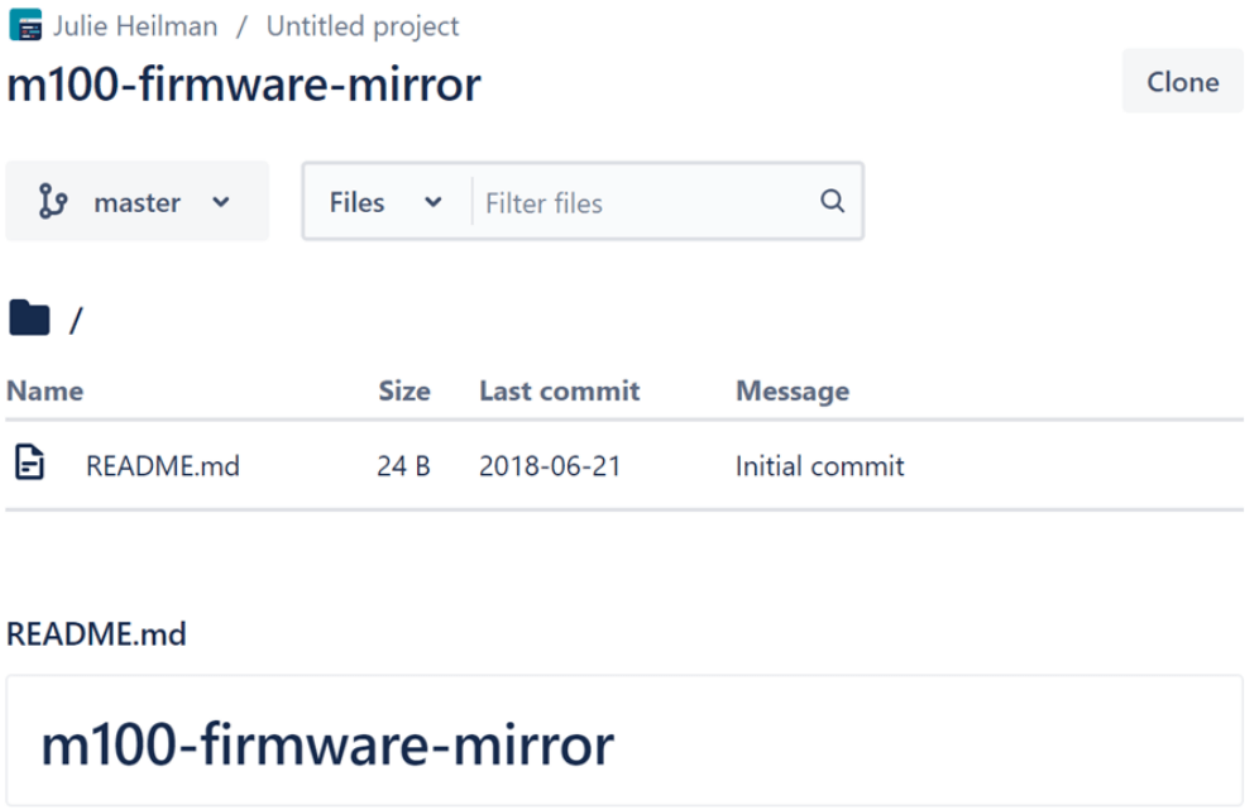
On both cases when PowerShell is available, the body of malware archive itself is stored inside the registry key **Software\Microsoft\Windows\CurrentVersion\Shell**, Base64 encoded and surrounded with hundreds of spaces, and is started by mentioned above PowerShell scripts.

On Linux hosts the malware process hides itself under name **(sd-pam)**. And has a rich set of ways to establish persistence: it may be system or user **systemd** service, autostarted **.desktop** file, or additional line inside one of suitable **/etc/rc***, **profile, bashrc** or **inittab** file. The malware executable file is placed in /tmp directory with a random name.

