Design of web vulnerability scanner based on go language

Jingxia Chen[1,*], *Xiuling* Chen[2], and *Bo* Yu[3]

[1]Harbin Guangsha University, School of Information, 150030 Songbei New District of Harbin, China
[2]Chongqing Chemical Industry Vocational College, School of Big Data and Automation, 401228 Changshou District of Chongqing, China
[3]Harbin University of Science and Technology, School of Software and Microelectronic, 150080 Nangang District of Harbin, China

Abstract. The vulnerability scanner designed in this paper completed the collection of information and scanning of vulnerability, including six parts: input assets, asset collection, vulnerability profile, plug-in upload, single case detection and report display. The framework of vue realized the front end that included six pages, and the framework of gin realized the back end. The interface completed the separation of the front and back end. The database using MySQL designed seven tables. This scanner can avoid tedious and repetitive work, it can realize automatic scanning and testing of network vulnerabilities.

1 Introduction

Information digital society makes human life more intelligent. There are no drivers' cars, schools without textbooks, remote health management and so on. In the era of artificial intelligence, pervasive computing that can access and process information in any way anytime and anywhere can be realized. At the same time, network-based information security protection is an important issue in the current and future long-term network environment research. This paper designed a web vulnerability scanner based on the network information security. The results of information collection and vulnerability scanning can be viewed through the background management page anytime and anywhere. The function module consists of six parts: input assets, asset collection, vulnerability profile, plug-in upload, single case detection and report display.

2 Vulnerability scanning technology

2.1 classification

2.1.1 Active scanning technology

* Corresponding author: 1369993930@qq.com

Active scanning technology mainly included two parts: information collection and vulnerability scanning. Information collection collects all relevant information about the scanned assets, such as all links in the website, the real IP address of the website, the port opening of the website, etc. Vulnerability scanning is to scan all the information collected, so as to detect the risk points of the website[1]. The advantage of active scanning is that once the configuration items are configured, the scanning can be started immediately, and the penetration personnel only need to wait for the final penetration results.

2.1.2 Passive scanning technology

Passive scanning technology is to send the traffic generated by our active browsing web pages to the passive scanner for scanning. The usual way is to set up a traffic agent to forward all the traffic to the passive scanner [2].

2.2 Common web vulnerabilities

2.2.1 SQL injection vulnerability

In the process of dynamic website development, there must be database operation. Any user input data is not trusted, attackers can construct special characters to splice SQL statements, so as to execute some unauthorized commands on the database, which may lead to data destruction and deletion[3].

2.2.2 Cross site scripting vulnerability

The main attack way of cross site scripting vulnerability is to add JavaScript malicious code to HTML page, and obtain sensitive information in browser through the malicious code [4].

2.2.3 Arbitrary file download vulnerability

Arbitrary file download vulnerability exists in the download function of Web site. The realization of download function is to request local or remote resources through the network. In the case of controllable web resource paths, the files may be downloaded from a malicious site.

3 Analysis and design of the system

3.1 Design of vulnerability scanner

3.1.1 Design of information collection module

The information collection module mainly includes information collection of embedded links, IP addresses, port opening and sensitive directory opening of assets. Crawlergo is used to crawl the embedded links of assets. Crawlergo can obtain all the links in the station, and obtain more resources in the station by intelligent de duplication and intelligent filling form; the asset IP address can be resolved through the net package in go language; the asset port can be resolved by resolving the asset link into IP address, and then judging whether the asset port is open through the TCP handshake time; the asset sensitive directory can use

the The domain name and dictionary are combined, and then the head request is sent to determine whether there is sensitive directory opening through the returned status code.

3.1.2 design of vulnerability scanning module

Vulnerability scanning module is to detect the vulnerability of asset links and IP in information collection, mainly relying on the plug-ins uploaded by users.After all links and IP probes are completed, the module will store the vulnerability information in the database.

3.1.3 design of plug-in mechanism

Plug in mechanism can maximize the extension of plug-ins from the outside. Users only need to meet the rules of plug-in customization, they can upload the plug-in for vulnerability detection. To facilitate the customization of scripts, plug-ins are allowed to be written in Python and Go languages [5].

3.2 design of data tables

Data information is the basis for vulnerability scanner to run. This web scanner designs seven data tables, which are assets table, buginfo table, baseinfobase table, pocinfo table, primaryinfocol table, singleonbuginfo table and user table. The assets table contains three fields: assetsname, assetsaddress, execstatus; buginfo table contains four fields: ipdomain, urladdress, bugname, bugpoc; baseinfobase table contains four fields: assetsname, assetsaddress, starttime, Endtime; pocinfo table contains five fields: pocname, pocclass, poccpntent, pocvulnreport, pocfilepath; primaryinfocol table contains eight fields: assetsname, assetsaddress, ipdomain, port, urladdress, dir, starttime, Endtime; singleonbuginfo table includes four fields: pluginname, pluginpath, urladdress, returninfo; user table includes three fields: ID, username, password.

4 Implementation of front desk

The user interface is mainly implemented by Vue and its open source component library element UI. The front end mainly contains six files, the results are represented in Table 1 below.

