2/6/2024

Marty Ryan

Integrating Network Intrusion Detection System and Neural Networks with Tensorflow

# Abstract

In the world today every company is now a potential for cyberattacks. The global average cost for a data breach in 2023, was $4.45 million, a 15% increase over 3 years. Over 51% of organization are planning to increase security investments as a result of a breach. The average saving for organizations that use AI and automation extensively is $1.76 million compared to the organizations that don’t. **[1]**

Neural Networks are a subset of machine learning and are a huge part of deep learning. Neural networks name and structure are inspired by the human brain, mimicking the way that biological neurons signal to one another. Neural networks rely on training data to learn and import their accuracy over time. **[2]**

Network intrusion detection system are used to examine data exchanged among computers in a network and capture network traffic from the wire as it travels to a host. It can be analysed for a particular signature of for unusual or abnormal behaviours. If any suspicious or malicious behaviour occurs, it triggers an alarm and passes the message to the central computer system or an administrator to take action on this alarm. **[3]**

In this project, we will attempt to link these two systems together, to develop an autonomous system where network traffic is captured using a security information and event manager system and this network traffic is then passed through a neural network which will output if the traffic captured is malicious or normal.

The aim of this project is to develop a system that will lower the costs of network security for organizations and provide a high level of sophistication and accuracy when dealing with network attacks.

Table of Contents

[Abstract 1](#_Toc158112573)

[Introduction 4](#_Toc158112574)

[Machine Learning 4](#_Toc158112575)

[System infrastructure and requirements 4](#_Toc158112576)

[Kali Linux 4](#_Toc158112577)

[Installing LOIC 5](#_Toc158112578)

[Windows server 2022 6](#_Toc158112579)

[Firewall Configuration 6](#_Toc158112580)

[Solarwinds Security Event Manager 7](#_Toc158112581)

[Configure centralized manager server 8](#_Toc158112582)

[Installing Solarwinds SEM agent on VM’s 9](#_Toc158112583)

[Kali Linux Solarwinds agent installation 9](#_Toc158112584)

[Windows server Solarwinds agent installation 10](#_Toc158112585)

[Solarwinds SEM Rules and Dashboard Configuration 11](#_Toc158112586)

[VirtualBox Network Configuration 12](#_Toc158112587)

[Denial of Service attack 13](#_Toc158112588)

[LOIC Denial of Service attack 14](#_Toc158112589)

[Slowloris DoS attack 16](#_Toc158112590)

[TensorFlow 16](#_Toc158112591)

[Keras: The high-level API for TensorFlow 17](#_Toc158112592)

[Keras API Components 17](#_Toc158112593)

[Data set 18](#_Toc158112594)

[Neural Network with TensorFlow steps 19](#_Toc158112595)

[Neural Network code 19](#_Toc158112596)

[Importing Libraries 19](#_Toc158112597)

[Data set pre-processing 20](#_Toc158112598)

[Building and training Neural Network 24](#_Toc158112599)

[Shallow Neural Network 25](#_Toc158112600)

[Build the model 25](#_Toc158112601)

[Compile the model 26](#_Toc158112602)

[Train the neural Network 26](#_Toc158112603)

[Predictions 27](#_Toc158112604)

[Test data set 28](#_Toc158112605)

# Introduction

I believe machine learning will have a major impact on security and how security is implemented in a company. The reason I decided on this project was a personal one, I worked for a IT government entity in Abu Dhabi which had a lot of clients, in this company there was a SOC department which monitored many networks in the region. This was a 24/7 operation and as a virtualization systems engineer, we would get calls at all hours from this team. Then I started to wonder why are they there at this time, is technology not advanced enough to have autonomous systems, could they not check a report or a confusion matrix in the morning instead of working 24/7. How accurate would a neural network algorithm be monitoring alerts? Is it trust worthy? Would the company benefit financially from this?

I wanted to show what I have learned during this Masters, I will be performing DOS attacks from a VM to gather network traffic on a security information and event manager, which will include information systems and security module, next I will code the neural network in python and finally I will design the neural network which integrates machine learning.

# Machine Learning

Machine learning can be classified into three main categories, supervised, unsupervised and reinforcement learning. In this project we will be using supervised learning.

Supervised machine learning is used for making predictions from data. We need to know what we are trying to predict this is known as the target variable (Y). In this case we are trying to find the Attack Type which is our dependent variable. The data sets where the target variable is known are called labelled datasets, which we use to train algorithms (neural networks) to properly categorize data or predict outcomes.

Supervised learning deals with two distinct problems, classification and regression.

# System infrastructure and requirements

## Kali Linux

Kali Linux is an open source, Debian-based Linux distribution geared towards various information security tasks, such as Penetration Testing, Security Research, Computer Forensics and Reverse Engineering. **[4]**

A prebuilt Kali Linux will be used to perform a DOS (denial of service) attack to a victim machine. To aim of this task is to target a victim target and have the network captured by a SIEM.

The Kali Linux iso file will be downloaded from the official Kali Linux website <https://www.kali.org/get-kali/#kali-virtual-machines> and imported into Virtualbox.

Kali Linux VM system is 3 CPU’s, 20 GB’s VHD, 10GB of memory.

To perform the DOS attack, we will need a stresser tool, after some consideration between a Slowloris attack, HPing3 and LOIC(Low Orbit Ion Cannon), I have decided to work with LOIC.

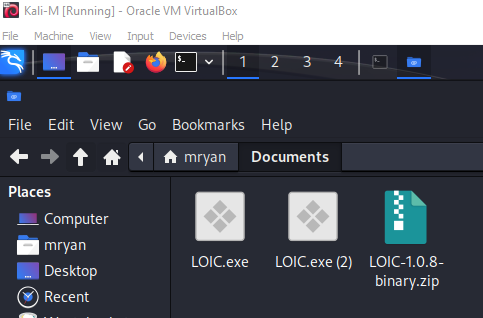
LOIC is a tool commonly used for DDOS and DOS attacks. It is open-source and mostly used with malicious intent. It is known for being very user friendly and accessible tool, it has been used by members of Anonymous. One user using LOIC can generate enough junk traffic to make a serious impact on a target, they can create a “Hivemind” version of LOIC which lets one user control several networked secondary computers. This is popular approach because owners of the secondary devices can claim they were innocent victims of the involuntary botnet. **[5]**

It works by flooding a target server with TCP, UDP or HTTP packets with the goal of disrupting service.

### Installing LOIC

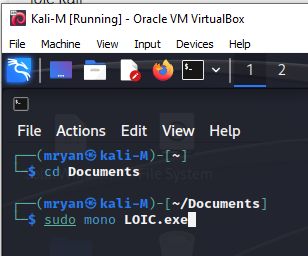
To install LOIC we first download the package from the SourceForge Website. **[6]**

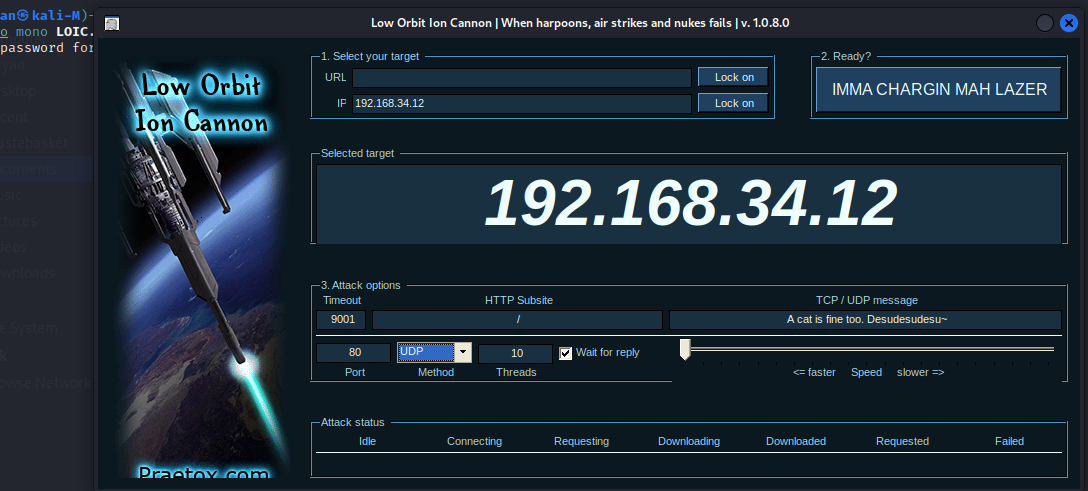
Create a folder on your Kali Linux Documents folder and export the files from LOIC zip folder to this location in **Figure 1**

 **Figure 1**

Next we update the database on the Kali Linux VM, using the command “ sudo apt-get update” and enter our administrator password. This is used to update the package index files on the prebuilt Kali system, which contain information about available packages and their versions.

