Cloud-based machine learning for the detection of anonymous web proxies

Cloud-based machine learning for the detection of anonymous web proxies

Shane Miller
Faculty of Computing and Engineering
Ulster University
Northern Ireland
miller-s5@email.ulster.ac.uk

Kevin Curran
Faculty of Computing and Engineering
Ulster University
Northern Ireland
Kj.curran@ulster.ac.uk

Tom Lunney
Faculty of Computing and Engineering
Ulster University
Northern Ireland
Tf.lunney@ulster.ac.uk

***Abstract*— The emergence and growth of cloud computing has made a serious impact on the IT industry in recent years with large companies starting to offer powerful, reliable and cost-efficient platforms for businesses to build and reshape their business models. Showing no sign of slowing down, cloud computing capabilities now include machine learning, with facilities for both designing and deploying models. With this capability of machine learning using cloud computing comes the increasing need to be able to classify whether an incoming connection is from a legitimate originating IP address or if it is being sent through an intermediary like a web proxy. Taking inspiration from Intrusion Detection Systems that make use of machine learning capabilities to improve anomaly detection accuracy, this paper proposes that cloud based machine learning can be used in order to detect and classify web proxy usage by capturing packet data and feeding it into a cloud based machine learning web service.**

Keywords—Cloud Computing, Machine Learning, Anonymous proxy, network traffic, Traffic analysis, SSL/TLS encryption

I. INTRODUCTION

There has been an increase over the years in the use of various anonymous communication systems on the internet [1]. This has been in part due to a rise in censorship of the internet in some countries where certain online services like social media sites may be blocked. In situations like these, anonymous communication systems can allow for the access of an online service whilst protecting the identity of the person accessing it. Anonymous communication systems can be categorised into one of two groups: message based/high-latency applications or flow based/low-latency applications [2]. High latency applications can include email and e-voting systems. Low latency anonymous communication systems have been researched extensively recently due to the increasing need for anonymity alongside high speed communication applications such as web browsers and instant messaging platforms. This has led to the development of various kinds of low latency systems including the popular system Tor as well as various kinds of HTTP and SOCKS proxy services. Systems such as Tor fall under the category of multihop anonymous communications models while HTTP and SOCKS proxies fall under the category of single-hop anonymous communication models. HTTP proxy servers are servers that act as an intermediary for resource requests located on servers on the Internet.

One type of HTTP proxy server is a reverse proxy server. Reverse proxies are typically used as an Internet facing server that handles a number of different tasks such as load balancing to distribute requests between several web servers and acting as a cache for static content such as pictures and other graphical content. Anonymising proxies are based on another type of HTTP proxy known as an open proxy. Open proxies are a proxy that is available to any user on the Internet. They are mostly used to set up anonymous proxy websites and are categorised as a single-hop anonymous communication model. Anonymising proxy websites provide anonymity by concealing the users IP address from web servers on the Internet. This type of server is regularly used as a means to access blocked content. There are a number of risks with using an anonymous proxy as a method to bypass network filters on a company network. The anonymous proxy server might not be a simple intermediary that only forwards requests and fetches the results. It could be logging all the requests and information that pass through it in the hopes of finding usernames, passwords or financial information. The operators of the proxy site may use these to steal the identity linked to the credentials to gain further access into the company network or to defraud the company.

II. BACKGROUND

A. Integrating Machine Learning into IDSs

Intrusion detection is the process of monitoring connections coming to and leaving from a computer or network and then analysing those connections for signs of potential violations or incidents that go against security and acceptable use policies [3]. Causes of these incidents can include attackers gaining unauthorised access to systems, malware such as spyware and Trojan viruses and misuse of system privileges by users or attempts to gain additional privileges. An intrusion detection system is the software that automates this process. When detecting possible incidents, an IDS can take a number of actions. One would be to report the incident to a system security administrator, who could then initiate a response to mitigate the effects of the incident. Alongside alerting an administrator, the IDS could also keep a record of incident that could be referenced at a later date and as a way to help prevent future cases of that particular incident. There are a number of

