Assignment: Special 1

Justin Kennelly

University Of Arizona

CYBV 454 MALWARE THREATS & ANALYSIS

Special 1: iOS Malware

Professor Galde

April 19, 2022

**MACDownlad.o**

* MD5 Checksum Value: **b2de03de5af8f9e41614953d4c51ebdd**

**MACDownload2.o**

* MD5 Checksum Value: **fdd6fb2b1dfe07b0e57d4cbfef9c8149**

**Question 1**

***What is the name of this malware according to VirusTotal / HybridAnalysis / Intezer?***

***Graphical user interface, application

Description automatically generated***

*Figure 1: VirusTotal results for MACDownlad.o*

*Graphical user interface, text, application, email

Description automatically generated*

*Figure 2: VirusTotal details for MACDownlad.o*

Graphical user interface, text, application

Description automatically generated

*Figure 3: HybridAnalysis results for MACDownlad.o*

After running MACDownlad.o through VirusTotal.com I get hits on 31 out of 58 antivirus softwares as it being a malicious file. Looking closer at the results I instantly see that a reoccurring name keeps appearing called “Silver Sparrow” (Figure 1). In Figure 2, I went to the Details tab of VT and looked at the information. There was not much information available but I did notice that this file type is unknown despite it being labeled as a .o or MACH-O file. Next, I go to HybridAnalysis.com and run the file through there. Still I get results of an unknown filetype and still get pretty confident hits on antivirus software that tells me this is in fact a malicious file (Figure 3).

Graphical user interface

Description automatically generated

*Figure 4: VirusTotal results for MACDownload2.o*

*Graphical user interface, text, application, email

Description automatically generated*

*Figure 5: VirusTotal details for MACDownload2.o*

*Graphical user interface, application

Description automatically generated*

*Figure 6: Virus Total community response to MACDownload2.o*

I move on to MACDownload2.o and run the file through VirusTotal.com. I get 34 out of 56 hits as malicious software from anti-virus software (Figure 5). Again I see the name for this file reoccur: “Silver Sparrow” in the antivirus results (Figure 5). Next I go to the details tab in VT, where I get much more data from MACDownload2.o. First, I see that the filetype here is for Apple Software Package (Figure 6). I also notice in the Names field there is a hash correlating to MACDownload2.o shown as update.pkg (Figure 6). Finally, I go to the community tab in VT and see that several people are referring to this file as Silver Sparrow. I am confident that both of these files are related to a piece of malware called Silver Sparrow at this point.

***What does the internet say these files do?***

***Graphical user interface, text, application

Description automatically generated***

*Figure 7: Snippet from Redcanary.com/blog/clipping-silver-sparrows-wings/*

According to the internet, MACDownload2.o a file that is associated with Mac OS malware that infected thousands of devices and over 153 countries. At the time of the red canary article there had been no payloads associated with the Silver Sparrow malware samples analyzed. One of the samples analyzed, seen in Figure 7, has an MD5 hash that matches MACDownload2.o exactly. For this sample, it is identified as Silver Sparrow Version 2. This is an installer package for Mac OS systems that includes a MACH-O binary compiled for Intel x86\_64 and M1 ARM64 architectures.

Graphical user interface, text, application

Description automatically generated

*Figure 8: VirusTotal details of MACDownlad.o Closer look*

**

*Figure 9: Changed file extension of MACDownlad.o*

*Text

Description automatically generated*

*Figure 10: Extracted files from MACDownlad.o*

Looking back at MACDownlad.o I noticed in the VirusTotal results that there was a section in its’ properties named **TrID** the value for the file was: CPIO archive (portable) 100% (Figure 8). This led me to look into CPIO which turns out to be an archive file. I then went to the file in my folders and changed the extension to .cpio (Figure 9) and used 7Zip to extract the archive. I was then able to see an extracted folder called tasker.app with several documents inside (Figure 10).

*Graphical user interface, text, website

Description automatically generated*

*Figure 11: redcanary article on Silver Sparrow malware*

*Graphical user interface, text, application, email

Description automatically generated*

*Figure 12: MD5 Hash for tasker file found in tasker.app>Contents>MacOS*

Reading further into the Redcanary article I noticed that the V2 version of Silver Sparrow (the one that correlates to MACDownload2.o) it gives reference to a tasker file and hash that is responsible for targeting the M1 ARM64 architecture using Silver Sparrow (Figure 11). I noticed a file called tasker when I extracted the MACDownlad.o turned MACDownlad.cpio file and looked at its contents. I then ran the tasker file in WinMD5 to get its hash value which was an exact match to the one referenced in the article.

