An Investigation into Incident Response for the Modern macOS

Jack Clark
School of Design and Informatics
Abertay University
DUNDEE, DD1 1HG, UK

2nd October 2019

Abstract

Context

As the market share of macOS increases, so too does the number of malware designed specifically for it. Due to this Incident Response is becoming more important as the attack surface of said malware has increased exponentially over recent years.

Aim

To develop software to assist in the Incident Response process for macOS by automating said process and producing an easy to read report. Additional aims include implementing a malware detection system and a user-specified flag to highlight specific files while gathering.

Method

A collection of scripts, in Python, Bash or another language will be developed to create a tool to be used during Incident Response. Research of common Incident Response techniques will be performed thoroughly beforehand so to develop an understanding of what must be included in the end product.

Results

The final tool will be tested against several malware samples to ensure that it can provide the relevant information, when gathering data. If the produced document is enough to allow the user to conclude the cause of the compromise, then the test will be successful.

Conclusion

This project will demonstrate how Incident Response techniques for other Operating Systems can be applied to macOS and highlight the effectiveness of an automated tool for data collection during Incident Response.

Keywords

macOS, Incident Response, Cybersecurity, Research, Defense.

1 Introduction

Over recent years, the market share of macOS-capable devices has significantly increased, both in the consumer and business markets. Alongside this increase in the market, there has been an exponential growth of bad actors developing macOS specific malware, 93,924 different malicious macOS programs in 2018 in comparison to 28,925 in 2017 (See Figure 1).

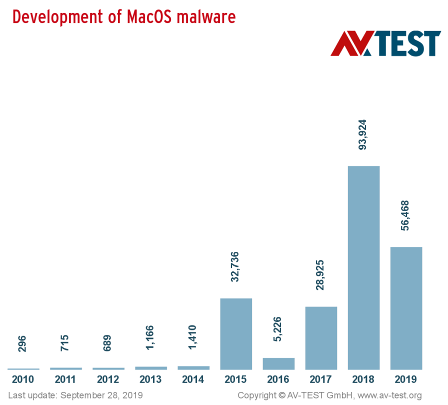

Figure 1: Malware for macOS Over a 10 Year Period(*Malware Statistics & Trends Report* 2019)

Malware comes in a vast array of forms and has become increasingly complex in how it achieves its goals, allowing for increasingly successful attacks. Malware attacks affect both individuals and businesses, and in the case of businesses can cause a large amount of damage due to the likes of data and money loss alongside downtime.

To assist in reducing the damage that could be caused by downtime, a business may employ a company or individual to perform Incident Response (IR) when an attack or intrusion is detected. Their role includes collecting data from a compromised device or network and then analysing the data to determine how the compromise occurred. This analysis, when completed, can then give the employer a strategy for preventing the same compromise to occur again.

2 Background

Incident Response is a much researched topic, however, most papers discuss IR for the Windows Operating System (OS). There are no papers researching IR for macOS however, there is a vast array of research into other aspects of macOS in particular malware that is designed for macOS.

2.1 Literature review

This research was inspired by multiple talks at Objective by the Sea v2.0 (OBTS), "the Mac Security Conference"[1]. One of the talks presented at OBTS was by Richie Cyrus, "*Detecting macOS Compromise with Venator"* (Cyrus 2019), discussing an open-source tool that he developed called Venator[2]. This tool is written in Python and assists with data gathering during IR on macOS. While Venator is a fantastic tool, Apple recently announced that it would stop including scripting languages with its upcoming macOS installations[3], and that if a scripting language is required then it will need to be installed separately. This will provide an issue if Python has not installed on the victim device.

Due to there being scarce research for IR on macOS, the inspiration for this topic came from said scarcity, IR for Windows and Venator. The techniques discussed in research papers for IR for Windows can commonly be used for macOS, with some changes. The paper *"On-site investigation methodology for incident response in Windows environments"* (K. Lee, S. Lee and C. Lee 2013) discusses the common techniques and tools used during Windows IR in great depth. The tools discussed are all Windows-specific however, macOS has tools that are built-in that provide the same features. For example, the article provides `Process Explorer` as a tool to analyse currently running processes whereas macOS has the tool `Activity Monitor` that performs the same task. The paper also suggests using additional tools that need to be downloaded. This can cause an issue if there is no internet connection on the victim machine or if it has been disabled due to the incident. The proposed tool will aim to only use built-in tools so that this issue can be avoided. In relation to the techniques presented, these provide a useful guideline for what to investigate and collect for a Windows system and can be adapted for macOS.

As previously mentioned, there are many research papers on macOS malware and how it operates. These papers are incredibly useful as it gives insight into where malware can be stored, how it can be installed and the effects it can have. The paper *"Methods of Malware Persistence on Mac OS X"* (Wardle 2014) discusses where malware can be stored so that it can be persistent and how it can ensure that it is executed when the system is booted. This is vital information as different malware may use the same directories and techniques to create persistence and the research providing this gives a list of files and directories to investigate. Although this particular paper is from 2014, it is still relevant as the persistence and start-up techniques are still used to date. The paper also discusses the built-in security applications of macOS that can provide useful if a file, in particular, is collected and needs to be investigated further.