different types of IDS. These are: Network based, Host based, Network Behaviour and Wireless [3]. Network based systems monitor the traffic of a network using sensors placed at certain parts of the network and IDS management servers. They analyse the activity recorded by the sensors in order to identify incidents of intrusion. Fig.1 shows the typical layout of a network that includes a network based IDS. Host based systems differ from network based systems by monitoring a single host. NBA systems monitor network traffic in order to identity threats that generate unusual traffic flows such as malware or port scanning attempts. Wireless IDSs apply similar techniques to network based systems specifically to wireless network traffic that makes use of wireless networking protocols. The proposed research aims to create a type of network based intrusion detection system. Integrating machine learning techniques into the IDS will be used as a method to increase the ability and accuracy of the detection system. Machine learning techniques include various kinds of artificial neural networks and classification techniques as well as genetic algorithms and fuzzy logic. There has been various research studies looking into integrating machine learning into IDSs with the recent trend being in improving the machine learning aspect by combining different techniques to increase detection accuracy and to decrease the computational effort required to train the systems. [4] proposed a feature representation technique using a combination of the cluster centre and nearest neighbour approaches. Experiments that were carried out made use of the KDD-Cup99 dataset and showed that the approach required less computational effort to provide similar levels of accuracy to k-NN. [5] proposed a multiple level hybrid classifier that combined supervised tree classifiers with unsupervised Bayesian clustering. Performance of this approach was also measured using the KDD-Cup99 dataset and experiments showed that it provided a low false negative rate of 3.23% and a false positive rate of 3.2% with a high detection rate for both known and unknown attacks. [6] made use of a Support Vector Machine (SVM) for classification and a clustering tree technique called Dynamically Growing Self-Organising Tree (DGSOT) to improve the training times of the SVM. Experiments were carried out using the DARPA98 dataset and showed that using a clustering tree helped to

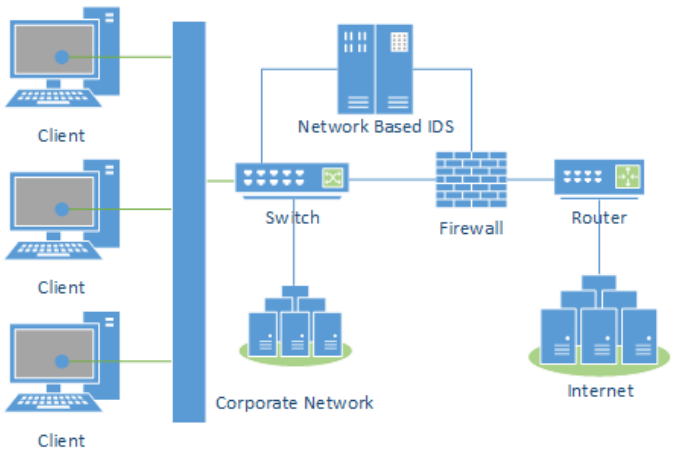

Fig. 1. Network Based Intrusion Detection System.

increase the accuracy rate of the SVM and lower the rates of false positives and false negatives. [7] provided a system that made use of both genetic algorithms and fuzzy logic to create a genetic fuzzy classifier to predict different behaviours in networked computers. Their results showed that there was a benefit to using fuzzy logic to pre-screen rules before classifying with the genetic algorithm as it decreased the time needed to train the system. However the systems accuracy in detection did not show much increase and actually showed a decrease in accuracy in some classes compared to other approaches. An earlier study used 3 different anomaly detection techniques for classifying program behaviour [8]. These techniques were an equality matching algorithm for determining what was and wasn't anomalous behaviour, a feed forward backpropagation neural network for learning the program behaviour and the third being a recurrent neural network called an Elman network for recognising recurrent features of program behaviour. Their study showed that the performance of intrusion detection benefited greatly from the use of the backpropagation network and the Elman network. The general consensus that can be gathered from these studies is that the use of machine learning techniques does improve the accuracy and performance of intrusion detection systems.