To run LOIC we need to install mono-complete. Mono is an open-source development platform based on the.NET Framework, allows developers to build cross-platform applications, it contains a C#Complier, class library and Mono runtime. Mono is used to launch the application from terminal windows. We can install Mono from the terminal using “sudo apt install mono-complete”. When mono install is complete we can open the LOIC application using the command seen in **Figure 2**.

**Figure 2**

 **Figure 3**

In **Figure 3** we can see the LOIC application UI. To perform a DOS attack on a victim VM, we enter the victim’s IP address, select Lock on, select our port and method type (TCP, UDP, HTTP) and select IMMA CHARGIN MAH LAZER.

## Windows server 2022

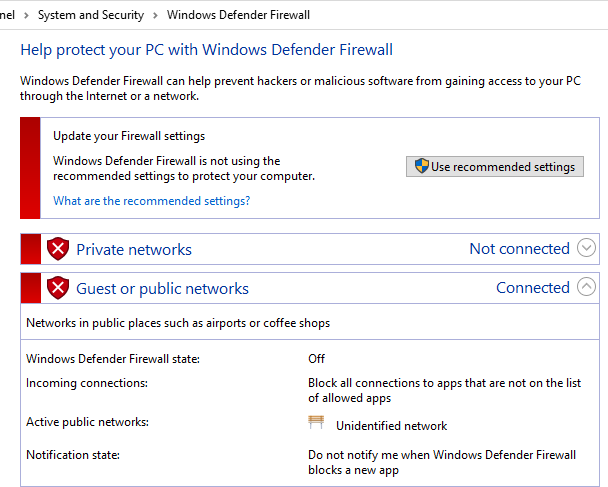
Our victim machine will be a Windows server 2022. We download the windows server iso file from the Microsoft evaluation center **[7]**

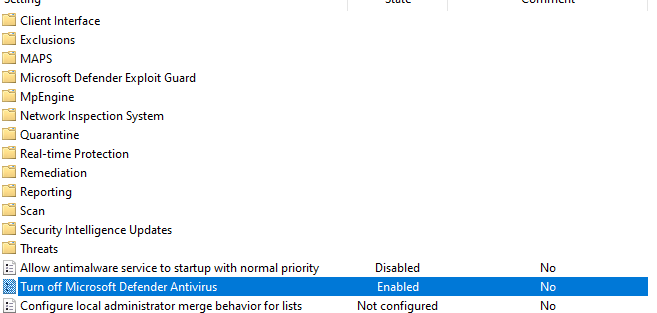
Import the iso file into Virtualbox and configure all system hardware settings. The windows server will be assigned the minimum required hardware.

System hardware is 1 CPU, 25GB VHD, 4GB of memory.

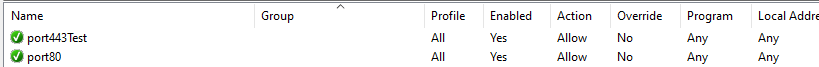
### Firewall Configuration

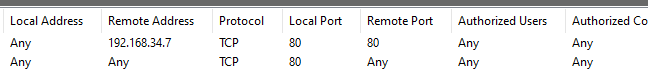
To make the Windows server vulnerable to a DOS attack, I turned off the local windows defender firewalls through the UI and local Group policy, see in **Figure 5** and **Figure 6**.

**Figure 5**

 **Figure 6**

As an extra step I configured an inbound firewall rule, to allow any any protocol over 80 from any IP address. **Figure 7** and **Figure 8**

**Figure 7**

**Figure 8**

These configurations make the windows server vulnerable to a DOS attack.

## Solarwinds Security Event Manager

Solarwinds Security event Manager (SEM) is built to make it easy to collect and correlate logs data from tens of thousands of devices. It is used to help improve security, flag threat event, automate detection with threat intelligence frameworks. Solarwinds SEM will be the Network Intrusion Detection System in this project.

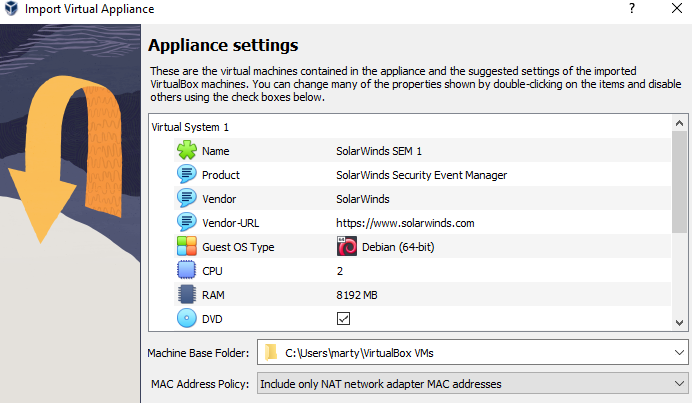
It has to the capabilities to collect, monitor and analyse real time log data from routers, switches, servers, applications and user endpoints. It is designed to correlate system and user activities to reconstruct a compliance violation or mitigate an emerging security threat. A feature will we be using is the activate threat intelligence management system in the form of threat feeds designed to automatically detect and respond to user, application and network threats.

In this project Solarwinds Security Event Manager will be used to monitor the network traffic between the master (Kali Linux) and the victim (Windows server). The captured network traffic will be exported to a CSV file, pre-processed and passed to our neural network.

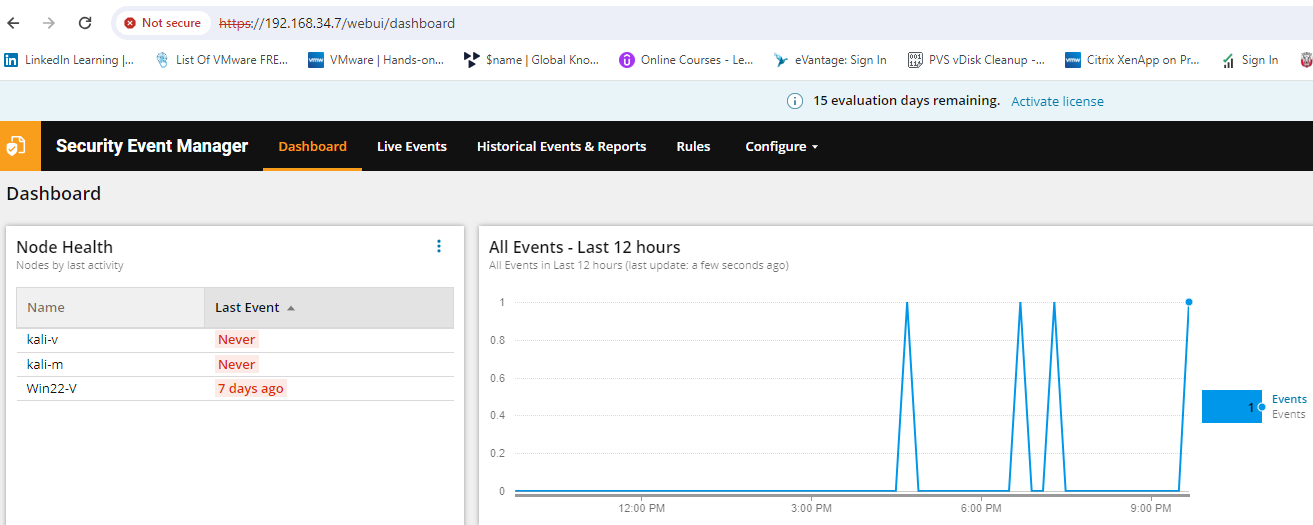
### Configure centralized manager server