Bitbucket repository

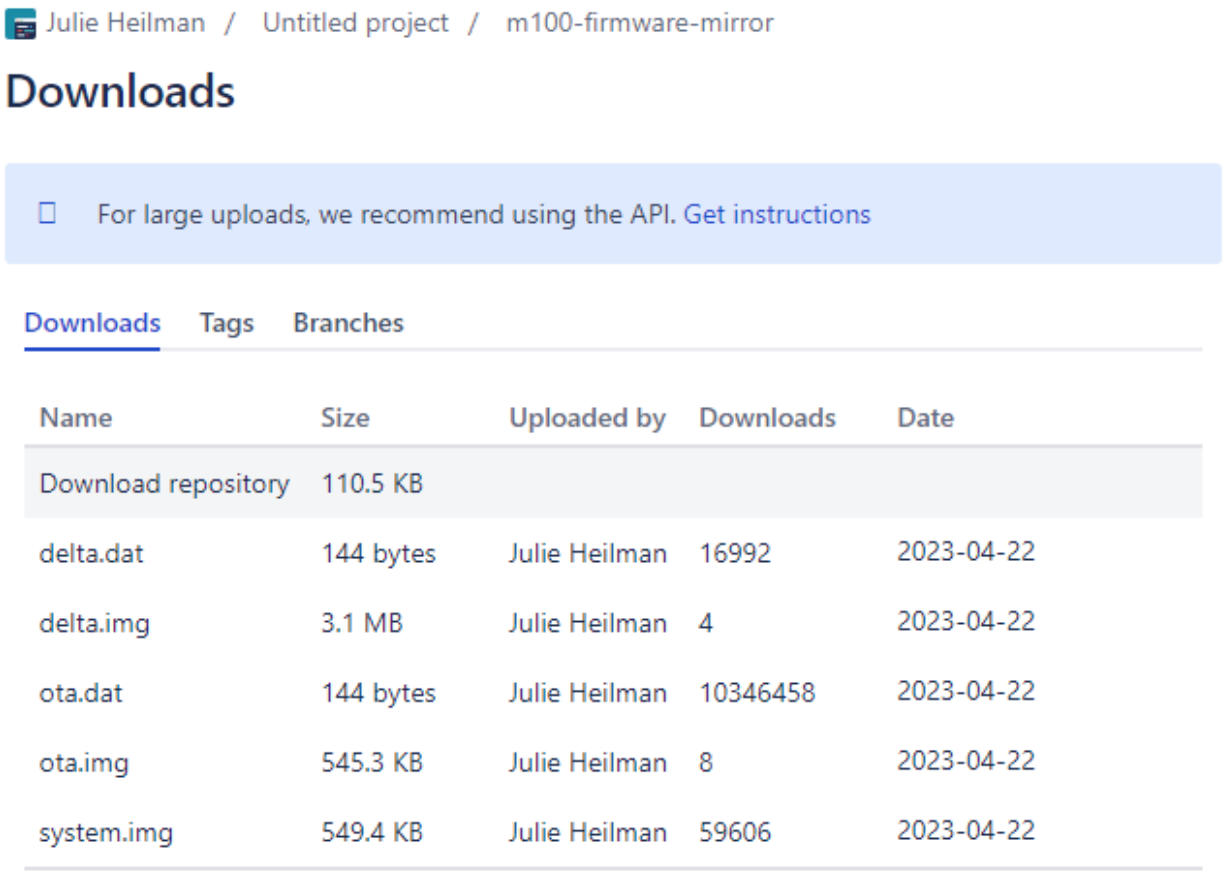
To minimize the footprint of the initial exploit, all elements that can be offloaded are encapsulated in an encrypted and compressed custom binary archive. This archive is discreetly hosted on legitimate websites, cleverly disguised as firmware binaries for enigmatic devices labeled “m100”.

The Bitbucket repository was created on June 21, 2018, under the account of Julie Heilman, and it remains the sole repository associated with this profile.



Bitbucket repository content

The repository houses only a **README.md** file with the project's name inside. Notably, the **Downloads** folder, which would normally contain compiled project binaries, contains five binary files: **delta.dat**, **delta.img**, **ota.dat**, **ota.img**, and **system.img**.



