

Angry Cow

Malware Analysis Report

SikoMode

Self-Deleting Data Exfiltration Malware

Mar 15, 2022 | Angry Cow, et al. | v1.0

# Table of Contents

[Table of Contents 2](#_Toc98245557)

[Preamble to Report 3](#_Toc98245558)

[Executive Summary 4](#_Toc98245559)

[High-Level Technical Summary 5](#_Toc98245560)

[Malware Composition 6](#_Toc98245561)

[Basic Static Analysis 7](#_Toc98245562)

[Basic Dynamic Analysis 8](#_Toc98245563)

[Advanced Analysis 9](#_Toc98245564)

[Advanced Analysis (Cont.) 10](#_Toc98245565)

[Indicators of Compromise 11](#_Toc98245566)

[Host-based Indicators 11](#_Toc98245567)

[Rules & Signatures 13](#_Toc98245568)

[Appendices 14](#_Toc98245569)

[A. Yara Rules 14](#_Toc98245570)

[B. Callback URLs 14](#_Toc98245571)

# Preamble to Report

This document is a sample report completed as a final project for a course from TCM Security (<https://academy.scm-sec.com>), titled “Practical Malware Analysis and Triage”, developed, and presented by Matt Kiely.

The course was a well prepared and presented entry level course. That being said, it was a realistic artfully contrived scenario with a very functional well controlled, ”malicious” file.

As an educational exercise, there were 13 challenge questions requiring answers in the content of this report. In a spirit of full disclosure there are two that I did not personally find the answers to but included in this report for completeness. 1. The Encryption Algorithm of RC4 – while I found evidence of encryption I did not determine what the method was on my own. 2. Mention of, or the significance of the ”Houdini” signature.

This was a thoroughly enjoyable practical exercise that left me feeling just a little bit more prepared to move forward with my education in this are.

Respectfully Submitted,

Larry Schlack

Aka. The Angry Cow

# Executive Summary

**Hash Values**

|  |  |
| --- | --- |
| Md5 | B9497FFB7E9C6F49823B95851EC874E3 |
| Sha1 | 6C8F50040545D8CD9AF4B51564DE654266E592E3 |
| Sha256 | 3ACA2A08CF296F1845D6171958EF0FFD1C8BDFC3E48BDD34A605CB1F7468213E |

|  |  |  |
| --- | --- | --- |
| File type | 64 bit executable | Written in the NIM programming language |

SikoMode *(named after the password in password.txt file)* is a self-deleting data exfiltration malware package. SikoMode can target specific location(s) on a compromised machine and is self-deleting when its task(s) are completed. The most likely method of delivery to the compromised machine is by targeted phishing.

Symptoms of attack may be noticed in reduced system performance because of continuous data exfiltration. Another indicator is the presence of a file named ‘password.txt’ located at C:\Users\Public\password.txt.

YARA signature rules are attached in Appendix A. Hashes submitted to VirusTotal have yielded the following sample results.

|  |  |
| --- | --- |
| **Virus Total** | 13 of 64 found malicious (sample ID's) |
| Kaspersky | Backdoor.Win32.PMax.auos |
| Fortinet | Malicious\_Behavior.SB |
| Alibaba | Backdoor:Win32/Meterpreter.09eb9990 |

# High-Level Technical Summary

SikoMode consists of a single executable delivered to the compromised machine via a fishing link. According to the Incident Response Team it was located at C:\Users|Public\. The initial task was connecting to the call back URL of hxxp://update.ec12-4-109-278-3-ubuntu20-04.local. The second task was to contact the exfiltration URL of hxxp://cdn.altimiter.local/ This connection was kept alive by a repeating request at 1 second intervals.

Diagram

Description automatically generated

Figure 1. Simplified execution flow

# Malware Composition

SikoMode malware consists of a single file unknown.exe with the following characteristics.

|  |  |  |
| --- | --- | --- |
| File type | 64 bit exe, Exfil data encrypted with RC4 | Written in NIM |

**Hash Values**

|  |  |  |  |
| --- | --- | --- | --- |
| Md5 | B9497FFB7E9C6F49823B95851EC874E3 | | |
| Sha1 | 6C8F50040545D8CD9AF4B51564DE654266E592E3 | | |
| Sha256 | 3ACA2A08CF296F1845D6171958EF0FFD1C8BDFC3E48BDD34A605CB1F7468213E | | |
|  |  | | |
| **Virus Total** | | 13 of 64 found malicious (sample ID's) |
| Kaspersky | | Backdoor.Win32.PMax.auos |
| Fortinet | | Malicious\_Behavior.SB |
| Alibaba | | Backdoor:Win32/Meterpreter.09eb9990 |
|  | |  |

SikoMode also writes a file named password to the file system located at: C:\users\public\password.txt containing the encryption key value of “SikoMode”.

