VulAware: Towards Massive-Scale Vulnerability Detection in Cyberspace

Zhiqiang Wang[1], Pingchuan Ma[1](✉), Ruming Wang[2], Shichun Gao[1], Xuying Zhao[1], and Tao Yang[3]

[1] Beijing Electronic Science and Technology Institute, Beijing 100070, People's Republic of China
wangzq@besti.edu.cn, 20162308@mail.besti.edu.cn
[2] Hainan University, Haikou 570100, People's Republic of China
[3] Key Lab of Information Network Security of Ministry of Public Security, Shanghai 200000, People's Republic of China

Abstract. Due to the delay of threat warning and vulnerability fixing, the critical servers in cyberspace are under potential threat. With the help of vulnerability detection system, we can reduce risk and manage servers efficiently. To date, substantial related works have been done, combined with unenjoyable performance. To address these issues, we present VulAware, which is a distributed framework for detecting vulnerabilities. It is able to detect remote vulnerabilities automatically. Finally, empirical results show that VulAware significantly outperforms the state-of-the-art methods in both speed and robustness.

Keywords: Cyber security · Vulnerability detection
Network attack · Security vulnerability

1 Introduction

With the widely-using of Internet technology, cyber security is now of great significance to state security and social stability. Hence, the confidentiality, availability, and integrity of information system are faced with more and more threats and challenges. CNVD[1] had disclosed 10,822 vulnerabilities in 2016 at the 33.9% year-on-year growth. According to the report by CNCERT/CC[2] in 2016, about 40,000 IP address have attacked and created backdoor over 82,000 sites at the 9.3% year-on-year growth.

As cyber security situation becomes rigorous, the existing methods deal inefficiently with massive hosts in cyberspace. As a result, there is a great requirement of high-performance framework for detecting vulnerabilities. Hence, we proposed a distributed framework for detecting vulnerabilities, where load-balancing and efficient scheduling are realized to enhance vulnerability detector's performance

[1] China National Vulnerability Database.
[2] National Internet Emergency Centre.

L. Meng and Y. Zhang (Eds.): MLICOM 2018, LNICST 251, pp. 144–149, 2018.
https://doi.org/10.1007/978-3-030-00557-3_15

and efficiency. And ablation studies show our framework outperforms other baselines. Cyber security situation can be further enhanced, and thus the process of discovering vulnerability can be accelerated, which makes active defence come true.

The sections of this paper is as follows: In Sect. 2, We review and analyze some related works and their problems. In Sect. 3, we introduce the proposed distributed vulnerability detection framework. In Sect. 4, we evaluate the performance and accuracy of the developed prototype system. At last, we discuss our work in Sect. 5.

2 Related Works

As is mentioned in Ref. [7], researchers have proposed several approaches, including front-end detection [4,6], remote detection [3,8]. However, as cyberspace becomes more and more complicated, these approaches suffer from several limitations, such as poor adaptability, scalability and performance.

Reference [6] developed a generic web vulnerability scanner called "SecuBat" that analyses websites aimed at finding SQL injection and XSS vulnerabilities automatically. However, the module can only be used to identify individual vulnerability and can't deal with various web applications.

References [4,5] proposed a new way to infer web applications' internal state machine, which drives a black-box web application vulnerability scanner. The scanner, according to the awareness of state, generates fuzzing test cases toward a web page and automatically discover vulnerabilities. But their approach suffers from a severe problem: they don't fully support the web applications based on Ajax and the number of these web applications is sharply increasing.

In addition, Ref. [6] as well as Refs. [4,5] only supportted front-end vulnerability discovery. Server-end vulnerabilities are not considered in their approaches.

Reference [8] proposed a distributed module aimed at vulnerability scanning, which improves the efficiency of vulnerability discovery. Using PoC to provide an in-depth detection to remote hosts, the module extends standard vulnerability databases and registers the vulnerabilities' description, classification and test. The module overcomes the limitation of single machine deployment. Still Ref. [8] highly relies on the central server and can't resist any accident. Thus, hardly can high availability be ensured when deployed in a large scale, according to the disadvantaged architecture.

Reference [3] proposed a security scanning system based on C/S architecture. It works at server side and designs a unified interface to describe vulnerability. The system realized expansion of external vulnerabilities. However, similar to Refs. [3,8] didn't realize a distributed system.

While great progress has made in the existing works, many problems remain to be solved, including unpractical deployment modes and poor scalability. Directed at the problems mentioned above, we carried out our research which will be described in the following sections.