Downloads folder of the repository

This folder lacks any versioning, and the download counter only reflects the number of downloads since the last file update. In particular, the **system.img** file serves as the authentic payload archive used for initial Windows system infections. The download counter for this file accurately reflects the number of new infections since its last update. During our analysis, the file was last updated on February 24, 2022, and the number of initial infections stood at 160,000 victims as of June 2022. However, as of September 2023, the number had dropped to 60,000 since the last update in April 2023.

The files **ota.img** and **delta.img** are used to update the malware, where ota.img corresponds to the Windows version and delta.img corresponds to the Linux version. Interestingly, **system.img** and **ota.img** are functionally identical, although **ota.img** includes supplementary metadata for integrity verification, while **delta.img** serves as the initial infection payload for Linux hosts compromised via SSH by the Windows version.

The **ota.dat** and **delta.dat** files, along with the version files, serve as tools for the malware to check for the availability of new updates. However, it's worth noting that the download counters for **ota.img** and **delta.img** do not accurately reflect the current number of infected victims. This is because the malware primarily receives updates from its C2 server, and only resorts to downloading update files from the repository when the C2 server is unresponsive.

During our analysis, approximately one million updates were obtained from the repository. At the time of writing, there were only eight updates for Windows systems and four for Linux systems, suggesting one of two scenarios: either there are minimal active infections, or the C2 server remains active and responsive to all infected victims.

The TOR

The C2 server is nestled in the TOR network and has the .onion address **gpiekd65jgshwp2p53igifv43aug2adacdebmuuri34hduvijr5pfjad[.]onion:1111**.

In order to communicate with the C2, the malware employs a custom, lightweight implementation of a TOR client. Interestingly, this implementation does not appear to be based on any known open-source TOR implementations. Many standard TOR features such as routing, directory listing, relay and exit node modes, and support for control protocols are conspicuously absent.

At regular intervals, the malware initiates TCP connections with the C2 server, transmitting a greeting message containing the victim's unique ID. It then proceeds to send an empty beacon message every minute.

The level of dedication demonstrated by this functionality is remarkable. The goal of hiding the C2 server at all costs drove the development of a unique and time-consuming project – the creation of its own TOR client. Such an approach is by no means common

among APT and crimeware developers, and this notable example underscores the sophistication of this malware against the background of many others. Its functional complexity and elegance remind us of the elegant code implementing delay tolerant Equation communications networking and other libraries, reinforcing its classification as a highly advanced threat.

The modules

The malware payload itself is structured as a monolithic binary executable code designed to support pluggable modules to extend or update its functionality. This architectural approach is a hallmark of APT malware. Each module is responsible for implementing and registering a callback function that is triggered when a connection to the C2 server is established or lost, or when a message is received from the C2 server. The functionality within these modules is divided into two types: service and extended functionality modules.

Service modules

Configuration storage

The module securely stores the AES-encrypted malware configuration by creating a registry key resembling a GUID within the **HKCU\Software\Classes\TypeLib** key for the Windows version. The Linux version hides this information in randomized hidden folders located in the user's home directory.

Upgrade/Uninstall

When the initial connection to the C2 server is established, the service module generates an 8-byte victim ID, stores it, and then reuses it along with the hash of the utilized **system.img** file for reporting back to the server. This module is designed to implement just two specific commands:

- The server sends a new version of **system.img** and the upgrade process is executed by either a generated script or a generated executable.
- Perform a full uninstall.

If the C2 server remains offline for more than 20 minutes and this condition persists, the module initiates an attempt to download the **ota.dat** file (or **delta.dat** for Linux) and subsequently verifies its integrity. If the version of the file has changed, the module triggers the upgrade procedure by downloading the appropriate .img file: ota.img for Windows and delta.img for Linux.

Functionality modules

Reverse proxy

The module grants access to the victim's network and allows the execution of remote actions on behalf of the victim.

Miscellaneous command handler

The module encompasses a range of commands designed to interact with the victim's file systems, capture screenshots, retrieve system versions, and obtain the active X11 display on Linux (the default is **WinSta0** on Windows). It also includes a command capable of executing shellcode received from the C2 server.