There are three circumstances where the malware will self-delete as shown in figure1 above.

1. If the initial callback address is not contacted
2. If at any time communication is interrupted during the exfiltration process
3. Once the data exfiltration is completed

# Basic Static Analysis

{Screenshots and description about basic static artifacts and methods}

During the basic static analysis portion there was very little observed that provided indications of the function of this program. Finding some sort of string information is more typical than not with malware products. Later analysis was able to reveal strings that help identify that the malware was written in the Nim programming language, The fact that these strings were not found earlier attests to some level of potential sophistication of the malware author(s). The strings were only visible for us at runtime while working in X64dbg, and Cutter. There were a few interesting strings available from PE Studio seen in Figure 2 below.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | 12,0x0001A480 | InternetOpen | | | 15,0x0001A48E | InternetOpenUrl | | | 19,0x0001A49F | InternetCloseHandle | | | 25,0x0001A85E | QueryPerformanceFrequency | | |  |  | | | 11,0x0001A0CC | | getaddrinfo | | 0x0001A0D8 | | freeaddrinfo | | 13,0x0001A0ED | | FindFirstFile | | |  |  | | --- | --- | | 19,0x00020FC6 | GetCurrentProcessId | | 18,0x00020FDC | GetCurrentThreadId | | 19,0x000210AE | RtlAddFunctionTable | | 22,0x000210D8 | ,RtlLookupFunctionEntry | | 16,0x0002112C | TerminateProcess | | 14,0x00021188 | VirtualProtect | | 6,0x00021356 | getenv | |

Figure 2 Sample Strings from PE Studio

# Basic Dynamic Analysis

{Screenshots and description about basic dynamic artifacts and methods}

It was during the basic dynamic analysis that we were able to begin determining what we might have utilizing iNetSim and WireShark

|  |
| --- |
| **INetSim Log Information**  Our first DNS request is to time.windows.com,  Second is to update.ec12-4-109-278-3-ubuntu20-04.local  Third (below) is to cdn.altimiter.local    2022-02-17 19:12:08 DNS connection, type: A, class: IN, requested name: time.windows.com    2022-02-17 19:14:46 DNS connection, type: A, class: IN, requested name: update.ec12-4-109-278-3-ubuntu20-04.local    2022-02-17 19:14:46 HTTP connection, method: GET, URL: [hxxp://update.ec12-4-109-278-3-ubuntu20-04.local/](http://update.ec12-4-109-278-3-ubuntu20-04.local/), file name: /var/lib/inetsim/http/fakefiles/sample.html    2022-02-17 19:14:47 DNS connection, type: A, class: IN, requested name: cdn.altimiter.local    2022-02-17 19:14:47 HTTP connection, method: GET, URL: [hxxp://cdn.altimiter.local/feed?post=A8E437E8F0367592569A2870BBDD382A1DFBB01A15FC23999D7788C33502AD9256E481B402BDC6BC25167B6478F204C49A9BADD68C4AC2A617437ECCBBA9](http://cdn.altimiter.local/feed?post=A8E437E8F0367592569A2870BBDD382A1DFBB01A15FC23999D7788C33502AD9256E481B402BDC6BC25167B6478F204C49A9BADD68C4AC2A617437ECCBBA9), file name: /var/lib/inetsim/http/fakefiles/sample.html    2022-02-17 19:14:48 HTTP connection, method: GET, URL: [hxxp://cdn.altimiter.local/feed?post=B69A1CF6853645A440A0337BA0FB38291DE0B01A07FC129199658DDD4C1286BE45FEA8851D9BC6BC34220A6466D404C49A988BD6895AF291136076CCAFA9](http://cdn.altimiter.local/feed?post=B69A1CF6853645A440A0337BA0FB38291DE0B01A07FC129199658DDD4C1286BE45FEA8851D9BC6BC34220A6466D404C49A988BD6895AF291136076CCAFA9), file name: /var/lib/inetsim/http/fakefiles/sample.html    2022-02-17 19:14:49 HTTP connection, method: GET, URL: [hxxp://cdn.altimiter.local/feed?post=B69C1CF58536758272963755A8FB34291DEBB01907FC28919D7789E440128EBE45FDA88C199BC6BC08240E5C72D40CC49A9B8BC2895AC6B7666571CEBBA9](http://cdn.altimiter.local/feed?post=B69C1CF58536758272963755A8FB34291DEBB01907FC28919D7789E440128EBE45FDA88C199BC6BC08240E5C72D40CC49A9B8BC2895AC6B7666571CEBBA9), file name: /var/lib/inetsim/http/fakefiles/sample.html    **This process above repeated once per second** This appears to be the exfiltration taking place as it is continuously repeating with a different string attached each time. This was an indication of data exfiltration as they were each individually distinct. |