The same topic is discussed in the paper *"Detecting objective-C malware through memory forensics"* (Case and Richard 2016), however it focuses on malware created using the Objective-C (Obj-C) language, developed by Apple, and so has many API's that only exist in Obj-C making it typically more powerful than other generic malware. The techniques described in this paper allows for the detection of more advanced malware and so are vital for this project. The paper is also more recent than the previously mentioned paper and so can give techniques that are more relevant.

The papers above show that although there is not any papers on IR for macOS, the resources to support it are there. The techniques from Windows IR, and research on malware for macOS provides critical insight into what to investigate and gather during IR.

3 Method

3.1 Research

Before commencing with the development of the prototype, it is first vital to understand the techniques of Incident Response. This includes what needs to be investigated and so gathered, and to complete this reading previous research needs to be a priority. A number of publications and macOS security researchers have already been identified by the author as having work that is relevant to this project. Decisions also need to be made on other questions that are key to the development of the prototype.

3.1.1 Development Language

The previously discussed tool Venator is developed using Python, and the issues with that were discussed previously. Due to these issues, the prototype would ideally be developed in Bash as it is native to macOS. With Bash being bundled with macOS it means that it will always be installed, however in the upcoming version of macOS (10.15) Bash has been replaced as the default shell with Zsh. The decision needs to be made as to whether to develop with Bash, which offers backwards compatibility and familiarity, or Zsh, allowing for developing in the native language of a newer macOS.

[1] https://objectivebythesea.com/v3/index.html

[2] Available at https://github.com/richiercyrus/Venator

[3] Apple stated that these languages would not be included in future versions, but no specific deadline was given.

3.1.2 Target macOS Version

The target version of macOS also needs to be decided. In an ideal world, the tool would cover as many versions as possible, however for the initial prototype it would be preferred to only intentionally target one version (typically between macOS releases, files may change directories or become non-existent so it would be realistic to only focus on a single version for this prototype). At the time of writing, macOS version 10.14 is the latest, with 10.15 set to be released in October. If the upcoming version is to be the development target, then it means the prototype would work with the newest version of macOS, however, it is not released until October and so development can not be started until then[4].

3.2 Execution

On completion of the majority of the research, development of the prototype can begin. This will be developed in the development language that is decided upon while researching, and will target a single specific macOS version. Throughout development, GitHub[5] and Travis CI will be used, which allows for version control and a log to be kept of what has been completed and when.

3.3 Evaluation

To evaluate the prototype, a Virtual Machine (VM) of the target macOS version will be generated. This provides some challenges as macOS typically runs poorly inside a VM, however a live system can not be used without significant expense. A VM is crucial however, as it will eventually have a live malware sample executed in it to ensure that the prototype can gather the correct information to allow for it to be identified. The live testing introduces a risk for damage, and so that is why a VM is being used as it can be deleted or reset back to a time before the malware, and the malware will only be retrieved from a known-good source. The prototype will be tested against multiple types of malware to ensure a fair test.

A successful prototype will include the main features previously mentioned. The following criteria must also be met:

- Full automation
- Easy to use
- Success rate >50%

A success, in terms of this project, will be defined as the prototype gathering enough information to allow the user to discover the source of the incident. If the project meets the above criteria then it will be deemed a success.

3.4 Timeframe

Time management will be vital throughout the development and testing of the prototype. The aim is to include the main features, gathering data and presenting it in an easy to read format, and thoroughly test. If time allows, then additional features may be added. This will be decided upon at the completion of the initial prototype.

4 Summary

The Incident Response landscape for macOS is somewhat barren. With the market share of macOS devices increasing rapidly it becomes a more and more urgent matter. This project aims to fill this gap by developing a tool to automate the data gathering and presenting it in a readable format. It should be able to gather relevant data for a target macOS version and present it to the responder so that time can be spent focusing on analysing the data rather than gathering it. The outcome of this project will be a fully operational prototype that meets the aforementioned criteria.

References

Case, Andrew and Golden G. Richard (Aug. 2016). “Detecting objective-C malware through memory forensics”. English. In: *Digital Investigation* 18, S3–S10. DOI: `10.1016/j.diin.2016.04.017`. URL: `https://www.sciencedirect.com/science/article/pii/S1742287616300524` (visited on 30/09/2019).

Cyrus, Richie (29th July 2019). *Detecting macOS Compromise with Venator*. URL: `https://youtu.be/8oMxegxZva8` (visited on 29/09/2019).

Lee, Keungi, Sangjin Lee and Changhoon Lee (May 2013). “On-site investigation methodology for incident response in Windows environments”. English. In: *Computers and Mathematics with Applications* 65.9, pp. 1413–1420. DOI: `10.1016/j.camwa.2012.01.029`. URL: `https://www.sciencedirect.com/science/article/pii/S0898122112000399` (visited on 29/09/2019).

Malware Statistics & Trends Report (25th Apr. 2019). URL: `https://www.av-test.org/en/statistics/malware/` (visited on 28/09/2019).

Wardle, Patrick (24th Sept. 2014). “Methods of Malware Persistence on Mac OS X”. In: *Virus Bulletin 2014*. (Visited on 29/09/2019).

[4] Apple has released a beta version of 10.15, however since it is a beta it may differ from the release.

[5] https://github.com