B. Cloud Computing

Cloud computing is defined by the National Institute of Standards and Technology (NIST) as a model for enabling ubiquitous, convenient, on-demand access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction [9]. The idea of cloud computing is not a recent one. The concept of computing facilities being provided to the public as a utility was already being explored as early as the 1960s however the actual term "cloud" only started to gain popularity after it was used to describe the business model of providing services across the internet by Eric Schmidt in 2006 [10]. Today the industry is growing rapidly with costs decreasing rapidly and is largely dominated by companies based in the USA, such as Amazon, Microsoft, Google and IBM [11]. There are a number of different service models available which describe the capability of specific cloud services. These are Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) [9]. SaaS describes the capability provided to the consumer to use the provider's applications running on the provider's infrastructure. PaaS is the capability provided to the consumer to deploy consumer-created or acquired applications created using programming languages, libraries, services and tools supported by the provider. The consumer does not manage or control the underlying infrastructure that is left to the provider. The consumer however has control of the deployed applications and possibly settings for the hosting environment. IaaS describes the capability provided to the consumer to provision processing, storage and other fundamental computing resources where the consumer is able to deploy and run arbitrary software including operating systems and applications. The consumer does not manage the underlying infrastructure but has control over the operating systems that will be used, how much and what type of storage to use and what applications will be deployed with the possibility of limited control of select

networking components. There are a number of possible deployment methods of a cloud computing service. These are: Community cloud, Private cloud, Public cloud and Hybrid cloud. A community cloud deployment is where the infrastructure is provisioned for use by a specific community of consumers from multiple organisations that have a shared goal. This can be managed and operated by one of more of the organisations in the community or a third party. A private cloud deployment is where the cloud infrastructure is provisioned for exclusive use by a single organisation of multiple consumers. It can be managed and operated by the organisation, a third party or a combination of the two. A public cloud deployment is for open use by the general public. This instance can be owned, managed and operated by a business, academic or government organisation. Finally, a hybrid cloud deployment is a composition of two or more of the above cloud infrastructures. With the rise in popularity of machine learning in research, cloud computing providers have, in recent years started to incorporate machine learning into the services that they are offering. All of the companies mentioned above offer this, with amazon offering it as part of their AWS platform , google offering a Prediction API , IBM offering its Watson Analytics service and Microsoft offering its Azure Machine Learning studio . For this study, Microsoft's Azure will be used.

C. Azure Machine Learning

Azure Machine Learning studio is a cloud service that provides an IDE-like workspace to allow for easier building, testing and deployment of predictive analytic models. Models can be constructed by dragging and dropping dataset and analysis modules into a workspace area. Modules can be added iteratively to help pinpoint problems. Predictive analysis helps you predict what will happen in the future. It is used to predict the probability of an uncertain outcome. Azure offers various types of statistical and machine learning algorithms aimed at predictive analysis such as neural networks, boosted decision trees and linear regression. Azure outlines a 5 step process to building an experimental predictive model: gather raw data, pre-process the data to clear the data of missing values or other mistakes, define the features that the model will be trained on, choose and train a learning algorithm, test the algorithm [12]. Once the model is trained and is predicting the test data accurately it can be deployed as a web service. Azure replaces the original dataset with a module to allow input from the web. Using the C#, python or R programming languages in conjunction with the URL of the deployed web service and a generated key, data can be sent to the web service to be analysed. There have been a number of recent studies that have made use of azure's machine learning studio. [13] proposed a generalised flow within azure that would accept multi-class and binary classification datasets and process them to maximise the overall classification accuracy. Two sets of experiments were run. The first was to benchmark the azure machine learning platform using three public datasets. The second was to evaluate the proposed generalised flow using a generated multi-class dataset. The results showed that the generalised flow improved accuracy in all but one of the comparisons with prior work. [14] describes a methodology to obtain a real-time view of traffic issues in a city using status updates, tweets and comments on social media networks using state of the art machine learning. The machine learning was provided by the Azure machine learning studio. Data from various social networks is polled continuously by a worker role process hosted in azure. The machine learning studio is used to process the data and analyse the text being imported. For the experiment they annotated 1100 social network feeds for 4 cities. This data was then split into training, cross validation and testing datasets which were used to train and test the machine learning algorithms in azure machine learning studio. Classification accuracy for one social network ranged from 90-95% whereas on another it was just higher than 75%. [15] proposed a method aimed at grading short test answers using machine learning techniques implemented in azure. Experiments were run using 152 samples of student answers to a computer science question. The experiment showed that the system was able to grade all of the answers correctly after testing. [16] proposed an anti-fraud web service that employed machine learning algorithms for predictive analytics in order to reduce the costs of infrastructure and software. Azure machine learning studio was used to provide the machine learning aspect. When building the machine learning model in azure, they experimented with several algorithms for two-class classification. Using Azure's built in Score Model module, they were able to achieve an accuracy of 88% and went on to publish the model as a web service that was capable of performing anti-fraud activities whilst reducing the cost of such a service to virtually zero.

