MINISTRY OF EDUCATION AND TRAINING

FPT UNIVERSITY

CAPSTONE PROJECT DOCUMENT

Building A Vulnerability Management Solution For Enterprise

IAT491_G2		
Group Member	Nguyễn Anh Việt	SE62219
	Trần Anh Đức	SE04515
	Lê Đình Mạnh	SE04601
	Nguyễn Khắc Hùng	SE04644
	Nguyễn Đức Anh	SE04607
Supervisor	Hà Bách Nam	
Capstone Project code		

- Hanoi, August/2019 -

TABLE OF CONTENTS

Table of Contents

LIST OF FIGURES

ACKNOWLEDGMENT

Our capstone project would have taken far longer to complete without the encouragement from many supporters. It is a delight to acknowledge those people who have supported us over four months.

We are particularly thankful for the help and advice of our instructor, Mr. Ha Bach Nam throughout the semester. The constructive recommendations given by him has been a great help in developing our project. Without his support and guidance, our accomplishment would not have been possible.

Furthermore, we would like to thank all of the CSD (Cyber Security Department) members from FPT Information System for helping our team to create a testing environment.

We would like to thank Dr. Phan Truong Lam for the orientation time. We would also like to express my very great appreciation to Dr. Nguyen Khac Viet for his arrangement about the room for our team.

Our special thanks are extended to the staff of FPT University for their support and guideline throughout our study during the last four years. Shall you find as a reflection of the knowledge and experiences you have brought to us.

ABSTRACTION

From the beginning of the Internet, there is no organization or IT environment concern about security vulnerabilities until there is a disclosure of sensitive information. But nowadays many organizations worldwide are really considering this is an important issue that needs attention.

"Prevention is better than cure" is always true. We need to know our weakness, our threat, and our vulnerabilities in order to detection/prevention or minimize the impact.

Vulnerability management is widely described as the practice of identifying, classifying, remediating, and mitigating weaknesses in an IT environment. It is also described as the discovery, reporting, prioritization, and response to vulnerabilities in your network.

Given the countless examples of the devastating consequences that result when threat actors are able to exploit weaknesses, having a vulnerability management program is no longer optional for organizations. In fact, it is becoming required by multiple compliance, audit, and risk management frameworks.

You can't stop what you can't see. Organizations need to know what is on their network in order to monitor and protect it, which is why a vulnerability management program should be a foundation of your security infrastructure. A good vulnerability management program can help you proactively understand the risks to every asset in order to keep it safe.

Our capstone project develops a Vulnerabilities Management Solution for Enterprise – A solution that integrated many free, open-source tool for vulnerability detection, combine them to create a vulnerability management solution which can help organizations easily manage their vulnerabilities, without the need of building then implementing for their own complicated infrastructure.

I. INTRODUCTION

I.1 Project Information

- Project name: Building a vulnerability management solution for enterprises.
- Project name in Vietnamese: Xây dựng giải pháp quản lý lỗ hổng cho doanh nghiệp.
- Timeline: from April 22nd, 2019 to August 31st, 2019

I.2. People:

I.2.1. Team Name:

FPT SpaceBoy Neva Sleep

I.2.2. Supervisors:

Full Name	E-mail	Title
Hà Bách Nam	NamHB4@fe.edu.vn	Lecturer

I.2.3. Team Members:

No	Full Name	Student ID	E-mail	Role
1	Nguyễn Anh Việt	SE062219	VietNASE62219@fpt.edu.vn	Leader
2	Trần Anh Đức	SE04515	DucTASE04515@fpt.edu.vn	Member
3	Lê Đình Mạnh	SE04601	ManhLDSE04601@fpt.edu.vn	Member
4	Nguyễn Khắc Hùng	SE04644	HungNKSE04644@fpt.edu.vn	Member
5	Nguyễn Đức Anh	SE04607	AnhNDSE04607@fpt.edu.vn	Member

I.3. Background Of Our Project:

In today’s world, there are many organizations and companies which do not have a security department to search for system’s vulnerabilities and have the ability the migrate bugs. They often have to hire vulnerability assessment services to detect threat/vulnerability in their system. And then they hire another service to migrate those threats frequently, usually twice a year. Some specific organizations like the Department of Defense or in government, they perform more often than twice a year. The cost for this service is expensive and consume a lot of time and personnel of both parties.

Sometimes, when the migrating team comes, they don’t know the history of vulnerabilities of that system, makes the migration much harder. If the system administrator knows their vulnerability in fingertips, it makes the work of fixing easier, also the administrator can do some workaround in order to avoid the impact of the vulnerability before the attacker exploit the vulnerability.

Hire outsource services make the organization dependence on these services while the organization can fully take the initiative vulnerabilities assessment if they have an automatic tool with a reasonable price and a trained team. They will know the vulnerabilities in their system and manage it easily and proactively in fixing the vulnerability.

For those reasons, our project was created to solve these problems in the most thorough way. We will create a tool that allows automating the work of vulnerability scans, give a detailed report of the level of vulnerability, the effect of the vulnerability found, track the vulnerability if it hasn't been fixed. Perform periodic and automatic vulnerability scans, give alerts to administrators if there is anything changed about their system.

I.4. The Initial Idea Of Our Group:

Doing all the vulnerabilities assessment by hand is not a great idea because it takes a lot of time and the accuracy is not really high. So don’t we take all the steps automatically?

Because we know that the pentester or the hacker will first use some information gathering and vulnerabilities assessment tools such as: **Nmap**, **Wappalyzer**, **Nikto**, **Acunetix**, **Nessus**,... to have a closer look at your IT system. But what if your IT system has thousands of computers, routers, firewalls, and mobile devices? It is a kind of no one can manage that much of devices. So we decide to build a vulnerability management solution to have not just pentester but also IT administrators to always know the current vulnerability state of their system. So they can manage their system no matter how much devices their system has.

So, how can we make it work? We will learn all the steps which pentester or hackers do when they want to assess the vulnerability of a system. Then we will make all of that step automatically. Then, all the remaining steps they need to do is a simple click.

I.5. A Brief Overview Of Current Vulnerabilities Assessment Tools On The World:

I.5.1. What Are Vulnerabilities Assessment Tools?

The vulnerability assessment process is intended to identify threats and the risks they pose typically involves the use of automated testing tools, such as network security scanners, and the result is listed in a vulnerability assessment report.

The vulnerability assessment tools do these the process of identifying, quantifying, and prioritizing (or ranking) the vulnerabilities in a system automatically. Examples of systems for which vulnerability assessments are performed include, but are not limited to, information technology systems, energy supply systems, water supply systems, transportation systems, and communication systems. Such assessments may be conducted on behalf of a range of different organizations, from small businesses up to large regional infrastructures.

Vulnerability assessment has many things in common with risk assessment. Assessments are typically performed according to the following steps: [2.7]

1. Cataloging assets and capabilities (resources) in a system.
2. Assigning quantifiable value (or at least rank order) and importance to those resources.
3. Identifying the vulnerabilities or potential threats to each resource.
4. Mitigating or eliminating the most serious vulnerabilities for the most valuable resources.

I.5.2. Overview Of Some Popular Tools.

Nowadays, there are many automated tools help with information gathering and vulnerability assessment like Nmap, NSE, Wappalyzers, Acunetix, Nikto, Nessus, … Those are popular tools, each tool has different advantages and disadvantages, combining tools together will get the most effective result. Because those tools can cover each other disadvantages.

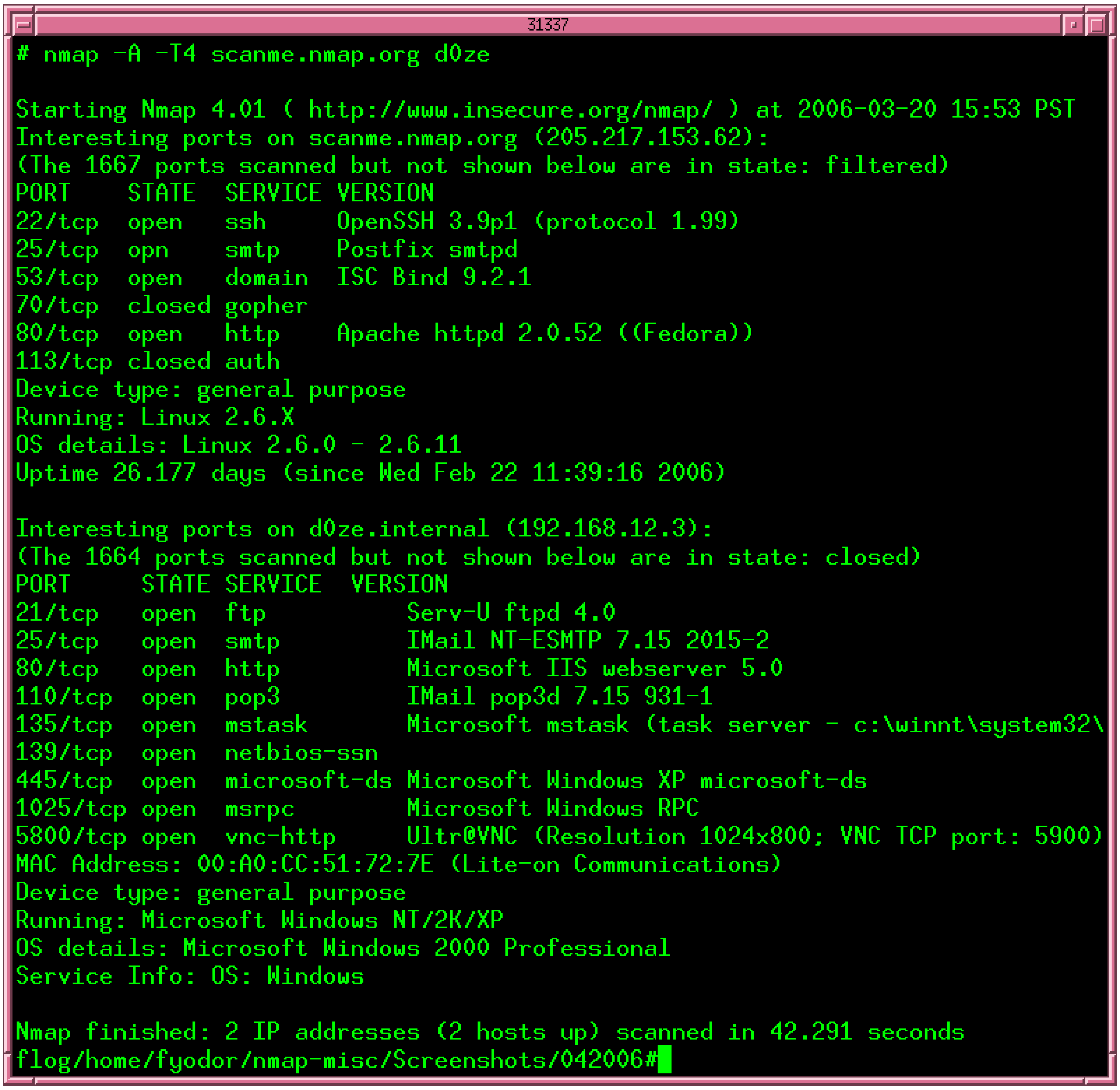
Below is the flow of work (active stream) about some popular tools. What is it used for and how it works.

I.5.2.1 Nmap & NSE [2.2]

Nmap ("Network Mapper") is a free and open-source (license) utility for network discovery and security auditing. Many systems and network administrators also find it useful for tasks such as network inventory, managing service upgrade schedules, and monitoring host or service uptime.

It is a network scanning and host detection tool that is very useful during several steps of penetration testing. Nmap is not limited to merely gathering information and enumeration, but it is also a powerful utility that can be used as a vulnerability detector or a security scanner. So Nmap is a multipurpose tool, and it can be run on many different operating systems including Windows, Linux, BSD, and Mac. Nmap is a very powerful utility that can be used to:

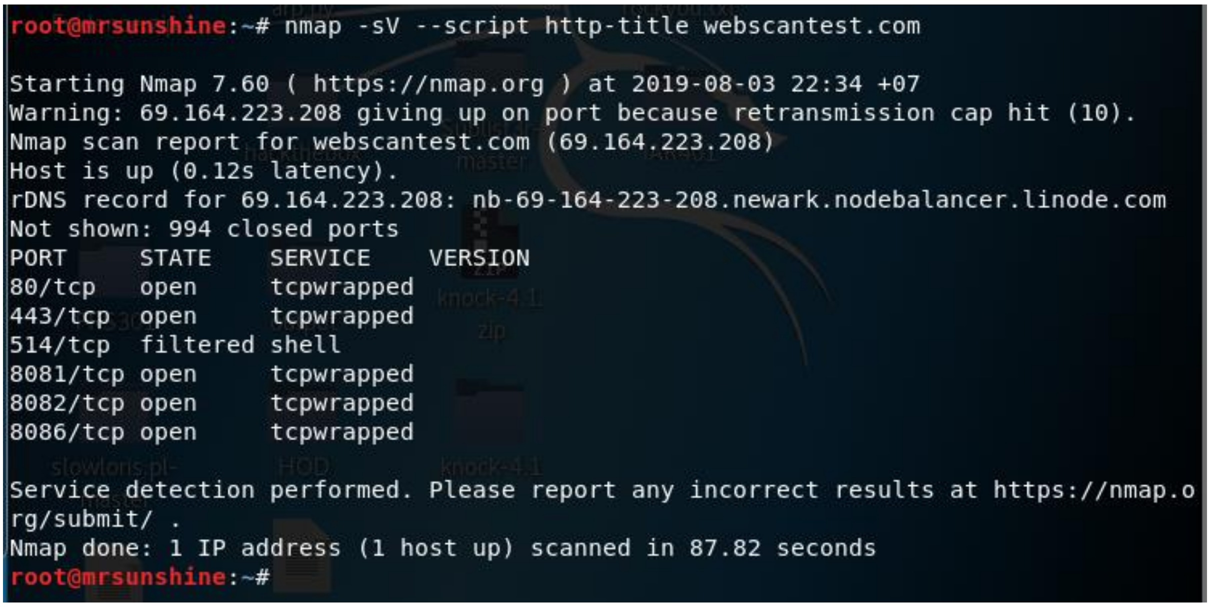
- Detect the live host on the network (host discovery)
- Detect the open ports on the host (port discovery or enumeration)
- Detect the software and the version to the respective port (service discovery)
- Detect the operating system, hardware address, and the software version
- Detect the vulnerability and security holes (Nmap scripts)



```
# nmap -A -T4 scanme.nmap.org d0ze

Starting Nmap 4.01 ( http://www.insecure.org/nmap/ ) at 2006-03-20 15:53 PST
Interesting ports on scanme.nmap.org (205.217.153.62):
(The 1667 ports scanned but not shown below are in state: filtered)
PORT    STATE  SERVICE VERSION
22/tcp  open   ssh     OpenSSH 3.9p1 (protocol 1.99)
25/tcp  opn    smtp    Postfix smtpd
53/tcp  open   domain  ISC Bind 9.2.1
70/tcp  closed gopher
80/tcp  open   http    Apache httpd 2.0.52 ((Fedora))
113/tcp closed auth
Device type: general purpose
Running: Linux 2.6.X
OS details: Linux 2.6.0 - 2.6.11
Uptime 26.177 days (since Wed Feb 22 11:39:16 2006)

Interesting ports on d0ze.internal (192.168.12.3):
(The 1664 ports scanned but not shown below are in state: closed)
PORT     STATE SERVICE      VERSION
21/tcp   open  ftp          Serv-U ftpd 4.0
25/tcp   open  smtp         IMail NT-ESMTP 7.15 2015-2
80/tcp   open  http         Microsoft IIS webserver 5.0
110/tcp  open  pop3         IMail pop3d 7.15 931-1
135/tcp  open  mstask       Microsoft mstask (task server - c:\winnt\system32\
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds Microsoft Windows XP microsoft-ds
1025/tcp open  msrpc        Microsoft Windows RPC
5800/tcp open  vnc-http     Ultr@VNC (Resolution 1024x800; VNC TCP port: 5900)
MAC Address: 00:A0:CC:51:72:7E (Lite-on Communications)
Device type: general purpose
Running: Microsoft Windows NT/2K/XP
OS details: Microsoft Windows 2000 Professional
Service Info: OS: Windows

Nmap finished: 2 IP addresses (2 hosts up) scanned in 42.291 seconds
flog/home/fyodor/nmap-misc/Screenshots/042006#
```

Figure I.5.2.1.a. Nmap demo scan result

NSE - Nmap can find vulnerabilities in the network through the Nmap Script Engine (NSE) - a flexible feature activated with the -sC option that allows users to write scripts for task automation. NSE comes with a rich collection of scripts that can help in the network discovery process, with vulnerability exploitation, and backdoor detection.



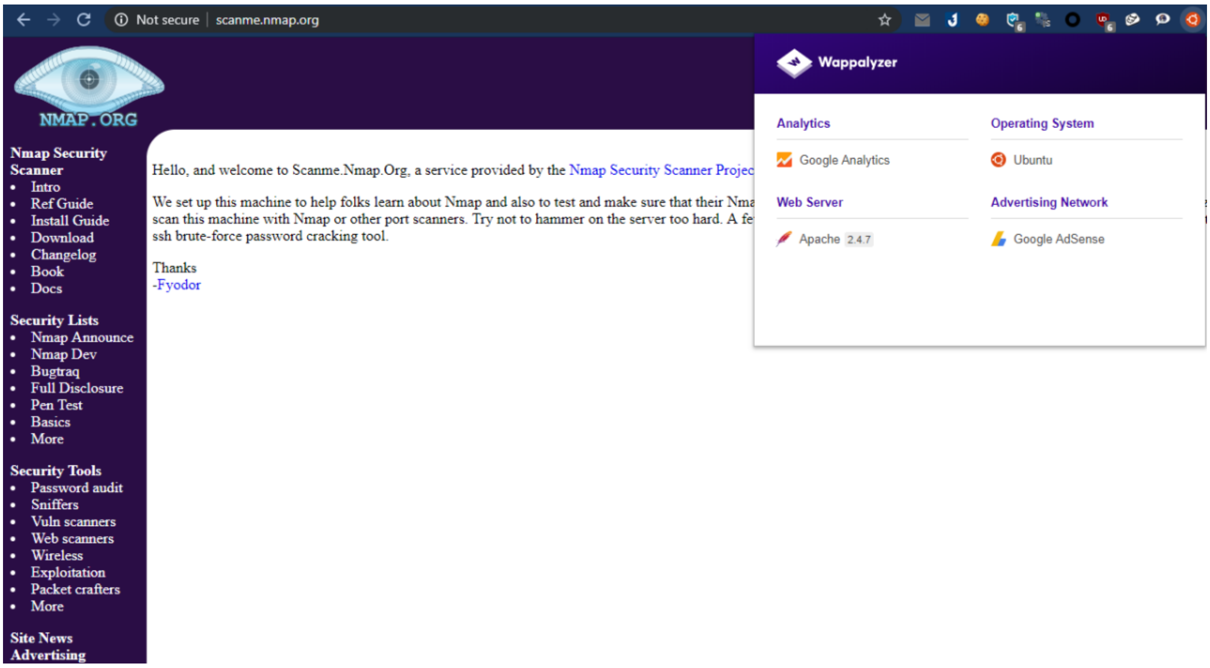
```
root@mrsunshine:~# nmap -sV --script http-title webscantest.com

Starting Nmap 7.60 ( https://nmap.org ) at 2019-08-03 22:34 +07
Warning: 69.164.223.208 giving up on port because retransmission cap hit (10).
Nmap scan report for webscantest.com (69.164.223.208)
Host is up (0.12s latency).
rDNS record for 69.164.223.208: nb-69-164-223-208.newark.nodebalancer.linode.com
Not shown: 994 closed ports
PORT     STATE    SERVICE    VERSION
80/tcp   open     tcpwrapped
443/tcp  open     tcpwrapped
514/tcp  filtered shell
8081/tcp open     tcpwrapped
8082/tcp open     tcpwrapped
8086/tcp open     tcpwrapped

Service detection performed. Please report any incorrect results at https://nmap.o
rg/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 87.82 seconds
root@mrsunshine:~#
```

Figure I.5.2.1.b. Nmap NSE script demo scan result

I.5.2.2 Wappalyzers [2.3]

Wappalyzer is an open-source, platform-independent utility capable of identifying 1,238 different web technologies. It was created by Elbert Alias in 2009, has received contributions from hundreds of developers worldwide and enjoys an active user base of a million people.

Wappalyzer fingerprints software using unique patterns found in website source code, response headers, script variables, and several other methods. Wappalyzer is written in JavaScript and can be used on any operating system natively using Node.js. It can be run as a stand-alone application or included as a module in a larger application. Wappalyzer collects data anonymously and organically through the browser extensions. This information is processed and made available through APIs and datasets, providing valuable insights into the software industry.


Not secure | scanme.nmap.org

NMAP.ORG

Nmap Security Scanner
- Intro
- Ref Guide
- Install Guide
- Download
- Changelog
- Book
- Docs

Security Lists
- Nmap Announce
- Nmap Dev
- Bugtraq
- Full Disclosure
- Pen Test
- Basics
- More

Security Tools
- Password audit
- Sniffers
- Vuln scanners
- Web scanners
- Wireless
- Exploitation
- Packet crafters
- More

Site News
Advertising

Hello, and welcome to Scanme.Nmap.Org, a service provided by the Nmap Security Scanner Projec

We set up this machine to help folks learn about Nmap and also to test and make sure that their Nma scan this machine with Nmap or other port scanners. Try not to hammer on the server too hard. A fe ssh brute-force password cracking tool.

Thanks
-Fyodor

Wappalyzer

Analytics	Operating System
Google Analytics	Ubuntu

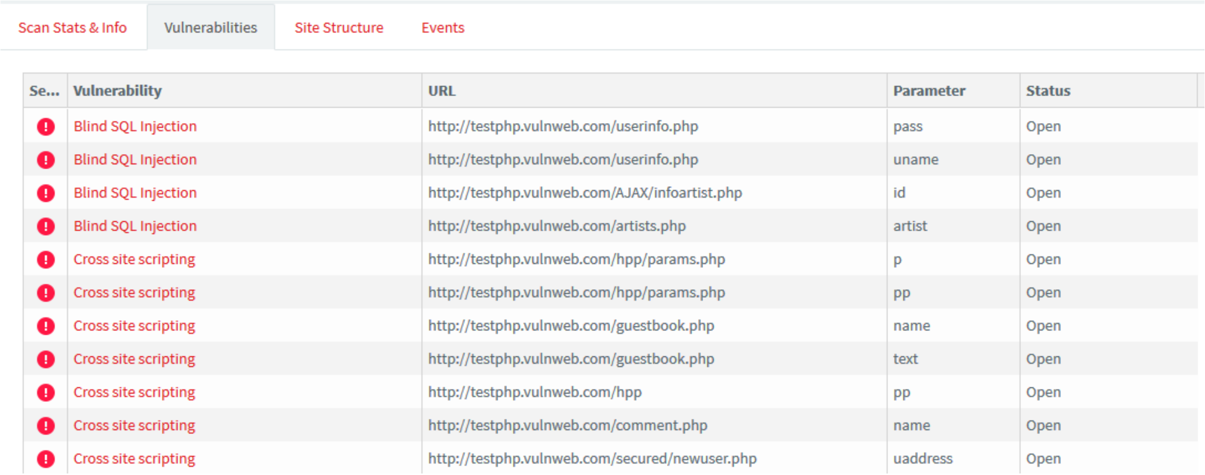
Web Server	Advertising Network
Apache 2.4.7	Google AdSense

Figure I.5.2.2.a. Wappalyzers chrome extension result

I.5.2.3 Acunetix [2.4]

Acunetix is a software product for web application security testing which helps you quickly and easily identify known vulnerabilities, as well as vulnerabilities in any website or web application, including sites built with hard-to-scan HTML5 and JavaScript Single Page Applications (SPAs). With Acunetix you can:

- Discover in excess of more than 4,500 security vulnerabilities
- Detect SQL Injection and Cross-site Scripting and all of their variants
- Automatically scan all website files with custom form authentication or other custom access controls and session management



Se...	Vulnerability	URL	Parameter	Status
!	Blind SQL Injection	http://testphp.vulnweb.com/userinfo.php	pass	Open
!	Blind SQL Injection	http://testphp.vulnweb.com/userinfo.php	uname	Open
!	Blind SQL Injection	http://testphp.vulnweb.com/AJAX/infoartist.php	id	Open
!	Blind SQL Injection	http://testphp.vulnweb.com/artists.php	artist	Open
!	Cross site scripting	http://testphp.vulnweb.com/hpp/params.php	p	Open
!	Cross site scripting	http://testphp.vulnweb.com/hpp/params.php	pp	Open
!	Cross site scripting	http://testphp.vulnweb.com/guestbook.php	name	Open
!	Cross site scripting	http://testphp.vulnweb.com/guestbook.php	text	Open
!	Cross site scripting	http://testphp.vulnweb.com/hpp	pp	Open
!	Cross site scripting	http://testphp.vulnweb.com/comment.php	name	Open
!	Cross site scripting	http://testphp.vulnweb.com/secured/newuser.php	uaddress	Open

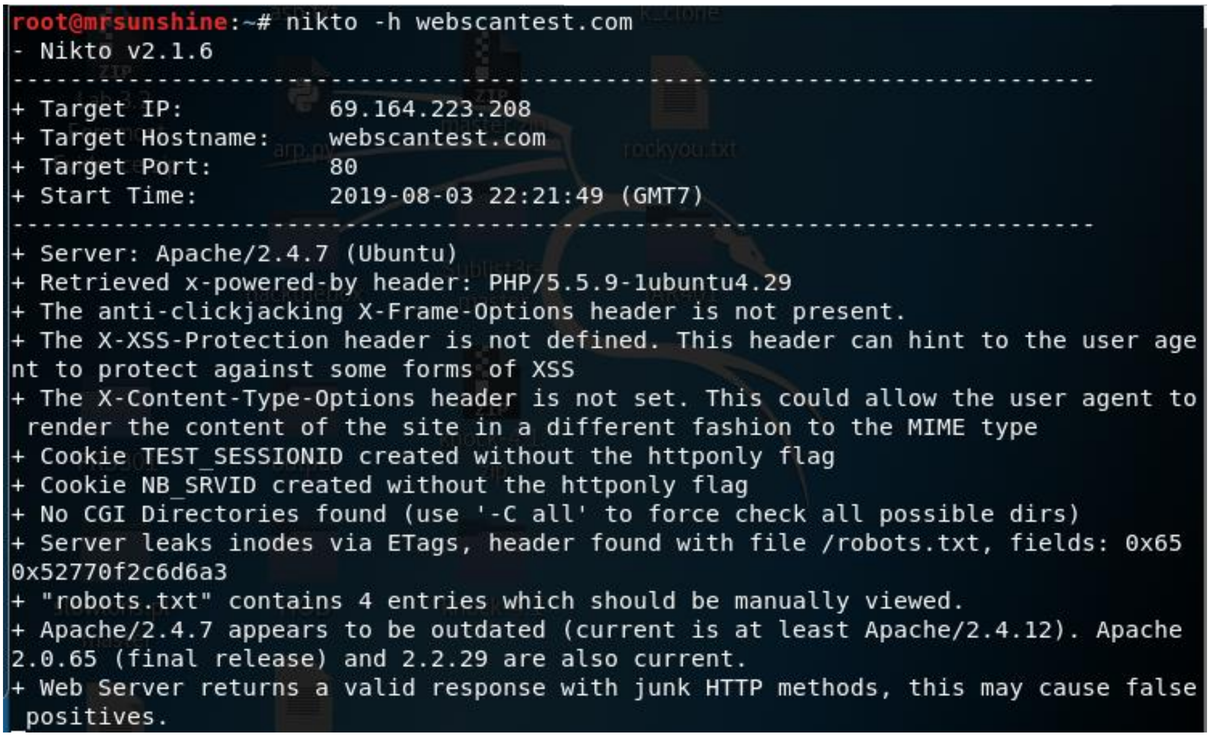
Figure I.5.2.3.a. Acunetix scan result

Additionally, Acunetix can find security issues beyond the typical black-box scanning approach thanks to its AcuSensor gray-box scanning technology. With AcuSensor, Acunetix can automatically examine Java, ASP.NET and PHP server-side code that is being executed. This allows Acunetix to pinpoint the exact line of code where vulnerabilities lie, as well as dramatically reduce an already low false-positive rate.

I.5.2.4 Nikto [2.5]

Nikto is a web server assessment tool. It is designed to find various default and insecure files, configurations, and programs on any type of web server. Examine a web server to find potential problems and security vulnerabilities, including:

- Server and software misconfigurations
- Default files and programs
- Insecure files and programs
- Outdated servers and programs



```
root@mrsunshine:~# nikto -h webscantest.com
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          69.164.223.208
+ Target Hostname:    webscantest.com
+ Target Port:        80
+ Start Time:         2019-08-03 22:21:49 (GMT7)
---------------------------------------------------------------------------
+ Server: Apache/2.4.7 (Ubuntu)
+ Retrieved x-powered-by header: PHP/5.5.9-1ubuntu4.29
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user age
nt to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to
 render the content of the site in a different fashion to the MIME type
+ Cookie TEST_SESSIONID created without the httponly flag
+ Cookie NB_SRVID created without the httponly flag
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Server leaks inodes via ETags, header found with file /robots.txt, fields: 0x65
0x52770f2c6d6a3
+ "robots.txt" contains 4 entries which should be manually viewed.
+ Apache/2.4.7 appears to be outdated (current is at least Apache/2.4.12). Apache
2.0.65 (final release) and 2.2.29 are also current.
+ Web Server returns a valid response with junk HTTP methods, this may cause false
 positives.
```

Figure I.5.2.4.a. Nikto scanning host result

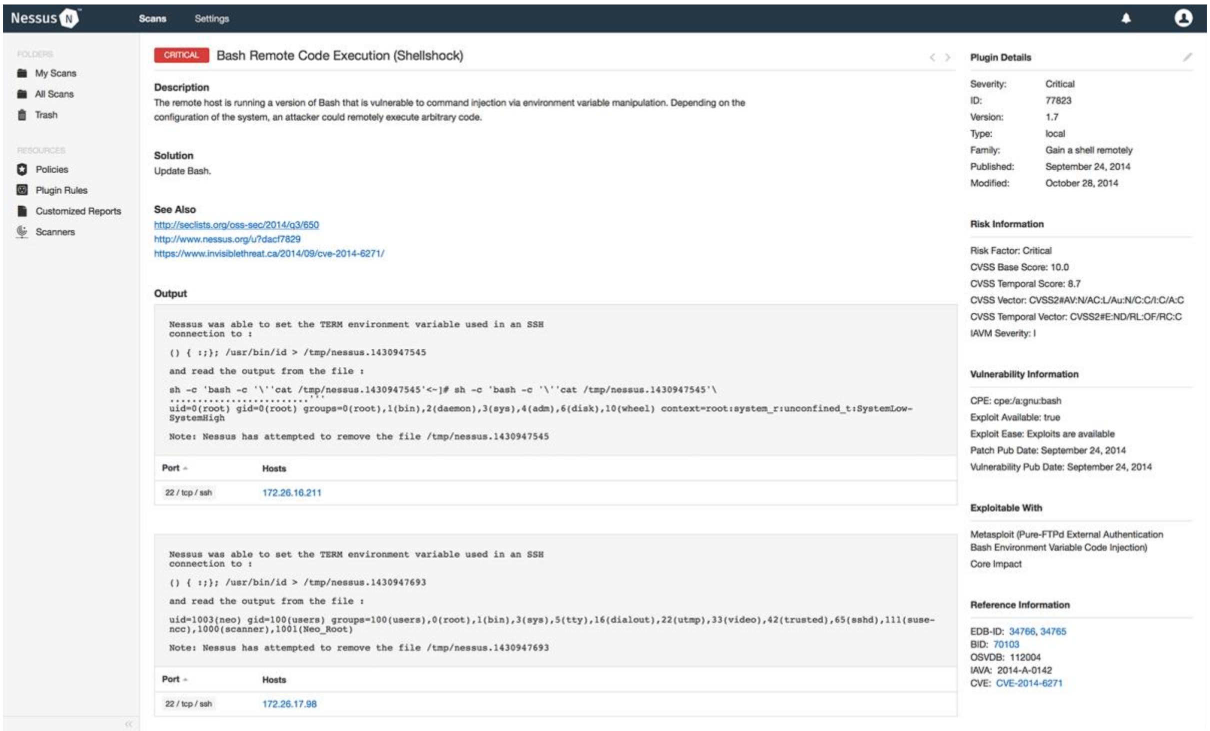
Nikto is built on LibWhisker2 (by RFP) and can run on any platform which has a Perl environment. It supports SSL, proxies, host authentication, attack encoding and more. Because many servers do not properly adhere to RFC standards and return a 200 "OK" response for requests which are not found or forbidden, this can lead to many false-positives.

I.5.2.5 Nessus [2.6]

Nessus was built from the ground-up with a deep understanding of how security practitioners work. Every feature in Nessus is designed to make vulnerability assessment simple, easy and intuitive. The result: less time and effort to assess, prioritize, and remediate issues. Nessus main features are as follow:

- Up-to-date security vulnerability database: By using the command Nessus-update-plugins, The Nessus security checks database (which is updated on a daily basis) can be retrieved.
- Plug-ins: Each security test is materialized as an external plug-in, written in NASL, which means that updating Nessus does not involve downloading potentially threatening binaries from the internet.
- Scalable: Nessus will quickly exploit the systems strengths, so it can increase its scanning efficiency. The more power you give to it, the faster it will scan a network.
- NASL: Nessus includes NASL, (Nessus Attack Scripting Language) a language designed to rapidly write security test.
- Smart service recognition: It isn’t in Nessus beliefs that the target hosts will respect the IANA assigned port numbers. Thus it will identify a FTP server running on a non-standard port, or a web server running on port 8080.
- Multiple services: Nessus will test all of the services that are run twice or more by a host run.
- Full SSL support: Nessus has the capability to test SSLized services such as https, smtps, imaps, and can even be supplied with a certificate so that it can be integrated into a PKI type environment.
- Non-destructive or thorough: Nessus gives you the option to either perform a regular non-destructive security audit on a daily basis, or to throw everything you can at a remote host to test its mettle, and see how it will withstand attacks from intruders.
- The biggest user base: Nessus has approximately at least 50.000 users worldwide and there could potentially be even more, if the daily downloads of it are taken into account.

- Proven maturity: The first public release of Nessus was in 1998. The technology behind it has been extensively tested and perfected, on huge networks over time.



Bash Remote Code Execution (Shellshock)

Description
The remote host is running a version of Bash that is vulnerable to command injection via environment variable manipulation. Depending on the configuration of the system, an attacker could remotely execute arbitrary code.

Solution
Update Bash.

See Also
http://seclists.org/oss-sec/2014/q3/650
http://www.nessus.org/u?dacf7829
https://www.invisiblethreat.ca/2014/09/cve-2014-6271/

Output

```
Nessus was able to set the TERM environment variable used in an SSH
connection to :

() { :;}; /usr/bin/id > /tmp/nessus.1430947545

and read the output from the file :

sh -c 'bash -c '\''cat /tmp/nessus.1430947545'<~]# sh -c 'bash -c '\''cat /tmp/nessus.1430947545'\
..........................'''
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel) context=root:system_r:unconfined_t:SystemLow-
SystemHigh

Note: Nessus has attempted to remove the file /tmp/nessus.1430947545
```

Port	Hosts
22 / tcp / ssh	172.26.16.211

```
Nessus was able to set the TERM environment variable used in an SSH
connection to :

() { :;}; /usr/bin/id > /tmp/nessus.1430947693

and read the output from the file :

uid=1003(neo) gid=100(users) groups=100(users),0(root),1(bin),3(sys),5(tty),16(dialout),22(utmp),33(video),42(trusted),65(sshd),111(suse-
ncc),1000(scanner),1001(Neo_Root)

Note: Nessus has attempted to remove the file /tmp/nessus.1430947693
```

Port	Hosts
22 / tcp / ssh	172.26.17.98

Plugin Details

Severity:	Critical
ID:	77823
Version:	1.7
Type:	local
Family:	Gain a shell remotely
Published:	September 24, 2014
Modified:	October 28, 2014

Risk Information

Risk Factor: Critical
CVSS Base Score: 10.0
CVSS Temporal Score: 8.7
CVSS Vector: CVSS2#AV:N/AC:L/Au:N/C:C/I:C/A:C
CVSS Temporal Vector: CVSS2#E:ND/RL:OF/RC:C
IAVM Severity: I

Vulnerability Information

CPE: cpe:/a:gnu:bash
Exploit Available: true
Exploit Ease: Exploits are available
Patch Pub Date: September 24, 2014
Vulnerability Pub Date: September 24, 2014

Exploitable With

Metasploit (Pure-FTPd External Authentication Bash Environment Variable Code Injection)
Core Impact

Reference Information

EDB-ID: 34766, 34765
BID: 70103
OSVDB: 112004
IAVA: 2014-A-0142
CVE: CVE-2014-6271

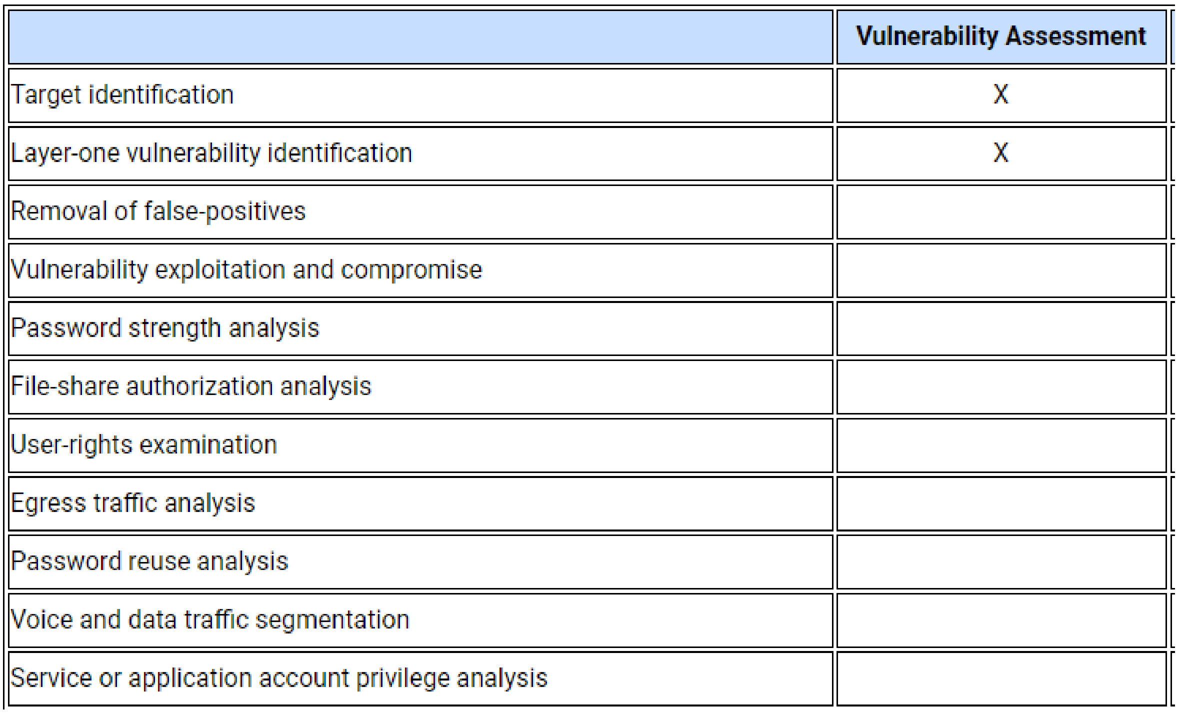
Figure I.5.2.5.a. Nessus detail plugins scan result

I.5.3. Disadvantages Of Those Tools.

In a vulnerability assessment, an automated tool scans the IT infrastructure and reports the results. The tool's job is to identify all systems and the associated applications and services they are running. Based on this information, the tool attempts to identify issues such as missing patches, default passwords, and known exploits.

All the problems the tool has identified are then presented in a vulnerability assessment report. Note that a typical vulnerability assessment doesn't include confirmation or validation of the identified issues, so the tool's accuracy is often not verified. Rather than being removed, false-positive findings are usually left for IT administrators to determine whether they are truly issues.

A vulnerability assessment does not explore a purported issue's impact outside of rudimentary factors that are often based on tool output. For example, a vulnerability scanning tool would identify a weak password in a database and rank it as a high-risk vulnerability. However, the tool would fail to take into account the fact that the database might not contain sensitive information and that the default password allows no unauthorized user to access the underlying operating system or escalate the user’s privileges.



	Vulnerability Assessment
Target identification	X
Layer-one vulnerability identification	X
Removal of false-positives	
Vulnerability exploitation and compromise	
Password strength analysis	
File-share authorization analysis	
User-rights examination	
Egress traffic analysis	
Password reuse analysis	
Voice and data traffic segmentation	
Service or application account privilege analysis	

Figure I.5.3.a. Vulnerability Assessment Functions.

Vulnerability assessments tools do not comprehensively quantify the potential impact of findings or identify the remediation issues that should be the organization’s real priorities.

I.6. The Proposed Idea Of Our Group:

After researching about the above tools and the way how a hacker or pentester use them in a real case when they penetrate a target, we come with a conclusion that the steps they take usually look like **Figure I.6.a.**

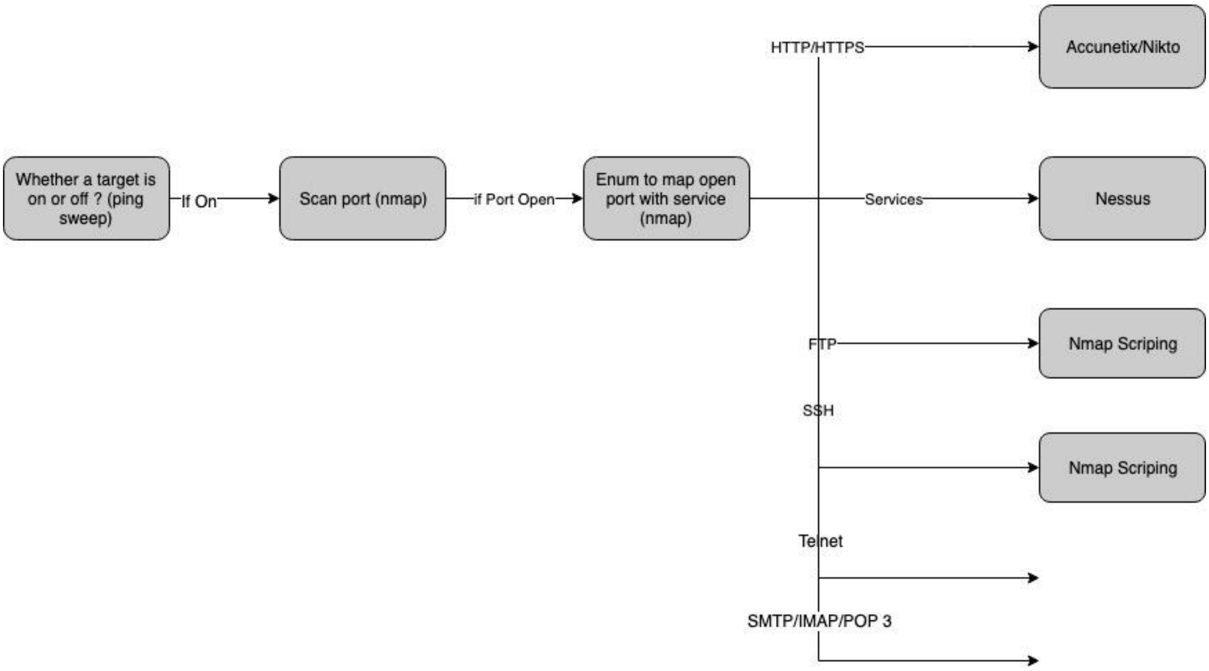


Figure I.6.a. How tools are used by hacker/pentester in real world

The first step is to test if the target is alive or not, they usually use the ping technique. If the ping response, they can conclude that this target is alive. Otherwise, they can use some other technique to determine if this target is really down at the moment or they have blocked ping requests.

After they know the host is alive, they come to the second step: port scanning. In this step, they scan the target to know which ports are open and which services are currently running on these ports, including their version as well.

Then, when they have the information about the currently running service on a specific target. For each service, they will use the appropriate tools to scan that target. For example, if there is a web service, they can use Nikto or Acunetix, if the target has an FTP or SSH service, they can use NSE or Nessus. And they also usually use Nessus to scan all the services of the target.

As we mentioned above, the output information of a tool is the input of the next one. So they all can’t be running at the same time and take a really long time to run all the above step with a large number of targets and it is really hard to manage all the information of that much targets.

So, they will need a solution which can do above step automatically for them. And that solution must have a feature to manage all information scanned by those tools. That is what our group wants to build.

II. IA PROJECT MANAGEMENT PLAN

II.1. Problem Setting

II.1.1. Name Of The Capstone Project:

Building A Vulnerability Management Solution For Enterprise

II.1.2. Problem Abstraction.