A 30-day trial fully functional version of Solarwinds SEM is available to download as an OVF file. **[8]**

After the OVF file is downloaded, it is imported into Virtualbox, **Figure 9**

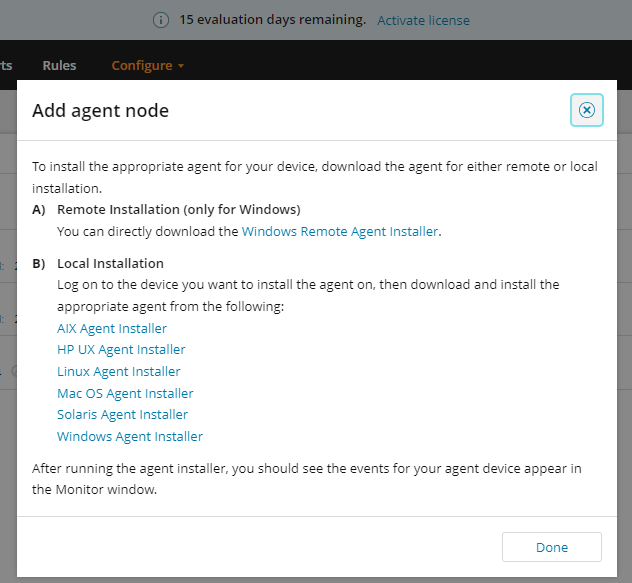
 **Figure 9**

Start the Solarwinds VM and access the graphical user interface through our local browser, **Figure 10**

 **Figure 10**

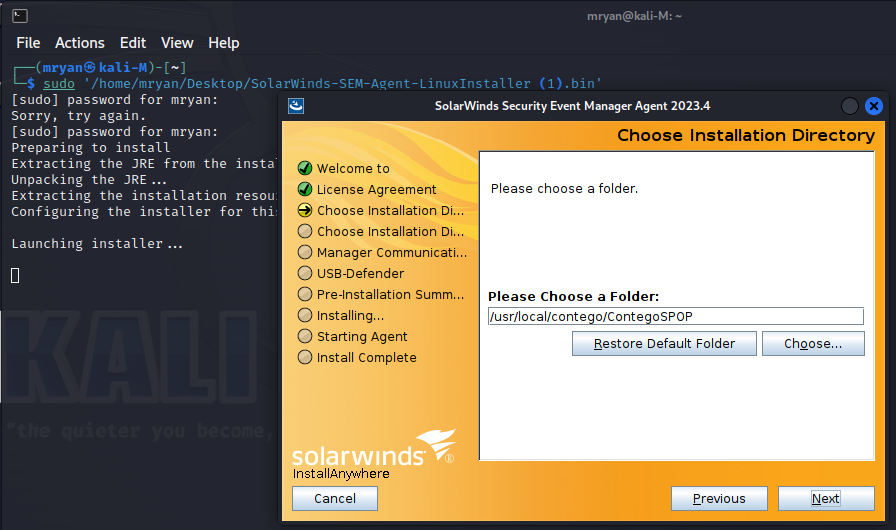
### Installing Solarwinds SEM agent on VM’s

The installer files for the Solarwinds agent is on the SEM server, we download and install for Linux and Windows, **Figure 11**

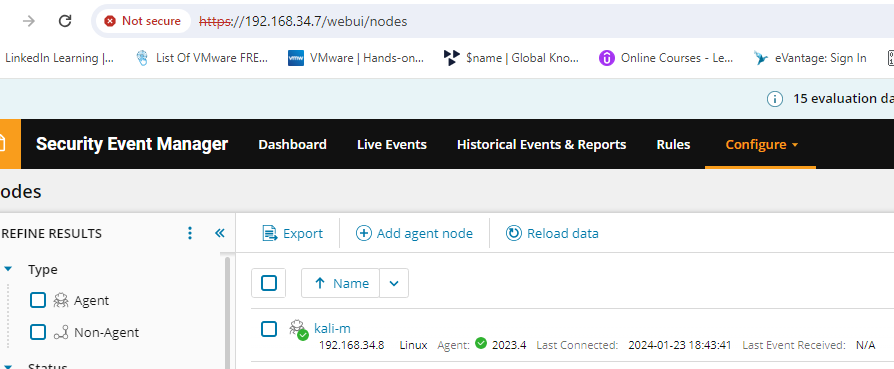
 **Figure 11**

### Kali Linux Solarwinds agent installation

The Linux installer is downloaded and moved to a folder on the desktop of the Kali Linux through drag and drop, it’s run using the “sudo” command. **Figure12**

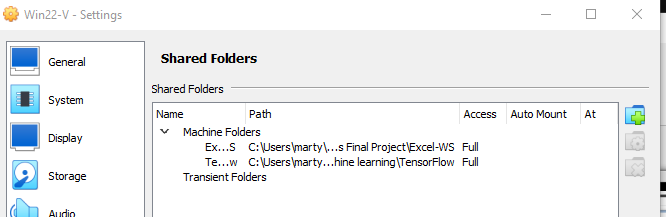
**Figure 12**

Folder directory is by default, a required step is to enter our Solarwinds SEM centralized server IP address. Installation completed and the node appears in the SEM centralized server. **Figure 13**

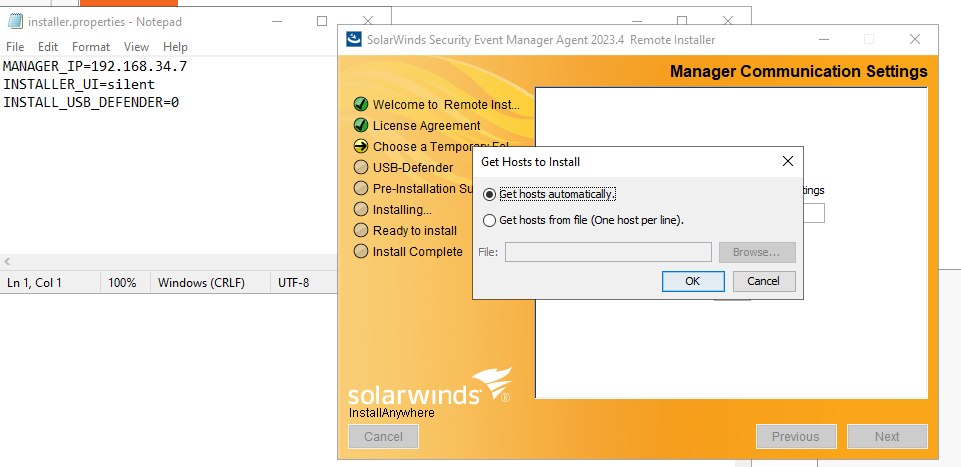
 **Figure 13**

### Windows server Solarwinds agent installation

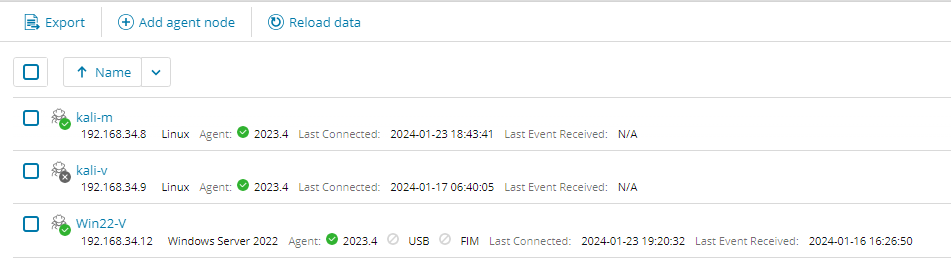
Drag “n” Drop feature would not work for windows so I created Share Folders on the Virtualbox side to map a folder from my local disk to the Windows VM to access the Solarwinds agent installer., **Figure 14**

