Using Deep Learning for Information Security

Authors: Balamurali A R and Satnam Singh

In the next few years, deep learning will transform and expand as a decision engine across every enterprise business layer from product development to operations to finance to sales. While, internet biggies like Google, Facebook, Microsoft, and Salesforce have already embedded deep learning into their products, the cybersecurity industry is also catching up to leverage it for various use cases.

In this Part 1 of the technical white paper series, we briefly introduce Deep Learning (DL) along with a few existing InfoSec applications it enables. We then take a deep dive into the interesting problem of anonymous tor traffic detection. We present a DL-based solution to detect TOR traffic detection.

Deep learning is not a silver bullet that can solve all the InfoSec problems because it needs extensive labeled datasets and no such labeled datasets are readily available. However, there are several InfoSec use cases where the deep learning networks are making significant improvements to the existing solutions. Malware detection and network intrusion detection are two such areas where deep learning has shown significant improvements over the rule-based and classic machine learning-based solutions. Network intrusion detection systems are typically rule-based and signature-based controls that are typically deployed at the perimeter to detect known threats. Adversaries change the malware signatures and easily evade the traditional network intrusion detection systems. Quamar et al. [1], in their IEEE transaction paper showed that deep learning (DL)-based systems using self-taught learning to be promising in detecting unknown network intrusions. Traditional security use cases such as malware detection and spyware detection have been tackled with deep neural net-based systems [2].

The generalization power of DL-based techniques is better compared to traditional ML-based approaches. Jung et al.'s [3] DL based system can even detect zero-day malware. Daniel Gibert [2], a Ph.D. graduate from University of Barcelona has done extensive work related to convolutional neural network (CNN, a type of DL architecture) and malware detection. In his Ph.D. thesis, he says that CNN can detect even polymorphic malware. The DL-based neural nets are now getting used in User and Entity Behaviour Analytics (UEBA). Traditionally, UEBA employs anomaly detection and machine learning algorithms which distill the security events to profile and baseline every user and network element in the enterprise IT environment. Any significant deviations from the baselines were triggered as anomalies that further raised alerts to be investigated by the security analysts. UEBA enhanced the detection of insider threats, albeit to a limited extent. Now, deep learning-based systems are used to detect many other types of anomalies. Paweł Kobojek from Warsaw university, Poland [4] uses keystroke dynamics to verify the user using an LSTM network. Jason Trost, director of security data engineering at Capital One, has published several blogs [5] that has a list of technical papers and talks on applying deep learning in InfoSec.

A Brief Overview of Feed Forward Neural Network

The artificial neural network is inspired from the biological neural network. Neurons are the atomic unit of a biological neural network. Each neuron consists of dendrites, nucleus, and axons. It receives signals through dendrites and is carried out through axons (Figure 1). The computations are performed in the nucleus. The entire network is made up of chain of neurons. AI researchers borrowed this idea to develop the artificial neural network. In this

setting, each neuron accomplishes three actions: 1) it accumulates input from various other neurons or inputs in a weighted manner, 2) it sums up all input signals 3) based on the summed value it calls an activation function. Each neuron thus can classify whether a set of inputs belong to one class or another. This power is limited when only a single neuron is used. However, combining a set of neurons make it a powerful machinery for classification and sequence labelling task.

