Cryptojacker Detection and Evasion Strategies

Apollo Lo and Panat Taranat

Introduction

One of the ways to obtain cryptocurrency is through mining. Mining usually requires solving computationally intensive mathematical problems[1][2]. This mining process involves high CPU usage, resulting in high electricity usage and high bills. As the prices of cryptocurrencies surged, attackers started using hidden scripts to mine cryptocurrencies without high expenses. Cryptojacking is the malicious act of using an unwilling victim's computational power to mine cryptocurrency. The unauthorized mining on the victim's device results in increased electricity consumption and increased CPU usage for the victim, degrading the performance and usage of the victim's computer.

Cryptojacking, or web-based crypto mining works only when a user visits the webpage. Once the user visits a web page, HTML and Javascript associated with the webpage are being executed on the user's computer. Within these javascript exists a mining script that is used for crypto mining. Usually the script connects to an external server that sends mining payload in the form of webassembly. These mining payloads will allocate CPUs depending on the computer's availability and communicate with a mining pool for mining tasks. Mining process will begin when the user's computer receives mining tasks. These mining payloads do not store any data on the user's computer, instead it sends back all the computational results back to the mining pool. Most of the web-based crypto mining focus on mining Monero, which provides anonymity to Monero owners. Monero also uses Cryptonight algorithms to perform hashing, which provides advantages to CPU miners.[3]

The rise of mining scripts as a service, such as Coinhive[4] and Cryptoloot[5] has made it easier for attackers to deploy mining scripts on potential victims. These companies provide scripts for website owners that can turn visitors into miners for the Monero cryptocurrency. Monero is an attractive cryptocurrency for cryptojackers due to its anonymity and ease of mining on consumer CPUs. The scripts were intended to be an alternative to ads for site owners to generate visitor-based revenue. However, attackers exploited the scripts to mine cryptocurrencies on unwilling victims, e.g. by injecting victims' websites with the scripts or hiding scripts in their own websites. Studies have shown that during the 2017-2018 cryptojacking boom, 81% of cryptojacking websites use scripts provided by Coinhive[6].

When Coinhive shut down in March 2019, the owners claimed economic reasons from a Monero hard fork affected their profitability[7]. This decision affected the cryptojacking industry.

---

[1] G. Hong, Z. Yang, S. Yang, L. Zhang, Y. Nan, Z. Zhang, M. Yang, Y. Zhang, Z. Qian, and H. Duan, "How You Get Shot in the Back: A Systematical Study about Cryptojacking in the Real World," in ACM SIGSAC Conference on Computer and Communications Security.

[2] S. Eskandari, A. Leoutsarakos, T. Mursch, and J. Clark, "A First Look at Browser-Based Cryptojacking," in Proceedings - 3rd IEEE European Symposium on Security and Privacy Workshops, EURO S and PW 2018

[3] https://en.bitcoin.it/wiki/CryptoNight

[4] http://web.archive.org/web/20190130232758/https://coinhive.com

[5] https://web.archive.org/web/20180515073236/https://crypto-loot.com/

[6] M. Saad, A. Khormali, and A. Mohaisen, "End-to-End Analysis of In-Browser Cryptojacking," arXiv preprint arXiv:1809.02152., sep 2018. [Online]. Available: http://arxiv.org/abs/1809.02152

[7] http://web.archive.org/web/20190405011157/https://coinhive.com/blog

However, alternative web-based mining services emerged to fill in the niche of Coinhive[8][9]. Variloglu et al. showed that although 99% of sites detected by CMTracker, 1% of websites still run mining scripts[10].

In this project, we explored how the cryptojacking world has changed since Coinhive shut down. We examined the robustness and limitations of previous cryptojacking detection mechanisms. We also explored, implemented, and evaluated potential evasion techniques that could be used by attackers to evade certain detection mechanisms.

