Vulnerability analysis of a device connected to the Internet

A Summer Internship Project under the sponsorship of
Indian Academy of sciences at Super Computer Education and Research Centre,
IISc Bangalore.

by

HARISH PICHUKALA
SRKR Engineering College
Bhimavaram, West Godavari
Andhra Pradesh

Under the guidance of

Dr. N. Balakrishnan, Professor
Supercomputer Education and Research Centre
IISc Bangalore.

Contents

ABSTRACT

Globalization provides the world fastest and easiest way of communication through internet. A major challenge in the present world is to provide secure communication by satisfying the information attributes like confidentiality, integrity and availability of the assets. Day-by-day Adversaries adopting new methodologies, technologies and strategies to attack the target. In this project, a framework has been developed on vulnerability analysis of device which is connected to internet which will list the attack vectors and available exploits to launch the exploitation. The framework is useful to fix and improve the security of the device. In this analysis, the framework used CVE database to get the details of the known vulnerabilities using common platform enumeration (CPE).While performing the testing of the vulnerabilities, framework uses various vulnerability analysis and pentesting tools and codenomican tools. Using this framework security awareness of the user can be increased by knowing the vulnerabilities and by fixing these vulnerabilities in the device, the threat potentiality decreases.

INTRODUCTION

The Internet is termed as a huge network of various interconnected networks of business, government, Academic and so on. These networks are used for various purposes like sharing of documents, websites etc. The Internet has been playing a significant role since 20th century. Today we are using various websites like E-Commerce, online courses, e-news, social networking sites, e-news and so on which are provided by the common source called internet. Everything is dependent on the internet, we can't imagine the world without internet in the present scenario and it became an integrated part of every individual. As per the statistics given by the International Telecommunications Union (ITU) 40.4% of the people are internet users among the world's population. As per the UNODC [1] survey the internet users will be 70% of the world's population by 2017 by using mobile broadband subscriptions and it became hard to imagine the cybercrime. So, secure communication is the major challenge for the software engineers.

The essential requirements for the security models are as follows:

- Confidentiality which deals with protection of the information from unauthorized access. We are using encryption to achieve information confidentiality.
- Integrity related to the data that cannot be modified by any unauthorized user.
- Availability makes the information must be available when ever needed, security policies are used to achieve it.

In addition to these things there are many other attributes for the security model.

Attackers' basic motivation is to find the available vulnerabilities on the target device. Generally, Vulnerability is due to mistake in software design or in implementation which helps the attacker to develop or use the available exploits to access over it stated in [2]. Exploit is a piece of code takes vulnerability as an advantage to execute arbitrary commands referred as payloads to gain access of the target system and attackers are adopting advanced techniques of Machine learning, Artificial intelligence to automate the attacks. Attack vectors are the methods by which an attacker reaches its target. Common Vulnerability and Exposures (CVE) databases provides the details of known vulnerabilities and the impact of them. Mann and Christey (1999) given the CVE idea. Various penetration tools like Nmap [3], Nessus [4] detect the vulnerabilites.

Every device has some details like operating system, vendor etc. Common Platform Enumeration (CPE) gives the details like name of the vendor, product name, versions of the product and also specifies details like whether it is application or operating system or hardware. The present framework uses python as a scripting language. The framework is graphical user interface which uses python Tk interface library.

Simon Hanson et al. proposed a taxonomy of network and computer attack [5] to categorise the attack dimensions. Andrea Bittau et al. given how attackers are exploiting the target without having the binary or source code of the services of the target [6].Qiang Zeng et al. suggested Target therapy for program bugs which will perform the detection, diagnosis of bug [7].Isham chokshi et al. worked on Exploit dependency graph which will give us all possible attacks [8].

MOTIVATION

Cyber Attacks are due to known and unknown vulnerabilites. These attacks can be prevented up to some extent by patching the vulnerabilites. Penetration testing helps us to identify the vulnerabilites and system weakness. The real-world security based issues motivated to learn Cyber Security field.

PRIOR ART

Vulnerabilites are the major causes of security threat. There are various vulnerability analysis and pentesting (VAPT) tools to the details of vulnerabilites of the target system. Pentesting or penetration testing is entering into the network with permission of the organisation to get the access of the resources before the attacker attempts. The different methodologies in pentesting are black box, white box, gray box testing. Before discussing various VAPT tools, there is a need to discuss various vulnerabilites.

Common Vulnerabilites:

As per the Open Web Security Application Project (OWSAP) [9] the top

Vulnerabilites are given below

Command Injection: It will inject and execute the commands on the vulnerable application. These are due to improper input validation. In some causes command injection can cause privilege escalation.

Stored XSS: This vulnerability is most dangerous because the script saves in storage location and used by all users. Most of the websites possess these vulnerabilites

External Control of file:It allows the user input to control or influence path or filename are used in File system operations. Due to this vulnerability CIA triads constraints are violated.

Weak Captcha: Captcha is used to test whether the user is human or not. Most of the captcha implementations are insecure. Using automated recognition techniques breaks captcha.

SQL injection authentication bypass: This vulnerability uses SQL injection which can inject or insert of SQL query via input data from client. It is similar to command execution.

Malicious File upload: It leads to serious problem which allows the attacker to upload a file which will perform command execution on the servers and give the access of the target system to the attacker.

