Malware Lab

Joshua Main-Smith

2020-10-11

Contents

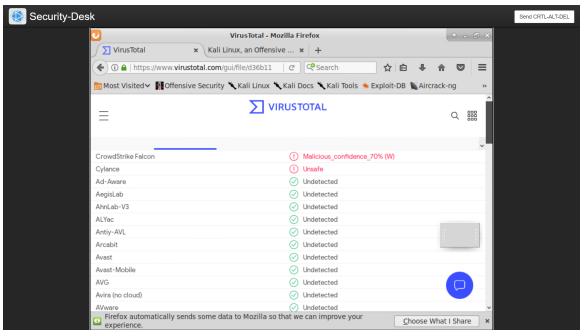
Technical Report

Malware Analysis

VirusTotal

Submitting the binary to VirusTotal indicated that two anti-virus products detect the binary as being malicious, which may indicate that it's not too common in the wild as of yet. The creation time of the binary was on 2017-08-17 while the first submission was on 2018-02-23. The magic number for the binary indicates that it's a PE32 executable for Windows. The computed hashes are as follows:

MD5	4a00c7483080667b9fd2fc71396e64e5
SHA-1	9dad7a3a39235a8399c0cd715ee95dfd550a8d79
SHA-256	d36b118bacd8620378beab682138d003e44f51feaf0fb32fcf2987c9b2efe74f
Vhash	01509e06551d155d05155az18!z
Imphash	cd58ca1bc446bd2beeb1896d153ecd7f



Hybrid-Analysis

Uploading the binary to Hybrid-Analysis determined this is a Trojan, which is consistent with our assessment in the program likely mimicking the ClamWin Anti-Virus. Some additional properties to note here is the binary makes a lot of ARP requests to to various network devices, including **192.168.240.2/32, 192.168.241.19/32, 192.168.241.134/32, 192.168.241.216/32, 192.168.242.37/32**. Something else to note is that there is a program database file (**.pdb**) string pointing to the location of where this was saved (discussed more below under **Strings**).

Strings

Using the strings command **strings clamfam.exe | less**, we discover that there are several indicators that this is indeed a malicious file.

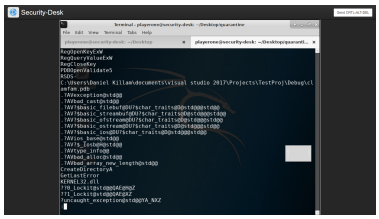


Figure 1: Strings 1

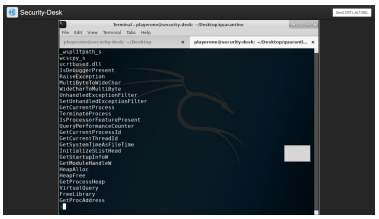


Figure 2: Strings 2

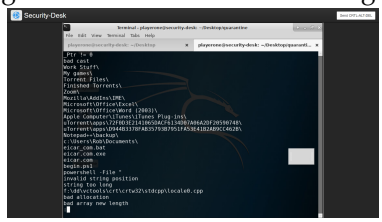


Figure 3: Strings 3

Several of these strings are consistent various types of malware. For instance, under **Strings 1** we see the string **Kernel32.dll**, indicating that the malware is importing an API used in executing lower level commands in an attempt to avoid AV detection. Further, there appears to be various AV handling. As discussed in the Hybrid-Analysis section, the string pointing to the program database file is located here. It appears the file was created using Microsoft Visual Studio 2017 by Daniel Killam. Under **Strings 2** we see various Windows API calls typical in malware binaries. Particularly, **IsDebuggerPresent** is used in thwarting a malware analysts attempt in probing binary behavior. **GetCurrentProcess, TerminateProcess, GetCurrentProcessId, GetCurrentThreadId** are all used in creating and terminating processes. This is used in tandem with performance measurements so as not to overwhelm the system in CPU usage, which is necessary in avoiding detection. **Strings 3** appears to show the various areas in which the malware was spawning child malware files.

Method of Quarantine and Removal

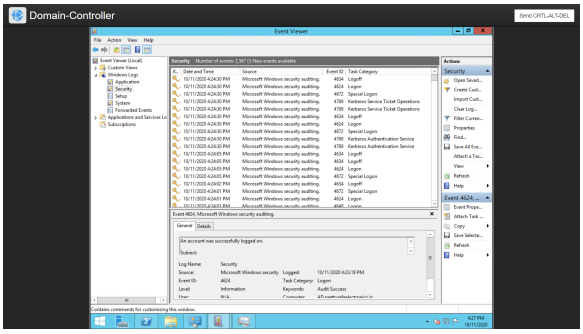
Introduction

The anomalous behavior was first noted by our intern Rob, who had communicated with the team that there has been 'weird activity' in the Domain-Controller and had suspected that there to be malware activity. To validate this claim, a malware scan was ran to see if there is indeed any malware activity. This resulted in several files being flagged as malicious. The anti-virus would continue to find flag files as being malicious after running it several more times after a period of inactivity. The only option forward was to find and discard

