Design of Autonomous Systems for Cybersecurity Threat Detection Using Deep Learning

Strahil Sokolov
University of Telecommunications and Post
Department of Information Technologies
Sofia, Bulgaria
e-mail: strahil.sokolov@gmail.com

***Abstract*—In this paper, an approach is proposed for designing autonomous systems featuring machine learning and neural networks for cybersecurity threat detection. It is proposed that neural models are trained on monitoring data obtained from cloud environments that service enterprise applications. Cybersecurity is a hot topic and a broad field of science that spreads over activities, such as protecting infrastructure, computers and servers, industrial and telecommunications equipment, applications and data. All modern networks are capable of substantial throughput due to enormous volumes of generated traffic. A design is proposed for autonomous threat detection systems, which is based on combining traditional and deep neural networks for cloud monitoring data analysis and an algorithm for combining classifier results. The proposed autonomous system design delivers promising results that are comparable to existing approaches and can become useful in enterprise cloud applications.**

Keywords-cybersecurity; autonomous threat detection; deep learning.

I. INTRODUCTION

Research in the field of cybersecurity has been ongoing for decades. With the continual increase of data volumes, protecting computer and telecommunication systems has become a primary concern. There are several approaches which are currently in use: traffic analysis, content analysis, application and user behavior analysis.

There exist a number of layers with common groups of threats, existing protection capabilities and Information and Communication Technology (ICT) resources that are under constant attack nowadays. The most popular applications based on traffic analysis [1] can be grouped into the fields of: network intrusion detection, botnet detection and malware detection. Over the recent years, approaches emerged based on machine learning algorithms for each of these fields. Some of the Intrusion Detection and Protection Systems (IDPS) are trained to recognize abnormalities in traffic, e.g., in peer-to-peer applications. There are Intrusion Detection Systems (IDS) for protecting against Distributed Denial-Of-Service (DDoS) attacks. There are e-mail protection services, which are able to detect harmful applications that steal information; mobile malware applications are also widespread [2]. Malware application behaviours are analyzed and detectors are trained to classify an application or part of it as harmful [2]. Another type of threat is the botnet: many compromised devices or hosts, infected with malware and connected to the Internet, that are controlled and manipulated by botmasters [3]. Botnets are mainly used for sending spam emails, DDoS attacks, identity thefts or just making use of the victim's computational resources for purposes of, e.g., tunnelling, proxying or even cryptocurrency mining.

There are several modern proposals that have appeared on the usage of advanced techniques for intrusion detection [4]. The authors propose a cybersecurity framework based on two-stage Markov model for early prediction of malicious edge devices as well as legitimate edge devices in fog computing.

In [5], focus has been given to the recent rise of security incidents affecting critical infrastructure, such as power grids and water suppliers. The German cybersecurity office - Bundesamt für Sicherheit in der Informationstechnik (BSI) - reported that not all of the incidents were due to hacking. Another recent publication [6] shows flaws and vulnerabilities in an entire European country. The author shows how vulnerability scanning can be organized by a single person and justifies the importance of cybersecurity threat detection software.

This paper is organized as follows: in Section 2, an overview is given on existing techniques for cyberthreat detection based on network traffic analysis. In Section 3, the proposed approach for design of autonomous threat detection techniques is described. Section 4 describes the technique for combining classifier results. The paper’s conclusion is in Section 5.

II. THREAT DETECTION TECHNIQUES BASED ON NETWORK TRAFFIC ANALYSIS

A. Intrusion Detection Systems (IDS) and Intrusion Prevention (IPS) Systems

Both IDS and IPS are entitled to try and recognize malicious traffic from normal traffic. There are Host-Based Intrusion Detection Systems (HIDS) and Network Intrusion Detection systems (NIDS) [7]. To achieve this goal, both IDS and IPS rely on network traffic analysis. Most of the existing systems rely on rule-based classification to detect the nature of the attacks; the malicious traffic is often concealed within botnet, DDoS attack traffic or spam traffic. It can be expected that the accuracy of such systems is relatively low [8], due to the limits of their operation modes: signature based and anomaly-based [7]. Signature based threat detection uses a set of predetermined rules that are available from the community or vendors. These rules contain signature patterns of threats similar to antivirus software. The anomaly based detection function is to detect abnormalities in the current network traffic or states of services in logs.

The big manufacturers of network equipment offer bundles of IDS/IPS systems, which claim high accuracy and machine learning capabilities. It is worth exploring some of the open source available systems.