Web-Dav Authentication bypass: Web based distributed authorization and versioning is the set of HTTP extensions that allow collaborate management and editing of files collected on remote servers. The Microsoft IIS's implementation of Web-Dav handles Unicode tokens may allow the Authentication bypass.

Various VAPT tools:

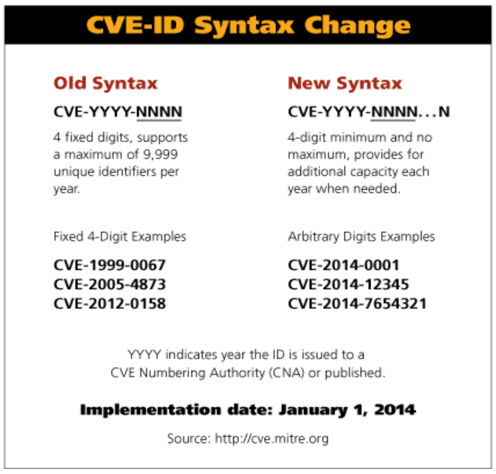
Name	Present Version	Platform	Security checks	Limitations
Nmap	6.46	Open source, cross platform	Vulnerability check scanning like open ports and services running on ports and auditing	Fingerprinting is not accurate
Zen map	6.46	Open source, cross platform	Same as Nmap but it is GUI	slow when compared with Nmap
Wireshark	1.10.7	Open source, cross platform	It inspects how packets are going	slow performance, buffer overflow attacks
Hping	3.0.0 alpha 2	Open source, Cross platform	Scanning, sniffing	speed
Nessus	5.2.6	Commercial Cross platform	Credential scan, patch management, Malware analysis	Some plugins are only for registered users
Metasploit	4.9.6	Open source, Cross platform	Detects various vulnerabilites and provides various exploits	Majority of exploits are windows exploits
Sqlmap	0.9	Open source, Cross platform	whether it will exploit remote database and extract info from database	performance hog on server-side
Dsniff	2.3	Open source, Cross platform	Explore security vulnerabilites of packet switching network.	Adds delay and overhead

OWSAP ZAP	2.3.1	Open source, Cross platform	Finds vulnerabilites like xss, sql injection	Speed

These are the various VAPT tools plays a crucial role in detecting vulnerabilites.

CVE Databases:

Common vulnerability and exposure (CVE) databases provide the information about commonly known vulnerabilities and the impact of the vulnerability. The syntax of the cve id is shown in figure (a).The CVE database is publicly available to all provided by NIST NVD. NVD databases are available in XML format in tree structure and structure of xml NVD is shown in figure (b).



CVE-ID Syntax Change

Old Syntax	New Syntax
CVE-YYYY-NNNN	CVE-YYYY-NNNN...N
4 fixed digits, supports a maximum of 9,999 unique identifiers per year.	4-digit minimum and no maximum, provides for additional capacity each year when needed.
Fixed 4-Digit Examples	Arbitrary Digits Examples
CVE-1999-0067	CVE-2014-0001
CVE-2005-4873	CVE-2014-12345
CVE-2012-0158	CVE-2014-7654321

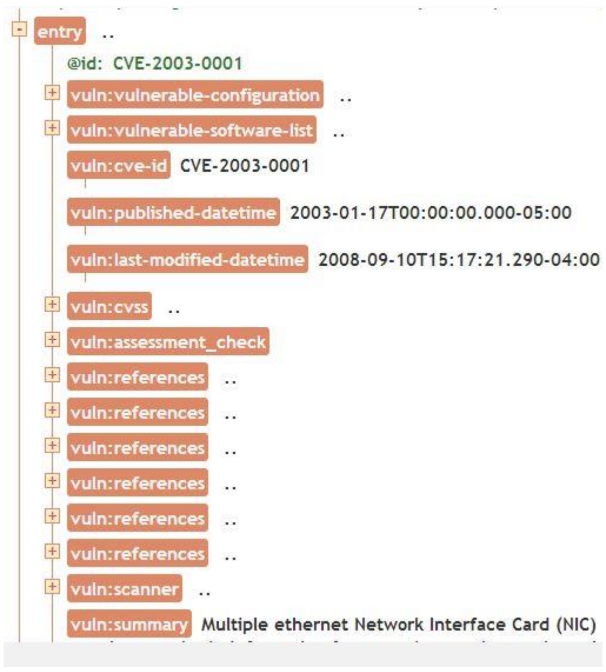
YYYY indicates year the ID is issued to a CVE Numbering Authority (CNA) or published.

Implementation date: January 1, 2014

Source: http://cve.mitre.org

Figure (a) syntax CVE ID

The structure of NVD XML tree contains vulnerability configuration list, software list, cvss, impact, summary and so on.



```
entry ..
  @id: CVE-2003-0001
  vuln:vulnerable-configuration ..
  vuln:vulnerable-software-list ..
  vuln:cve-id CVE-2003-0001
  vuln:published-datetime 2003-01-17T00:00:00.000-05:00
  vuln:last-modified-datetime 2008-09-10T15:17:21.290-04:00
  vuln:cvss ..
  vuln:assessment_check
  vuln:references ..
  vuln:references ..
  vuln:references ..
  vuln:references ..
  vuln:references ..
  vuln:references ..
  vuln:scanner ..
  vuln:summary Multiple ethernet Network Interface Card (NIC)
```

Figure (b): NVD vulnerability entry structure

The cvss is the common vulnerability scoring system which gives the base score to measure the severity of the vulnerability.