In fact, the process by which a hacker or pentester enters the system, they usually take the following steps:

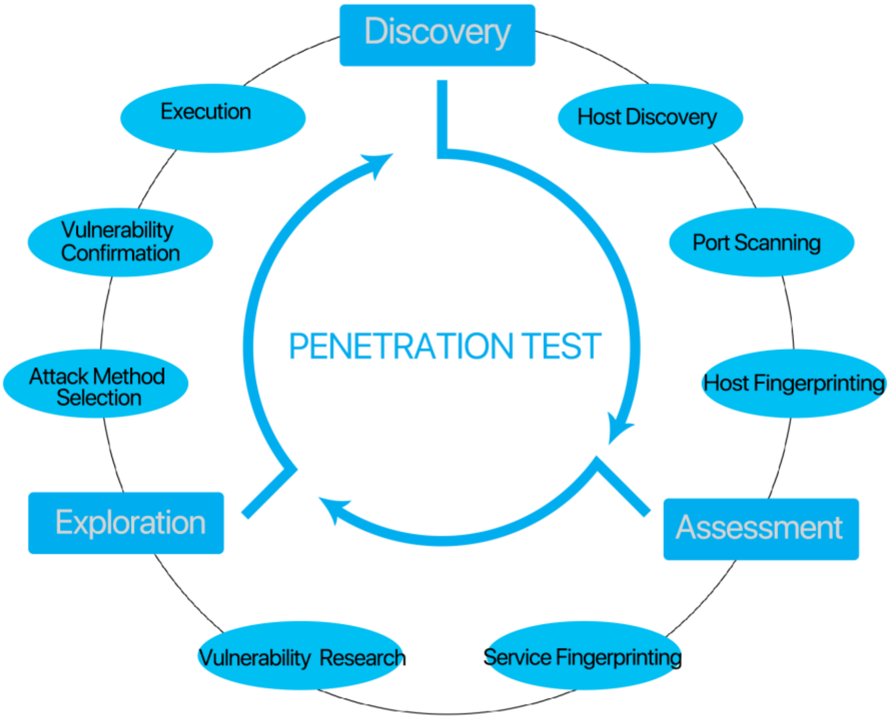


Figure II.1.2.a. Steps taken by hacker/pentester

The first step - Discovery: host discovery, port scanning, host fingerprinting, ... They must use common tools from free to paid such as Nmap, Wappalyzer, Dirsearch,... to collect information about "targets". Of course, they have to manually work on each of the above tools and on each target, then they come to the next step.

Next step - Assessment, This is the process of identifying, quantifying, and prioritizing vulnerabilities in a system after they know some information about the system. They will have to use another kind of tool such as **Nikto**, **Acunetix**,

Nessus… Each of them will scan each field. For example, **Nikto** and **Acunetix** will scan web service, **Nessus** will scan all the services which are running on the target machine.

When some vulnerabilities are scanned, they come to the next step, Exploitation. This step is about how they can use those vulnerabilities to attack the device. There are some cases that vulnerabilities that exist but the attacker can not exploit because the developer had fixed that bug, but the vulnerability assessment tool they used do not know about that. But if some vulnerabilities can be exploited, it would be a nightmare to your system and you will never like this to happen.

Don't stop there, what if your IT system has thousands of computers, routers, firewalls, and mobile devices? In addition to concrete problems involving **management** and **time optimization** is put on top.

So we decide to build a vulnerability management solution to have not just pentester but also IT administrators to always know the current vulnerability state of their system. With our solution, simple click - all the steps they need to do before will automatically do.

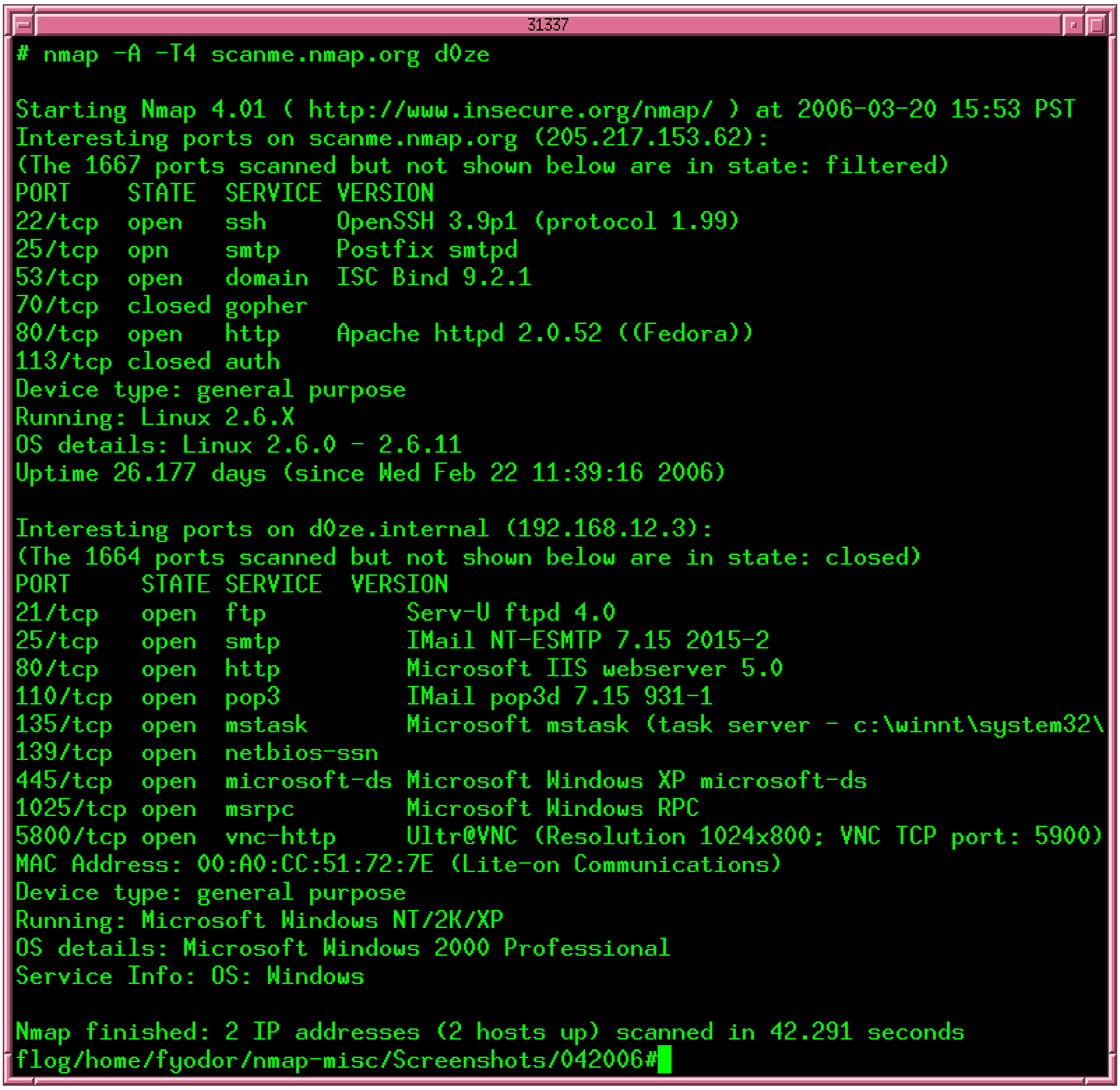
In the end, it's **easy** to manage and **running time** is what we want to emphasize here.

II.1.3. Project Overview.

II.1.3.1. Current Situation

Just imagine, What will happen if a pentester want to do a vulnerability assessment on some target?

Surely that first thing he will do is get some information about that target with **nmap**. Then he will get some information about which ports are open on that target, and which service is currently running and also if he is lucky, he can also get the version of that service or even the **CPE** ***(****Common Platform Enumeration)* of that service.

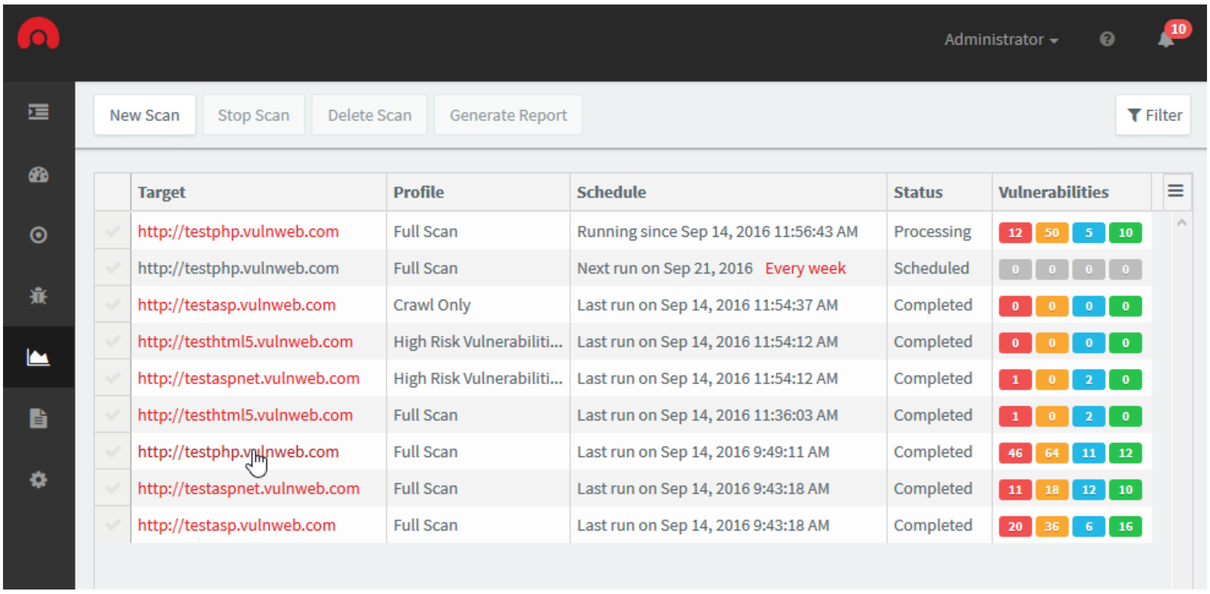


```
# nmap -A -T4 scanme.nmap.org d0ze

Starting Nmap 4.01 ( http://www.insecure.org/nmap/ ) at 2006-03-20 15:53 PST
Interesting ports on scanme.nmap.org (205.217.153.62):
(The 1667 ports scanned but not shown below are in state: filtered)
PORT     STATE  SERVICE VERSION
22/tcp   open   ssh     OpenSSH 3.9p1 (protocol 1.99)
25/tcp   opn    smtp    Postfix smtpd
53/tcp   open   domain  ISC Bind 9.2.1
70/tcp   closed gopher
80/tcp   open   http    Apache httpd 2.0.52 ((Fedora))
113/tcp closed auth
Device type: general purpose
Running: Linux 2.6.X
OS details: Linux 2.6.0 - 2.6.11
Uptime 26.177 days (since Wed Feb 22 11:39:16 2006)

Interesting ports on d0ze.internal (192.168.12.3):
(The 1664 ports scanned but not shown below are in state: closed)
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          Serv-U ftpd 4.0
25/tcp    open  smtp         IMail NT-ESMTP 7.15 2015-2
80/tcp    open  http         Microsoft IIS webserver 5.0
110/tcp   open  pop3         IMail pop3d 7.15 931-1
135/tcp   open  mstask       Microsoft mstask (task server - c:\winnt\system32\
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds Microsoft Windows XP microsoft-ds
1025/tcp open  msrpc        Microsoft Windows RPC
5800/tcp open  vnc-http     Ultr@VNC (Resolution 1024x800; VNC TCP port: 5900)
MAC Address: 00:A0:CC:51:72:7E (Lite-on Communications)
Device type: general purpose
Running: Microsoft Windows NT/2K/XP
OS details: Microsoft Windows 2000 Professional
Service Info: OS: Windows

Nmap finished: 2 IP addresses (2 hosts up) scanned in 42.291 seconds
flog/home/fyodor/nmap-misc/Screenshots/042006#
```

Figure II.3.1.a. Sample nmap output when scanning a target.

And after he has some information about the target, what he will do next? It depends on what the results show him. We will show some cases here:

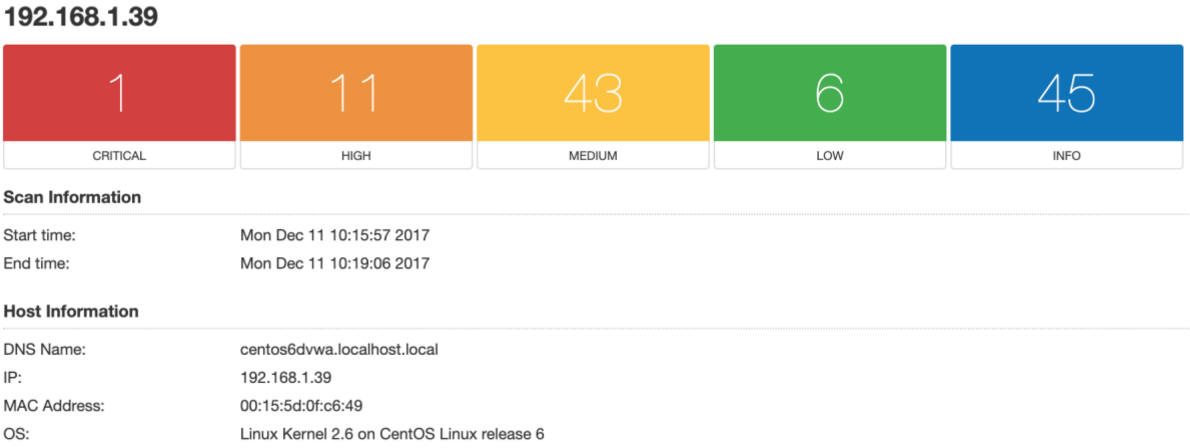
- If there is a service which he has full information, it means he got the **CPE** of that service, he will do a google search to know that if that **CPE** has any vulnerability or not. If that service has some exploitable vulnerability he will exploit that service.
- If **SSH, FTP** or some well-known service is running, he will use **NSE** *(Nmap Scripting Engine)* to test if that ssh service has some vulnerability or not.
- If there is a **web-server** running on that target. He will use some website vulnerability assessment tools like **Nikto**, **SkipFish** or **Acunetix** to test if that webserver has any exploitable vulnerabilities or not.



Target	Profile	Schedule	Status	Vulnerabilities
http://testphp.vulnweb.com	Full Scan	Running since Sep 14, 2016 11:56:43 AM	Processing	12 50 5 10
http://testphp.vulnweb.com	Full Scan	Next run on Sep 21, 2016 Every week	Scheduled	0 0 0 0
http://testasp.vulnweb.com	Crawl Only	Last run on Sep 14, 2016 11:54:37 AM	Completed	0 0 0 0
http://testhtml5.vulnweb.com	High Risk Vulnerabiliti...	Last run on Sep 14, 2016 11:54:12 AM	Completed	0 0 0 0
http://testaspnet.vulnweb.com	High Risk Vulnerabiliti...	Last run on Sep 14, 2016 11:54:12 AM	Completed	1 0 2 0
http://testhtml5.vulnweb.com	Full Scan	Last run on Sep 14, 2016 11:36:03 AM	Completed	1 0 2 0
http://testphp.vulnweb.com	Full Scan	Last run on Sep 14, 2016 9:49:11 AM	Completed	46 64 11 12
http://testaspnet.vulnweb.com	Full Scan	Last run on Sep 14, 2016 9:43:18 AM	Completed	11 18 12 10
http://testasp.vulnweb.com	Full Scan	Last run on Sep 14, 2016 9:43:18 AM	Completed	20 36 6 16

Figure II.1.3.1.b. Sample Acunetix summary result

- If he has some powerful tool like **Nessus**, he can use **Nessus** on the target and select which plugin he wants to run and then wait for the result to know that if there is any service have any vulnerabilities or not?



192.168.1.39

CRITICAL	HIGH	MEDIUM	LOW	INFO
1	11	43	6	45

Scan Information

Start time:	Mon Dec 11 10:15:57 2017
End time:	Mon Dec 11 10:19:06 2017

Host Information

DNS Name:	centos6dvwa.localhost.local
IP:	192.168.1.39
MAC Address:	00:15:5d:0f:c6:49
OS:	Linux Kernel 2.6 on CentOS Linux release 6

Figure II.1.3.1.c. Sample Nessus summary result

We can see a problem here that he has to spend some time to make a decision about which tools he will use next and in which situation. This can lead to some mistakes when he forgets some steps. And a more serious problem will arise when he deal with 100 or more targets. How can he know which result in this tool is sync with the other results in other tools? And it also really easy for a human to make mistakes when dealing with this large number of target.

III.1.3.2. The Proposed Solution

The goal of our group is to create a solution to deal with the above situation. Our solution will do all the above steps automatically, then all the results from tools will be merged to only one result. The administrator now can view every result in just one place. This solution must be fast and safe. It also has to ease to use and can be extendable to speed up more easily.

We also have to use a database which is easy to use and help the administrator in searching and analyzing data. So we will use ElasticSearch and Kibana because it is familiar with a lot of administrators.

Our project also needs a message queue to transfer the message from the output of this tool to input of another tool. So we decided to use Kafka because it is easy to use and it's an active community.

To give the project power of extendable for the ease of speeding up. We decided to have a model have a master server to assign tasks to other agents server, so we can have any number of agents, the more the better, the more the faster.

For the ease of use for users, we provide a web portal to visualize everything on the web interface. So the administrator now can manage everything and show all the result in just one place.

So our model will look like Figure III.1.3.2.a below. For more details, we will describe in section V. Specifications, Development and Implementation Plan

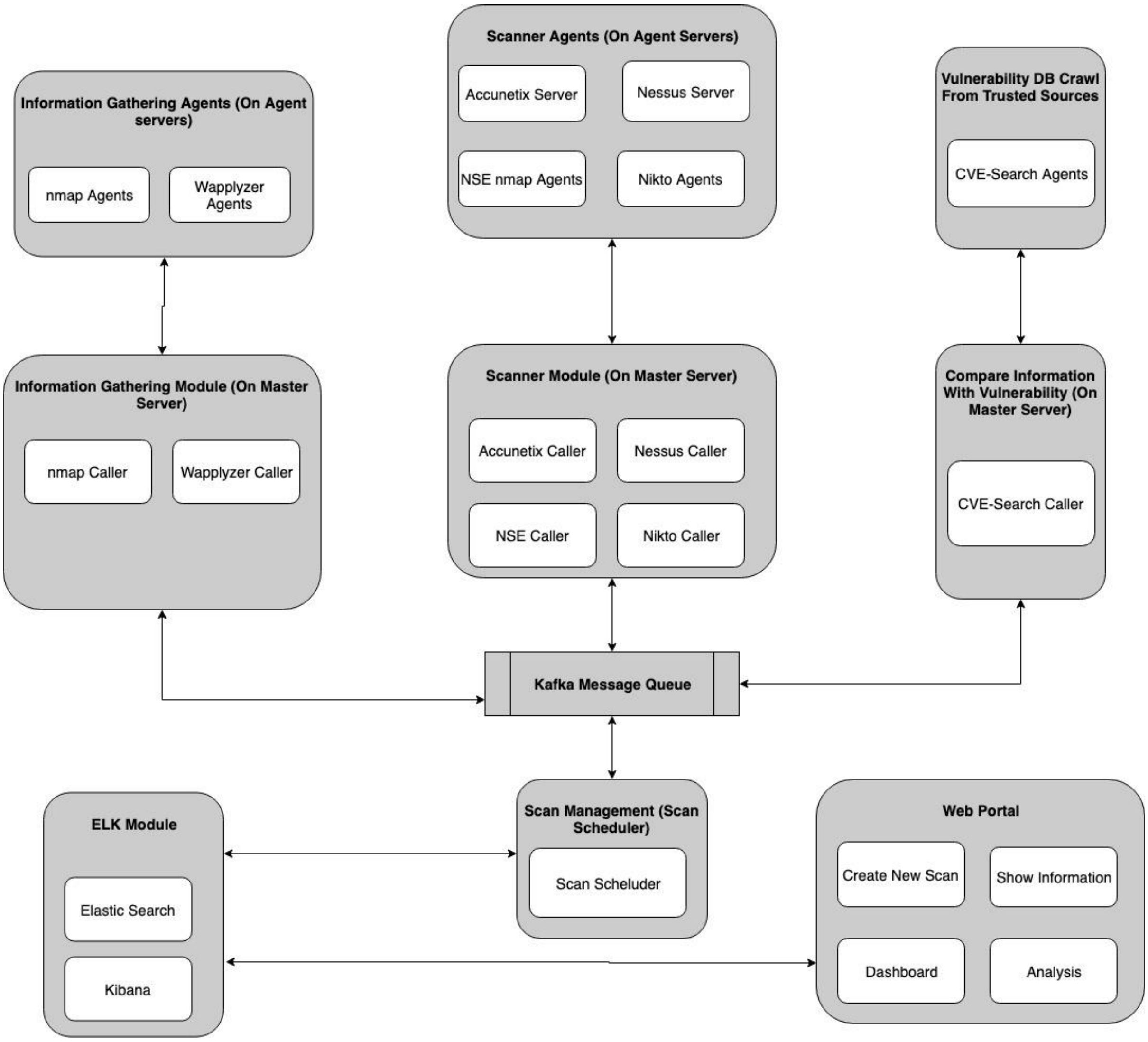


Figure III.1.3.2.a. A logical model of the project

II.2. Project Organization.

II.2.1. Agile Process Model.

In earlier days the Iterative Waterfall model was very popular to complete a project. But nowadays developers face various problems while using it to develop software. The main difficulties included handling change requests from customers during project development and the high cost and time required to incorporate these changes. To overcome these drawbacks of Waterfall model, in the mid-1990s the Agile Software Development model was proposed.

The Agile model was primarily designed to help a project to adapt to change requests quickly. So, the main aim of the Agile model is to facilitate quick project completion. To accomplish this task, agility is required. Agility is achieved by fitting the process to the project, removing activities that may not be essential for a specific project. Also, anything that is wastage of time and effort is avoided.

So our team decided to use the Agile Process Model as our solution process model.

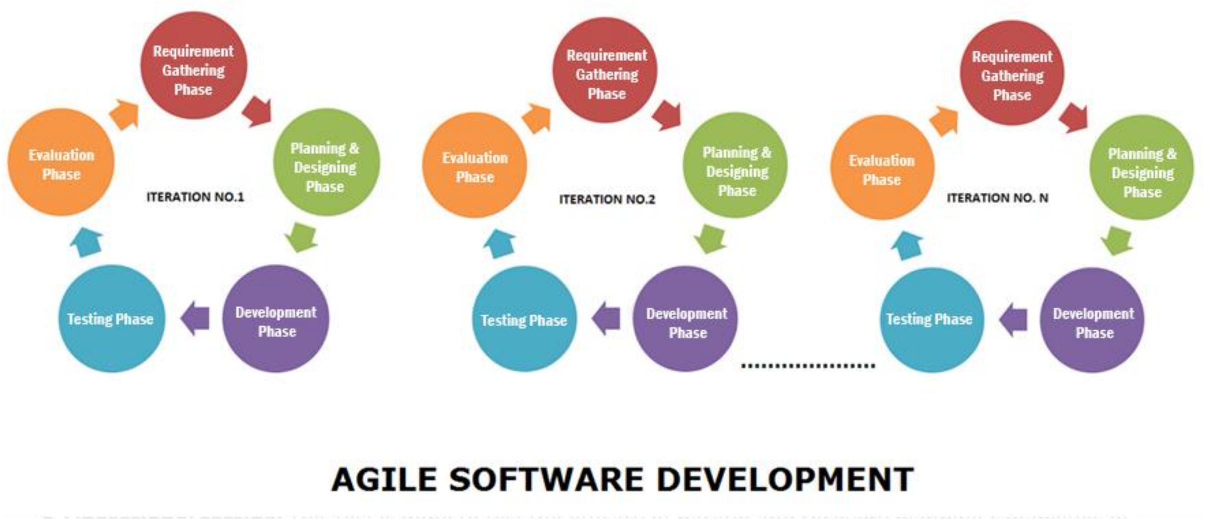


Figure II.2.1.a. Process of Agile Process Model

In the Agile model, the requirements are decomposed into many small parts that can be incrementally developed. The Agile model adopts Iterative development. Each incremental part is developed over an iteration. Each iteration is intended to be small and easily manageable and that can be completed within a couple of weeks. At a time one iteration is planned, developed and deployed to the customers. Long-term plans are not made.

- **Requirement Gathering Phase**: Like most development projects, the first step is to go through the initial planning stage to outline the requirements, objectives and initial setup of the project.
- **Planning & Designing Phase**: After planning is complete, an analysis is made to give the appropriate business logic, database models. The design phase also occurs here, setting up any technical requirements (language, data layer, service, etc.).
- **Development Phase**: After the planning and analysis has been completed, actual implementation and coding can begin.
- **Testing Phase**: when the software has been coded and implemented, the next step is to go testing to identify and locate any potential bugs or issues.

- **Evaluation Phase**: When the previous stages are completed, it is time for everyone to evaluate what the project has done and what needs to be changed.

Advantages of Agile Process Model:

- Working through Pair programming produce well written compact programs which have fewer errors as compared to programmers working alone.
- It reduces the total development time of the whole project.
- Customer representative gets the idea of updated software products after each iteration. So, it is easy for him to change any requirements if needed.

Disadvantages of Agile Process Model:

- Due to the lack of formal documents, it creates confusion and important decisions taken during different phases can be misinterpreted at any time by different team members.
- Due to the absence of proper documentation, when the project completes and the developers are assigned to another project, maintenance of the developed project can become a problem.

II.2.2. Role And Responsibilities.

Phases	Name	Roles	Responsibilities
Requirement Gathering	Nguyễn Anh Việt	Team leader	- Gather team members and supervisors. - Identify goals and objectives of project. - Identify the rules for team members. - Research about solution process model. - Collect and research information to find out an appropriate project.
	Trần Anh Đức	Team member	- Determine and research about technologies will be used in the project.
	Nguyễn Khắc Hùng	Team member	- Research and planning timeline. - Identify goals and objectives of project.

	Lê Đình Mạnh	Team member	- Identify goals and objectives of project. - Collect and research information to find out an appropriate project.
	Nguyễn Đức Anh	Team member	- Collect and research information to find out an appropriate project. - Identify goals and objectives of project.
Planning & Designing	Nguyễn Anh Việt	Team leader	- Research about appropriate technologies and determine which one will be used. - Design database for elasticsearch database - Research and design the logical model of project.
	Trần Anh Đức	Team member	- Research about how to merge all the scanning tools, and which one will be used in the project. - Research about some similar projects.
	Nguyễn Khắc Hùng	Team member	- Design front-end mockup. - List threats, risks affecting the system.
	Lê Đình Mạnh	Team member	- Research about front-end technologies (Flask) - List threats, risks affecting the system.
	Nguyễn Đức Anh	Team member	- Research about how acunetix and nessus works. - List threats, risks affecting the system.
Development and Deployment	Nguyễn Anh Việt	Team Leader	- Develop and operate service module. - Develop and operate some agents module. - Develop and operate webportal. - Develop and operate elasticsearch database. - Operate kafka message queue. - Assigned task for members.
	Trần Anh Đức	Leader Agents/ Member	- Develop and operate all agents module. - Leader of agent module. - Develop and operate elasticsearch database. - Operate kafka message queue. - Assigned task for members.
	Nguyễn Khắc Hùng	Leader Web/ Member	- Develop and operate webportal. - Leader of webportal - Develop and operate elasticsearch database. - Assigned task for members.

	Lê Đình Mạnh	Team member	- Develop and operate webportal. - Develop and operate some agents module.
Testing	Nguyễn Anh Việt	Team leader	- Testing, debug and operate service module. - Testing, debug and operate some agents module. - Testing, debug and operate webportal. - Testing, debug and operate elasticsearch database. - Testing, debug and operate kafka message queue.
	Trần Anh Đức	Team member	- Testing, debug and operate all agents module. - Testing, debug and operate elasticsearch database. - Testing, debug and operate kafka message queue.
	Nguyễn Khắc Hùng	Team member	- Testing, debug and operate webportal. - Testing, debug and operate elasticsearch database.
	Lê Đình Mạnh	Team member	- Testing, debug and operate webportal. - Testing, debug and operate some agents module.
Evaluation and report	Nguyễn Anh Việt	Team leader	- Collect information and summarize the document. - Writing outline, assigned task for members.
	Trần Anh Đức	Team member	- Collect information and summarize the document. - Gather data, information for the document
	Nguyễn Khắc Hùng	Team member	- Collect information and summarize the document. - Capture image of front-end feature.
	Lê Đình Mạnh	Team member	- Collect information and summarize the document.
	Nguyễn Đức Anh	Team member	- Collect information and summarize the document.

II.2.3. Tools And Techniques Used.

Tools and techniques		
Development	Visual Studio Code	IDE for Python 3, html, css, js, jQuery
	Github	Source code version control
	Sublime Text	Sublime Text is a sophisticated text editor for code, markup and prose
	Ubuntu 16.04 ubuntu	Operating System for development and coding
	Docker docker	Docker is a set of coupled software-as-a-service and platform-as-a-service products that use operating-system-level virtualization to develop and deliver software in packages called containers
	ElasticSearch elastic	Elasticsearch is a search engine based on the Lucene library. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents.

	Kafka kafka	Used for building real-time data pipelines and streaming apps
	Trello Trello	Keep tracking of work
	Chrome	Web browser, testing environment
	Xinetd xinetd	xinetd performs the same function as inetd: it starts programs that provide Internet services. Instead of having such servers started at system initialization time, and be dormant until a connection request arrives
Programming Languages	Python 3 python™	Python is an interpreted, high-level, general-purpose programming language.
	HTML HTML 5	HTML is the standard markup language for creating Web pages. HTML stands for Hyper Text Markup.
	CSS CSS 3	CSS is a language that describes the style of an HTML document. CSS describes how HTML elements should be displayed.
	JavaScript JS 5	JavaScript, often abbreviated as JS, is a high-level, interpreted programming language that conforms to the ECMAScript specification. JavaScript has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first-class functions

	jQuery	jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.
Communication	Skype	Group video calls, chat, file sharing. Contact with supervisor.
	Facebook messenger	Group video calls, chat, file sharing
	Mobile phone	Instant call
	Gmail	Share link resources and link between only team members.
Document, Take note	Word	Writing document and report for the project.
	PowerPoint	Create and design a slideshow for presenting a capstone project.
	Draw.io	Create and design diagramming and vector graphic for the document

Python library	Flask	Flask is a lightweight WSGI (Web Server Gateway Interface) web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications.
	Json	JSON is a syntax for storing and exchanging data. JSON is text, written with JavaScript object notation.
Scanning Tools	Nmap	Nmap ("Network Mapper") is a free and open source (license) utility for network discovery and security auditing. Many systems and network administrators also find it useful for tasks such as network inventory, managing service upgrade schedules, and monitoring host or service uptime.
	NSE	Nmap Scripting Engine (NSE) is one of Nmap's most powerful and flexible features. It allows users to write (and share) simple scripts to automate a wide variety of networking tasks. Those scripts are then executed in parallel with the speed and efficiency you expect from Nmap. Users can rely on the growing and diverse set of scripts distributed with Nmap, or write their own to meet custom needs.
	Wappalyzer	Wappalyzer is a cross-platform utility that uncovers the technologies used on websites. It detects content management systems, ecommerce platforms, web frameworks, server software, analytics tools, ...
	CVE Search	cve-search is a tool to import CVE (Common Vulnerabilities and Exposures) and CPE (Common Platform Enumeration) into a MongoDB to facilitate search and processing of CVEs. The main objective of the software is to avoid doing direct and public lookups into the public CVE databases.
	Acunetix	Acunetix is an end-to-end web security scanner.

	Nikto **Nikto**	Nikto is a free software command-line vulnerability scanner that scans web-servers for dangerous files/CGIs, outdated server software and other problems. It performs generic and server type specific checks.
	Nessus	Vulnerabilities assessment tool. Was introduced is #1 payment tools.

II.3. Project Management Plan

II.3.1. Tasks:

II.3.1.1 Project initialization, planning.

- Description: Initialization idea of project, identify long term goal of our project is to create a solution (including a web portal to interact with user more easily) to help enterprises management vulnerabilities. We need to develop a more helpful solution which can solve other vulnerability assessment tools disadvantages.
- Deliverables: After planning and research, we decided to create a solution to provide enterprises a way to manage vulnerabilities.
- Resources: Internet, man-power.
- Risk: Power-off, lost internet.

II.3.1.2 Technical studies.

- Description: Find and research existing technologies which suitable for the project, choose a suitable programming language and process model.
- Deliverables: Understand the fundamentals of the technology they use, how to use them, the advantages and disadvantages of each technology.
- Resources: Internet, book online, docs online, man-power.
- Risk: Power-off, lost internet, lack of knowledge.

II.3.1.3 Design and analysis project

- Description: Design mockup of web portal, database model, project logical model, how each feature works and how to interact with users.
- Deliverables: Database model, mockup of web portal, database model, project logical model.
- Resources: draw.io, kibana, elastic-search, chrome, man-power.
- Risk: lack of knowledge, time to research from the beginning.

II.3.1.4 Implementation

- Description: Design and implementation provide a tools with web GUI for vulnerabilities management in enterprise.
- Resources: VS code, SublimeText, git, internet, chrome, man-power, ….
- Risk: an unexpected error occurs, unsaved-code, push wrong repositories on GitHub, team work, lack of knowledge.

II.3.1.5 Testing and Fix Bug

- Description: Testing every function of solution, does it work properly or wrongly, search for as many as possible bugs and fix it or minimize impact of that bugs if it hard or impossible to fix.
- Deliverables: Test all functions, fix as many as possible bugs.
- Resources: VS code, SublimeText, Git, Internet, Chrome.
- Risk: unsaved-code, push wrong repositories on GitHub, some bugs can not fix.

II.3.1.6 Deployment

- Description: Deploy projects on 2 companies X and Y. Detect problems arising when implementing in real enterprises, difficulties and problems. feasibility of implementation, difficult level of deployment.
- Deliverables: Knowing what is needed to be able to deploy in the enterprise.
- Resources: Hardware, github, internet.
- Risk: Hardware requirement does not enough, external and internal networks in the enterprise, speeds of the Internet.

II.3.1.7 Document development

- Description: Detailed presentation of what the project has done.
- Deliverables: a complete docs with no spelling errors.
- Resources: Word, google docs, translator, spell-check extension.
- Risk: unsaved Word.

II.3.2. Tasks Schedule Sheet: Assignments and Timetable

Task	SubTask	Details	Assignment	Time
Start Project : 15/04/2019				
Project initialization	**Idea brainstorming**			**16/04 – 30/04/2019**
	Idea outline	Planning:	VietNA, AnhND, HungNK, ManhLD, DucTA	
		Schedule of weekly meeting		
		Preparation schedule, study.		
		Schedule of project development		
		Document implementation schedule		
Technical studies	**Existing solution study**	Wappalyzer	DucTA	**16/04 – 30/04/2019**
		Nessus	AnhND	
		Watchdog	DucTA	
		NSE	AnhND	
		Nmap	ManhLD	
		Shodan, NetCraft,..	HungNK	
		Accunetix	AnhND	
	Techniques	Elastic search Stack: LogStash, Elastic Search, Kibana	VietNA	**16/04 - 10/05/2019**
		Exploit DataBase	HungNK	
		Write API and APIdocs in Flask	ManhLD	

		Web Framework: Django vs Flask	VietNA	
		Realtime, notification in python web framework... (notification)	ManhLD	
Design and analysis project	**Design Activity Diagram**	Logical Flow Diagram	VietNA	**16/04-15/05/2019**
		WebApp Flow Diagram	VietNA	
	Design DataBase	Design DataBase schema	VietNA, AnhND, HungNK, ManhLD, DucTA	
	UI Design	Design mockup for frontend	HungNK	
Implementation	**Module**	Module Nmap Agent -Caller	ManhLD	**16/05 - 30/6/2019**
		Crawl Data from exploitDB	HungNK	
		Building ELK	DucTA	
		Module ScanningPortCaller - ScanningPortAgent	VietNA	
		Nessus Module Agent	DucTA	
		Get data from kafka nmapOutput and push them to ElasticSearchDB	VietNA	
		Get data from output of nmap and compare with CVESearch	VietNA	
		ReParse Json from XML (nmap output)	DucTA	
		Get data from CVESearch and push them to ElasticSearchDB	VietNA	
		Add Try Excecpt to every where can have a excetion and logging the temporary message	VietNA	
		Create a logger mechanism to know how service is running	VietNA	
		All Kafka Producer use the same Producer() to work with thread safe	VietNA	
		Nmap NSE agent Module	VietNA	

		Writing deploy script nse for agent	VietNA	
		Create custom Nessus Profile Scanner	DucTA	
		Wapllyzer Agent	DucTA, VietNA	
		Acunetix module	VietNA	
		Nikto module in master server	VietNA	
		Code a wrapper to create a scan and get result from Acunetix	DucTA	
	Webportal	Mockup demo data from Elasticsearch to Web portal	HungNK, ManhLD	**15/06 - 30/07/2019**
		Web portal, searching	HungNK	
		Web portal Sort Data	ManhLD	
		Display NSE Data	ManhLD	
		Display Acunetix Data	HungNk	
		Display Wappalyzer Data	ManhLD	
		Display CVE search Data	ManhLD	
		Paging in Elasticsearch	VietNa	
		Building Page To Funtion "New Scan"	HungNK	
		Design front-end navigate, header, foooter	ManhLD	
		Display Vulnerabilities -Detail Data	HungNK, ManhLD	
		Drawing PieChart for [Total vulnerabilities]	VietNA	
		Clean & Beautiful Web Interface Design	HungNK	
Testing and Fix Bug	**Adopt New DataBase**	Adopt new CVE_Search Database	ManhLD	**16/05- 30/07/2019**
		Adopt new Wappalyzer Database	ManhLD	
		Adopt Nessus Module	VietNA	
		Adopt new NSE Database	HungNK	
		Adopt new Nmap Database	HungNK	

		Adopt new Acunetix Database	HungNK	
	FixBug	Bug Logic	VietNK, DucTA	**20/07-05/08/2019**
		Bug Font-end	HungNK	
Deployment	**A Company [Name Hided]**		VietNA, AnhND, HungNK, ManhLD, DucTA	**27/06 , 17/07/19**
	A Company [Name Hided]		AnhND	
Document development	**I. INTRODUCTION**	I. 1 - I.4 :The Initial Idea Of Our Group	VietNA	**15/07-10/08/2019**
		I.5. A Brief Overview Of Current Vulnerabilities Assessment Tools On The World	AnhND	
		I.6 The Proposed Idea Of Our Group	AnhND	
	II. IA PROJECT MANAGEMENT PLAN	II.1. Problem Setting	ManhLD	
		II.2. Project Organization.	VietNA	
		II.3. Project Management Plan	AnhND, ManhLD	
	III. RISK ASSESSMENT	III.1 The Need Of Assessment	AnhND	
		III.2 Identify Critical Information Assets	HungNK	
		III.3 -III.4 Risk Identification,Risk Analysis, Control Identification and Assessment	DucTA	
	IV. RISK MANAGEMENT PLAN	IV.1 - IV.3 : Objectives of RMP, Assigning Responsibilities, Describing Procedures and Schedules for Accomplishment	ManhLD	
		IV.4 Reporting Requirements	HungNK	
	V . SPECIFICATIONS, DEVELOPMENT AND IMPLEMENTATION PLAN	V.3. How Does A Target Being Scanned When A User Create New Scan. V.1. Logical Model Of Our System. V.5. Config Files. V.6How Logging Have Been Saved And	VietNA	

		Managed? V.2. Master Service, How We Assign Jobs For Agents And How Do Master And Agent Can Exchange Data?		
	VI. PROJECT VALIDATION	VI.1 - VI.3 Project Idea ,Result , Future	AnhND	
Project finished: 26/08/2019				

II.3.3. All Meeting Minutes.

Subject	Gather members and supervisor
Date	15/04/2019
Time	15h - 17h
Location	21st floor of the KeangNam building
Supervisor	NamHB
Attendees	VietNA, HungNK, DucTA, ManhLD, AnhND
Key points discussed	
No.	**Topic**
1	Introduction to membership.
2	Working in group and discuss project topic, project name.
3	Research and assigning the work.

Subject	Weekly meeting
Date	15/04 - 06/05/2019
Time	15h - 17h
Location	Azzan Coffee, Hoa Lac Campus
Supervisor	NamHB

Attendees	VietNA, HungNK, DucTA, ManhLD, AnhND
Key points discussed	
No.	**Topic**
1	Review assigned tasks, assign new tasks
2	Demo works done from previous weeks
3	Find difficulties and find solutions for it

Subject	Weekly meeting
Date	06/05 - 12/06/2019
Time	10h- 12h
Location	P204, Beta Building ,Hoa Lac Campus
Supervisor	NamHB
Attendees	VietNA, HungNK, DucTA, ManhLD, AnhND
Key points discussed	
No.	**Topic**
1	Review assigned tasks, assign new tasks
2	Discuss about model of the project
3	Demo works done from previous weeks
4	Find difficulties and find solutions for it

Subject	Weekly meeting
Date	13/06 - 15/07/2019

Time	10h- 12h
Location	R2006, Sky City Tower
Supervisor	NamHB
Attendees	VietNA, HungNK, DucTA, ManhLD, AnhND
Key points discussed	
No.	**Topic**
1	Review assigned tasks, assign new tasks
2	Discuss about webportal, master service and agent service
3	Find difficulties and find solutions for it
4	Demo works done from previous weeks

Subject	Weekly meeting
Date	15/07 - 31/08/2019
Time	13h- 15h
Location	HB205R, Alpha Building ,Hoa Lac Campus
Supervisor	NamHB
Attendees	VietNA, HungNK, DucTA, ManhLD, AnhND
Key points discussed	
No.	**Topic**
1	Review assigned tasks, assign new tasks
2	Review documentation and code of master service and agent services
3	Demo get result data from some enterprise.

III. RISK ASSESSMENT

III.1. The Need Of Risk Assessment

Risk assessment is a long process requires carefully review the workplace to identify vulnerabilities, threats, situations, etc. that may cause harm to the project, the people, company, or organization in order to evaluate and control it. Thus they form an integral part with the goal to answer the following questions:

- **What can happen, in which situation?**
  We will identify risks and risks factor in this process.
- **What are the consequences?**
  In this process, the estimated risk is compared against the given risk criteria to determine the significance of it.
- **How often the consequences occur?**
  Monitor the likelihood to occur of the risk to determine how to control it.
- **Is risk-controlled effectively? Can it be handled more effective?**
  In this final step, we review all the steps from the very start to prepare for the next time.

III.2. Identify Critical Information Assets

III.2.1. Information Asset Classification

The goal of Information Security is to protect the confidentiality, integrity and availability of Information Systems. Identify Critical Information Assets is the identification of information based on its sensitivity and impact on the organization. Identify Critical Information Assets to determine which security methodology is appropriate to protect that information. Critical Information should be classified into four categories:

- Public Information
- Internal Information
- Restricted Information
- Confidential Information

Information Classes	Description	Assets	Level of confidentiality
Confidential	This is the critical level,	- Vulnerabilities	Critical

	it contains important information and database we saved about vulnerabilities, how tools work.	database - Source code - Log event services - Model of system	
Restricted	If it is exploited, your organization may break down.	- Login information - Database credential	High
Internal	It contains information about project, system of our team have. If it is exploited, our system used to run code and demo will be break down	- Information about hardware, software - Contract with customers	Medium
Public	It is the lowest level, contain information about using our tool, information about project, contract of your organization. If it is exploited, organizations can suffer serious damage.	- Information about how to use services - FAQ - Help for customer	Low

III.2.1.1. Master Service Components

- **The Probability of a threat exploiting a vulnerability in an asset**

The probability of threats coming from master service components is low. Some service if not update to the latest version can be exploited by hackers or even 0-day vulnerability attack. And the entry-point to the master server is small because it just receive data from database and agents.

- **The Impact of a threat exploiting a vulnerability in an asset**

If one service been exploited, the function of web portal might work not correct. If the hacker got root by privilege escalation attack, all systems are controlled in hacker's hand.

III.2.1.2. Agent Service Components

- **The Probability of a threat exploiting a vulnerability in an asset**

The probability of threats to the agent service components is medium. Threat can come from nature: natural disasters cause power outages, physical damage to hardware, etc,... System services are also likely to be DOS attacks.

- **The Impact of a threat exploiting a vulnerability in an asset**

Although the threats are average but when it affected, the scanning and tracking vulnerabilities in real-time will stop working, which make the client unable to login or views anything on the customer’s system.

III.2.1.3 Web portal Components

- **The Probability of a threat exploiting a vulnerability in an asset**

The web application are not have many entry points to exploit or attacks but the probability of a threat exploiting are high, because it is the main interface to interact with users.

- **The Impact of a threat exploiting a vulnerability in an asset**

Hacker can use all that vulnerabilities to exploit on system server. Stolen sensitive data or use our server as a botnet or mining coin. Also the hacker can privilege escalation attack for further harming action.

III.2.1.4 ElasticSearch Database Components

- **The Probability of a threat exploiting a vulnerability in an asset**

The probability of threat coming from the ElasticSearch Database Components are usually high because it is an open-source project. In history development process of ElasticSearch there have been exist RCE vulnerability. We also have been careful with 0-day vulnerabilities.

- **The Impact of a threat exploiting a vulnerability in an asset**

If the database were leak, the whole information about systems, vulnerabilities, entry point to exploit are explored. The attacker can use that to attack any specific point in the system with that knowledge.