- OSSEC [9] stands for Open Source Security. It is an open source host intrusion detection system owned by Trend Micro, one of the leading names in IT security.
- SNORT [10] is an open source intrusion prevention system capable of real-time traffic analysis and packet logging.
- Suricata [11] is a free and open source, mature, fast and robust network threat detection engine. The Suricata engine is capable of Real Time Intrusion Detection (RTID), Inline Intrusion Prevention (IIP), Network Security Monitoring (NSM) and offline pcap processing.
- Zeek [12] (ex. Bro) is a powerful network analysis framework that consists of event engine and policy scripts.
- The Samhain [13] Host-based Intrusion Detection System (HIDS) provides file integrity checking and log file monitoring/analysis, as well as rootkit detection, port monitoring, detection of rogue Set User ID (SUID) executables, and hidden processes.
- Fail2ban [14] scans log files (e.g. /var/log/apache/error_log) and bans IPs that show the malicious signs – too many password failures, seeking for exploits, etc.
- Security Onion [15] is a free and open source Linux distribution for intrusion detection, enterprise security monitoring, and log management. It includes Elasticsearch, Logstash, Kibana, Snort, Suricata, Bro, Wazuh, Sguil, Squert, CyberChef, NetworkMiner, and many other security tools.

B. *Malware analysis*

Malware detection has been a field of interest for computer virologists for a long time. In order to address the automated classification of malware based on behavioral analysis, the researchers usually need a virtual machine where they can start and analyze the malware behaviour in all of it aspects , such as function calls [2].

According [8], there is a growing number of malware threats worldwide and also the level of technological sophistication of malicious software is increasing mainly due to the popularity of smartphones. This is what makes malware analysis an important task in cybersecurity. Malware detection systems which detect malicious traffic are usually able to classify threads in the following categories: unclassified (0-day), misc-attack, Trojan-activity, not-suspicious, and misc-activity.

Among the most wide-spread malwares on the Internet as of November 2018 according to [16], the following are listed: *Coinhive; Cryptoloot; Andromeda; Roughted; Dorkbot; Jsecoin; Emotet; Conficker; XMRig* and *Nivdort.*

Among the mobile devices, [16] reports the following threats: *Triada; Hiddad* and *Lokibot.* The three most exploited Common Vulnerability Exposures (CVE) are reported as:

- Microsoft IIS WebDAV ScStoragePathFromUrl Buffer Overflow (CVE-2017-7269) - 48% of organizations have dealt with this threat;
- OpenSSL TLS DTLS Heartbeat Information Disclosure (CVE-2014-0160; CVE-2014-0346) – An attacker can leverage this vulnerability to disclose memory contents of a connected client or serve that had global impact of 44%.
- OpenSSL tls_get_message_body Function init_msg Structure Use After Free (CVE-2016-6309) – A remote, unauthenticated attacker could exploit this vulnerability by sending a crafted message to the vulnerable server. Successful exploitation allows the attacker to execute arbitrary code on the system impacting 42% of organizations.

C. *Botnet detection*

Compromised devices in botnets provide attackers with means to send spams, launch DDoS attacks, run brute-force password cracking, steal private information, and hide the origin of cyber attacks [3][17]. Malware network traffic can spread rapidly through various platforms and this is what makes botnet detection an important part in cybersecurity. According to the structure of botnets, two categories exist: Peer-to-Peer (P2P) and centralized botnet [8]. In a P2P botnet, the botmaster can control each bot with distributed commands sent from peers; whereas in a centralized botnet, the centralized Command & Control (C&C) architecture is formed with protocols like Internet Relay-Chat (IRC) and HTTP.

Network traffic analysis serves for detection of the botnets. The typical approach to detect compromised hosts on the network and filter botnet traffic is to maintain a blacklist of openly available C&C domains. The efficiency is poor because the blacklist has to be updated manually. There are botmasters who often use unchanged P2P-based C&C structures with pseudo random domain generation algorithms to evade the detection by blacklisting and to increase the reliability of the botnet. That is, the bots search for working C&C servers by periodically generating a set of pseudo-random domain names and resolving the generated domain names to IP addresses through DNS queries [18]. Therefore, these botnets can still survive even after some C&C servers are detected and blocked.

Machine Learning (ML) techniques are vital for the statistical based traffic classification [19]. The traffic can be processed by supervised learning, also known as classification, or by unsupervised learning, also known as clustering [20][21]. The disadvantage of the ML approaches for network traffic analysis comes mainly from the lack of online (or as some authors refer to it: real-time) detection capabilities [22]. There are many prerequisites for the successful application of supervised learning [23] with – the most important of which is the annotation of the dataset. This is what makes the unsupervised clustering ML techniques, rule-based and anomaly-based approaches preferable in these scenarios.