CWE is the common weakness enumeration specification provides a common language of discussing, finding and deals with the causes of software security.

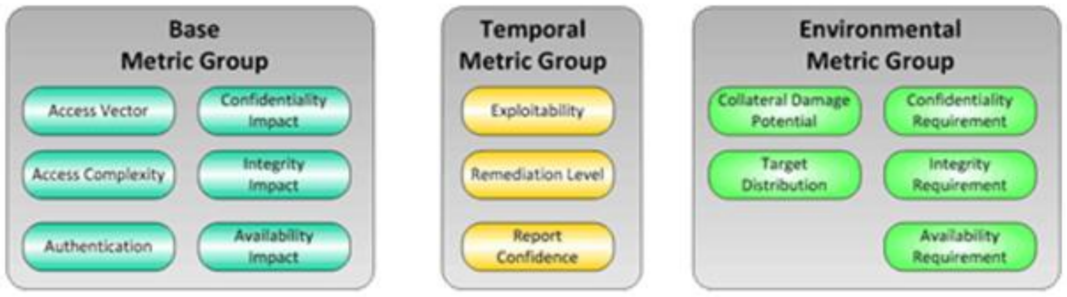
<u>Networking</u>:

The OSI model and TCP/IP are the two basic things in computer networks. The different layers in OSI model are Physical, data link, network, transport, session, presentation, application layers. Each layer has its own importance. TCP/IP contains only 5 layers it does not have presentation and session layers. The network or Internet layer in TCP/IP provides ip addressing and subnetting. The ip addresses are unique number ID assigned to one host or interface in a network and these addresses are classified into classes like A, B, C networks.

Subnetting allows to create multiple network that exists within a single class A, B, C.

CVSS (Common Vulnerability Scoring System):

CVSS attempts to measure of how much concern a vulnerability warrants, compared to other vulnerabilites with a base score. The CVSS is composed of three metric groups: Base, Temporal, and Environmental. During the calculation of the base score all these metric groups are considered.



If the base score is in range of 7-10 then the vulnerability is critical, 4-6.7 major and 0-3.9 is minor.

Python is a scripting language used while developing the framework which uses Tkinter module for GUI and threading for multithreading.

Simon Hanson et al. taxonomy on network and computer attacks [5] gives the idea on how to use this taxonomy to categorize various attacks and proposed taxonomy has 4 dimensions which are attack vector and behaviour of attack, classification of attack targets, vulnerability classification, payloads.

Qiang Zeng et al. Proposed Therapy for program bugs will detect the bugs and diagnosis them [7].It will helpful to patch the services containing bugs.

Isham chokshi et al. suggested exploit dependency graph which gives us all various attack modelling techniques includes finding required exploits to perform post exploitation etc. [8].

As per the Miguel Garcia et al. “Analysis of OS diversity for Intrusion Tolerance” [9] tells that the OS will give the possible vulnerabilites. OS vulnerability data is available from NIST National vulnerability Database(NVD).OS fingerprinting can be performed with tools like Nmap by observing the TTL value, window size, Initial sequence number(ISN) and so on.

Approach

For the development of this framework, Python is used as scripting language and it uses Tkinter module for GUI development. The framework includes the various modules which perform the following tasks

1. Network discovery
2. Host discovery
3. Vulnerabilites details
4. Available exploits

Network Discovery is performed by using Ping [10] utility. The network discovery includes the finding the alive nodes in the network the alive nodes can be obtained using the alive_nodes algorithm. It takes ip range as input and gives us the alive hosts in the network.

Input: ip range

Output: alive hosts

```
Algorithm   Alive_nodes
    Initialise alive hosts to empty list
    Hosts ← ip range
    For host in hosts
    do
        Response ← Ping (host)
        Analyse Response and decide whether host is status
        If host is alive then
            Add host to alive hosts
        End
    End
    Return alive hosts
```

The alive_nodes algorithm uses ping[10] utility which sends the ICMP echo requests to the device and by analysing the response corresponding to the request and decide whether it is active or not. The Response will tell whether device is active, unreachable, could not find host. The following figure (a) explains how the algorithm alive node works

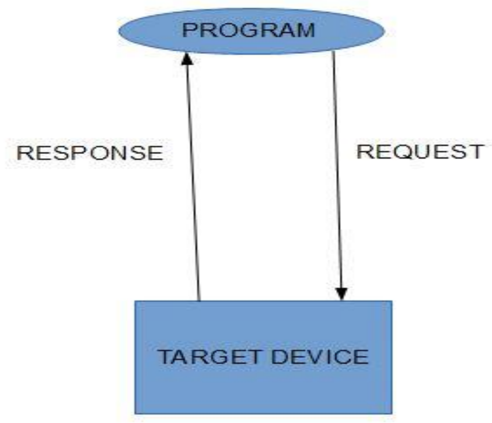


Figure (a)

