vb 2023 LONDON

4 - 6 October, 2023 / London, United Kingdom

# TETRISPHANTOM: TARGETED ATTACKS USING SECURE USB

Noushin Shabab

*Kaspersky, Australia*

noushin.shabab@kaspersky.com

## ABSTRACT

In early 2023 we discovered an ongoing attack targeting government entities in the APAC region by compromising a specific type of a secure USB drive, which provides hardware encryption.

The malware modules used in these attacks execute commands and collect files and information from compromised machines and pass them on to further machines using the same or other secure USB drives as a carrier. They are capable of executing other malicious files on the infected systems too.

The attack comprises sophisticated tools and techniques such as:

- Virtualization-based software obfuscation for malware components
- Low-level communication with the USB drive using direct SCSI commands
- Self-replication through connected secure USB drives to propagate to other air-gapped systems
- Injection of code into a legitimate access management program on the USB drive, which acts as a loader for the malware on a new machine.

The attacks investigated in this research were extremely targeted and had a quite limited number of victims. Our investigation revealed a high level of sophistication in the malicious tools used in the deployment of the attacks. We believe these attacks have been carried out by a highly skilled and resourceful threat actor interested in espionage activities in sensitive and protected government networks, and therefore it's very important to build a deep understanding of the TTPs of this threat actor and to watch out for future attacks.

In this presentation we will look at the various aspects of this investigation with a particular focus on the technical analysis of the malicious files involved in the attacks.


## INTRODUCTION

In early 2023 we discovered an ongoing attack targeting government entities in the APAC region by compromising a specific type of a secure USB drive. Such secure USB drives are used by the government organizations of the target country to securely store and transfer data physically between computer systems. The USB drive contains a protected partition which can only be accessed via custom software named 'UTetris.exe', which is bundled on an unencrypted part of the USB drive, and a passphrase known to the user.

We discovered two trojanized versions of the UTetris.exe program deployed on compromised USB drives, version 1.0 being used since September 2022 and version 2.0 from October 2022 until now. The trojanized UTetris.exe is paired with a very sophisticated malicious module we named 'XMKR'. The name XMKR is derived from log strings found in the binary file of the malware and seems to be the name used by the malware authors for this module.

The XMKR module is deployed on a *Windows* machine and is responsible for maintaining infection on a compromised machine as well as spreading the infection to other secure USB drives.

Further investigation of the attacks revealed that the attack campaign has been ongoing for at least a few years and involved a number of other malicious modules which were used for several purposes:

- To steal documents and other files from the target systems
- To help the initial phases of research and development of the XMKR and trojanized UTetris.exe by collecting very specific details about USB drives used in the target environment
- Used alongside the XMKR and trojanized UTetris.exe to connect back to the attack servers to receive commands and also exfiltrate information captured from the secure USB drives (likely coming from air-gapped systems).

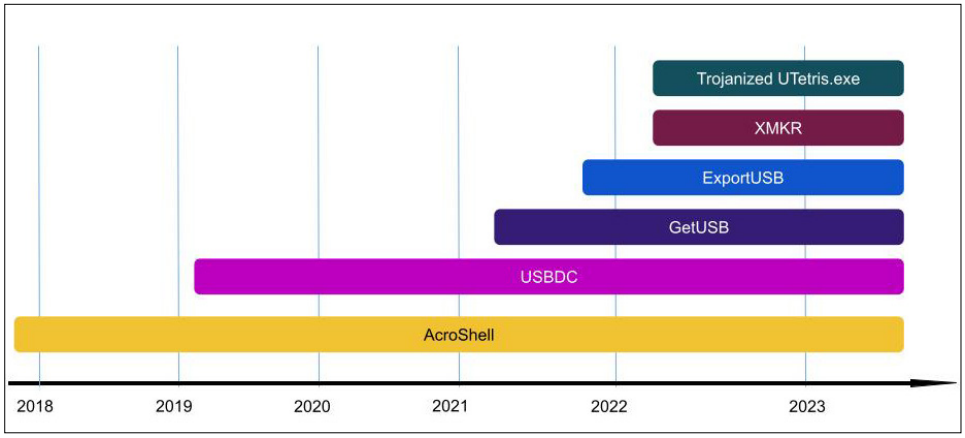


*Figure 1: Timeline of the development and use of the malware components.*

## SPREADING

A compromised USB drive has the ability to infect the systems that it gets connected to. We assume such systems are likely air-gapped systems used to store and process classified data of the victim government organization.