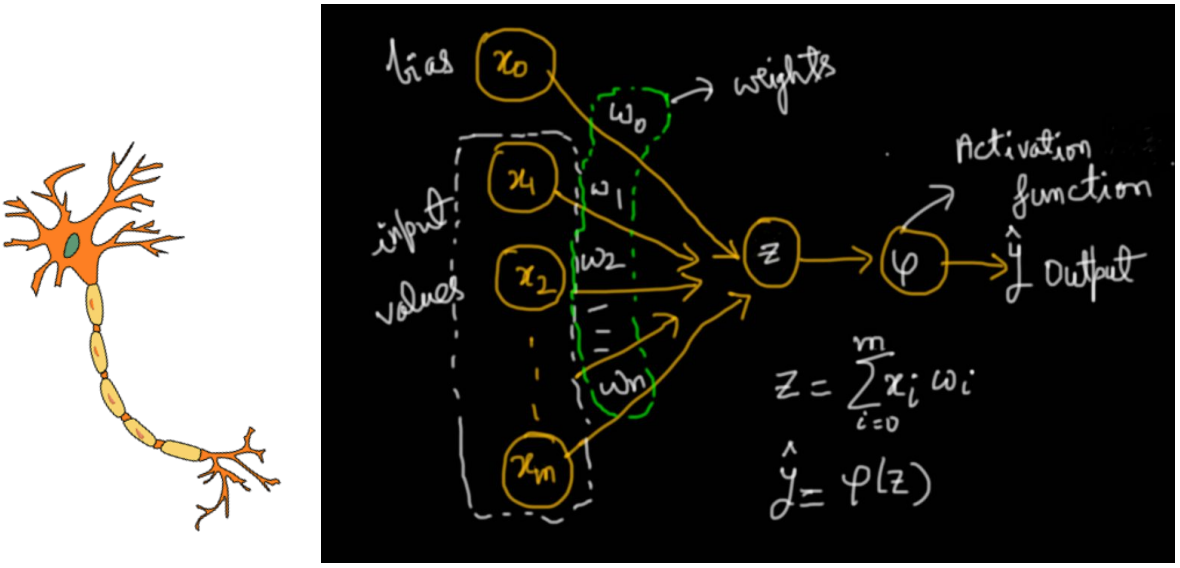

***Figure 1**: Greatest inspiration that we can get is from the nature- figure depicts a biological neuron and artificial neuron.*

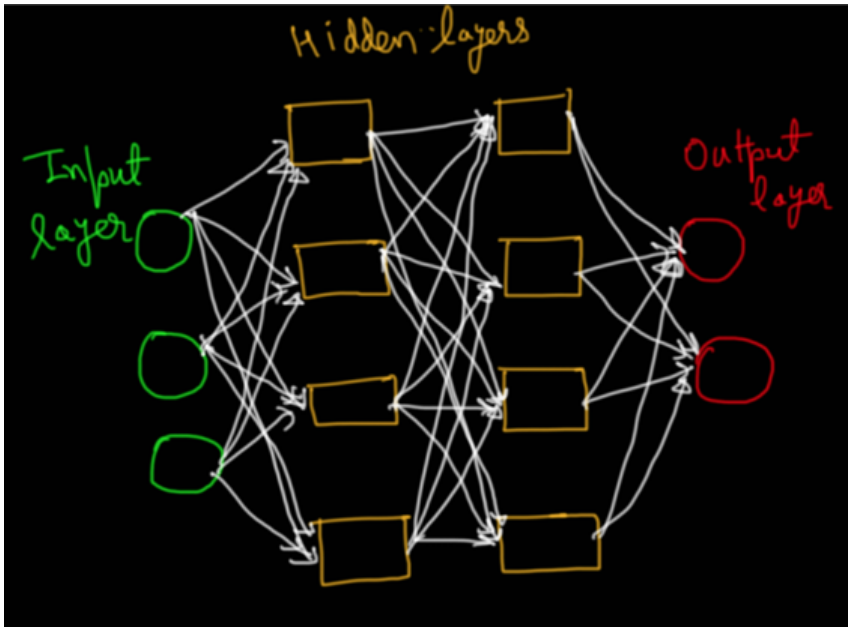

***Figure 2**: A feed forward network with two hidden layers*

A set of neuron layers can be used to create a neural network. The network architecture differs based on the objective it needs to achieve. A common network architecture is an Feed Forward Neural Network (FFN). Neurons are arranged linearly without any cycles to

form a FFN. It is called a feed forward because information travels forward direction inside the network, first through the input neurons layer, then through the hidden neurons layers, and the output neurons layer (Figure 2). Like any supervised machine learning model, the FFN needs to be trained using the labeled data. The training is in the form of optimizing the parameters by reducing the error between the output value and the true value. One such important parameter to optimize is the weight each neuron gives to each of its input signals. For a single neuron, the weight can be easily computed using the error. However, when a set of neurons are collated in multiple layers, it is challenging to optimize the neuron weights in multiple layers based on the error computed at the output layer. The backpropagation algorithm helps to address this issue [6]. Backpropagation is an old technique which comes under branch of computer algebra, automatic differentiation which is used calculate the gradient that is needed in the calculation of the weights to be used in the network. In an FFN, based on activation of each linked neuron, the output is obtained. The error is propagated layer by layer. Based on the correctness of the output with the final outcome, the error is calculated. This error is then in turn back propagated to fix errors of internal neurons. For each of the data instance, the parameters are optimized by going through multiple iterations.