*Diagram, timeline

Description automatically generated*

*Figure 13: Image from Redcanary on execution of Silver Sparrow V2*

Reading further about Silver Sparrow V2 I come to understand that MacOS threats are usually deployed as a self-contained installer either as a PKG or DMG. These malicious files usually pose as a trusted piece of software that a victim would not suspect as being malicious. With V2 shown in red in Figure 13, the main difference from V1 is its bystander binary -> tasker. The way it works is the installer uses the MacOS API to execute commands using pre or post install scripts. By executing the JavaScript in the packages Distribution definition XML file the telemetry pattern is changed where you have a parent process: Installer and the Process: bash. According to the internet these files have the ability to run JavaScript cod in pre or post installation and then achieve connection to a C&C server. Once connected the LaunchAgent Plist XML files continue to launch scripts and wait for commands from the maintainers.

***Are these files the same, or are they different? If they are different, how?***

***Chart

Description automatically generated***

*Figure 14: Placeholder from bystander binary Credit: RedCanary*

As it turns out these files are very different. They are used in the same malicious attack however they serve different purposes in carrying that out. As seen in Figure 13, the updater (MACDownload2.o) is used to act as the distributor of the installation which also executes the malicious JavaScript code. MACDownload2.o is also responsible for executing the plist launchagents to establish persistence and executes shell script to the C&C to receive commands. MACDownlad.o seems to be the bystander binary that is compiled for both Intelx86\_64 and M1 ARM64 architectures. Red Canary identifies this binary as a placeholder for now, it simply displays “You did it!” in a MAC OS window (Figure 14).

**Question 2**

***Run strings on these files, provide analysis on each file, and compare the two.***

***A picture containing text, screenshot

Description automatically generated***

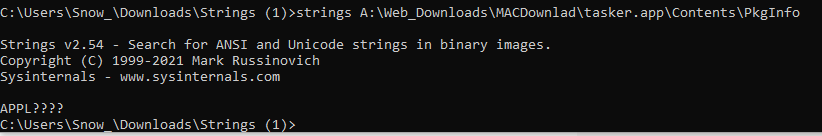
*Figure 15: Strings from MACDownload2.o*

The strings from MACDownload2.o appear to be pretty uninteresting. It could be that the file is packed or obfuscated and the gibberish of characters is just encoded. Looking at the structure of the characters and how they are sectioned of with brackets it is quite possible there is something more interesting beyond what is seen (Figure 15).