A hidden copy of the XMKR module is deployed on the USB drive using the SCSI WRITE command to make it inaccessible through common file operations such as using the ReadFile() API. Once a compromised USB drive is connected to a machine, the trojanized UTetris.exe deploys the XMKR module onto the victim's machine.

In turn, the XMKR module has a copy of the trojanized UTetris.exe as a resource entry. On a *Windows* system the deployed XMKR module waits for a target USB drive to be plugged into the machine. Once a USB drive with the desired characteristics is connected to the machine, the XMKR module replaces the legitimate UTetris.exe with the trojanized version from its own resource entry.

In case a compromised USB drive is found on the machine, the XMKR module replaces the trojanized UTetris.exe on the USB drive with its own version, if the USB drive has an older version of the malware. This way, an infected *Windows* machine spreads the infection to more systems by compromising new secure USB drives.

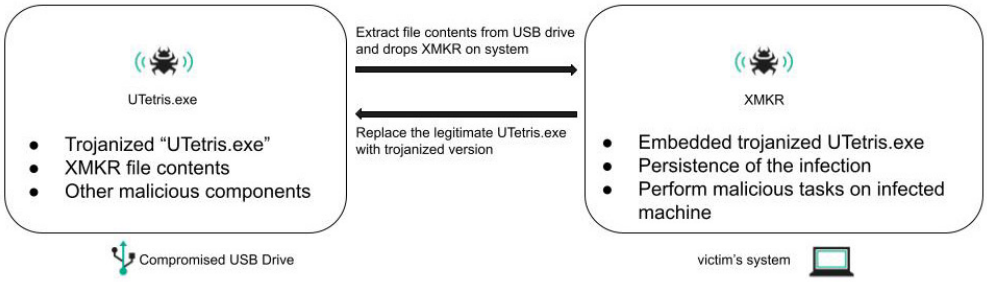


*Figure 2: Spreading to new machines using trojanized UTetris.exe and 'XMKR'.*

Even though the high-level view of the spreading seems straightforward, the details of how these two malicious modules operate are intricate and nuanced, and it is these characteristics that have made these attacks quite stealthy and sophisticated.

## TROJANIZED UTETRIS.EXE

The target USB drive is a very particular secure USB drive. It appears to have been developed by a government entity to be used only inside specific networks with sensitive and valuable information. It is not available to common consumers.

We already mentioned the UTetris.exe program that is bundled with this type of secure USB drive. When connected to a system the target USB drive uses one of the following two names: 'USBAV' or 'USB_AV'.

Once an infected USB drive is connected to a machine and the trojanized UTetris.exe program is executed by the user when unlocking an encrypted partition, the malicious code injected into the UTetris.exe program is executed. The malicious code from the UTetris.exe program extracts the XMKR file contents from a hard-coded offset of the USB drive via an SCSI READ command sent as an IOCTL code to the drive, and drops and executes the file on the victim's machine. The use of SCSI commands to interact with the USB drive is a stealthy technique that is deployed into both the trojanzied UTetris.exe and the XMKR module.

The attackers took a number of measures to make sure the trojanized UTetris.exe file does not raise any suspicions.

First, the legitimate UTetris.exe found on target USB drives does not have a digital signature, so patching some parts of the binary to include malicious code was safe for the attackers.

Second, the attackers either only patched small portions of the binary which were a few bytes long and did not affect the whole legitimate operations of the program, or added code to the end of a section which was left empty (filled with null bytes) for alignment purposes.

```
.text:00417D25                 jmp     __CxxFrameHandler
.text:00417D25 ; } // starts at 417D10
.text:00417D25 ; END OF FUNCTION CHUNK FOR sub_410830
.text:00417D25 ; ---------------------------------------------------------------------
.text:00417D2A                 align 400h
.text:00417D2A _text           ends
.text:00417D2A
.idata:00418000 ; Section 2. (virtual address 00018000)
.idata:00418000 ; Virtual size                  : 00003B11 (  15121.)
.idata:00418000 ; Section size in file          : 00004000 (  16384.)
.idata:00418000 ; Offset to raw data for section: 00018000
.idata:00418000 ; Flags 40000040: Data Readable
.idata:00418000 ; Alignment     : default
```