D. Web Proxies

Anonymous web proxies come in many different forms. Some proxy scripts are produced using PHP based or CGI (Common Gateway Interface) based scripts. The reasoning behind the use of these technologies is that they both provide the functionality that an anonymous proxy requires and they are compatible with both UNIX-like and Windows hosts. To access the anonymous proxy a user client needs to connect to the proxy server first. From there, they are then able to send a request to the website anonymously. The proxy script takes the clients request and issues its own request to the destination website, receives the data back and forwards it on to the client. This is shown in Fig. 2. Glype is a PHP based script and is one of the most common and popular web proxy scripts available on the internet. This is due to its support for content like JavaScript and to its ease of

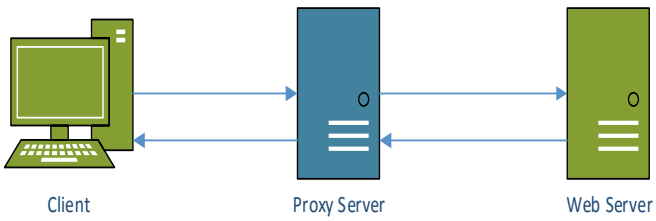

Fig. 2. Proxy Connection.

set up and use. To set up a Glype proxy server, a user must download the proxy files from the Glype website and then relocate the files to the correct directories on their webserver. Another option would be to use one of the many existing proxy sites already available. The Glype website provides a list of working proxy servers whose administrators have paid to have their site listed in the hope of increasing the popularity of their own server. At the time of writing this list contained 3,389 unique servers. This list, however only represents those that have paid to have better exposure; there are possibly many more Glype proxy servers. This presents a problem when trying to block access to these proxies because there are so many. This makes it difficult to compile a complete list to add to an IP block list or ACL. In addition, because it is so easy to set up the proxy, new servers are being added all the time. URL filtering will not work either as the majority of proxy servers based on the Glype script will have some form of URL obfuscation available. The most popular methods of obfuscation are encoding the URL using either base64 or ROT-13 encoding. Other methods of encoding exist, but these are the main ones used by the Glype script.

The CGIProxy is a Common Gateway Interface (CGI) script that acts as a HTTP, HTTPS or FTP proxy. CGI scripts can be programmed in a number of different languages. CGIProxy is programmed using the interpreted language Perl. While Glype proxies enable URL obfuscation by default, the CGIProxy script does not. ROT13 encoding can be enabled by removing the line comments for the methods proxy_encode() and proxy_decode() in the script. The script also provides support for custom encoding code to be added such as hexadecimal encoding.

III. Methodology

The proposed system will be a form of network-based intrusion detection system, developed specifically to detect the use of anonymous proxy scripts. This system will consist of a supervised machine learning backed traffic classification system that will be augmented by a method for getting around SSL encryption used by a growing number of proxy sites. Underlying these components will be the capability to capture network packets in real time in a similar way to the packet analysis software Wireshark. The method for getting around SSL encryption that may be used by proxy sites will be based on a penetration testing tool called sslstrip. It is a form of MITM attack that forces a user's browser into communicating with an adversary in plain-text over HTTP. This is possible because many HTTPS sites are normally accessed from a HTTP 302 redirect on a HTTP page. The connection is intercepted before the redirect can take place and modify it to redirect to the HTTP version of a site e.g. https://twitter.com would become http://twitter.com. The adversary then acts like a proxy and forwards the communication on to the internet as normal, using either HTTP or HTTPS depending on what is being requested whilst maintaining the HTTP connection between user and adversary. The "adversary" in the case of this project would be the proxy detection system in an attempt to gain access to encrypted network packets that are being sent to and from anonymous proxies for deeper analysis of their