 **Figure 14**

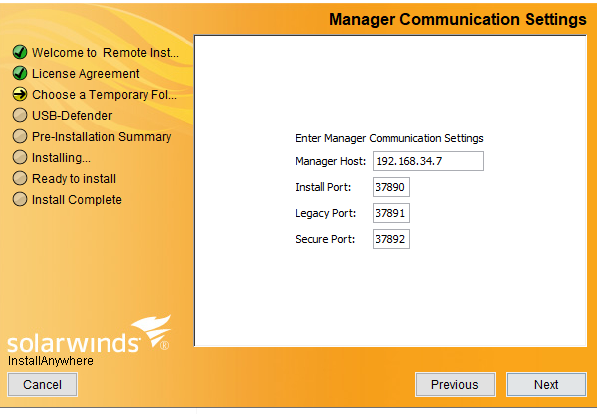
During this installation I had to create a silent installer file and a host file with the windows server IP address. **Figure 15**

**Figure 15**

The Windows server shows on the Solarwinds SEM server after installation. **Figure 16**

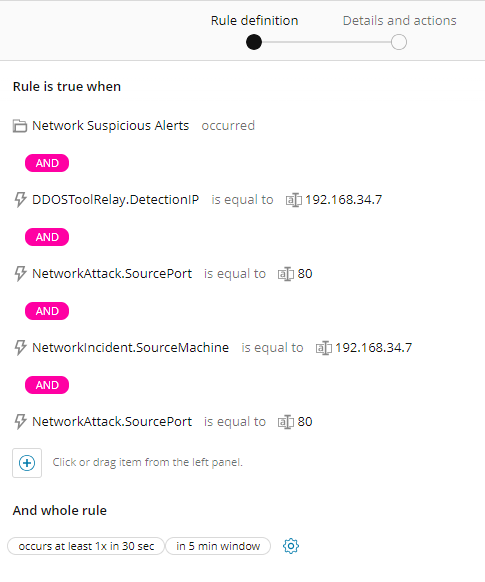
 **Figure 16**

Solarwinds SEM server management communication settings when installing the agent. **Figure 17**

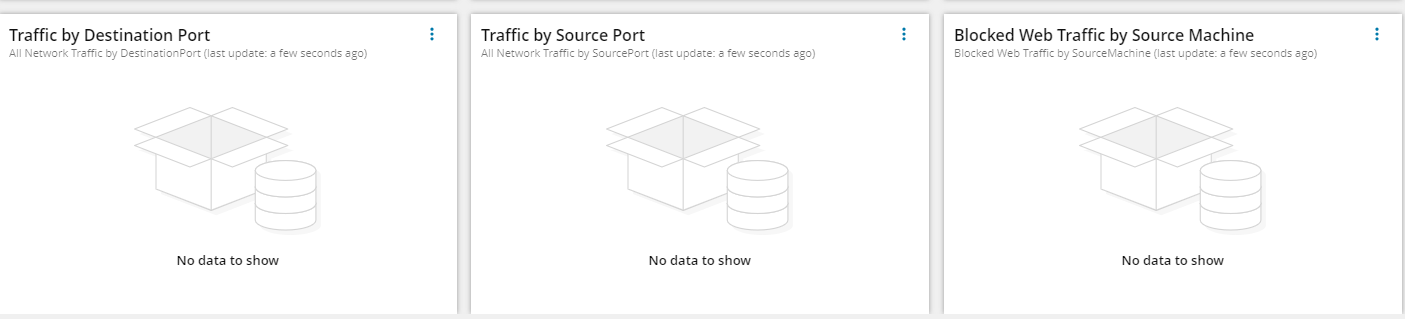
 **Figure 17**

### Solarwinds SEM Rules and Dashboard Configuration

To create alerts in Solarwinds SEM for unusual network activity I, have created rules on the SEM GUI page. **Figure 18**

**Figure 18**

To view the network traffic from the Kali Linux VM (master) to the Windows Server VM (victim), I added dashboards to the Solarwinds SEM GUI, this would allow me to monitor the traffic in real time. **Figure 19**

**Figure 19**

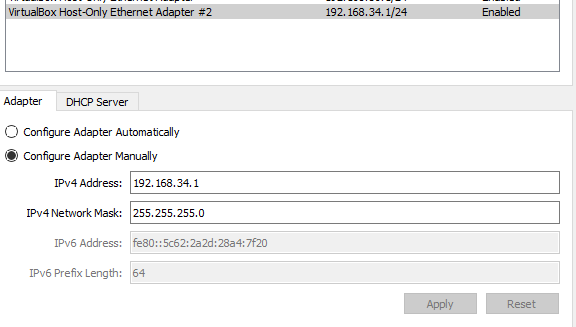
## VirtualBox Network Configuration

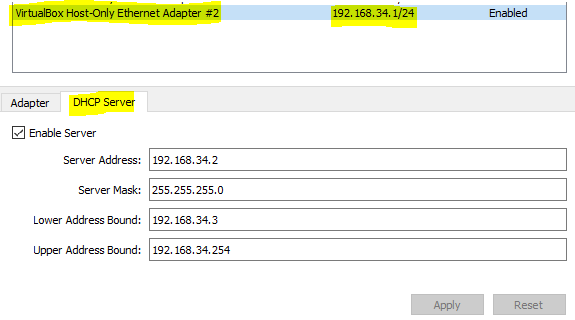
To allow all VM’s to communication, I implemented a Host-only network, which can be thought of as a hybrid between bridged and internal networking modes.

With bridged networking, the virtual machines can talk to each other and the host as if they were connected through a physical Ethernet switch, as with Internal networking, a physical interface does not need to be present and the virtual machines cannot talk to the world outside since they are not connected to a physical networking interface. **[9]**

Host-only networking is particularly useful for preconfigured virtual appliances, where virtual machines are shipped together and designed to cooperate. **[9]**

A host-only network allowed me create a DHCP server which would automatically assign an IP address in a configured range. **(192.168.34.1/24) Figure 20**

**Figure 20**

**Figure 20**

Having this network configuration did bring its challenges, if I wanted to access the internet to download software or apply some updates, I would have to shut the virtual machine down, change the network configurations to NAT (network address translation), start up the virtual machine, download what was required then shut the virtual machine again and change the network settings back to host-only on the virtual machine network setting and start the machine.