Methodology

Cryptojacking Website

We used a Go HTTP server to serve HTML pages with embedded Javascript cryptojacking code. Templates of HTML and Javscript code were taken and modified from the Minero Hidden Miner and Javascript API[11]. When a visitor visits the page, query strings in the URL are used to configure the parameters for the miner, such as threads, throttling, and site key. Threads indicate the number of CPU threads used for mining, throttling is the fraction of threads that should be active, and site key is the Minero API key so that hashes can be associated with the API user

To evade detectors that relied on patterns based on samples or blocklists, we obfuscated our code. We used an online Javascript Obfuscator Tool[12], typically used to make code harder to understand or reuse. The Crypto-Loot miner used this strategy, but had the effect of making it more difficult for us to modify the code. To further obfuscate strings that refer to Minero, we used `atob()` and `btoa()` base-64 conversion functions, to convert URLs, variables, and functions into a base-64 string that would evade pattern-matching.

Even with obfuscation, some detectors rely on the name of the websocket endpoint to detect cryptojacking activity. To hide the Minero websocket endpoint, we used a websocket reverse proxy, and changed the address of the endpoint in the mining script.

Detection Methods

The first detection method we implemented was MineSweeper which was introduced at the ACM Conference on Computer and Communications Security in 2018. It is a web crawler that can conduct static analysis on Alexa's Top 1 Million websites to find any crypto mining services or activities. In their paper, they mentioned that most crypto mining on browsers uses Webassembly, which are transferred from the external server to the user's computer as a mining payload. Webassembly is used instead of javascript since it performs optimization once at compile time, which results in faster execution time.

The first step of Minesweeper was to save the HTML page and any embedded Javascript. Then the analyzer will perform string matching to find any keyword associated with known crypto mining services like Coinhive.

---

[8] https://minero.cc/

[9] https://webminepool.com/

[10] S. Varlioglu, B. Gonen, M. Ozer and M. Bastug, "Is Cryptojacking Dead After Coinhive Shutdown?," *2020 3rd International Conference on Information and Computer Technologies (ICICT)*, 2020, pp. 385-389, doi: 10.1109/ICICT50521.2020.00068.

[11] https://minero.cc/documentation

[12] https://obfuscator.io/

Table 2: Types of mining services in our initial dataset and their keywords.

Mining Service	Keywords
Coinhive	new CoinHive\.Anonymous | coinhive.com/lib/coinhive.min.js | authedmine.com/lib/
CryptoNoter	minercry.pt/processor.js | \.User\(addr
NFWebMiner	new NFMiner | nfwebminer.com/lib/
JSECoin	load.jsecoin.com/load
Webmine	webmine.cz/miner
CryptoLoot	CRLT\.anonymous | webmine.pro/lib/crlt.js
CoinImp	www.coinimp.com/scripts | new CoinImp.Anonymous | new Client.Anonymous | freecontent.stream | freecontent.data | freecontent.date
DeepMiner	new deepMiner.Anonymous | deepMiner.js
Monerise	apin.monerise.com | monerise_builder
Coinhave	minescripts\.info'
Cpufun	snipli.com/[A-Za-z]+\" data-id='
Minr	abc\.pema\.cl | metrika\.ron\.si | cdn\.rove\.cl | host\.dns\.ga | static\.hk\.rs | hallaert\.online | st\.kjli\.fi | minr\.pw | cnt\.statistic\.date | cdn\.static-cnt\.bid | ad\.g-content\.bid | cdn\.jquery-uim\.download'
Mineralt	ecart\.html\?bdata= | /amo\.js\"> | mepirtedic\.com'

Their next step is to download all Webassembly modules that are either created or executed on the webpage and perform another static analysis. The assumption is that webassembly modules that the one conducting hashing algorithms for mining, so their static analysis for these modules also look for keywords like `cryptonight_hash` and `CryptonightWasmWrapper`. The third step of Minesweeper is to monitor CPU usage over time. In their source code, they have a threshold of 50, which means they identify any CPU usage over 50 percent as a possibility that the webpage is conduction cryptomining. The last step is to monitor communication to the mining pool. Minesweeper will monitor Websocket frames and also all AJAX requests and responses from the webpage, then conduct string matches on keywords that are associated with Stratum mining protocol[13]. In the end, Minesweeper will present three different cases of detection, detection from profile, general crypto mining activity, and cryptonight algorithms detected.

