**IMPLEMENT CROSS SITE SCRIPTING**

**AIM:**

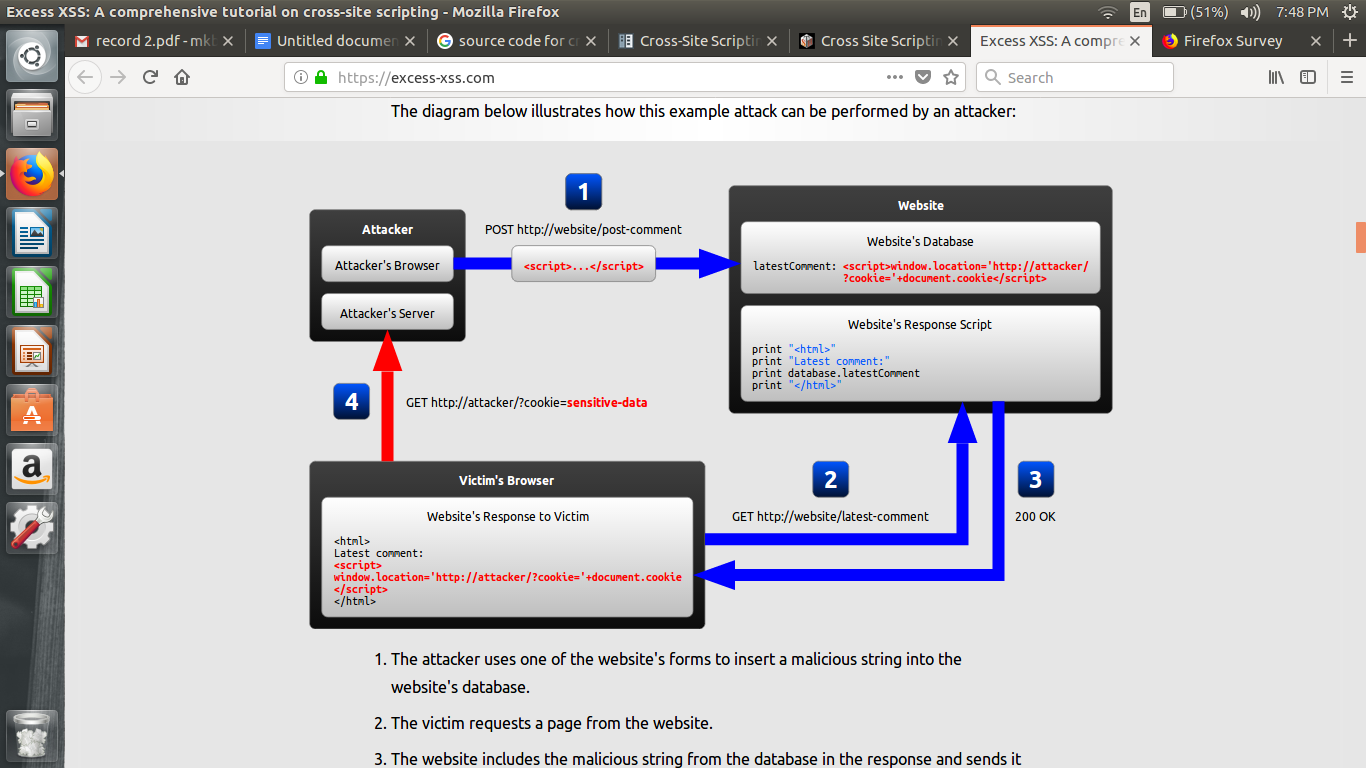
To implement cross site scripting.

**CROSS SITE SCRIPTING:**

Cross Site Scripting attack is a malicious code injection, which will be executed in the victim’s browser. Malicious script can be saved on the web server and executed every time when the user calls the appropriate functionality. It can also be performed with the other methods – without any saved script in the web serve

**PROCEDURE:**

HOW THE ATTACK WORKS



* The attacker uses one of the website's forms to insert a malicious string into the website's database.
* The victim requests a page from the website.
* The website includes the malicious string from the database in the response and sends it to the victim.
* The victim's browser executes the malicious script inside the response, sending the victim's cookies to the attacker's server.