Credential harvester

The module operates a dedicated thread that runs periodic scans every two hours. During these scans, it collects a range of sensitive information from all active users. This information includes website login usernames and passwords, as well as personal autofill data such as name, address, phone number, company, and job title. It also captures known Wi-Fi network names and the associated passwords, as well as SSH, FTP, and WebDav credentials from popular software clients such as FileZilla, Cyberduck, and WinSCP.

It's worth noting that web browser support for credential harvesting extends beyond well-known browsers such as Chrome, Firefox, and Internet Explorer, and includes such lesser-known browsers as Nichrome, Xpom, RockMelt, Vivaldi, SaMonkey, Epic Privacy, and Brave.

In the Linux version, it also gathers OpenSSH keys stored in **$HOME/.ssh**, compiles a list of known hosts from **$HOME/.ssh/known_hosts**, and includes functionality to retrieve secrets from the Libsecret vault. However, this particular functionality is currently broken because there is no **dlopen** API implementation in the linked musl libc library.

Repeatable tasks

The module boasts a few built-in functional tasks that can be executed either once or on a repeatable scheduled basis, with the condition that specific windows must be visible for these tasks to proceed.

The following are brief descriptions of the tasks:

- Take a screenshot and get a list of windows that are visible at that moment.
- Execute a process with a given command line, redirect its output, and filter it using regular expressions.
- Record the microphone input.

- The task involves gathering a list of files with specific extensions, such as those related to images, documents, sounds, videos, archives, databases, certificates, source code files, and other critical user data files. This process scans all local drives and network shares, with the exception of system folders. This is the only task that works in the Linux version of the malware.

Recon module

This module compiles extensive system information and transmits it to the C2 server upon connection. The data collected includes a wide range of details, including operating system version, computer name, list of hardware MAC addresses, current username for Windows systems, /etc/passwd file for Linux systems, IP address of the machine, IP address of the currently connected TOR network exit node, system startup time, malware uptime, time and time zone, total and available memory amount, user administrative privileges, and specific Windows-related information such as the UI language and keyboard layouts, presence of antivirus software, NetBIOS name, DNS domain, owner's Windows license details, and the presence of the PowerShell command interpreter.

SMBv1 and SSH infectors

There are two modules dedicated to the penetration capabilities of the malware, which make up the core worming functionality.

The SSH infector springs into action once the credential harvester module completes its tasks. It filters the results for SSH keys and credentials and, if any are found, activates a dedicated thread. This thread operates with random timeouts ranging from 10 minutes to two hours and initiates the penetration process. First, it retrieves **delta.dat** and **delta.img** either from the cache or directly from **bitbucket[.]org**. It then proceeds to verify the integrity of these files and dynamically loads the **libeay**, **zlib** and **libssh2** libraries from **delta.img**. The next step is to attempt to connect to the remote SSH server. If the connection is successful, it invokes and parses the output of the **/bin/sh -c 'uname -nmo'** command. If the remote system is supported, the malware uploads its binary to the remote **/tmp** folder under a random name and executes it with the command **/bin/sh -c 'cat > %s; chmod +x %s; nohup sh -c "%s; rm %s" &>/dev/null'**. This approach ensures compatibility with x86, amd64, arm, aarch64 Linux CPU architectures, as well as Cygwin x86 and amd64 remote hosts using a generated MZ-PE loader.

The SMBv1 infection module serves as the primary penetration tool for Windows victims, using a custom EternalBlue exploit. Upon initial execution, it immediately disables the SMBv1 protocol by modifying the **HKLM\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters** registry key on the victim's system. It then launches two dedicated threads to perform periodic worming routines.