Text

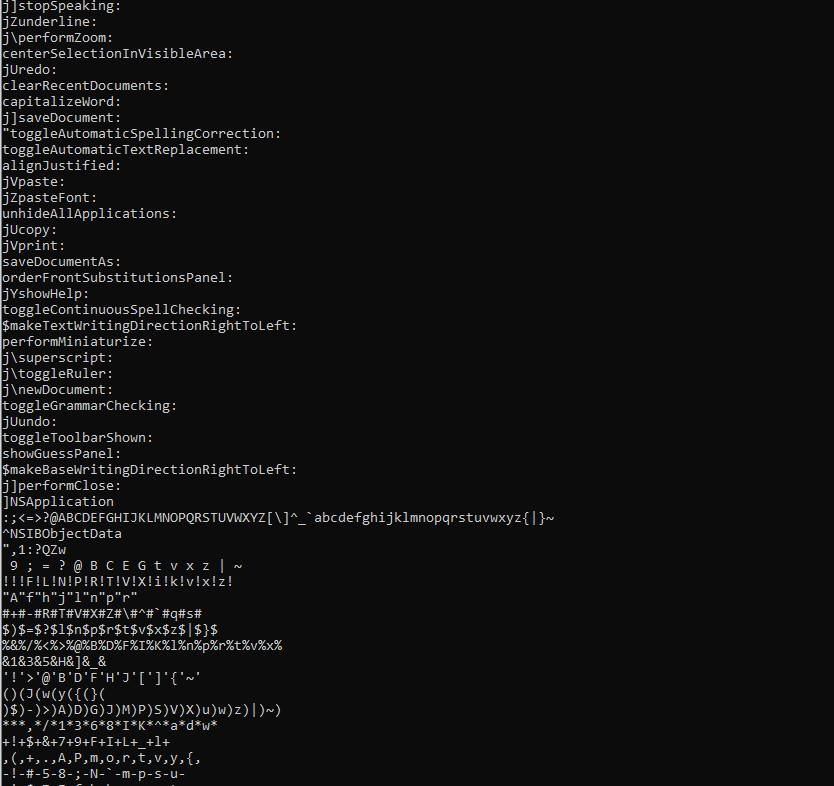
Description automatically generated

*Figure 16: Stings from extracted MACDownlad.cpio > Info.plist*

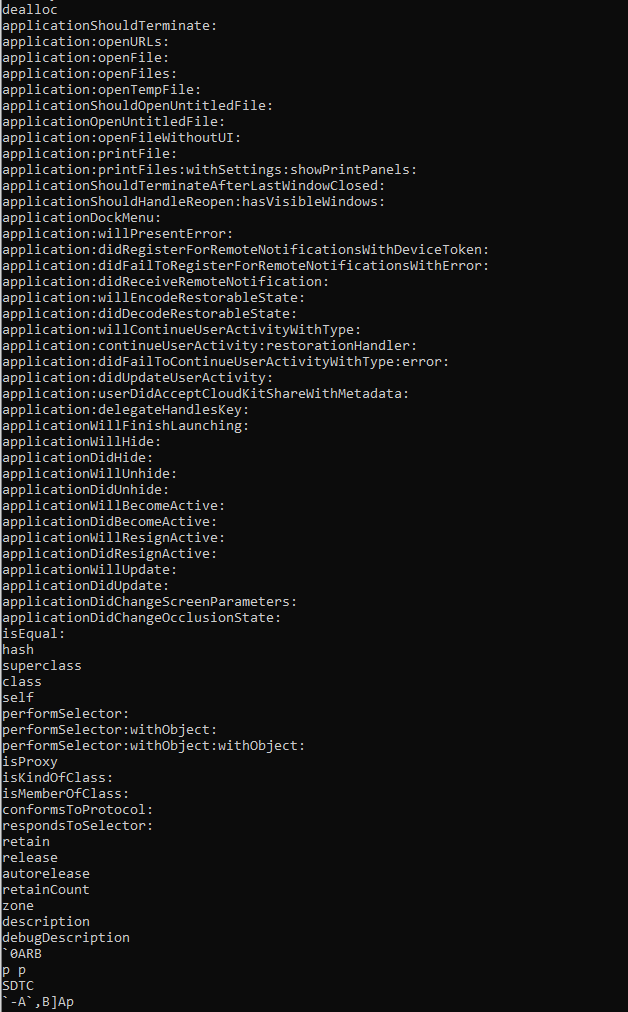
**

*Figure 17: Stings from extracted MACDownlad.cpio > PkgInfo file*

*Text

Description automatically generatedFigure 18: Stings from extracted MACDownlad.cpio > MainMenuNib 1*

*Figure 19: Stings from extracted MACDownlad.cpio > Main.MenuNib 2*

**

*Figure 20: Stings from extracted MACDownlad.cpio*

When extracting the original file: MACDownlad.o with the strings command nothing is returned so then I turned to focusing on the individual files. Running strings on the InfoPlist file there were only a few strings which appeared to be pointers to other files in the directory. Bplist00 which appears to be a binary property list in the version 00, it is the most common version in Apple OS (Figure 16). I did not find much information on the other strings in that file. The PkgInfo file had one string which shows APPL???? (Figure 17). Again, not sure what this is used for. The strings in Figure 18 are for the MainMenuNib file. Within those strings the tasker is referenced quite a bit. Additionally it appears there are strings related to functionality of the program. In the strings for MACDownlad.cpio we get a view of everything included in all the files within the directory (Figure 20). I find several application commands as well as some language that looks like it is from the Swift programming language like: isEqual, isProxy, isMemberOfClass, isKindOfClass, retain, release, etc.

