COUNT UPON SECURITY

Increase security awareness. Promote, reinforce and learn security skills.

JUN 07 2017
2 COMMENTS

BY LUIS ROCHA

DIGITAL FORENSICS AND INCIDENT RESPONSE, INCIDENT HANDLING AND HACKER TECHNIQUES, INTRUSION ANALYSIS

THREAT HUNTING IN THE ENTERPRISE WITH APPCOMPATPROCESSOR

Last April, at the SANS Threat Hunting and IR Summit, among other things, there was a new tool and technique released by Matias Bevilacqua. Matias's presentation was titled "ShimCache and AmCache enterprise-wide hunting, evolving beyond grep" and he released the tool AppCompatProcessor. Matias also wrote a blog post "Evolving Analytics for Execution Trace Data" with more details.

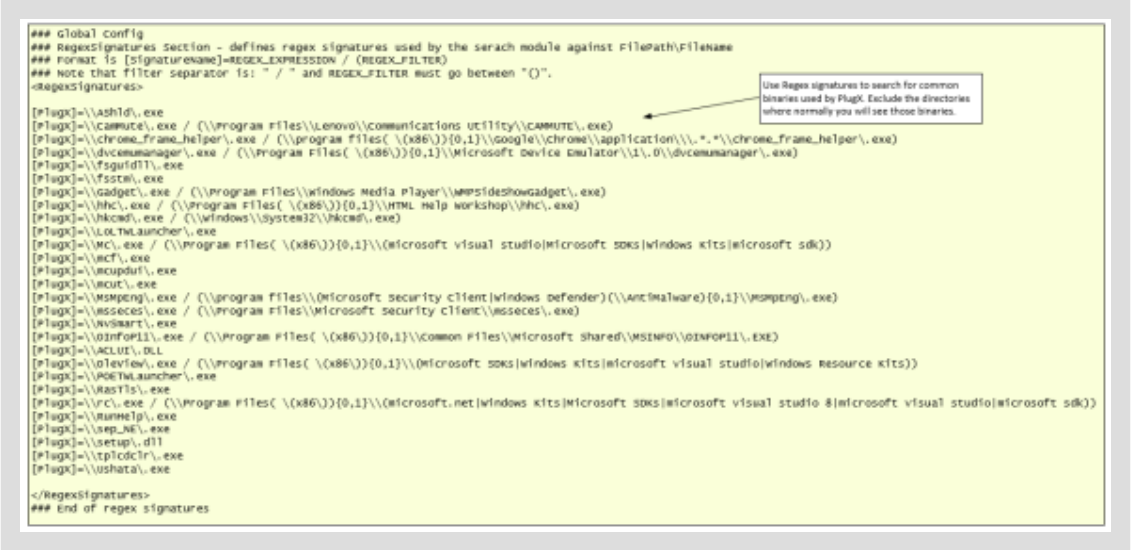
In this article, I want to go over a quick exercise on how to use this tool and expand the existing signatures. First, let me write that, in case you have a security incident and you are doing enterprise incident response or you are performing threat hunting as part of your security operations duties, this is a fantastic tool that you should become familiar with and have on your toolkit. Why? Because it allows the security teams to digest, parse and analyze, at scale, two forensic artifacts that are very useful. The forensic artifacts are part of the Windows Application Experience and Compatibility features and are known as ShimCache and the AMCache.

To give you more context, the ShimCache can be obtained from the registry and from it we can obtain information about all executable binaries that have been executed in the system since it was rebooted. Furthermore, it tracks its size and the last modified date. In addition, the ShimCache tracks executables that have not been executed but were browsed for example through explorer.exe. This makes a valuable source of evidence for example to track executables that were on the system but weren't executed – consider an attacker that used a directory on a system to move around his toolkit. The AMCache is stored on a file and from it we can retrieve information for every executable that run on the system such as the PATH, last modification time and created, SHA1 and PE properties. You can read more about those 2 artifacts in the article I wrote last year.