The first thread is responsible for inspecting the network adapter's IP address and subnet mask. It then attempts to cause infections within the entire LAN subnet. In contrast, the second thread periodically attempts to select a random internet IP address, with the following exclusions:

- Bogon networks like 0.0.0.0/8, 10.0.0.0/8, 100.64.0.0/10, 127.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16, 198.18.0.0/15, 224.0.0.0/4, 240.0.0.0/4
- 169.255.0.0/16 – mostly South Africa. This may be a bug; the authors probably meant 169.254.0.0/16 – the missing part of the bogon networks list
- 3.0.0.0/8, 15.0.0.0/8, 16.0.0.0/8, 56.0.0.0/8 – Amazon, HP, US Post and more
- 6.0.0.0/8, 55.0.0.0/8 – United States Army Information Systems Command HQ (USAISC)
- 7.0.0.0/8, 11.0.0.0/8, 21.0.0.0/8, 22.0.0.0/8, 26.0.0.0/8, 28.0.0.0/8, 29.0.0.0/8, 30.0.0.0/8, 33.0.0.0/8, 214.0.0.0/8, 215.0.0.0/8 – US Department of Defense Network Information Center (DNIC).

Supported Windows versions include Windows Vista, Windows 7, Windows Server 2008 R2, Windows 8, Windows Server 2012, and Windows 10 up to build 14392.

Monero cryptocurrency mining module

The icing on the cake is the Monero mining module. It operates in a separate process and skillfully disguises itself as a **chrome.exe** process located in the **Google\Chrome\Application\Services** directory, which can be found in either the common or local AppData directory. This deceptive facade even includes alterations to the version information and process icon of the masqueraded executable. Periodically, the malware functionality within the main module monitors the puppet mining process and restarts it if necessary. It also dutifully reports hash rate, work time, discovered nonces, and error statistics to the C2 server.

DNS resolutions for pool servers are cleverly concealed behind DNS over HTTPS requests to the Cloudflare DoH (DNS over HTTPS) service, adding an extra layer of stealth to its operations.

We strongly suspect that this module is the primary reason for the malware's ability to evade detection for an extended period. Its existence is primarily driven by the need for effective masquerading. It's worth noting that the Monero coin, which this module mines, peaked at $542.33 on January 9, 2018, after hovering around $10 in 2017. As of 2023, it is trading at around $150. While this module is certainly profitable, it's not necessarily the most lucrative use of such malware. Hunting for unencrypted binary wallets or wallet credentials, for example, could yield significantly higher profits.