After obtaining the alive hosts, the framework need to perform the host discovery to get the details of the hosts what is the operating system of the host. To obtain CPE, the algorithm uses Nmap [11] utility which is port scanner lists out the open ports and services running on the device. It also performs OS Finger printing [12] used various techniques to get the CPE of the host. The algorithm uses alive hosts and gives OS details of the devices. The Algorithm uses multithreading approach to reduce the execution time. As a result time complexity decreases when compared to previous case.

```
Input: alive hosts

Output: host details

Algorithm Host discovery

    host details ← empty list

    CPE ←null

    For host in alive hosts

    do

        CPE ←get CPE using Nmap

        Add CPE to host details

    End
```

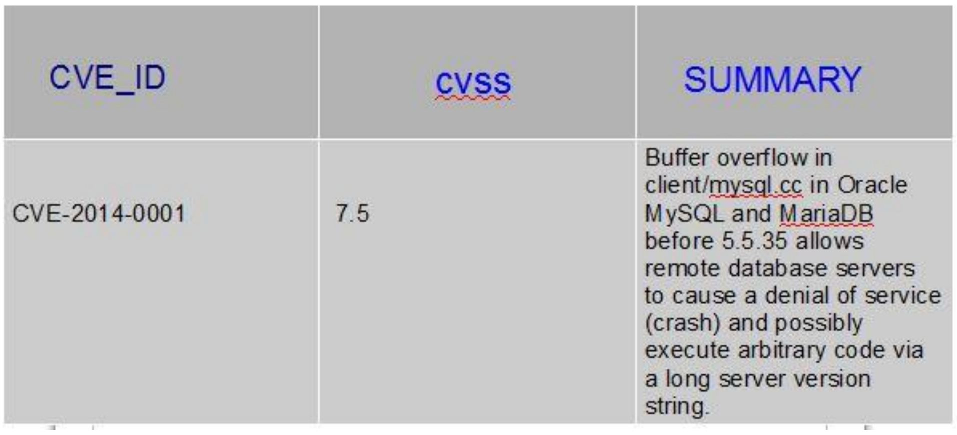
Return host details

After obtaining the CPE details of host, the vulnerabilites can be obtained by searching on NVD database which are publicly available in XML format. Searching the CPE and get the vulnerabilites efficiently. So, NVD in XML formats are converted into Database concepts. The Framework uses NVD from 2001-2014 xml files as dataset downloaded from NVD [13].

The xml_db module converts xml files into databases which includes 3 parsers. Let us consider set of XML files as X and schema as S, described using Relational Algebra. The proposed schema for this application is S, database schema which consists of set of relational schema. The Relational schema are

CVE_DETAILS (cve_id: String, cvss: String, summary: String)

The figure (b) shows the example for CVE_DETAILS

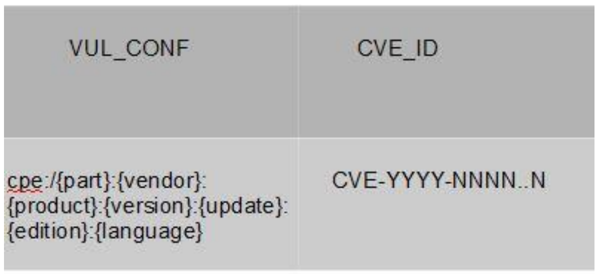


CVE_ID	cvss	SUMMARY
CVE-2014-0001	7.5	Buffer overflow in client/mysql.cc in Oracle MySQL and MariaDB before 5.5.35 allows remote database servers to cause a denial of service (crash) and possibly execute arbitrary code via a long server version string.

Figure (b) CVE_DETAILS

VUL_CONF_LIST (vul_conf : String, cve_id: String)

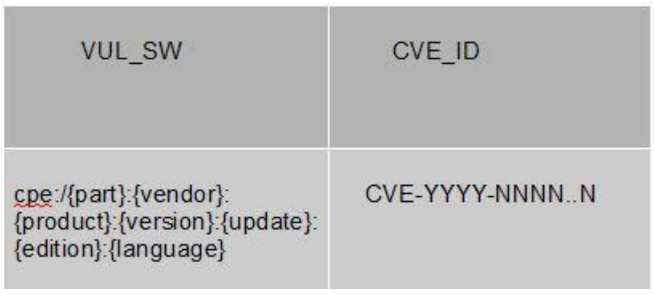
It is shown in figure (c), shows the table used in database



VUL_CONF	CVE_ID
cpe:/{part}:{vendor}:{product}:{version}:{update}:{edition}:{language}	CVE-YYYY-NNNN..N

Figure (c) Vulnerability configuration details

VUL_SW_LIST (vul_sw: String, cve_id: String) is shown in figure (d)



VUL_SW	CVE_ID
cpe:/{part}:{vendor}:{product}:{version}:{update}:{edition}:{language}	CVE-YYYY-NNNN..N

Figure (d): Vulnerability Software details

cve_id is the primary key which are used to identify the tuples in a Relation.

The parser go through the XML file and performs mining and get the CVE-ID, CVSS, summary, Vulnerability configuration and vulnerability software. The parser uses the elementary tree which is available in python standard xml module. The parser goes through the xml document and parse the children and get there values and insert them into the table as per the schema specified. XML_DB module takes X, S as input and gives the database say D. It performs parsing on each xml document and get the required fields. The parsing function is given as P and relation as r.

For xi belongs to X, Q is the set of required fields. The parsing function P which gives us the tuple of values corresponding to respective fields.