*Figure 3: End of .text section in original UTetris.exe program.*

```
.text:00417D25                          jmp     __CxxFrameHandler
.text:00417D25 ; } // starts at 417D10
.text:00417D25 ; END OF FUNCTION CHUNK FOR sub_410830
.text:00417D25 ; ---------------------------------------------------------
.text:00417D2A                          align 10h
.text:00417D30 ; START OF FUNCTION CHUNK FOR sub_408870
.text:00417D30
.text:00417D30 patch_drop_XKMR:                         ; CODE XREF:
.text:00417D30                          pusha
.text:00417D31                          call    extract_XMKR_from_USB
.text:00417D36                          popa
.text:00417D37                          call    sub_403FB0
.text:00417D3C                          jmp     loc_408A14
.text:00417D3C ; END OF FUNCTION CHUNK FOR sub_408870
.text:00417D3C ; ---------------------------------------------------------
.text:00417D41                          align 8
```

*Figure 4: End of the .text section in a trojanized UTetris.exe.*

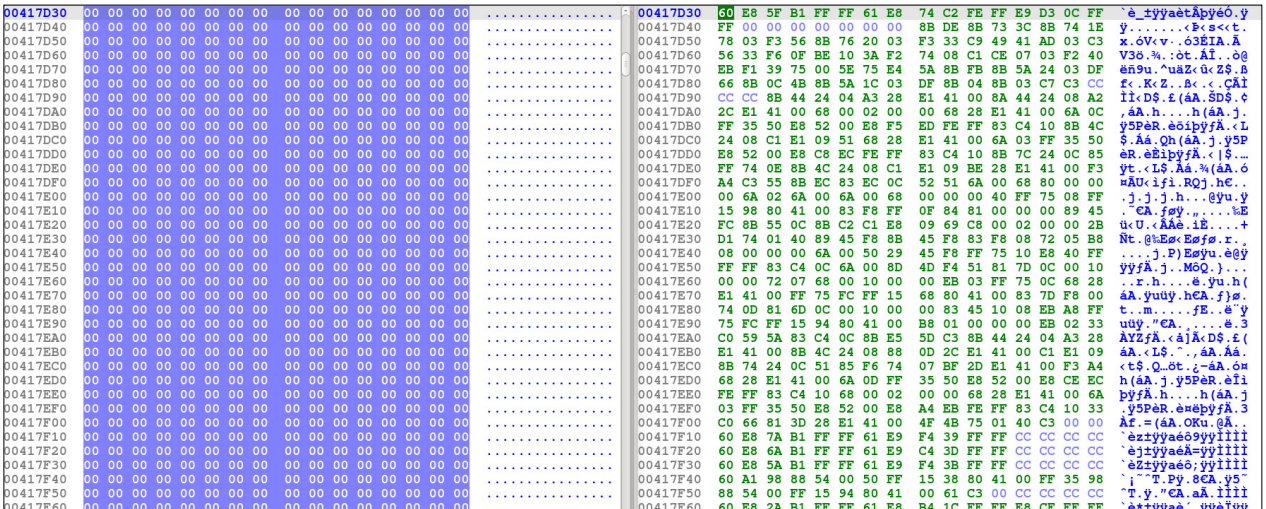

*Figure 5: Malicious code added to the end of the .text section.*

Only in one part did the attackers replace a statically linked library function with their own malicious code. The overwritten function is called 'vlineColor' and is used in the rendering of the UI of the UTetris.exe program, and patching it would not raise too much alarm.