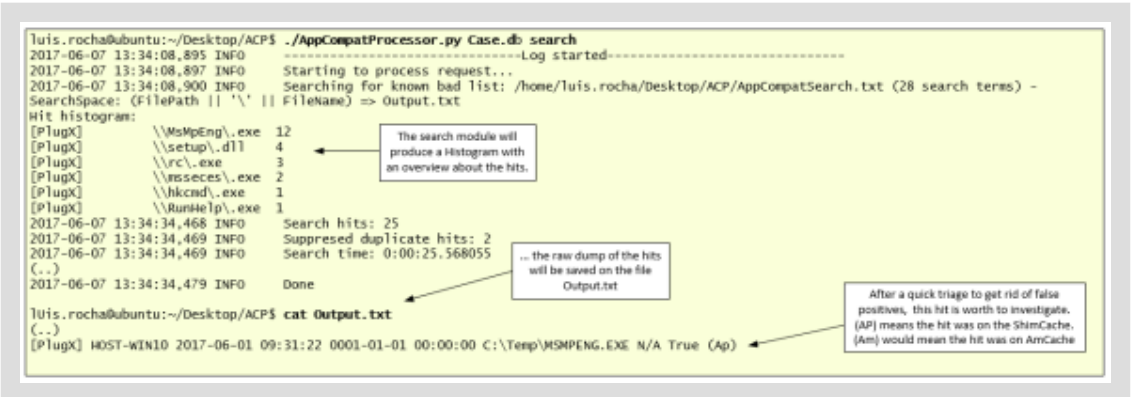
So, I won't go over on how to acquire this data at scale – feel free to share you technique in the comments – but, AppCompatProcessor digests data that has been acquired by ShimCacheParser.py, Redline and MIR but also consumes raw ShimCache and AMCache registry hives. I will go directly to the features.At the time of this writing the tool version is 0.8 and one of the features I would like to focus today is the search module. This module allows us to search for known bad using regex expressions. The search module was coded with performance in mind, which means the regex searches are quite fast. By default, the tool includes more than 70 regex signatures for all kinds of interesting things an analyst will look for when perform threat hunting. Signatures include searching for dual usage tools like psexec , looking for binaries in places where they shouldn't normally be, commonly named credential dumpers, etc. The great thing is that you can easily include your own signatures. Just add a regex line with your signature!

For this exercise, I want to use the search module to search for binaries that are commonly used by the PlugX backdoor family and friends. This backdoor is commonly used by different threat groups on targeted attacks. PlugX is also refered as KORPLUG, SOGU, DestroyRAT and is a modular backdoor that is designed to rely on the execution of signed and legitimated executables to load malicious code. PlugX, normally has three main components, a DLL, an encrypted binary file and a legitimated executable that is used to load the malware using a technique known as DLL search order. I won't go discuss the details about PlugX in this article but you can read the White Paper "PlugX – Payload Extraction" done by Kevin O'Reilly from Context, the presentation about Plugx at Black Hat ASIA in 2014 given by Takahiro Haruyama and Hiroshi Suzuki, the analysis done by the Computer Incident Response Center Luxembourg and the Ahnlab threat report. With this and other reports you could start compiling information about different PlugX payloads. However, Adam Blaszczyk from Hexacorn, already did that job and wrote an article where he outlines different PlugX payloads seen in the wild.

Ok, with this information, we start creating the PlugX regex signatures. Essentially we will be looking for the signed and legitimate executables but in places where they won't normaly be. The syntax to create a new regex signature is simple and you can add your own signatures to the existing AppCompatSearch.txt file or just create a new file called AppCompatSearch-PlugX.txt which will be consumed automatically by the tool. The figure below shows the different signatures that I produced. . At the time of this writing, this is still work in progress but is a starting point.

