Scanning For Vulnerable Apache Servers

Enterprise Network Security

Dylan Bray and David Chun

ABSTRACT

Describe the overall area of contribution, the crux of the problem, and end with highlights of results. For the initial report, end with the proposed experiments and what you aim to find out.

1. INTRODUCTION

First paragraph on the technology/society trends that lead to the problem at hand.

Second para: describe the key problem that if solved would make an impact. Why the current approaches leave a gap?

Third: describe your approach. Key insight that enables your approach, and what is novel/interesting about the insight.

Fourth, fifth: Delve deeper into the approach and experimental setup. In the final report, describe key findings.

End with outline or what comes next and why.

2. MOTIVATION

In a paper by Li, Avellino, Janies, and Collins (2016) [2], they discuss a tool designed to improve vulnerability management via detection and monitoring called the Software Asset Analyzer (SAA). This scanner addresses a few key scenarios that could lead to vulnerable systems, namely unapproved software download, failure to upgrade, lack of knowledge of installed applications, and remediation verification. To detect the potential install of vulnerable CPEs and verify it has been properly dealt with, the SAA tool takes a blacklist and checks network hosts for unauthorized CPEs by analyzing their individual configurations. By first scanning the system, the SAA tool will mark it as safe if three distinct checks are passed: all required CPEs are present, no blacklisted CPE is present, and only one version of a CPE is installed at a time. Once marked as safe, this configuration is used for future comparison analysis, thus accounting for deviations between CPE configurations on different machines. While this particular tool concentrates on categorizing the CPEs, a next step mentioned in the paper is to "incorporate a scoring system such as CVE, NVD, and CVSS" [2] to assess overall vulnerability of network machines. We will take this emphasis on analyzing for CVEs and scan for such vulnerabilities, specifically concentrating on the BlueKeep vulnerability for the Windows operating system. Once we filter out and detect the vulnerable machines, we can then collect network addresses, site type, and similar holistic data to gather statistics for BlueKeep vulnerable environments in order to try to detect commonalities.

3. OUR ARCHITECTURE

Our tool will scan the network for vulnerable IPs by fingerprinting each OS, firstly by http header and secondly via methods including SYN packet fingerprinting described by Shamsi, Nadnwani, Leonard, and Loguinov (2016) [3] as well as a TCP/IP method described by Jiao and Wu (2007) [1]. We can then proceed to filter by open ports and finally by scanning the actual host for the BlueKeep vulnerability. This scan can consist of detecting the RDP is installed and then attempting a request/response comparison to check if the vulnerability has been patched or not. Once we have detected a BlueKeep vulnerable machine, we can then proceed to extract holistic data about our targets, including OS version, open ports, geographic location, among others. This data can then be used to aggregate common environment or network features which could give insight into the applicability, danger, and relevancy of BlueKeep.

4. EXPERIMENTAL RESULTS

For the chosen BlueKeep vulnerability, we expect to see a very wide range of vulnerable machines as the CVE is new and applicable only for older OS without a patch. At a count of almost 500k Windows systems still vulnerable as reported by Shodan, a number only half of the original release, we expect to find comparable ratios of vulnerable machines in our network scans. We also expect the detected machines to be running older websites, deprecated web stacks, and similar such dated software. Additionally, these machines should be more likely to have high numbers of other possible vulnerabilities due to update neglect indicated by the lack of a BlueKeep patch.

5. RELATED WORK

Since the CVEs chosen are all related to Apache Server and more specifically versions of Apache server, it is important to determine a method of detecting which server version is actually running to correspond appropriate CVE

vulnerabilities with machines on the network. This is what Shodan does to fingerprint each Apache machine on a network. However, normal methods including parsing web banners or sending invalid requests to each server may not work as banners can be forged and requests may not be handled correctly or at all. However, by crafting special requests and parsing the response, it is possible to more robustly fingerprint servers. Methods for creating these requests include detecting request methods, detecting by URL, detecting by protocol statement, and detecting by protocol version. Using a combination of these four in conjunction with the previously discussed normal methods can yield more accurate results [1].

6. CONCLUSIONS

7. REFERENCES

[1] J. Jiao, Z. Xu, and W. Wei. Method of identify os based on tcp/ip fingerprint. 3:353–359, 02 2007.

[2] X. Li, P. Avellino, J. Janies, and M. P. Collins. Software asset analyzer: A system for detecting configuration anomalies. In *MILCOM 2016 - 2016 IEEE Military Communications Conference*, pages 998–1003, Nov 2016.

[3] Z. Shamsi, A. Nandwani, D. Leonard, and D. Loguinov. Hershel: Single-packet os fingerprinting. *ACM SIGMETRICS Performance Evaluation Review*, 42, 06 2014.