It is given by

R=P (xi, Q) where R= {k/k in V (Q) and length of Q and R are equal}

Where V is the value function which gives the value of the set of fields. Inserting into tuple into respective relation r is given by

$$\mathbf{r \leftarrow r \cup R}$$

After converting XML files into database the search on database is performed to get the CPE details. As a result, DB_Functions is developed to perform database operation such as searching, deleting the contents, total entries and so on. The module includes various db functionalities. Using this module we can search on the NVD database to get the corresponding vulnerabilites. The detailed flowchart is shown in figure (e)

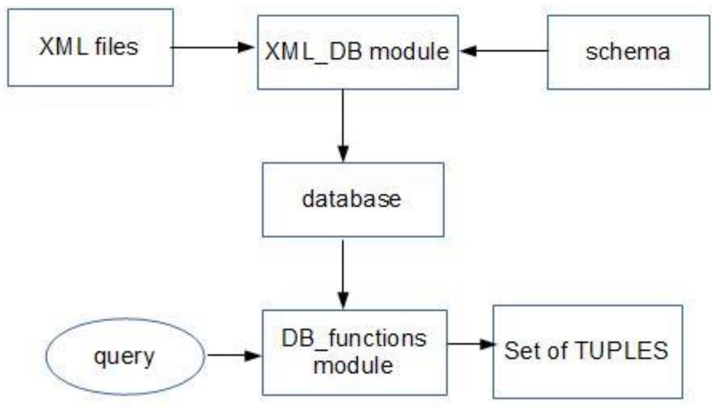


Figure (e): gets data from database

The vulnerabilities are obtained by using get_vulnerabilities algorithm which takes host details and gives vulnerability details corresponding to the host. It uses NIST NVD database to get details of vulnerability. The algorithm is given below

```
Input: host details

Output: Vulnerability details

Algorithm   get_vulnerabilities

    Initialise vulnerability details to empty

    OS details ←host details

    For ip, cpe in OS details

    do

        Search cpe on NVD database

        Get the vulnerabilites for particular CPE
        Add vulnerabilities to vulnerability list corresponding to each
    host
    End
    Return vulnerability list
```

After getting the vulnerabilites, framework would need to list the available exploits for the vulnerabilites. The framework uses exploit-db database to get the source code, author, published date and summary of the exploit. The schema for exploit database is shown below

Exploit_details (id, file, description, pdate, author, platform, type, port)

The file gives the location where the exploit is located and description gives the summary about exploit. The id is the primary key which is used to identify exploit uniquely. There is one more database which contains the details like exploit id and cve details corresponding to the exploit. The schema for exploit_cve is as follows Exploit (eid, cve)

where eid is the id of the exploit and cve contains the cve id of the vulnerabilites. The following algorithm lists out the available exploits. The algorithm takes vulnerability details and give the available exploits to launch exploitation. The algorithm uses the Exploit_details and exploit_cve databases to list available exploits

Input: vulnerability details

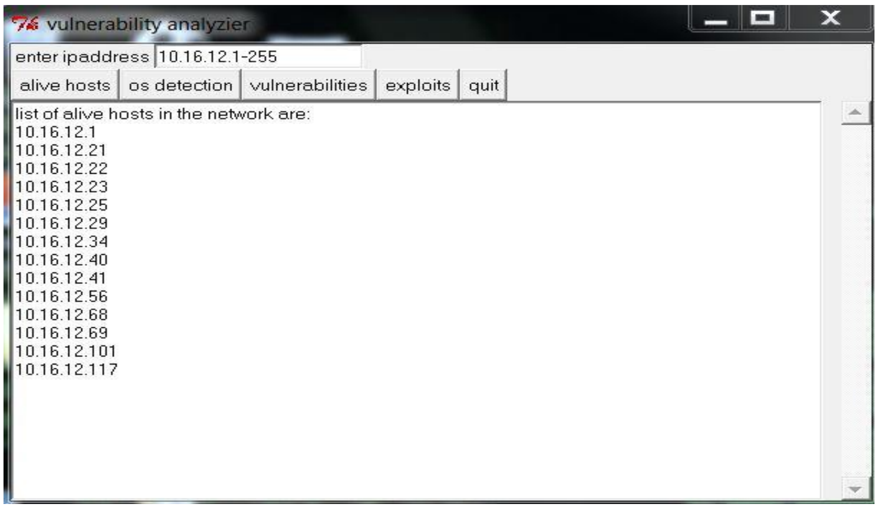
Output: available exploits

```
Algorithm get_exploits
    vul ←vulnerability details
    available exploits ←null
    for v in vul
    do
        cve_id ←Extract CVE-ID from v
        exploit_id ←Exploit_CVE (cve_id)
        e ←get_details (exploit_id)
        Add e to available exploits
    end
    Return available exploits
```

The above algorithm extracts cve id from vulnerability and uses exploit_cve database to get details of exploits. All the algorithms specified so far are implemented using multithreading approach to decrease the execution time. For implementing in multithreading the framework uses threading module in the python standard library. The framework also includes some module which will calculate the average CVSS score for each alive device and gives the average CVSS details of all alive hosts and some modules will create exploit databases, datasets and so on.