Case Study: Tor Traffic Detection using Deep Learning

The primary goal of cyber-attacks is to steal the enterprise customer data, sales data, intellectual property documents, source codes and software keys. The adversaries exfiltrate the stolen data to the remote servers in encrypted traffic along with the regular traffic. Most often adversaries use an anonymous network that makes it difficult for the security defenders to trace the traffic. Moreover, the exfiltrated data is typically encrypted, rendering a rule-based network intrusion tools and firewalls to be ineffective to detect the exfiltration. Recently, anonymous networks have also been used for C&C by specific variants of ransomware/malware. For instance, Onion Ransomware [7] uses TOR network to communicate with its C&C.

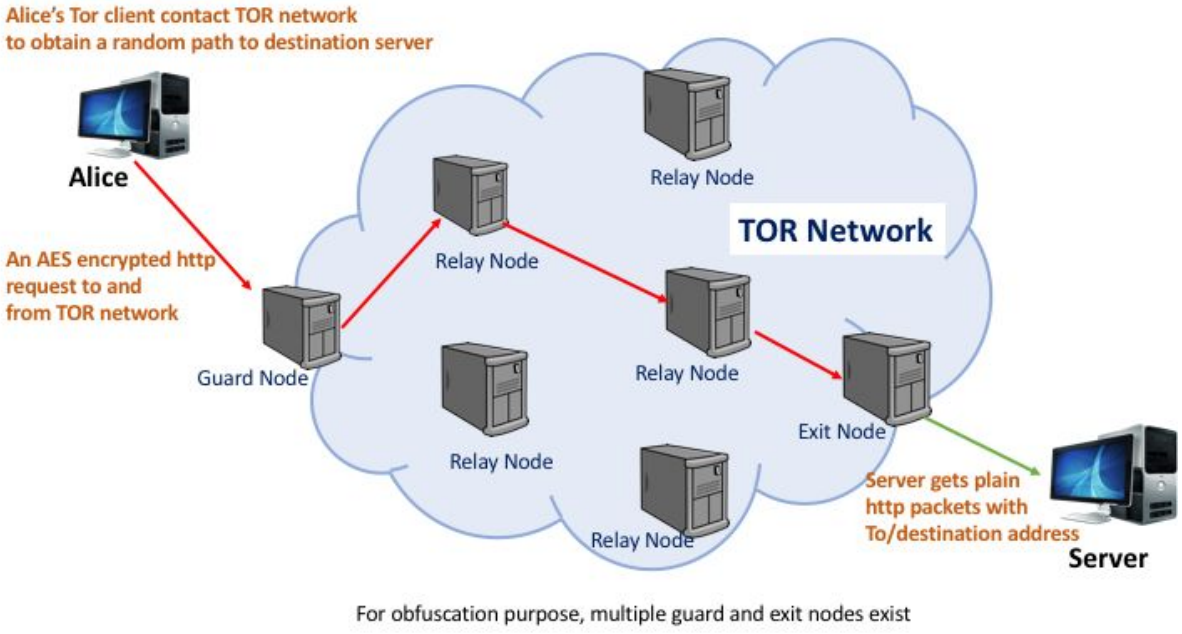

***Figure 3**: An illustration of TOR communication between Alice and destination server. The communication starts with Alice requesting a path to the server. TOR network gives path which is AES encrypted. The randomization of path happens inside the TOR*

network. The encrypted path of the packet is shown the red. Upon reaching the exit node, which is the periphery node of the TOR network, the plain packet is transferred to the server.

Anonymous network/traffic can be accomplished through various means [8]. They can be broadly classified into two: network based (TOR, I2P,Freenet) or custom os based(subgraph OS, Freepto). Among them, TOR is one of the popular choices. TOR is a free software that enables anonymous communication over the internet through a specialized routing protocol known as the onion routing protocol [9]. The protocol depends on redirecting internet traffic over various freely hosted relays across the world. During the relay, like the layers of onion peel, each HTTP packet is encrypted using the public key of the receiver. At each receiver point, the packet can be decrypted using the private key. Upon decryption, the next destination relay address is revealed. This carries on until the exit node of TOR network is met, where the decryption of the packet ends, and plain HTTP packet is forwarded to the original destination server. An example routing scheme between Alice and the server is depicted in Figure 3 for illustration.