III.2.2. System Characterization

III.2.2.1. Logical Architecture

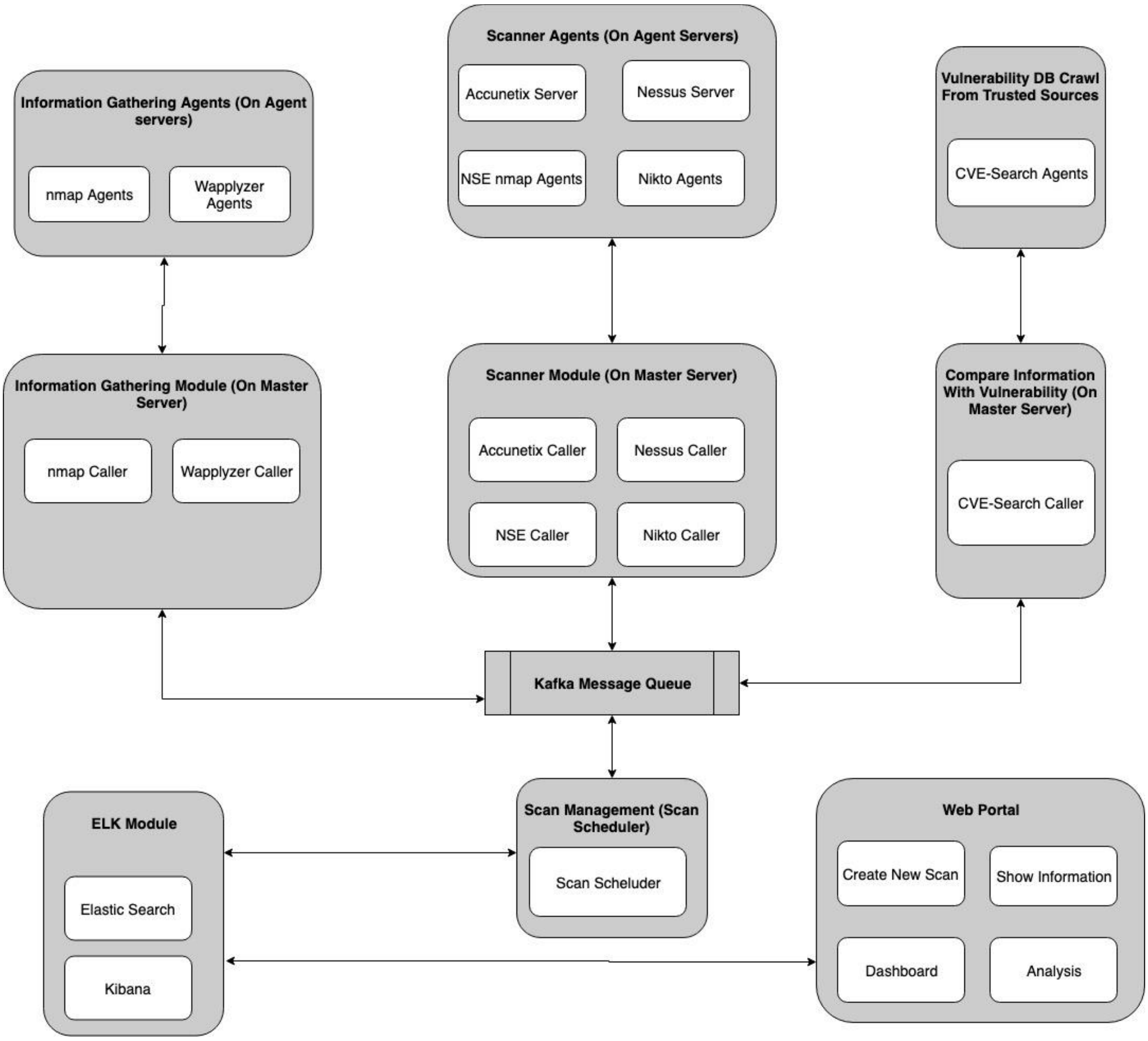


Figure III.2.2.1. A logical model of the project

III.2.2.2. System Components

System Components	Name
Team member	PC, Laptop
Agent 1	Nmap, NSE
Agent 2	Wappalyer, Nessus
Agent 3	Acunetix, Nikto

Master	Database, log events, source code tools

III.2.2.3. Users of the System

We have two types of user on the system:

- Docker account used to configuration agent machines.
- Administrator account of web application.

III.2.2.4. Security and Compliance Requirements

To ensure system security, docker account and administrator account need to comply with the following security requirements:

- Regularly update or check web application server system.
- Do not use unidentified software in the master and agent machine.
- Do not let password of account admin leak outside.
- For member of team, protect source code, model component of system and data localhost.

III.2.2.5. Information Protection Priorities

Information Protection Priorities follow by table. We calculated according to point 3 (1- Highest, 2 – Medium, 3 - Lowest).

Number	Information Assets	Priority
1	Log event services	1
2	Model of system	1
3	Login information	1
4	Contract with customers	3
5	Information about hardware, software	2
6	Information about how to use services - FAQ - Help for customer	3

III.3. Risk Identification

The following formula is often used when pairing threats with vulnerabilities:

Risk = Threat * Vulnerability [7]

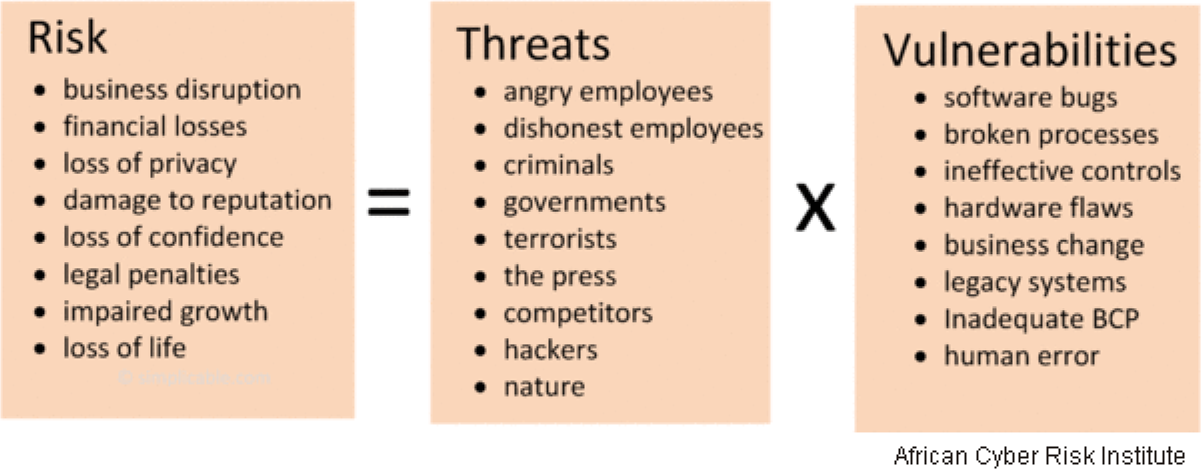


*Figure III.3.a. Risk = Threats * Vulnerabilities*

However, this isn’t a true mathematical formula. Threat and vulnerability don’t always have numerical values. Instead, the formula shows the relationship between the two.

If we can identify the value of the asset, the formula is slightly modified to:

Total Risk = Threat * Vulnerability * Asset Value [7]

III.3.1. Threat Identification

In Figure 3.3, threats can be categorized into two types: Human or Natural. Human threats can also be internal (angry employees, dishonest employees…) or external (hackers, competitors …).

Some natural threats such as power outages, natural disasters can cause the server system to stop working.

	Reason	Threat & Impact		
		Portal website	**Master service**	**Agent services**
Internal	Employees who are not satisfied with the company or organization. They can collect the data of the company or organization.	Collect and delete records on the portal and the administrator doesn’t know the vulnerability to fix. Reveal the secret information or sell them to other companies.	Change the configuration or shutdown service to make all the application stop working	Shutdown or misconfigured an agent service will make entire scan result wrong
External	The hacker who want to attack for gathering information or privilege escalation	If they can control the portal, they will know which machine on the company has vulnerabilities to attack	Denial of service attacks makes this service out of service.	Control the agent service will cause the hacker to have access to all the IP in the company if the agents have the serious vulnerabilities and a hacker can do the RCE on this machine.
Natural	The unusual reason such as weather, electric problem...	Natural factors can cause power outages, making the entire service stop working.		

III.3.2. Vulnerability Identification

Vulnerability is a weakness which can be exploited, this project has these type of services which may have vulnerability:

- Portal website
- Master service
- Agent services

The table below shows the vulnerabilities that may be encountered in the services:

Service	Vulnerability	Details
Portal website	IDOR (INSECURE DIRECT OBJECT REFERENCE)	Insecure Direct Object References allow attackers to bypass authorization and access resources directly by modifying the value of a parameter used to directly point to an object. It usually appears on a web application.
	CSRF Attack	Cross-Site Request Forgery (CSRF) is an attack technique that uses the user's authentication authority for a website. A CSRF is a hacking technique where the hacker can perform actions that require authentication like a login form.
	RCE (Remote code execution)	Remote code execution is the ability an attacker has to access someone else's computing device and make changes, no matter where the device is geographically located.
Master service and Agent services	DDoS Attack	Hacker can use DDoS attack to make all other services out of service
	Unauthorized access to service	Service can send requests to this service to run and receive the result

III.4. Risk Analysis

There are many different ways to risk analysis. In this case, we use OWASP: Risk Rating Methodology [8]

Risk = Likelihood * Impact

We have 3 most likely risks to analyze:

- Scenario 1: Internal employees destroy the data or install malicious software into the system
- Scenario 2: Hacker attacks the services system, customer
- Scenario 3: One of the services of the system stops working

III.4.1. Likelihood Assessment

When identifying potential risk, the first is to estimate "likelihood". The following is a preliminary measure of the vulnerability that a specific vulnerability exploits with an attacker. There are three levels of ability, low, medium or high.

Threat Agent Factors			
Risk	1	2	3
Skill level	6	8	9
Motive	1	1	4
Opportunity	4	8	1
Size	4	5	3
Mark	3.75	5.5	4.25

Vulnerability Factors			
Risk	1	2	3
Ease of discovery	6	5	2
Ease of exploit	4	3	3
Awareness	6	4	3

Intrusion detection	2	8	8
Mark	4.5	5	4

III.4.2. Impact Assessment

The technical impact can be broken down into factors such as confidentiality, integrity, usability, and accountability. The purpose is to estimate the impact on the system when a vulnerability is exploited.

Technical Impact Factors			
Risk	1	2	3
Loss of confidentiality	9	6	5
Loss of integrity	9	6	7
Loss of availability	9	7	6
Loss of accountability	2	7	7
Mark	7.75	6.5	6.25

III.4.3. Risk Determination (Rating)

In this section, estimates of likelihood and impact are put together to calculate the severity. The severity score scale is shown below

Likelihood and Impact Level	
0 to <3	Low
3 to <6	Medium
6 to 9	High

The rank of severity when combined:

Overall Risk Severity				
Impact	High	Medium	High	Critical
	Medium	Low	Medium	High
	Low	Note	Low	Medium
		Low	Medium	High
	Likelihood			

Below are Determining Severity table:

Determining Severity			
Risk	1	2	3
Impact	High	High	High
Likelihood	Medium	Medium	High
Risk Severity	High	High	Critical

III.5. Control Identification and Assessment

III.5.1. Risk Monitoring and Controlling

We need to monitor and control risk to:

- Ensure the implementation of risk management plans and assess the effectiveness of those plans.
- Monitoring of identified risks
- Monitor residual risks and identify new risks arising during project execution.

III.5.1.1. Inputs to Risk Monitoring and Control

Risk Monitoring and Control requires the following:

- Risk management plans
- The results of risk identification include identified risks & owners, risk responses, triggers, and warning signs

III.5.1.2. Outputs from Risk Monitoring and Control

- Updates on risks, causes and results of actual responses
- Corrective action includes the implementation of contingency plans or solutions Recommended Preventive Actions used to direct the project to comply with the project management plans
- Organizational Process Assets Updates include: Information gained through the risk management, processes are collected and kept for use by future, templates for the risk management plan, learned
- Project Management Plan Updates to the project management plans as a result of the approval of requested changes.

III.5.2. Preventive Measures

For technical measures to protect the system, major risk factors come from people, here are some safeguards for the risks that come from internal and external people.

Human Threat	Object	Control Methods
Internal	Angry employees…	Permission to use on service systems, Authentication, Authorization, Access control
External	Hacker, competitor	Using DDoS attack defense hardware or early attack detection systems

For threats from the system, we use regular backups, using the tool to check for live services. Logging faulty events in the system and fix as soon as possible.

IV. RISK MANAGEMENT PLAN

IV.1. Objectives of RMP

IV.1.1. Lists of Threats & Vulnerabilities

No.	Type	Threats & Vulnerabilities
1	Portal website	CSRF Attack
		IDOR
		RCE
2	Master service and Agent services	DDoS Attack
		Unauthorized access to service

IV.1.2. Costs associated with risks

- **Confidentiality:**

Rate	Description	Point
Low	No affect or effect a little.	1
Medium	The system can accept the risk.	2
High	Seriously affect to the system.	3

- **Integrity:**

Rate	Description	Point
Low	No affect.	1
Medium	The system can accept the risk.	2
High	Seriously affect to the system, it can’t continue work.	3

- **Availability:**

Rate	Description	Point
Low	No affect or effect a little.	1
Medium	The system can accept the risk.	2
High	Seriously affect to the system.	3

- We have 4 levels for Rating Threat/Vulnerabilities with costs:

Rate	Point	Cost
Low	P < 4.	~5.000$
Medium	4<= P < 6	~10.000$
High	6<= P < 8	~50.000$
Critical	8<= P	~100.000$ or more

Note: P = point
Below is the Threat and Vulnerabilities Rate table:

No	Type	Threats & Vulnerabilities	Confidentiality	Integrity	Availability	Total	Rate
1	Portal website	CSRF Attack	3	2	1	6	High
2		IDOR	3	3	1	7	High
3		RCE	3	3	3	9	Critical
4	Master service and Agent services	DDoS Attack	1	1	2	4	Medium
5		Unauthorized access to service	3	3	2	8	Critical

IV.1.3. List of Recommendations to Reduce the Risks

	Threats & Vulnerabilities	Recommendation
Portal website	CSRF Attack	Pentest
	IDOR	Pentest
	RCE	Pentest
Master service and Agent services	DDoS Attack	Using clouding servers or a server which a strong enough
	Unauthorized access to service	Update Authentication Policy

IV.1.4. Costs Associated with Recommendations

	Threats & Vulnerabilities	Recommendation	Cost
Portal website	CSRF Attack	Pentest	1.500$
	IDOR	Pentest	2.000$
	RCE	Pentest	5.000$
Master service and Agent services	DDoS Attack	Using clouding servers	3.000$
	Unauthorized access to service	Update Authentication Policy	0(Manpower)

IV.1.5. CBA (Cost-Benefit Analysis)

	Risk	Recommendations	Loss before control	Loss after control	Cost of control	Benefit
Portal website	CSRF Attack	Pentest	10.000$	1.000$	1.500$	7.500$
	IDOR	Pentest	10.000$	500$	2.000$	7.500$
	RCE	Pentest	15.000$	2.000$	5.000$	8.000$
Master service and Agent services	DDoS Attack	Using clouding servers	6.000$	700$	3.000$	2.300$
	Unauthorized access	Update Authentication	100.000$	15.000$	0(Manpower)	85.000$

	to service	Policy				

IV.2. Assigning Responsibilities

Full Name	Role	Description
Nguyễn Anh Việt	Leader	Assign tasks for members. Research and collect information about risk. Fix vulnerabilities.
Trần Anh Đức	Member	Pentest master and agent services. Research and collect information about risk. Fix vulnerabilities.
Lê Đình Mạnh	Member	Research and collect information about risk. Writing risk management report.
Nguyễn Khắc Hùng	Member	Pentest web portal. Research and collect information about risk. Fix vulnerabilities.
Nguyễn Đức Anh	Member	Research and collect information about risk. Update system.

IV.3. Describing Procedures and Schedules for Accomplishment

- Create a backup for server and database.
- Check update operating system and software.
- Update firewall, policies.
- Create a business continuity plan, recovery plan.

IV.4. Reporting Requirements

IV.4.1. Present Recommendations

Type	Risks	Cause	Criteria	Affect
Master Service	Linux kernel exploit	The kernel has vulnerabilities and	The out of date kernel	The system can’t work.

and Agents Services		the hacker found it		Lost control of the system. Hacker can RCE this machine.
	Misconfiguration	The lack of responsibility of the administrator	Don’t have the configuration checklist standard. The lack of the policy.	The service can’t work correctly. Attacker easy to attack the system. The system can be destroyed.
	DDOS attack	The hardware of the server is still weak. The attacker always tries to launch a dos attack.	The hardware of the system still weak. Don’t use a load balancing system	Performance of service is slow down. Everything is, can’t continue running the hold system, company.
	OS out of date	The attacker always to attack the system.	The lack of responsibility of the administrator	The system can be exploited.
	Natural disasters	Tsunami, earthquake suddenly appeared	Don’t make the business continuity plan. Don’t make the recovery plan.	Everything is shut down, can’t continue running the hold system, company.
	One of service stops working	The lack of responsibility of the administrator. The attacker always try to attack the service	Don’t use the backup service. Don’t have a plan to deal with this.	The system can’t work correctly. Directly affect the system and the user.
Web portal	XSS/ CSS injection	The attacker always try to XSS	Missing validation on front-end Missing expired time session	The data would be stolen. The front-end could be change

			Missing encrypt URL	
	CSRF	The attacker tries to inject the CSRF	Don’t use the correct Get and Post method. Don’t use captcha.	The account could be stolen. The lost of data
	IDOR (INSECURE DIRECT OBJECT REFERENCE)	Allow attackers to bypass authorization and access resources directly by modifying the value of a parameter	The developer does not aware of this bug.	The secret information of service can be revealed by a hacker.

IV.4.2. Document Management Response to Recommendations

Type	Risks	Recommendation	Accept	Transfer	Mitigate
Master Service and Agents Services	Linux kernel exploit	Update and upgrade Linux kernel			√
	Misconfiguration	Configuration checklist standard			√
	DDOS attack	Using a cloud server		√	
	OS out of date	Update OS			√
	Natural disasters	Backup electricity supply Store data on a cloud	√	√	
	One of service stops working	Implement a bot to check live services	√		
Web portal	XSS/ CSS injection	Pentest			√
	CSRF	Pentest			√

	IDOR (INSECURE DIRECT OBJECT REFERENCE)	Pentest			√

IV.4.3. Document and Track Implementation of Accepted Recommendations

Type	Risks	Recommendation	Response	Description
Master Service and Agents Services	Linux kernel exploit	Update and upgrade Linux kernel	Mitigate	Keep Linux kernel up to date.
	Misconfiguration	Configuration checklist standard	Mitigate	Created configuration checklist standard as soon as possible. Validate checklist configuration each time configuration
	DDOS attack	Using a cloud server	Transfer	Apply solution immediately
	OS out of date	Update OS	Mitigate	Keep OS up to date.
	Natural disasters	Backup electricity supply Store data on the cloud	Accept & Transfer	Electricity supply is to be purchased as soon as possible.
	One of service stops working	Implement a bot to check live services	Accept	Implement as soon as possible.
Web portal	XSS/ CSS injection	Pentest	Mitigate	Don’t need to hire the Pentest team we using our manpower.
	CSRF	Pentest	Mitigate	
	IDOR (INSECURE DIRECT OBJECT REFERENCE)	Pentest	Mitigate	

V. Specifications, Development and Implementation Plan

V.1. Logical Model Of Our System.

As we mentioned before, our logical model looks like in Figure V.1.a. This is a kind of microservice architecture follow the principles of service-oriented architecture (SOA) design. We separate each component to a microservice and connect them together.

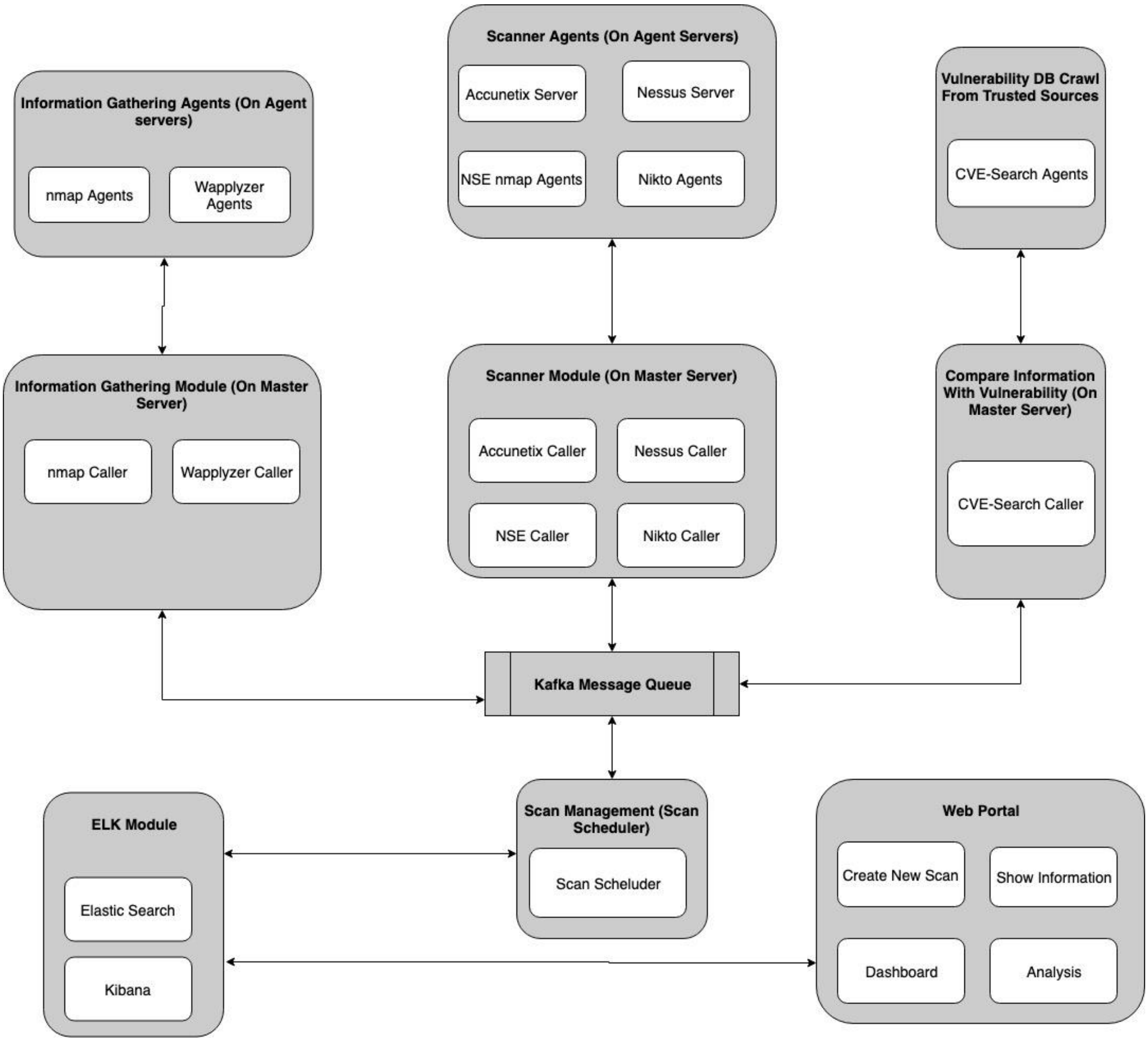


Figure V.1.a. Logical Model Of Our System.

- **Pros of this model:**

  By using the above model, we have so many pros, especially is about speed and expandable because it has all the pros of microservices.

  - Deploy each microservice on a different platform, using different programming languages and developer tools.

  - Microservices use APIs and communication protocols to interact with each other, but they don’t rely on each other otherwise.
  - Our team can develop, maintain, and deploy each microservice independently.
  - Single-responsibility leads to other benefits as well. Each member has their own responsibility for their service.
  - Scale better, as you can scale them separately, whenever it’s necessary.
  - Isolated services have a better failure tolerance. It’s easier to maintain and debug a lightweight microservice than a complex application.
  - Speed up the scan speed, if we want the scan to be faster, we can use more scan server.
  - Faster development cycles (easier deployment and debugging).
- **Cons of this model:**
  - Communication can be hard without using automation and advanced methodologies such as Agile.
  - The simplicity of single-responsibility microservices, lose on the complexity of the network.
  - Communication between microservices can mean poorer performance, as sending messages back and forth comes with a certain overhead. Poorer performance, as microservices need to communicate (network latency, message processing, etc.)
  - We need to manage the whole lifecycle of the microservice, from start to end.
  - Harder to test and monitor because of the complexity of the architecture.
  - Security issues (harder to maintain transaction safety, distributed communication goes wrong more likely, etc.).
  - If a microservice does not respond, we have to detect that issue and fix the microservice as soon as possible.

V.1.1. Kafka Message Queue.

- **Why we need a message queue?**

  Because our system will have multiple states, each state will process a different kind of data and they need to communicate with each other. So we need a thing for them to do the above works and message queue is the best option for us. It does not depend on which programming language we choose because it is a separate system.

- **Why do we choose Kafka?**

  Actually on the world right now, there is a lot of mesythonqueue like RabbitMQ, Kafka, ActiveMQ, … And after our research, we consider between RabbitMQ and Kafka. This table below, we compare between RabbitMQ and Kafka.

Kafka	RabbitMQ
Data Pipeline	Message Base
Do not save the state of consumers, so there is no guarantee that consumers would receive messages that they subscribed.	Save the state of consumers to ensure that all consumers will receive messages that they subscribed.
After all consumer receive, the message is deleted	The message is not deleted until a specific time.
A consumer can choose which message to get.	A consumer will only receive a new message.

Then we decide to use Kafka as our message queue because of it more suitable for our system than RabbitMQ, and it also has a great community use Python with Kafka.

- **How do we use Kafka Message Queue to exchange data between components?**

  First, we will create the following topics and then create **Consumer** and **Producer** for each of them.

Topic	Description
nmapScan	This is the first topic a target will come. This target will be scanned with Nmap.
nmapOutput	After a target being scanned with Nmap done. Its information will come to this topic. Prepare for the next stage.
nseScan	This topic store information for targets that will be scanned

	with NSE
nseOutput	This topic store information for targets which were scanned with NSE. And these targets will come to the next stage.
CVESearchScan	This topic store information for targets that will be scanned with CVESearch
CVESearchOutput	This topic store information for targets which were scanned with CVESearch. And these targets will come to the next stage.
wappalyzerScan	This topic store information for targets that will be scanned with CVESearch
wappalyzerOutput	This topic store information for targets which were scanned with NSE. And these targets will come to the next stage.
nessusScan	This topic store information for targets that will be scanned with CVESearch
nessusOutput	This topic store information for targets which were scanned with Nessus. And these targets will come to the next stage.
acunetixScan	This topic store information for targets that will be scanned with CVESearch
acunetixOutput	This topic store information for targets which were scanned with Acunetix. And these targets will come to the next stage.
niktoScan	This topic store information for targets that will be scanned with CVESearch
niktoOutput	This topic store information for targets which were scanned with Nikto. And these targets will come to the next stage.
elasticSend	Target's information in this topic will be sent and stored in Elasticsearch Database.

V.1.2. ELK Module (Elasticsearch Database).

V.1.2.1. Elastic Search

- **Why do we choose ElasticSearch instead of Relation Database or other NoSQL Database?**

  There is a lot of database type, from the relational database, NoSQL database to in-memory database, each of them have their own pros and cons. In our project, our data will always be in the form of JSON, so the NoSQL will be more relevant. And then we consider between Elasticsearch and MongoDB. But to interact with ElasticSearch is much easier then MongoDB because they provide a REST API so we can interact with them through our **browser** or **CURL**. And ElasticSearch give us a wonderful search mechanism, it is very handy when dealing with analyzing data. Beside these pros, they have an **ELK stack** we can use **Logtash** to collect data and **Kibana** to visualize our data. It is just exactly what we need.

- **How we save data in ElasticSearch?**

  To save data in ElasticSearch, especially using python 3 as in our project, first we need to install the ***elasticsearch***, a Python client for ElasticSearch package through pip:

```
pip3 install elasticsearch
```

Secondly, we create an ElasticSearch object to connect with the ElasticSearch database through a ***connect_elasticsearch()*** method:

```
from elasticsearch import Elasticsearch
from config.config import ElasticConfig


def connect_elasticsearch():
   # Connect to cluster over SSL using auth for best security:
   es_header = [{
       'host': ElasticConfig.HOSTNAME,
       'port': ElasticConfig.PORT,
       'use_ssl': ElasticConfig.USESSL,
       'http_auth': (ElasticConfig.USERNAME, ElasticConfig.PASSWORD)
   }]
   es = Elasticsearch(es_header)
   return es
```

As you can see, the header used to connect to the ElasticSearch database in our project is a JSON data type with 4 key, value pairs. The keys are:

- **Host**: The host of the ElasticSearch database
- **Port**: Port we will use to connect
- **Use_ssl**: Use Secure Socket Layer or not (True or False)
- **Http_auth**: The username and password to the Host above

And all above values are saved in the config file:

```
class ElasticConfig:
    HOSTNAME = 'vulnerabilitymanabem-1160079090.us-west-2.bonsaisearch.net'
    USERNAME = 'rugowuyofm'
    PASSWORD = 'nfsnbt15y2'
    USESSL = True
    PORT = 443
```

After the connection is created successfully, we use the ***index()*** method from the connection to save data to the ElasticSearch database. Two arguments are passed:

- Index: The index (or table) on the ElasticSearch database we want to save to
- Body: The data we want to save, in JSON type

Below is an example of this method when we create a new scan:

```
# Import to Elastic
           es = db_connect.connect_elasticsearch()
           body = {
               'name' : name,
               'description' : description,
               'target' : target_raw,
               'next_run_at' : scan_time_epoch,
               'run_interval' : scan_interval,
               'created_date' : created_time_epoch,
               'scanned_time' : 0,
               'scan_type' : scan_type
               }
           es.index(index=ElasticConfig.SCAN_INDEX, body=body)
```

V.1.2.2. Kibana

To visualize ElasticSearch's data, Kibana helped us a lot. First of all, it's the ***Dev tools***. From the very first day of developing the Web Portal, we used ***Postman*** to test the API(s), getting data from the databases. But it's only convenient in some simple API, examples:

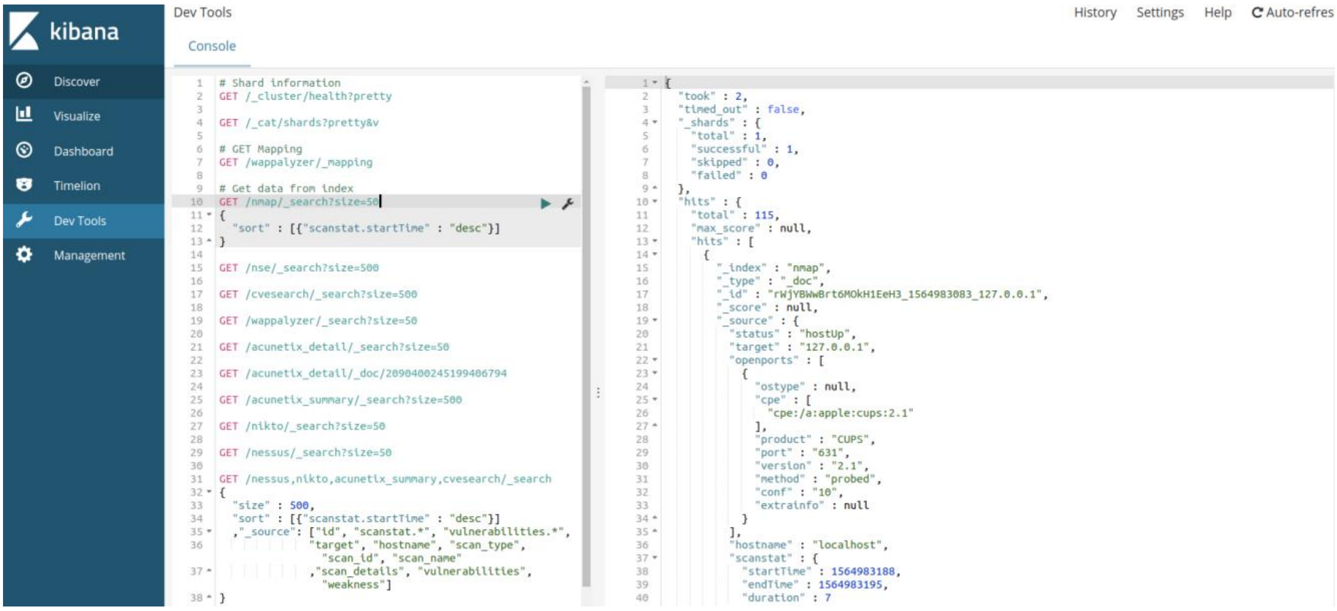
- **nmap/_search?size=50** (get first 50 records of the nmap index)
- **nmap/_doc/rWjYBWwBrt6MOkH1EeH3_1564983083_127.0.0.1** (get only record in nmap index with the specific id)

When you started to need to test the more complicated request, things got more complicated as well, for instance, what if we want to sort the 50 results in the first request above in scan time from latest?

We have to add this in the Body of the request in JSON type:

```
{
      "sort" : [{"scanstat.startTime" : "desc"}]
}
```

And there’re even more when we want to handle multiple request at the same time. It’s the time when we were suggested to use the Kibana **Dev tools** by the leader. Here’s the interface of the **Dev tools:**

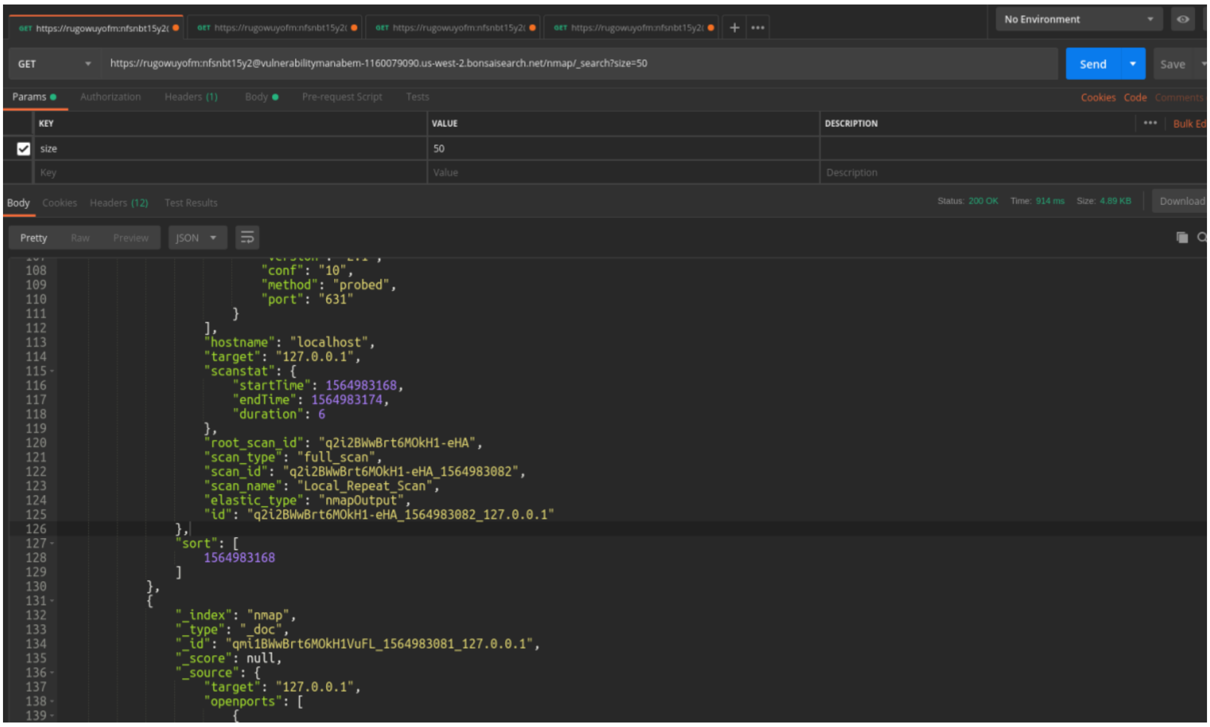


```
# Shard information
GET /_cluster/health?pretty

GET /_cat/shards?pretty&v

# GET Mapping
GET /wappalyzer/_mapping

# Get data from index
GET /nmap/_search?size=50
{
  "sort" : [{"scanstat.startTime" : "desc"}]
}

GET /nse/_search?size=500

GET /cvesearch/_search?size=500

GET /wappalyzer/_search?size=50

GET /acunetix_detail/_search?size=50

GET /acunetix_detail/_doc/2090400245199406794

GET /acunetix_summary/_search?size=500

GET /nikto/_search?size=50

GET /nessus/_search?size=50

GET /nessus,nikto,acunetix_summary,cvesearch/_search
{
  "size" : 500,
  "sort" : [{"scanstat.startTime" : "desc"}]
  ,"_source": ["id", "scanstat.*", "vulnerabilities.*",
              "target", "hostname", "scan_type",
                "scan_id", "scan_name"
              ,"scan_details", "vulnerabilities",
                "weakness"]
}
```

```
{
  "took" : 2,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : 115,
    "max_score" : null,
    "hits" : [
      {
        "_index" : "nmap",
        "_type" : "_doc",
        "_id" : "rWjYBWwBrt6MOkH1EeH3_1564983083_127.0.0.1",
        "_score" : null,
        "_source" : {
          "status" : "hostUp",
          "target" : "127.0.0.1",
          "openports" : [
            {
              "ostype" : null,
              "cpe" : [
                "cpe:/a:apple:cups:2.1"
              ],
              "product" : "CUPS",
              "port" : "631",
              "version" : "2.1",
              "method" : "probed",
              "conf" : "10",
              "extrainfo" : null
            }
          ],
          "hostname" : "localhost",
          "scanstat" : {
            "startTime" : 1564983188,
            "endTime" : 1564983195,
            "duration" : 7
```

Figure V.1.2.2.a. Kibana dev tools interface

As you can see, i’m having about 14~15 written requests to test at the same monitor to test and extremely easy to modify and handle, here are 4 requests in ***Postman:***



GET https://rugowuyofm:nfsnbt15y2@vulnerabilitymanabem-1160079090.us-west-2.bonsaisearch.net/nmap/_search?size=50

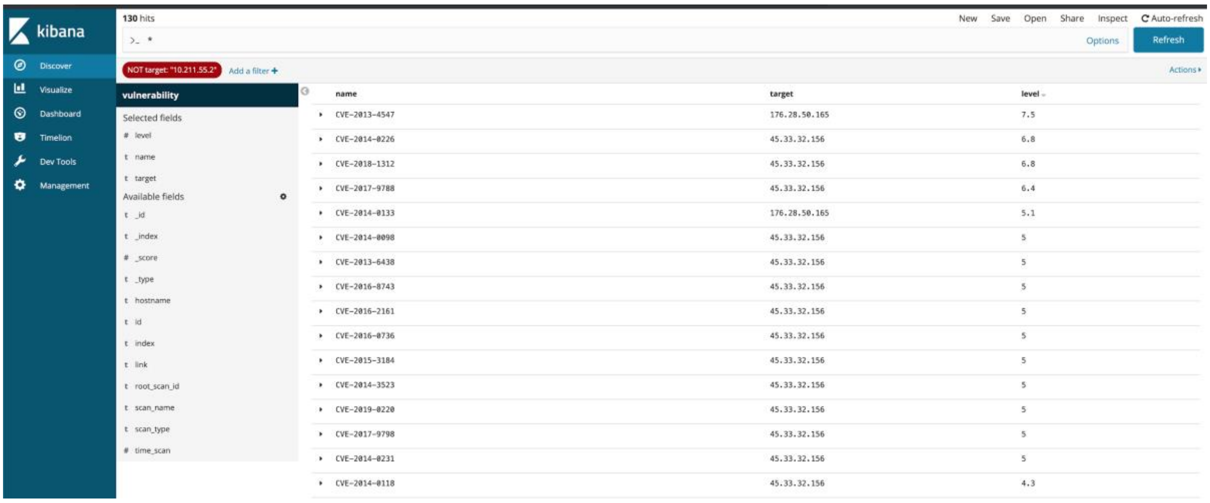
	KEY	VALUE	DESCRIPTION
☑	size	50	
	Key	Value	Description

Status: 200 OK Time: 914 ms Size: 4.89 KB

```
                    "conf": "10",
                    "method": "probed",
                    "port": "631"
                }
            ],
            "hostname": "localhost",
            "target": "127.0.0.1",
            "scanstat": {
                "startTime": 1564983168,
                "endTime": 1564983174,
                "duration": 6
            },
            "root_scan_id": "q2i2BWwBrt6MOkH1-eHA",
            "scan_type": "full_scan",
            "scan_id": "q2i2BWwBrt6MOkH1-eHA_1564983082",
            "scan_name": "Local_Repeat_Scan",
            "elastic_type": "nmapOutput",
            "id": "q2i2BWwBrt6MOkH1-eHA_1564983082_127.0.0.1"
        },
        "sort": [
            1564983168
        ]
    },
    {
        "_index": "nmap",
        "_type": "_doc",
        "_id": "qmi1BWwBrt6MOkH1VuFL_1564983081_127.0.0.1",
        "_score": null,
        "_source": {
            "target": "127.0.0.1",
            "openports": [
                {
```

Figure V.1.2.2.b. Postman interface

We can also use kibana to manage our solution instead of the web interface, because kibana can interact with the database. But it requires more technical skills. We can also search for what we want in kibana



130 hits

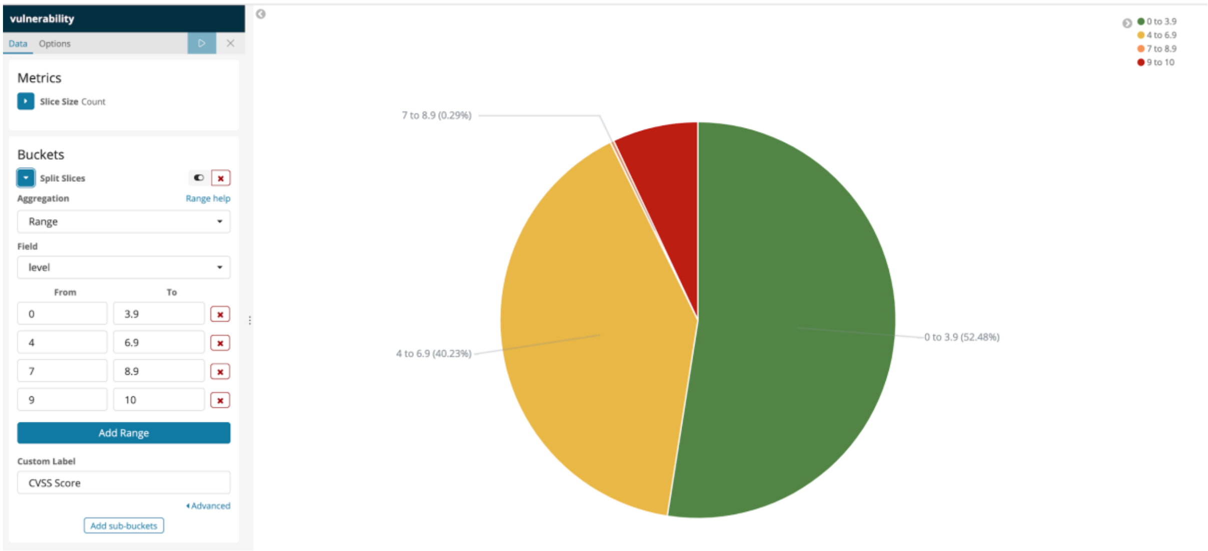
`>_ *`

NOT target: "10.211.55.2"

name	target	level
CVE-2013-4547	176.28.50.165	7.5
CVE-2014-0226	45.33.32.156	6.8
CVE-2018-1312	45.33.32.156	6.8
CVE-2017-9788	45.33.32.156	6.4
CVE-2014-0133	176.28.50.165	5.1
CVE-2014-0098	45.33.32.156	5
CVE-2013-6438	45.33.32.156	5
CVE-2016-8743	45.33.32.156	5
CVE-2016-2161	45.33.32.156	5
CVE-2016-0736	45.33.32.156	5
CVE-2015-3184	45.33.32.156	5
CVE-2014-3523	45.33.32.156	5
CVE-2019-0220	45.33.32.156	5
CVE-2017-9798	45.33.32.156	5
CVE-2014-0231	45.33.32.156	5
CVE-2014-0118	45.33.32.156	4.3

Figure V.1.2.2.c. Using Kibana search for vulnerabilities

We can also visualize the data in Kibana with some simple step like so:



Slice Size Count

Split Slices

Aggregation: Range

Field: level

From	To
0	3.9
4	6.9
7	8.9
9	10

Custom Label: CVSS Score

Figure V.1.2.2.d. Visualize data in kibana

We can also create a dashboard by import some chart to monitor the system like this:

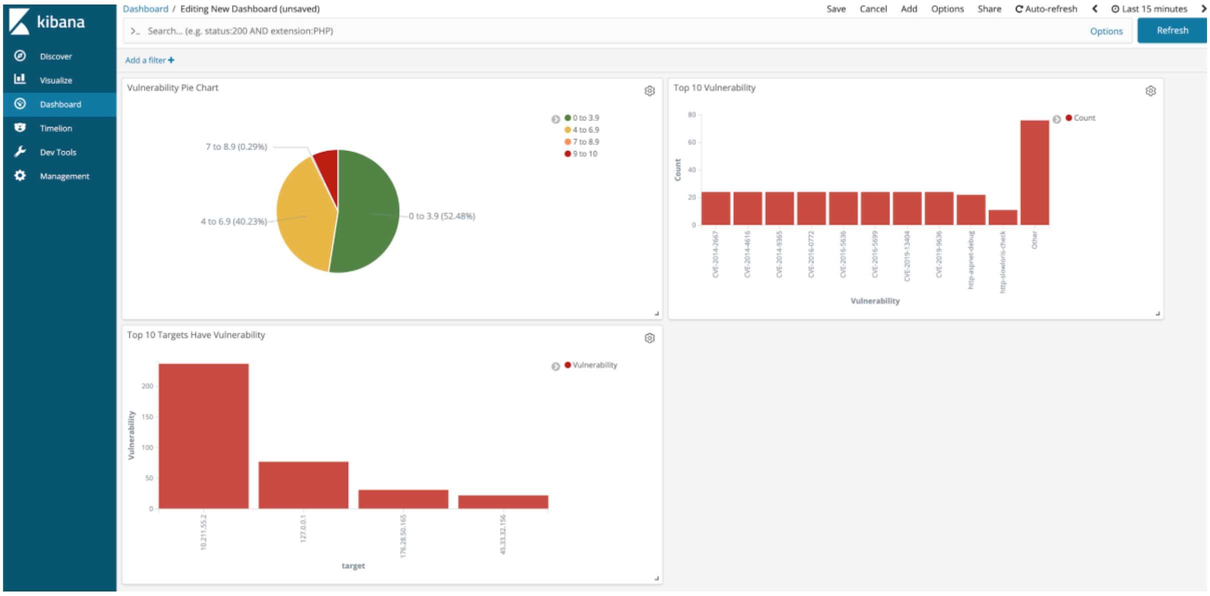


Figure V.1.2.2.e. Dashboard in Kibana

V.1.3. Scan Management (Scan Scheduler).