## Denial of Service attack

A Denial-of service (DoS) attack is an attack meant to shut down a machine or a network, making it inaccessible to its intended users. DoS attacks accomplish this by flowing the target with traffic or sending it information that triggers a crash. DoS attacks do not typically result in theft or loss of significant information, they can cost the victim a great deal of time and money to handle. **[10]**

Groups like Anonymous perform DoS attacks, to disrupt a victim’s services and make them unusable, like media outlets or social media websites. Additional type of DoS attack is the Distributed Denial of Service (DDoS) attack. A DDoS attack occurs when multiple systems orchestrate a synchronized DoS attack to a single target. This references to the “Hivemind” with LOIC, a DDoS LOIC attack uses many “compromised” devices to perform a DDoSattack. **[10]**

The essential difference is that instead of being attacked from one location, the target is attacked from many locations and provides many advantages such as: **[10]**

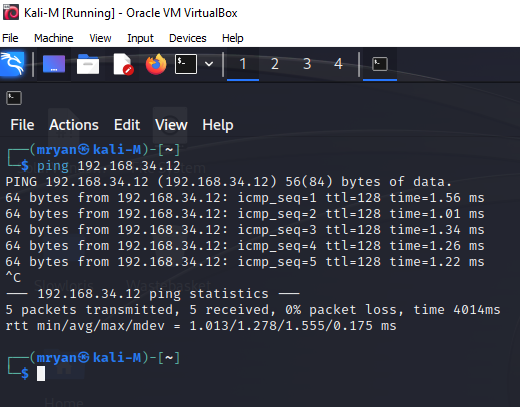
* The attacker can leverage the greater volume of computational power to execute a seriously disrupted attack.
* It is difficult to shut down multiple machines
* The location of the attack is difficult to detect due to the random distribution of attacking systems

In 2020 Amazon was hit with the largest DDOS attack ever recorded. An attack with a previously unseen volume of 2.3 terabytes-per-second, the attack last three days. If the attackers were successful it would have had a devastating effect on all amazon services which would have not only effected Amazon but millions of users of their cloud services. **[11]**

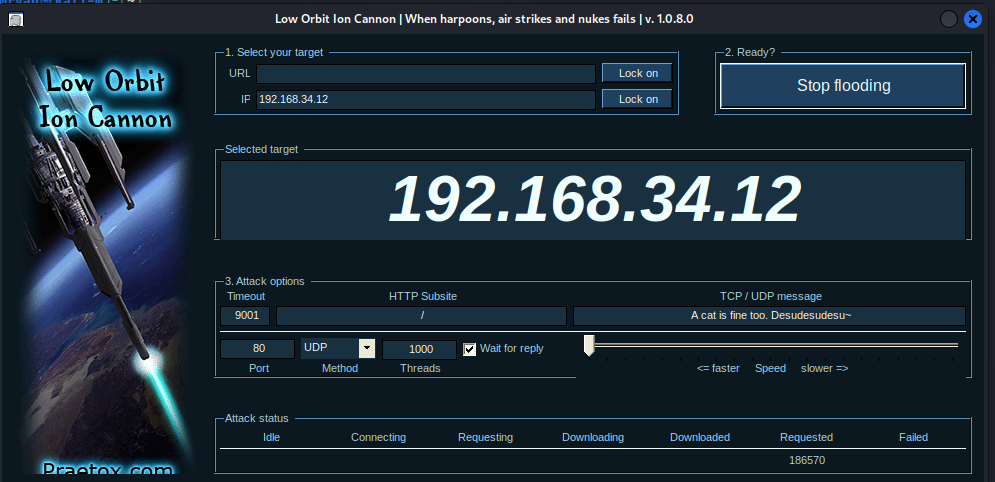
The attack was a so called Connection-Less Lightweight Directory Access Protocol reflection-based attack. A CLDAP reflection attack involved an attacker sending a CLDAP request to a LDAP server with a spoofed IP address to the target’s IP address. The server mounts a bulked-up response to the targets IP address, causing the reflection attack. **[11]**

## LOIC Denial of Service attack

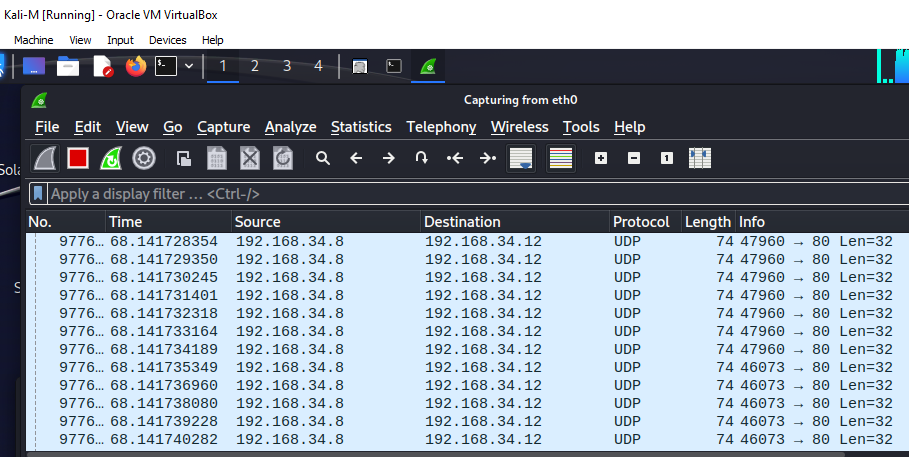
To attack the Windows server virtual machine from kali Linux virtual machine on VirtualBox, first we check if the machines can talk to each other, we do a ping test. **Figure 21**

 **Figure 21**

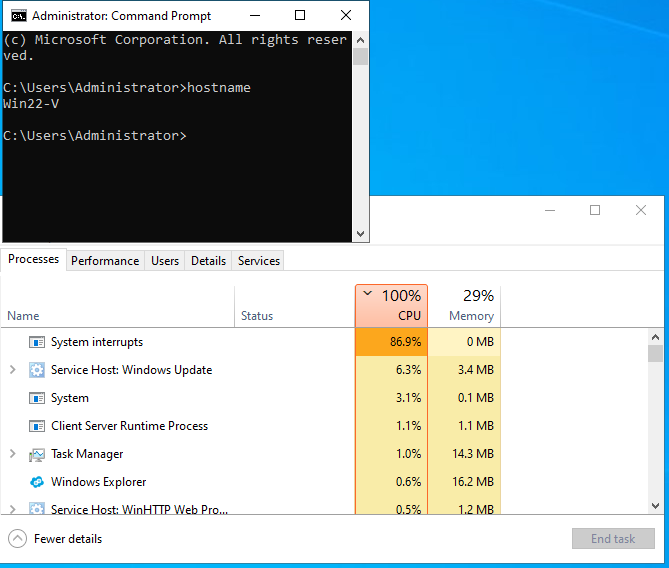
Launch LOIC application on the Kali Linux machine from the terminal. We then decide what port and method (TCP, UDP, HTTP) we will attack. For this attack we will flood port 80 over UDP. **Figure 22**

 **Figure 22**

For a successful attack from the LOIC side, the requested tab on the UI should start to go up. We can also check from Wireshark if the packets have been successfully sent to the windows machine. **Figure 23**

**Figure 23**

Now we check the Windows server task manager to see if the attack was successful. If the attack was successful there should be a sharp jump on the machines resources and the virtual machine should not be usable. **Figure 24**

 **Figure 24**

The attack was successful. This attack would not register on the Solarwinds security event manager, I tried two more DoS attacks to register malicious traffic on the SEM server, using Slowloris and HPing3 but they were unsuccessful.

### Slowloris DoS attack

Slowloris can be used to perform DDoS attacks on any webserver. The Slowloris application was downloaded from github and installed on my Kali Linux virtual machine prior to attack **[12].** For the Slowloris attack I added the Web Server IIS role on the windows server and added a local website on port 80. This attack was not recorded on the Solarwinds SEM server.