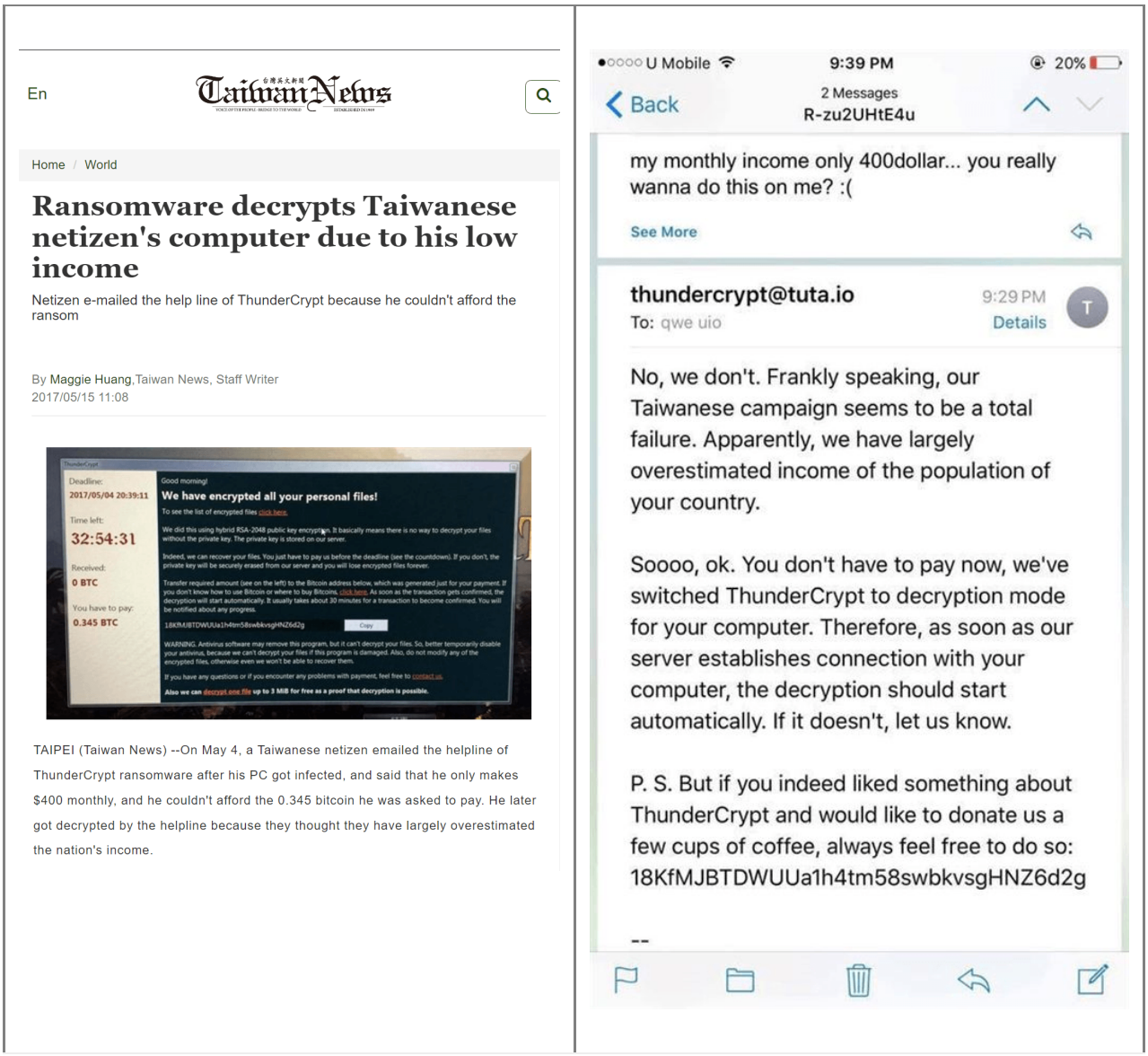
Furthermore, the presence of unencrypted strings related to mining within the malware code provides additional evidence of its potential secondary purpose.

ThunderCrypt

During our analysis, we stumbled upon an earlier version of the malware, which led us to the discovery of a related ransomware variant called ThunderCrypt. It turned out that both malware strains share the same underlying codebase and, more importantly, they communicate with the same C2 server located at **ghtyqipha6mcwxiz[.]onion:1111**.

The ThunderCrypt ransomware exhibited a strikingly similar array of functionality and modules when compared to StripedFly. These included the TOR client, configuration storage, upgrade/uninstall, and reconnaissance module, with one notable exception being the absence of the SMBv1 infection module. Interestingly, the ransomware used the file listing component of the repeatable task module as an integral part of its ransom encryption process.

Telemetry data shows that ThunderCrypt first appeared on April 23, 2017, with the main spike in activity occurring in the following month of May. Intriguingly, it attracted the attention of Taiwan News because of a rather amusing incident. A Taiwanese netizen, unable to afford the 0.345 bitcoin ransom demanded for decryption, decided to contact the attackers via the support email address provided. In his email, he candidly explained his predicament, citing a modest monthly income of just $400. To the surprise of many, the attackers responded by conceding that they had overestimated the income of the Taiwanese population, and the attack was deemed a complete failure.



Taiwan News regarding ThunderCrypt

EternalBlue

We assess that there are parallels between the EternalBlue exploit and the authors behind StripedFly. Our assumptions rely on the accuracy of the PE timestamps, and although it is not possible to validate the authenticity of the timestamps of the initial EternalBlue module in question, subsequent updates of the malware contained timestamps that approximately match with telemetry data, so it is likely that the initial timestamps are also accurate. The timeline we've reconstructed is as follows:

- April 9, 2016: Earliest known version of StripedFly incorporating EternalBlue, as indicated by PE timestamps.
- August 2016: Initial leak by the Shadow Brokers group.
- March 14, 2017: Microsoft releases security bulletin MS17-010, introducing a patch for the EternalBlue exploit.
- April 14, 2017: Shadow Brokers release a leak containing the EternalBlue exploit.
- April 15, 2017: The first EternalBlue-infused ransomware, ExPetr, appears.