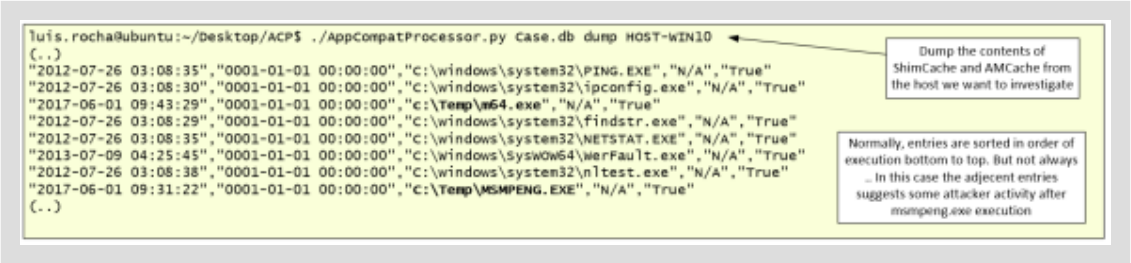


Next step, launch AppCompatProcessor against our data set using the newly created signatures. The following picture shows how the output of the search module looks like. In this particular case the search produced 25 hits and a nicely presented summary of the hits is displayed on a histogram. The raw dumps of the hits are saved on the file called Output.txt. As an analyst or investigator, you would look at the results and verify which ones would be worth to further investigate and which ones are false positives. For this exercise, there was a hit that triggered on the file "c:\Temp\MsMpEng.exe". This file is part of the Windows Defender suite but could be used by PlugX as part of DLL search order hijack technique. Basically, the attacker will craft a malicious DLL named MpSvc.dll and will place that in the same directory as the MsMpEng.exe file and execute MsMpEng.exe. The DLL would need to be crafted in a special way but that is what PlugX specializes in. This will load the attacker code.



Following these findings, we would want to look at the system that triggered the signature and view all the entries. The picture below shows this step where we use the dump module. The output shows all the ShimCache entries for this particular system. The entries are normally sorted in order of execution from bottom to top, and in this case, adjacent to the "c:\Temp\MsMpEng.exe" file there are several windows built-in commands that were executed and a file named "c:\Temp\m64.exe". This is what Matias calls a strong temporal execution correlation. This is indicative that an attacker obtained access to the system, executed several windows built-in commands and and executed a

file called “m64.exe” which likely is Mimikatz or a cousin.



Following those leads, you might want to obtain those binaries from the system and perform malware analysis in order to extract indicators of compromise such as the C&C address, look at other artifacts such Windows Event Logs, UsnJournal, memory, etc.. and have additional leads. In addition, you might want to further use AppCompatProcessor to search for the “m64.exe” file and also use the tstack module, to search across all the data set for binaries that match the date of those two binaries. With these findings, among other things, you would need to scope the incident by understanding which systems the attacker accessed, find new investigation leads and pivot on the findings. AppCompatProcessor is a tool that helps doing that. This kind of finding would definitely trigger your incident response processes and procedures.

That’s it, hopefully, AppCompatProcessor will reduce the entry barrier for your security operations center or incident response teams to start performing threat hunting in your environment and produce actionable results. If you find this useful, contribute with your threat hunting signatures in AppCompatProcessor GitHub repo and Happy Hunting!

Share this:

Twitter Facebook LinkedIn Reddit Tumblr Email

Loading...

Tagged AmCache, AppCompatProcessor, Emterprise Incident Response, PlugX, ShimCache, SOGU, Threat Hunting

2 thoughts on “Threat Hunting in the Enterprise with AppCompatProcessor”

Eric Zimmerman says: June 9, 2017 at 5:52 pm

https://github.com/mandiant/ShimCacheParser is and has been broken for a while now, at least on Creators update.

https://binaryforay.blogspot.com/2017/03/windows-10-creators-update-vs-shimcache.html

This tool works with all OS versions

https://github.com/EricZimmerman/AppCompatCacheParser

★ Like

Reply

Luis Rocha says: August 13, 2017 at 11:36 am

Hi Eric, Adam Witt fixed the Shimcache Parser for the Windows 10 Creators update version.

★ Like

Reply

Leave a comment

← Previous post Next post →

FOLLOW ME ON TWITTER

My Tweets

Search

RECENT POSTS

- FireEye Endpoint Security (HX) – Supplementary Tools
- Notes on Linux Memory Analysis – LiME, Volatility and LKM's
- Six Years ago …
- Digital Forensics – PlugX and Artifacts left behind
- Malware Analysis – PlugX – Part 2

ARCHIVES

- May 2021 (1)
- October 2019 (1)
- November 2018 (1)
- June 2018 (1)
- May 2018 (1)
- April 2018 (1)
- March 2018 (1)
- February 2018 (1)
- November 2017 (1)

CATEGORIES

- Anniversary
- Blockchain
- Common Criteria
- Digital Forensics and Incident Response
- Exploitation
- Fuzzing
- Gamification

- December 2012 (5)
- November 2012 (12)

TAGS

botnet CryptoWall CVE CVE-2013-2551 Digital Forensics Gaining Access honeypot Incident Handling and Hacker Techniques Incident Response intrusion analysis log2timeline malware Malware Analysis network security OpenIOC penetration testing PlugX REMnux SANS security hands-on-training Security Intelligence ShimCache SOGU SQL Injection Threat Intelligence

AUTHORS

- Luis Rocha
- Ricardo Dias
- Angel Alonso-Parrizas

Blog at WordPress.com.