The original intent of launching TOR was to safeguard the privacy of the users. However, adversaries have hijacked the good Samaritan objective to use it for various nefarious means instead. As of the 2016 report, around 20% of the tor traffic accounts for illegal activities [9]. In an enterprise network, TOR traffic is curtained by not allowing the installation of TOR client or blocking the Guard or Entry node IP address. However, there are numerous means through which adversaries and malware can get access to TOR network to transfer data or information. The IP blocking strategy is not a sound strategy. Adversaries can spawn different IPs to carry out the communication. A bad bot landscape report by distil networks [5] shows that 70% of automated attacks in 2015 used multiple IPs, and 20% of automated attacks used over 100 IPs.

Another way to detect TOR traffic is through traffic analysis. This requires big data technologies to boil the ocean. However, using Acalvio's patented Shadowplex deception solution, TOR traffic can be detected without any of these challenges. To enable this detection, we leverage Deep Learning-based classification models.

TOR traffic can be detected by analyzing the traffic packets. This analysis can be on the TOR node, or in between the client and the entry node. The analysis is done on a single flow of packet. Each flow constitutes a tuple of source address, source port, destination address, and destination port. Network flows for different time intervals are extracted and analysis is carried on them. G. He et al. [10] in their paper "Inferring Application Type Information from Tor Encrypted Traffic" extracted burst volumes and directions to create HMM model to detect the TOR applications that might be generating that traffic. Most of the popular works in this area leverages time-based features along with other features like size and port information [11,13,14] to detect TOR traffic. We take inspiration from Habibi et al's " Characterization of Tor Traffic using Time based Features" [11] paper and follow a time-based approach over extracted network flow to detect TOR traffic for this paper. However, our architecture uses a plethora of other meta-information that can be obtained to classify the traffic. This is inherently due to Deep Learning architecture that has been chosen to solve this problem. More about this will follow later in the article.

Data Experiments - Tor Traffic Detection:
We obtained the data from Habibi Lashkari et al. [11] at the University of

New Brunswick for the data experiments in this article. Their data consists of features extracted from traffic analysis of the university internet traffic. Extracted meta information from the PCAP data is give in the table below:

Meta-Information parameter	Parameter Explanation
FIAT	Forward Inter Arrival Time, the time between two packets sent forward direction (mean, min, max, std).
BIAT	Backward Inter Arrival Time, the time be- tween two packets sent backwards (mean, min, max, std).
FLOWIAT	Flow Inter Arrival Time, the time between two packets sent in either direction (mean, min, max, std).
ACTIVE	The amount of time time a flow was active before going idle (mean, min, max, std).
IDLE	The amount of time time a flow was idle before becoming active (mean, min, max, std).
FB PSEC	Flow Bytes per second. Flow packets per second. duration: The duration of the flow.

Table 1: Meta information parameters obtained from [1]

Apart from these parameters, other flow-based parameters are also included. A sample instance from the dataset is shown in Figure 4.

```
Source IP, Source Port, Destination IP, Destination Port, Protocol, Flow Duration, Flow Bytes/s, Flow Packets/s, Flow IAT Mean, Flow IAT Std, Flow IAT Max, Flow IAT Min,Fwd IAT Mean, Fwd IAT Std, Fwd IAT Max, Fwd IAT Min,Bwd IAT Mean, Bwd IAT Std, Bwd IAT Max, Bwd IAT Min,Active Mean, Active Std, Active Max, Active Min,Idle Mean, Idle Std, Idle Max, Idle Min,label
10.0.2.15,53913,216.58.208.46,80,6,435,0,4597.7011494253,435,0,435,435,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,nonTOR
```

***Figure 4**: An instance of the dataset used for this blog.*

Please note that source IP/port and destination IP/port, along with the protocol field has been removed from the instance as it overfits the model. We process all other features using a deep feed forward network with N hidden layers. The architecture of the neural network is shown in Figure 5.