The second detection method we looked at was CMTracker. CMTracker is similar to Minesweeper as both utilize web crawling techniques on Alex's Top websites to look for crypto mining and both papers are introduced in the same conference. However, CMTracker conducts threshold based dynamic analysis on web pages to determine if there are any crypto mining activities. It has two profilers that identify crypto mining, hash-based profiler and stack structure based profiler. Hash-based profilers detect the time the webpage spends on performing hashing during a user's visit. The compare functions executed by the webpage to other nine commonly accessible hash library interfaces, like "cryptonight_hash","sha256","crypto". These libraries can be identifiable by a set of fixed signatures from open-sourced cryptocurrency or commercial mining services. In their source code, their threshold for hash-based profiling is 10, which means that if the webpage spends 10% of the time performing hashing, then there is crypto mining activity. Stack structure based profiler looks for repeated pattern in function execution. The assumption is that crypto mining runs heavy workloads with repeated behavioral patterns, while a normal webpage would not repeat the same calling stack for more than 5% of the execution time. The threshold implemented for stack structure based profilers is 30%, far greater than what a normal webpage would have.

We evaluated a cryptojacking scanner called CM-Screener. CM-Screener was developed as part of a retrospective analysis using large-scale data gathered during the peak of cryptojacking to measure the impact on web users[14]. The researcher used scans of pages to validate the coverage and accuracy of their classifiers and generate input data. The scanner

[13] https://en.bitcoin.it/wiki/Stratum_mining_protocol

[14] Holz, R., Perino, D., Varvello, M., Amann, J., Continella, A., Evans, N., Leontiadis, I., Natoli, C. and Scheitle, Q., 2020. A Retrospective Analysis of User Exposure to (Illicit) Cryptocurrency Mining on the Web. *arXiv preprint arXiv:2004.13239*.

was developed in Go. The scanner tests pages for the presence of inline Javascript code associated with cryptominers, such as links to mining pools. This method of mining was common in early phases of cryptojacking. Since attackers diversified deployment methods in an attempt to evade detection, the scanner also retrieves inline and linked Javascript code.

We examined how CMScreener researchers derived classifiers. Mining code typically consists of two parts: a part responsible for mining using WASM, and helper code to configure the miner [13]. The researchers found that most algorithms are based on Cryptonight and its variants, as well as variants of Coinhive and CryptoLoot implementations [13]. They also used as input publically available blocklists of common URLs of mining domains, websocket endpoints, Javascript filenames, and mining-specific keywords. The results were three categories of classifiers: pattern matching, signatures, and namespaces. Pattern matching classifiers include string patterns for obfuscated and unobfuscated mining code. Signature classifiers rely on MD5, fuzzy hashing, and Yara signatures to classify variants of mining scripts. Mining code tends to use global variables to simulate namespaces, so namespaces were also used as a classifier. The researchers suggested that the limitation of their approach is that they could only detect known samples and closely related derivatives that preserved key characteristics [13]. Using this limitation, we first verified that our mining code was detected by CMScreener based on the classifications, then we modified the mining code, preserving as many key characteristics as possible while still evading detection.