III. Autonomous Systems for Cybersecurity Threat Detection Based on Deep Learning Techniques

The main idea of this work is to present a linear autonomous system for prepossessing of incoming traffic. The proposed system has the capability for file content analysis and is targeted towards cloud applications, which serve multimedia (Figure 1). The incoming traffic is analyzed in an IDS; cyberthreats are blocked based on rules, anomaly detection and correlation analysis.

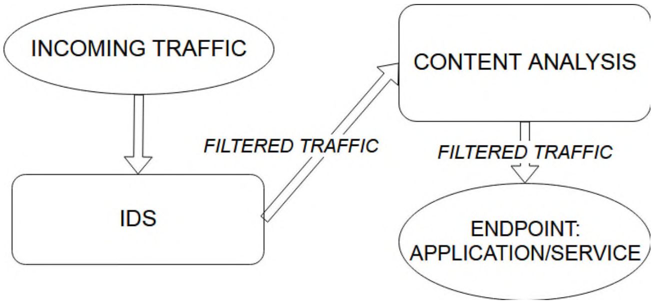

Figure 1. Workflow of the proposed protection cybersecurity protection system for cloud applications.

In the experiments, Suricata was used as well as a Surricata module based on the Google TensorFlow framework for Deep Learning [24]. The IDS filtered traffic was then subjected to content analysis where the traffic is decoded in a proxy server and the incoming text, video and images were analyzed with deep neural network classifiers (Figure 2) [25].

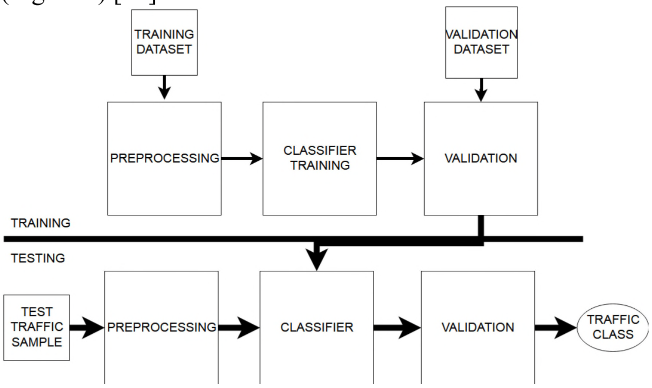

Figure 2. Classifier for network traffic analysis

With the appearance of large quantities of unstructured (or partially structured data) – the so called Big Data – and the improvement of computing power, deep learning has become extremely popular both for research and commercial purposes. ML algorithms are highly dependent on the choice of features. There are described cases with Bayesian classifiers where feature selection can greatly improve classification accuracy [25]. Deep learning techniques solve some of these challenges by automatically combining low-order features of the input, transforming and arranging them in order to calculate high-order features. In such scenario, it is not needed to add a manual step to eliminate for calculation of higher-order features of the training set. To an extent, the deep neural network structure is similar to the multi-layer neural network which includes input layer, hidden layer and output layer (Figure 3).

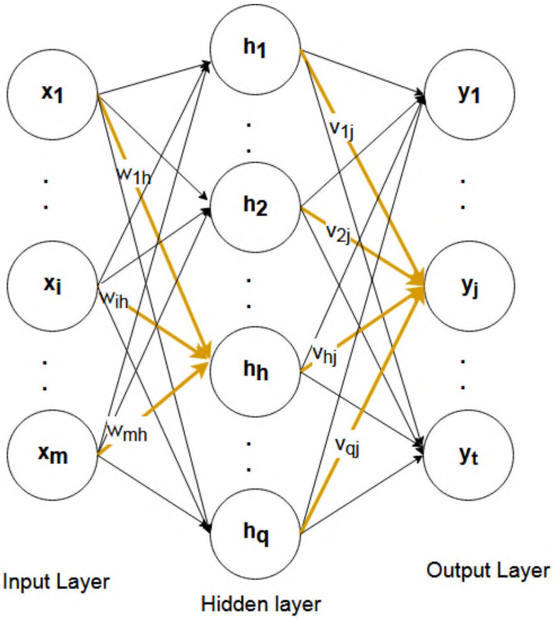

Figure 3. Multi-layer neural network general structure

The network parameters are initialized with random values, and the neuron weights are updated using the Back Propagation (BP) algorithm. In the standard neural network schema (Figure 2), the input for the of the j-th neuron from the output layer is calculated as follows:

$$o_j = \sum_{h=1}^{q} v_{hj} h_h \tag{1}$$

where v_{hj} is the weight of the connection of the hidden neuron *h* to the output neuron *j* and h_h is the output from the *h-th* hidden neuron. For the input of the *h-th* hidden neuron, the following is calculated:

$$\sigma_h = \sum_{i=1}^{q} w_{ih} x_i \tag{2}$$

where w_{hj} is the weight of the connection of the input neuron *i* to the hidden neuron *h* and x_i is the *i-th* input. For the k-th training sample (x_k, y_k), the output of the neural network is $\hat{y}_k = (\hat{y}_1^k, \hat{y}_2^k, \ldots, \hat{y}_t^k)$. With another representation known as offset term ϵ_j, it is given as $\hat{y}_t^k = f(o_j - \epsilon_j)$. The aim of the back-propagation training algorithm is to minimize the mean square error of the network on the k-th training sample. It is used for automatic update of the weights of the neural network. The regular multi-layer neural network carries the pitfalls of the disappearing gradient. With the increase in the number of layers, the number of weight parameters correspondingly grows, leading to a more complex model which can overfit [25]. Deep learning introduced the ReLU activation function, a new weight initialization method, a new loss function and new anti-fitting method (Dropout, regularization) to solve the traditional multi-layer perceptron disadvantages in terms of network structure and training capabilities.

IV. Combination of Classifier Results

The classifier combination in the proposed approach depends on the modality of the cyberthreat in each classifier. The final score is given through:

$$C_{out} = \operatorname{argmax}(C_i) \tag{3}$$

where C_{out} is the final class label and C_i is the output from the i-th classifier. The final score represents the most certain classifier [26] out of several classifiers which use different modalities and learning algorithms.

V. Experimental results

The Pytbull framework was used [27] to test the rules in Suricata. The accuracy of the detection with the most current rule sets was about 85%. The test setup included 4 virtual machines in private cloud infrastructure at the University of Telecommunications and Post, Sofia, Bulgaria.

A neural classifier was created using datasets obtained from [28][29] and modeled a neural network in the Weka [30] tool. The model delivered the highest accuracy of about 83% with 115 inputs four hidden neuron layers and 11 output neurons. The used dataset was derived from [23] containing 10 types of data with 249 attributes. Some of the classes contain fewer training samples and it was observed that other researches have excluded them from their training set.

Content analysis in terms of Spam detection was realized with a Convolutional Neural Network (CNN) trained in the Weka tool. The model was tested on the dataset [31] and the achieved accuracy in two classes was about 70%. Image classification was based on previous work [26] on human emotion analysis and is intended to be used on image data uploaded to a transparent proxy on the system. The achieved classification accuracy in 5 classes is about 73%.

VI. Conclusion and future work

In this paper, an approach was presented based on deep neural networks for design of autonomous cybersecurity threat detection systems in cloud applications. The proposed system uses 4 neural classifiers for network traffic, spam comments, spam email and images. The achieved results are comparable with contemporary approaches. The achieved accuracy for the individual components is comparable to other authors. The next steps will include expanding this framework and adopting it at the University of Telecommunications and Post, Sofia, Bulgaria.

Acknowledgment

This work is supported by the University of Telecommunications and Post (UTP), Sofia, Bulgaria, internal research grant Nr. NID16/03.04.2018 - "UNICLOUD2.0: Development and integration of cloud services in the learning process of UTP".

References

[1] J. Zhang, X. Chen, Y. Xiang, W. Zhou, and J. Wu, "Robust network traffic classification." IEEE/ACM Transactions on Networking 23, no. 4, 2015, pp. 1257-1270.

[2] G. Wagener, R. State and A. Dulaunoy, “Malware behaviour analysis”. Journal in Computer Virology. Vol.4., 2013, pp.279-287.

[3] F. Haddadi et al., “Botnet Behaviour Analysis using IP Flows With HTTP filters using classifiers”, Proceedings of the 28th International Conference on Advanced Information Networking and Applications Workshops, 2014, pp. 7-12

[4] A. S. Sohal, R. Sandhu, S. K. Sood, and V. Chang, "A cybersecurity framework to identify malicious edge device in fog computing and cloud-of-things environments." Computers & Security 74, 2018, pp. 340-354.