3 Architecture

Typical conventional tools for detecting vulnerabilities run with other security tools, such as firewalls, IDS and authentication devices. (See Fig. 1) Due to poor performance and lack of scalability, conventional tools are disadvantaged when encountered in high-loading situation.

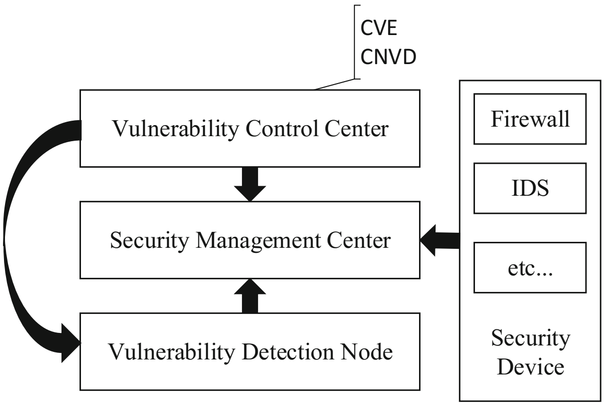
CVE
CNVD
Vulnerability Control Center
Security Management Center
Vulnerability Detection Node
Firewall
IDS
etc...
Security Device

Fig. 1. Conventional architecture

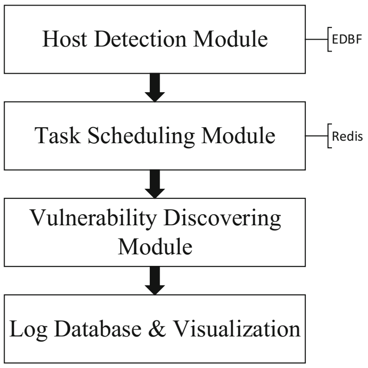
Host Detection Module
EDBF
Task Scheduling Module
Redis
Vulnerability Discovering Module
Log Database & Visualization

Fig. 2. The Architecture of VulAware

To address these challenges, we design a distributed framework called "VulAware", which decouples every components. VulAware is based on Producer-Consumer model and connected by a distributed queue. The advantage is that accident of any individual node wouldn't directly impact on run-time systems. VulAware includes host detection module, task scheduling module, vulnerability discovering module, log database and visualization module. (See Fig. 2) Similar to web crawler, the host detection module generates detection tasks. Task scheduling module is a task queue based on Redis[3], which serves as a bridge between vulnerability and host detection module. Vulnerability discovering module simulates attack behaviours with an objected-oriented method. The log database will save the test results as a log, and visualization module will show the results in a chart.

3.1 Host Detection Module

As a fundamental component of VulAware, host detection module generates the initial data in order to create further tasks. Firstly, the module renders remote pages by chrome-headless[4] and parses the elements of the pages in the Dom-tree. Secondly, it matches the URLs with the same origin. Thirdly, with distributed Bloom Filter, it removes the duplicated URLs. Finally, it adds available URLs into the task queue.

According to Refs. [1,2], Bloom Filter is an algorithm proposed by Burton H. Bloom in 1970, used to test a series of messages one-by-one for membership in a given set of messages. Compared to conventional algorithms, such as HashMap, it is of excellent time-space performance at the time complexity of $O(1)$. However, the length of binary vector must be fixed before using it. As a result, with the growth of data set, the accuracy would sharply decrease, and the existing algorithms can't handle dynamic data set. Meanwhile, the single machine deployment can't handle retrieval in a massive scale.

To address the inability of existing algorithms, we designed EDBF (Extendable Distributed Bloom Filter) which is formed by several sub-filters (the number of sub-filters is based on the error rate and initial capacity). Each sub-filter is a distributed Bloom Filter of fixed capacity, and the distributed filter contains an improved Bloom Filter and remote storage module. EDBF can add sub-filters automatically to meet the requirements of data set and error rate. Additionally, the data in each sub-filter will be saved on more than one machine and automatically synchronize among every node. Furthermore, atomic operation contributes to data persistence and load balancing. Faced with massive data, EDBL realizes both steady performance and high availability. EDBF transforms URL into a value of 256 Bits with HASH256 and thus check the vector from sub-filter to remove the duplicated URLs.

3.2 Task Scheduling Module

The task scheduling module is a bridge in the framework, which passes through the total process of VulAware. The module is a task queue based on Redis which is an open-source in-memory database project. After collected by host detection module, tasks are added to the task queue, and the vulnerability detection module will pull from the queue and detect the specific hosts.

3 https://redis.io.
4 https://developers.google.com/web/updates/2017/04/headless-chrome.