**SOURCE CODE:**

<?php

// Variables for later use

$file\_name = "msgs.txt";

$msg\_begin = "~~~~MSG BEGIN~~~~\n";

$msg\_end = "~~~~MSG END~~~~\n";

// If there is a submission, write it to the file on the server.

if(($\_REQUEST["subject"] != "") && ($\_REQUEST["comments"] != ""))

{

$file = fopen($file\_name, "a"); // Open/create file to write in append mode

fwrite($file, "\n");

fwrite($file, $msg\_begin);

if (isset($\_COOKIE["user"]))

fwrite($file, "User: " . $\_COOKIE["user"] . "\n");

fwrite($file, "Date: " . date("r") . "\n");

// If the post contains a quotation symbol (e.g. "), it is by default replaced

// with a backslash and quote (\") when written to the file. The purpose of

// stripslashes below is to remove the backslash when it is written to the

// file. Note that this enables the Javascript to be inserted into the

// subject or comment text unaltered, causing a potential XSS hole.

fwrite($file, "Subject: " . stripslashes($\_REQUEST["subject"]) . "\n");

fwrite($file, "Comment:\n" . stripslashes($\_REQUEST["comments"]) . "\n");

fwrite($file, $msg\_end);

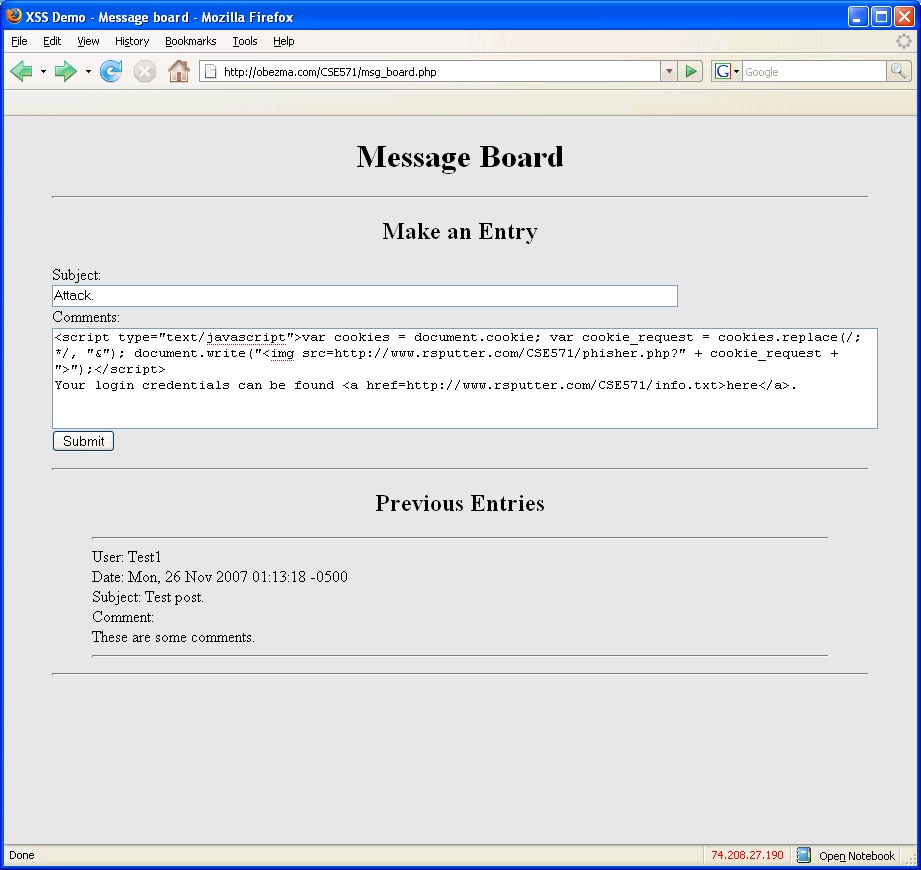
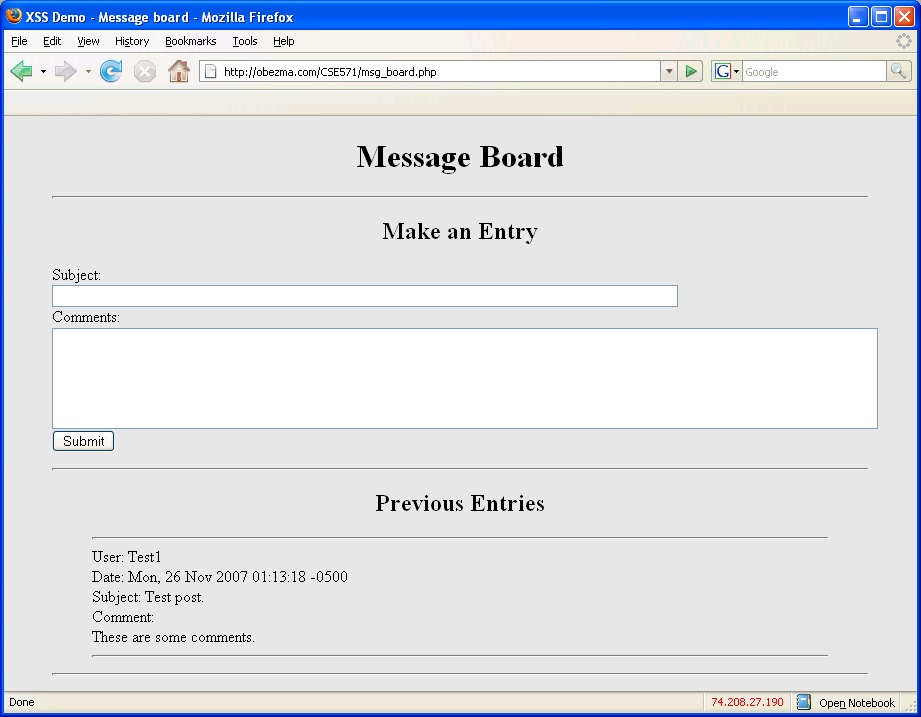
fwrite($file, "\n\n");

