10,000 Files Later: How Hackers Use Your Webserver

1 Introduction

Content Management Systems (CMS) are frameworks for users to create and manage websites. Due to accessibility and ease-of-use, CMS frameworks have become nearly ubiquitous. In fact, CMS-based websites make up 55% of the internet, or more than 45 million websites [1]. Sadly, this widespread adoption has resulted in CMS-websites becoming a high-profile target for hackers.

Despite the rise in CMS-website attacks, there has been little research aiming to understand the motives of adversaries after the website has been compromised. Traditionally, “honeypots” (purposefully vulnerable web-applications) are used to collect malware [2]–[4]. Although honeypots offer convenience and control, they are infected by unsophisticated hackers, mostly automated bots — making the data unrepresentative of sophisticated attacks. In addition, recent literature [3], [4] has relied on audit logs for behavioral analysis. Unfortunately, these techniques are hardly deployed in practice. Specifically, fine-grained logging solutions still incur notable performance/space overhead and require instrumenting and training with the target systems [4], [5].

In collaboration with CodeGuard [6], the largest commercial CMS backup service, the proposed research will analyze 300,000 production websites. This research will build upon the work of Pai Kasturi et al. [5], who discovered that of those 300,000 websites, over 20,000 were infected. Pai Kasturi et al. found that they were unable to determine the attackers’ underlying behavior because production servers do not contain sufficient audit logs for log-based behavioral analysis. Instead, I found that hackers upload tens of thousands of malicious files onto compromised servers. The discovered malware demonstrated advanced persistent threat attributes, defeating the traditional backup and restore defense. Therefore, this research proposes to leverage scalable program analysis techniques to extract the attackers’ motivation and behavior behind these attacks.

2 Intellectual Merit

Objective This research will systematically analyze the CMS production malware. The following steps will achieve this objective:

1. Enumerate attacker behavior using novel static analysis techniques
2. Use file similarity techniques to determine differences between honeypot collected malware and malware collected from production servers.
3. Utilize attacker behavioral attributes to predict and prevent future attacks.

Scalable Web Attack Behavioral Analysis Dynamic analysis, running the malicious code, is not scalable because it is impossible to simulate an application with user interactions for all websites. I propose to automatically enumerate malware behavior using a novel static analysis technique which I am currently developing. My approach is to trace pathways in malicious code without running the program to determine malware behavior. The goal is to efficiently analyze and classify malware automatically in order to handle the large volume present in the dataset, and eventually all CMS websites.

PHP Malware Deobfuscation State-of-the-art commercial and open-source tools failed to deobfuscate the collected malware samples [2], [3], [7], [8]. To address the challenge, I developed a novel deobfuscation technique. The technique instruments the

PHP interpreter to create custom function hooks inserted into the malware's decryption routine — automatically extracting the hidden code.

Malware Sample Similarity This project will analyze differences between honeypot collected malware and production malware. Honeypot-collected malware identified from published datasets [3] is compared with malware collected from production servers. Manual reverse engineering of samples has already revealed that the production websites contain malware from both populations. This project will develop an automated technique to compare malware populations by (1) removing superfluous information from the malware, and (2) classifying the malware. The novel technique will determine which malware can be collected using honeypots.

Classification of collected malware will be determined by leveraging file hash signatures. Fuzzy-hash techniques will be used to accommodate related malware that are marginally different. It is common for CMS adversaries to embed malicious code into preexisting files — defeating file hash detection methods. The proposed technique will also detect embedded malware using semantic code similarity techniques (i.e. Word2Vec [9]).

Evaluation Plan The proposed behavioral analysis technique will be manually verified by reverse engineering an extensive amount of samples. The novel file deobfuscation tool results will be compared with manual deobfuscation and state-of-the-art deobfuscation tool results [7], [8]. File similarity will be verified with manual review of a sample of similarly identified files. After manual review, the malware classifications will undergo statistical analysis for further validation.

3 Broader Impact

The broader impact of the project is three-fold. (1) The proposed project will enhance national security by implementing a novel defense method that can detect, predict, and stop attacks. Understanding fundamental attack behaviors and techniques are crucial for developing this defense. (2) The results will be disseminated and expanded throughout the research community encouraging further development of safer CMS platforms. (3) Increased security for CMS websites will encourage people of diverse and non-STEM backgrounds to create websites without worrying about hackers taking over their server.

References

[1] W3Techs, *Usage of content management systems for websites*, https://w3techs.com/technologies/overview/content_management/all, Accessed: 2019-10-6.

[2] D. Canali and D. Balzarotti, "Behind the scenes of online attacks: An analysis of exploitation behaviors on the web", *NDSS*, 2013.

[3] O. Starov, J. Dahse, S. Ahmad, T. Holz, and N. Nikiforakis, "No honor among thieves: A large-scale analysis of malicious web shells", *in Proc. 25th WWW*, pp. 1021–1032, 2016.

[4] O. Catakoglu, M. Balduzzi, and D. Balzarotti, "Automatic extraction of indicators of compromise for web applications", *in Proc. 25th WWW*, pp. 333–343, 2016.

[5] R. P. Kasturi, Y. Sun, R. Duan, O. Alrawi, E. Asdar, V. Zhu, Y. Kwon, and B. Saltaformaggio, "Tardis: Rolling back the clock on cms-targeting cyber attacks", *in Proc. 41st IEEE S&P*, 2020.

[6] *Codeguard*, https://www.codeguard.com/, Accessed: 2019-10-8.

[7] UnPHP, *The online php decoder*, https://www.unphp.net/.

[8] *Revphp*, https://github.com/bediger4000/reverse-php-malware.

[9] T. V. Nguyen, A. Nguyen, H. Phan, T. Nguyen, and T. Nguyen, "Combining word2vec with revised vector space model for better code retrieval", *in Proc. 39th IEEE*, 2017.