The last detection mechanism we evaluated was the CMBlock browser extension[15]. CMBlock uses two approaches: a blocklist-based approach and a behavior-based scan. CMBlock uses publicly available mining blocklists to detect the presence of blocklisted domains in the scripts of a website. The element used for matching the blocklist was the URL pattern with the name of the mining Javascript file. Behavior-based scanning looks for deviations from a known standard, and evaluates whether this anomaly can be ignored or not. CMBlock intercepts cryptomining scripts embedded inside HTML. It also intercepts cryptomining scripts running on a proxy network. The researchers claimed that signature detection using this mining behavior is more efficient than blocklists since attackers can make their scripts undetectable through blocklist filtering. We evaluated this claim by obfuscating strings to evade blocklists, and hiding the websocket endpoint behind a proxy server.

Results

	Method	Monitor Websocket	Blocklists	CPU Monitor	Detection from plain script	Evasion success
Minesweeper	Static/ Dynamic	Yes	Yes	Yes	Yes	N/A
CMTracker	Dynamic	No	No	No	No	N/A
CM-Screener	Dynamic	No	Yes	No	Yes	Yes
CMBlock	Dynamic	Yes	Yes	N/A	Yes	No

[15] Razali, Muhammad & Mohd Shariff, Shafiza. (2019). CMBlock: In-Browser Detection and Prevention Cryptojacking Tool Using Blacklist and Behavior-Based Detection Method. 10.1007/978-3-030-34032-2_36.

The Minesweeper tool was able to detect our mining script based on the Minero keyword. It was also able to identify that the webpage is establishing communication with the minero mining pool with Websocket and track CPU usage overtime. Comparing benign websites and our mining webpage, there is a significant difference in CPU usage. However, the CPU usage did not pass the threshold, thus it was not marked as crypto mining activity. CMTracker failed to detect our mining script from the beginning, thus we did not perform any evasion techniques to test the limitation of this detection method.

We were able to evade the detection mechanisms in CM-screener. Since CM-Screener relied on matching string patterns from the classifier, our obfuscation strategy was successful. We obfuscated strings in the Minero configuration code that links to common URLs associated with Minero. By obfuscating strings one by one, we were able to use the output of CM-Screener to understand what part was detected, e.g. the mining library, the miner URL, websocket endpoint. We also verified that using the websocket proxy has the same effect as obfuscating the websocket URL since it no longer has the Minero keyword in the URL.

We were unable to evade the detection mechanisms in CMBlock. Using what we learned from evading CM-screener, we used the same methods of obfuscation and proxying in our code. However, CMBlock was able to detect Minero keywords as well as Cryptonight algorithms. This led us to suspect that since CMBlock was a browser extension, it had access to information from the Chrome browser. Using Chrome developer tools, we saw that connections to the websocket endpoint were blocked. It is possible that despite obfuscating and proxying the websocket endpoint, Chrome exposes the connection information to browser extensions. Since the detection did not rely on keywords, the results imply that our mining script was detected by the method that the CMBlock developers described as behavior-based scanning.

Discussion

We faced many challenges while implementing the first two detection methods, Minesweeper and CMTracker. Code modifications were necessary for the program to run. Even after the program executes, the both two detection methods failed to detect cryptomining. Minesweeper faced the problem that it could not retrieve WASM modules from the webpage during execution. It uses the -dump-asm flag on the chromium browser and stores the wasm modules in a designated folder. Two of the criteria, general crypto activity and cryptonight algorithm detection, are done from static analysis of wasm modules. CMTracker could execute after modification of the database. Originally, the code required fetching urls from redis database and storing the results in MySQL. We modified the code to execute only on the website we want. However, despite retrieving results, the hash-based profiling and stack structure profiling were unsuccessful when analysing our mining script. The thresholds were not surpassed even when cryptoming activating is occurring. It detects a repeated execution of a certain function, however the function is not a hashing function, thus it was counted by the profilers.

CPU throttling is effective against detection that monitors CPU usage, so it is recommended that future crypto jacking detectors should not focus only on using threshold, as it can easily evade detection. Future research can be done by creating our own mining script or mining pool. Since our project utilizes mining scripts that many of these detection already knew existed, it became harder to evade when the detector is using keyword searches. By having our own mining script, this will be able to evade detection easily.