## TensorFlow

TensorFlow is a software library, developed by Google Brain Team within Google’s Machine Learning Intelligence research organization, for the purpose of conducting machine learning and deep neural research. **[13]**

TensorFlow combines the computational algebra of compilation optimization techniques, making easy the calculation of many mathematical expressions where the problems is the time required to perform the computation. **[13]**

The main features include:

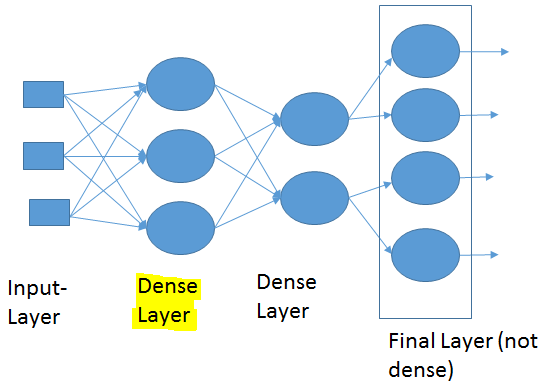
* Defining, optimizing and efficiently calculating mathematical expressions involving multi-dimensional arrays (tensors)
* Programing support of deep neural networks and machine learning techniques.
* Transparent use of GPU computing, automating management and optimization of the same memory and the data used. You can write the same code and run it either on CPU’s or GPU’s. TensorFlow will figure out which parts of the computation should be moved to the GPU.
* High Scalability of computation acrss machines and huge data sets. **[13]**

### Keras: The high-level API for TensorFlow

Keras is the high level API of the TensorFlow platform. It provides an approachable, highly-productive interface for solving machine learning problems, with a focus on modern deep learning. Keras covers every step of machine learning workflow, from dta processing to hyperparameter tuning to deployment. It was developed with a focus on enabling fast experimentation. **[14]**

### Keras API Components

The core data structures of Keras are layers and models. A layer is a simple input/output transformation and a model is a directed acyclic graph (DAG) of layers. An example of a layer used in this project is a Dense layer. This is a fully connected layer to another layer in a neural network. **[14] Figure 25 [15]**



A model is an object that groups layers together and that can be trained on data. The simplest type of model is the Sequential model, which is a linear stack of layers. Sequential model uses the Dense layer to build the model.

The tf.keras.Model class features built-in training and evaluation models that are used through this project:

* Tf.keras.Model.fit: Trains the model for a fixed number of epochs
* Tf.keras.Model.predict: Generates output predictions for the input samples.
* Tf.keras.Model.evaluate: Returns the loss and metrics values for the model, configured via the tf.keras.Model.compile method **[14]**

## Data set

As we could not get network traffic to register on our Solarwind’s Security Event Manager, we will use the kddcup data (kddcup.data.gz) set, which contains a standard set of network traffic data to be audited, which includes a wide variety of intrusions simulated in a military network environment. **[16]**

This data set was used for The Third International Knowledge Discovery and Data Mining Tools Competition, the competition was to build a network intrusion detection, a predictive model capable of distinguishing between intrusions/attacks and good/normal connection. **[17]**

There are 4,898,430 total entries in this data set, 3,883,370 are DoS (Denial of Service) attacks, 41,102 are Probe (probing attacks), 1126 are R2L (root to local), 52 are U2R (User to Root) and 972,780 are registered as Normal. To make it easier on my machine and to speed up the process I will be using the data set which contains 10% of the full data set (kddcup.data\_10\_percent)**[22].** The data set entries are as follows Total entries 494,020, DoS 391,458, Probe 4107, U2R 52, and Normal 97,277. **[18]**

* Denial of Service: is an attack which disrupts a service through various stressing tools
* Root2Local: unauthorized access from a remote machine
* User2Root: unauthorized access to local superuser (root) privileges
* Probe: surveillance and invasive methods to bypass security measures

Type of the attacks in the data set: **[19]**

|  |  |
| --- | --- |
| **Attack Class** | **Attack type** |
| DoS | Back, land, Neptune, pod, teardrop, slowloris, LOIC |
| R2L | ftp-write, guess\_passwd, spy, imap |
| U2R | Buffer\_overflow, loadmodule,perl, rootkit |
| Probe | Ipsweep, nmap, portsweep, satan |

The data set contains 42 columns (features) and 23 different attack entries which will be assigned to the four attack classes mentioned above and added to a new feature called Target.

## Neural Network with TensorFlow steps

Preparing your data for the Neural Network is pivotal, below are the steps we will take to deliver the best possible results using our data set in a Neural Network.

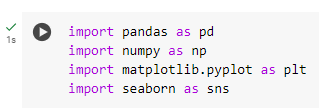
1. Pre-processing the data
2. Visualizing the data
3. Remove highly correlated columns
4. Label encoding the features
5. Building and raining a Neural network
6. Testing the Neural Network

## Neural Network code

### Importing Libraries

All our code is written in python and ran in the google colab environment.

First we import the libraries for pre-processing, visualizing and data handling. **Figure 26**

 **[22]** **Figure 26**

Pandas is a python library used for working with data sets. It has functions for analysing, cleaning, exploring and manipulating data. Panadas can clean messy data sets and make them readable and relevant. **[20]**

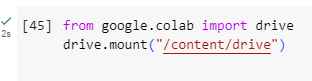
Numpy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform and matrices. Numpy aims to provide an array object that is up to 50 times faster than traditional Python lists. **[21]**

Matplotlib.pyplot will allow us to plot and visualize our data in different forms such as bar graphs or histograms

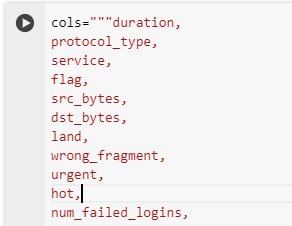
Seaborn will be used to implement a heatmap that will show the correlation between the features and we will be able to perform some pre-processing on these results, removing highly correlated features.

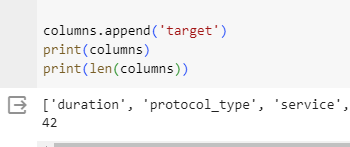
## Data set pre-processing

To allow google colab to access the data set, I connect my google drive to the google colab environment. **Figure 27**

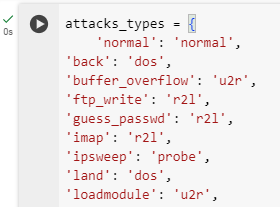
 **Figure 27**

Next we are appending columns to the data set and adding a new column called “target” to the dataset**. [22]** **Figure 28 and Figure 19**

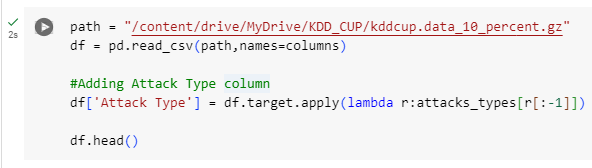
 **Figure 28**

 **Figure 29**

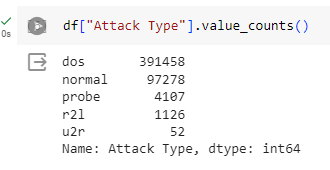
Create a dictionary of attack types, assigning the different entries to our attacks classes. **[22**] **Figure 30**