# Advanced Analysis

{Screenshots and description about findings during advanced analysis}

During the advanced analysis of unknown.exe we were able to determine the most likely language the malware was written in. While other tools identified this as written in C, later indication were of Nim. See figures 3 and 4,

|  |
| --- |
|  |
| Figure 3 Strings from X64dbg (partial) Referencing NIM |

|  |
| --- |
|  |
| Figure 4 Strings from Cutter (partial) referencing NIM |

# Advanced Analysis (Cont.)

{Screenshots and description about advanced artifacts and methods}

ProcMon was helpful in confirming the existence of encryption and locating the key.

|  |
| --- |
|  |
| Processes from ProcMon showing possible encryption key and target |
| bcryptprimitives.dll        0x7ffd02d40000        0x83000        C:\Windows\System32\bcryptprimitives.dll Microsoft Corporation        10.0.19041.1202 (WinBuild.160101.0800)        12/8/2012 10:40:38 PM    bcrypt.dll        0x7ffd02dd0000        0x27000        C:\Windows\System32\bcrypt.dll         Microsoft Corporation        10.0.19041.1 (WinBuild.160101.0800)        5/26/2020 5:18:52 AM |
| Additional references to encryption methodOperation: CreateFile from Thread 1068 (last CreateFile item in screenshot above), C:\Users\Public\passwrd.txt |

Figure 5 Additional References

# Indicators of Compromise

|  |  |
| --- | --- |
| Domain or URL’s | hxxp://update.ec12-4-109-278-3-ubuntu20-04.local  hxxp://cdn.altimiter.local |
| Hashs | |  |  | | --- | --- | | Md5 | B9497FFB7E9C6F49823B95851EC874E3 | | Sha1 | 6C8F50040545D8CD9AF4B51564DE654266E592E3 | | Sha256 | 3ACA2A08CF296F1845D6171958EF0FFD1C8BDFC3E48BDD34A605CB1F7468213E | |
| Files | C:\Users\Public\unknown.exe  C:\Users\Public\passwrd.txt |

## Host-based Indicators

{Description of host-based indicators}

|  |  |  |  |
| --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | |
| Figure 6 Regshot Results |
| |  |  | | --- | --- | | **Keys Deleted** | 1 | | Keys Added | 63 | | Values Added | 68 | | Values Modified | 47 | | Files/(Attrib) Modified | 11 | |
|  |

RegShot Compare Results

|  |
| --- |
| **Sample Values Added** |
| HKU\S-1-5-21-4083756768-584771330-1588837462-1001\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\SessionInfo\1\ApplicationViewManagement\W32:000000000020030C\VirtualDesktop: 10 00 00 00 30 30 44 56 22 1A 67 40 BA 3F 0A 44 AE 37 A5 45 5E 46 58 AA |
| HKU\S-1-5-21-4083756768-584771330-1588837462-1001\SOFTWARE\Classes\Local Settings\Software\Microsoft\Windows\Shell\Bags\55\Shell\SniffedFolderType: "Generic" |
| HKU\S-1-5-21-4083756768-584771330-1588837462-1001\_Classes\Local Settings\Software\Microsoft\Windows\Shell\Bags\55\ComDlg\{5C4F28B5-F869-4E84-8E60-F11DB97C5CC7}\GroupByDirection: 0x00000001 |
| HKU\S-1-5-21-4083756768-584771330-1588837462-1001\_Classes\Local Settings\Software\Microsoft\Windows\Shell\Bags\55\Shell\SniffedFolderType: "Generic" |

# Rules & Signatures

A full set of YARA rules is included in Appendix A.

# Appendices

## Yara Rules

rule SikoModeTest {

    meta:

        last\_updated = "20220315"

        author = "The Angry Cow"

        description = "A rule to find SikoMode malware"

    strings:

        // Fill out identifying strings and other criteria

        $string1 = "SikoMode" ascii

        $string2 = "nim"

        $PE\_magic\_byte = "MZ"

    condition:

        // Fill out the conditions that must be met to identify the binary

        $PE\_magic\_byte at 0 and

        ($string1 and $string2)

}

## Callback URLs

|  |  |
| --- | --- |
| Domain | Port |
| hxxps:// update.ec12-4-109-278-3-ubuntu20-04.local | 80 |
| hxxps:// cdn.altimiter.local/fees? Post=(any string) | 80 |
|  |  |