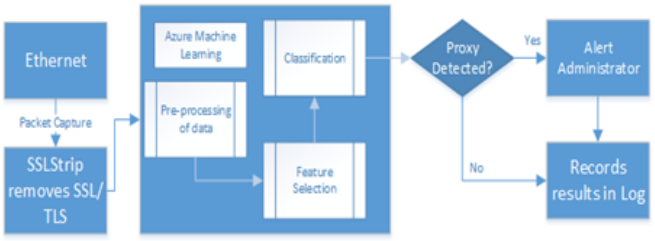

Fig. 3. System Architecture

contents. The detection of proxy traffic will be carried out by using traffic classification based on the Azure machine learning cloud web service. This service will be trained to recognise and generalise both proxy traffic and non-proxy traffic. If an anonymising proxy is detected then the system will create a log of the detection that includes the packet that was analysed and the time and date it was captured at. An alert to the network administrator will also be sent to alert them to the detection.

A. System Design

This system will monitor a network by capturing packets as they go to and from the network then sending those packets to the azure machine learning web service for pre-processing and classification. Fig. 3 shows how the machine learning aspect of the system will be utilised as part of the system. Fig. 4 shows the contextual architecture of an example network for the operating proxy detection system. The system is positioned between 2 firewalls; one controlling access to and from the internet and another controlling access to the innermost network where the client machines reside. This creates an area known as a Demilitarised Zone (DMZ) which is a subnetwork that provides an additional layer of security to a network, separating a business' local intranet from the wider Internet. This is known as the perimeter of the network. A dataset for training the network to recognize and classify anonymous proxy traffic will need to be created as none are available for this specific area. Reference in the background section are two datasets for training intrusion detection systems; DARPA98 and KDD-Cup99. A more recent dataset was defined by [17] as an improvement upon the KDD-Cup99 dataset. This dataset will be used as a template for the creation of the proposed proxy detection dataset. The feature set includes 41 different attributes with a number of them dedicated to classifying different types of attack. Feature extraction and selection will need to be performed on proxy packets to find out how many of the 41 features would be necessary for proxy detection. Features included range from basic network connection details such as the protocol type and the number of bytes transferred from the source to the destination to more specific features such as the number of failed login attempts [18]. The Azure

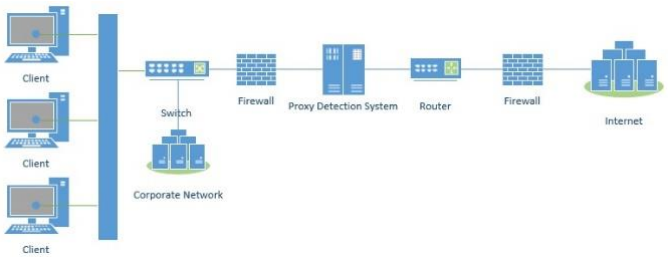

Fig. 4. Network Architecture

module Sweep Parameters can be used when preparing the data to pick the most suited features and reduce the dimensionality of the feature set to increase the classification accuracy of the trained model. For preliminary experiments a dataset of 200 samples will be used. This will be split into 70% samples for training the network and 30% for testing and validating it. This can be done using the Azure module Sample and Split. Once the data has been prepared, it can then be used to train a machine learning model based on one of several algorithms that azure provides. Using azure, it is possible to build and evaluate many different algorithms simply by choosing which algorithm you would like to try and connecting it to the rest of the system by dragging and dropping the module into the correct place. Using the Score Model module to evaluate each algorithm will then allow for the best algorithm to be chosen based on the classification problem. Fig. 5 shows an example machine learning model from the Azure machine learning studio.

B. *Software Analysis*

Potential software tools are being investigated for the development of the proxy detection system. These tools include the general programming language Python as well as the network penetration testing tool sslstrip. Python is supported on both Windows and Unix-like based systems which means that the system will not be dependent on a single operating system. It also has generous support for packet snuffing and capture through the inclusion of the Scapy or Libpcap libraries. SSL stripping is a concept that was developed by Moxie Marlinspike in 2009. It is a form of man