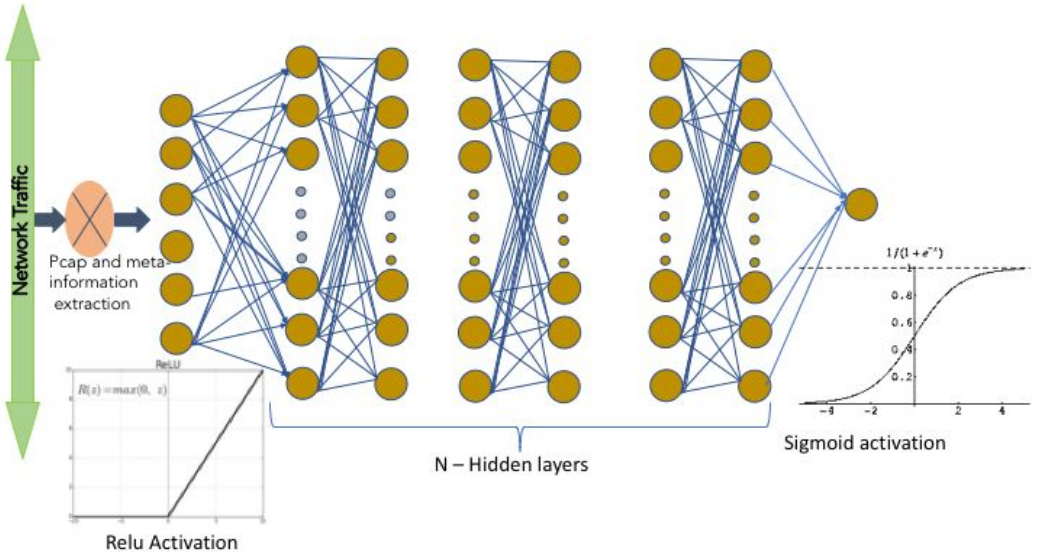

Figure 5: *Deep learning network representation used for TOR traffic detection.*

The hidden layers are varied between 2 to 10. We found N=5 to be optimal. For activation, Relu is used for all the hidden layers. Each layer of Hidden layer is dense in nature of dimension 100.

```
model = Sequential()
model.add(Dense(feature_dim, input_dim= feature_dim, kernel_initializer='normal',
activation='relu'))
for _ in range(0, hidden_layers-1):
    model.add(Dense(neurons_num, kernel_initializer='normal', activation='relu'))
model.add(Dense(1,kernel_initializer='normal', activation='sigmoid'))
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=["accuracy"])
```

Figure 6: *A Python Code Snippet of the FFN in Keras.*

The output node is activated by a sigmoid function. This was used so as the output is binary classification - Tor or Non-Tor.

We use Keras with Tensorflow in the backend to train the DL module. Binary cross entropy loss was used for optimizing the FFN. The model was trained for different epochs. Figure 7 below shows training simulation for a run depicting the increasing performance and decreasing loss value as the number of epochs increase.

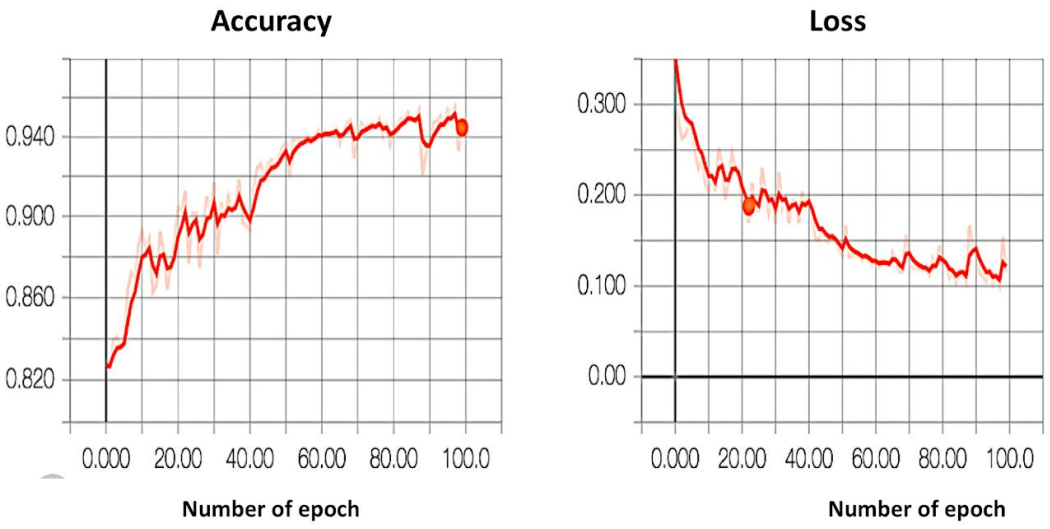

Figure 7: Tensorboard generated statics depicting the network training process

The results of the deep learning system are compared with various other estimators. Standard classification metrics of Recall, Precision and F-Score is used to measure the efficacy of the estimators. DL-based system is able to detect TOR class well. However, it is the Non-Tor class that we need to give more importance. It is seen that Deep Learning-based system can reduce the false positives cases for Non-Tor category samples. The results are shown in the table below:

Classifier used	Precision	Recall	F-Score
Logistic Regression	0.87	0.87	0.87
SVM	0.9	0.9	0.9
Naïve Bayes	0.91	0.6	0.7
Random Forest	0.96	0.96	0.96
Deep Learning	0.95	0.95	0.95

Table 2: The output of ML and DL Models for the Tor Traffic Detection

Among various classifiers, Random Forest and Deep learning based approach perform better than the rest. The result shown is based on 55K training instances. The dataset used in this data experiment is comparatively a smaller dataset for the DL-based systems. As the training data increases, performance would increase further for both DL-based and Random forest classifier. However, for large datasets, DL-based classifier typically outperform other classifiers, and they can be generalised for similar types of applications. For example, if one needs to train a classifier to detect the application used by TOR, then only the output layer

needs retraining, and other layers can be kept same whereas other ML-classifiers need to be retrained for the entire dataset. Retraining the model may take significant computing resources for large datasets.

Conclusion:

Anonymized traffic detection is a nuance that every enterprise face. The adversaries use TOR channels to exfiltrate data in anonymous mode. Current approaches by tor traffic detection vendors depend on blocking known entry nodes of the TOR network. This is not a scalable approach and can be easily bypassed. A generic method is to use deep learning-based techniques. In this paper, we presented a deep learning-based system to detect the TOR traffic with high recall and precision.

Acalvio's Shadowplex deception solution can detect the lateral movement, privilege escalation and data exfiltration by the adversaries that have already crossed the perimeter and hiding within the enterprise network. Shadowplex has the capabilities to engage with the threats using different types of high interaction deceptions, e.g., hosts, databases, and shares. When the adversary tries to exfiltrate the content from the high interaction deceptions, Shadowplex detects them using a combination of the host intrusion detection systems and deep learning-based techniques. In the next paper, we will share more such capabilities of deep learning-based models in detecting hidden threats to improve the security defense.

Reference:

[1]: Quamar Niyaz, Weiqing Sun, Ahmad Y Javaid, and Mansoor Alam, "A Deep Learning Approach for Network Intrusion Detection System," IEEE Transactions on Emerging Topics in Computational Intelligence, 2018.

[2]: Daniel Gibert, "Convolutional Neural Networks for Malware Classification," Thesis 2016.

[3]: Wookhyun Jung, Sangwon Kim,, Sangyong Choi, "Deep Learning for Zero-day Flash Malware Detection," IEEE security, 2017.

[4]: Paweł Kobojek and Khalid Saeed, "Application of Recurrent Neural Networks for User Verification based on Keystroke Dynamics," Journal of telecommunications and information technology, 2016.

[5]:Deep Learning Security Papers, http://www.covert.io/the-definitive-security-datascience-and-machinelearning-guide/#deep-learning-and-security-papers, accessed on May 2018.

[6]: "Deep Learning," Ian Goodfellow, Yoshua Bengio, Aaaron Courville; pp 196, MIT Press, 2016.

[7]: "The Onion Ransomware," https://www.kaspersky.co.in/resource-center/threats/onion-ransomware-virus-threat, Retrieved on November 29, 2017.

[8]: "5 best alternative to TOR.," https://fossbytes.com/best-alternatives-to-tor-browser-to-browse-anonymously/, Retrieved on November 29,2017.

[9]: Tor. Wikipedia., https://en.wikipedia.org/wiki/Tor_(anonymity_network), Retrieved on November 24, 2017.

[10]: He, G., Yang, M., Luo, J. and Gu, X., " Inferring Application Type Information from Tor Encrypted Traffic," Advanced Cloud and Big Data (CBD), 2014 Second International Conference on (pp. 220-227), Nov. 2014.

[11]: Habibi Lashkari A., Draper Gil G., Mamun M. and Ghorbani A., "Characterization of Tor Traffic using Time based Features," Proceedings of the 3rd International Conference on Information Systems Security and Privacy - Volume 1,pages 253-262, 2017.

[13]: Juarez, M., Afroz, S., Acar, G., Diaz, C. and Greenstadt, R., “A critical evaluation of website fingerprinting attacks,” Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security (pp. 263-274), November 2014
[14]: Bai, X., Zhang, Y. and Niu, X., “Traffic identification of tor and web-mix,” Intelligent Systems Design and Applications, ISDA'08. Eighth International Conference on (Vol. 1, pp. 548-551). IEEE, November 2008