Results and Discussion

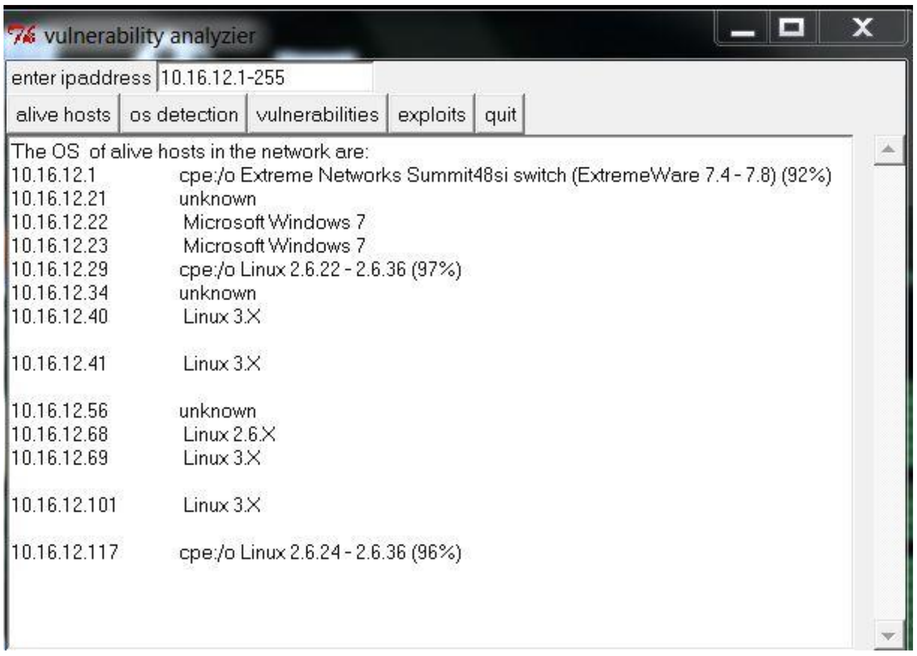
The framework list the available host, OS details, Vulnerabilites and available exploits. The available hosts i.e. figure (f) shown by GUI application shown below



```
list of alive hosts in the network are:
10.16.12.1
10.16.12.21
10.16.12.22
10.16.12.23
10.16.12.25
10.16.12.29
10.16.12.34
10.16.12.40
10.16.12.41
10.16.12.56
10.16.12.68
10.16.12.69
10.16.12.101
10.16.12.117
```

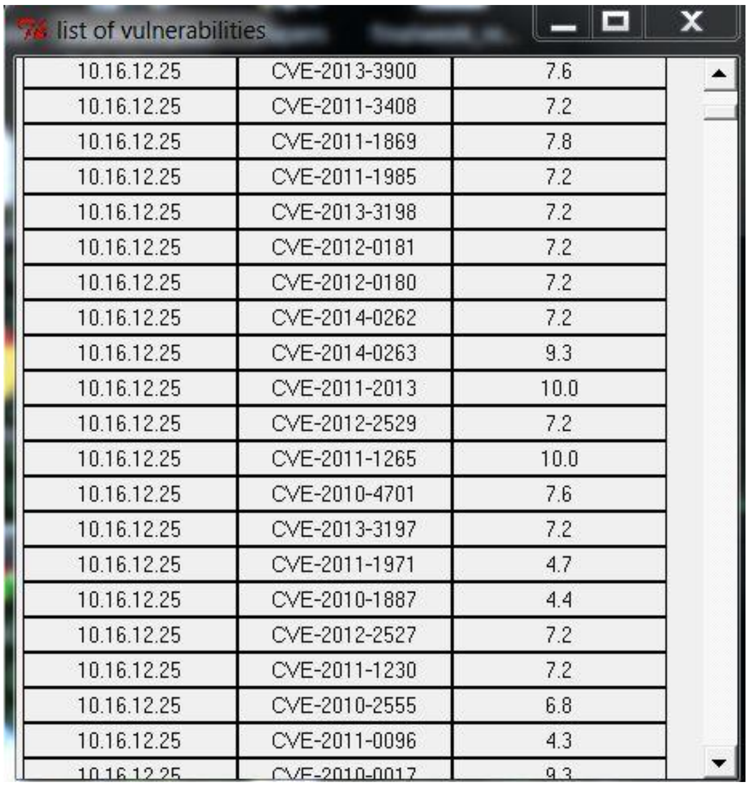
Figure (f): Alive hosts

The OS detection performed by the application is shown in figure (g) and vulnerabilites in figure (h) and available exploits in figure (i)