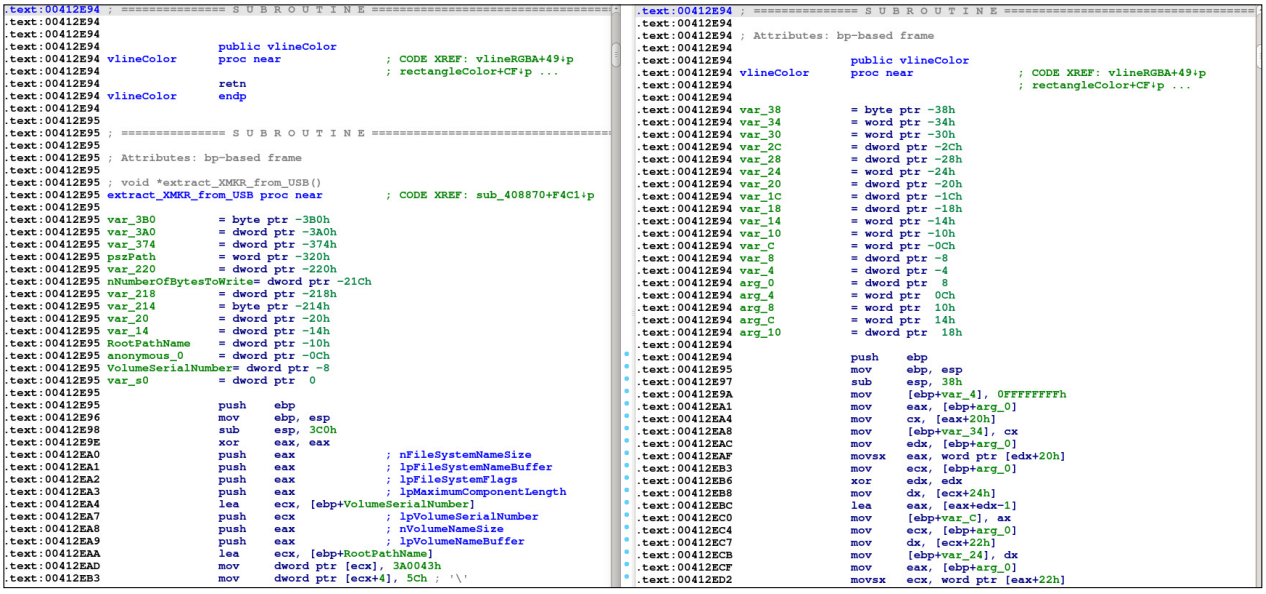

*Figure 6: vlineColor function overwritten in UTetris.exe.*

Third, the attackers added extra waiting time and used mutexes to synchronize disk operations such as deleting, renaming and copying files on the USB drive to ensure that the malware components don't conflict with the UTetris.exe program used by the user.

Last but not least, there are other modules/components of the attack that are deployed on the compromised USB drive in a stealthy way. These components include a copy of the malicious XMKR file and are kept at specific offsets of the raw USB drive via SCSI WRITE commands. None of these components are appended to or injected into the trojanized UTetris.exe file, therefore they don't affect or change the layout of the UTetris.exe binary.

Overall, the total size of the binary, size and characteristics of each section, and various file headers and data directories remain the same.

### XMKR

This module is deployed to the computer systems by an infected secure USB drive. We have discovered three versions of the XMKR module and there are various samples of each version which are sometimes packed with the UPX packer. Samples of version 1 were compiled and used in July and August 2022; version 2 in September and October 2022; and the latest version we observed is version 3, which was first used in November 2022 with the latest sample we found compiled in February 2023. Version 3 of this module is still actively used against the targets.

It's worth mentioning at this point that the attackers had extensive knowledge of the target USB drive. They had knowledge of the default passphrase of the target USB drive, the number of allowed invalid attempts to unlock the drive, as well as the specific offsets of the protected partition of the USB drive in which these values are stored.

Both the trojanized UTetris.exe and the XMKR module use IOCTL codes to send SCSI READ and SCSI WRITE commands to the USB drive in order to read and write data to hard-coded specific offsets of the USB drive.

### Persistence and USB drive monitoring

Once executed on a system, the XMKR module registers itself under the Run registry key in order to ensure the persistence of the infection by executing at every system reboot. It uses 'MSVC_SERVICE_PROCEDURE' for the registry value name to disguise itself as a common system service. Afterwards, this malware expects a target USB drive to be attached to the system.

As soon as a USB drive is plugged in, the XMKR module first checks whether the drive name is either 'USB_AV' or 'USBAV', then confirms the presence of the UTetris.exe program, either the legitimate one or a trojanized version. If the USB drive meets these two conditions, then the malware continues either to infect the USB drive with a trojanized version of UTetris.exe or, in case of an already infected USB drive, the module extracts and updates various components present on the drive.