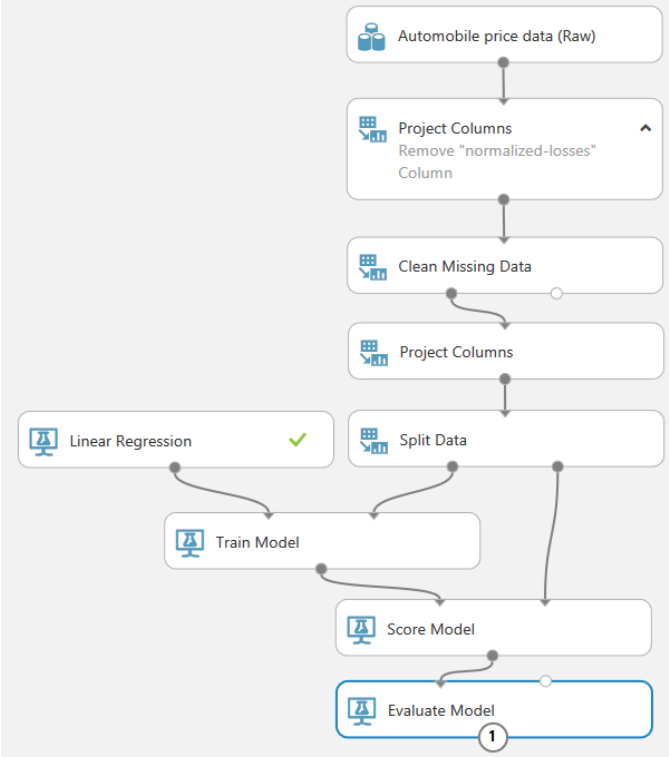

Fig. 5. Example machine learning model from Azure

in the middle attack that allows an attacker to prevent a web browser from upgrading an unencrypted HTTP connection to a HTTPS connection that is encrypted using SSL or TLS. He developed the tool sslstrip that was previously discussed above. The idea behind sslstrip is that users only encountered SSL in one of two ways, they either clicked on a hyperlink such as a login button or through a HTTP 302 redirect. What happens with the 302 redirect is a user will usually not type the "https://" prefix into the URL address bar. Instead they will type in "website.com" which the browser will automatically interpret as a request for "http://www.website.com". If the website being requested only normally runs on HTTPS then the web server of the site will reply to the HTTP request with the 302 redirect code, telling the users browser to request the HTTPS URL instead. Fig. 6 shows what this looks like in the network analysis tool Wireshark.

```
HTTP        337 GET / HTTP/1.1
HTTP        670 HTTP/1.1 302 Moved Temporarily
```

Fig. 6. HTTP GET request for a website followed immediately by a 302 redirect

The website requested normally runs on HTTPS as it contains a login form, however the URL request defaulted to HTTP. Therefore the web server sent a redirect telling the browser to instead request the HTTPS version of the website. Sslstrip monitors HTTP traffic on a network and whenever it detects "https://" in a URL request, it intercepts the communication and changes it to "http://". Whenever such a connection is detected, sslstrip will then initiate a SSL connection to the desired server and then forwards on the request as normal as if nothing had changed. This way the server never knows that the connection is being forwarded by sslstrip. Everything that is passed along through this connection can be read and logged in an unencrypted format. Incorporating this into the proxy detection system should theoretically allow for network packets being captured by the system to be in an unencrypted format and for the packets to appear normal outside of the system. This would allow the system to apply its proxy detection techniques to proxy packets that would normally be encrypted and unreadable.

IV. Conclusion

This report has summarised the challenges of detecting anonymising proxy usage and has provided a review into the current techniques and their limitations. Also discussed was the increasing trend regarding the use of the encryption technologies SSL/TLS to obfuscate the activities of an anonymising proxy user. In this report, an alternative method for detecting anonymous proxy usage is proposed. This method involves using machine learning approaches to detect the usage of proxies. Specifically the proposed method will be based on a network intrusion detection system that will make use of a multi-perceptron backpropagation neural network to classify anonymous proxy traffic. This system will be trained from a mixed dataset of HTTP proxy traffic and HTTP traffic that is

not from a proxy. Also discussed in this paper is how this dataset will need to be generated as none are available for proxy detection. For creating the dataset, guidance can be drawn from existing intrusion detection datasets which include feature sets that may be relevant to proxy traffic. Included in this research is the use of the SSLStrip man in the middle penetration testing tool to remove the encryption from any proxy traffic coming from a HTTPS proxy website. Future work will involve designing and developing the dataset and then applying the dataset to simulated tests before progressing to testing in a real network environment. These proposed experiments will demonstrate the potential of using machine learning to support the detection of anonymous proxy usage.