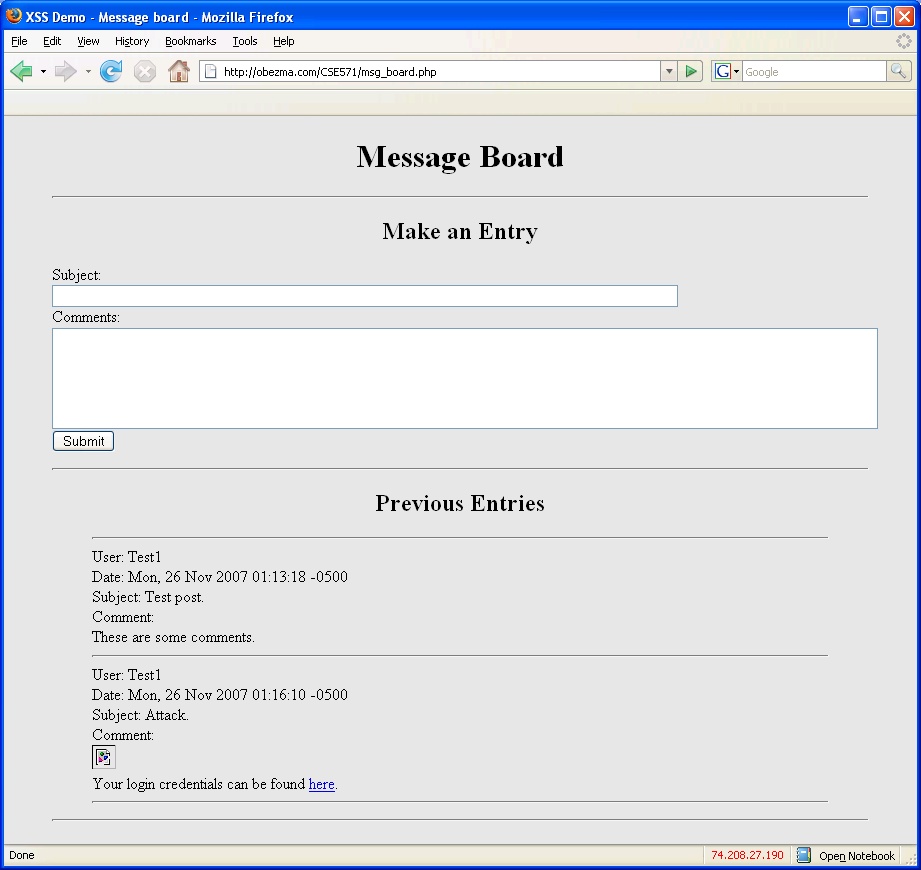
fclose($file);

}

?>

**OUTPUT:**

￼



**RESULT:**

Thus cross site scripting have been successfully implemented.

**IMPLEMENT SQL INJECTION ATTACK**

**AIM:**

To implement sql injection attack.

**SQL INJECTION**:

SQL injection is a code injection technique that might destroy your database.SQL injection is one of the most common web hacking techniques.SQL injection is the placement of malicious code in SQL statements, via web page input

**IMPLEMENTATION:**

## SQL Injection Based on 1=1 is Always True

If there is nothing to prevent a user from entering "wrong" input, the user can enter some "smart" input like this:

UserId:123 OR 1=1

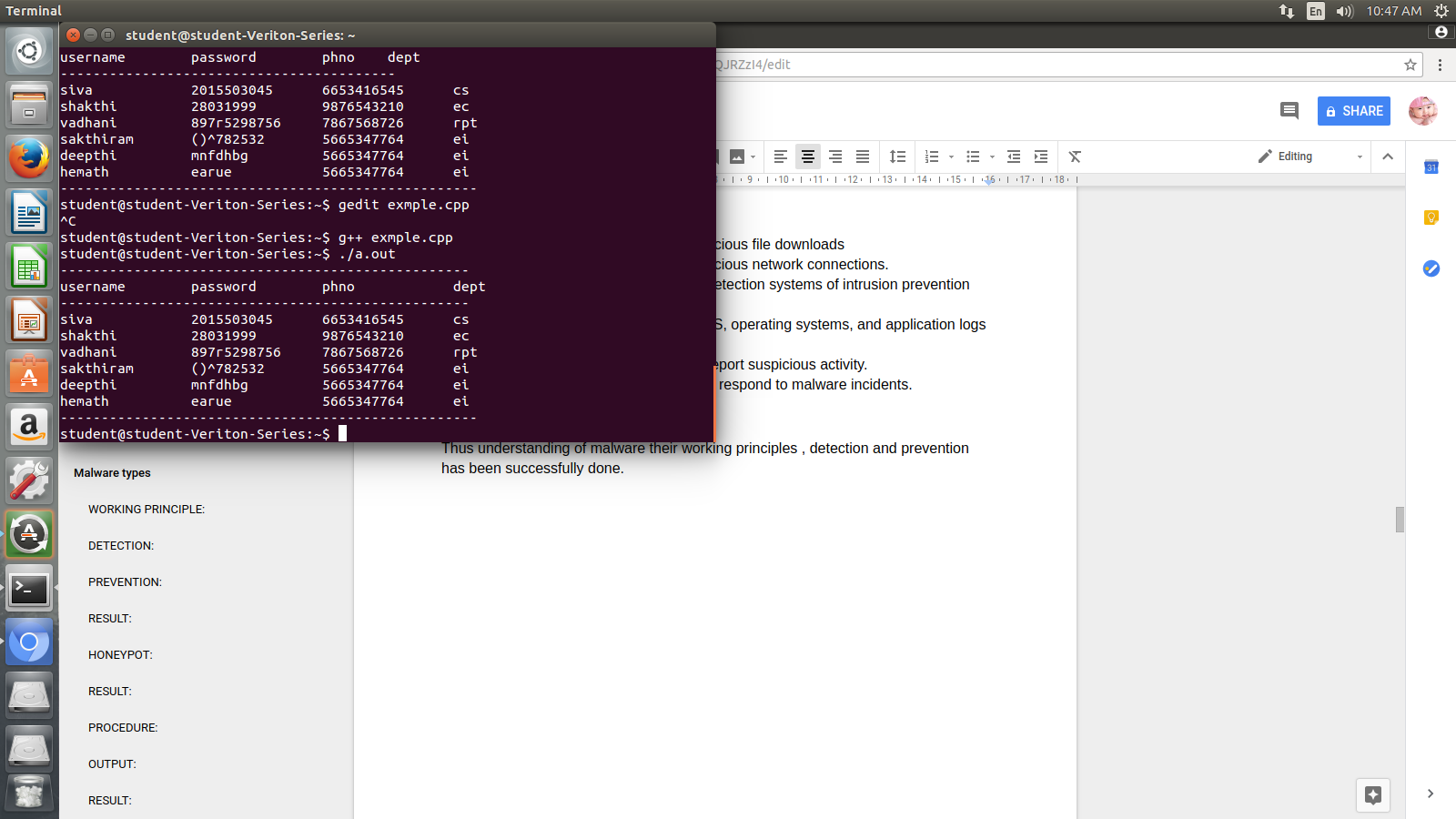
Then, the SQL statement will look like this:

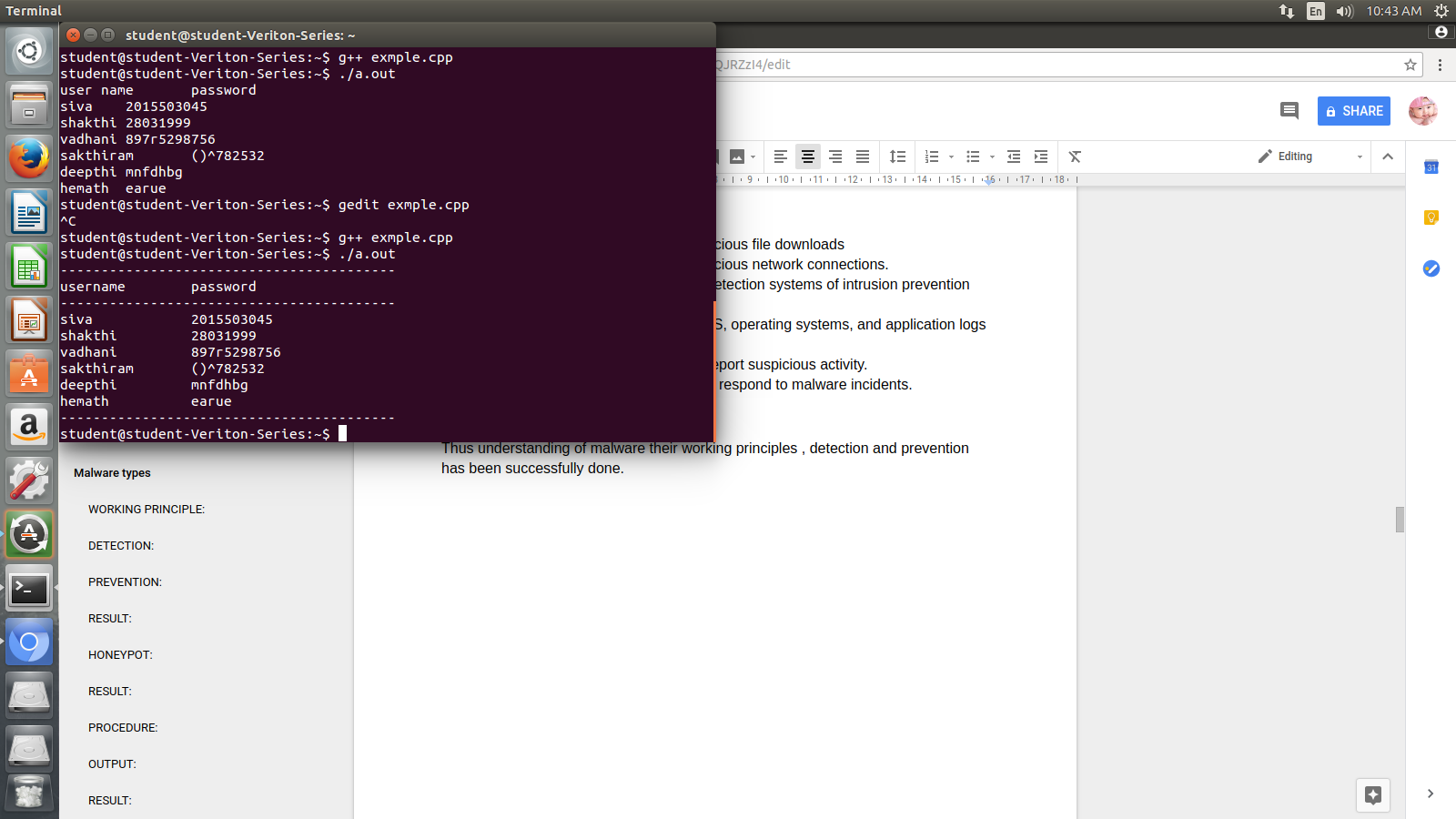
SELECT \* FROM Users WHERE UserId = 105 OR 1=1;

The SQL above is valid and will return ALL rows from the "Users" table, since **OR 1=1** is always TRUE

**OUTPUT:**

Original table





**RESULT:**

Thus the sql injection is implemented successfully.

**IMPLEMENT BUFFER OVERFLOW ATTACK**

**AIM:**

To implement buffer overflow attack

**BUFFER OVERFLOW ATTACK:**

A **buffer** is a temporary area for data storage. When more data (than was originally allocated to be stored) gets placed by a program or system process, the extra data overflows. It causes some of that data to leak out into other buffers, which can corrupt or overwrite whatever data they were holding.

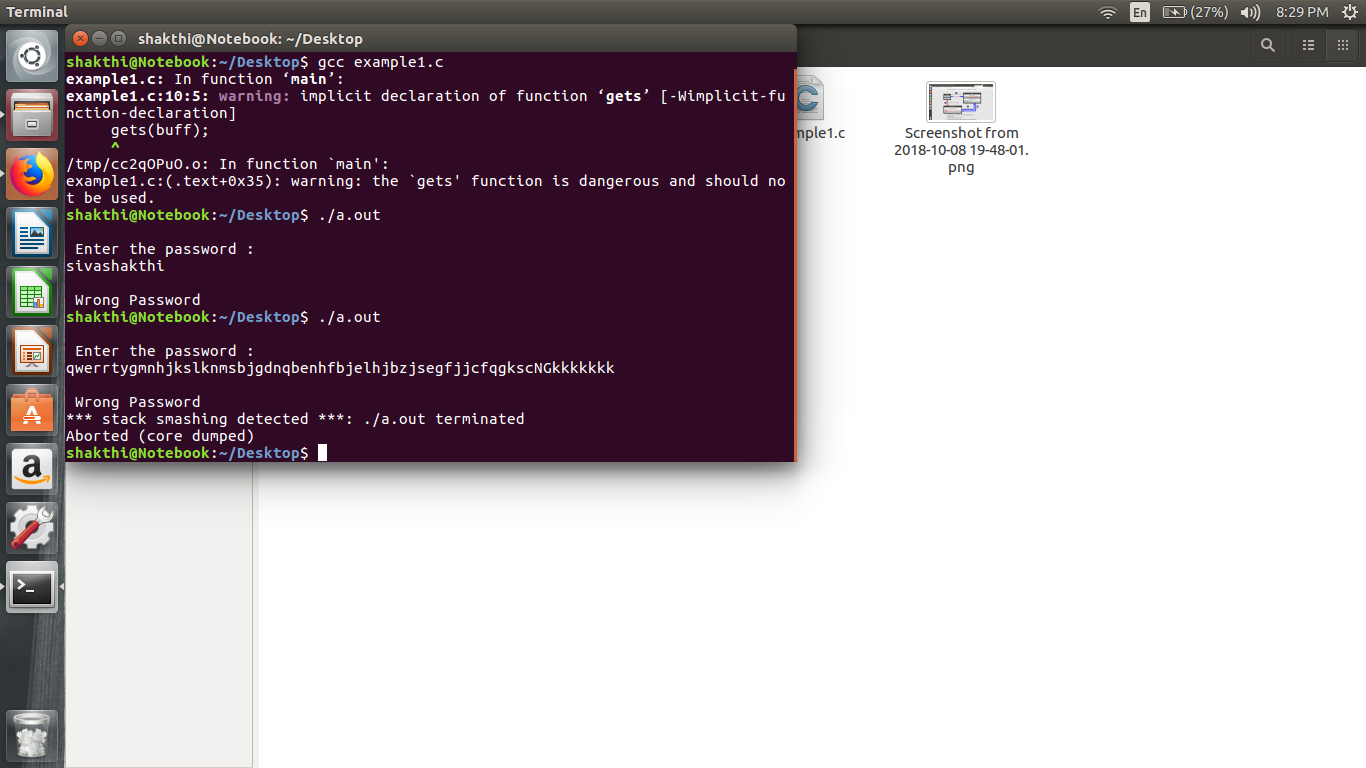
In a **buffer-overflow attack,** the extra data sometimes holds specific instructions for actions intended by a hacker or malicious user; for example, the data could trigger a response that damages files, changes data or unveils private information.