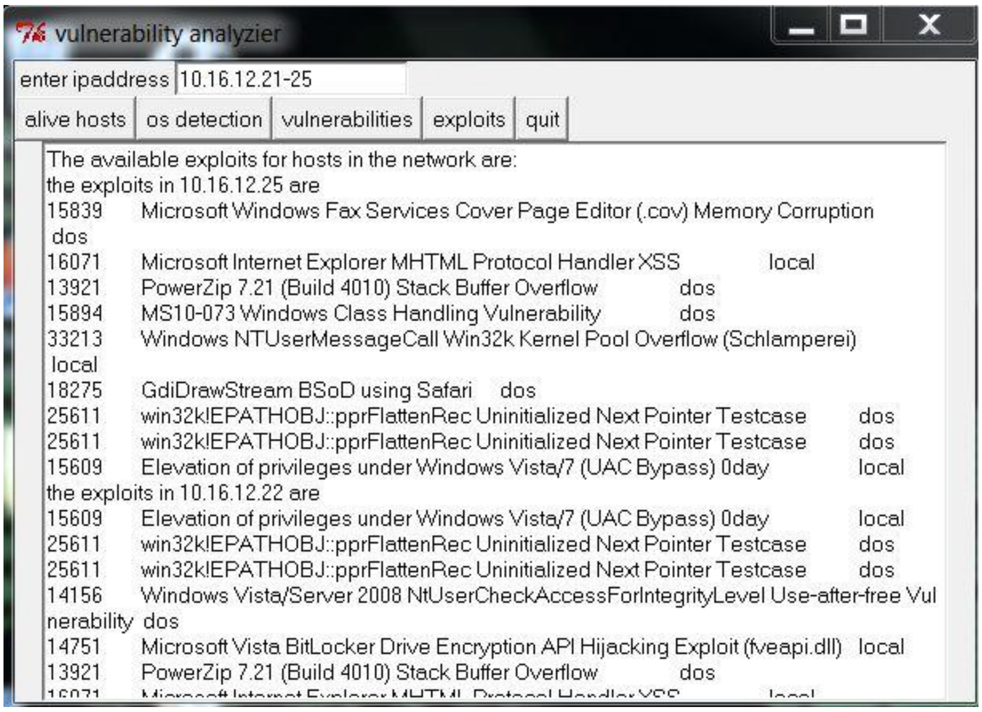
```
The OS  of alive hosts in the network are:
10.16.12.1        cpe:/o Extreme Networks Summit48si switch (ExtremeWare 7.4 - 7.8) (92%)
10.16.12.21       unknown
10.16.12.22        Microsoft Windows 7
10.16.12.23        Microsoft Windows 7
10.16.12.29       cpe:/o Linux 2.6.22 - 2.6.36 (97%)
10.16.12.34       unknown
10.16.12.40        Linux 3.X

10.16.12.41        Linux 3.X

10.16.12.56       unknown
10.16.12.68        Linux 2.6.X
10.16.12.69        Linux 3.X

10.16.12.101       Linux 3.X

10.16.12.117      cpe:/o Linux 2.6.24 - 2.6.36 (96%)
```

Figure (g): OS detection



10.16.12.25	CVE-2013-3900	7.6
10.16.12.25	CVE-2011-3408	7.2
10.16.12.25	CVE-2011-1869	7.8
10.16.12.25	CVE-2011-1985	7.2
10.16.12.25	CVE-2013-3198	7.2
10.16.12.25	CVE-2012-0181	7.2
10.16.12.25	CVE-2012-0180	7.2
10.16.12.25	CVE-2014-0262	7.2
10.16.12.25	CVE-2014-0263	9.3
10.16.12.25	CVE-2011-2013	10.0
10.16.12.25	CVE-2012-2529	7.2
10.16.12.25	CVE-2011-1265	10.0
10.16.12.25	CVE-2010-4701	7.6
10.16.12.25	CVE-2013-3197	7.2
10.16.12.25	CVE-2011-1971	4.7
10.16.12.25	CVE-2010-1887	4.4
10.16.12.25	CVE-2012-2527	7.2
10.16.12.25	CVE-2011-1230	7.2
10.16.12.25	CVE-2010-2555	6.8
10.16.12.25	CVE-2011-0096	4.3
10.16.12.25	CVE-2010-0017	9.3

Figure (h): Vulnerabilites



enter ipaddress 10.16.12.21-25

```
The available exploits for hosts in the network are:
the exploits in 10.16.12.25 are
15839     Microsoft Windows Fax Services Cover Page Editor (.cov) Memory Corruption
 dos
16071     Microsoft Internet Explorer MHTML Protocol Handler XSS           local
13921     PowerZip 7.21 (Build 4010) Stack Buffer Overflow           dos
15894     MS10-073 Windows Class Handling Vulnerability           dos
33213     Windows NTUserMessageCall Win32k Kernel Pool Overflow (Schlamperei)
 local
18275     GdiDrawStream BSoD using Safari     dos
25611     win32k!EPATHOBJ::pprFlattenRec Uninitialized Next Pointer Testcase        dos
25611     win32k!EPATHOBJ::pprFlattenRec Uninitialized Next Pointer Testcase        dos
15609     Elevation of privileges under Windows Vista/7 (UAC Bypass) 0day        local
the exploits in 10.16.12.22 are
15609     Elevation of privileges under Windows Vista/7 (UAC Bypass) 0day        local
25611     win32k!EPATHOBJ::pprFlattenRec Uninitialized Next Pointer Testcase        dos
25611     win32k!EPATHOBJ::pprFlattenRec Uninitialized Next Pointer Testcase        dos
14156     Windows Vista/Server 2008 NtUserCheckAccessForIntegrityLevel Use-after-free Vul
nerability  dos
14751     Microsoft Vista BitLocker Drive Encryption API Hijacking Exploit (fveapi.dll)  local
13921     PowerZip 7.21 (Build 4010) Stack Buffer Overflow           dos
16071     Microsoft Internet Explorer MHTML Protocol Handler XSS           local
```

Figure (i): Available Exploits

The attack graph generated by calculating the average CVSS for each host is shown below