3.3 Vulnerability Discovering Module

Vulnerability discovering module is the core module of VulAware. After getting a task from the task scheduling module, it will verify remote hosts, recognize their service, retrieve services in the database, run PoC, and finally analyze the response of hosts.

The existing methods tend to simulate an attack to remote hosts by sending a specific packet and assess whether remote hosts have security vulnerabilities based on the response of hosts. For the sake of compliance with penetration test, we delete the sensitive codes and test remote hosts with the guarantee of availability.

4 Evaluation and Empirical Result

4.1 Result

The framework runs on a virtual machine platform based on five vSphere ESXi servers (32 Cores, 128 GB Memory). We deploy several virtual machines (Ubuntu 16.02), and each machine is connected with 1 GB local area network. To evaluate the performance of our framework, we implement the prototype system based on Docker. We build the vulnerable testing environments in different scales.

Struts 2 is an open-source web application framework for developing Java EE web applications and has a history of critical security bugs. Hence, we deployed a schedule app and a management platform based on Struts2 and tested these two web applications. According to the results, VulAware has found 6 vulnerabilities in schedule app within 4 s and 73 vulnerabilities in management platform within 114 s.

4.2 Evaluation

In the ideal network environment, we compare VulAware to Refs. [3,4,6,8]. In terms of distribution, Refs. [3,8] and VulAware realize distributed deployment. And in the view of robustness, VulAware is of high robustness and able to meet any accident of individual node due to Docker[5] which performs virtualization also known as containerization. In term of scalability, we simulate attack behaviours by objected-oriented methods and resolve the problem, and Ref. [8] can detect external vulnerabilities by developing plugins of their detecting module. Last but not least, when it comes to data processing ability, due to EDBF, we break through the bottleneck of massive data. Reference [3,4,8] consider little of wide-scale deployment and can't handle this situation. (See Table 1).

Overall, VulAware can detect web applications' vulnerabilities in a brief time and is advantaged in availability and reliability.

[5] https://www.docker.com.

Table 1. Evaluation.

Research	Distributed	Robustness	Scalability	Speed
This paper	Y	High	High	High
Reference [6]	N	Low	Low	Low
Reference [4]	N	Low	Low	Low
Reference [8]	Y	Low	High	Low
Reference [3]	Y	Low	Low	Low

5 Conclusion

We design and implement a distributed framework for detecting vulnerabilities and enhance the disadvantages of conventional tools, and can scan massive cyber hosts quickly. Due to the distributed deployment, it can handle any accident of individual node. Thus, high robustness and stability are achieved. Generally, VulAware mitigates the security risk in cyberspace, speeds up the process of vulnerability response, and finally the security of information system is improved from the perspective of attackers.

Despite the enormous performance, VulAware still has some limitations. It can only detect known vulnerabilities which are in the database. In the future, we would research on discovering web application vulnerabilities automatically.

Acknowledgment. Our research is supported by Key Lab of Information Network Security of Ministry of Public Security, Open Project Foundation of Information Technology Research Base of Civil Aviation Administration of China (NO. CAAC-ITRB-201705), Beijing Common Construction Project (2017), National Innovation and Start-up Training Program (201710018026).

References

1. Bloom, B.H.: Space/time trade-offs in hash coding with allowable errors. Commun. ACM **13**(7), 422–426 (1970)
2. Broder, A., Mitzenmacher, M.: Network applications of bloom filters: a survey. Internet Math. **1**(4), 485–509 (2004)
3. Chen, T.M., Cai, J.M., Jiang, R.R., Feng, X.C.: Design of network security scanning system based on plug-in. Comput. Eng. Des. (2004)
4. Doupé, A., Cavedon, L., Kruegel, C., Vigna, G.: Enemy of the state: a state-aware black-box vulnerability scanner. In: USENIX Security Symposium (2012)
5. Doupé, A., Cova, M., Vigna, G.: Why Johnny can't pentest: an analysis of black-box web vulnerability scanners. In: Kreibich, C., Jahnke, M. (eds.) DIMVA 2010. LNCS, vol. 6201, pp. 111–131. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14215-4_7
6. Kals, S., Kirda, E., Kruegel, C., Jovanovic, N.: SecuBat: a web vulnerability scanner. In: International Conference on World Wide Web, pp. 247–256 (2006)
7. Liang, L., Zhang, Y., Gao, Y., Qian, X.: Research and implementation of a vulnerability detection and initiative recover system model. Comput. Eng. **3**(3), 1–7 (2004)
8. Zhan, S.: Research and application of distributed vulnerability scanning model. Ph.D. thesis, Guangdong University of Technology (2013)