Attacker would use a buffer-overflow exploit to take advantage of a program that is waiting on a user’s input. There are two types of buffer overflows: stack-based and heap-based. Heap-based, which are difficult to execute and the least common of the two, attack an application by flooding the memory space reserved for a program. Stack-based buffer overflows, which are more common among attackers, exploit applications and programs by using what is known as a stack: memory space used to store user input.

**SOURCE CODE:**

#include <stdio.h>  
#include <string.h>  
  
int main(void)  
{  
 char buff[15];  
 int pass = 0;  
  
 printf("\n Enter the password : \n");  
 gets(buff);  
  
 if(strcmp(buff, "thegeekstuff"))  
 {  
 printf ("\n Wrong Password \n");  
 }  
 else  
 {  
 printf ("\n Correct Password \n");  
 pass = 1;  
 }  
  
 if(pass)  
 {  
 /\* Now Give root or admin rights to user\*/  
 printf ("\n Root privileges given to the user \n");  
 }  
  
 return 0;  
}

**OUTPUT:**



**RESULT:**

Thus bufferoverflow attack has been successfully implemented.

**UNDERSTANDING MALWARE WORKING PRINCIPLES,DETECTION AND PREVENTION**

**AIM:**

To understand the malware working principles,detection and principles.

**MALWARE:**

The term “malware” refers to harmful software that disrupts or manipulates an electronic device’s normal operation.1 Malware can infect personal computers, smartphones, tablets, servers and even equipment — basically any device with computing capabilitie

**WORKING PRINCIPLE:**

Malware typically infects a machine by tricking users into clicking and/or installing a program that they shouldn’t from the Internet. When the click or installation occurs, the malicious code executes actions that the user doesn’t anticipate or intend, which could include:

1. Self-replication in different parts of the file system
2. Installing applications that capture keystrokes or commandeer system resources, often running without the user being aware, while slowing the system down considerably
3. Blocking access to files, programs or even the system itself, sometimes forcing the user to make a payment to regain access
4. Bombarding a browser or desktop with ads
5. Breaking essential system components and rendering a device inoperable

Execution can be triggered by a number of user actions, but the most common trigger is a click, typically on a link or pop-up. The descriptions might say something provocative like, “Claim your prize” or “Your account has been compromised. Please log in and verify recent charges.” Many times, a pop-up will be displayed immediately after clicking the link, such as, “Your system is infected! Click here to run a scan.” The next click often triggers the download of a malicious payload, even if the user doesn’t select one of the options and instead tries to close the program using the corner X.

## **Malware types**

Malware can be delivered in several different forms, depending on the intention of the person who developed it.

1. A computer **virus** is designed to reproduce itself and spread from one file or program to another, and, less frequently, to other computers on a network.
2. **Trojan horses** masquerade as harmless programs, but when activated, they damage their host computer. Unlike a virus, a Trojan horse does not replicate itself; instead, this malware usually attempts to steal files or passwords.
3. Computer **worms** replicate themselves to spread through a network. A computer worm will spread across computer networks, as opposed to viruses that usually spread from file to file on a single computer.
4. **Spyware** infects and operates on a user’s computer to monitor user activity and extract information. For instance, while spyware runs on a machine, the hacker can monitor the programs used and sites visited while tracking keystrokes to determine login and password information.3
5. **Logic bombs** are concealed in programs and can either be triggered by a user’s action or released at a predetermined time. They can crash a system or wipe a hard drive.