[5] M. Chambers, Reuters [retrieved: May 2019], “Germany sees big rise in security problems affecting infrastructure”, https://www.reuters.com/article/us-germany-cybersecurity-idUSKCN1Q60CS

[6] C. Haschek [retrieved: May 2019], “I scanned the whole country of Austria and this is what I've found”, https://blog.haschek.at/2019/i-scanned-austria.html

[7] R. L.-Langlois, [retrieved: May 2019], “Top 10 Intrusion Detection Tools: Your Best Free Options for 2019”, https://www.addictivetips.com/net-admin/intrusion-detection-tools/

[8] Y. Miao et al., "Automated Big Traffic Analytics for Cyber Security." arXiv preprint arXiv:1804.09023, 2018.

[9] Open Source Host-based Intrusion Detection System, [retrieved: May 2019], http://www.ossec.net/

[10] M. Roesch, “Snort - lightweight intrusion detection for networks,” inProceedings of the 13th USENIX conference on System administration, LISA ’99, 1999, pp. 229–238

[11] “Suricata, open source ids/ips/nsm engine.” [retrieved: May 2019], https://suricata-ids.org/

[12] “The Zeek Network Security Monitor”, [retrieved: May 2019], https://www.zeek.org/

[13] “The SAMHAIN file integrity / host-based intrusion detection system”, [retrieved: May 2019], https://www.la-samhna.de/samhain/

[14] “Fail2Ban – an intrusion prevention software (IPS) framework that protects computer servers from brute-force attacks”, [retrieved: May 2019], http://www.fail2ban.org

[15] “Security Onion – a free and open source Linux distribution for intrusion detection, enterprise security monitoring, and log management”, [retrieved: May 2019], https://securityonion.net/

[16] Check Point Software: Latest Global Threat Index November 2018’s Most Wanted Malware: The Rise of the Thanksgiving Day Botnet, [retrieved: May 2019] https://blog.checkpoint.com/2018/12/11/november-2018s-most-wanted-malware-the-rise-of-the-thanksgiving-day-botnet/

[17] Kaspersky Labs Technical Report, [retrieved: May 2019], “Botnet activity in H1 2018: Multifunctional bots becoming more widespread”, https://www.kaspersky.com/about/press-releases/2018_botnet-activity-in-h1-2018-multifunctional-bots-becoming-more-widespread

[18] DNS-BH- Malware Domain Blocklist, [retrieved: May 2019]. Available: http://www.malwaredomains.com/

[19] T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning." IEEE Communications Surveys & Tutorials 10, no. 4, 2008, pp.56-76.

[20] D. Zhao et al., „Botnet detection based on traffic behavior analysis and flow intervals”, Computers & Security,Vol. 39, 2013, pp. 2-16

[21] S. García, A. Zunino, and M. Campo, "Botnet behavior detection using network synchronism.", In Privacy, Intrusion Detection and Response: Technologies for Protecting Networks, IGI Global, 2012, pp. 122-144.

[22] S. Keshapagu and S. Suthaharan, "Analysis of datasets for network traffic classification.", In Topics from the 8th Annual UNCG Regional Mathematics and Statistics Conference, Springer, New York, NY, 2013, pp. 155-168.

[23] A. K. J. Michael, E. Valla, N. S. Neggatu, and A. W. Moore. “Network traffic classification via neural networks”, No. UCAM-CL-TR-912. University of Cambridge, Computer Laboratory, 2017.

[24] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin et al. "Tensorflow: a system for large-scale machine learning." In OSDI, vol. 16, 2016, pp. 265-283.

[25] J. H. Shu, J. Jiang, and J. X. Sun. "Network Traffic Classification Based on Deep Learning." Journal of Physics: Conference Series, vol. 1087, no. 6, 2018, p. 062021.

[26] S. Sokolov. "Neural Network Based Multimodal Emotion Estimation." ICAS 2018 vol 12, 2018, pp 4-7.

[27] Pytbull IDS/IPS testing framework, [retrieved: May 2019], http://pytbull.sourceforge.net/index.php?page=home

[28] D. Duaand and E. K. Taniskidou, “UCI Machine Learning Repository”, Irvine, CA: University of California, School of Information and Computer Science, 2017.

[29] Y. Meidan et al., "N-BaIoT—Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders." IEEE Pervasive Computing 17, no. 3, 2018, pp: 12-22.

[30] F. Eibe, M. A. Hall, and I. H. Witten, "The WEKA Workbench. Online Appendix for Data Mining: Practical Machine Learning Tools and Techniques." Morgan Kaufmann 2016.

[31] A. L. Maas et al., "Learning word vectors for sentiment analysis." In Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1, Association for Computational Linguistics, 2011, pp. 142-150.