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 system designed to improve vulnerability management via detection and monitoring. One of the core issues covered in this paper is the case of user's failure to upgrade to a patched version of their software services, leading to increased system vulnerability. While they concentrate 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 vulnerability of networks. We will take this emphasis on network vulnerability and scan for vulnerabilities, specifically concentrating on Apache server CVEs. Then, we will collect network addresses, Apache version and similar holistic data to gather statistics on common vulnerability environments.

3. OUR ARCHITECTURE

Our tool will scan the network for vulnerable IPs . Once we detect the CVE, we can then proceed to fingerprint the server and extract holistic data about our targets, including possibly 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 each CVE.

4. EXPERIMENTAL RESULTS

For the chosen CVEs, we expect to see a very wide range of vulnerable machines as the range of exploitable versions is high for many. For the longer running CVEs, the machines found should most likely be a mix of older and newer OS's while the more recently discovered CVEs, we would expect to be running on more recent machines. In comparison to Shodan, we expect to detect a similar number of vulnerable machines on the network.

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] Z. Huang, C. Xia, B. Sun, and H. Xue. Analyzing and summarizing the web server detection technology based on http. *2015 6th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, Nov 2015.

[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.