Implementing the scan scheduler function for our system is quite different from the usual way because our solution is a distributed system, we use micro-service and connect those services. And in our solution, the user creates a new scan on the **web portal component**, but the scan is implemented in the **master component**, and usually, they are run on two different machines. So we come up with another solution.

The JSON below is our **database model** for **Scan** in ElasticSearch.

```
{
   "name" : "new scan",
    "description" : "this scan is a testing scan",
    "target" : "192.168.1.0/24 kenh14.vn",
    "next_run_at" : 2563949690,
    "run_interval" : 1000000000,
    "created_date" : 1563900617,
    "scanned_time" : 1,
    "scan_type" : "non-commercial"
}
```

Notice the `` `next_run_at` `` and `` `run_interval` `` field. Because using these two fields we can make a scan scheduler.

- `Next_run_at` means the timestamp of this scan next run.
- `Run_interval` means the second between two consecutive runs of this scan.

First, we will have a **cron service** on our master server to repeatedly query to Elasticsearch database and get all the scan which have the `Next_run_at` field smaller or equal to the current timestamp.

Secondly, the master server will run this scan and update the `Next_run_at = current_timestamp + Run_interval.`

The two above steps helps us implement our scan scheduler. If a scan just needs to run only once, we will set the `Run_interval` field to infinity or one billion seconds.

V.1.4. Agent Callers (Deployed On Master Server).

To transfer data from the master server to agent servers, we have to use a socket to make a communication channel between them. And for each type of tool, we need to transfer a different type of data. Beside that, we have to manage some cases of the network, for example, what if the connection is failed? what if agents return unexpected value? How to extract the information we need from the output of agents? Now, we will talk more specific for each tool.

V.1.4.1. Nmap Caller.

The code below shows how Nmap on master service makes the connection to Nmap agent service.

```
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.connect((self.HOST, self.PORT))
    jData = json.dumps(self.data) + '\n'
    s.sendall(jData.encode('utf-8'))

    jOutputData = b''
    while True:
        _data = s.recv(1024)
        if not _data:
            break
        jOutputData += _data
```

And if there is some error when scanning, we will retry by resending the target data back to Kafka queue by the code below:

```
def resendDataToNmapScanQueue(self):
    self.data['retryTimes'] = self.data['retryTimes'] + 1
    nmapScanProducer = NmapScanProducers()
    nmapScanProducer.sendDataToQueue(self.data)
```

And this is how we parse the output data of Nmap agent.

```
try: # Nmap agents can send invalid data
    outputData = json.loads(jOutputData.decode('utf-8'))

    scan_stat = dict()
    scan_stat["startTime"] = _startTime
    scan_stat["endTime"] = int(time.time())
    scan_stat["duration"] = (scan_stat["endTime"] - scan_stat["startTime"])

    outputData["scanstat"] = scan_stat

    status = outputData.get('status')
    if status == None:
        raise ValueError("Error with json")
    elif status == 'error':
        error = outputData.get('detail', "Error do not have detail")
        self.logger.error("nmap Agents Error = {} .. Resend {} to nmapScan
queue".format(error, self.data))
        self.resendDataToNmapScanQueue()
    elif status == 'hostUp' or status == 'hostDown':
        outputData['root_scan_id'] = self.data.get('root_scan_id')
        outputData['scan_type'] = self.data.get('scan_type')
        outputData['scan_id'] = self.data.get('scan_id')
        outputData['scan_name'] = self.data.get('scan_name')
        nmapOutputProducer = NmapOutputProducers()
        nmapOutputProducer.sendDataToQueue(outputData)
except:
    self.logger.exception("There are something wrong with return data from nmap Agent for
target= {}: Recived data = {} ... Resend {} to nmapScan queue".format(self.data,
jOutputData, self.data))
    self.resendDataToNmapScanQueue()
```

V.1.4.2. Wappalyzer Caller.

The code below shows how Wappalyzer on master service makes the connection to Wappalyzer agent service. It is the same way with Nmap.

```
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.connect((self.HOST, self.PORT))
    jData = json.dumps(self.data) + '\n'
    s.sendall(jData.encode('utf-8'))

    jOutputData = b''
    while True:
        _data = s.recv(1024)
        if not _data:
            break
        jOutputData += _data
```

And this is how we parse the output data of Wappalyzer agent.

```
try: # To handle agents can send invalid data
    outputData = json.loads(jOutputData.decode('utf-8'))
    outputData["hostname"] = self.data.get("hostname")
    scan_stat = dict()
    scan_stat["startTime"] = _startTime
    scan_stat["endTime"] = int(time.time())
    scan_stat["duration"] = (scan_stat["endTime"] - scan_stat["startTime"])

    outputData["scanstat"] = scan_stat
    outputData['root_scan_id'] = self.data.get('root_scan_id')
    outputData['scan_type'] = self.data.get('scan_type')
    outputData['scan_id'] = self.data.get('scan_id')
    outputData['scan_name'] = self.data.get('scan_name')

    wappalyzerOutputProducer = WappalyzerOutputProducers()
    wappalyzerOutputProducer.sendDataToQueue(outputData)
except:
    self.logger.exception("There are something wrong with return data from nmap
Wappalyzer for target= {}: Recived data = {}".format(self.data, jOutputData))
    self.resendDataToWapplyzerScanQueue()
```

V.1.4.3. CVE-Search Caller.

This module we will talk more in the “**V.1.5**” because we used both socket and web API to communicate between master and agents.

V.1.4.4. NSE Caller.

The code below shows how NSE on master service makes the connection to NSE agent service. It is the same way with Nmap.

```
def resendDataToNseScanQueue(self):
    self.data['retryTimes'] = self.data['retryTimes'] + 1
    nseScanProducers = NseScanProducers()
    nseScanProducers.sendDataToQueue(self.data)
```

And if there is some error when scanning, we will retry by resending the target data back to Kafka queue by the code below:

```
def resendDataToNseScanQueue(self):
    self.data['retryTimes'] = self.data['retryTimes'] + 1
    nseScanProducers = NseScanProducers()
    nseScanProducers.sendDataToQueue(self.data)
```

And this is how we parse the output data of NSE agent.

```
try: # NSE agents can send invalid data
    outputData = json.loads(jOutputData.decode('utf-8'))

    scan_stat = dict()
    scan_stat["startTime"] = _startTime
    scan_stat["endTime"] = int(time.time())
    scan_stat["duration"] = (scan_stat["endTime"] - scan_stat["startTime"])

    outputData["scanstat"] = scan_stat

    status = outputData.get('status')
    if status == None:
        raise ValueError("Error with json")
    elif status == 'error':
        error = outputData.get('detail', "Error do not have detail")
        self.logger.error("NSE Agents Error = {} .. Resend {} to NSEScan
queue".format(error, self.data))
        self.resendDataToNseScanQueue()
    elif status == 'hostUp' or status == 'hostDown':
        outputData['root_scan_id'] = self.data.get('root_scan_id')
        outputData['scan_type'] = self.data.get('scan_type')
        outputData['scan_id'] = self.data.get('scan_id')
        outputData['scan_name'] = self.data.get('scan_name')
        nseOutputProducers = NseOutputProducers()
        nseOutputProducers.sendDataToQueue(outputData)
except:
    self.logger.exception("There are something wrong with return data from NSE Agent for
target= {}: Recived data = {} ... Resend {} to NSEScan queue".format(self.data,
jOutputData, self.data))
    self.resendDataToNseScanQueue()
```

V.1.4.5. Nikto Caller.

The code below shows how Nikto on master service makes the connection to Nikto agent service. It is the same way with Nmap.

```
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.connect((self.HOST, self.PORT))
    jData = json.dumps(self.data) + '\n'
    s.sendall(jData.encode('utf-8'))

    jOutputData = b''
    while True:
        _data = s.recv(1024)
        if not _data:
            break
        jOutputData += _data
```

And this is how we parse the output data of Nikto agent.

```
try: # Nikto agents can send invalid data
    outputData = json.loads(jOutputData.decode('utf-8'))

    if 'msg' in outputData:
        self.logger.info(outputData.get('msg'))
        return

    scan_stat = dict()
    scan_stat["startTime"] = _startTime
    scan_stat["endTime"] = int(time.time())
    scan_stat["duration"] = (scan_stat["endTime"] - scan_stat["startTime"])
    outputData["scanstat"] = scan_stat

    if "port" in outputData:
        del outputData["port"]
    if "host" in outputData:
        del outputData["host"]
    if "ip" in outputData:
        del outputData["ip"]

    outputData["target"] = self.data.get("target")
    outputData["hostname"] = self.data.get("hostname")
    outputData["portScanned"] = self.data.get("portScanned")
    outputData['root_scan_id'] = self.data.get('root_scan_id')
    outputData['scan_type'] = self.data.get('scan_type')
    outputData['scan_id'] = self.data.get('scan_id')
    outputData['scan_name'] = self.data.get('scan_name')

    niktoOutputProducer = NiktoOutputProducers()
    niktoOutputProducer.sendDataToQueue(outputData)

except:
    self.logger.exception("There are something wrong with return data for target= {}:
Recived data = {}".format(self.data, jOutputData))
```

V.1.4.6. Acunetix Caller.

The code below shows how Acunetix on master service makes the connection to Acunetix agent service. It is the same way with Nmap.

```
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.connect((self.HOST, self.PORT))
    jData = json.dumps(self.data) + '\n'
    s.sendall(jData.encode('utf-8'))

    jOutputData = b''
    while True:
        _data = s.recv(1024)
        if not _data:
            break
        jOutputData += _data
```

And this is how we parse the output data of Acunetix agent.

```
try: # Acunetix agents can send invalid data
    outputData = json.loads(jOutputData.decode('utf-8'))

    scan_stat = dict()
    scan_stat["startTime"] = _startTime
    scan_stat["endTime"] = int(time.time())
    scan_stat["duration"] = (scan_stat["endTime"] - scan_stat["startTime"])
    outputData["scanstat"] = scan_stat
    outputData["vuln_stats"] = outputData.get("scan_stat").get("vuln_stats")
    if "scan_stat" in outputData:
        del outputData["scan_stat"]
    outputData["hostname"] = self.data.get("hostname")
    outputData['target'] = self.data.get('target')
    outputData["portScanned"] = self.data.get("portScanned")
    outputData['root_scan_id'] = self.data.get('root_scan_id')
    outputData['scan_type'] = self.data.get('scan_type')
    outputData['scan_id'] = self.data.get('scan_id')
    outputData['scan_name'] = self.data.get('scan_name')

    acunetixOutputProducers = AcunetixOutputProducers()
    acunetixOutputProducers.sendDataToQueue(outputData)

except:
    self.logger.exception("There are something wrong with return data for target= {}:
Recived data = {}".format(self.data, jOutputData))
```

V.1.4.7. Nessus Caller.

The code below shows how Nessus on master service makes the connection to Nessus agent service. It is the same way with Nmap.

```
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.connect((self.HOST, self.PORT))
    jData = json.dumps(self.data) + '\n'
    s.sendall(jData.encode('utf-8'))

    jOutputData = b''
    while True:
        _data = s.recv(1024)
        if not _data:
            break
        jOutputData += _data
```

And this is how we parse the output data of Nessus agent.

```
try: # Nessus agents can send invalid data
    nessusOutputData = json.loads(jOutputData.decode('utf-8'))

    outputData = dict()
    outputData['scan_details'] = self.processNessusOutputData(nessusOutputData)
    outputData['vuln_stats'] = {
        "informational" : 0,
        "low" : 0,
        "medium" : 0,
        "high" : 0
    }
    for scan_detail in outputData['scan_details']:
        if scan_detail['cvss'] < 4:
            outputData['vuln_stats']['low'] += 1
        elif scan_detail['cvss'] < 7:
            outputData['vuln_stats']['medium'] += 1
        else:
            outputData['vuln_stats']['high'] += 1

    scan_stat = dict()
    scan_stat["startTime"] = _startTime
    scan_stat["endTime"] = int(time.time())
    scan_stat["duration"] = (scan_stat["endTime"] - scan_stat["startTime"])
    outputData["scanstat"] = scan_stat

    outputData["hostname"] = self.data.get("hostname")

    outputData['target'] = self.data.get('target')
    outputData["ports"] = self.data.get("ports")
    outputData['root_scan_id'] = self.data.get('root_scan_id')
    outputData['scan_type'] = self.data.get('scan_type')
    outputData['scan_id'] = self.data.get('scan_id')
    outputData['scan_name'] = self.data.get('scan_name')


    nessusOutputProducers = NessusOutputProducers()
    nessusOutputProducers.sendDataToQueue(outputData)

except:
    self.logger.exception("There are something wrong with return data for target= {}:
Recived data = {}".format(self.data, jOutputData))
```

V.1.5. Vulnerability Database Crawl From Trusted Sources (CVE-Search).

CVE search is a tool to import CVE (Common Vulnerabilities and Exposures) and CPE (Common Platform Enumeration) to facilitate the search and processing of CVEs. The main objective of the software is to avoid doing direct and public lookups into the public CVE databases.

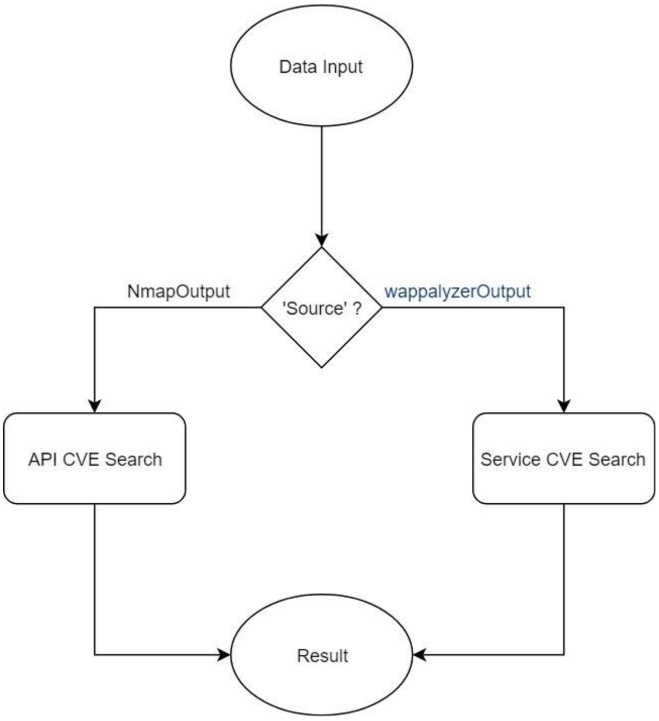


Figure V.1.5. Logical Model Of CVE Search.

With this solution, CVE Search is responsible for receiving data after scanning Nmap or Wappalyzer. From there, we get information about the CPE list and continue processing data. There are 2 cases here:

```
if self.data.get("source") == "nmapOutput":
       weakness = self.scanWithAsyncIO(cpes)
```

- As a result, from `nmapOutput`, we use asynchronous processing to process data faster and more.

```
def scanWithAsyncIO(self, cpes):
    loop = asyncio.new_event_loop()
    asyncio.set_event_loop(loop)
    result_cpes = loop.run_until_complete(self.request_all_vuls(cpes))
    loop.close()

    return result_cpes


async def request_all_vuls(self, cpes):
    async with aiohttp.ClientSession() as session:
        tasks = []
        for cpe in cpes:
            tasks.append(self.sendRequest(session, cpe))
        return await asyncio.gather(*tasks, return_exceptions=True)
```

For each flow of CPE processing, we use the API CVE Search to get the result from http://cve.circl.lu/api/ and then return a result JSON.

Finally, we extract that result to get ***CVE***, ***CWE***, ***CVSS*** to save to scan results.

- The result is from Wapplayer, then call the agent's function:

```
elif self.data.get("source") == "wappalyzerOutput":
    result["weakness"] = self.scanWithSocket(cpes)
```

When the request is sent through the agent, each element in the CPEs list is retrieved from the transmitted data processed:

```
for cpe in cpes:
    cmd = ['python3', '/opt/cve/bin/search.py', '-p', cpe, '-o', 'json']
    process = subprocess.Popen(cmd, stdout=subprocess.PIPE, stderr=subprocess.PIPE)
    jobs.append((cpe, process))
```

For each CPE, the agent calls a CVE Search service returns the same result as the call API CVEsearch. It also returns a JSON result and we dissect that result to get ***CVE***, ***CWE***, ***CVSS***.

Taken together, we combine the results from two cases to create a record of that scan.

V.1.6. Agent Services (Deployed On Agent Servers).

All Agent Services runs on xinetd. We made a bash script on all Agent Services to make it easy to install on Agent Servers. Agent Services receives data from Master Services as a JSON type then it will start the action on the data just received. As soon as the action is done, it will transfer all the data of the action to Master Service also as a JSON type. The following sections describe how each Agent Service work.

V.1.6.1. Nmap Agent Service.

Nmap is the first service to run on over the project. On this agent, it has 3 actions to work. Firstly, it receives JSON data from Master Service and starts to run other function through the code below:

```
def main():
    rawData = input()
    jData = json.loads(rawData)
    scan = Scan(jData)
    scan.run()
```

Secondly, it will run Nmap and export scan data to XML file by option -oX of Nmap. All the actions is by the following code:

```
def gen_cmd(self, target):
   opt = ['nmap', '-sV', '-oX', self.fileName, '--top-ports 1000', '-p 1-10000', '--
open', '-T4', target, '> /dev/null 2>&1']
   cmd = ' '.join(opt)
   return cmd

def gen_fileName(self, target):
   times = time.ctime()
   self.fileName = time.strftime('/var/log/nmap/' + target + '_%Y%m%d-%H%M%S.xml')

def runCmds(self, target):
   cmd = self.gen_cmd(target)
   os.system(cmd)
```

In the end, it will parse data from XML file was exported by nmap to JSON type and print out to Master Service receive it. On this action, we made a library for this. It uses ElementTree from python library to get data from XML type. The code below is a part of code describe how this library works:

```
    openports = []
    dports = dom.findall('host/ports/port')
    for i in dports:
        portid = i.get('portid')
        dservice = i.find('service')
        if dservice != None:
            product = dservice.get('product')
            version = dservice.get('version')
            extrainfo = dservice.get('extrainfo')
            ostype = dservice.get('ostype')
            method = dservice.get('method')
            conf = dservice.get('conf')
            dcpe = dservice.findall('cpe')
            cpes = []
            for j in dcpe:
                cpes.append(j.text)
        else:
            product = None
            version = None
            extrainfo = None
            ostype = None
            method = None
            conf = None
            dcpe = None
            cpes = []

        openports.append({
            'port': portid,
            'product': product,
            'version': version,
            'extrainfo': extrainfo,
            'ostype': ostype,
            'method': method,
            'conf': conf,
            'cpe': cpes
        })

    scan_result['openports'] = openports

    return scan_result
```

V.1.6.2. Wappalyzer Agent Service.

This Agent Service has the same way to receive data from Master Service as Nmap Agent Service.

```
def main():
    targetJson = input()
    targetObject = json.loads(targetJson)
    openports = targetObject.get('openports')
    openports.append('')
    resultObject = dict()
    resultObject['result'] = []
    resultObject['target'] = targetObject.get('target')
```

Next, it will run the wappalyzer application on Server by subprocess. We use subprocess because it supports run multiple processes to speed up the scan.

```
jobs = []
    prefix = ['http://']
    commonPaths = ['', 'admin', 'login', 'admin.php', 'login.php']
    for openport in openports:
        for pre in prefix:
            for commonPath in commonPaths:
                finalTarget = pre + targetObject.get('target') + ':' + openport + '/' +
commonPath
                # finalTarget = targetObject.get('target')
                cmd = ['wappalyzer', finalTarget]
                result = subprocess.Popen(cmd, stdout=subprocess.PIPE,
stderr=subprocess.PIPE)
                jobs.append(result)
                time.sleep(1) # Wait for wappalyzer to send request, if we dont wait, the
result can be incorrect

    solvePort = []
    for job in jobs:
        try:
            output = job.communicate()[0]
            _result = output.decode('utf-8')
            __result = json.loads(_result)
```

```python
            finalTarget = list(__result.get('urls').keys())[0]
            if 'error' not in __result.get('urls').get(finalTarget):
                normalizedData = normalizeOutputData(__result)
                port = str(normalizedData.get('port'))
                path = normalizedData.get('path')
                if port+path not in solvePort:
                    solvePort.append(port+path)
                    resultObject['result'].append(normalizedData)
        except:
            pass
    print (json.dumps(resultObject))
```

Before printing out the result, we normalize the data to make it match the data type of Elasticsearch database

```python
def normalizeOutputData(outputData):
    # reconstruct application to array
    applications = outputData.get('applications')

    finalApplications = []
    for application in applications:
        categories = application.get('categories')
        finalCategories = []
        for category in categories:
            for categoryName in category.values():
                finalCategories.append(categoryName)
        application['categories'] = finalCategories

        application['confidence'] = int(application.get('confidence'))
        finalApplications.append(application)
    outputData['applications'] = finalApplications

    # del "meta" key
    if 'meta' in outputData:
        del outputData['meta']

    # reconstruct "urls"
    urls = outputData.get('urls')
    port = None
    path = '/'
    for url in urls:
        url = url.split('/')
        if len(url[2].split(':')) == 2:
            port = url[2].split(':')[1]
        path = '/'.join(url[3:])
    if 'urls' in outputData:
        del outputData['urls']

    outputData['port'] = port
    outputData['path'] = path
    return outputData
```

V.1.6.3. CVE-Search Agent Service.

The following code shows how CVE-Search Agent Service receives data from Master Service, the same way to Nmap Agent Service.

```
def main():
   jData = input()
   data = json.loads(jData)

   cpes = data.get('cpes', [])
```

After that, it will run CVE-Search application on server as multiple process.

```
jobs = []

   for cpe in cpes:
       cmd = ['python3', '/opt/cve/bin/search.py', '-p', cpe, '-o', 'json']
       process = subprocess.Popen(cmd, stdout=subprocess.PIPE, stderr=subprocess.PIPE)
       jobs.append((cpe, process))
```

Finally, it extracts data from each job and appends to return a result then print it out.

```
for job in jobs:
       try:
           process = job[1]
           jOutputs = process.communicate()[0].decode('utf-8').split('\n')
           cpe = job[0]
           for jOutput in jOutputs:
               try:
                   output = json.loads(jOutput)
                   res = dict()
                   res['cve'] = output.get("id")
                   res['cvss'] = output.get("cvss")
                   res['cwe'] = output.get("cwe")
                   res['cpe'] = cpe
                   outputResult.append(res)
               except Exception as e:
                   pass
       except Exception as e:
           print("Exception =", e)
           traceback.print_exc()
   print(json.dumps(outputResult))
```

V.1.6.4. NSE Agent Service.

This agent almost has the same way to run like Nmap Agent Service. The first step is receiving data from Master Service.

```
def main():
   rawData = input()
   jData = json.loads(rawData)
   scan = Scan(jData)
   scan.run()
```

Next step, generate command and run it. Nmap will export an XML file by option -oX of it.

```
   def gen_cmd(self, target, port):
       opt = ['nmap', '-oX', self.fileName, '-p', port, '--script=vuln,exploit', target,
'> /dev/null 2>&1']
       cmd = ' '.join(opt)
       return cmd

   def gen_fileName(self, target, port):
       times = time.ctime()
       self.fileName = time.strftime('/var/log/nse/' + target + '_' + port + '_%Y%m%d-
%H%M%S.xml')
       #self.fileName = 'testScript.xml'

   def runCmds(self, target, port):
       cmd = self.gen_cmd(target, port)
       os.system(cmd)
```

End of it also has a library to convert an XML file to JSON data. Same with Nmap Agent Service, it uses ElementTree to read data from XML type. A part of code following shows how it works:

```
dhosts = dom.find("runstats/hosts")
if dhosts != None:
    up = dhosts.get('up')
    down = dhosts.get('down')
else:
    return returnError('Could\'t find up and down status')

if up == '0' and down == '0':
    return returnError('Nmap error, 0 up 0 down')
elif down == '1':
    scan_result['status'] = 'hostDown'
    return scan_result

dhost = dom.find("host")
if dhost != None:
    if dhost.find("address") != None:
        scan_result['target'] = dhost.find("address").get('addr')
    else:
        return returnError('Nmap error, not found IP in XML file')
    if dhost.find("hostnames/hostname") != None:
        scan_result['hostname'] = dhost.find("hostnames/hostname").get('name')
    else:
        scan_result['hostname'] = None
else:
    return returnError('Nmap error, nmap did not finished')


nseOutputs = []

dports = dom.findall('host/ports/port')
for i in dports:
    portid = i.get('portid')
    dscript = i.findall('script')
    if dscript != None:
        for j in dscript:
            resultz = dict()
            script = j.get('id')
            output = j.get('output')
            resultz['script'] = script
            resultz['output'] = output
            resultz['port'] = portid
            nseOutputs.append(resultz)

scan_result['nseOutputs'] = nseOutputs

return scan_result
```

V.1.6.5. Nikto Agent Service.

Nikto Agent Service has the same way to get data from Master Service as another Agent Service before.

```
def main():
   rawData = input()
   jData = json.loads(rawData)
   scan = Scan(jData)
   scan.run()
```

Next step, it generates command and executes it like the code below.

```
   def gen_cmd(self, target):
       path = sys.argv[0].replace("niktoAgent.py", "")
       opt = ['perl', path + 'nikto/program/nikto.pl', '-host', target, '-o',
self.fileName, '> /dev/null 2>&1']
       cmd = ' '.join(opt)
       return cmd

   def gen_fileName(self, target):
       times = time.ctime()
       self.fileName = time.strftime('/var/log/nikto/' + '_%Y%m%d-%H%M%S.json')

   def runCmds(self, target):
       cmd = self.gen_cmd(target)
       os.system(cmd)
```

Because the JSON data of Nikto output was not the correct format of JSON, so we have to fix it by code

```
   def fixJson(self, target):
       try:
           with open(self.fileName, 'r') as js:
               result = js.read()
               result = result.replace(',]', ']').replace(',}', '').replace('\n', '')
       except Exception as e:
           return json.dumps({'error': e})
       return result
```

After all, it will print out the final result.

V.1.6.6. Acunetix Agent Service.

We write a library to communicate with Acunetix web application. Firstly, it should initiate some attribute

```
    def __init__(self, username=None, password=None, domain=None, ssl_verify=True, *args,
**kwargs):
        if any([not username, not password, not domain]):
            raise ValueError("username, password and domain are required")

requests.packages.urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)
        super(Acunetix, self).__init__()
        url = ["https://", domain]
        self.verify = ssl_verify
        self.timeout = 2
        self.headers = {
            "Accept": "application / json, text / plain, * / *",
            "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/64.0.3282.186 Safari/537.36",
            "Content-Type": "application/json;charset=UTF-8",
            "Accept-Encoding": "gzip, deflate, br",
            "Accept-Language": "en-US,en;q=0.9",
        }

        self.authenticated = False
        self.max_redirects = 0
        self.username = username
        self.password = hashlib.sha256(password.encode("utf-8")).hexdigest()
        self.url = "".join(url)
        self.check_connectivity()
```

The following is code of some functions used in this Service:

- Login

```
    def login(self):
        try:
            url = self.url + "/api/v1/me/login"
            data = {"email": self.username, "password": self.password, "remember_me":
True}
            resp = self.post(url, json=data)
            if resp.status_code == 204 and "X-Auth" in resp.headers:
                self.authenticated = True
                self.headers.update({"X-Auth": resp.headers['X-Auth']})
                return self.me
            else:
                raise Exception("Failed to authenticate")
        except Exception as e:
            raise e
```

- Create a target, it will return a JSON data that contain target ID

```
    def create_target(self, address, description):
        try:
            url = self.url + "/api/v1/targets"
            data = {"address": str(address), "description": str(description)}
            resp = self.post(url, json=data)
            if resp.status_code == 201:
                return resp.json()
            return {'error': 'Failed to create target'}
        except Exception as e:
            raise e
```

- Delete a target by target ID

```
def delete_target(self, target_id):
    try:
        url = self.url + "/api/v1/targets/{}".format(target_id)
        if self.delete(url).status_code == 204:
            return True
        return {'error': 'Failed to delete target'}
    except Exception as e:
        raise e
```

- Run scan of a target by target ID, it will return scan ID.

```
def create_scan(self, target_id, scan_type, report_templated_id=None):
    try:
        url = self.url + "/api/v1/scans"
        data = {
            "target_id": target_id,
            "profile_id": scan_type,
            "schedule": {
                "disable": False,
                "start_date": None,
                "time_sensitive": False
            }
        }
        if report_templated_id:
            data.update({"report_template_id": report_templated_id})

        resp = self.post(url, json=data)
        scan_id = resp.headers['Location'].split('/')[-1]
        return scan_id
    except Exception as e:
        raise e
```

- Get scan status by scan ID, it will return a JSON data.

```
def scan_status(self, scan_id, extra_stats=False):
    try:
        url = self.url + "/api/v1/scans/{}".format(str(scan_id))
        resp = self.get(url).json()

        if 'code' in resp and resp['code'] == 404:  # if scan doesn't exists on server
            return None
```

```
           progress = resp['current_session']['progress']
           status = resp['current_session']['status']
           vuln_stats = None
           if status != "scheduled":
               vuln_stats = resp['current_session']['severity_counts']
               vuln_stats["informational"] = vuln_stats.pop("info")

           data = {'progress': progress, 'status': status, 'vuln_stats': vuln_stats,
                   'session_id': resp['current_session']['scan_session_id']}

           if extra_stats:
               url = url +
'/results/{}/statistics'.format(resp['current_session']['scan_session_id'])
               resp = self.get(url).json()
               aborted = resp['scanning_app']['wvs']['abort_requested']
               start_date = resp['scanning_app']['wvs']['start_date']
               end_data = resp['scanning_app']['wvs']['end_date']
               data.update({'aborted': aborted, 'start_date': start_date, 'end_date':
end_data})

           return data
       except Exception as e:
           raise e
```

- Get list vulnerabilities of a scan by scan ID, it will return a JSON data.

```
   def get_scan_vulnerabilities(self, scan_id):
       try:
           url = self.url + "/api/v1/scans/{}".format(str(scan_id))
           resp = self.get(url).json()
           url = url +
'/results/{}/vulnerabilities'.format(resp['current_session']['scan_session_id'])
           resp = self.get(url).json()['vulnerabilities']
           return resp
       except Exception as e:
           raise e
```

- Get vulnerability by id

```
   def get_vulnerability_by_id(self, scan_id, vulnerability_id, scan_session_id=None):
       try:
           if not scan_session_id:
               scan_session_id = self.scan_status(scan_id)['session_id']
           url = self.url +
"/api/v1/scans/{}/results/{}/vulnerabilities/{}".format(scan_id, scan_session_id,

vulnerability_id)
           resp = self.get(url).json()
           return resp
       except Exception as e:
            raise e
```

Because Acunetix only allow 1 user login at a time, so if Master Service send 2 scans at a time, the second scan will can't run because it will fail on login. Then we have to write a function to save the entire Acunetix library object to a file and reuse it after.

```
def save_object(obj):
    try:
        with open('./obj_acu.pkl', 'wb') as obj_file:
            pickle.dump(obj, obj_file)
    except:
        return None

def load_object():
    try:
        with open('./obj_acu.pkl', 'rb') as obj_file:
            obj = pickle.load(obj_file)
            return (obj)
    except:
        return None
```

Now is the main function of this Agent Service. It still has the same way to receive data from Master Service as other Agent Service before.

```
def main():
    rawData = input()
    jData = json.loads(rawData)
    scan_type = jData.get('acunetix_scan_type')
    address = jData.get('target_url')
```

Nextly, it will load some data from config file to create an object of Acunetix library. Then it also loads the object from the file if it is not the first run on this server. After this, it will check if the object loaded from file is still can communicate with Acunetix website. If not it will do function login and save the object to file.

```
def run(address, scan_type):
    cf = config.Config()
    acunetix = Acunetix(username = cf.username, password = cf.password, domain = 
cf.domain, ssl_verify = cf.ssl_verify)
    try:
        _acunetix = load_object()
        if _acunetix != None:
            _acunetix.url = 'https://' + cf.domain
            if _acunetix.check_logging() == False:
                acunetix.login()
                save_object(acunetix)
            else:
                acunetix = _acunetix
        else:
            acunetix.login()
            save_object(acunetix)
```

Next step is create target and run the scan:

```
        target = acunetix.create_target(address=address, description='Creating scan for '
+ address)
        target_id = target.get('target_id')
        scan_id = acunetix.create_scan(target_id=target_id, scan_type=scan_type)
```

After run scan, it will check every 10 seconds if the scan is done:

```
        while True:
            time.sleep(10)
            scan_stat = acunetix.scan_status(scan_id=scan_id, extra_stats=False)
            if scan_stat.get('status') == 'completed':
                break
```

Finally, when the scan is done, it will get all vulnerabilities of this scan and print it out. Before return the result to Master Service, it will remove the target from Acunetix web application.

```
        scan_stat = acunetix.scan_status(scan_id=scan_id, extra_stats=True)
        objects['scan_stat'] = scan_stat

        scan_vuls = acunetix.get_scan_vulnerabilities(scan_id=scan_id)
        for scan_vul in scan_vuls:
            vul_id = scan_vul.get('vuln_id')
            vuls_details = acunetix.get_vulnerability_by_id(scan_id=scan_id,
vulnerability_id=vul_id)
            objects['scan_details'].append(vuls_details)

        acunetix.delete_target(target_id)
        objects['target'] = address
        print (json.dumps(objects))
```

V.1.6.7. Nessus Agent Service.

The same to Acunetix Agent Service, we also write a library to communicate with Nessus web application.

Here is function that initiate the Nessus library object:

```
    def __init__(self, username=None, password=None, accessKey=None, secretKey=None,
domain=None, ssl_verify=True, X_API_Token=None, *args, **kwargs):
        if not domain:
            raise ValueError("domain are required")
        if not ((username and password) or (accessKey and secretKey)):
            raise ValueError("username, password or APIKeys are required")

requests.packages.urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)
        super(Nessus, self).__init__()
        url = ["https://", domain]

        self.verify = ssl_verify
        self.timeout = 5
        self.headers = {
            "Accept": "application / json, text / plain, * / *",
            "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/64.0.3282.186 Safari/537.36",
            "Content-Type": "application/json;charset=UTF-8",
            "Accept-Encoding": "gzip, deflate, br",
            "Accept-Language": "en-US,en;q=0.9",
        }
        self.authenticated = False
        self.max_redirects = 0
        self.headers.update({"X-API-Token": X_API_Token})
        self.url = "".join(url)
        if username and password:
            self.username = username
            self.password = password
        elif accessKey and secretKey:
            self.accessKey = accessKey
            self.secretKey = secretKey
            apiKeys = ["accessKey=" + accessKey, "secretKey=" + secretKey]
            self.headers.update({"X-ApiKeys": ';'.join(apiKeys)})
            if not self.check_logging():
                raise Exception('X-ApiKeys is not working')
        self.check_connectivity()
```

There are some functions used in this Agent Service:

- Login

```
    def login(self):
        while True:
            try:
```

```
            url = self.url + "/session"
            data = {"username": self.username, "password": self.password}
            resp = self.post(url, json=data)
            if resp.status_code == 200:
                token = resp.json()['token']
                self.authenticated = True
                self.headers.update({"X-Cookie": "token=" + token})
                return True
            else:
                raise Exception('Failed to authenticate')
        except requests.exceptions.ConnectionError:
            time.sleep(3)
            pass
        except Exception as e:
            raise e
```

- Logout

```
def logout(self):
    while True:
        try:
            url = self.url + "/session"
            resp = self.delete(url)
            if resp.status_code == 200:
                return True
            else:
                raise Exception('Failed to logout')
        except requests.exceptions.ConnectionError:
            time.sleep(3)
            pass
        except Exception as e:
            raise e
```

- Create a scan, it will return a JSON data that contain scan ID

```
def create_scan(self, target, uuid, ports):
    while True:
        try:
            url = self.url + "/scans"
            setting = {
                "name": target,
                "text_targets": target,
                "launch_now": True,
                "portscan_range": ports,
                "ping_the_remote_host": "no"
            }
            data = {"uuid":uuid, "settings": setting}
            resp = self.post(url, json=data)
            if resp.status_code == 200:
                return resp.json()
            raise Exception('Failed to create scan')
        except requests.exceptions.ConnectionError:
            time.sleep(3)
            pass
        except Exception as e:
            raise e
```

- Delete scan by scan ID

```
def delete_scan(self, scan_id):
    while True:
        try:
            url = self.url + "/scans/" + str(scan_id)
            resp = self.delete(url)
            if resp.status_code == 200:
                return
            raise Exception('Failed to delete scan')
        except requests.exceptions.ConnectionError:
            time.sleep(3)
            pass
        except Exception as e:
            raise e
```

- Get scan details by scan ID, it will return a JSON data that contain plugin name

```
def details_scan(self, scan_id):
    while True:
        try:
            url = self.url + "/scans/" + str(scan_id)
            resp = self.get(url)
            if resp.status_code == 200:
                return resp.json()
            raise Exception('Failed to get scan details')
        except requests.exceptions.ConnectionError:
            time.sleep(3)
            pass
        except Exception as e:
            raise e
```

- Get a list of plugin details by scan ID and plugin name, it will return a JSON data

```
def plugins_details(self, scan_id, plugin):
    while True:
        try:
            url = self.url + "/scans/" + str(scan_id) + "/plugins/" + str(plugin)
            resp = self.get(url)
            if resp.status_code == 200:
                return resp.json()
            raise Exception('Failed to get plugin details')
        except requests.exceptions.ConnectionError:
            time.sleep(3)
            pass
        except Exception as e:
            raise e
```

The main function of Nessus Agent Service also has a function to receive data from Master Service as another.

```
def main():
    rawData = input()
    jData = json.loads(rawData)
    scan_type = jData.get('nessus_scan_type')
    address = jData.get('target')
    ports = jData.get('ports')
    _ports = ','.join(ports)
```

It will load some config from file config to initiate an object of Nessus library. It has 2 ways to authorize, by username and password of Nessus user or by Access key and Secret key.

```
def run(target, scan_type, ports):
    cf = config.Config()
    try:
        if cf.username and cf.password:
            ns = Nessus(username = cf.username, password = cf.password, domain =
cf.domain, ssl_verify = cf.ssl_verify, X_API_Token=cf.X_API_Token)
            ns.login()
        elif cf.accessKey and cf.secretKey:
            ns = Nessus(accessKey = cf.accessKey, secretKey = cf.secretKey, domain =
cf.domain, ssl_verify = cf.ssl_verify, X_API_Token=cf.X_API_Token)
```

Nextly, it will create a scan and get scan ID:

```
        scan = ns.create_scan(target=target, uuid=scan_type, ports=ports)
        scan_id = scan['scan']['id']
```

After created scan, it will wait until the scan is done:

```
        while True:
            details_scan = ns.details_scan(scan_id)
            if len(details_scan['info']) > 0 and details_scan['info']['status'] ==
'completed':
        break
    time.sleep(5)
```

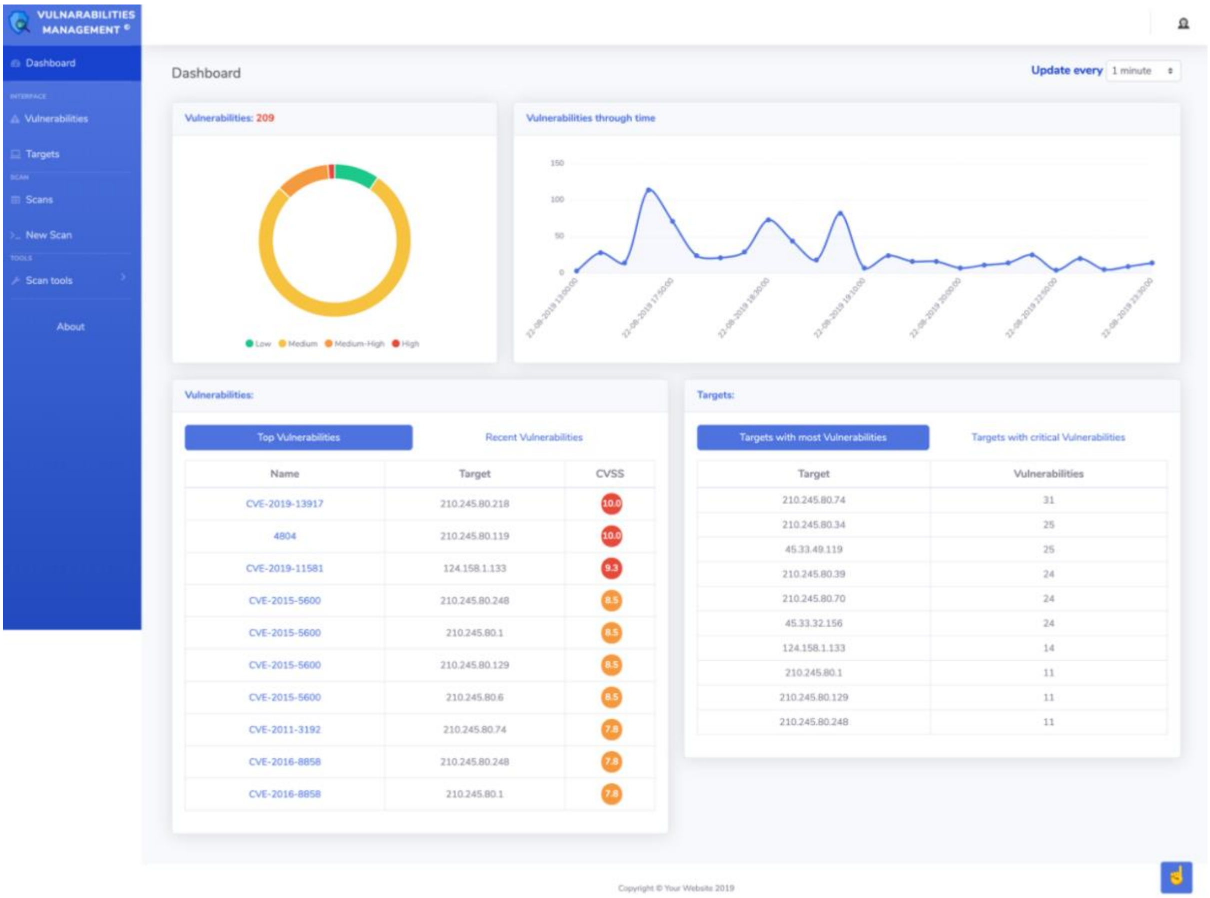
In the end, it will list all plugin appear on that scan and append it to a dict object. After that, it will remove the scan and print out the final result.

```
objects = dict()
objects['scan_details'] = list()
objects['target'] = target
for scan_vul in details_scan['vulnerabilities']:
    vuls_details = ns.plugins_details(scan_id, scan_vul['plugin_id'])
    objects['scan_details'].append(vuls_details)
    break
ns.delete_scan(scan_id)
print (json.dumps(objects))
```

V.1.7. Web Portal.

Figure V.1.7.2.a. Interface Of Create New Scan Form

Dashboard is the overview of the vulnerabilities and all the scanned targets in ***Vulnerability Management***. It brings the IT manager a statistical view including doughnut chart of critical vulnerabilities from low to high (percentage), amount of vulnerabilities through time (time range selected by user), ***top critical vulnerabilities*** and ***top recent vulnerabilities discovered***. About targets, it brings up with ***Targets with most Vulnerabilities (amount)*** and ***Targets with critical Vulnerabilities (critical).*** Additionally, the IT manager can set it to refresh periodically to update it automatically, the time to choose is between, 1, 5, 10 and 30 minutes.



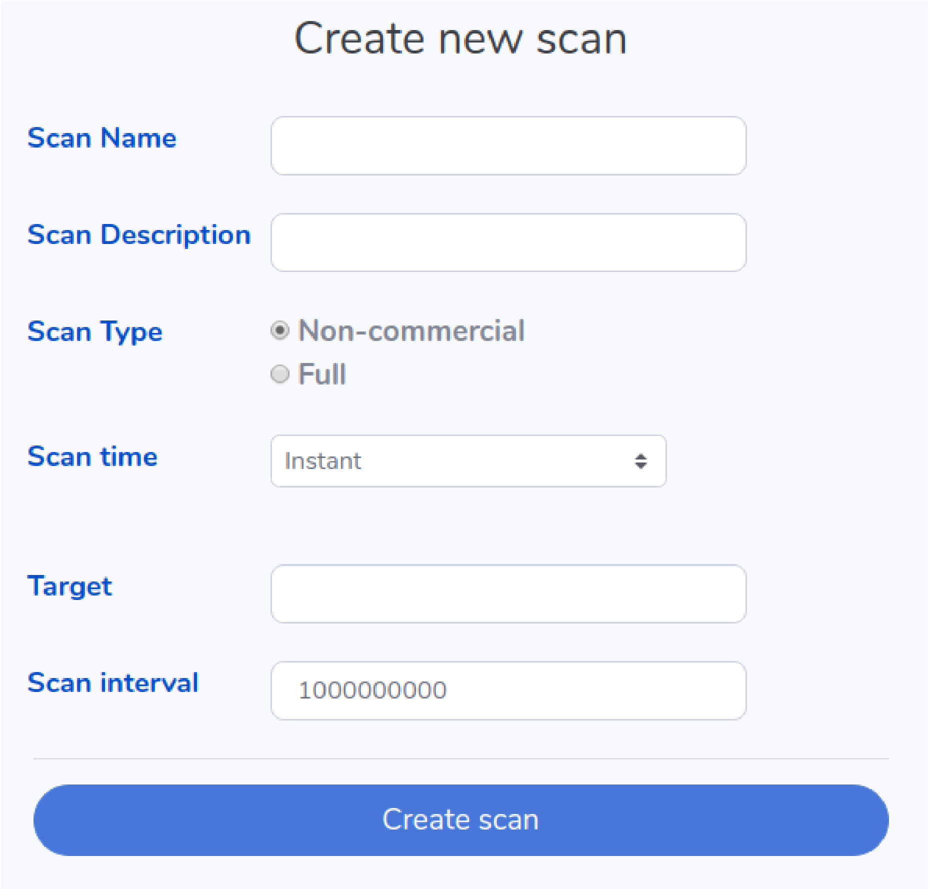
Name	Target	CVSS
CVE-2019-13917	210.245.80.218	10.0
4804	210.245.80.119	10.0
CVE-2019-11581	124.158.1.133	9.3
CVE-2015-5600	210.245.80.248	8.5
CVE-2015-5600	210.245.80.1	8.5
CVE-2015-5600	210.245.80.129	8.5
CVE-2015-5600	210.245.80.6	8.5
CVE-2011-3192	210.245.80.74	7.8
CVE-2016-8858	210.245.80.248	7.8
CVE-2016-8858	210.245.80.1	7.8

Target	Vulnerabilities
210.245.80.74	31
210.245.80.34	25
45.33.49.119	25
210.245.80.39	24
210.245.80.70	24
45.33.32.156	24
124.158.1.133	14
210.245.80.1	11
210.245.80.129	11
210.245.80.248	11

Figure V.1.7.1. Dashboard of Vulnerabilities Management

V.1.7.2. Create New Scan.