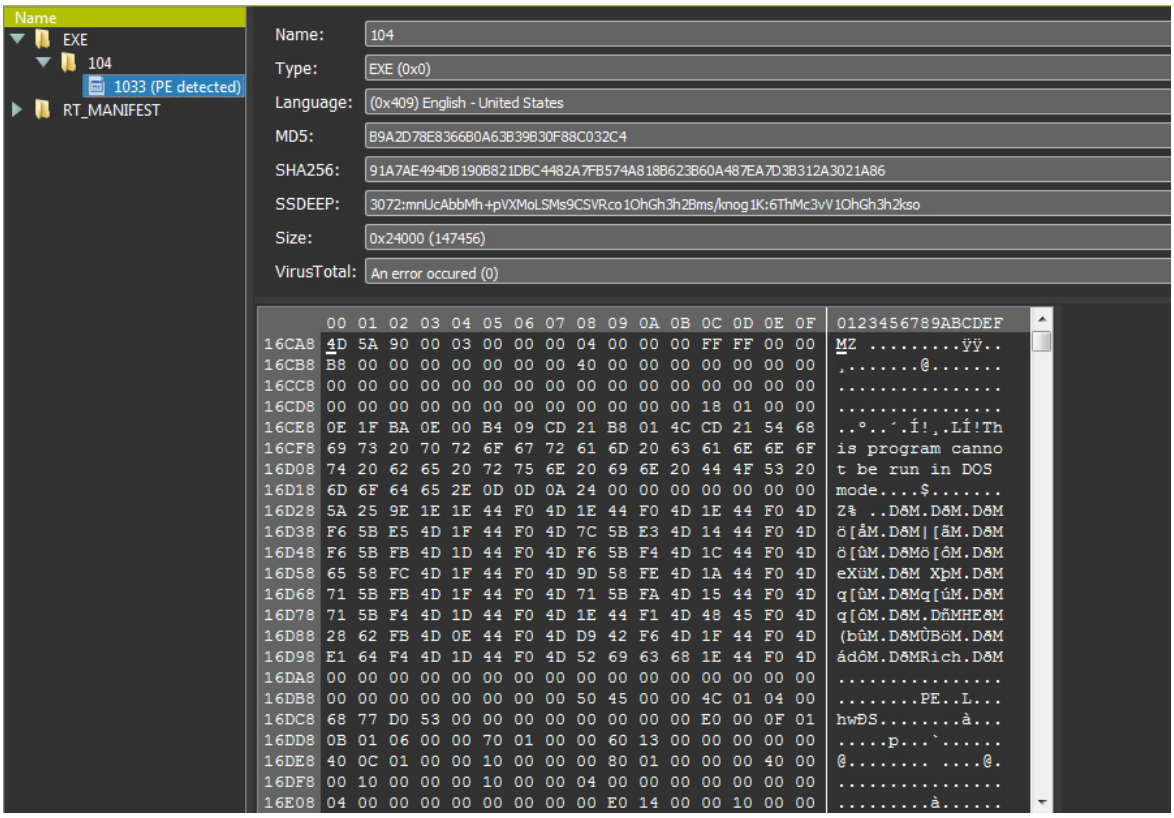


*Figure 7: Trojanized UTetris.exe as a resource entry of XMKR.*

### Unlocking the USB drive

The XMKR module continues by attempting to unlock the USB drive. It first tries to read from a specific offset of the drive via the SCSI READ command in order to see if the drive is already unlocked. If the drive is unlocked, the malware extracts a value known to it as 'USB_ID' from a hard-coded offset of the drive. Otherwise, if the drive is locked, the malware reads the number of failed unlocking attempts from the drive to make sure it doesn't go over a predefined threshold (the default is 10). If the number of failed unlock attempts does not exceed the predefined threshold, it uses the SCSI WRITE command to try the default passphrase, which is '111111111111111111111111111111111'. If it manages to access the drive, it continues malicious operations as described further.

### Interaction with a compromised USB drive

There are several components present on a compromised USB drive that are read and updated by the XMKR module every time the USB drive is connected to an infected system. These components are all written directly on the drive using the SCSI WRITE commands and there is no file handle attached to them.