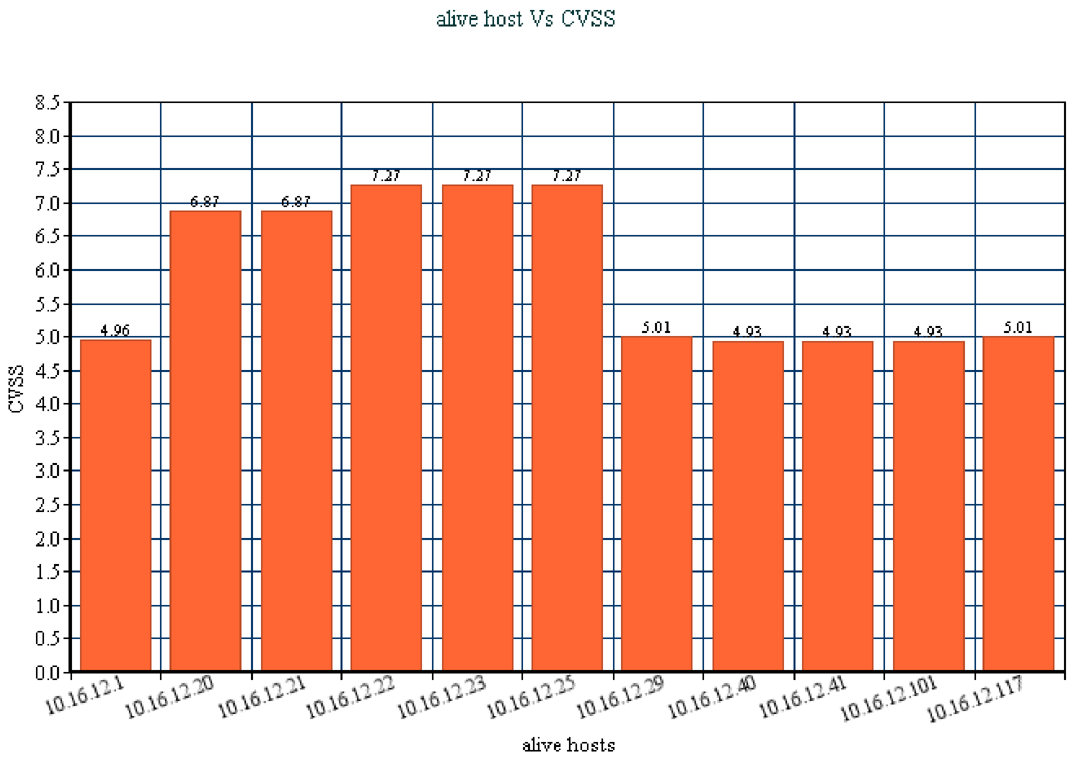
The average CVSS value is obtained by using following formula

$$f(\text{cvss}) = \sum_{n=v}^{k} (cvss_n)$$

$$where\ n\ indicates\ the\ vulnerability$$

$cvss_n$ *gives the cvss value corresponding to vulnerabilit* ,K is the total no of vulnerabilites. The average cvss is given by

Average CVSS=f(cvss)/k



The framework uses Nmap to perform OS figure printing which does not give the correct result. As a result false positives occurs.
Ex: It shows windows 7 professional as windows Vista. We can overcome the false positives by using some advanced machine learning techniques .As specified by Juan caballero et al. “Automatic Fingerprinting Generation”

CONCLUSION AND FUTURE WORK

The present framework list the vulnerabilites and available exploits corresponding to vulnerabilities. The framework is
Useful for the network administrator and system administrators to know the vulnerabilites in there Network and patch the vulnerable Application.
There are several things to be improved and further explored in this work. The framework only uses OS CPE and get the vulnerabilites from NVD database. Getting application details (CPE) is efficient when compared with OS.
Further, the framework need to include Automation exploitation to launch the exploits on devices.

ACKNOWLEDGEMENTS

I am extremely happy to express my thanks to the professor Dr. N. BalaKrishnan for his esteemed guidance. I would also like to thank K.P.Raghuraman and Pritam padhy for their support and valuable suggestions. It is my pleasure to express my gratitude to Indian Academy of science for their support for this fellowship Last but not least, I would like to thank that those who directly or indirectly involved in doing this project.

Thanks
Pichukala Harish

REFERENCES

1. UNODC [Online] available: http://www.unodc.org
2. MITRE, http://cve.mitre.org/about/terminolgy.html#vulnerability
3. Nmap, http://www.insecure.org/nmap/index.html
4. Nessus, http://www.nessus.org
5. Simon Hanson, Ray Hunt "A taxonomy of network and computer attacks"
6. Andrea Bittau, A. Belay, A. Mashtizadeh, D. Mazieres, and Dan Boneh, “Hacking blind,” in Proceedings of the 35th IEEE Symposium on security and Privacy, 2014
7. Qiang Zeng, mingyi Zhao, Peng liu "Target therapy for programming bugs"
8. Isham Chokshi, Nirnay Ghost and Soumya K Ghost, "Efficient Generation of Exploit dependency by customize Attack modelling Techniques"
9. OWSAP [online] available: http://www.owsap.org
10. Wikipedia, http://en.wikipedia.org/wiki/Ping_(networking_utility)
11. Nmap [online] available: http://nmap.org/man.html
12. SANS,"OS and application fingerprinting techniques" by John Mark Allen
13. NIST, National Vulnerability Database [online] available http://nvd.nist.gov