As a function of the Web portal, we support users to create new scan through a form. Besides the other field has its own default value, the *Scan Name* and the *Target* field are required to input manually. The form we support can be referred as below.



Create new scan

Scan Name

Scan Description

Scan Type ● Non-commercial ○ Full

Scan time: Instant

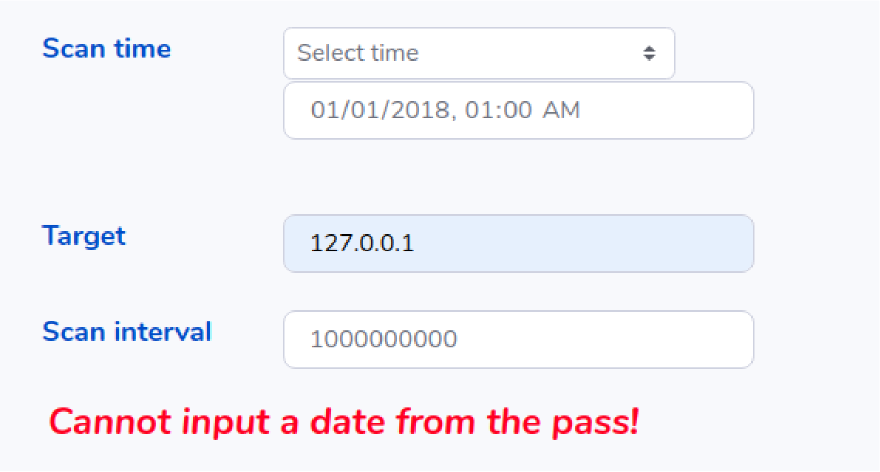
Target

Scan interval: 1000000000

Figure V.1.7.2.a. Interface Of Create New Scan Form

The ***Scan Name*** and ***Scan Description*** fields help us define and describe our scan to be created. ***Scan Type*** field currently allows selecting between non-commercial scan tools or full scan tools including paid ones (Acunetix and Nessus at the moment).

Two current options in the *Scan time* is ***Instant*** and ***Select time***. If we select ***Instant***, the valid scan we create will be pushed into the queue and will be active as soon as the previous scans are done. Otherwise, an input field with the type of "datetime-local" will show up and allow us to pick the date and time for the scan to start. Date from the past is not allowed.



Scan time

Select time

01/01/2018, 01:00 AM

Target

127.0.0.1

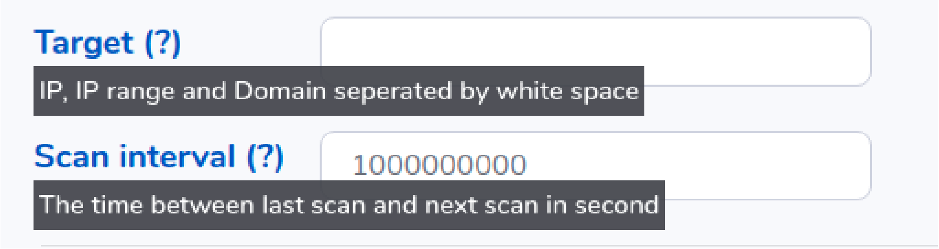
Scan interval

1000000000

Cannot input a date from the pass!

Figure V.1.7.2.b. Invalid date in create new scan form

With the ***Target*** field, there are 3 types of target allowed: ***IP address***, ***IP range*** and ***Domain*** (ex: 35.240.138.138, 35.240.138.0/24, google.com).



Target (?)

IP, IP range and Domain seperated by white space

Scan interval (?)

1000000000

The time between last scan and next scan in second

Figure V.1.7.2.c. Target and Scan interval tooltip

User can input multiple targets at the same time separated by an empty space “ ”, the string *target* then will be split into substrings by space and test one by one by generated RegEx, if one of the substring failed to match all 3 RegEx(s) then the whole *target* will considered invalid. The code of this process is shown below:

```
const REGEX_IP = /^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-
9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])$/g;
const REGEX_IP_RANGE = /^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-
9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])(\/(3[0-2]|[1-2][0-9]|[0-9]))$/g;
const REGEX_DOMAIN = /^(?!:\/\/)([a-zA-Z0-9-_]+\.)*[a-zA-Z0-9][a-zA-Z0-9-_]+\.[a-zA-
Z]{2,11}?$/igm;
```

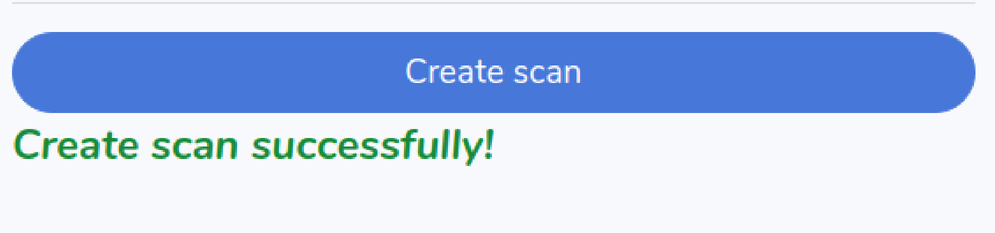
```
for (let i = 0; i < targets.length; i++) {
   if (targets[i].match(REGEX_IP) !== null ||
       targets[i].match(REGEX_IP_RANGE) !== null ||
    targets[i].match(REGEX_DOMAIN) !== null) continue;

else {
       $('#noti').text("Please enter valid IP, IP range or domain separated by space!");
       $('#noti').show();
       return false;
    }
}
return true;
```

The last field is ***Scan Interval***, determine how many seconds until the next scan from the last time scan in second (see *Figure V.1.7.c.*), default value is **1000000000s (~ 31.7 years).**

After a valid form is submitted, all the data will be sent to the backend to the address /newscan/create with HTTP method POST. In here the target will be tested with the RegEx(s) once again and add to the ElasticSearch database with the ***index()*** method.

```
body = {
    'name' : name,
    'description' : description,
    'target' : target_raw,
    'next_run_at' : scan_time_epoch,
    'run_interval' : scan_interval,
    'created_date' : created_time_epoch,
    'scanned_time' : 0,
    'scan_type' : scan_type
}
es.index(index=ElasticConfig.SCAN_INDEX, body=body)
create_result = 1
```

Finally, create result will be sent back to the browser to notify the user.



Create scan successfully!

Figure V.1.7.2.d. Interface When Scan Have Been Created Successfully

V.1.7.3. Show Scan Informations.

V.1.7.3.a. Tools scan result

Throughout the document, there are a variety of tools were mentioned that used in our scan progress to determine vulnerabilities on the system. And before combining all of them to bring the user an overview, we support viewing the specific result of those tools as well. In the Web Portal, they are split into 2 sections ***Non-commercial*** and ***Commercial*** as below.

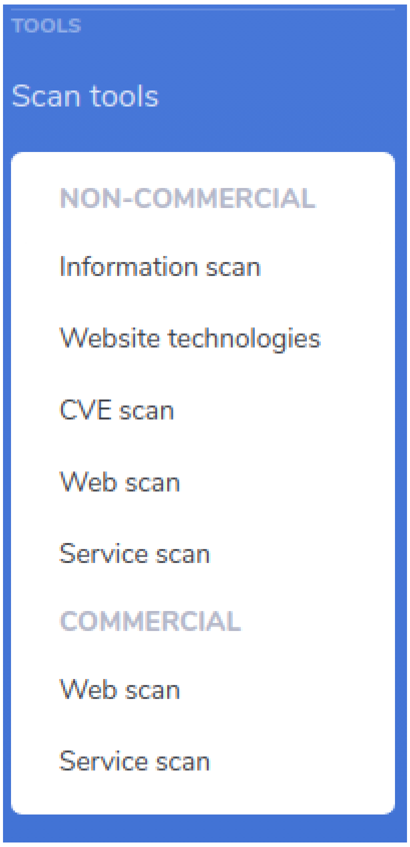


Figure V.1.7.3.a.1. Scan tools

The tools in the Non-commercial section are:

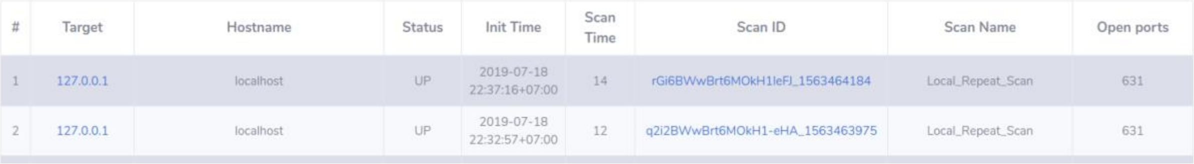
- *Nmap* (Port information)
- *Wappalyzer* (Website technology)
- *CVE-Search* (Common Vulnerabilities and Exposures)
- *Nikto* (Web server)
- *NSE* (Nmap Scripting Engine)

And the tools in the Commercial section:

- *Acunetix* (Advance Web server scan)
- *Nessus* (Service scan)

Before going deep into specific information of all the tools above, we will start with the common information fields first. These fields are:

1. **Target**: The IP address of the viewing target
2. **Hostname**: The hostname corresponding to the target above
3. **Scan ID**: The ID of the scan the target was found
4. **Scan Name**: Name of the scan above
5. **Init Time**: Time the target was found, in the UTC-7 time zone
6. **Scan Time**: The scan duration of the scan mentioning, in second

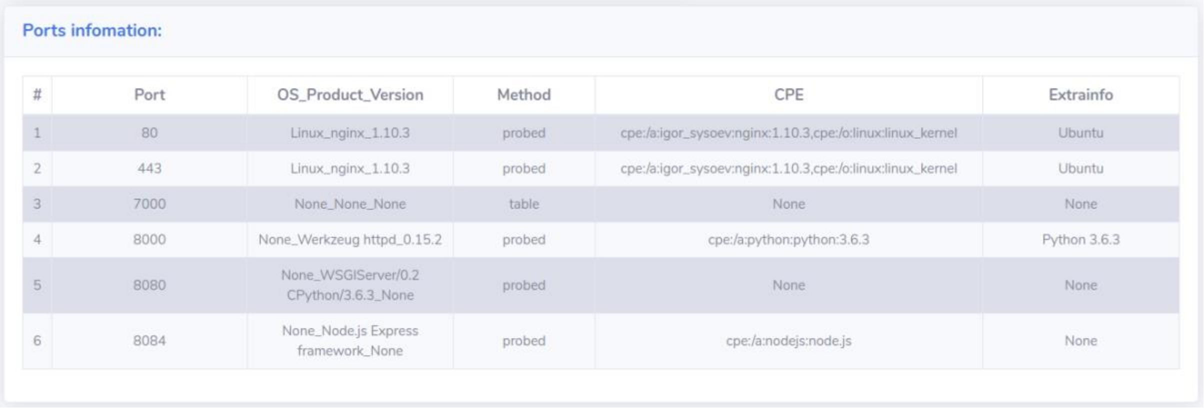


#	Target	Hostname	Status	Init Time	Scan Time	Scan ID	Scan Name	Open ports
1	127.0.0.1	localhost	UP	2019-07-18 22:37:16+07:00	14	rGi6BWwBrt6MOkH1leFJ_1563464184	Local_Repeat_Scan	631
2	127.0.0.1	localhost	UP	2019-07-18 22:32:57+07:00	12	q2i2BWwBrt6MOkH1-eHA_1563463975	Local_Repeat_Scan	631

Figure V.1.7.3.a.2. Scan tools common fields

Next is the specific fields of each tool:

- ***Nmap***
  - **Status**: Determine that the host is UP or DOWN
  - **Port**: The port number on that host
  - **OS_Product_Version**: The combination string of the OS, Product and Version found on the mentioned port (if any), separated by the underscore character.
  - **Method**: The method Nmap used to found the target
  - **CPE:** Common Platform Enumeration found on the port (if any)
  - **Extrainfo:** Extra information that notable (if any)

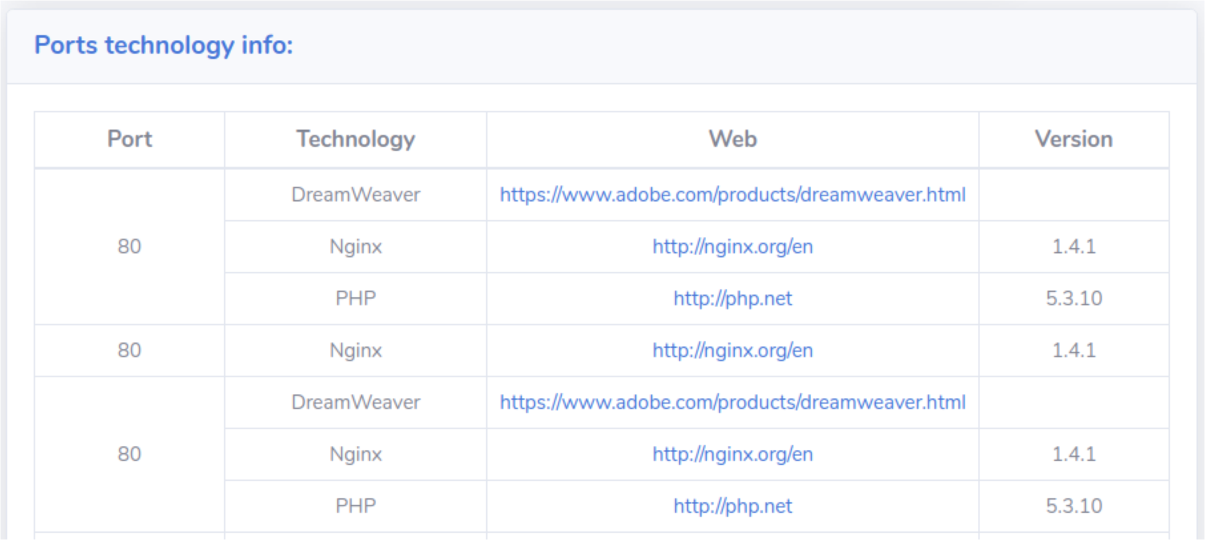


Ports infomation:

#	Port	OS_Product_Version	Method	CPE	Extrainfo
1	80	Linux_nginx_1.10.3	probed	cpe:/a:igor_sysoev:nginx:1.10.3,cpe:/o:linux:linux_kernel	Ubuntu
2	443	Linux_nginx_1.10.3	probed	cpe:/a:igor_sysoev:nginx:1.10.3,cpe:/o:linux:linux_kernel	Ubuntu
3	7000	None_None_None	table	None	None
4	8000	None_Werkzeug httpd_0.15.2	probed	cpe:/a:python:python:3.6.3	Python 3.6.3
5	8080	None_WSGIServer/0.2 CPython/3.6.3_None	probed	None	None
6	8084	None_Node.js Express framework_None	probed	cpe:/a:nodejs:node.js	None

Figure V.1.7.3.a.3. Nmap ports information

- ***Wappalyzer***
  - **Port**: Number of the viewing port
  - **Technology**: Technology using on that port
  - **Web:** The website with corresponding technology
  - **Version**: Version of the technology using

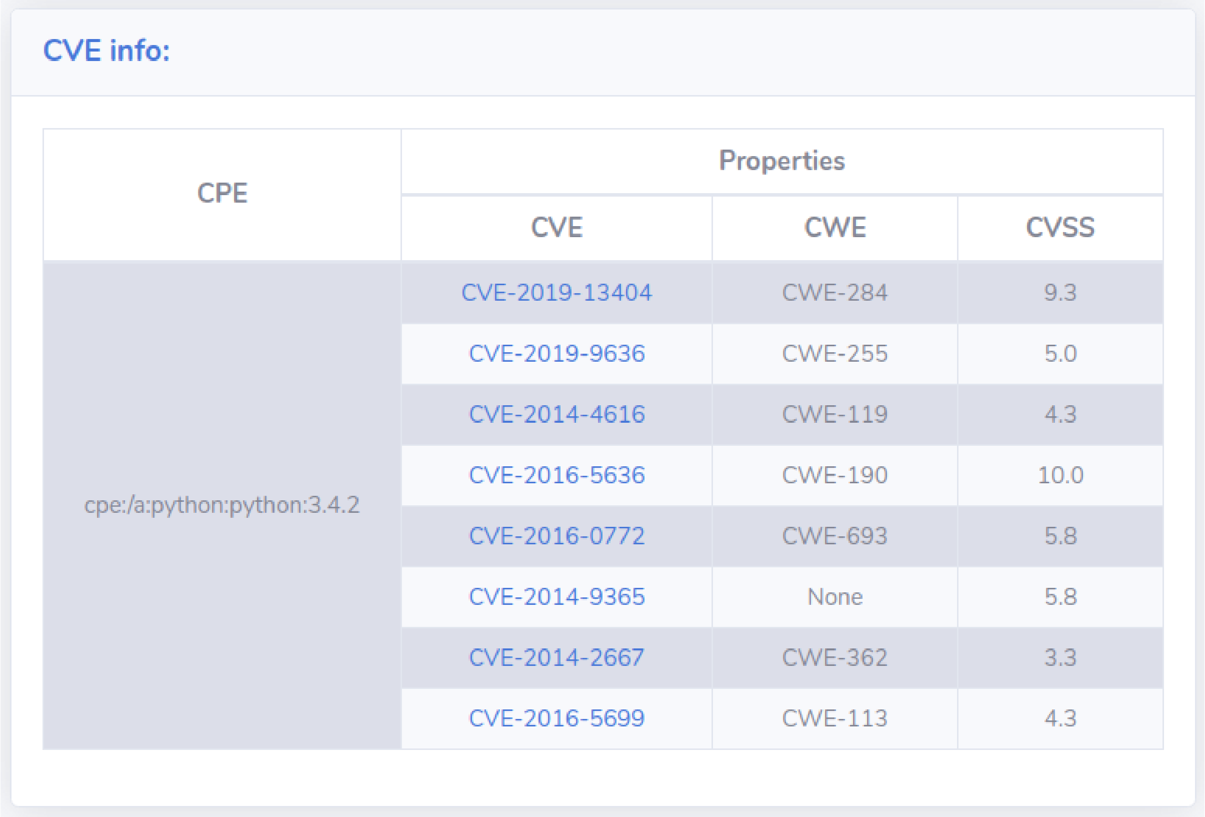


Ports technology info:

Port	Technology	Web	Version
80	DreamWeaver	https://www.adobe.com/products/dreamweaver.html	
	Nginx	http://nginx.org/en	1.4.1
	PHP	http://php.net	5.3.10
80	Nginx	http://nginx.org/en	1.4.1
80	DreamWeaver	https://www.adobe.com/products/dreamweaver.html	
	Nginx	http://nginx.org/en	1.4.1
	PHP	http://php.net	5.3.10

Figure V.1.7.3.a.4. Wappalyzer ports technology info

- ***CVE-Search***
  - **CPE**: The viewing CPE (Common Platform Enumeration)
  - **CVE**: The CVE(s) found on the viewing CPE
  - **CWE**: Common Weakness Enumeration found
  - **CVSS**: The severity of the CVE (from 0-10)



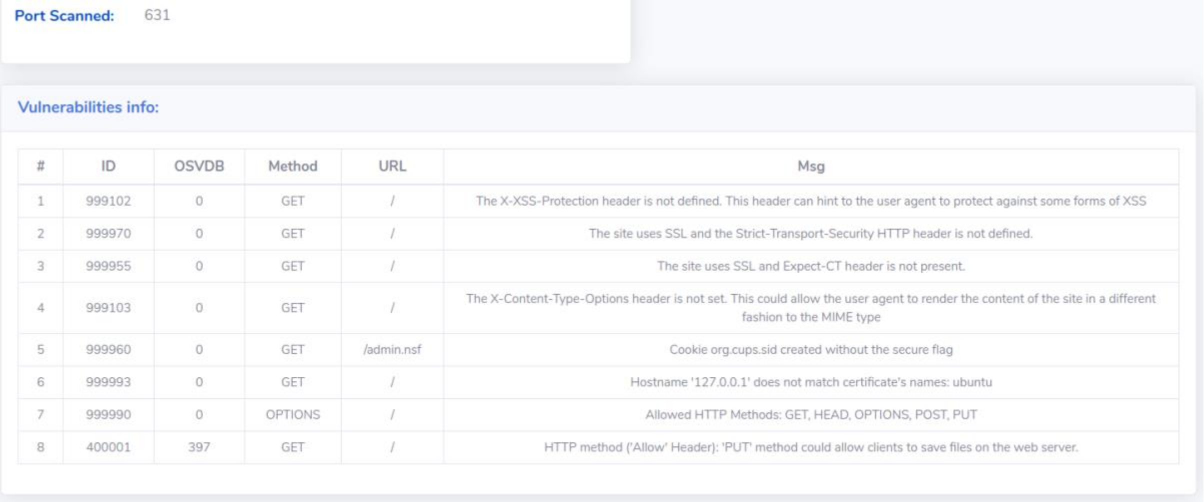
CVE info:

CPE	Properties		
	CVE	CWE	CVSS
cpe:/a:python:python:3.4.2	CVE-2019-13404	CWE-284	9.3
	CVE-2019-9636	CWE-255	5.0
	CVE-2014-4616	CWE-119	4.3
	CVE-2016-5636	CWE-190	10.0
	CVE-2016-0772	CWE-693	5.8
	CVE-2014-9365	None	5.8
	CVE-2014-2667	CWE-362	3.3
	CVE-2016-5699	CWE-113	4.3

Figure V.1.7.3.a.5. CVE-Search CVE info

- ***Nikto***

    - **Port Scanned**: Number of the port
    - **OSVDB**: ID of the Vulnerability on the Open Source Vulnerability Database
    - **Method**: HTTP Method having the Vulnerability
    - **URL**: URL having the Vulnerability
    - **Msg**: Message from Nikto



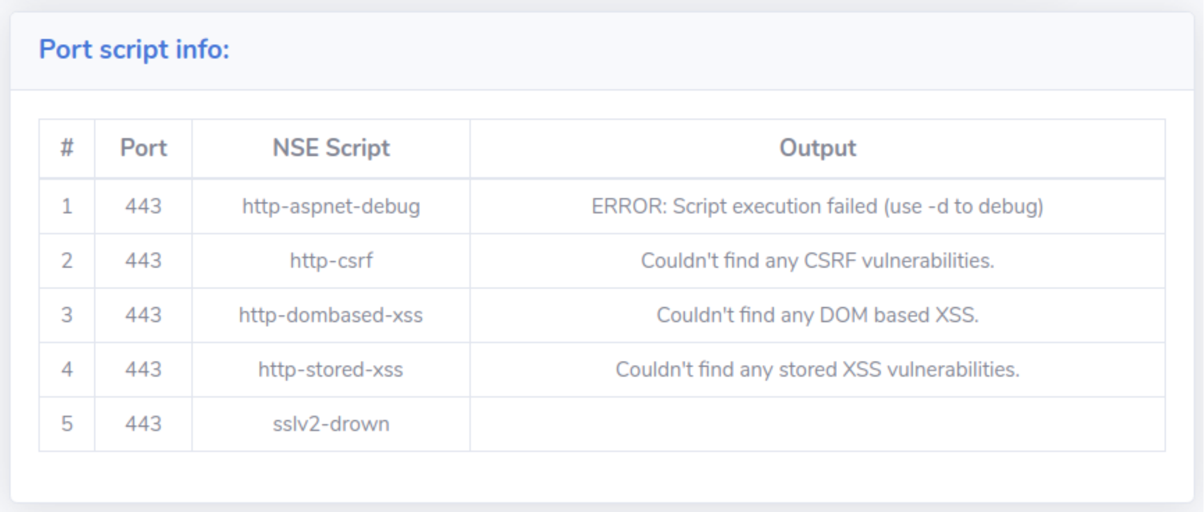
Port Scanned: 631

Vulnerabilities info:

#	ID	OSVDB	Method	URL	Msg
1	999102	0	GET	/	The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
2	999970	0	GET	/	The site uses SSL and the Strict-Transport-Security HTTP header is not defined.
3	999955	0	GET	/	The site uses SSL and Expect-CT header is not present.
4	999103	0	GET	/	The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
5	999960	0	GET	/admin.nsf	Cookie org.cups.sid created without the secure flag
6	999993	0	GET	/	Hostname '127.0.0.1' does not match certificate's names: ubuntu
7	999990	0	OPTIONS	/	Allowed HTTP Methods: GET, HEAD, OPTIONS, POST, PUT
8	400001	397	GET	/	HTTP method ('Allow' Header): 'PUT' method could allow clients to save files on the web server.

Figure V.1.7.3.a.6. Nikto vulnerabilities info

- ***NSE***
    - **Port**: Number of the viewing port
    - **NSE Script**: The NSE script executed on the viewing port
    - **Output**: Output of the script



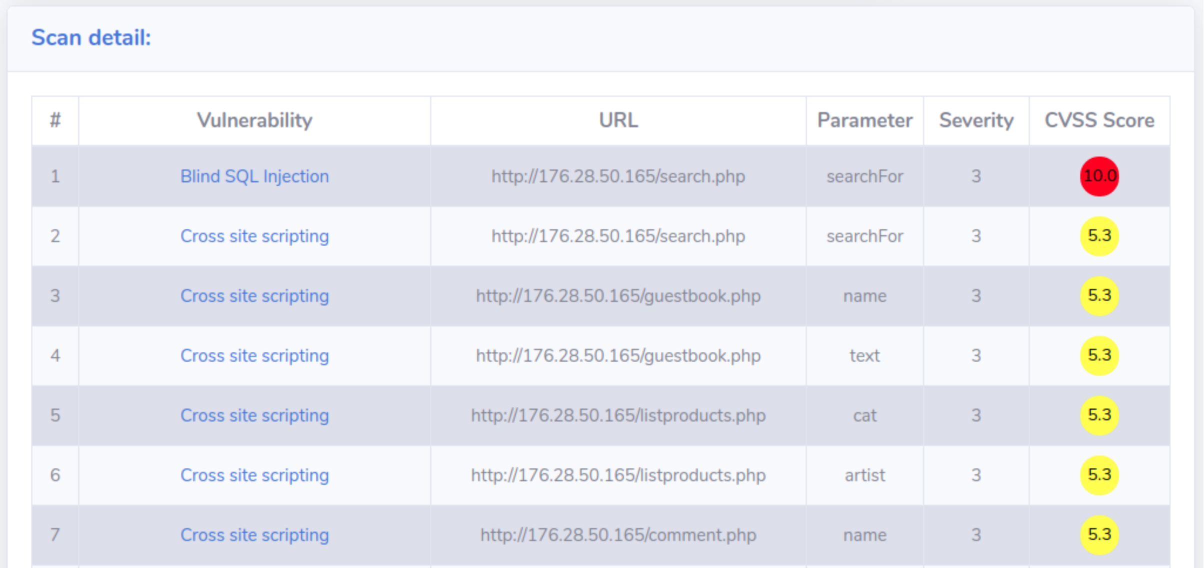
Port script info:

#	Port	NSE Script	Output
1	443	http-aspnet-debug	ERROR: Script execution failed (use -d to debug)
2	443	http-csrf	Couldn't find any CSRF vulnerabilities.
3	443	http-dombased-xss	Couldn't find any DOM based XSS.
4	443	http-stored-xss	Couldn't find any stored XSS vulnerabilities.
5	443	sslv2-drown	

Figure V.1.7.3.a.7. NSE port script info

- ***Acunetix***
    - **Vulnerability**: Name of the vulnerability
    - **URL**: URL of the vulnerability was found
    - **Parameter**: The parameter can be exploited

- **Severity**: the degree of vulnerability impact and the difficulty involved in exploiting it.
- **CVSS Score**: CVSS Score of the Vulnerability

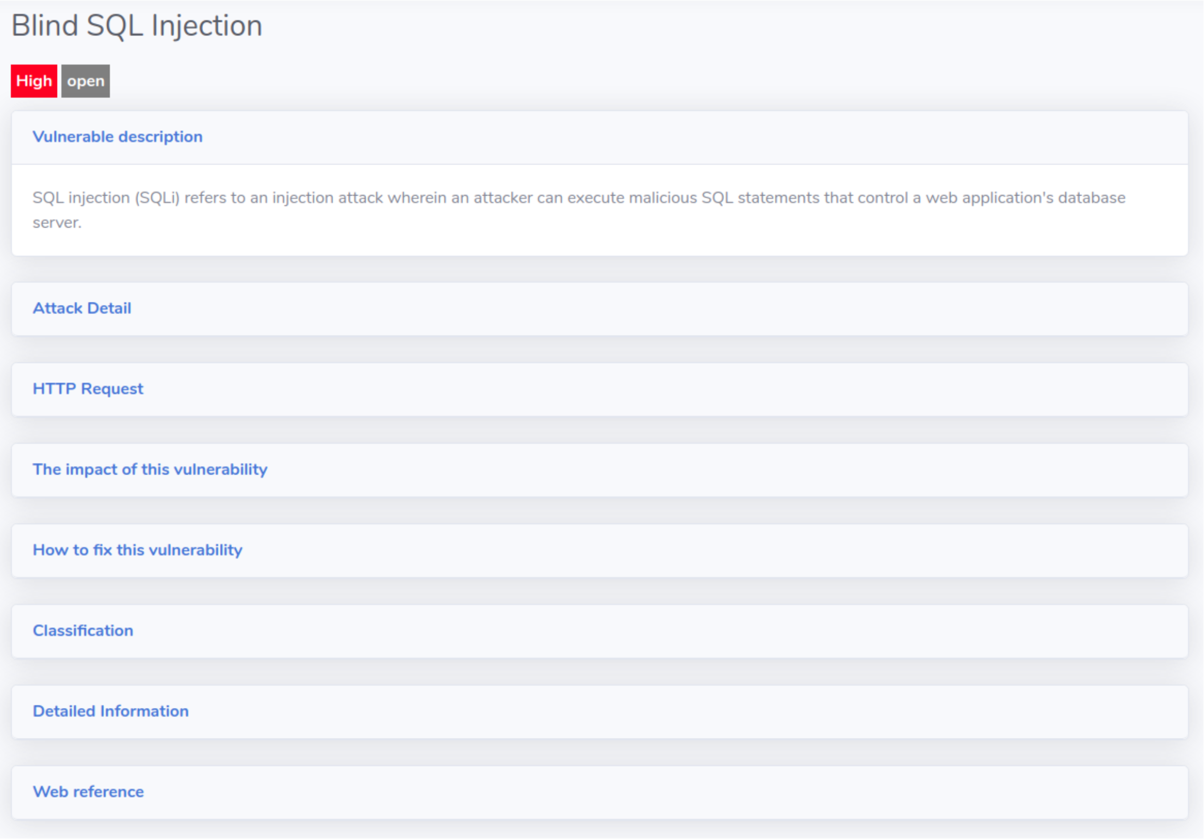


Scan detail:

#	Vulnerability	URL	Parameter	Severity	CVSS Score
1	Blind SQL Injection	http://176.28.50.165/search.php	searchFor	3	10.0
2	Cross site scripting	http://176.28.50.165/search.php	searchFor	3	5.3
3	Cross site scripting	http://176.28.50.165/guestbook.php	name	3	5.3
4	Cross site scripting	http://176.28.50.165/guestbook.php	text	3	5.3
5	Cross site scripting	http://176.28.50.165/listproducts.php	cat	3	5.3
6	Cross site scripting	http://176.28.50.165/listproducts.php	artist	3	5.3
7	Cross site scripting	http://176.28.50.165/comment.php	name	3	5.3

Figure V.1.7.3.a.8. Acunetix vulnerabilities scan info

All the vulnerabilities in this list have more detailed page as below.



Blind SQL Injection

High open

Vulnerable description

SQL injection (SQLi) refers to an injection attack wherein an attacker can execute malicious SQL statements that control a web application's database server.

Attack Detail

HTTP Request

The impact of this vulnerability

How to fix this vulnerability

Classification

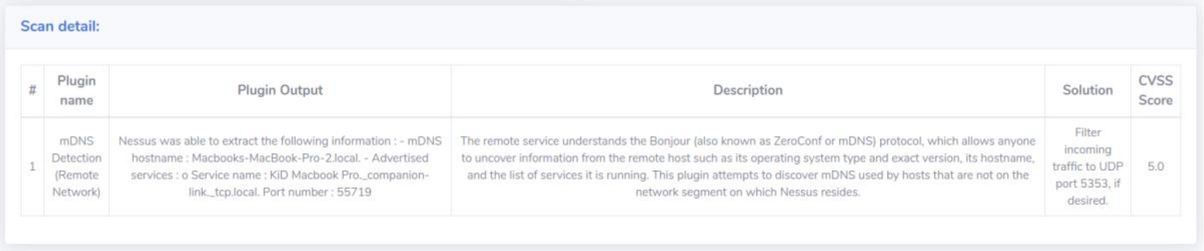
Detailed Information

Web reference

Figure V.1.7.3.a.9. Acunetix vulnerability detail

- ***Nessus***
  - **Plugin name**: Name of the plugin Nessus used to scan on the target

- **Plugin Output**: Output of the plugin (threat, port number, etc)
- **Description**: Description of the vulnerability found
- **Solution**: Suggestion for covering the vulnerability
- **CVSS Score**: CVSS Score of the Vulnerability



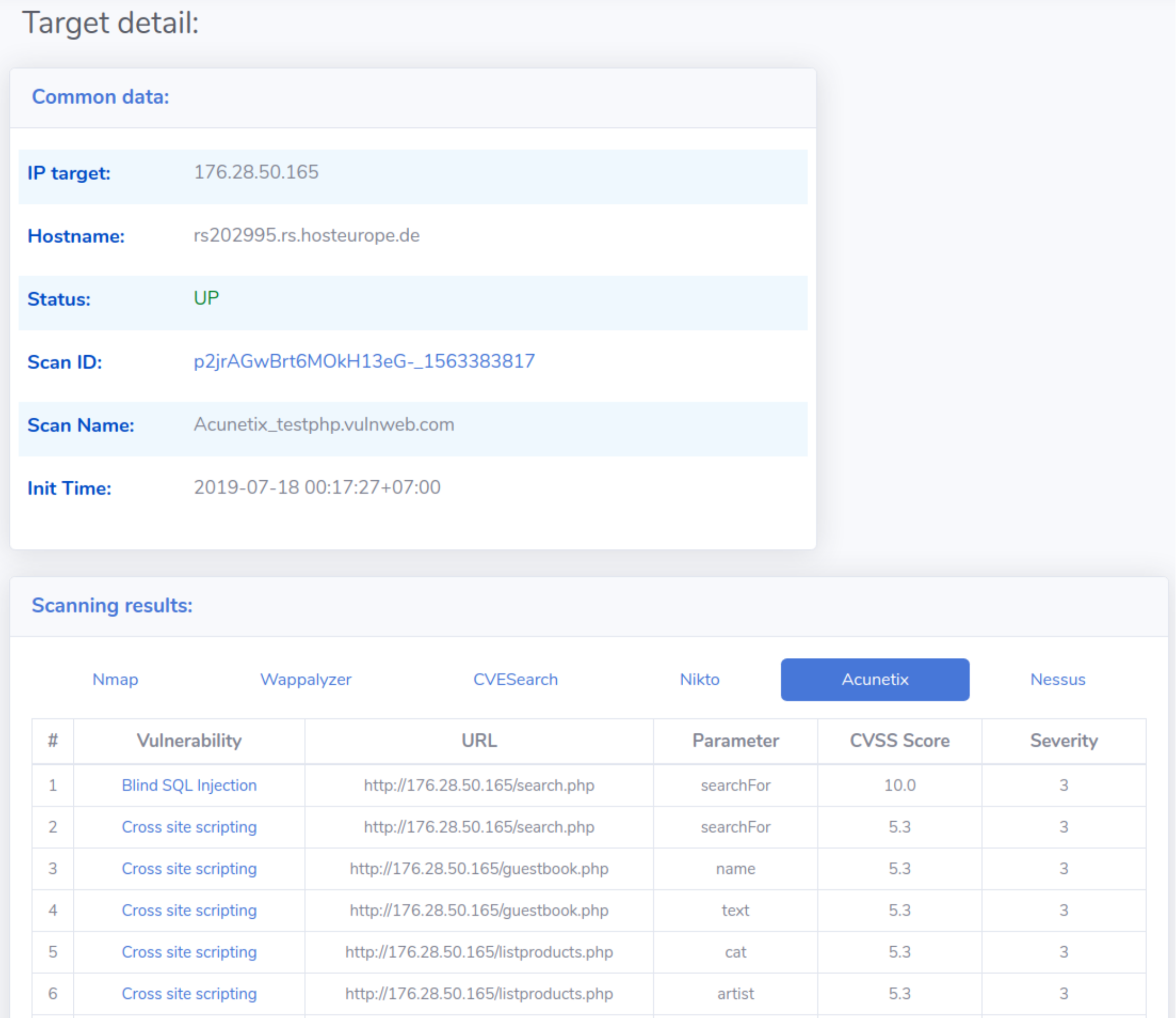
Scan detail:

#	Plugin name	Plugin Output	Description	Solution	CVSS Score
1	mDNS Detection (Remote Network)	Nessus was able to extract the following information : - mDNS hostname : Macbooks-MacBook-Pro-2.local. - Advertised services : o Service name : KiD Macbook Pro._companion-link._tcp.local. Port number : 55719	The remote service understands the Bonjour (also known as ZeroConf or mDNS) protocol, which allows anyone to uncover information from the remote host such as its operating system type and exact version, its hostname, and the list of services it is running. This plugin attempts to discover mDNS used by hosts that are not on the network segment on which Nessus resides.	Filter incoming traffic to UDP port 5353, if desired.	5.0

Figure V.1.7.3.a.10. Nessus scan detail

V.1.7.3.b. Target result (combined from tools)

After the scanning process is done, all the data from the tools are collected, we group it with the key is the IP target and get an overview of all the information of the target. From hostname, what ports are opening, vulnerabilities on those ports... to the vulnerabilities on the webserver of a single target.



Target detail:

Common data:

IP target:	176.28.50.165
Hostname:	rs202995.rs.hosteurope.de
Status:	UP
Scan ID:	p2jrAGwBrt6MOkH13eG-_1563383817
Scan Name:	Acunetix_testphp.vulnweb.com
Init Time:	2019-07-18 00:17:27+07:00

Scanning results:

#	Vulnerability	URL	Parameter	CVSS Score	Severity
1	Blind SQL Injection	http://176.28.50.165/search.php	searchFor	10.0	3
2	Cross site scripting	http://176.28.50.165/search.php	searchFor	5.3	3
3	Cross site scripting	http://176.28.50.165/guestbook.php	name	5.3	3
4	Cross site scripting	http://176.28.50.165/guestbook.php	text	5.3	3
5	Cross site scripting	http://176.28.50.165/listproducts.php	cat	5.3	3
6	Cross site scripting	http://176.28.50.165/listproducts.php	artist	5.3	3

Figure V.1.7.3.b. Target scan detail

V.1.7.3.c. Vulnerabilities result (combined from tools)

As an important function of the Web portal, we support users to manage and statistics Vulnerabilities. Our function can be introduced as follows:

- Screen Transition: from the left menu, click the button Vulnerabilities

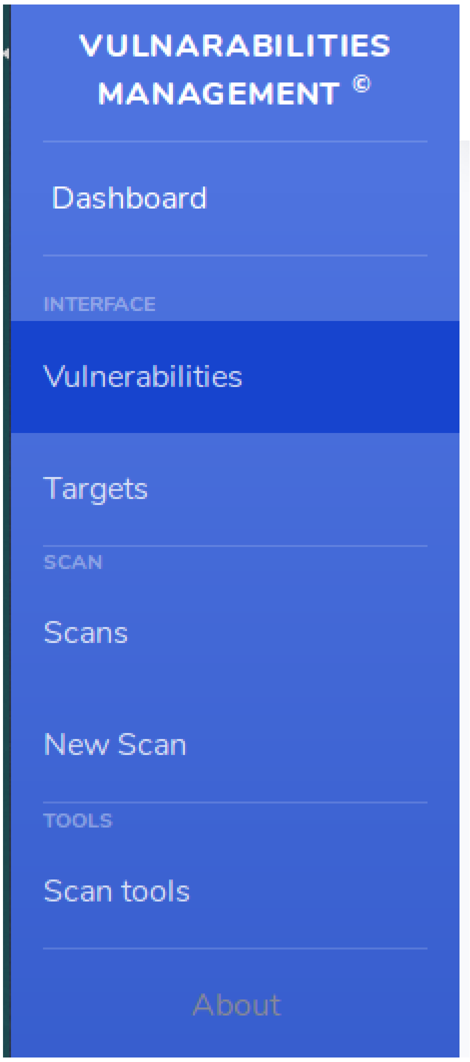
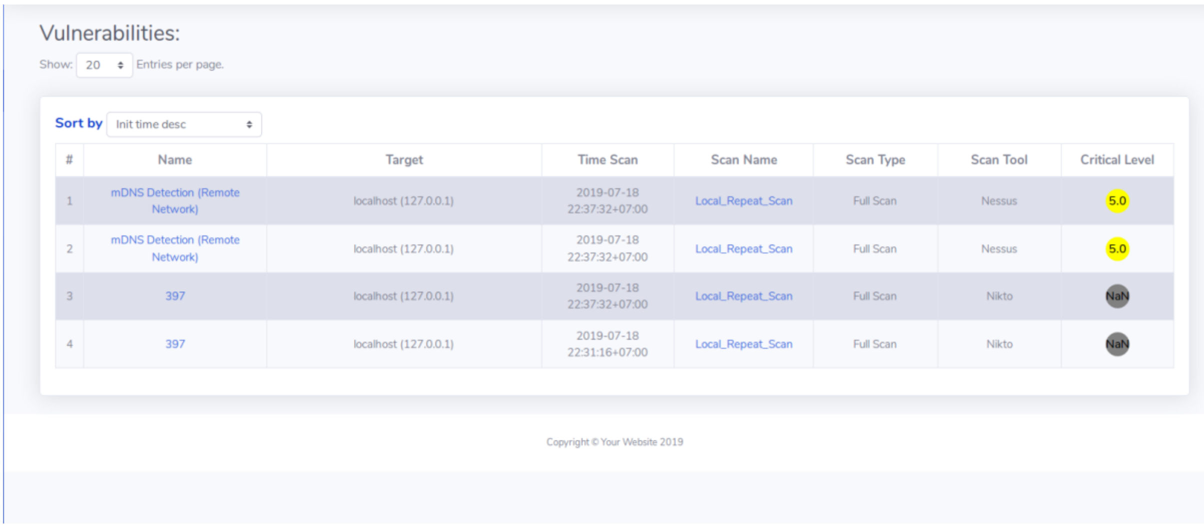


Figure V.1.7.e. Button Vulnerabilities on left menu

- Screen:



Vulnerabilities:

Show: 20 Entries per page.