Table 1. Front pages

File name	Function Description
enterAssets.vue	Page for entering assets
assetCollection.vue	Page for showing assets
overviewVulner.vue	Page for overviewing vulnerability
pluginUpload.vue	Page for uploading plug-in
singleCaseDetect.vue	Page for detection of single case
displayReport.vue	Page for displaying report

5 Implementation of vulnerability scanning

5.1 the kernel code

Vulnerability scanning will start when the information collection is completed. All resource links in the station are executed through the exectarget method. The method of ExecTarget is shown in Fig.1.

```
func ExecTarget(target, address string) {
    fmt.Println( a...: "target", address)
    var rows *sql.Rows
    rows = models.GetPocFilePath()
    PocFilePathData := make([]models.PocInfo, 0)
    fmt.Println(PocFilePathData)
    for rows.Next() {
        var pocInfo models.PocInfo
        rows.Scan(&pocInfo.PocName, &pocInfo.PocFilePath)
        //fmt.Println(pocInfo.PocFilePath)
        PocFilePathData = append(PocFilePathData, pocInfo)
    }
    fmt.Println(PocFilePathData)
    PocFilePathDataLen := len(PocFilePathData)
    for i := 0; i < PocFilePathDataLen; i++ {
        fmt.Println(PocFilePathData[i].PocFilePath)
        taskName := PocFilePathData[i].PocFilePath
        suffix := strings.Split(taskName, sep: ".")
        scriptClass := strings.Split(suffix[0], sep: "/") //Script Categories go java 。。。
        fmt.Println(scriptClass)
        fmt.Println(scriptClass[2])
        fmt.Println( a...: "PocFilePathData[i].PocName:::", PocFilePathData[i].PocName)
        scriptClassDistribute(taskName, target, PocFilePathData[i].PocName, address, suffix, scriptClass)
        //PocFilePathData[i].PocName is fine here.
    }
}
```

Fig. 1. The method of ExecTarget.

5.2 the method of judge Ip Address Or url Address

In this method, judge ipaddressorurladdress method is used to determine whether the incoming IP address is true or false. If it is an IP address, the plug-in related to host will be executed; if not, the plug-in related to web will be executed. The method of judgeIpAddressOrurlAddress is shown in Fig. 2.

```
func judgeIpAddressOrurlAddress(target string) bool {
    ipv4 := target
    // ParseIP This method can be used to check if the ip address is correct; if not, it returns nil.
    address := net.ParseIP(ipv4)
    if address == nil {
        return false
    } else {
        fmt.Println( a...: "Correct IP address", address.String())
        return true
    }
}
```

Fig. 2. the method of judge Ip Address Or url Address.

5.3 System operation interfaces

The interface of System vulnerability profile shows vulnerability detection date, end date, asset address and operation. The interface of System vulnerability profile is shown in Fig. 3.

Fig.3. The interface of System vulnerability profile.

The function of Plug-in upload can realize specifying the name of plug-in, the selection of plug-in category, the description of plug-in, the of hazards of vulnerability, uploading the file, etc.The interface is shown in Fig. 4.

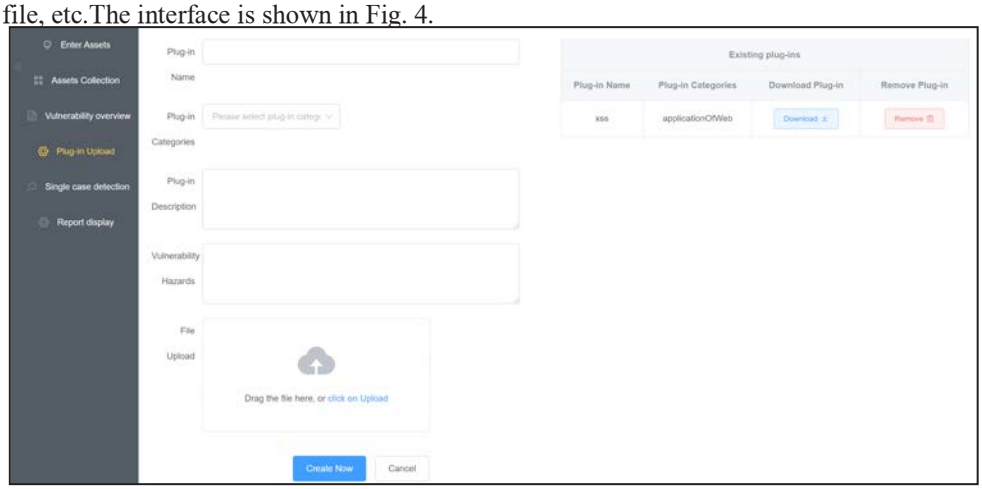

Fig. 4. The page of plug-in upload.

6 Conclusion

The feature of this web vulnerability scanner system is that the development method realizes the separation of the front and back ends. This system well demonstrates the advantages of Go language. A reasonable choice of programming language and the use of the characteristics of different languages can make the project more elegant.

This work was supported by the Natural Science Foundation of Heilongjiang Province of China (Grant No. F2017014).

References

1. M S Raunak,Richard Kuhn,R Kogut,Hot Topics in the Science of Security **10**,1-2(2020)
2. M. Alsaleh,N. Alomar,Security and Communication Networks, **2017**, 6 (2017)
3. R. Amankwah, J. Chen, P. K. Kudjo, Practice and Experience, **50**, 9 (2020)
4. Y Heribertus, T Agung, MSE **1**, 879(2020)
5. H Jin, S Boxiang, L Yan, Theory and Experiment, **1550**, 3 (2020)