- April 20, 2017: Introduction of the earliest version of ThunderCrypt ransomware (without EternalBlue).
- April 23, 2017: First detection of ThunderCrypt in our telemetry data.
- May 12, 2017: WannaCry ransomware attack utilizing EternalBlue.
- June 27, 2017: ExPetr attack using EternalBlue.
- August 24, 2017: First detection of StripedFly in our telemetry, one year after the date provided by the initial PE timestamps.

Taken together, these various data points suggest the similarities to Equation malware, although there is no direct evidence that they are related. Discovery of the malware was facilitated by the signatures associated with the Equation malware family, and the coding style and practices resemble those seen in SBZ malware.

Conclusion

This article represents an effort to bring the story out of the confines of a private technical report that was released last year. Created quite some time ago, StripedFly has undoubtedly fulfilled its intended purpose by successfully evading detection over the years. Many high-profile and sophisticated malicious software have been investigated, but this one stands out and it truly deserves attention and recognition.

What was the real purpose? That remains a mystery. While ThunderCrypt ransomware suggests a commercial motive for its authors, it raises the question of why they didn't opt for the potentially more lucrative path instead. The prevailing narrative often centers around ransomware actors collecting anonymous ransoms, but this case seems to defy the norm.

The question remains, but only those who crafted this enigmatic malware hold the answer. It's difficult to accept the notion that such sophisticated and professionally designed malware would serve such a trivial purpose, given all the evidence to the contrary.

Indicators of compromise

C2 servers

gpiekd65jgshwp2p53igifv43aug2adacdebmuuri34hduvijr5pfjad[.]onion
ghtyqipha6mcwxiz[.]onion
ajiumbl2p2mjzx3l[.]onion

URLs

bitbucket[.]org/JulieHeilman/m100-firmware-mirror/downloads/
bitbucket[.]org/upgrades/um/downloads/
bitbucket[.]org/legit-updates/flash-player/downloads
gitlab[.]com/JulieHeilman/m100-firmware-mirror/raw/master/
gitlab[.]com/saev3aeg/ugee8zee/raw/master/

github[.]com/amf9esiabnb/documents/releases/download/
tcp://pool.minexmr[.]com:4444
tcp://mine.aeon-pool[.]com:5555
tcp://5.255.86[.]125:8080
tcp://45.9.148[.]21:80
tcp://45.9.148[.]36:80
tcp://45.9.148[.]132:8080

system.img
b28c6d00855be3b60e220c32bfad2535
18f5ccdd9efb9c41aa63efbe0c65d3db
2cdc600185901cf045af027289c4429c
54dd5c70f67df5dc8d750f19ececd797
d32fa257cd6fb1b0c6df80f673865581
c04868dabd6b9ce132a790fdc02acc14
c7e3df6455738fb080d741dcbb620b89
d684de2c5cfb38917c5d99c04c21769a
a5d3abe7feb56f49fa33dc49fea11f85
35fadceca0bae2cdcfdaac0f188ba7e0

delta.dat
00c9fd9371791e9160a3adaade0b4aa2
41b326df0d21d0a8fad6ed01fec1389f

delta.img
506599fe3aecdfb1acc846ea52adc09f
6ace7d5115a1c63b674b736ae760423b

ota.dat
2e2ef6e074bd683b477a2a2e581386f0
04df1280798594965d6fdfeb4c257f6c

ota.img
abe845285510079229d83bb117ab8ed6
090059c1786075591dec7ddc6f9ee3eb

ThunderCrypt
120f62e78b97cd748170b2779d8c0c67
d64361802515cf32bd34f98312dfd40d

- Data theft
- Encryption
- EternalBlue
- Linux
- Malware Descriptions
- Malware Technologies
- Miner

- Targeted attacks
- TOR

StripedFly: Perennially flying under the radar

Your email address will not be published. Required fields are marked *