Sort by Init time desc

#	Name	Target	Time Scan	Scan Name	Scan Type	Scan Tool	Critical Level
1	mDNS Detection (Remote Network)	localhost (127.0.0.1)	2019-07-18 22:37:32+07:00	Local_Repeat_Scan	Full Scan	Nessus	5.0
2	mDNS Detection (Remote Network)	localhost (127.0.0.1)	2019-07-18 22:37:32+07:00	Local_Repeat_Scan	Full Scan	Nessus	5.0
3	397	localhost (127.0.0.1)	2019-07-18 22:37:32+07:00	Local_Repeat_Scan	Full Scan	Nikto	NaN
4	397	localhost (127.0.0.1)	2019-07-18 22:31:16+07:00	Local_Repeat_Scan	Full Scan	Nikto	NaN

Copyright © Your Website 2019

Figure V.1.7.f. Interface of Vulnerabilities Web

- Item :
  - Show [10] Entries per page: A *Selected Attribute* that allows users to adjust the number of items displayed on the page.
  - Sort by [Init Time Desc]: A *Selected Attribute* that allows the user to select the displayed page sorted by Time Scan ascending or descending
  - Table displays item list:
    - #: Numerical order
    - Name: Name of Vulnerability (based on the scanned tool)
    - Target: The IP address of the target having that Vulnerability
    - Time Scan: Time the Vulnerability was found
    - Scan Name: The name that the user places when scanning
    - Scan Type: Commercial and Non-commercial
    - Scan Tool: Tool used has scanned out Vulnerability
    - Critical Level: Displays the severity of Vulnerability
- Backend:

To display the Vulnerability page we have incorporated all the data that the user has scanned. That is the connection of many databases such as Nessus, Nikto, Accunetix, CVE Search to create an object of Scan overview and detail:

```
def generate_info_based_on_index(vul, index, source):
    vuls = []
    if index == "nessus":
        vul['index'] = "Nessus"
        for vul_record in source.get('scan_details'):
```

```
        vul['name'] = vul_record.get('pluginname')
        vul['level'] = vul_record.get('cvss')
        vul['link'] = "/nessus/detail/" + vul['id']
elif index == "nikto":
    vul['index'] = "Nikto"
    for vul_record in source.get('vulnerabilities'):
        vul['name'] = vul_record.get("OSVDB") if vul_record.get("OSVDB") != "0" else "None"
        vul['level'] = "NaN"
        vul['link'] = "/nikto/detail?id=" + vul['id']
elif index == "acunetix_summary":
    vul['index'] = "Acunetix"
    for vul_record in source.get('scan_details'):
        vul['name'] = vul_record.get('vt_name')
        vul['level'] = vul_record.get('cvss_score')
        vul['link'] = "/acunetix/detail/" + vul['id'] + "/" + vul_record.get('vuln_id')
elif index == "cvesearch":
    vul['index'] = "CVE-Search"
    for vul_record in source.get('weakness'):
        vul['name'] = vul_record.get('cve')
        vul['level'] = vul_record.get('cvss')
        vul['link'] = "/cve/detail?id=" + vul['id']
vul['cvss_color'] = generate_cvss_color(vul['level'])

vuls.append(vul)
return vuls
```

V.1.7.3.d. Scan result (combined from tools)

As an important function of the Web portal, we support users to manage and statistics Vulnerabilities. Our function can be introduced as follows:

- Screen Transition: from the left menu, click the button **Scans**

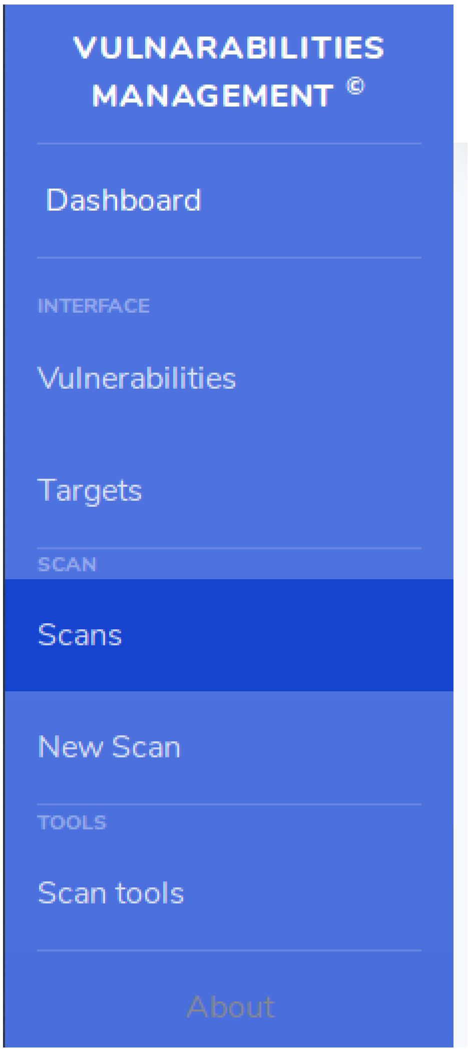
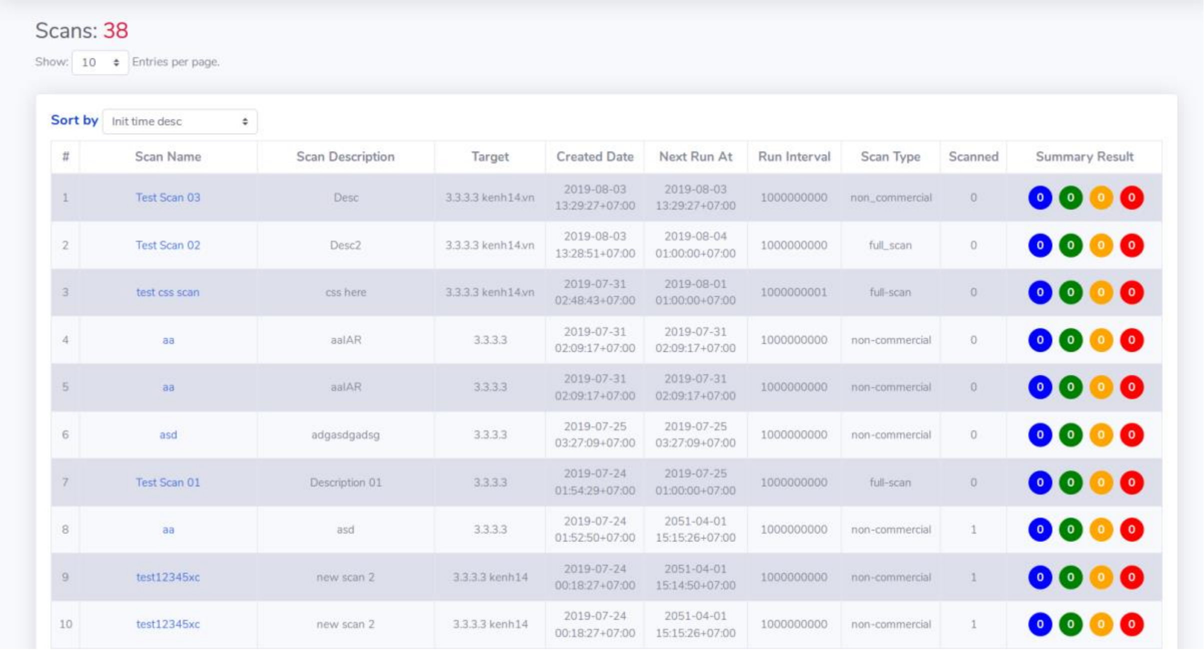


Figure V.1.7.g. Button Scans on left menu

- Screen:



Scans: 38

Show: 10 Entries per page.

Sort by Init time desc

#	Scan Name	Scan Description	Target	Created Date	Next Run At	Run Interval	Scan Type	Scanned	Summary Result
1	Test Scan 03	Desc	3.3.3.3 kenh14.vn	2019-08-03 13:29:27+07:00	2019-08-03 13:29:27+07:00	1000000000	non_commercial	0	0 0 0 0
2	Test Scan 02	Desc2	3.3.3.3 kenh14.vn	2019-08-03 13:28:51+07:00	2019-08-04 01:00:00+07:00	1000000000	full_scan	0	0 0 0 0
3	test css scan	css here	3.3.3.3 kenh14.vn	2019-07-31 02:48:43+07:00	2019-08-01 01:00:00+07:00	1000000001	full-scan	0	0 0 0 0
4	aa	aalAR	3.3.3.3	2019-07-31 02:09:17+07:00	2019-07-31 02:09:17+07:00	1000000000	non-commercial	0	0 0 0 0
5	aa	aalAR	3.3.3.3	2019-07-31 02:09:17+07:00	2019-07-31 02:09:17+07:00	1000000000	non-commercial	0	0 0 0 0
6	asd	adgasdgadsg	3.3.3.3	2019-07-25 03:27:09+07:00	2019-07-25 03:27:09+07:00	1000000000	non-commercial	0	0 0 0 0
7	Test Scan 01	Description 01	3.3.3.3	2019-07-24 01:54:29+07:00	2019-07-25 01:00:00+07:00	1000000000	full-scan	0	0 0 0 0
8	aa	asd	3.3.3.3	2019-07-24 01:52:50+07:00	2051-04-01 15:15:26+07:00	1000000000	non-commercial	1	0 0 0 0
9	test12345xc	new scan 2	3.3.3.3 kenh14	2019-07-24 00:18:27+07:00	2051-04-01 15:14:50+07:00	1000000000	non-commercial	1	0 0 0 0
10	test12345xc	new scan 2	3.3.3.3 kenh14	2019-07-24 00:18:27+07:00	2051-04-01 15:15:26+07:00	1000000000	non-commercial	1	0 0 0 0

Figure V.1.7.h. The interface of Scans Web

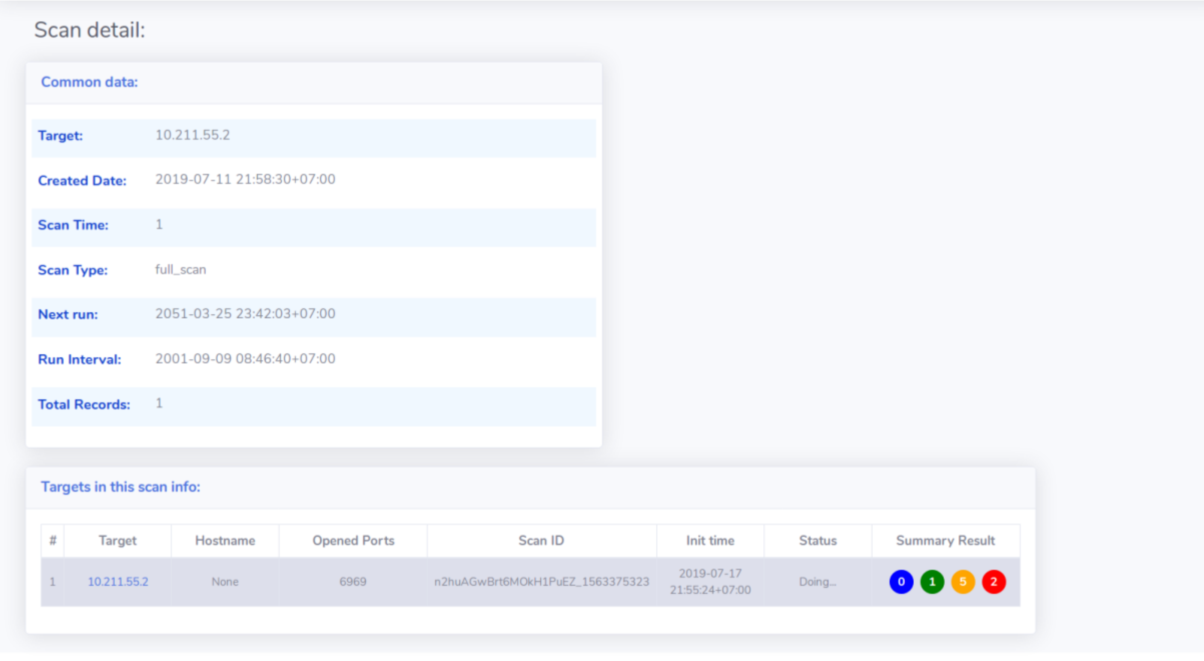
- Item :
  - Show [10] Entries per page: A Selected Attribute that allows users to adjust the number of items displayed on the page.
  - Sort by [Init Time Desc]: A Selected Attribute that allows the user to select the displayed page sorted by Time Scan ascending or descending
  - Table displays item list:
    - #: Numerical order.
    - Scan Name: The name that the user places when scanning.
    - Scan Description: Detailed description of the scan.
    - Target: The IP address /IP range that user Scanned.
    - Created Date: Time to create "**New Scan**".
    - Next Run At :It is the last time scanned with the IP address / IP range.
    - Run Interval:
    - Scan Type: Commercial and Non-commercial.
    - Scanned: The number of scans repeated on the IP address or IP range.
    - Summary Result: Displays the number of vulnerabilities in different levels from Informational to High.
- Backend:

To display the Scan page we have incorporated all the data that the user has scanned. That is the connection of many databases such as Nessus, Accunetix, CVE Search to create an object of Scan overview and

Scan Detail :

- Screen Transition: click the **Target** in table Scans

- Screen :



Scan detail:

Common data:

Target:	10.211.55.2
Created Date:	2019-07-11 21:58:30+07:00
Scan Time:	1
Scan Type:	full_scan
Next run:	2051-03-25 23:42:03+07:00
Run Interval:	2001-09-09 08:46:40+07:00
Total Records:	1

Targets in this scan info:

#	Target	Hostname	Opened Ports	Scan ID	Init time	Status	Summary Result
1	10.211.55.2	None	6969	n2huAGwBrt6MOkH1PuEZ_1563375323	2019-07-17 21:55:24+07:00	Doing...	0 1 5 2

Figure V.1.7.i. The interface of Scan Detail Web

  - *Common data:* Display information about target
    - Item :
      - Target: The IP address /IP range that user Scanned.
      - Created Date: Time to create "**New Scan**".
      - Scan Time: The number of scans repeated on the IP address or IP range.
      - Scan Type: Commercial (full_scan) and Non-commercial.
      - Next run: It is the last time scanned with the IP address / IP range.
      - Total Records: Total number of scanned targets.
  - *Targets in this scan info :* Specific information about scanned targets
    - Item:
      - #: Numerical order.
      - Target: The IP address /IP range that user Scanned.
      - Hostname: Name of the website, host
      - Opened Ports: Ports detected open on IP address
      - Scan ID:
      - Init time: Scan time of that IP address
      - Status: Status of the Scan : *Doing*

- Summary Result: Displays the number of vulnerabilities in different levels from Informational to High.

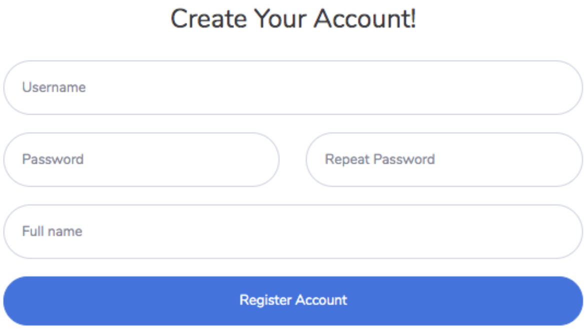
V.1.7.4. Analysis Scan Informations.

V.1.7.5. Authentication.

The most important part in Web Portal is authentication because this contains a lot of sensitive data. If a hacker or an unauthorized user can access Web Portal it will make a critical risk on all over the system of organization who is using our project. Because of this, we decided to apply some protection on this part.

First deploy:

When the first deploy the Web Portal, system admin have to create an admin account. After create account this endpoint will can't access anymore.



Create Your Account!

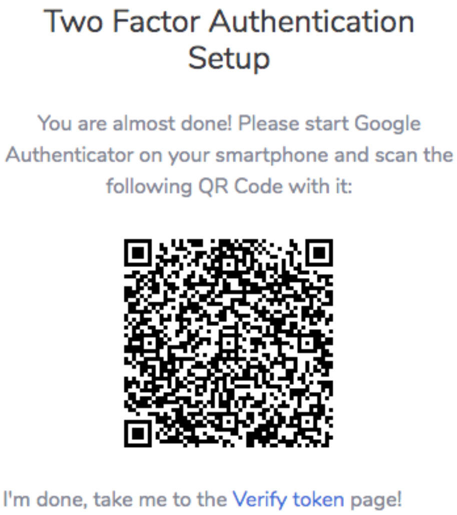
Figure V.1.7.j. First deploy create account

Login:

A user has to login before using any function on this Web Portal.

After login will have 2 cases: **First login user** or **User has been logged before**.

If in case **First login user**, it will require user to use a Time-based One-time Password (TOTP) application such as Google Authenticator or any application like that on mobile phone to create Two-factor Authentication by scanning QR code. This QR code just show only one time.



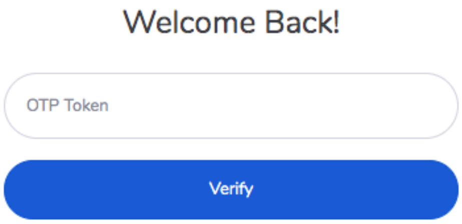
Two Factor Authentication Setup

You are almost done! Please start Google Authenticator on your smartphone and scan the following QR Code with it:

I'm done, take me to the Verify token page!

Figure V.1.7.k. Two-Factor Authentication Setup

In another case, **User has been logged before**, user has to provide valid Token display in Authenticator application on mobile phone to access to Web Portal.



Welcome Back!

Figure V.1.7.l. Verify OTP Token

Role of user on system:

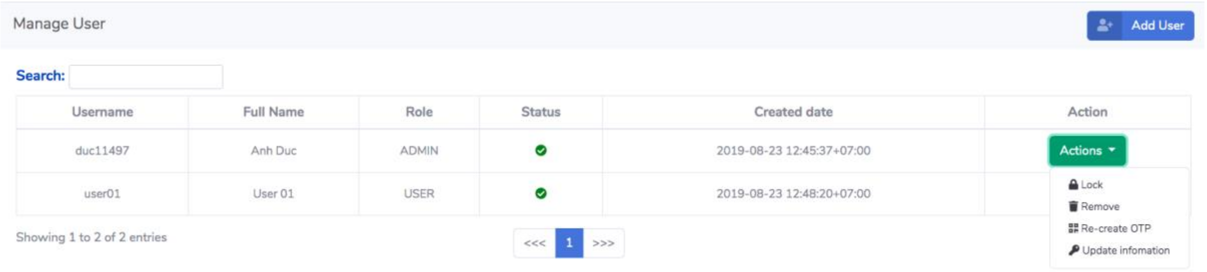
There are 2 roles on Web Portal: **Administrator** and **Normal User**

Normal User can use some functions such as: view information of scan or create scan, change password by themselves...

Administrator can control or created other **Administrator** and **Normal User** and can use all the functions like **Normal User**

Manage user on system (Administrator Function):

Only **Administrator** can access **Manage user** page. As I said before, **Administrator** can control other **User** on system like: Create a new account, lock or unlock account, Re-create OPT...



Manage User

Username	Full Name	Role	Status	Created date	Action
duc11497	Anh Duc	ADMIN	✓	2019-08-23 12:45:37+07:00	Actions
user01	User 01	USER	✓	2019-08-23 12:48:20+07:00	

Showing 1 to 2 of 2 entries

Figure V.1.7.m. Manage user page

V.2. Master Service, How We Assign Jobs For Agents And How Do Master And Agent Can Exchange Data?

V.2.1. How we used Kafka Message Queue to know which tools to be used next?

As we mentioned before at V.1.1. Kafka Message Queue that we have multiple topics for each tool. And the flow of a target when it comes to our solution will be presented more details at V.4. How Does A Target Being Scanned When A User Create New Scan. In this section, we will present how Kafka Message Queue helps us to know which tools to be used next.

At each tool, we created their topic in Kafka, and each tool we also have their two consumers, one for input to the topic and the other solve the output data of the tool. The below diagram shows how it works.

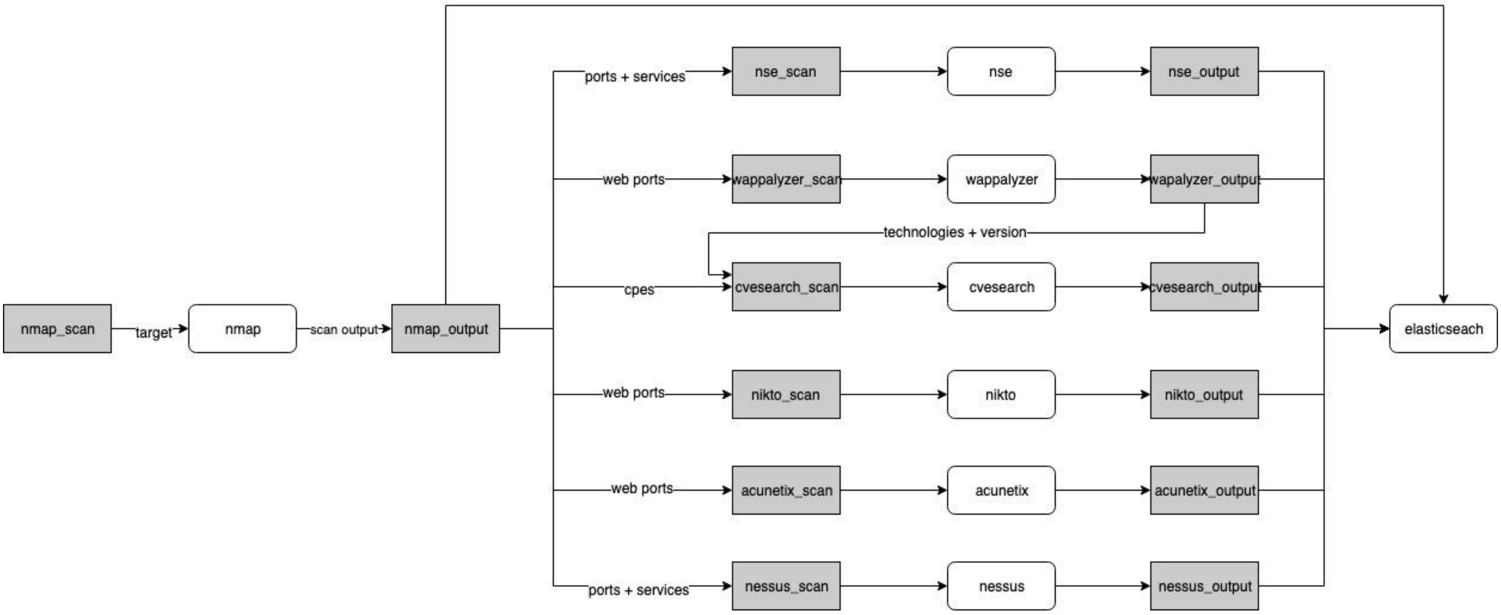


Figure V.2.1.a. Diagram Of Kafka Topic Flow

- ❑ The white box is the **tool.**
- ❑ The gray box is the **Kafka topic consumer**.

At first, we pass the target to **nmap_scan** and it will be scanned with **Nmap**, then the output of Nmap will pass to **nmap_output**. At here, we process data and decide where it will go next based on their information.

- Ports + service will go to **nse_scan**, **nessus_scan.**
- Web ports will go to **wappalyzer_scan**, **nikto_scan**, **acunetix_scan.**

- Cpes (the result of Nmap) and technologies (the result of wappalyzer) will go to **cvesearch_scan.**

After each tool above finish their scan, they will send the output data to **ElasticSearch database.**

V.2.2. What Is Xinetd? And Why And Where We Used Xinetd To Communicate Between Master And Agent?

To use microservice architecture, we have to have a way to communicate between services through the network. Usually, if we have a client-server model, they will use RESTful API to communicate between them. But we want to create a socket channel for client and server can have a persistent channel to communicate. So we decided to use a socket.

But to manage socket, we have to code a lot of handles a lot of cases, optimize to connection and secure them. So we found out a much more convenient way to use socket is use **Xinetd**.

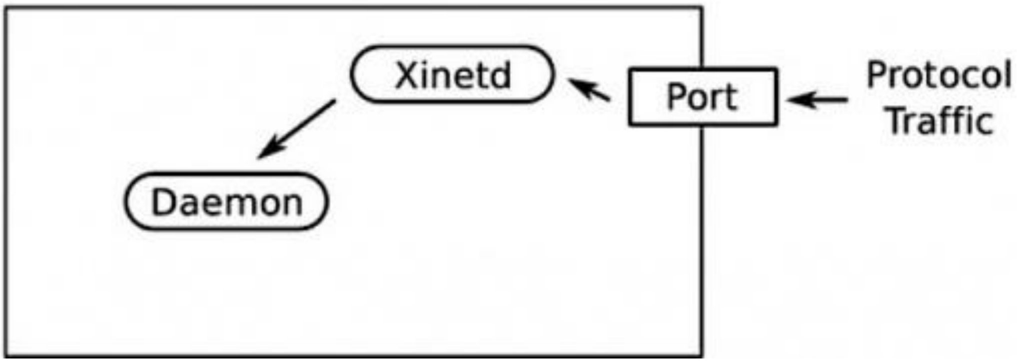


Figure V.2.2.a Xinetd Flow Diagram

Xinetd is an open-source super-server daemon, runs on many Unix-like systems and manages Internet-based connectivity. We just need to code a normal python file read from stdin and print output to stdout… Xinetd will help us to make this application to a network service. What we need is a deploy file like below:

```
#!/bin/bash
apt-get update
apt-get install python3 xinetd -y
python3 deploy.py
```

```
#!/usr/bin/python3
#coding=utf8

import os
from os import system
import sys

cwd = os.getcwd()

cveSearchAgent = '''service cveSearchAgent
{
        socket_type     = stream
        protocol        = tcp
        user            = root
        wait            = no
        server          = /usr/bin/python3
        server_args     = %s/cveSearchAgent.py -u
        port            = 25799
}
''' % cwd

cveSearchService = 'cveSearchAgent       25799/tcp                       #
cveSearchAgent\n'


open('/etc/xinetd.d/cveSearchAgent','w').write(cveSearchAgent)
print('[OK] Added to xinetd.d')

open('/etc/services','a').write(cveSearchService)
print('[OK] Added new service to /etc/services')

system('/etc/init.d/xinetd restart')

# create Log Path
path = "/var/log/cveSearch"
os.mkdir(path)

print ("Log Path is created")

print('''
============================
||  [+] Deploy finish :)  ||
============================
''')
```

Now, in the Agent server already have a service running. In the master server, we can use a socket to connect to it as normal.

```
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
       s.connect((self.HOST, self.PORT))
       jData = json.dumps(self.data) + '\n'
       s.sendall(jData.encode('utf-8'))
```

```
        jOutputData = b''
        while True:
            _data = s.recv(1024)
            if not _data:
                break
            jOutputData += _data
```

V.2.3. What Is Web REST API? And Why And Where We Used Web RESTful API To Communicate Between Master And Agent?

Beside connect through the socket and use the help of **Xinetd**, we also can use Web RESTful API to communicate between master and agent in model client-server. We use it in CVE-Search module because they already provided us the Web API to search for CVE.

By using the Web RESTful API, it helps us to reduce the burden of coding and managing socket so we can save a lot of time. And the pros of this method is we can use async to speed up instead of using multi-thread like in xinetd. The code of async is below:

```
def scanWithAsyncIO(self, cpes):
    loop = asyncio.new_event_loop()
    asyncio.set_event_loop(loop)
    result_cpes = loop.run_until_complete(self.request_all_vuls(cpes))
    loop.close()

    return result_cpes

async def sendRequest(self, session, cpe):
    try:
        url = CVESearchConfig.URL + CVESearchConfig.API_CVEFOR + cpe
        async with session.get(url) as response:
            response = await response.json()
            result = []
            for res in response:
                cve = {}
                cve["cvss"] = float(res.get("cvss"))
                cve["cwe"] = res.get("cwe")
                cve["cve"] = res.get("id")
                cve["cpe"] = cpe
                result.append(cve)
            return result
    except:
        self.logger.exception()
        return []
```

V.3. How Does A Target Being Scanned When A User Create New Scan.

In this section, we will show you which step a target will go through when it be passed to our system. The below diagram shows all the steps, it looks like Figure V.2.1.a but have some different components.

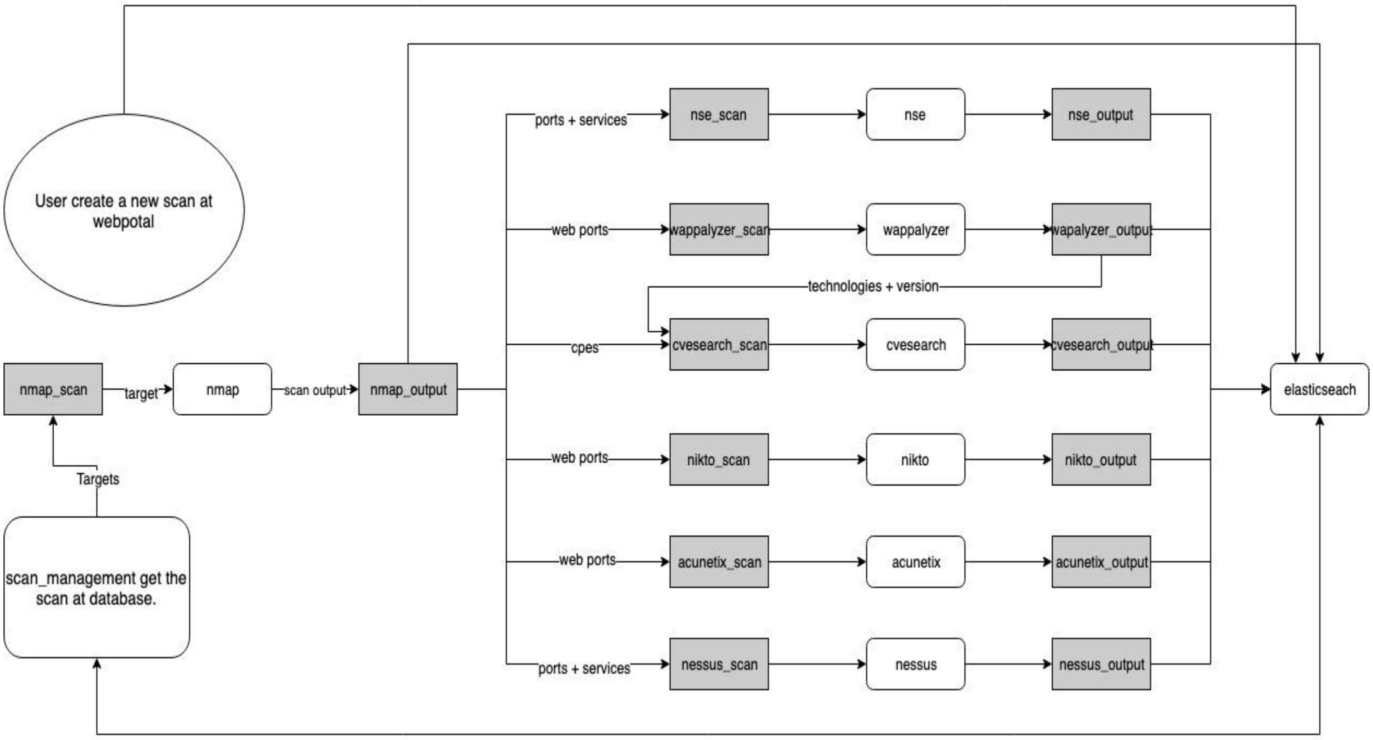


Figure V.3.a. Diagram Scan Life Cycle.

First, the user will create a new scan at the web portal, in the create new scan section, user can choose if this scan is a scheduled scan or not and set the type of scan. We have two different types of scan, that is non-commercial and commercial. The non-commercial will not contain **acunetix** and **nessus**. Then all the information which user have just inputted be sent to our Elasticsearch database.

We have a module in master service that is called "scan_management" which is a cron task will always check for a new scan on the Elasticsearch database. We get all the scan which have the "`next_run_at`" field which is lower or equal to the current time . below is the query of our "scan_management".

```
body = {
    "query": {
        "range": {
            "next_run_at": {
                "lte": time.time()
            }
        }
    }
}
```

Then we get the scan information, we will split all the targets in that scan and put it to the same process as we have presented at the "**V.2.1. How we used Kafka Message Queue to know which tools to be used next?**" section.

At each tool, the master will look for the agent which run that tool and have a remaining slot for this target to be scanned, if we found out that agent, we assign job for that agent to scan this target. The way to assign a job for an agent has been presented at the "**V.2.2**" and "**V.2.3**" sections. Then the master will create a new thread to wait for the agent to finish the job, that process is presented in the "**V.1.4**" section.

V.4. Configuration Files.

- **Why do we choose Python as a config file but not others type of config file?**
  If a developer needs to use a config file, they usually choose JSON, XML or INI. But we decided to use a python file as a configuration file. Following table is a comparison between these formats.

Python	JSON	XML	INI (Linux)
Can write comment	Can NOT write comment	Can write comment	Can write comment
Easy for humans to read	Easy for humans to read	Not Easy for humans to read	Not Easy for humans to read
Do not need to use any library	Need to use library	Need to use library	Need to use library
Just support Python	Support many programing languages	Support many programing languages	Support many programing languages
Can include code at the configuration files	Can NOT include code	Can NOT include code	Can NOT include code
Do not need to learn new format	Need to know about JSON	Need to know about XML	Need to know about INI

After comparing all these type, we decided to use Python because it is the most suitable type of configuration files for our project.

- **What do the configuration files contains?**

The following code is the configuration file of master server. It includes every configuration the master server need to be runnable. Each class represent for config of a module in master server.

```
class KafkaConfig:
    BOOTSTRAPSERVER     = 'localhost:9092'

class KafkaTopicNames:
    NMAPSCAN            = 'nmapScan'
    NMAPOUTPUT          = 'nmapOutput'
    CVESEARCHSCAN       = 'CVESearchScan'
    CVESERACHOUTPUT     = 'CVESearchOutput'
    NSESCAN             = 'nseScan'
    NSEOUTPUT           = 'nseOutput'
    ELASTICSEND         = 'elasticSend'
    WAPPALYZERSCAN      = 'wappalyzerScan'
    WAPPALYZEROUTPUT    = 'wappalyzerOutput'
    NESSUSSCAN          = 'nessusScan'
    NESSUSOUTPUT        = 'nessusOutput'
    ACUNETIXSCAN        = 'acunetixScan'
    ACUNETIXOUTPUT      = 'acunetixOutput'
    NIKTOSCAN           = 'niktoScan'
    NIKTOOUTPUT         = 'niktoOutput'

class KafkaGroupIds:
    NMAPSCAN            = 'nmapScan'
    NMAPOUTPUT          = 'nmapOutput'
    CVESEARCHSCAN       = 'CVESearchScan'
    CVESERACHOUTPUT     = 'CVESearchOutput'
    NSESCAN             = 'nseScan'
    NSEOUTPUT           = 'nseOutput'
    ELASTICSEND         = 'elasticSend'
    WAPPALYZERSCAN      = 'wappalyzerScan'
    WAPPALYZEROUTPUT    = 'wappalyzerOutput'
    NESSUSSCAN          = 'nessusScan'
    NESSUSOUTPUT        = 'nessusOutput'
    ACUNETIXSCAN        = 'acunetixScan'
    ACUNETIXOUTPUT      = 'acunetixOutput'
    NIKTOSCAN           = 'niktoScan'
    NIKTOOUTPUT         = 'niktoOutput'

class ElasticConfig:
    HOSTNAME            = 'vulnerabilitymanabem-1160079090.us-west-2.bonsaisearch.net'
    USERNAME            = 'rugowuyofm'
    PASSWORD            = 'nfsnbt15y2'
    USESSL              = True
    PORT                = 443
    NMAPINDEX           = 'nmap'
    CVESEARCHINDEX      = 'cvesearch'
    NSEINDEX            = 'nse'
    WAPPALYZERINDEX     = 'wappalyzer'
    ACUNETIXINDEX       = 'acunetix'
    ACUNETIX_SUMARY_INDEX   = 'acunetix_summary'
    ACUNETIX_DETAIL_INDEX   = 'acunetix_detail'
    SCAN_INDEX          = 'scan'
    NIKTO_INDEX         = 'nikto'
    NESSUS_INDEX        = 'nessus'

class CVESearchConfig:
    URL                 = 'http://localhost:6969/'
    API_CVEFOR          = 'api/cvefor/'
    CVESEARCH_ADDRESS   = '127.0.0.1'
    CVESEARCH_PORT      = 25799

class NmapAgents:
    NMAPAGENTSADDRESS    = [{'HOST': '10.211.55.11', 'PORT': 25797}, {'HOST':
'192.168.31.196', 'PORT': 25797}]
    MAX_SCAN_PER_AGENT   = 1
    MAX_RETRY_TIMES      = 2
    THREAD_SLEEP_SECOND  = 10
    MAX_POLL_INTERVAL_MS   = 60*60*1000 # An hour in ms
    SESSION_TIMEOUT_MS     = 60*1000

class NseAgents:
    NSEAGENTSADDRESS    = [{'HOST': '10.211.55.11', 'PORT': 25798}, {'HOST':
'192.168.31.196', 'PORT': 25798}]
    MAX_SCAN_PER_AGENT   = 1
    MAX_RETRY_TIMES      = 2
    THREAD_SLEEP_SECOND  = 15
    MAX_POLL_INTERVAL_MS   = 60*60*1000 # An hour in ms
    SESSION_TIMEOUT_MS     = 60*1000

class WappalyzerAgents:
    WAPPALYZER_AGENT_ADDRESS    = [{'HOST': '10.211.55.11', 'PORT': 11497}]

class NessusAgents:
    NESSUS_AGENT_ADDRESS          = [{'HOST': '10.211.55.11', 'PORT': 25701}]
    MAX_POLL_INTERVAL_MS          = 60*60*1000 # An hour in ms
    MAX_SCAN_PER_AGENT            = 1
    THREAD_SLEEP_SECOND           = 20
    SESSION_TIMEOUT_MS            = 60*1000

class AcunetixAgents:
    ACUNETIX_AGENT_ADDRESS        = [{'HOST': 'localhost', 'PORT': 25700}]
    MAX_POLL_INTERVAL_MS          = 60*60*1000 # An hour in ms
    MAX_SCAN_PER_AGENT            = 1
    THREAD_SLEEP_SECOND           = 20
    SESSION_TIMEOUT_MS            = 60*1000

class NiktoAgents:
    NIKTO_AGENT_ADDRESS           = [{'HOST': '10.211.55.11', 'PORT': 25801}]
    MAX_POLL_INTERVAL_MS          = 60*60*1000 # An hour in ms
    MAX_SCAN_PER_AGENT            = 1
    THREAD_SLEEP_SECOND           = 20
    SESSION_TIMEOUT_MS            = 60*1000

class ScanManagement:
    GET_SCAN_INTERVAL           = 15
    DEFAULT_SIZE                = 10
```

V.5. How Logging Have Been Saved And Managed?

Logging is an essential factor of a system because it helps the developers to know the state of the system, are errors occur when running, how long does it take your system to run, etc. So we also developed a logging system for our solution to know how well everything is running.

We use the logger library of python to log important information, for example, the error of any module, how data is transferred through modules, etc. The logging configuration is shown below.

```
loggingConfig = {
    "version": 1,
    "disable_existing_loggers": False,
    "formatters": {
        "simple": {
            "format": "%(name)-45s - %(levelname)-6s - %(message)s"
        },
        "extended": {
            "format": "%(asctime)s - %(name)-45s - %(levelname)-6s - %(message)s"
        },
        "json": {
            "format": "name: %(name)s, level: %(levelname)s, time: %(asctime)s, message:
%(message)s"
            }
    },

    "handlers": {
        "console_handler": {
            "class": "logging.StreamHandler",
            "level": "DEBUG",
            "formatter": "simple",
            "stream": "ext://sys.stdout"
        },

        "info_file_handler": {
            "class": "logging.FileHandler",
            "level": "INFO",
            "formatter": "extended",
            "filename": "Log/info.log",
            "encoding": "utf8",

            "mode": "a"
        },
        "debug_file_handler": {
            "class": "logging.FileHandler",
            "level": "DEBUG",
            "formatter": "extended",
            "filename": "Log/debug.log",
            "encoding": "utf8",

            "mode": "a"
        },

        "error_file_handler": {
            "class": "logging.FileHandler",
            "level": "ERROR",
            "formatter": "extended",
            "filename": "Log/error.log",
            "encoding": "utf8",

            "mode": "a"
        }
    },

    "loggers": {
        "Module": {
            "level": "DEBUG",
            "propagate": True,
            "handlers": ["console_handler", "info_file_handler"]
        }
    },
    "root": {
        # Use this to log error message to file
        "level": "ERROR",
        "handlers": ["error_file_handler"]
    }
}
```

So the logs can be shown in the terminal and the log file at the same time. They are separated into info log and error log make the debugging work much easier.

VI. Project Validation

VI.1. Project Idea.

Today, the protection of organizations information are a top concern in the field of information security. Every organization needs a thorough risk management, vulnerability solution for their system so we want to build a technology system that can help system admin and also user normal can look on interface and known which vulnerabilities are exists on that system, has it fixed or any new vulnerabilities are explores.

VI.2. Result.

During the 4 months, we have built up a system of three components: Web Application, Agent systems, Back-end Services.

The web application system provides user authentication, vulnerability management interface, statistic vulnerabilities how dangerous they are, scored by third-party, web application also help system admin tracking their system and vulnerabilities.

Agent system can deploy on different servers or a single servers powerful enough configuration in a flexible way. Each agent has a seperate function and work together, data of agent are push on master machine then processing to display final perspicuous data to user.

Back-end services are built to process authentication and receive data got by agent and display to user and receive user command, process it then push back to agent handling in cycle.

VI.3. Future.

The need to secure critical information is increasing, so the demand for a system vulnerabilities management in enterprise will grow up. In the future, we intend to develop more feature integrated more tools to get more reliable result, improve performance, speed up scanning time. We also have plans to develop more function not only management but also scanning specific vulnerabilities on system, so our solution can be applied more widely.

DEFINITION AND ACRONYMS

Acronym	Definition	Note
CVE	The Common Vulnerabilities and Exposures	
CVSS	Common Vulnerability Scoring System	
NSE	Nmap Scripting Engine	
SQL	Structured Query Language	
CPE	Common Platform Enumeration	
API	Application Program Interface	
RCE	Remote Code Execution	
PDF	Portable Document Format	
HTTP	Hypertext Transfer Protocol	
HTTPS	Hypertext Transfer Protocol Secure	
GUI	Graphic User Interfaces	
SSD	Solid State Drive	
RAM	Random Access Memory	
VPS	Virtual Private Server	
CPU	Central Processing Unit	
OS	Operating System	
VPN	Virtual Private Network	
IDS	Intrusion Detection System	
IPS	Intrusion Prevention System	
URL	Uniform Resource Locator	
NMAP	Network Mapper	
IP	Internet Protocol	
JSON	JavaScript Object Notation	
XML	Extensible Markup Language	
WAN	Wide Area Network	
LAN	Local Area Network	
VA	Vulnerabilities Assessment	
XSS	Cross Site Scripting	

WAF	Web Application Firewall	
DOS	Denial of service	

Appendix A – References

A.1. Books, Newspaper, and Magazines

[1.1] Darril Gibson, Managing Risk in Information Systems, Jones & Bartlett Learning 2011.

A.2. Websites and Internet Resources

[2.1] Owasp. (2019). *OWASP Risk Rating Methodology - OWASP*. [online] Available at:
https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology
[Accessed 30 Jul. 2019].

[2.2] Nmap. (2019). *Introduction*. [online] Available at:
https://nmap.org/
[Accessed 03 Aug. 2019].

[2.3] Wappalyzer.com. (2019). *About*. [online] Available at:
https://www.wappalyzer.com/about
[Accessed 02 Aug. 2019].

[2.4] Acunetix. (2019). *Introduction to Acunetix*. [online] Available at:
https://www.acunetix.com/support/docs/introduction/
[Accessed 31 Jul. 2019].

[2.5] Nikto. (2019). *Description*. [online] Available at:
https://cirt.net/nikto2-docs/introduction.html
[Accessed 01 Aug. 2019].

[2.6] Nessus. (2019). *Nessus Built for practitioners, by practitioners*

. [online] Available at:

https://www.tenable.com/products/nessus

[Accessed 03 Aug. 2019].

[2.7] Vulnerability assessment. (2019). *Vulnerability assessment — one step further towards a better safety*

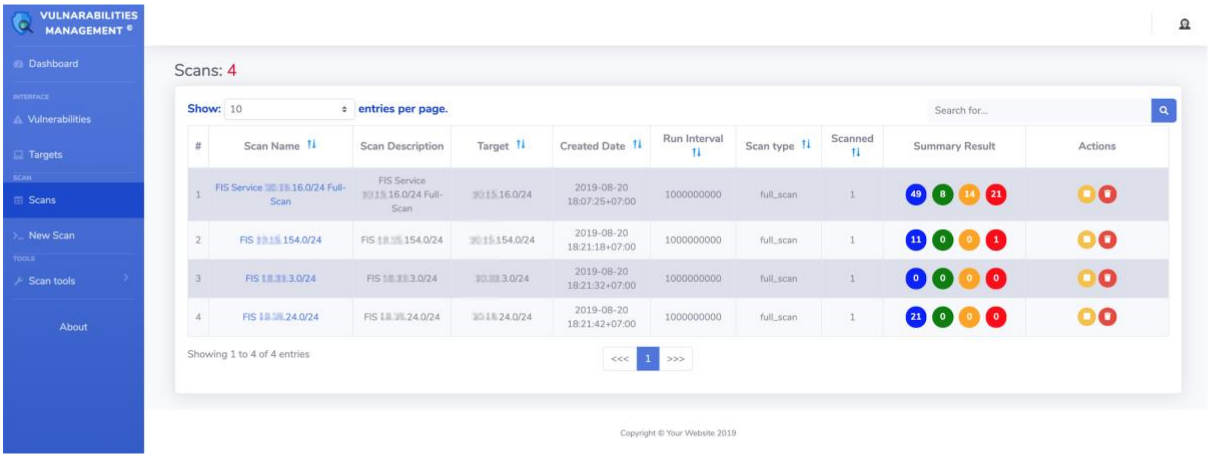
. [online] Available at:

https://www.icheme.org/media/8972/xxiv-poster-07.pdf

[Accessed 28 July. 2019].

Appendix B – Scan Result.

B.1. Scan at FIS (FPT Information System)



Scans: 4

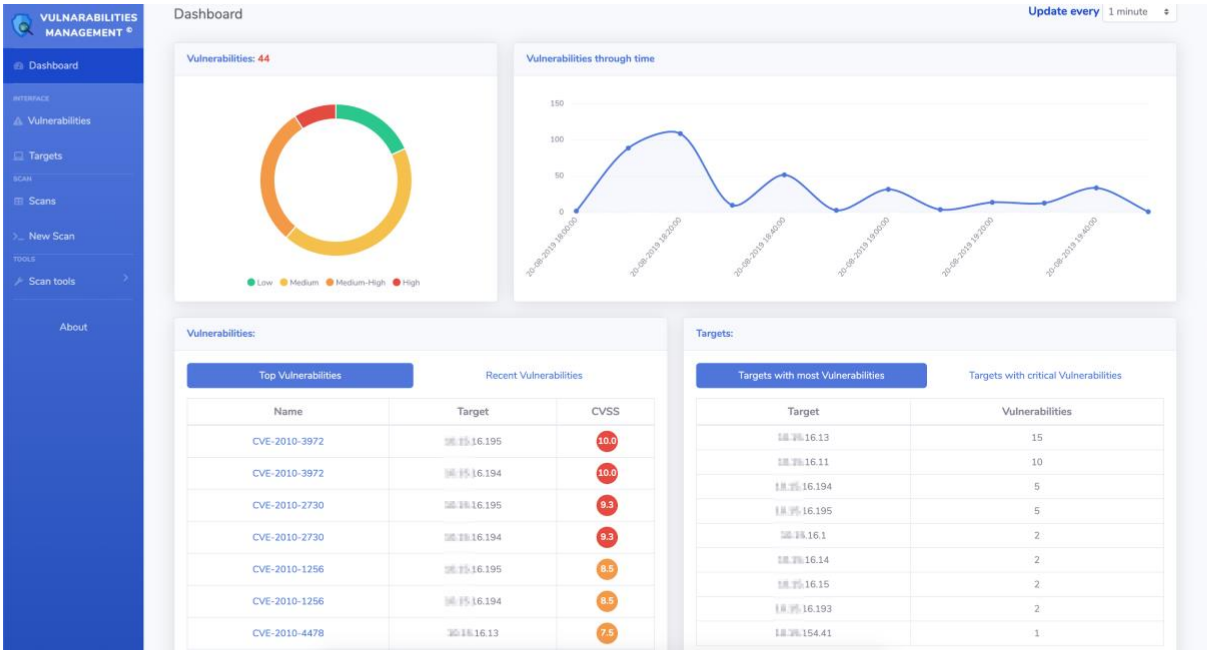
Show: 10 entries per page.