**DETECTION:**

* Impact the distribution channel by training users [not to click links](https://www.sagedatasecurity.com/blog/how-to-avoid-malware-infection-from-a-phishing-email) embedded in email, open unexpected email attachments, irresponsibly surf the Web, download games or music, participate in peer-to-peer (P2P) networks, and allow remove access to their desktop.
* Configure the firewall to restrict access.
* Do not allow users to install software on company-provided devices.
* Do not allow users to make changes to configuration settings.
* Do not allow users to have administrative rights to their workstations. Malware runs tin the security context of the logged-in user.
* Do not allow users to disable (even temporarily) anti-malware software and controls.
* Disable remote desktop connections.
* Apply operating system and application security patches expediently.
* Enable browser-based controls, including pop-up blocking, download screening, and automatic updates.
* Implement an enterprise-wide antivirus / anti-malware application. It is important that the anti-malware solutions be configured to update as frequently as possible because many new pieces of malicious code are released daily.

**PREVENTION:**

* Real-time firewall detection of suspicious file downloads
* Real-time firewall detection of suspicious network connections.
* Host and network-based intrusion detection systems of intrusion prevention systems (IDS/IPS).
* [Review and analysis of firewalls, IDS, operating systems, and application logs for indicators of compromise](https://www.sagedatasecurity.com/ndiscovery).
* User awareness to recognize and report suspicious activity.
* Help desk (or equivalent) training to respond to malware incidents.

**RESULT:**

Thus understanding of malware their working principles , detection and prevention has been successfully done.

**SETUP HONEYPOT AND MONITOR THE HONEYPOT NETWORK**

**AIM:**

To setup honeypot.

**HONEYPOT:**

A honeypot is a system designed to appear vulnerable to [attackers](https://www.deepdotweb.com/2017/05/27/types-cyberattacks-hitting-dark-web-research-paper/). The goal of a honeypot is to log all the attackers’ activities to study their behaviour, log their Ips, track their location, collect zero-days. The idea of “honeypot” is nothing but a server that offers any kind of services to the attacker, from SSH to telnet, showing several well known exploitable ports opened like 22, 23, 445, 135, 139 and so on. The server appears to have critical vulnerabilities but it actually rejects connections so it is not really exploitable. It could happen that a honeypot is really compromised but this would be the case of a bad configured honeypot and this argument goes beyond the scope of this article

**PROCEDURE**:

Pentbox is a little piece of software that allows you to open a port on your host and listen for incoming connections (eventually refused) from outside.

1 – Download Pentbox:

wget<http://downloads.sourceforge.net/project/pentbox18realised/pentbox-1.8.tar.gz>

2 – Unpack

tar -zxvf pentbox-1.8.tar.gz

3 – Move to the Pentbox’s directory

cd pentbox-1.8/

4 – Run Pentbox

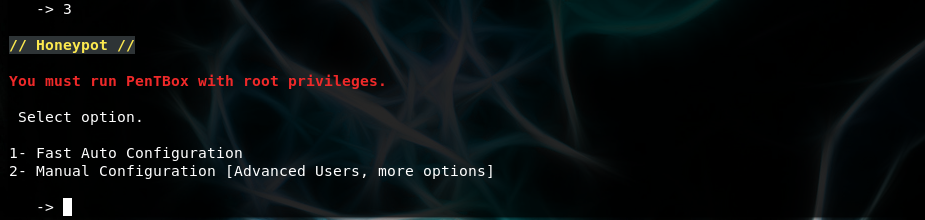
./pentbox.rb



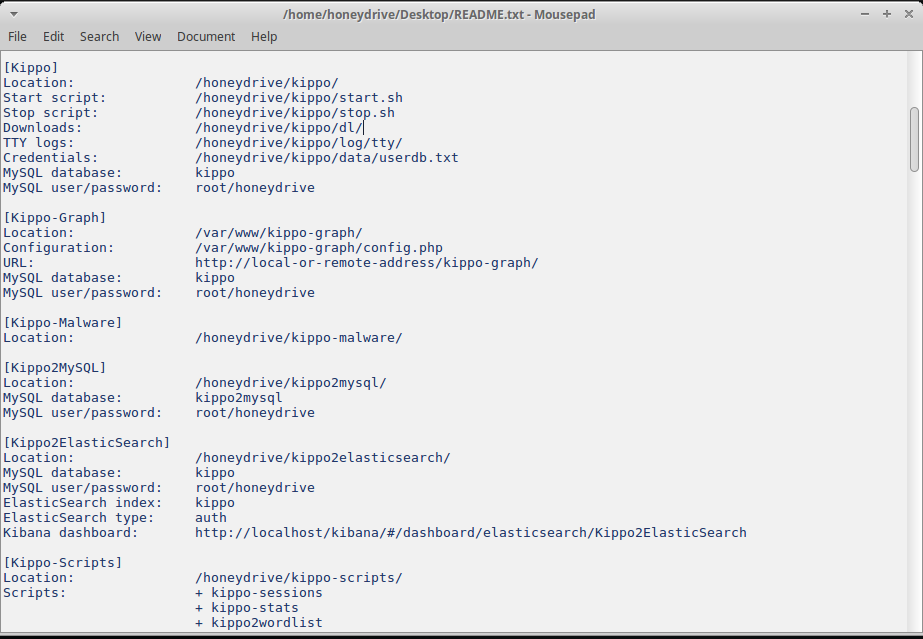
Select 2, then you’ll see the subsequent menu:



Select 3 and fast auto configuration:



Then “honeypot activated on port 80”. Just open your browser, connect to your VM’s IP and you’ll see an “access denied message”, while in your Pentbox terminal you’ll see the attack has been logged successfully



**RESULT:**

Thus honey pot is successfully installed.

**DEMONSTRATE INTRUSION DETECTION SYSTEM USING ANY TOOL**

**AIM:**

To demonstrate intrusion detection using snort tool.

**SNORT TOOL:**

SNORT can be configured to run in three modes:  
1. Sniffer mode  
2. Packet Logger mode  
3. Network Intrusion Detection System mode

Sniffer mode

snort -V Print out the TCP/IP packets header on the screen  
Snort –d show the TCP/IP ICMP header with application data in transit.

Packet Logger mode snort

dev –l c:\log [create this directory in the C drive] and snort will  
automatically know to go into packet logger mode, it collects every  
packet it sees and places it in log directory.  
snort –dev –l c:\log –h ipaddress/24 This rule tells snort that you want to  
print out the data link and TCP/IP headers as well as application data  
into the log directory.  
snort –l c:\log –b This is binary mode logs everything into a single file.

Network Intrusion Detection System

mode snort –d c:\log –h ipaddress/24 –c snort.conf This  
is a configuration file applies rule to each packet  
to decide it an action based upon the rule type in  
the file.  
Snort –d –h ipaddress/24 –l c:\log –c snort.conf  
This will cnfigure snort to run in its most basic  
NIDS form, logging packets that trigger rules  
specifies in the snort.conf

**PROCEDURE:**

1. Download SNORT from snort.org

2. Install snort with or without database support.

3. Select all the components and Click Next.

4. Install and Close.

5. Skip the WinPcap driver installation

6. Add the path variable in windows environment variable by selecting new classpath.

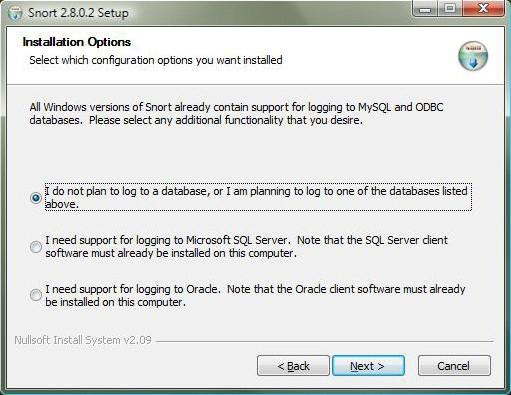
7. Create a path variable and point it at snort.exe variable namepath and variable

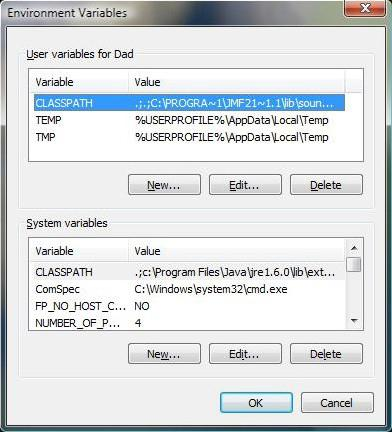
Valuec:\snort\bin.

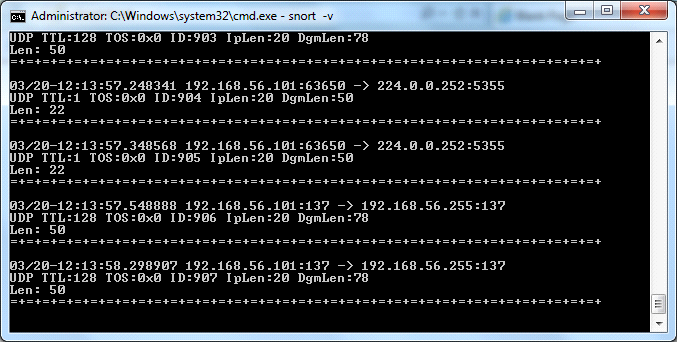
8. Click OK button and then close all dialog boxes.

9. Open command prompt and type the following commands:

**OUTPUT:**







**RESULT:**

Thus intrusion detection system is successfully implemented.