 **Figure 30**

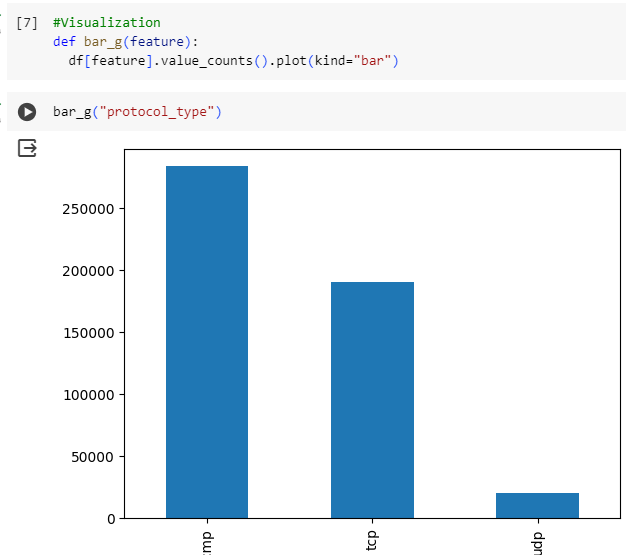
Using Pandas we can read the dataset, kddcup.data\_10\_percent.gz in our google drive and add the Attack Type feature to the dataset, where attack type feature has values of DoS, u2r, r2l, probe and normal values**. [22] Figure 31**

**Figure 31**

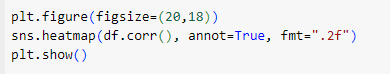
Print the Attack Type column and the number of entries per value (DoS, noemal, probe, u2r, r2l) **[22]** **Figure 32**

**Figure 32**

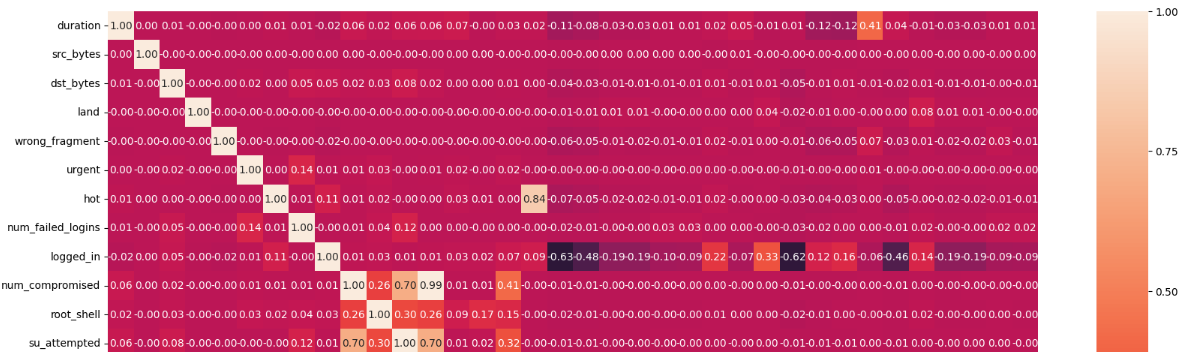
Next we visualize our data in a bar graph using matplotlib.pyplot (plt), we can pick a feature and print out the results, such as protocol type and analyse the results. Visualizing allows us to get a better understanding of the data. **[22]** **Figure 33**

 **Figure 33**

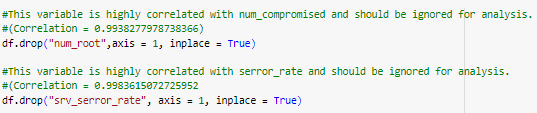
Next we use matplotlib.plyplot and seaborn to print out a heatmap which shows what features are highly correlated. **[22]** **Figure 34**

 **Figure 34**

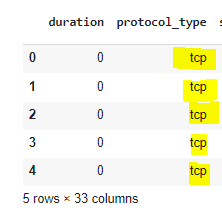
Correlation heatmaps are a type of plot that visualize the strength of relationships between the features. Each feature is represented by a column and the rows represent the relationship between each pair of features. Above 0.8 suggests highly correlated. Light colouring represents a high correlation and dark colouring represents a low correlation. When a feature is run against itself it will return a 1. **Figure 35**

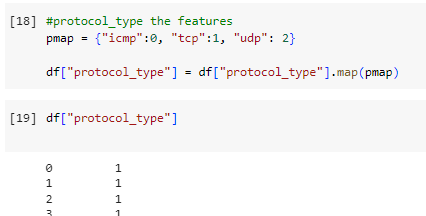
**Figure 35 [22]**

Remove highly correlated features. **[22]** **Figure 36**

 **Figure 36**

Preparing the data set for the Neural Network requires label encoding on some features. Encoding is required on features which have categorical values and convert them to numerical vales. A Neural network cannot accept stings or categorical data. In **Figure 37,** the protocol feature has entries “tcp”.In **Figure 38** the entries icmp, tcp and udp are all assigned numerical values. The feature Protocol\_type is printed in **Figure 38** ad we can see the values have been assigned. **[22]**

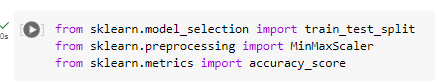
**Figure 37**

**Figure 38**

## Building and training Neural Network

Now our dataset is ready, we start to build our Neural network.

Import sklearn libraries **[22] Figure 39**

 **Figure 39**

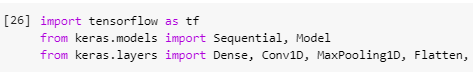
Train\_test\_split is used to split our dataset into a train dataset and test dataset. We train our Neural network on the train dataset and then test how accurate our neural network is by running the test dataset. 70% of the dataset will be the train dataset and the remaining 30% will be the test dataset.

MinMaxScaler scales the data, where the minimum of a feature is made equal to zero and the maximum feature is equal to one. MinMaxScaler shrinks the data within a given usually between 0 and 1. **[23]**

Accuracy score tells us how accurate our Neural Network is and if the accuracy score is low then we need to look at making changes to our design or dataset.

Import TensorFlow and Keras libraries **Figure 40**

These libraries allow us to build, fit, predict and evaluate our model.

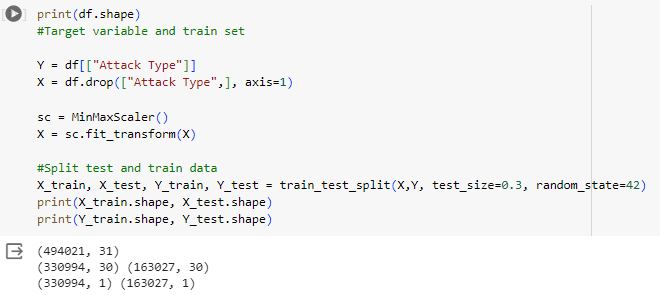
 **Figure 40**

Assigning X and Y variable to our features. Attack Type is assigned Y because it what we are trying to find, it is our dependent variable.

The rest of the dataset is assigned to X (independent varibles)

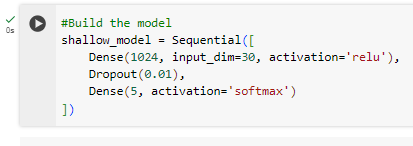
We perform MinMaxScaler on our X variable which is all the features excluding Y, attack Type.