#	Scan Name	Scan Description	Target	Created Date	Run Interval	Scan type	Scanned	Summary Result	Actions
1	FIS Service [redacted].16.0/24 Full-Scan	FIS Service [redacted].16.0/24 Full-Scan	[redacted].16.0/24	2019-08-20 18:07:25+07:00	1000000000	full_scan	1	49 8 14 21	
2	FIS [redacted].154.0/24	FIS [redacted].154.0/24	[redacted].154.0/24	2019-08-20 18:21:18+07:00	1000000000	full_scan	1	11 0 0 1	
3	FIS [redacted].3.0/24	FIS [redacted].3.0/24	[redacted].3.0/24	2019-08-20 18:21:32+07:00	1000000000	full_scan	1	0 0 0 0	
4	FIS [redacted].24.0/24	FIS [redacted].24.0/24	[redacted].24.0/24	2019-08-20 18:21:42+07:00	1000000000	full_scan	1	21 0 0 0	

Showing 1 to 4 of 4 entries

Copyright © Your Website 2019

We create four scans and run on four different subnet



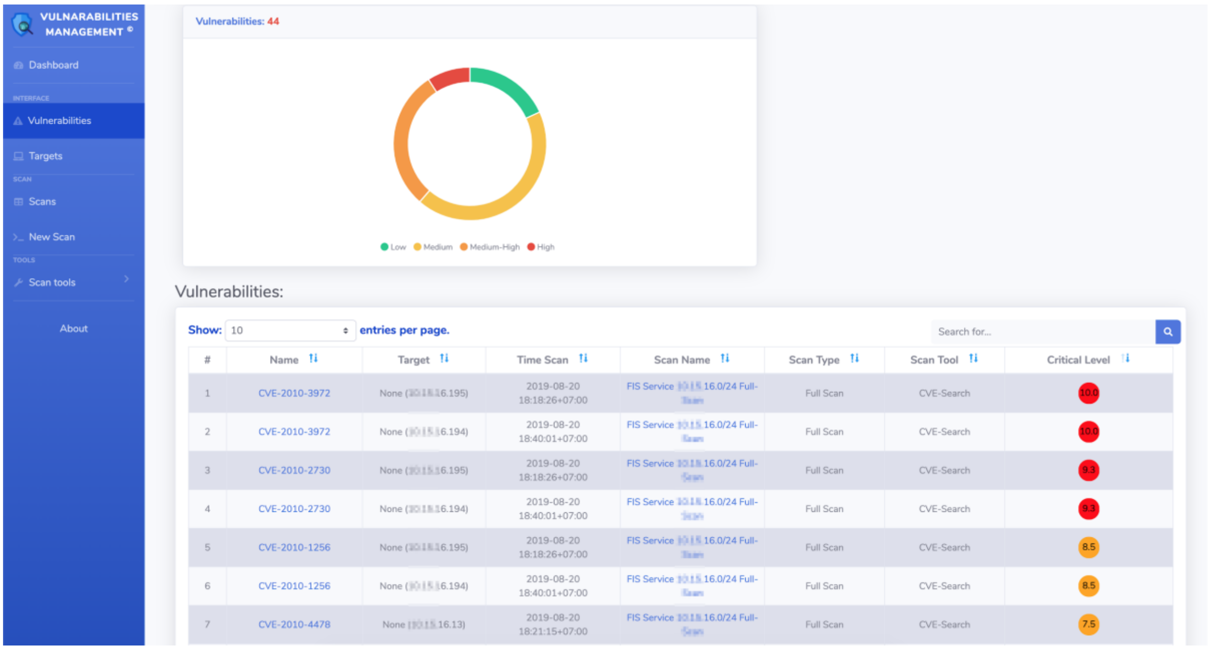
Dashboard

Update every 1 minute

Name	Target	CVSS
CVE-2010-3972	[redacted].16.195	10.0
CVE-2010-3972	[redacted].16.194	10.0
CVE-2010-2730	[redacted].16.195	9.3
CVE-2010-2730	[redacted].16.194	9.3
CVE-2010-1256	[redacted].16.195	8.5
CVE-2010-1256	[redacted].16.194	8.5
CVE-2010-4478	[redacted].16.13	7.5

Target	Vulnerabilities
[redacted].16.13	15
[redacted].16.11	10
[redacted].16.194	5
[redacted].16.195	5
[redacted].16.1	2
[redacted].16.14	2
[redacted].16.15	2
[redacted].16.193	2
[redacted].154.41	1

Our Dashboard show the analysis information about the system of FIS.



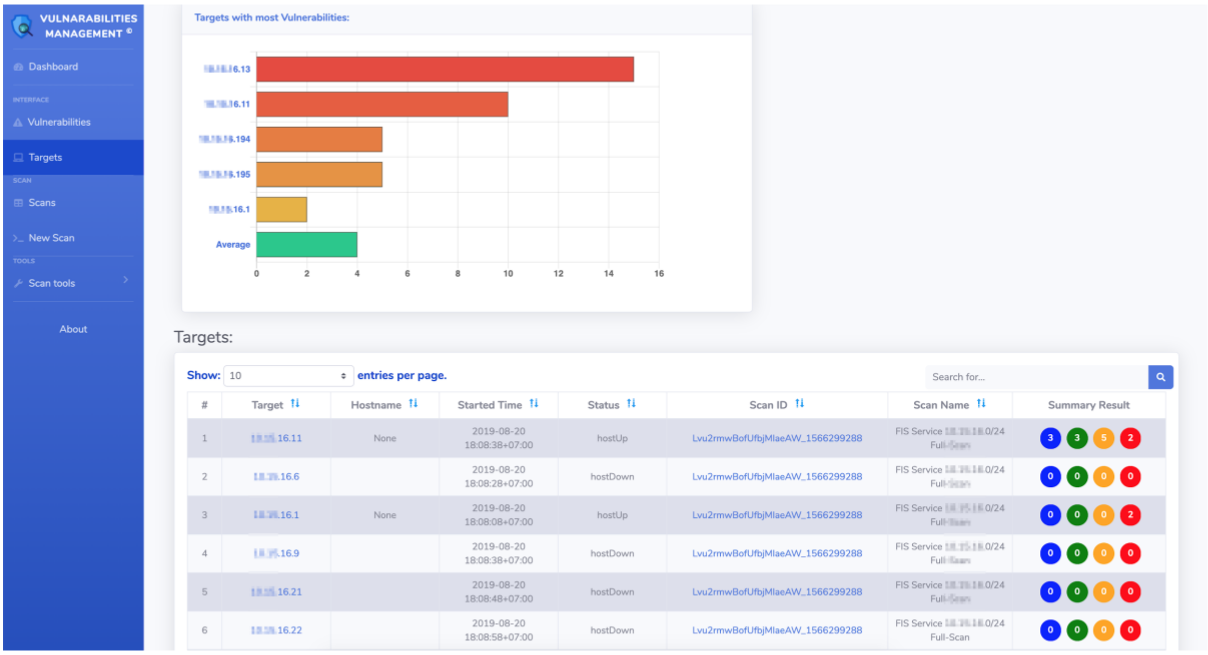
Vulnerabilities:

Show: 10 entries per page.

Search for...

#	Name	Target	Time Scan	Scan Name	Scan Type	Scan Tool	Critical Level
1	CVE-2010-3972	None ([redacted]6.195)	2019-08-20 18:18:26+07:00	FIS Service [redacted] 16.0/24 Full-[redacted]	Full Scan	CVE-Search	10.0
2	CVE-2010-3972	None ([redacted]6.194)	2019-08-20 18:40:01+07:00	FIS Service [redacted] 16.0/24 Full-[redacted]	Full Scan	CVE-Search	10.0
3	CVE-2010-2730	None ([redacted]6.195)	2019-08-20 18:18:26+07:00	FIS Service [redacted] 16.0/24 Full-[redacted]	Full Scan	CVE-Search	9.3
4	CVE-2010-2730	None ([redacted]6.194)	2019-08-20 18:40:01+07:00	FIS Service [redacted] 16.0/24 Full-[redacted]	Full Scan	CVE-Search	9.3
5	CVE-2010-1256	None ([redacted]6.195)	2019-08-20 18:18:26+07:00	FIS Service [redacted] 16.0/24 Full-[redacted]	Full Scan	CVE-Search	8.5
6	CVE-2010-1256	None ([redacted]6.194)	2019-08-20 18:40:01+07:00	FIS Service [redacted] 16.0/24 Full-[redacted]	Full Scan	CVE-Search	8.5
7	CVE-2010-4478	None [redacted]16.13)	2019-08-20 18:21:15+07:00	FIS Service [redacted] 16.0/24 Full-[redacted]	Full Scan	CVE-Search	7.5

This is vulnerability page of the system.



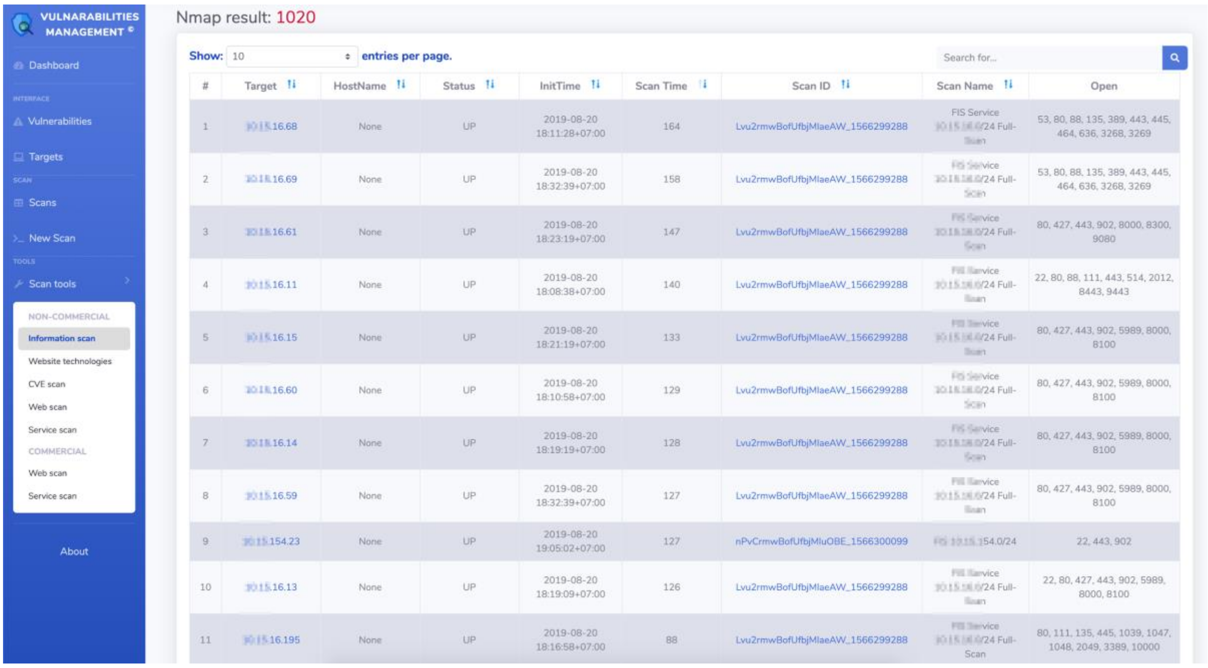
Targets:

Show: 10 entries per page.

Search for...

#	Target	Hostname	Started Time	Status	Scan ID	Scan Name	Summary Result
1	[redacted].16.11	None	2019-08-20 18:08:38+07:00	hostUp	Lvu2rmwBofUfbjMlaeAW_1566299288	FIS Service [redacted]0/24 Full-[redacted]	3 3 5 2
2	[redacted].16.6		2019-08-20 18:08:28+07:00	hostDown	Lvu2rmwBofUfbjMlaeAW_1566299288	FIS Service [redacted]0/24 Full-[redacted]	0 0 0 0
3	[redacted].16.1	None	2019-08-20 18:08:08+07:00	hostUp	Lvu2rmwBofUfbjMlaeAW_1566299288	FIS Service [redacted]0/24 Full-[redacted]	0 0 0 2
4	[redacted].16.9		2019-08-20 18:08:38+07:00	hostDown	Lvu2rmwBofUfbjMlaeAW_1566299288	FIS Service [redacted]0/24 Full-[redacted]	0 0 0 0
5	[redacted].16.21		2019-08-20 18:08:48+07:00	hostDown	Lvu2rmwBofUfbjMlaeAW_1566299288	FIS Service [redacted]0/24 Full-[redacted]	0 0 0 0
6	[redacted].16.22		2019-08-20 18:08:58+07:00	hostDown	Lvu2rmwBofUfbjMlaeAW_1566299288	FIS Service [redacted]0/24 Full-Scan	0 0 0 0

This page show information about targets

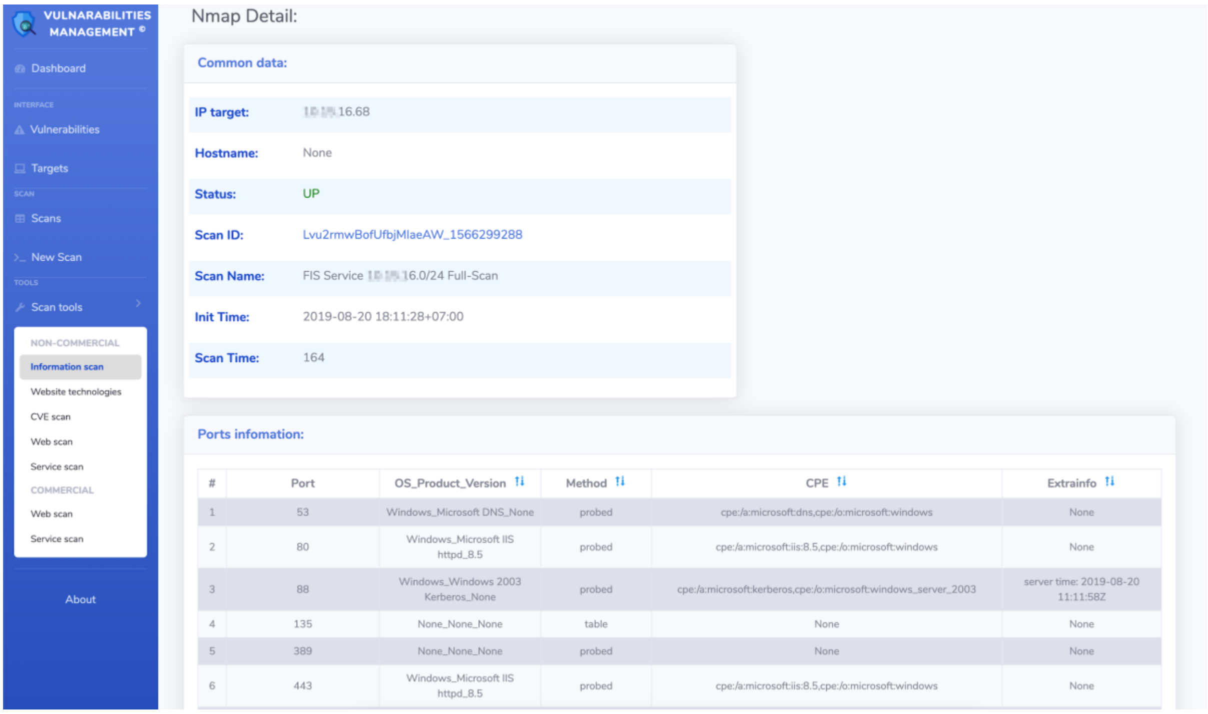


Nmap result: 1020

Show: 10 entries per page.

#	Target	HostName	Status	InitTime	Scan Time	Scan ID	Scan Name	Open
1	[redacted]16.68	None	UP	2019-08-20 18:11:28+07:00	164	Lvu2rmwBofUfbjMlaeAW_1566299288	FIS Service [redacted]/24 Full-[redacted]	53, 80, 88, 135, 389, 443, 445, 464, 636, 3268, 3269
2	[redacted]16.69	None	UP	2019-08-20 18:32:39+07:00	158	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]vice [redacted]/24 Full-[redacted]	53, 80, 88, 135, 389, 443, 445, 464, 636, 3268, 3269
3	[redacted]16.61	None	UP	2019-08-20 18:23:19+07:00	147	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]vice [redacted]/24 Full-[redacted]	80, 427, 443, 902, 8000, 8300, 9080
4	[redacted]16.11	None	UP	2019-08-20 18:08:38+07:00	140	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]vice [redacted]/24 Full-[redacted]	22, 80, 88, 111, 443, 514, 2012, 8443, 9443
5	[redacted]16.15	None	UP	2019-08-20 18:21:19+07:00	133	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]vice [redacted]/24 Full-[redacted]	80, 427, 443, 902, 5989, 8000, 8100
6	[redacted]16.60	None	UP	2019-08-20 18:10:58+07:00	129	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]vice [redacted]/24 Full-[redacted]	80, 427, 443, 902, 5989, 8000, 8100
7	[redacted]16.14	None	UP	2019-08-20 18:19:19+07:00	128	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]vice [redacted]/24 Full-[redacted]	80, 427, 443, 902, 5989, 8000, 8100
8	[redacted]16.59	None	UP	2019-08-20 18:32:39+07:00	127	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]vice [redacted]/24 Full-[redacted]	80, 427, 443, 902, 5989, 8000, 8100
9	[redacted]154.23	None	UP	2019-08-20 19:05:02+07:00	127	nPvCrmwBofUfbjMluOBE_1566300099	[redacted]54.0/24	22, 443, 902
10	[redacted]16.13	None	UP	2019-08-20 18:19:09+07:00	126	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]vice [redacted]/24 Full-[redacted]	22, 80, 427, 443, 902, 5989, 8000, 8100
11	[redacted]16.195	None	UP	2019-08-20 18:16:58+07:00	88	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]vice [redacted]/24 Full-Scan	80, 111, 135, 445, 1039, 1047, 1048, 2049, 3389, 10000

Information page, which is nmap result.



Nmap Detail:

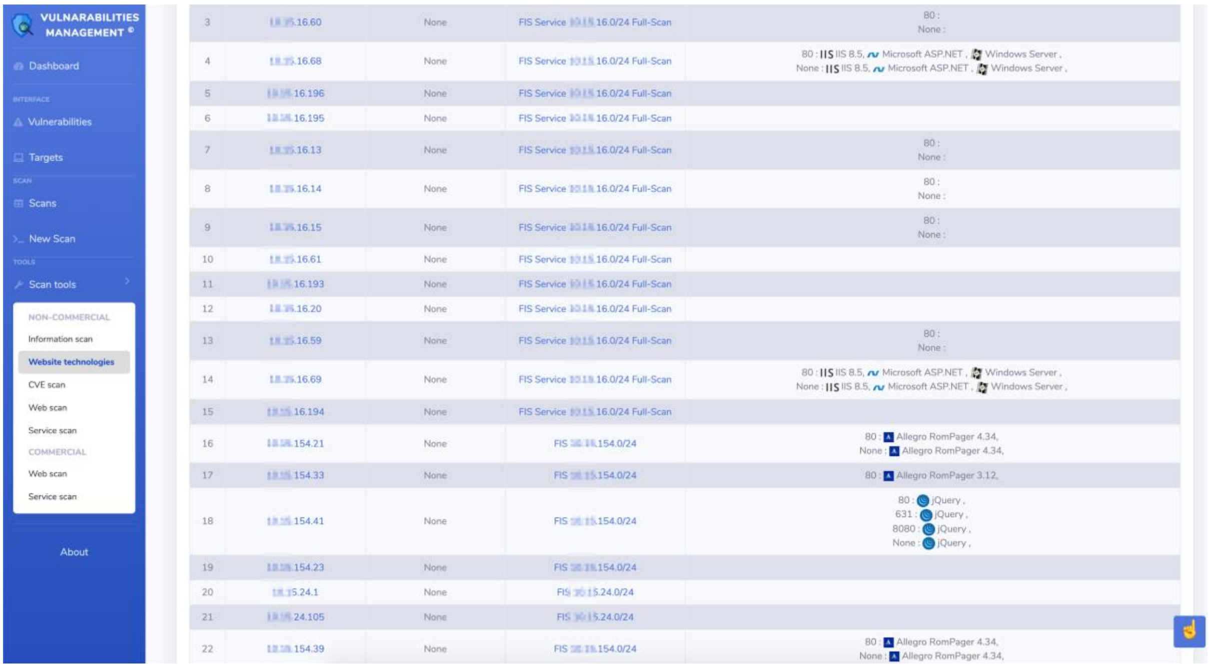
Common data:

IP target:	[redacted]16.68
Hostname:	None
Status:	UP
Scan ID:	Lvu2rmwBofUfbjMlaeAW_1566299288
Scan Name:	FIS Service [redacted]6.0/24 Full-Scan
Init Time:	2019-08-20 18:11:28+07:00
Scan Time:	164

Ports infomation:

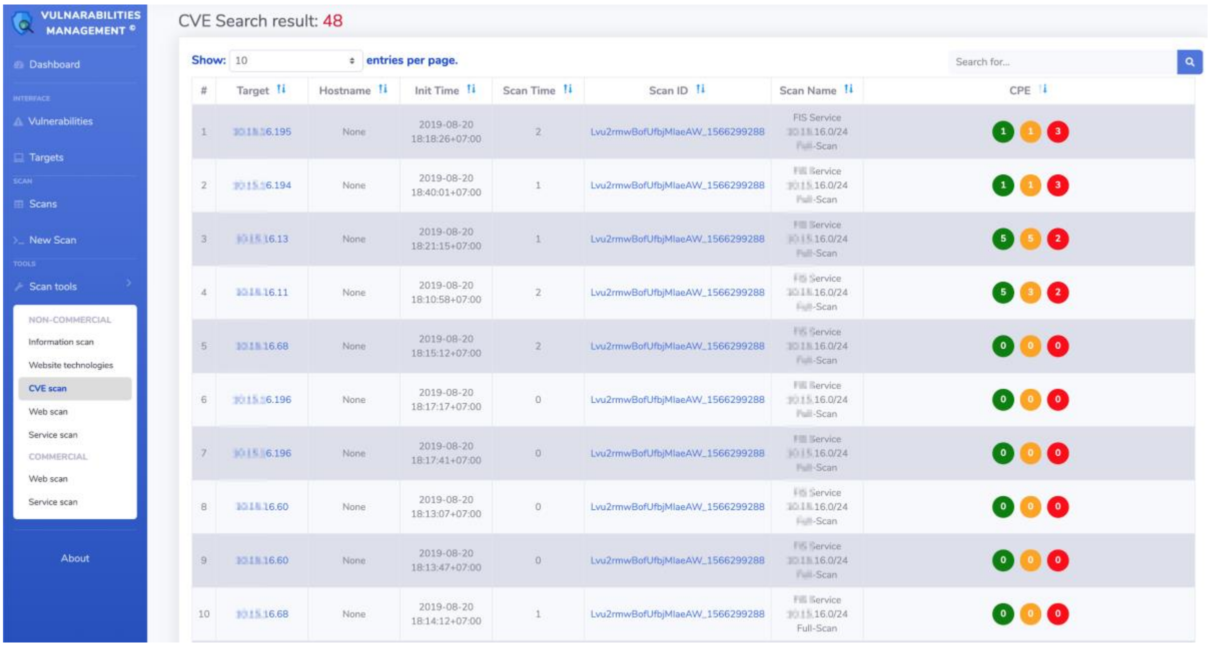
#	Port	OS_Product_Version	Method	CPE	Extrainfo
1	53	Windows_Microsoft DNS_None	probed	cpe:/a:microsoft:dns,cpe:/o:microsoft:windows	None
2	80	Windows_Microsoft IIS httpd_8.5	probed	cpe:/a:microsoft:iis:8.5,cpe:/o:microsoft:windows	None
3	88	Windows_Windows 2003 Kerberos_None	probed	cpe:/a:microsoft:kerberos,cpe:/o:microsoft:windows_server_2003	server time: 2019-08-20 11:11:58Z
4	135	None_None_None	table	None	None
5	389	None_None_None	probed	None	None
6	443	Windows_Microsoft IIS httpd_8.5	probed	cpe:/a:microsoft:iis:8.5,cpe:/o:microsoft:windows	None

Detail information scan of one target.



3	[redacted].16.60	None	FIS Service [redacted]16.0/24 Full-Scan	80 : None :
4	[redacted].16.68	None	FIS Service [redacted]16.0/24 Full-Scan	80 : IIS IIS 8.5, Microsoft ASP.NET , Windows Server , None : IIS IIS 8.5, Microsoft ASP.NET , Windows Server ,
5	[redacted]16.196	None	FIS Service [redacted]16.0/24 Full-Scan	
6	[redacted]16.195	None	FIS Service [redacted]16.0/24 Full-Scan	
7	[redacted].16.13	None	FIS Service [redacted]16.0/24 Full-Scan	80 : None :
8	[redacted].16.14	None	FIS Service [redacted]16.0/24 Full-Scan	80 : None :
9	[redacted].16.15	None	FIS Service [redacted]16.0/24 Full-Scan	80 : None :
10	[redacted].16.61	None	FIS Service [redacted]16.0/24 Full-Scan	
11	[redacted]16.193	None	FIS Service [redacted]16.0/24 Full-Scan	
12	[redacted].16.20	None	FIS Service [redacted]16.0/24 Full-Scan	
13	[redacted].16.59	None	FIS Service [redacted]16.0/24 Full-Scan	80 : None :
14	[redacted].16.69	None	FIS Service [redacted]16.0/24 Full-Scan	80 : IIS IIS 8.5, Microsoft ASP.NET , Windows Server , None : IIS IIS 8.5, Microsoft ASP.NET , Windows Server ,
15	[redacted]16.194	None	FIS Service [redacted]16.0/24 Full-Scan	
16	[redacted]154.21	None	FIS [redacted]154.0/24	80 : Allegro RomPager 4.34, None : Allegro RomPager 4.34,
17	[redacted]154.33	None	FIS [redacted]154.0/24	80 : Allegro RomPager 3.12,
18	[redacted]154.41	None	FIS [redacted]154.0/24	80 : jQuery , 631 : jQuery , 8080 : jQuery , None : jQuery ,
19	[redacted]154.23	None	FIS [redacted]154.0/24	
20	[redacted]5.24.1	None	FIS [redacted]5.24.0/24	
21	[redacted]24.105	None	FIS [redacted]5.24.0/24	
22	[redacted]154.39	None	FIS [redacted]154.0/24	80 : Allegro RomPager 4.34, None : Allegro RomPager 4.34,

This page displays the overview about the technologies used in targets.

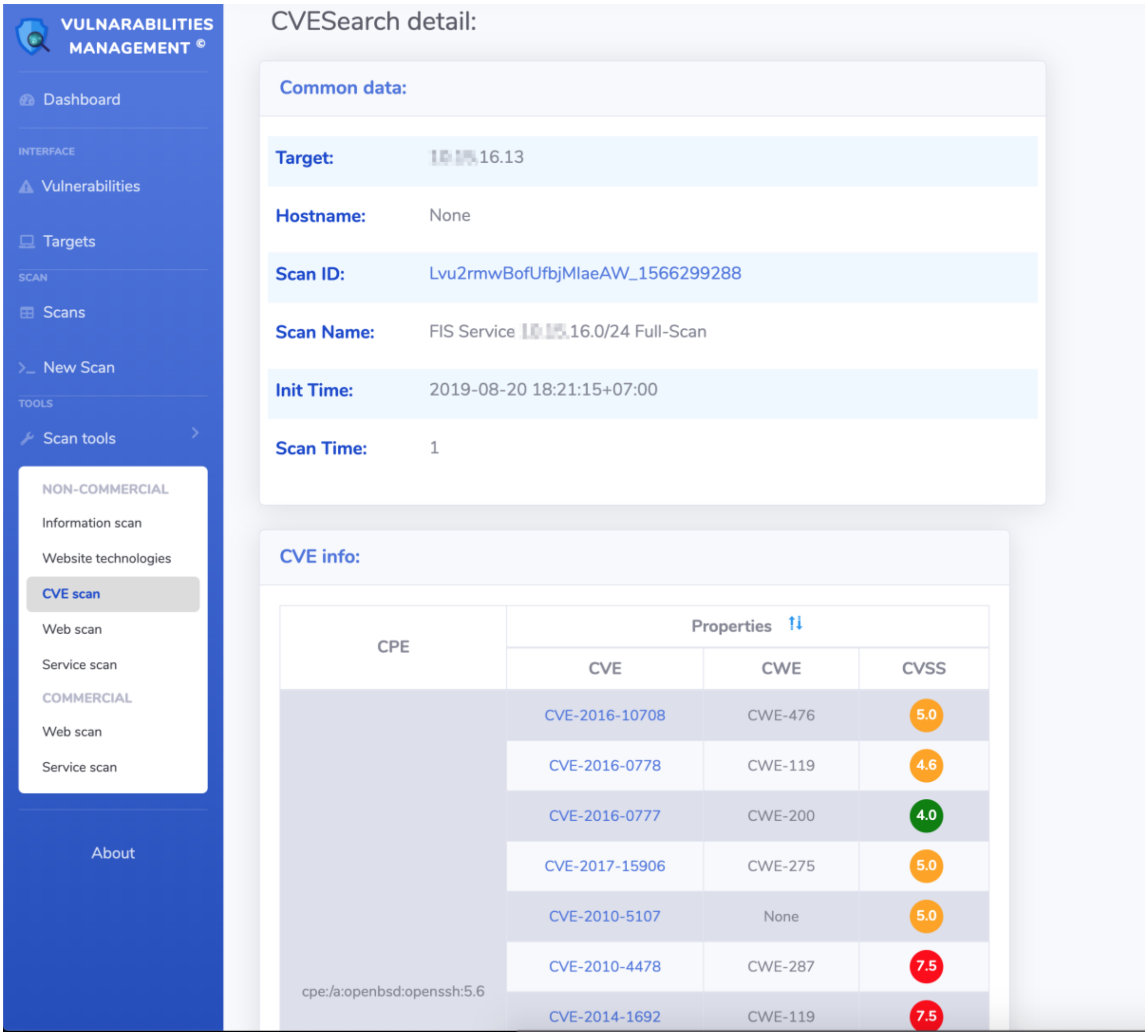


CVE Search result: 48

Show: 10 entries per page.

#	Target	Hostname	Init Time	Scan Time	Scan ID	Scan Name	CPE
1	[redacted]6.195	None	2019-08-20 18:18:26+07:00	2	Lvu2rmwBofUfbjMlaeAW_1566299288	FIS Service [redacted]16.0/24 [redacted]-Scan	1 1 3
2	[redacted]6.194	None	2019-08-20 18:40:01+07:00	1	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]Service [redacted]16.0/24 [redacted]-Scan	1 1 3
3	[redacted]16.13	None	2019-08-20 18:21:15+07:00	1	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]Service [redacted]16.0/24 [redacted]-Scan	5 5 2
4	[redacted]16.11	None	2019-08-20 18:10:58+07:00	2	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]Service [redacted]16.0/24 [redacted]-Scan	5 3 2
5	[redacted]16.68	None	2019-08-20 18:15:12+07:00	2	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]Service [redacted]16.0/24 [redacted]-Scan	0 0 0
6	[redacted]6.196	None	2019-08-20 18:17:17+07:00	0	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]Service [redacted]16.0/24 [redacted]-Scan	0 0 0
7	[redacted]6.196	None	2019-08-20 18:17:41+07:00	0	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]Service [redacted]16.0/24 [redacted]-Scan	0 0 0
8	[redacted]16.60	None	2019-08-20 18:13:07+07:00	0	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]Service [redacted]16.0/24 [redacted]-Scan	0 0 0
9	[redacted]16.60	None	2019-08-20 18:13:47+07:00	0	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]Service [redacted]16.0/24 [redacted]-Scan	0 0 0
10	[redacted]16.68	None	2019-08-20 18:14:12+07:00	1	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]Service [redacted]16.0/24 Full-Scan	0 0 0

This page displays the vulnerabilities of target, which is scanned by CVE-Search



CVESearch detail:

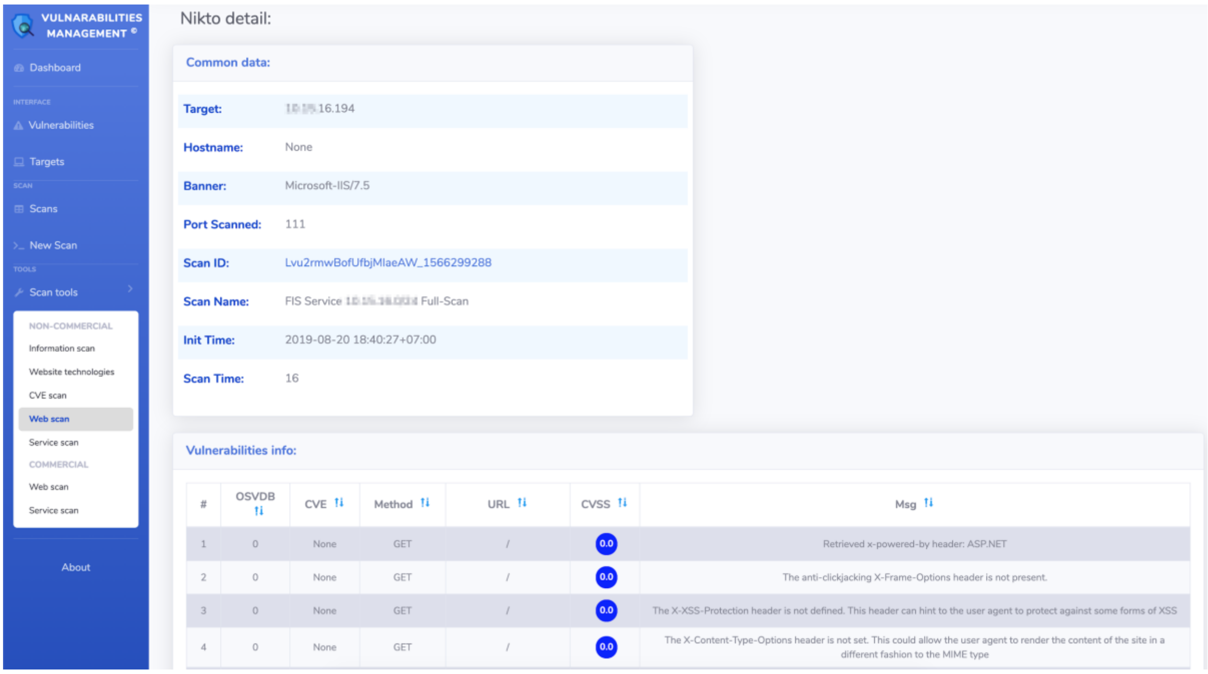
Common data:

Target:	[redacted].16.13
Hostname:	None
Scan ID:	Lvu2rmwBofUfbjMlaeAW_1566299288
Scan Name:	FIS Service [redacted].16.0/24 Full-Scan
Init Time:	2019-08-20 18:21:15+07:00
Scan Time:	1

CVE info:

CPE	Properties		
	CVE	CWE	CVSS
cpe:/a:openbsd:openssh:5.6	CVE-2016-10708	CWE-476	5.0
	CVE-2016-0778	CWE-119	4.6
	CVE-2016-0777	CWE-200	4.0
	CVE-2017-15906	CWE-275	5.0
	CVE-2010-5107	None	5.0
	CVE-2010-4478	CWE-287	7.5
	CVE-2014-1692	CWE-119	7.5

This page show the detail information of a target, which is scanned by CVE Search.



Nikto detail:

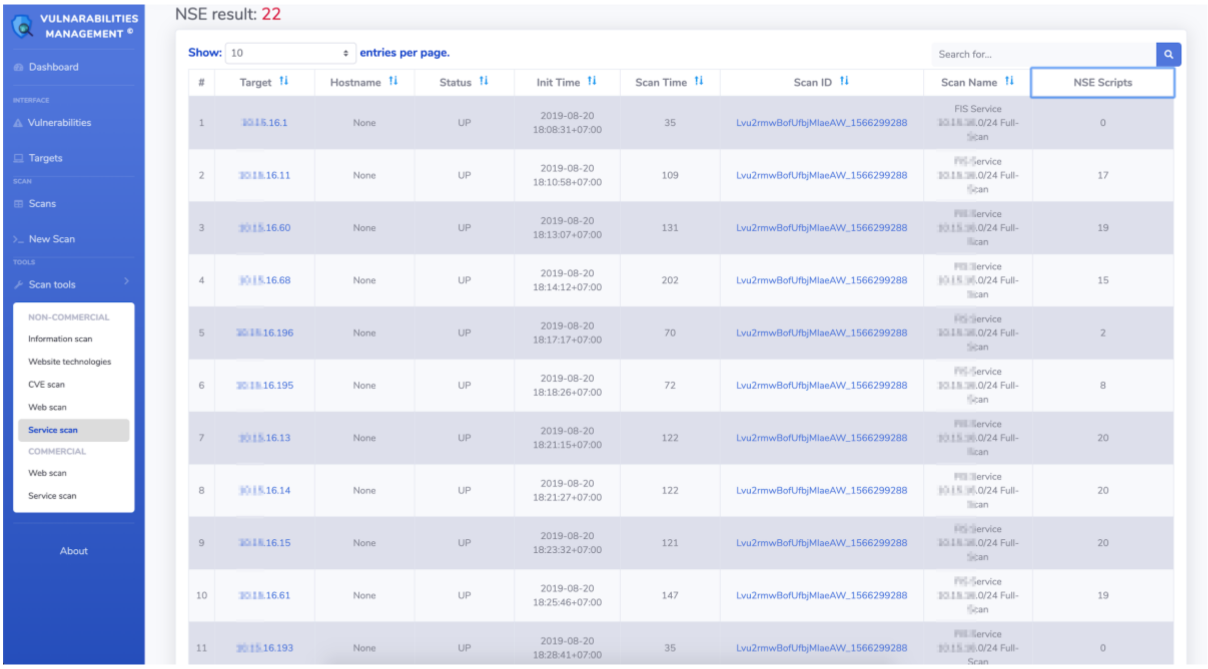
Common data:

Target:	[redacted].16.194
Hostname:	None
Banner:	Microsoft-IIS/7.5
Port Scanned:	111
Scan ID:	Lvu2rmwBofUfbjMlaeAW_1566299288
Scan Name:	FIS Service [redacted] Full-Scan
Init Time:	2019-08-20 18:40:27+07:00
Scan Time:	16

Vulnerabilities info:

#	OSVDB	CVE	Method	URL	CVSS	Msg
1	0	None	GET	/	0.0	Retrieved x-powered-by header: ASP.NET
2	0	None	GET	/	0.0	The anti-clickjacking X-Frame-Options header is not present.
3	0	None	GET	/	0.0	The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
4	0	None	GET	/	0.0	The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type

This page show detail for a target which is scanned by Nikto.

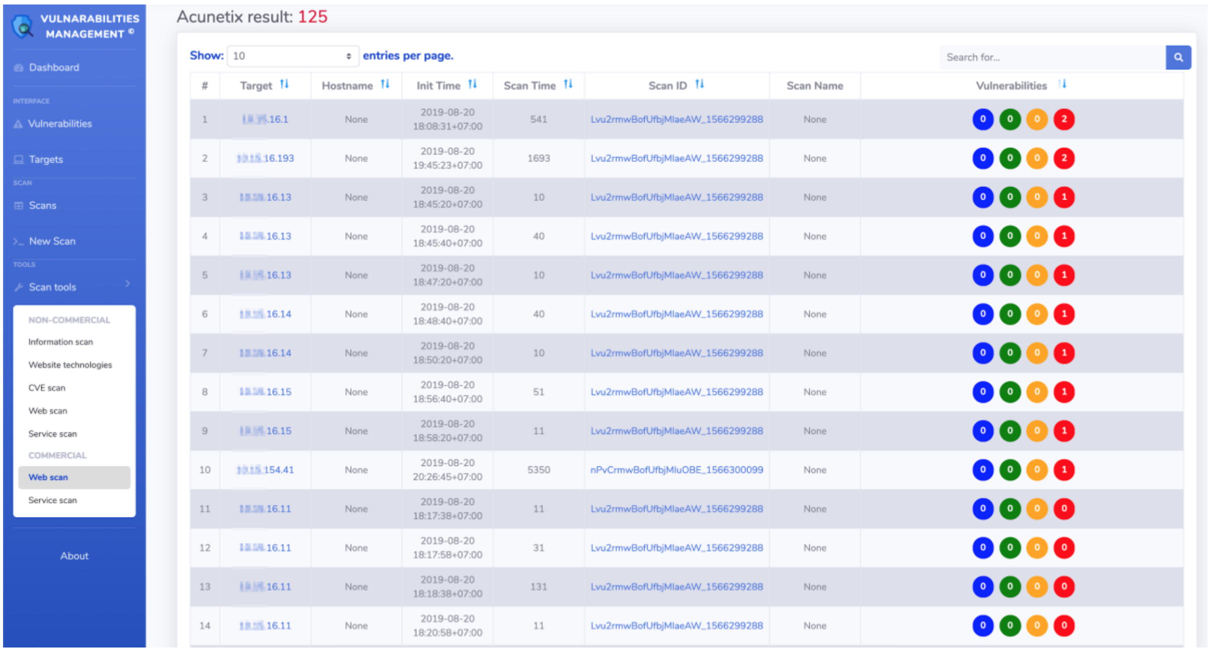


NSE result: 22

Show: 10 entries per page.

#	Target	Hostname	Status	Init Time	Scan Time	Scan ID	Scan Name	NSE Scripts
1	[redacted].16.1	None	UP	2019-08-20 18:08:31+07:00	35	Lvu2rmwBofUfbjMlaeAW_1566299288	FIS Service [redacted].0/24 Full-[redacted]can	0
2	[redacted].16.11	None	UP	2019-08-20 18:10:58+07:00	109	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]ervice [redacted].0/24 Full-[redacted]can	17
3	[redacted].16.60	None	UP	2019-08-20 18:13:07+07:00	131	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]ervice [redacted].0/24 Full-[redacted]can	19
4	[redacted].16.68	None	UP	2019-08-20 18:14:12+07:00	202	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]ervice [redacted].0/24 Full-[redacted]can	15
5	[redacted].16.196	None	UP	2019-08-20 18:17:17+07:00	70	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]ervice [redacted].0/24 Full-[redacted]can	2
6	[redacted].16.195	None	UP	2019-08-20 18:18:26+07:00	72	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]ervice [redacted].0/24 Full-[redacted]can	8
7	[redacted].16.13	None	UP	2019-08-20 18:21:15+07:00	122	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]ervice [redacted].0/24 Full-[redacted]can	20
8	[redacted].16.14	None	UP	2019-08-20 18:21:27+07:00	122	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]ervice [redacted].0/24 Full-[redacted]can	20
9	[redacted].16.15	None	UP	2019-08-20 18:23:32+07:00	121	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]ervice [redacted].0/24 Full-[redacted]can	20
10	[redacted].16.61	None	UP	2019-08-20 18:25:46+07:00	147	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]ervice [redacted].0/24 Full-[redacted]can	19
11	[redacted].16.193	None	UP	2019-08-20 18:28:41+07:00	35	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]ervice [redacted].0/24 Full-Scan	0

This page shows information for target which is scanned by NSE.

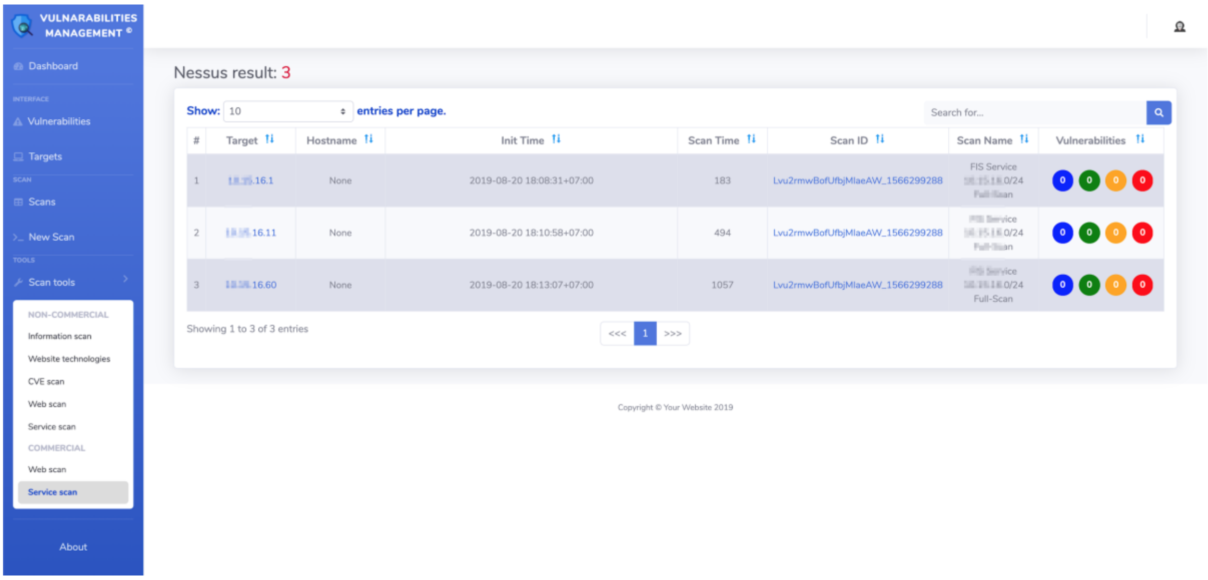


Acunetix result: 125

Show: 10 entries per page.