REFERENCES

[1] [1] Sandvine, "Global Internet Phenomena Spotlight: Encrypted Internet Traffic," 2015. [Online]. Available: https://www.sandvine.com/downloads/general/global-internet-phenomena/2015/encrypted-internet-traffic.pdf. [Accessed: 19-Nov-2015].

[2] [2] M. Yang, J. Luo, Z. Ling, X. Fu, and W. Yu, "De-anonymizing and countermeasures in anonymous communication networks," IEEE Commun. Mag., vol. 53, no. 4, pp. 60–66, Apr. 2015.

[3] [3] K. Scarfone and P. Mell, "Guide to intrusion detection and prevention systems (idps)," NIST Spec. Publ., vol. 800, no. 2007, p. 94, 2007.

[4] [4] W.-C. Lin, S.-W. Ke, and C.-F. Tsai, "CANN: An intrusion detection system based on combining cluster centers and nearest neighbors," Knowledge-Based Syst., vol. 78, pp. 13–21, Apr. 2015.

[5] [5] C. Xiang, P. C. Yong, and L. S. Meng, "Design of multiple-level hybrid classifier for intrusion detection system using Bayesian clustering and decision trees," Pattern Recognit. Lett., vol. 29, no. 7, pp. 918–924, May 2008.

[6] [6] L. Khan, M. Awad, and B. Thuraisingham, "A new intrusion detection system using support vector machines and hierarchical clustering," VLDB J., vol. 16, no. 4, pp. 507–521, Aug. 2006.

[7] [7] T. Özyer, R. Alhajj, and K. Barker, "Intrusion detection by integrating boosting genetic fuzzy classifier and data mining criteria for rule pre-screening," J. Netw. Comput. Appl., vol. 30, no. 1, pp. 99–113, Jan. 2007.

[8] [8] A. K. Ghosh, A. Schwartzbard, and M. Schatz, "Learning Program Behavior Profiles for Intrusion Detection.," in Workshop on Intrusion Detection and Network Monitoring, 1999, vol. 51462.

[9] [9] P. Mell and T. Grance, "The NIST definition of cloud computing - nistspecialpublication800-145.pdf," 2011.

[10] [10] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," J. Internet Serv. Appl., vol. 1, no. 1, pp. 7–18, Apr. 2010.

[11] [11] C. Ohmann, S. Canham, E. Danielyan, S. Robertshaw, Y. Legré, L. Clivio, and J. Demotes, "'Cloud computing' and clinical trials: report from an ECRIN workshop.," Trials, vol. 16, p. 318, Jan. 2015.

[12] [12] V. Fontama, R. Barga, and W. H. Tok, Predictive Analytics with Microsoft Azure Machine Learning. 2014.

[13] [13] M. Bihis and S. Roychowdhury, "A generalized flow for multi-class and binary classification tasks: An Azure ML approach," in 2015 IEEE International Conference on Big Data (Big Data), 2015, pp. 1728–1737.

[14] [14] A. Pathak, B. K. Patra, A. Chakraborty, and A. Agarwal, "A City Traffic Dashboard using Social Network Data," in Proceedings of the 2nd IKDD Conference on Data Sciences - CODS-IKDD '15, 2015, pp. 1–4.

[15] [15] R. Krithika and J. Narayanan, "Learning to Grade Short Answers using Machine Learning Techniques," in Proceedings of the Third International Symposium on Women in Computing and Informatics - WCI '15, 2015, pp. 262–271.

[16] [16] A. Tselykh and D. Petukhov, "Web service for detecting credit card fraud in near real-time," in Proceedings of the 8th International Conference on Security of Information and Networks - SIN '15, 2015, pp. 114–117.

[17] [17] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009, pp. 1–6.

[18] [18] L. Dhanabal and S. P. Shantharajah, "A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms," Int. J. Adv. Res. Comput. Commun. Eng., vol. 4, no. 6, 2015.

[19]