Split the dataset, into train and test datasets, with a 70/30 split, using sklearn’s train\_test\_split. random\_state keeps the split the same every time we run this code. The new dataset shapes are printed. **[22] Figure 41**

**Figure 41**

## Shallow Neural Network

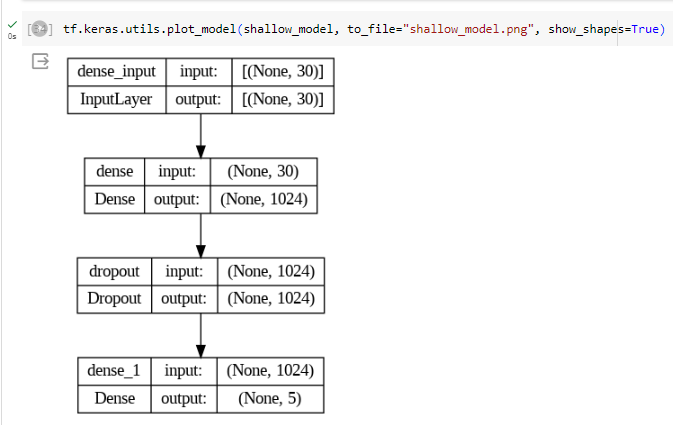
### Build the model

 **Figure 42 [22]**

In **Figure 42,** we have built a shallow model with one input layer, one hidden layer and one output layer. The hidden layer consists of 1024 nodes, the input shape is outlined using “input\_dim = 30”, 30 represents the number of features we will feed into the Neural Network. ReLu is the activation function used, (0,max). The term dropout refers to dropping out the nodes in neural network. All the forward and backwards connections with a dropped node are temporarily removed. Thus creating a new network architecture out of the parent network. Dropout is used to help with overfitting. **[22]**

The output layer is a Dense fully connected layer with 5 nodes (same of number of attack types) and softmax is the chosen activation function. Softmax scales numbers/ logits into probabilities. **[22]**

Model architecture **Figure 43 [22]**

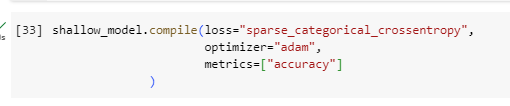


### Compile the model

To compile the model we define our Loss function which is used to find error or deviation in the learning process.

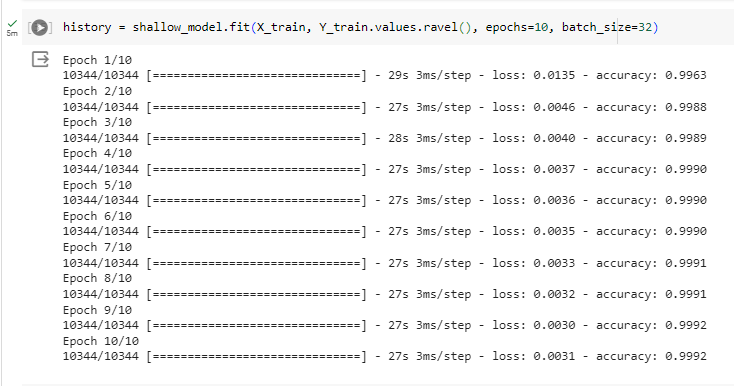
We define our optimizer which optimizes the input weights by comparing the prediction to the loss function.

The metrics hyperparameter is used to evaluate the performance of the model **[22] Figure 44**

 **[22]** **Figure 44**

### Train the neural Network

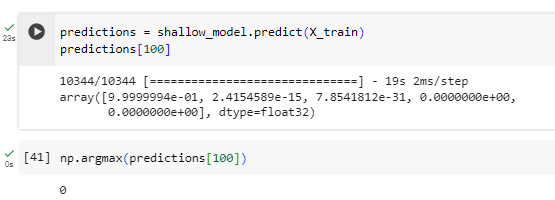
Now we run the train data set (X\_train, Y\_train) through our model. The data is feed through the network in batches of 32 and it will run through 10 times and gives us a loss and an accuracy output. **[22]** **Figure 45**

**Figure 45**

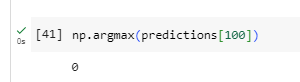
The model is very accurate on the train dataset with a 99.92% accuracy.

### Predictions

Run a prediction on our Neural network accuracy, using X\_train. **Figure 46**

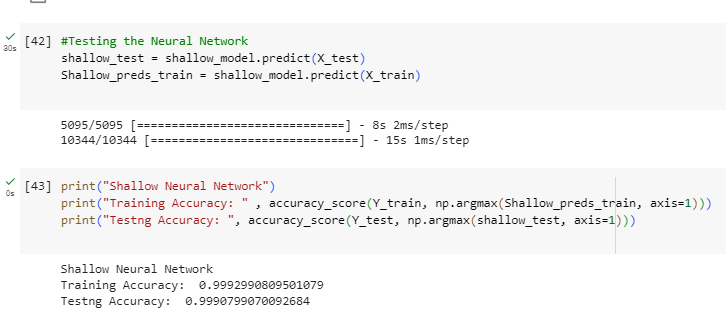
 **Figure 46**

In **Figure 46,** we run the prediction feature on the entry in the 101st entry to see if the model will predict the correct attack type. From **Figure 46** the model predicted that the attack is 0 (DoS). In **Figure 47**, we can check this by printing the 101st attack type and our model is correct it is 0

 **Figure 47**

### Test data set

The test data set is run through our model, if our test data set accuracy score is lower than the train data set accuracy then we might have a problem with overfitting **Figure 48**

 **Figure 48**

As we can see there is no issue with overfitting and our neural network is working perfectly. The test data set returned an accuracy of 99.9% which is very high.

# Conclusion

Machine Learning technology has advanced so much in the last ten years I believe it is time that company’s start to implement these systems. In the case of the 24/7 SOC, a Neural Network which could classify alerts and perform an action on these alerts could be implemented. Saying that, I would still like a human input during the standard working day, checking reports and confusion matrix reports for false positive and false negatives. As seen throughout this project a Neural Network can predict a correct attack to 99.2% accuracy.

The cost my company would save over the years on salary’s, hardware and office space, would make implementing autonomous systems a good financial decision.

Soon I believe machine learning autonomous algorithms will begin to implemented and start to take over from human especially on the security side of technology and this is why I want to get into this industry.

References

* Reference [1]: <https://www.ibm.com/reports/data-breach>
* Reference [2]: <https://www.ibm.com/topics/neural-networks>
* Reference [3]: Jugal K.Kalita, 2015, Network defense: Approaches, methods and techniques
* Reference [4]: <https://www.kali.org/>
* Reference [5]: <https://www.cloudflare.com/learning/ddos/ddos-attack-tools/low-orbit-ion-cannon-loic/>
* Reference [6]: <https://sourceforge.net/projects/loic/>
* Reference [7]: <https://www.microsoft.com/en-us/evalcenter/evaluate-windows-server-2022>
* Reference [8]: <https://try.solarwinds.com/pdp/security-event-manager>
* Reference [9]: <https://www.virtualbox.org/manual/ch06.html#network_hostonly>
* Reference [10]: <https://www.paloaltonetworks.com/cyberpedia/what-is-a-denial-of-service-attack-dos>
* Reference [11]: <https://siliconangle.com/2020/06/17/aws-mitigated-record-breaking-2-3-tbps-ddos-attack-february/>
* Reference [12]: <https://www.geeksforgeeks.org/slowloris-ddos-attack-tool-in-kali-linux/>
* Reference [13]: Getting started with TensorFlow, Giancarlo Zaccone, 2016 Packt Publishing
* Reference [14]: <https://www.tensorflow.org/guide/keras#:~:text=Keras>
* Reference [15]: <https://stackoverflow.com/questions/63675602/how-to-implement-a-neural-network-with-a-not-fully-connected-layer-as-the-final>
* Reference [16]: <https://www.kdd.org/kdd-cup/view/kdd-cup-1999/Data>
* Reference [17]: <https://www.kaggle.com/datasets/galaxyh/kdd-cup-1999-data>
* Reference [18]: <https://www.researchgate.net/figure/Number-of-samples-in-KDD-CUP-99-datasets_tbl2_282293396>
* Reference [19]: <https://www.kaggle.com/code/iamyajat/intrusion-detection-system-using-neural-networks/notebook> , Yajat Malhotra, 2023
* Reference [20]: <https://www.w3schools.com/python/pandas/pandas_intro.asp>
* Reference [21]: <https://www.w3schools.com/python/numpy/numpy_intro.asp>
* Reference [22]: <https://www.kaggle.com/code/iamyajat/intrusion-detection-system-using-neural-networks/notebook>
* Reference [23]: <https://www.geeksforgeeks.org/data-pre-processing-wit-sklearn-using-standard-and-minmax-scaler/>