#	Target	Hostname	Init Time	Scan Time	Scan ID	Scan Name	Vulnerabilities
1	[redacted].16.1	None	2019-08-20 18:08:31+07:00	541	Lvu2rmwBofUfbjMlaeAW_1566299288	None	0 0 0 2
2	[redacted].16.193	None	2019-08-20 19:45:23+07:00	1693	Lvu2rmwBofUfbjMlaeAW_1566299288	None	0 0 0 2
3	[redacted].16.13	None	2019-08-20 18:45:20+07:00	10	Lvu2rmwBofUfbjMlaeAW_1566299288	None	0 0 0 1
4	[redacted].16.13	None	2019-08-20 18:45:40+07:00	40	Lvu2rmwBofUfbjMlaeAW_1566299288	None	0 0 0 1
5	[redacted].16.13	None	2019-08-20 18:47:20+07:00	10	Lvu2rmwBofUfbjMlaeAW_1566299288	None	0 0 0 1
6	[redacted].16.14	None	2019-08-20 18:48:40+07:00	40	Lvu2rmwBofUfbjMlaeAW_1566299288	None	0 0 0 1
7	[redacted].16.14	None	2019-08-20 18:50:20+07:00	10	Lvu2rmwBofUfbjMlaeAW_1566299288	None	0 0 0 1
8	[redacted].16.15	None	2019-08-20 18:56:40+07:00	51	Lvu2rmwBofUfbjMlaeAW_1566299288	None	0 0 0 1
9	[redacted].16.15	None	2019-08-20 18:58:20+07:00	11	Lvu2rmwBofUfbjMlaeAW_1566299288	None	0 0 0 1
10	[redacted].154.41	None	2019-08-20 20:26:45+07:00	5350	nPvCrmwBofUfbjMluOBE_1566300099	None	0 0 0 1
11	[redacted].16.11	None	2019-08-20 18:17:38+07:00	11	Lvu2rmwBofUfbjMlaeAW_1566299288	None	0 0 0 0
12	[redacted].16.11	None	2019-08-20 18:17:58+07:00	31	Lvu2rmwBofUfbjMlaeAW_1566299288	None	0 0 0 0
13	[redacted].16.11	None	2019-08-20 18:18:38+07:00	131	Lvu2rmwBofUfbjMlaeAW_1566299288	None	0 0 0 0
14	[redacted].16.11	None	2019-08-20 18:20:58+07:00	11	Lvu2rmwBofUfbjMlaeAW_1566299288	None	0 0 0 0

This page displays information for targets which is scanned by Acunetix.



Nessus result: 3

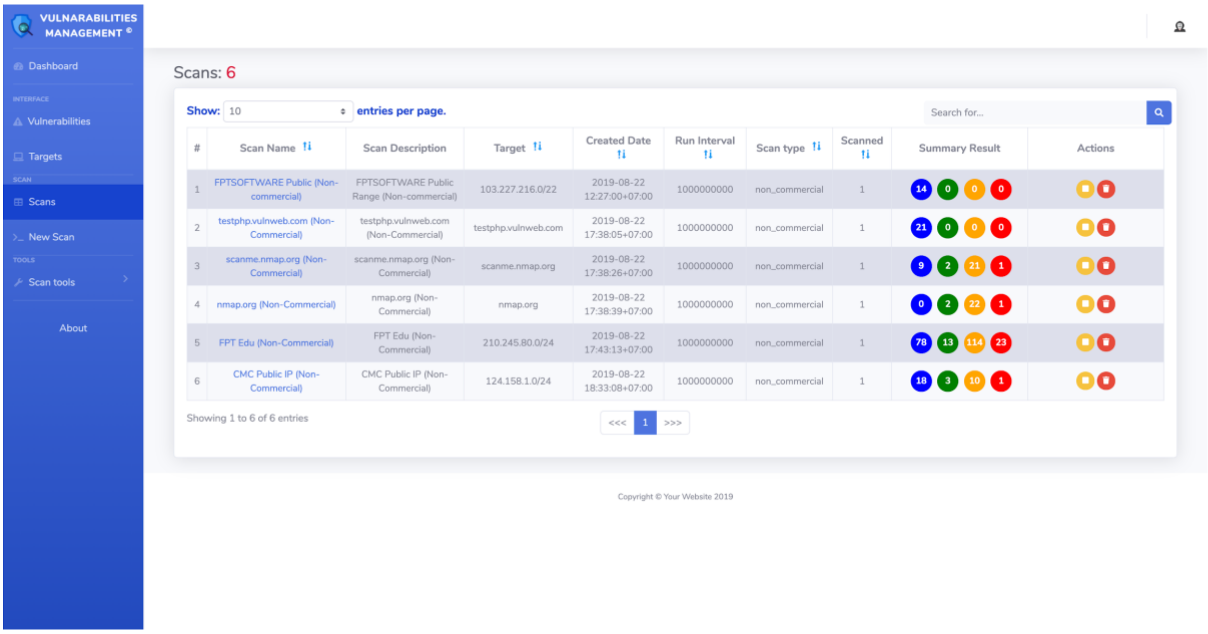
Show: 10 entries per page.

#	Target	Hostname	Init Time	Scan Time	Scan ID	Scan Name	Vulnerabilities
1	[redacted].16.1	None	2019-08-20 18:08:31+07:00	183	Lvu2rmwBofUfbjMlaeAW_1566299288	FIS Service [redacted]0/24 [redacted]an	0 0 0 0
2	[redacted].16.11	None	2019-08-20 18:10:58+07:00	494	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]vice [redacted]0/24 [redacted]an	0 0 0 0
3	[redacted].16.60	None	2019-08-20 18:13:07+07:00	1057	Lvu2rmwBofUfbjMlaeAW_1566299288	[redacted]vice [redacted]0/24 Full-Scan	0 0 0 0

Showing 1 to 3 of 3 entries

This page displays information for targets which is scanned by Nessus.

B.2. Scan Enterprise Public Range



Scans: 6

Show: 10 entries per page.

Search for...

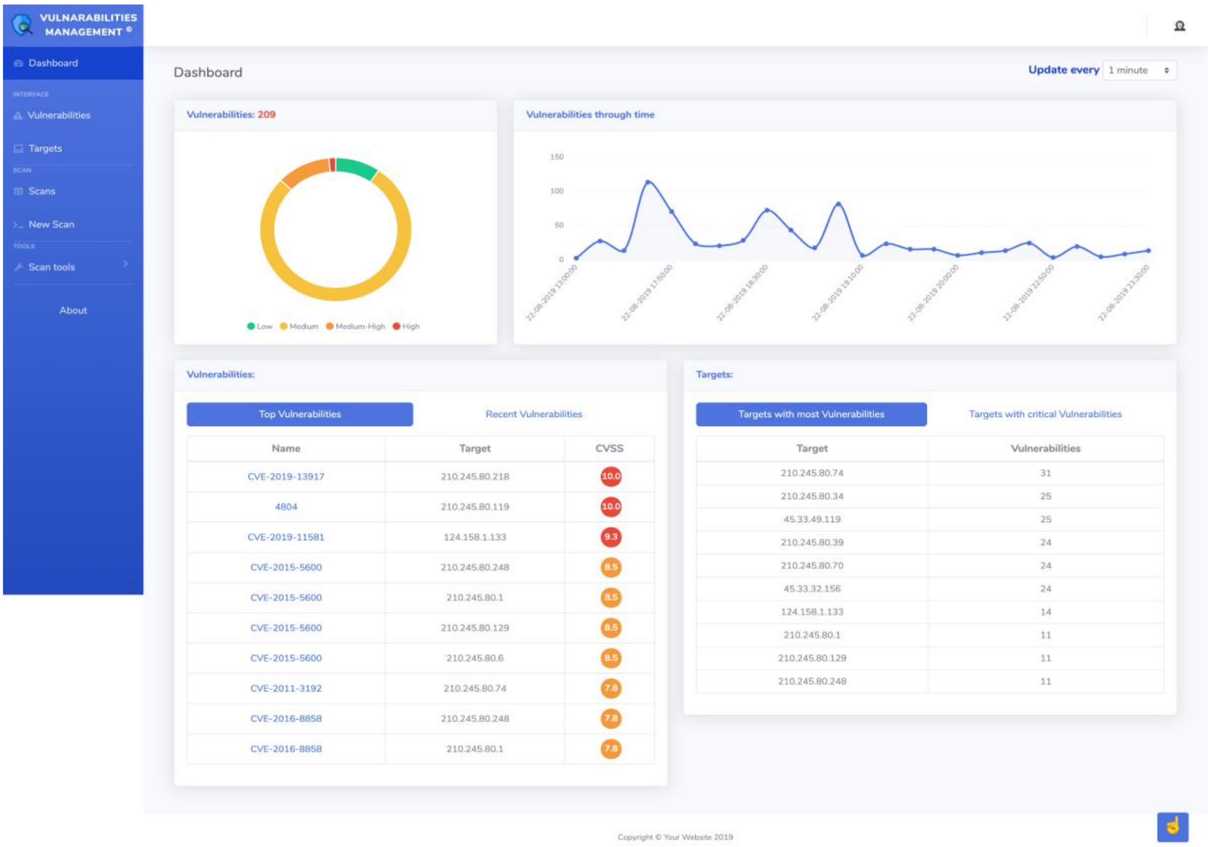
#	Scan Name	Scan Description	Target	Created Date	Run Interval	Scan type	Scanned	Summary Result	Actions
1	FPTSOFTWARE Public (Non-commercial)	FPTSOFTWARE Public Range (Non-commercial)	103.227.216.0/22	2019-08-22 12:27:00+07:00	1000000000	non_commercial	1	14 0 0 0	
2	testphp.vulnweb.com (Non-Commercial)	testphp.vulnweb.com (Non-Commercial)	testphp.vulnweb.com	2019-08-22 17:38:05+07:00	1000000000	non_commercial	1	21 0 0 0	
3	scanme.nmap.org (Non-Commercial)	scanme.nmap.org (Non-Commercial)	scanme.nmap.org	2019-08-22 17:38:26+07:00	1000000000	non_commercial	1	9 2 21 1	
4	nmap.org (Non-Commercial)	nmap.org (Non-Commercial)	nmap.org	2019-08-22 17:38:39+07:00	1000000000	non_commercial	1	0 2 22 1	
5	FPT Edu (Non-Commercial)	FPT Edu (Non-Commercial)	210.245.80.0/24	2019-08-22 17:43:13+07:00	1000000000	non_commercial	1	78 13 114 23	
6	CMC Public IP (Non-Commercial)	CMC Public IP (Non-Commercial)	124.158.1.0/24	2019-08-22 18:33:08+07:00	1000000000	non_commercial	1	18 3 10 1	

Showing 1 to 6 of 6 entries

<<< 1 >>>

Copyright © Your Website 2019

This page displays all targets which we scan. They are **`FPT Edu (Non-Commercial)`** and **`CMC Public IP (Non-Commercial)`**



Dashboard

Update every 1 minute

Vulnerabilities:

Top Vulnerabilities | Recent Vulnerabilities

Name	Target	CVSS
CVE-2019-13917	210.245.80.218	10.0
4804	210.245.80.119	10.0
CVE-2019-11581	124.158.1.133	9.3
CVE-2015-5600	210.245.80.248	8.5
CVE-2015-5600	210.245.80.1	8.5
CVE-2015-5600	210.245.80.129	8.5
CVE-2015-5600	210.245.80.6	8.5
CVE-2011-3192	210.245.80.74	7.8
CVE-2016-8858	210.245.80.248	7.8
CVE-2016-8858	210.245.80.1	7.8

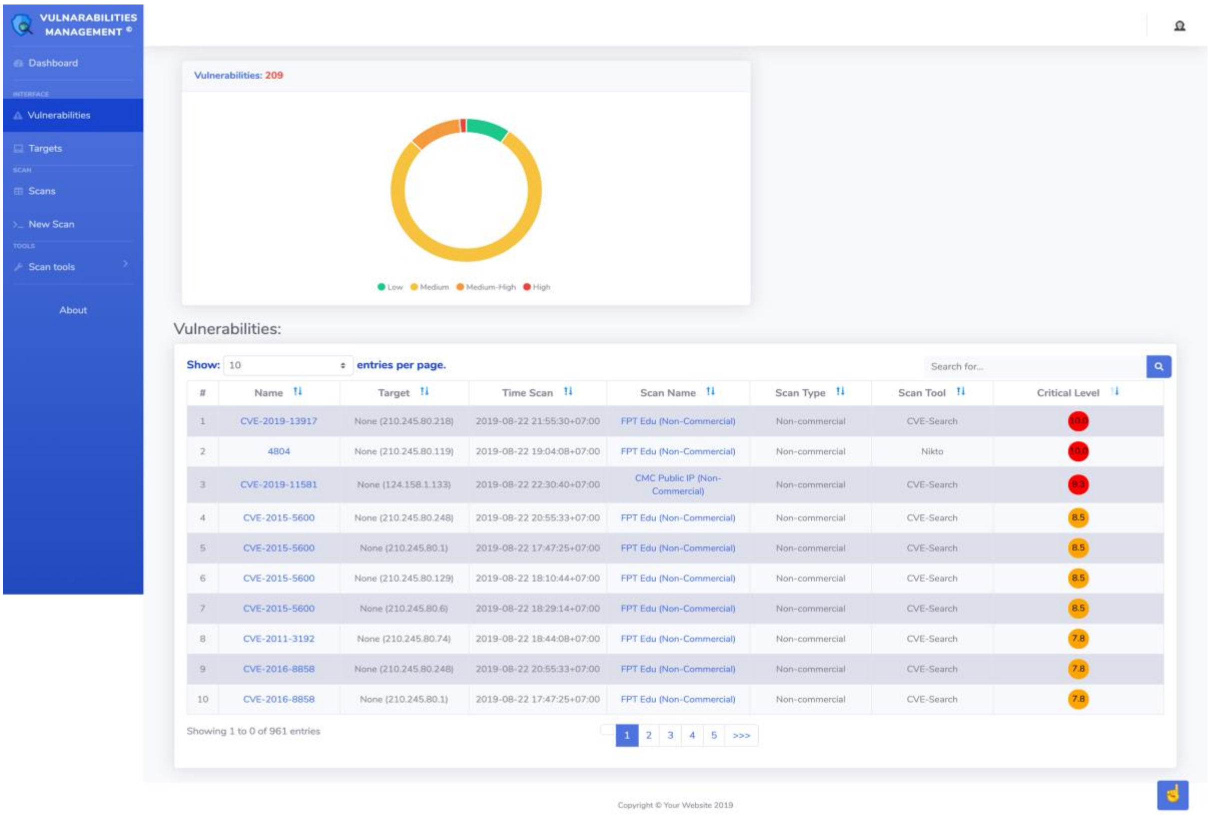
Targets:

Targets with most Vulnerabilities | Targets with critical Vulnerabilities

Target	Vulnerabilities
210.245.80.74	31
210.245.80.34	25
45.33.49.119	25
210.245.80.39	24
210.245.80.70	24
45.33.32.156	24
124.158.1.133	14
210.245.80.1	11
210.245.80.129	11
210.245.80.248	11

Copyright © Your Website 2019

This page displays the dashboard.



Vulnerabilities: 209

Vulnerabilities:

Show: 10 entries per page.

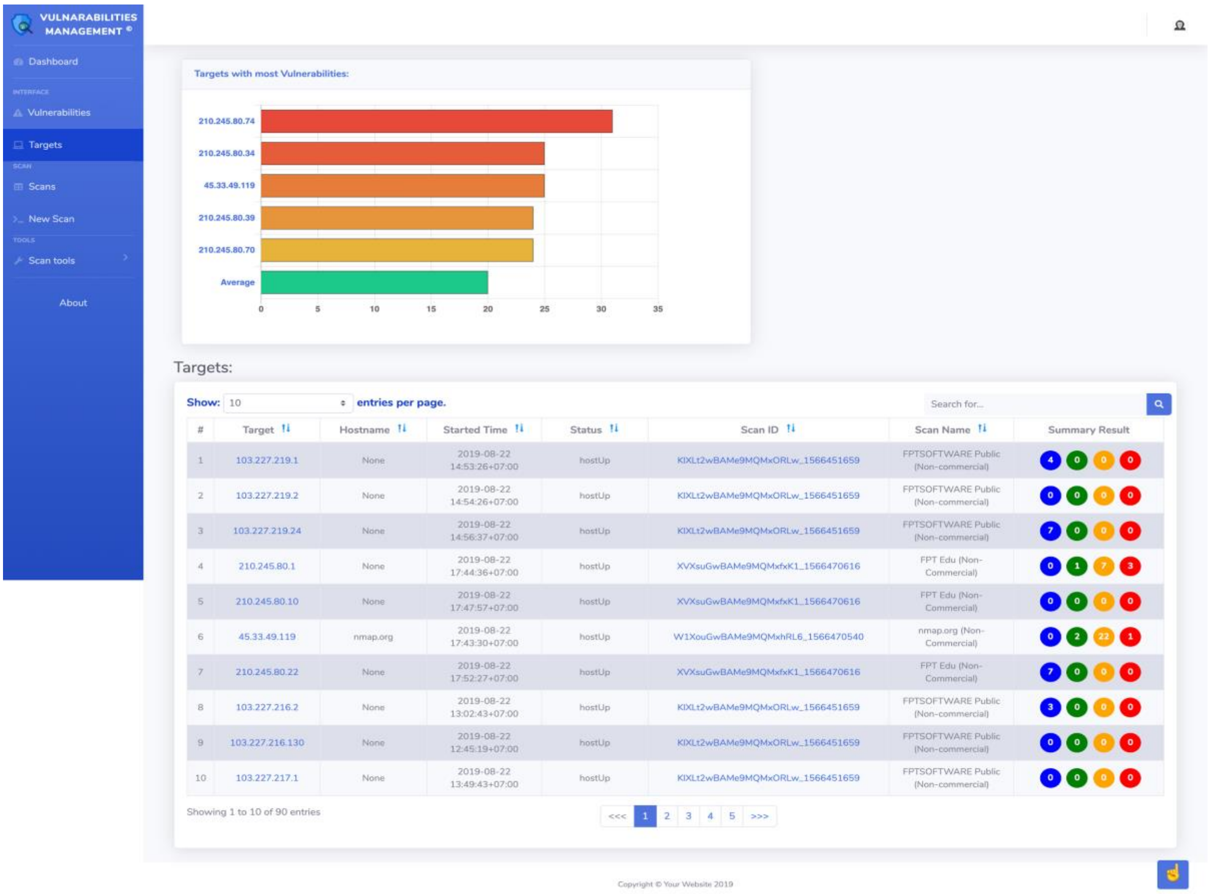
#	Name	Target	Time Scan	Scan Name	Scan Type	Scan Tool	Critical Level
1	CVE-2019-13917	None (210.245.80.218)	2019-08-22 21:55:30+07:00	FPT Edu (Non-Commercial)	Non-commercial	CVE-Search	10.0
2	4804	None (210.245.80.119)	2019-08-22 19:04:08+07:00	FPT Edu (Non-Commercial)	Non-commercial	Nikto	10.0
3	CVE-2019-11581	None (124.158.1.133)	2019-08-22 22:30:40+07:00	CMC Public IP (Non-Commercial)	Non-commercial	CVE-Search	9.3
4	CVE-2015-5600	None (210.245.80.248)	2019-08-22 20:55:33+07:00	FPT Edu (Non-Commercial)	Non-commercial	CVE-Search	8.5
5	CVE-2015-5600	None (210.245.80.1)	2019-08-22 17:47:25+07:00	FPT Edu (Non-Commercial)	Non-commercial	CVE-Search	8.5
6	CVE-2015-5600	None (210.245.80.129)	2019-08-22 18:10:44+07:00	FPT Edu (Non-Commercial)	Non-commercial	CVE-Search	8.5
7	CVE-2015-5600	None (210.245.80.6)	2019-08-22 18:29:14+07:00	FPT Edu (Non-Commercial)	Non-commercial	CVE-Search	8.5
8	CVE-2011-3192	None (210.245.80.74)	2019-08-22 18:44:08+07:00	FPT Edu (Non-Commercial)	Non-commercial	CVE-Search	7.8
9	CVE-2016-8858	None (210.245.80.248)	2019-08-22 20:55:33+07:00	FPT Edu (Non-Commercial)	Non-commercial	CVE-Search	7.8
10	CVE-2016-8858	None (210.245.80.1)	2019-08-22 17:47:25+07:00	FPT Edu (Non-Commercial)	Non-commercial	CVE-Search	7.8

Showing 1 to 0 of 961 entries

1 2 3 4 5 >>>

Copyright © Your Website 2019

This page displays all vulnerabilities.



Targets:

Show: 10 entries per page.

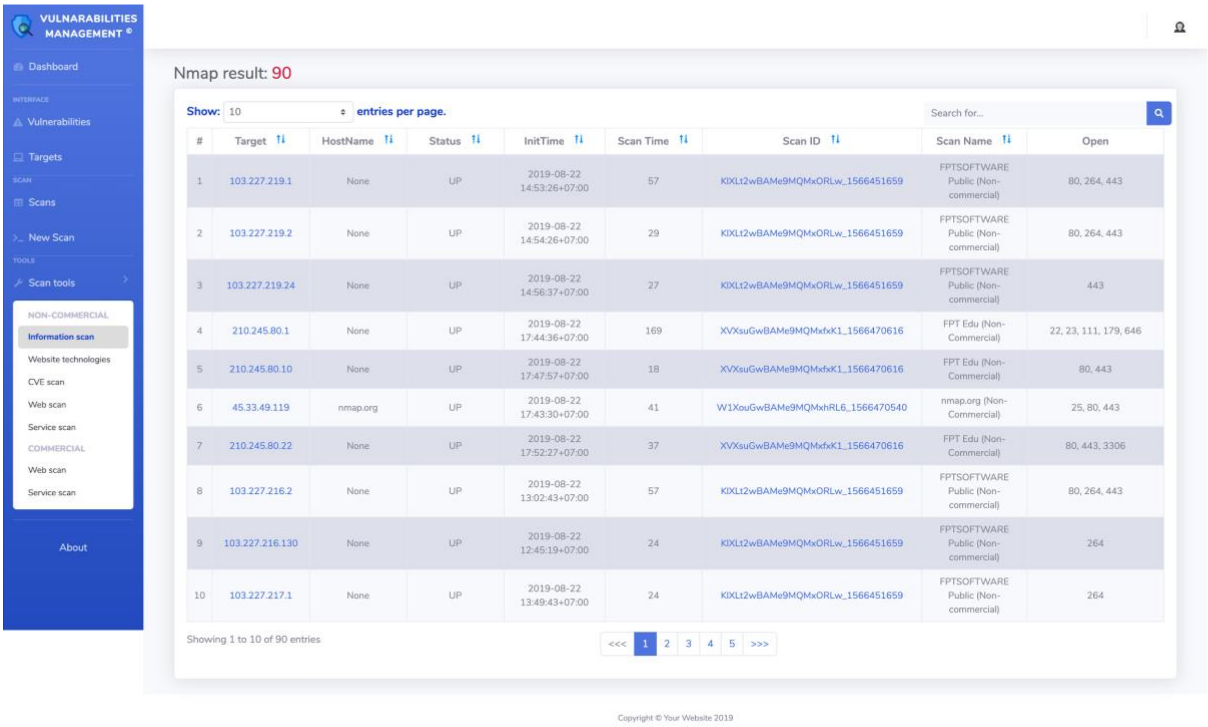
#	Target	Hostname	Started Time	Status	Scan ID	Scan Name	Summary Result
1	103.227.219.1	None	2019-08-22 14:53:26+07:00	hostUp	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	4 0 0 0
2	103.227.219.2	None	2019-08-22 14:54:26+07:00	hostUp	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	0 0 0 0
3	103.227.219.24	None	2019-08-22 14:56:37+07:00	hostUp	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	7 0 0 0
4	210.245.80.1	None	2019-08-22 17:44:36+07:00	hostUp	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	0 1 7 3
5	210.245.80.10	None	2019-08-22 17:47:57+07:00	hostUp	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	0 0 0 0
6	45.33.49.119	nmap.org	2019-08-22 17:43:30+07:00	hostUp	W1XouGwBAMe9MQMxhRL6_1566470540	nmap.org (Non-Commercial)	0 2 22 1
7	210.245.80.22	None	2019-08-22 17:52:27+07:00	hostUp	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	7 0 0 0
8	103.227.216.2	None	2019-08-22 13:02:43+07:00	hostUp	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	3 0 0 0
9	103.227.216.130	None	2019-08-22 12:45:19+07:00	hostUp	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	0 0 0 0
10	103.227.217.1	None	2019-08-22 13:49:43+07:00	hostUp	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	0 0 0 0

Showing 1 to 10 of 90 entries

<<< 1 2 3 4 5 >>>

Copyright © Your Website 2019

This page displays all the target.



Nmap result: 90

Show: 10 entries per page.

Search for...

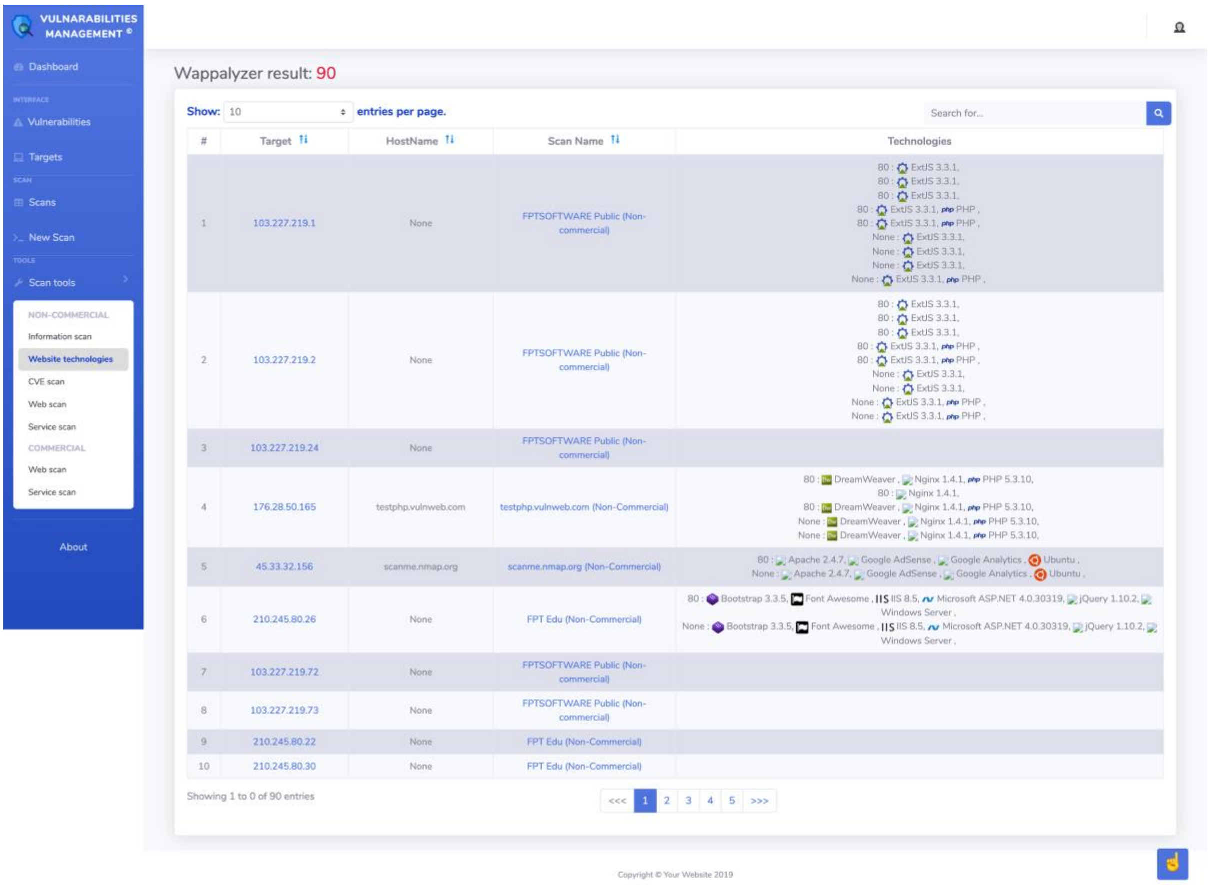
#	Target	HostName	Status	InitTime	Scan Time	Scan ID	Scan Name	Open
1	103.227.219.1	None	UP	2019-08-22 14:53:26+07:00	57	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	80, 264, 443
2	103.227.219.2	None	UP	2019-08-22 14:54:26+07:00	29	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	80, 264, 443
3	103.227.219.24	None	UP	2019-08-22 14:56:37+07:00	27	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	443
4	210.245.80.1	None	UP	2019-08-22 17:44:36+07:00	169	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	22, 23, 111, 179, 646
5	210.245.80.10	None	UP	2019-08-22 17:47:57+07:00	18	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	80, 443
6	45.33.49.119	nmap.org	UP	2019-08-22 17:43:30+07:00	41	W1XouGwBAMe9MQMxhRL6_1566470540	nmap.org (Non-Commercial)	25, 80, 443
7	210.245.80.22	None	UP	2019-08-22 17:52:27+07:00	37	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	80, 443, 3306
8	103.227.216.2	None	UP	2019-08-22 13:02:43+07:00	57	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	80, 264, 443
9	103.227.216.130	None	UP	2019-08-22 12:45:19+07:00	24	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	264
10	103.227.217.1	None	UP	2019-08-22 13:49:43+07:00	24	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	264

Showing 1 to 10 of 90 entries

<<< 1 2 3 4 5 >>>

Copyright © Your Website 2019

This page displays NMAP scan results.



Wappalyzer result: 90

Show: 10 entries per page.

Search for...

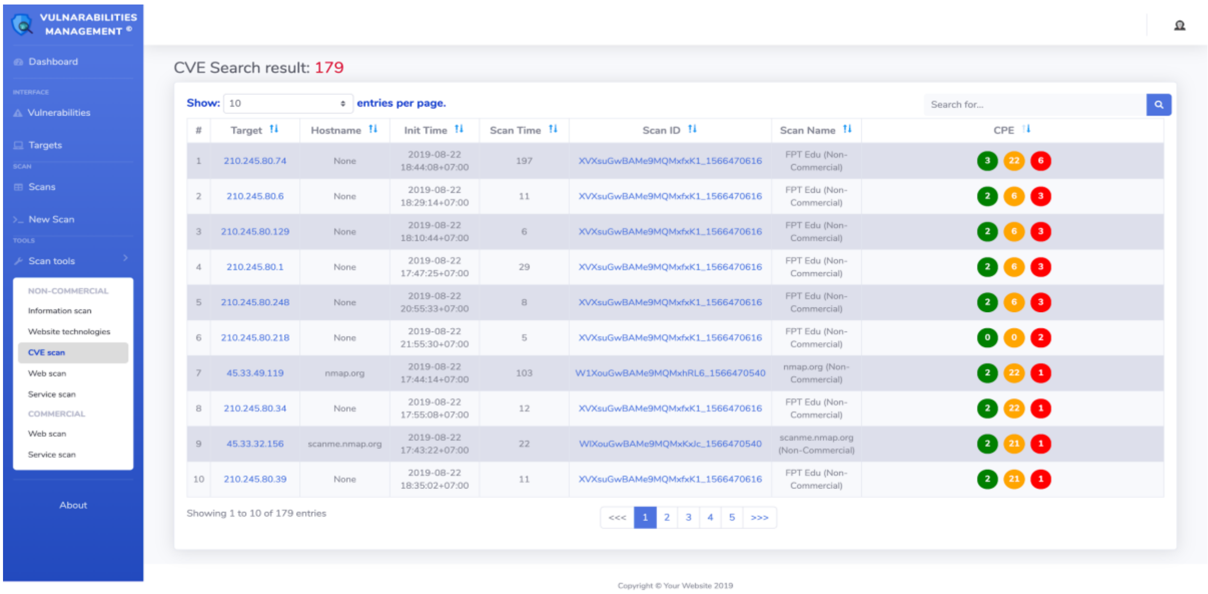
#	Target	HostName	Scan Name	Technologies
1	103.227.219.1	None	FPTSOFTWARE Public (Non-commercial)	80 : ExtJS 3.3.1, 80 : ExtJS 3.3.1, 80 : ExtJS 3.3.1, 80 : ExtJS 3.3.1, php PHP , 80 : ExtJS 3.3.1, php PHP , None : ExtJS 3.3.1, None : ExtJS 3.3.1, None : ExtJS 3.3.1, None : ExtJS 3.3.1, php PHP ,
2	103.227.219.2	None	FPTSOFTWARE Public (Non-commercial)	80 : ExtJS 3.3.1, 80 : ExtJS 3.3.1, 80 : ExtJS 3.3.1, 80 : ExtJS 3.3.1, php PHP , 80 : ExtJS 3.3.1, php PHP , None : ExtJS 3.3.1, None : ExtJS 3.3.1, None : ExtJS 3.3.1, php PHP , None : ExtJS 3.3.1, php PHP ,
3	103.227.219.24	None	FPTSOFTWARE Public (Non-commercial)	
4	176.28.50.165	testphp.vulnweb.com	testphp.vulnweb.com (Non-Commercial)	80 : DreamWeaver , Nginx 1.4.1, php PHP 5.3.10, 80 : Nginx 1.4.1, 80 : DreamWeaver , Nginx 1.4.1, php PHP 5.3.10, None : DreamWeaver , Nginx 1.4.1, php PHP 5.3.10, None : DreamWeaver , Nginx 1.4.1, php PHP 5.3.10,
5	45.33.32.156	scanme.nmap.org	scanme.nmap.org (Non-Commercial)	80 : Apache 2.4.7, Google AdSense , Google Analytics , Ubuntu , None : Apache 2.4.7, Google AdSense , Google Analytics , Ubuntu ,
6	210.245.80.26	None	FPT Edu (Non-Commercial)	80 : Bootstrap 3.3.5, Font Awesome , IIS IIS 8.5, Microsoft ASP.NET 4.0.30319, jQuery 1.10.2, Windows Server , None : Bootstrap 3.3.5, Font Awesome , IIS IIS 8.5, Microsoft ASP.NET 4.0.30319, jQuery 1.10.2, Windows Server ,
7	103.227.219.72	None	FPTSOFTWARE Public (Non-commercial)	
8	103.227.219.73	None	FPTSOFTWARE Public (Non-commercial)	
9	210.245.80.22	None	FPT Edu (Non-Commercial)	
10	210.245.80.30	None	FPT Edu (Non-Commercial)	

Showing 1 to 0 of 90 entries

<<< 1 2 3 4 5 >>>

Copyright © Your Website 2019

This page displays Wappalyzer scan results.



CVE Search result: 179

Show: 10 entries per page.

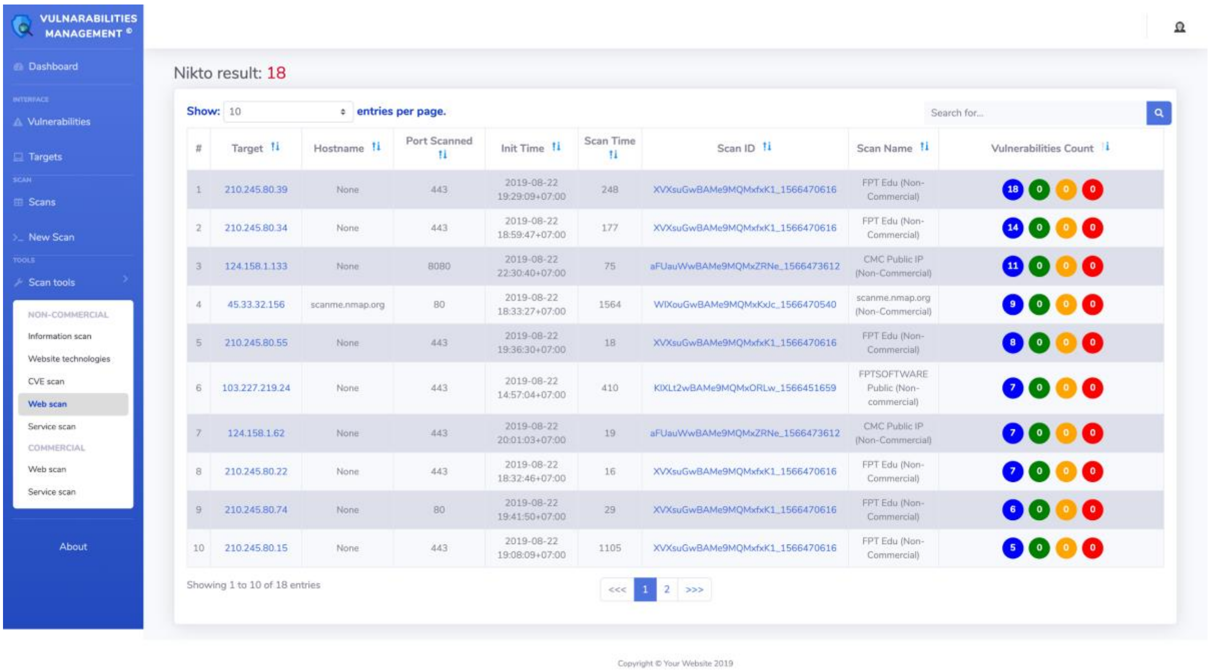
#	Target	Hostname	Init Time	Scan Time	Scan ID	Scan Name	CPE
1	210.245.80.74	None	2019-08-22 18:44:08+07:00	197	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	3 22 6
2	210.245.80.6	None	2019-08-22 18:29:14+07:00	11	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	2 6 3
3	210.245.80.129	None	2019-08-22 18:10:44+07:00	6	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	2 6 3
4	210.245.80.1	None	2019-08-22 17:47:25+07:00	29	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	2 6 3
5	210.245.80.248	None	2019-08-22 20:55:33+07:00	8	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	2 6 3
6	210.245.80.218	None	2019-08-22 21:55:30+07:00	5	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	0 0 2
7	45.33.49.119	nmap.org	2019-08-22 17:44:14+07:00	103	W1XouGwBAMe9MQMxhRL6_1566470540	nmap.org (Non-Commercial)	2 22 1
8	210.245.80.34	None	2019-08-22 17:55:08+07:00	12	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	2 22 1
9	45.33.32.156	scanme.nmap.org	2019-08-22 17:43:22+07:00	22	WIXouGwBAMe9MQMxKxJc_1566470540	scanme.nmap.org (Non-Commercial)	2 21 1
10	210.245.80.39	None	2019-08-22 18:35:02+07:00	11	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	2 21 1

Showing 1 to 10 of 179 entries

<<< 1 2 3 4 5 >>>

Copyright © Your Website 2019

This page display CVE Search scan results.



Nikto result: 18

Show: 10 entries per page.

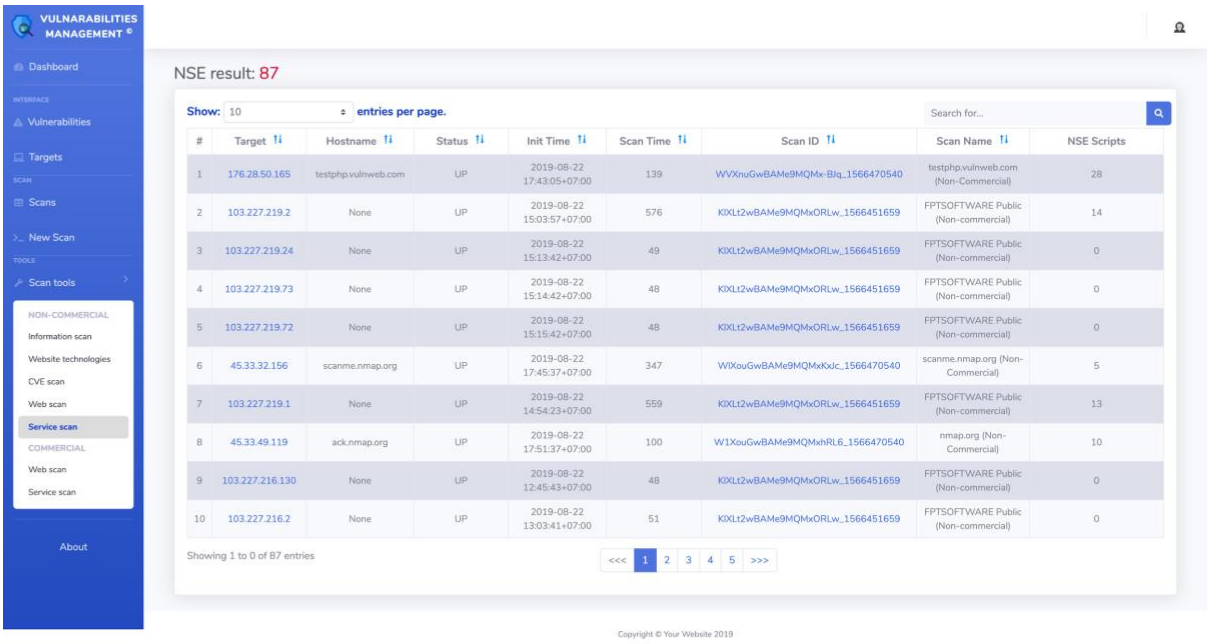
#	Target	Hostname	Port Scanned	Init Time	Scan Time	Scan ID	Scan Name	Vulnerabilities Count
1	210.245.80.39	None	443	2019-08-22 19:29:09+07:00	248	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	18 0 0 0
2	210.245.80.34	None	443	2019-08-22 18:59:47+07:00	177	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	14 0 0 0
3	124.158.1.133	None	8080	2019-08-22 22:30:40+07:00	75	aFUauWwBAMe9MQMxZRNe_1566473612	CMC Public IP (Non-Commercial)	11 0 0 0
4	45.33.32.156	scanme.nmap.org	80	2019-08-22 18:33:27+07:00	1564	WIXouGwBAMe9MQMxKxJc_1566470540	scanme.nmap.org (Non-Commercial)	9 0 0 0
5	210.245.80.55	None	443	2019-08-22 19:36:30+07:00	18	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	8 0 0 0
6	103.227.219.24	None	443	2019-08-22 14:57:04+07:00	410	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	7 0 0 0
7	124.158.1.62	None	443	2019-08-22 20:01:03+07:00	19	aFUauWwBAMe9MQMxZRNe_1566473612	CMC Public IP (Non-Commercial)	7 0 0 0
8	210.245.80.22	None	443	2019-08-22 18:32:46+07:00	16	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	7 0 0 0
9	210.245.80.74	None	80	2019-08-22 19:41:50+07:00	29	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	6 0 0 0
10	210.245.80.15	None	443	2019-08-22 19:08:09+07:00	1105	XVXsuGwBAMe9MQMxfxK1_1566470616	FPT Edu (Non-Commercial)	5 0 0 0

Showing 1 to 10 of 18 entries

<<< 1 2 >>>

Copyright © Your Website 2019

This page displays Nikto scan results.



NSE result: 87

Show: 10 entries per page.

#	Target	Hostname	Status	Init Time	Scan Time	Scan ID	Scan Name	NSE Scripts
1	176.28.50.165	testphp.vulnweb.com	UP	2019-08-22 17:43:05+07:00	139	WVXnuGwBAMe9MQMx-BJq_1566470540	testphp.vulnweb.com (Non-Commercial)	28
2	103.227.219.2	None	UP	2019-08-22 15:03:57+07:00	576	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	14
3	103.227.219.24	None	UP	2019-08-22 15:13:42+07:00	49	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	0
4	103.227.219.73	None	UP	2019-08-22 15:14:42+07:00	48	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	0
5	103.227.219.72	None	UP	2019-08-22 15:15:42+07:00	48	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	0
6	45.33.32.156	scanme.nmap.org	UP	2019-08-22 17:45:37+07:00	347	WIXouGwBAMe9MQMxKxJc_1566470540	scanme.nmap.org (Non-Commercial)	5
7	103.227.219.1	None	UP	2019-08-22 14:54:23+07:00	559	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	13
8	45.33.49.119	ack.nmap.org	UP	2019-08-22 17:51:37+07:00	100	W1XouGwBAMe9MQMxhRL6_1566470540	nmap.org (Non-Commercial)	10
9	103.227.216.130	None	UP	2019-08-22 12:45:43+07:00	48	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	0
10	103.227.216.2	None	UP	2019-08-22 13:03:41+07:00	51	KIXLt2wBAMe9MQMxORLw_1566451659	FPTSOFTWARE Public (Non-commercial)	0

Showing 1 to 0 of 87 entries

This page displays NSE scan result.

Appendix C - Scan performance

C.1. Internal result (LAN range)

The table below is about our solution when we run the internal IP. The number is just the result we measure when about ⅓ IPs is up.

	Non-Commercial Tools	Commercial Tools
1 IP	8m 43s (contain a web-server)	34m 37s (contain a web-server)
/24 Subnet	32m 2s (⅓ IPs is UP)	3h 47m 32s (⅓ IPs is UP)

Nmap performance:

	Time consuming
1 IP	2m 11s
/24 Subnet	9h 21m 10s

CVE Search performance:

	Time consuming
1 IP	1s
/24 Subnet	17s

Nikto performance. The number is just the result we measure when about ⅓ IPs is up.

	Time consuming
1 IP	59s
/24 Subnet	12m 14s

NSE performance.

	Time consuming
1 IP	1m 39s

/24 Subnet	26m 26s

Acunetix performance. The number is just the result we measure when about ⅓ IPs is up. And there are just a few web servers running on the subnet.

	Time consuming
1 IP	5m 6s
/24 Subnet	3h 55m 50s

Nessus performance.

	Time consuming
1 IP	7m 45s
/24 Subnet (Just 23 IPs)	2h 17m 20s

C.2. Internet Result (Public Range)

Internet results depend on many aspects, internet connection, WAF, Firewall, ... So, the result may be not really accurate.

Total time scan:

	Time consuming
1 IP	13m 23s
/24 Subnet (⅕ is UP)	5h 24m 20s (⅕ IPs is UP)

Nmap performance:

	Time consuming
1 IP	1m 13s
/24 Subnet (⅕ is UP)	4h 17m 10s

CVE Search performance:

	Time consuming

1 IP	1s
/24 Subnet	17s

Nikto performance. The number is just the result we measure when about ⅓ IPs is up.

	Time consuming
1 IP	7m 25s
/24 Subnet (⅕ is UP)	25m 47s

NSE performance.

	Time consuming
1 IP	9m 15s
/24 Subnet (⅕ is UP)	3h 52m 27s