

The Odyssey and Trojans again: MovieReaper attacks users in multiple countries through compromised torrents

[SL securelist.com/moviereaper-malware-torrent-odyssey-solana/121344](https://securelist.com/moviereaper-malware-torrent-odyssey-solana/121344)

Konstantin Isakov, Pavel Cheremushkin

September 17, 2026

[GReAT research](#)

6 minute read



Introduction

Torrent trackers have long been abused for distributing malicious software, disguised as popular films, games, and other content. Our previous research has shown that cybercriminals repeatedly turn to torrents as an initial infection vector, using trojanized cracks and installers to reach a large number of users. Installation guides for pirated software routinely instruct users to disable their antivirus, conditioning them to ignore the potential threats they are inviting onto their computers.

During our analysis of malware that leverages blockchain networks for its C2 infrastructure, we discovered a previously unknown modular, multi-stage framework that we dubbed MovieReaper. This report details the new crimeware campaign that began with the mass infection of users via compromised torrent tracker file storage. We have identified several hundred victims, including both individual users and organizations in multiple countries, such as Russia, Türkiye, Japan, Kenya, Uganda, and Colombia, as well as in several European

countries like Spain, the Netherlands, Belgium, and Germany. We analyze the techniques for evading detection by security and sandbox solutions, and examine the capabilities of the modular framework.

Kaspersky products detect this threat as HEUR:Trojan.Win64.Agent.gen.

Technical details

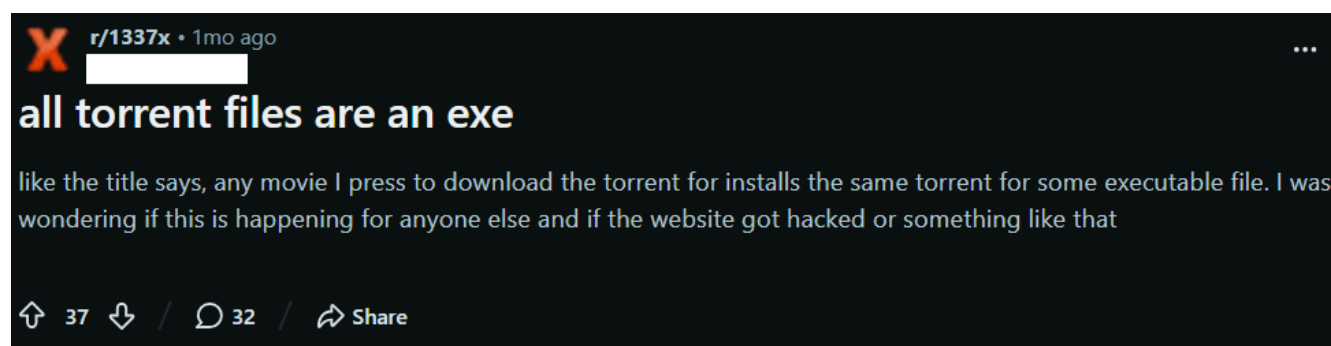
Background

In mid-August 2026, during our threat-hunting efforts, we identified a large-scale infection campaign involving previously unknown malware disguised as popular movies. The campaign affected both individuals and organizations across multiple countries. Our initial analysis revealed a common denominator: all of the victims had used torrent trackers. This finding prompted us to investigate the campaign further and analyze its distribution mechanism, overall scope, and unknown malware implants.

Initial infection and spread

Compromised torrent trackers are the primary vector used to distribute malware. During our investigation, we identified multiple user reports describing suspicious files being downloaded instead of the intended content.

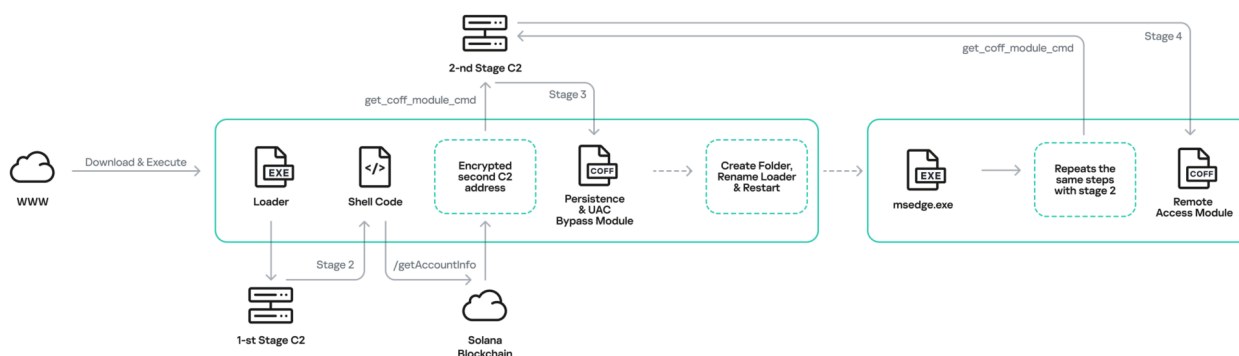
For example, a user of a popular movie torrent tracker reported the following case on Reddit:



Further analysis of the attack revealed that instead of compromising torrent trackers, the threat actor modified a widely used public repository of torrent files, `itorrents[.]org`. As a result, the trackers that relied on the repository began inadvertently distributing malicious torrent files to their users. This approach is particularly powerful because it lets the threat actor reach users of multiple trackers without having to compromise each platform individually.

As of the publication date of this report, the archive remains compromised. When a user attempts to download a torrent via a magnet link, the legitimate torrent archive returns a different torrent file. This malicious torrent leads to the download of the malware loader, used to deploy a framework that we dubbed MovieReaper.

The loader initiates the infection chain shown in the diagram below. Each stage of the chain is described in detail in the following sections.



Malware implants

The infection chain consists of several steps, where only the initial one is dropped on the disk before it is executed to avoid detection. The malware itself is not heavily obfuscated, apart from strings being encrypted with a custom stream cipher. Most of the countermeasures were aimed at avoiding detection by AV sandboxes.

Step 1: Loader

The most popular initial executable was distributed through torrent trackers under many different names (for example, the odyssey (2026) [1080p] [webrip] [5.1].exe), but the file hash (MD5: A0B13781EDD7CFDAB13D79AFF3C83C1) was identical across all downloads. We have seen multiple different loaders, where the executable file disguises itself with a long filename and an icon of some well-known application. Most of the filenames are rather long, presumably to hide the EXE extension at the end.

After the user manually starts the application, it establishes a global mutex to ensure that only one loader is executed at a time. We have seen several mutex variations in our samples, containing a randomly generated string (in the example Global\fnuLSktzSqVVLXHU). This executable then performs a series of operations in order to avoid detection by AV sandbox solutions.

While performing those operations, the malware avoids making `LoadLibrary` and `GetProcAddress` calls to acquire the addresses of required functions. Instead, it searches for loaded libraries by traversing the double-linked list taken from the `Ldr` field of the PEB and then performs manual parsing of the loaded DLL to calculate the function address.

After all initial checks have passed, the binary prepares to establish a network connection to a C2 web server to download the shellcode, map it to the RWX memory and execute. While doing so, the loader decodes the domain name `https://deadhub[.]org` and if connection to it has failed, it uses the IP address `http://193.23.118[.]155` as a fallback and connects to it using plain HTTP. The malware chooses a random group of strings and uses them as a path in the HTTP request to download the parts of the shellcode.

Example URLs:

```
/cloud/v192.4/ui/sync-status-icons.png  
/cloud/v192.4/onboarding/welcome-bg.jpg  
/cloud/v192.4/ui/file-preview-placeholder.png  
/cloud/v192.4/shared/link-banner.jpg
```

While mapping the address space and executing the shellcode, the loader registers a vectored exception handler and rewrites the handler address in memory in order to perform a debug break. This, will not crash the program, but instead redirect controlflow to the function that actually makes a raw `NtProtectVirtualMemory` syscall (via a previously located “0x0F 0x05” syscall instruction inside `ntdll`). Then it calls the undocumented `ntdll` function `EtwpCreateEtwThread`, which is a popular alternative to `CreateThread` for code execution, and executes the shellcode.

Step 2: Shellcode

The second stage of this malware performs an HTTPS request to the Solana blockchain at the endpoint `/getAccountInfo` endpoint for the account `6pnDGAiHgyPdmckM5Qt1YbanGzrX43WLEU159nRaNLdM`. The data in field of the response contains the base64-encoded address of a second C2, which is encrypted with a static XOR key located within the shellcode itself. To store data in this account, attackers used a simple Solana program (address: `CSiY8bQLBYPdfPkwipBzH6sijTVQVVsA279JQdvwHtL`).

By using the Solana blockchain network as a distribution layer of endpoints for the next stage, the attackers can increase the stability of their campaign and resist takedown efforts by defenders.

The second stage payload communicates with its C2 server strictly through HTTPS via a TLS-pinned certificate using the `nanopb` protobuf library as a container for transferred data. The

main logic of the stage 2 implant contains several initial commands, where the most important is the one that parses the COFF file, loads it into memory, and executes the `module_init` function from it. This provides a convenient interface for extension of the command list, which brings us to the next stage of the payload.