The malicious components on the compromised USB drive are as follows:

- A copy of the XMKR module. This module is overwritten only if a newer version is present on the infected machine.
- A list of all 'USB_ID' values extracted from various compromised USB drives. The 'USB_ID' value is extracted from a specific offset of the USB drive by the XMKR module.
- A list of the IP addresses and MAC addresses of the machines that have been infected so far.
- A list of tasks to perform on a victim's machine. This task list can also be extracted from a file named 'task_pub.dat' on the victim's machine. If such a file is found on the machine, the tasks extracted from this file are also added to the list present on the USB drive in order to pass on to other systems. The list is digitally signed and has a custom format and contains information required for a task. Details of the tasks are extracted and dropped on the victim's machine in the same 'task_pub.dat' file. The XMKR module then proceeds to perform each task.
- Task-specific blocks of data which store the status of each task at every stage of its operation, as well as the task result.

### Tasks performed by XMKR

The XMKR module can handle a number of tasks, which are listed below:

- Upload a file to the USB drive.
- Download a file from the USB drive.
- Run a file.
- Execute a command.
- Run a process with specific commandline arguments.

The ability to perform such tasks can help with the exfiltration of sensitive information collected by executing system commands on a target machine, or stealing files by uploading them to a USB drive. It also allows the attackers to send new malicious modules to air-gapped systems by deploying them on a USB drive paired with relevant tasks to download the file to a machine and execute it as a process.

The two malicious modules described so far seem to be aimed at compromising air-gapped systems or systems with limited internet access. Apart from these modules, we have found a number of other malicious modules developed in earlier phases of the attack campaign. These modules have been used by the attackers to serve various purposes.

## EXPORTUSB

The attackers utilize this module to copy files from a target USB drive into the local machine. The files are copied in an encrypted form and, in some of the versions, the copied files are compressed into a single password-protected archive file using the rar.exe tool.

We have found several versions of this module developed from September 2021 until the middle of 2022, and all of the samples are packed with the VMProtect packer. Table 1 shows the version names extracted from the samples together with their compilation timestamp.

Like the XMKR module, the 'ExportUSB' module waits for a target USB drive to be plugged into the machine. The process of attempting to access the unlocked USB drive is similar to the XMKR module.

Once this module gets access to an unlocked secure USB drive, it collects the list of the files and directories together with the system time in a log file. It also makes an encrypted copy of each file present on the drive on the local machine in a hidden directory. The versions that archive the files use the rar.exe CLI tool with the password '1qazxcde32ws' throughout all the samples.

We have found several versions of this module developed from September 2021 until June 2023 and all of the samples are packed with the VMProtect packer.

| Version | Compilation timestamp |
|---|---|
| Version 09_12 | 2021-Sep-14 09:14:18 |
| Version 09_27 | 2021-Sep-29 13:43:35 |
| Version 10_26 | 2021-Oct-26 05:21:08 |
| Version 22_01_26 CopyMode0_X | 2022-Jan-26 19:45:46 |
| Version 22_02_14 CopyMode0_X | 2022-Feb-14 02:03:13 |
| Version 22_02_14 CopyMode0_X | 2022-Apr-27 09:41:02 |
| Version 22_02_14 CopyMode0_X | 2022-Apr-27 09:41:02 |
| Version 22_06_12 Sector_RW(Mutex)_Test_X | 2022-Jun-20 04:12:14 |
| Version 23_06_12 CopyMode0_X | 2023-Jun-12 04:01:22 |

*Table 1: Version names extracted from the samples and their compilation timestamp.*

Based on the compilation timestamp of the samples, which mostly preceded the development of the XMKR and trojanized UTetris.exe, we suspect that these modules were part of an earlier phase of the attack campaign. And in a later phase, the attackers developed the capability to target air-gapped systems by compromising the secure USB drive.

### GETUSB

The malware is used to collect all the files from all the USB drives (secure or otherwise) that are connected to the infected machine. We call this malware 'GetUSB' based on the project name found in the PDB path left in some of the samples of such a module.

We have found two versions of this malware. Version 1 is used to copy all the files and directories present on a connected USB drive. Samples of this version were compiled from the end of 2021 to February 2022.

In version 2, the attackers added a function to collect additional metadata information from the files on the USB drive. This version looks for all removable drives on the machine and makes a list of the Volume Name of all of these drives. The attackers developed version 2 in late February 2022.