***Any hints on what these files are able to do? Provide analysis for why you gave your answer.***

Graphical user interface, application, Word

Description automatically generated

*Figure 21: Extracted files in Ubuntu*

*Text

Description automatically generated*

*Figure 22: Distribution file from MACDownload2.pkg*

*Text

Description automatically generated*

*Figure 23: Function Installation Check*

I decided to open my Linux VM to better access these files. After some thought I realized that the MacDownload2.o is technically the package file used in distributing the functionality of this malware so I decided to change the extension to MacDownload2.pkg in Ubuntu (Figure 21). I then opened the archive with archive manager and was able to access more files. I noticed the Distribution file so I opened it in the text editor to get a better look (Figure 22). The header of the file has several linkers to the tasker.pkg (Which holds the payload), the bundle whose ID is com.hello.tasker. Towards the end of the headers we see script tags and a function called installation\_check (Figure 23). Installation check is where our system.run is called for execution of several tasks. One of these tasks is the agent creation seen next which creates the persistence on the device using the property list buddy. Next we see a URL: var url <https://specialattributes.s3.amazonaws.com/applications/updater/ver.json> this is the url used to talk to the C&C controller. After that many AppendLine() functions are called which appear to be writing files to /usr/bin and remove lines from /tmp. This appears to play out as explained by the research found when looking to the internet to find more information on the discovered files and hashes.

**Question 3**

***When were these files compiled?***

***Graphical user interface, text, application

Description automatically generated***

*Figure 24: file data from tasker.pkg*

The files appear to be compiled: 12/14/2020 13:50:00 UTC. One file has a date of 12/31/1969 17:00:00 UTC and I am not sure why. Maybe this could be due to purposeful obfuscation.

***Are these files packed or obfuscated?***

*Graphical user interface, application, Word

Description automatically generated*

*Figure 25: Files associated with MACDownload2.o & MacDownlad.o*

The files appeared to be obfuscated in the sense that they were given different extensions. Once I was able to identify their purposes further I was able to apply the appropriate extensions and then glean more information to the files, packages, etc.

***Can you identify any host-based indicators?***

***Text

Description automatically generated***

*Figure 26: paths and identifiers to tasker package and app*

***Text

Description automatically generated***

*Figure 27: Directory paths used in the malware execution*

***Text

Description automatically generated***

***Text

Description automatically generated***

*Figure 28: More directory paths used in the malware execution*

Host Based indicators have been discovered through this process:

MD5 hash: fdd6fb2b1dfe07b0e57d4cbfef9c8149 (MacDownload2.o)

MD5 hash: b2de03de5af8f9e41614953d4c51ebdd (MACDownlad.o)

Tasker.app

Tasker.pkg

/usr/libexec/PlistBuddy

/usr/bin/curl

~/Library/.\_insu

***Can you identify any network-based indicators?***

******

*Figure 29: url variable from Distribution file code*

The url, seen in figure 29, is a network-based indicator pointing to the potential C&C server that issues commands to the malware on a target device.

**Question 4**

***Look up the malware and identify a blog that talks about this malware.***

***Graphical user interface, text, application

Description automatically generated***

*Figure 30: Red Canary website banner for the Silver Sparrow article*

***Do you agree with the assessment?***

I believe the blog referenced is most likely the one I have already come across in my research. I agree with the assessment done by Red Canary their analysis was thorough and well done. The technicality analysis of Silver Sparrow makes sense to me and then experience all the same findings assured me that this was an accurate assessment.

***Do you come to the same conclusion?***

Like the blog post, I also find it strange that the malware seems to have a lot of unfinished functionality. I don’t understand why something like this would be released to the wild while being seemingly incomplete. The placeholder of the binary seems like it was intended to carry out further functionality but simply does not. Also like, Red Canary points out, the use of ~/Library/.\_insu is odd in that it wipes itself but does not carry out any intermediary functions. I find the existence of Silver Sparrow to be a mystery just like Red Canary, however, I feel like it may have an underlying purpose or reason for existence then it to be strictly malicious.

I think there are two scenarios at play here: one would be that this malware exists by accident. Not that it was created by accident but distributed as widely as it was by accident. I feel the creator or creators of Silver Sparrow could be researchers in MacOS security (seeing as how it is typically regarded as a very secure operating system). Testing the boundaries of the OS and seeing how it can be threatened. This could or could not be for malicious purposes. The fact that the malware is not fully complete leads me to believe that it was being tested and then reached much further than intended before it could be completed and then subsequently discovered.

The other scenario could be that there are bigger players involved in this then are currently thought of. Not to sound too much like a tin foil hat person, but this could be a tool that was being researched by government or nationstate actors in an attempt for surveillance or other purposes. It is not farfetched that government agencies develop sophisticated cyber tools not intended to be leaked. It has happened before, ShadowBrokers, leaking NSA toolsets that were previously unknown which later led to the WannaCry ransomware attacks. Either way the thing I do agree with Red Canary on is that this malware is very intriguing and certainly a mystery.