Step 3: UAC Bypass and persistence

Notably, the recovered modules were compiled with symbols, which accelerated reverse engineering.

After receiving the next stage from the second C2 server, the newly loaded module performs several tasks right in the `module_init` function.

Stage 3 performs UAC Bypass and achieves persistence using public techniques, masquerades the original binary as

`C:\ProgramData\Microsoft\Windows\Telemetry\msedge.exe`, and restarts itself.

The respawned process starts with the initial loader, but with a special command-line argument, which allows it to skip most of the anti-sandboxing checks and proceed straight to downloading stage 2. The executable proceeds with the same steps as before, but this time, instead of downloading the persistence and UAC bypass module, a new one is downloaded from the second C2 server. This is because the malware uses the beacon to send a flag to the remote server, indicating whether the implant is running from the Telemetry folder, which allows the C2 to distinguish between first-run and respawned instances.

Step 4: The final implant

The final module (“file manager”) exposes 21 commands that give the operator filesystem access on the victim host. This allows remote the operator to download, upload, and read files on the system; list and enumerate directories; manipulate files with create, copy, rename, move, delete, `chmod`, and `symlink` commands; and use preview and thumbnail commands to exfiltrate previews of images and files before actually extracting them.

We suspect that other modules may be loaded on demand if requested by the operator.

Infrastructure

During this malware campaign, attackers are using various commercial hosting providers for their C2 infrastructure (see the IoC section for details). Furthermore, as noted above, the campaign leverages the legitimate Solana blockchain via the RPC endpoint `api.mainnet.solana.com` to deliver the address of the second-stage C2 server to the malware. This approach provides the attackers with decentralized storage for C2 addresses,

adding an additional layer of resilience and making it more difficult for defenders to disrupt the campaign by simply blocking the IP addresses of the C2 servers.

Victims

The observed campaign targeted both individuals and organizations across Europe, Asia, and Africa, with infection attempts identified in Russia, Spain, Germany, Finland, Türkiye, Japan, Nepal, Kenya, Tanzania, Ghana, and other countries. The targeted organizations span a wide range of sectors, including enterprise, government, IT, consulting, retail, transportation, and agriculture.

Conclusions

Our research uncovered activity by the same actor dating back to October 2025. Although the campaign has evolved over time, with the malware authors expanding their arsenal and making the loader harder to detect, the pattern remains the same. This includes encoded strings, parts of shellcode downloaded through plain HTTP, and several techniques used to avoid sandboxes and virtual machines. We will continue monitoring this actor's activity to catch new potential threats.

The first stage offers the clearest opportunity to disrupt this campaign: it relies on a single specific domain name and a single IP address to serve the shellcode, so taking down this server would prevent subsequent infection stages. This includes the second-stage payload, which uses the Solana blockchain network for a C2 and is therefore more resistant to conventional infrastructure takedowns.

However, this framework's self-containment, modularity and in-memory execution has potential for reuse in later campaigns with minimal rework.

Indicators of compromise

File hashes

[4334BBAEA8DE33BF9D45E9B4E4E3BC2](#)
[4843F9FAFCAE492F11E2D4D33DBB4CDD](#)
[5310CABAE3FBE6DB8742849B588093F9](#)
[A0B13781EDD7CFDAB13D79AFFF3C83C1](#)
[70060341CAF3338697A7DDFE0FB62875](#)
[AD4643EEA15AC286FA47D1131F9EF756](#)

[D0B967571AC8A3863C7F324BF5BDE99C](#)
[D88D550D0FB8E60CFFFF3EA61FF7A067](#)

File paths

%ProgramData%\Microsoft\Windows\Telemetry\msedge.exe

Mutexes

Global\E4AyDKzvEhe2hgAr Global\fnulSktzSqVVLXHU

Domains and IPs

First-stage C2: [deadhub\[.\]org](#)
[193.23.118\[.\]155](#)

Second-stage C2: [208.64.33\[.\]90](#)
[208.94.246\[.\]53](#)

- [Blockchain](#)
- [Malware Descriptions](#)
- [Malware Technologies](#)
- [MovieReaper](#)
- [shellcode](#)
- [Torrent](#)
- [Trojan](#)

Table of Contents

- [Introduction](#)
- [Technical details](#)
 - [Background](#)
 - [Initial infection and spread](#)
 - [Malware implants](#)
 - [Step 1: Loader](#)
 - [Step 2: Shellcode](#)
 - [Step 3: UAC Bypass and persistence](#)
 - [Step 4: The final implant](#)
- [Infrastructure](#)
- [Victims](#)
- [Conclusions](#)
- [Indicators of compromise](#)

- [File hashes](#)
- [File paths](#)
- [Mutexes](#)
- [Domains and IPs](#)