The timestamps of the compilation and detection logs for samples of this module suggest that it was used in an even earlier phase of the attack campaign before the attackers had gained knowledge of the type of secure USB drive used in the target environments.

### USBDC

This module is responsible for collecting detailed and highly specific information about the USB drives connected to a system.

The earliest samples of this module that we found were compiled as far back as 2019. However, the development and deployment of new samples of this module have been ongoing ever since. The module examines all the drives of the victim's machine and collects information about them. This information is stored in a file and shared with a remote server. Even though the malware collects information about all types of connected drives, it seems to be particularly interested in removable drives.

For a removable drive or a drive with an unknown type, the malware collects the following information and stores it in a hidden file by issuing a IOCTL_STORAGE_QUERY_PROPERTY request to the drive:

- Version
- TotalSize
- DeviceType
- DeviceTypeModifier
- RemovableMedia
- CommandQueueing
- Vendor Id
- Product Id
- Product revision
- Serial number
- Bus type

It also goes on to extract information about the physical disk geometry of all removable drives by issuing a IOCTL_DISK_GET_DRIVE_GEOMETRY request to the drive. It stores this extra information in a separate file. The type of information it collects is listed below:

- Number of cylinders
- The type of media
- Number of tracks per cylinder
- Number of sectors per track
- Number of bytes per sector

Some of the earlier samples of this module have hard-coded strings for commands to copy the '.doc' and '.docx' files from USB drives using the 'xcopy' system utility.

This module seems to have assisted the attackers in getting a better understanding of the target secure USB drives and is likely used for general USB reconnaisance.

## ACROSHELL

AcroShell is a malicious shellcode executed on a victim machine via a DLL side-loading technique. A malicious DLL obfuscated with a virtualization-based software is executed via a vulnerable legitimate executable file. The DLL module injects a piece of code to the EntryPoint of the executable file to read, decrypt and load the shellcode from an encrypted file.

The main responsibility of the final payload is to establish a communication channel with the attackers' servers to facilitate various malicious operations.

This module has been used since at least the end of 2017 and has been used for the following purposes throughout the years:

- Deploy USBDC, GetUSB and also ExportUSB modules throughout the years.
- Deliver 'XMKR' to initialize the process of compromising secure USB drives to a number of victims.
- Perform lateral movement via executing various commands to find nearby systems and copy malicious components, create registry keys for persistence, create and start services on the remote system and also execute malicious components on remote machines in various cases.
- Deliver the 'task_pub.dat' file, which is an important component of the 'XMKR' malware used in compromising potentially air-gapped systems via trojanized USB drives.
- Exfiltrate data from compromised machines. We have observed this module send log files created by different malicious modules described in this report as well as files copied from compromised USB drives to the attackers' servers.
- Steal specific document files of the attackers' interest from victim machines. In some cases, we observed that the attackers stole files containing licence information of security softwares or passphrases for Wi-Fi networks. In other cases, document files with filenames suggesting the sensitive nature of the content were stolen from compromised machines.

## CONCLUSION

The attack campaign described in this paper has been ongoing for a few years now, with the attackers constantly improving their toolset and adding new malicious modules. The amount of time and resources spent by the attackers in understanding their target environment and the specific software and hardware of the target systems and networks displays a high level of persistence.

Compromising the software used to 'securely' transfer data via proprietary USB drives developed and used only by government employees of the targeted country is an ingenious way to reach air-gapped systems and protected information.

Discovering other malicious tools used by this threat actor and examining the evolution of the attackers toolset led us to understanding various phases of this campaign.

Even though the two malware components involved in compromising the secure USB drives are the most sophisticated among all of the malware described in this paper, the other malware components involved in the attacks also show a great level of stealthiness. Limited propagation of the attack, and the virtualization-based software packer (VMProtect) used for malware components made it possible for the attackers to stay below the radar for many years.

Considering all of the above, we believe that these attacks have been carried out by a highly skilled and resourceful threat actor interested in espionage activities in sensitive and protected government networks.

As of now, no attribution links have been established with any existing threat actor. With this attack campaign still ongoing, we will continue to track this threat actor and expect to see more sophisticated attacks from them in the future.