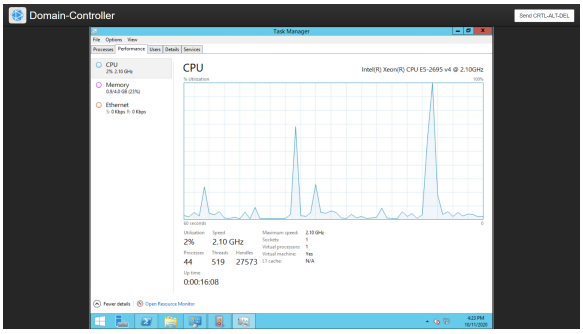
the original malware file.

Discovery

Given the above scenario, it becomes apparent that the malware is continually writing malicious files to the drive. The first step is to look at the log files of the system to see if there is any anomalous activity. This is accomplished by using the administrator tool Event Viewer (see screenshot below).

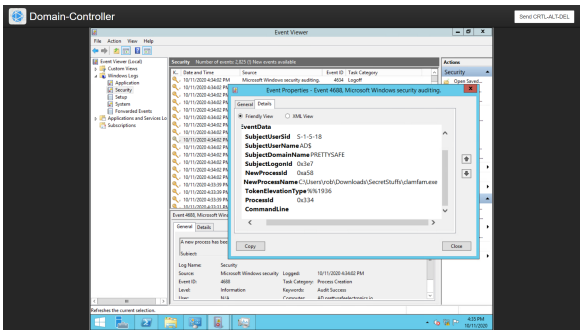


There didn't appear to be anything anomalous while looking at the log files. This may be due to the malicious file creating and terminating processes before it's able to be logged. To confirm that this is the case, we can view the Performance tab under Task Manager during a period of inactivity.



As can be seen above, there are indeed performance spikes likely coinciding with processes being created for the purpose of writing malicious files to the drive. To log these process creation/termination events, we need to change our security policy using the Local Security Policy administration tool. Under **Security Settings/Local Policies/Audit Policy/Audit Process Tracking** we can enable these attempts so that it will show under the Event Viewer tool.

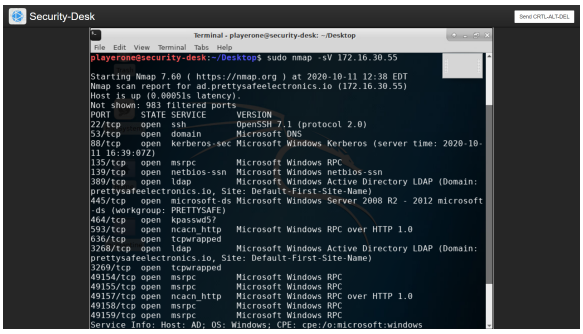
After process tracking has been enabled, we can go back to the Event Viewer and view logs categorized as **Process Creation** to see if there are any anomalies. One log that stood out came from Rob's download folder named **clamfam.exe**.



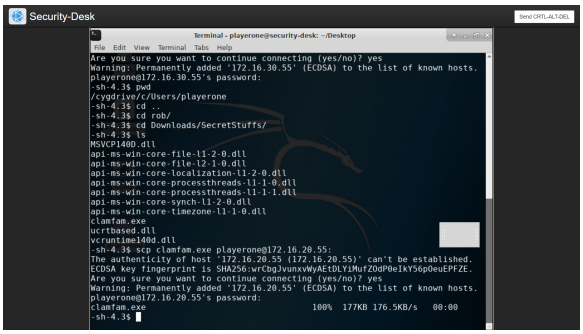
This being located in a suspicious folder named **SecretStuff** withing Rob's Download folder, this may be a Trojan binary that Rob had downloaded and ran. To investigate this further, we need to transfer this to our security workstation for analysis.

Quarantine

The first step we can do from our security workstation is scan the Domain-Controller to see if there's an open port we can connect to. We can do this by running an nmap scan by issuing **nmap -sV 172.16.30.55**.



As can be seen, the SSH port is open on the default port. So, we can connect to the Domain-Controller using the login credentials issuing the command **ssh playerone@172.16.30.55**, then entering the login password for the system. We can then navigate to the file containing the suspected malware in **c/Users/rob/Downloads/SecretStuffs/** then copy the file from our current remote directory to our local directory by issuing the command **scp clamfam.exe playerone@172.16.20.55** followed by entering the password for the system.



On our security workstation we move the file to our quarantine folder for analysis using **mv clamfam.exe Desktop/quarantine**. The analysis of the malware is written above under the Technical Report.

Deletion

After determining that this file is very likely malicious, it was permanently deleted from the Domain-Controller located under **C:/Users/rob/Downloads/SecretStuffs/clamfam.exe**. Additional scan was performed several times and the system functioned as expected.

Link to Binary

ClamFam Binary