Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2019) 000–000

Procedia Computer Science

www.elsevier.com/locate/procedia

The 17th International Conference on Mobile Systems and Pervasive Computing (MobiSPC)
August 9-12, 2020, Leuven, Belgium

Training Guidance with KDD Cup 1999 and NSL-KDD Data Sets of ANIDINR: Anomaly-Based Network Intrusion Detection System

Benedetto Marco Serinelli[a,*], Anastasija Collen[a], Niels Alexander Nijdam[a]

[a]*Centre Universitaire d'Informatique, University of Geneva, Geneva, Switzerland*

Abstract

In today's world, the protection of the computer networks remains one of the most crucial and difficult challenges in cyber security. In this work, a passive defence system ANIDINR is presented, aiming to monitor and protect computer networks. Our effort is focused on providing step-by-step guidance on methodologies selection and execution for the Machine and Deep Learning models' training. Taking as an input two data sets, five MDL models are evaluated. Our goals are to minimise the percentage of Undetected Attack, the percentage of False Alarm Rate and the overall testing time. Based on this set-up, the proposed system is capable to predict in near-to-real time well-known and zero-day computer network attacks.

Keywords: machine learning; deep learning; KDD Cup 1999 data set; NSL-KDD data set; Support Vector Machine; Random Forest; XGBoost; Neuralnet; Keras; anomaly-based network intrusion detection; network security

1. Introduction

Intrusion Detection Systems (IDSs) play a key role in *passive defence* [1], targeting to detect malicious activity in different application domains, such as Autonomous Vehicles (AV) [2] and Internet of Things (IoT) [3]. Furthermore, IDSs have been deployed in conjunction with *active defence systems*, such as honeypots. Two well known approaches exist in IDS research: Host-based Intrusion Detection System (HIDS) and Network Intrusion Detection System (NIDS). The first approach monitors the target machine's network interfaces and configurations, requiring specific settings attuned to the host machine [4]. For instance, Microsoft Windows has a different OS system configurations in comparison to Linux based systems, such as log files and OS calls. In contrast to the host based activity, a NIDS monitors all incoming and outgoing packets on the computer network and is designed upon *signature-* and *anomaly-based* approaches. A signature-based NIDS implements a predefined set of rules to detect attacks, such as

* Corresponding author. Tel.: +41-223-799-710
E-mail address: Benedetto.Serinelli@unige.ch

Snort rules[1]. Conversely, an anomaly-based NIDS is able to detect well-known and zero network attacks through recognition of network traffic profiles [4].

By overcoming the limitations of signature-based NIDS, such as detection of only well-known attacks and complex up-to-date collection of the rules, we focus our efforts on developing ANIDINR: an anomaly-based NIDS in *R*. It is composed of four modules, as shown in Fig.1, and one preliminary iterative phase. The *Packet Sniffer module* creates network packet profiles from captured network traffic. The *Training phase* takes as an input the KDD Cup 1999 data set (KDD) and NSL-KDD data set (NSL-KDD), generating the Machine and Deep Learning (MDL) prediction data structure of the computer network traffic profiles. If new data sets or MDL techniques are available, the above mentioned phase can be repeated several times. The *Prediction module* loads prediction data structures of the MDL models to evaluate the network profiles to foresee well-known and zero-day computer network attacks. Furthermore, if the Prediction module forecasts a computer network attack, the *Notification module* informs the users through a user-defined notifications, such as e-mail, Android and iOS notifications. Lastly, the *Decision Making module* is designed to put in place user-defined attack mitigation strategies. The effect of the mitigation strategies is retroactively verified by the Prediction module through new network traffic profiles.

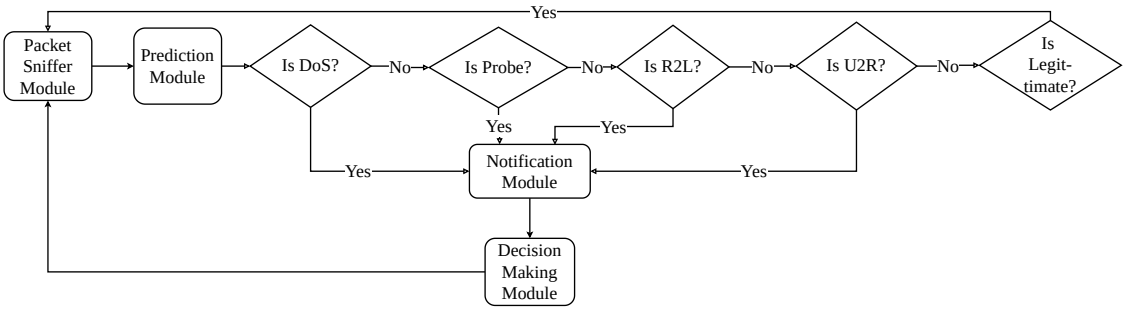

Fig. 1. Anomaly detection flow chart in the ANIDINR architecture

In conclusion, this research aims to describe a first evaluation of training phase with step-by-step guidance on methodologies selection and execution to bridge the gap in the existing academic literature.

2. Related work

The KDD and NSL-KDD data sets were chosen to design the training phase of ANIDINR. Taking as an input the same data sets, most existing studies have been based on several MDL approaches to propose intrusion detection models, such as Game Theory, Genetic Algorithm, Gaussian Naive Bayes, Logistic Regression, Decision Tree, Support Vector Machine (SVM), Random Forest (RF) and unsupervised clustering [5, 4, 6, 7, 8]. However, there is a number of shortcomings. Previous studies emphasised the use of *accuracy* to evaluate models' performance, but accuracy cannot be considered good metric, due to the unbalanced nature of the data sets [9], as shown in Tab. 1. In addition, the *testing time* metric should be used to evaluate the time needed to predict a computer network attack. The detection performance metrics, False Alarm Rate (FAR) and Undetected Attack (UA) [10, 11, 12], should be chosen to compare models' performance. The ANIDINR Training phase were implemented in R, due to the language characteristics known to be powerful, popular and open source programming language for statistical computing, employed to solve various data science competitions[2]. Finally, the previous studies are limited to the description of results, without providing guidelines how to dive into deep analysis phase. This paper addresses identified gap, proposing an exhaustive explanation on data analysis phase.

[1] https://www.snort.org/downloads/#rule-downloads
[2] https://www.kaggle.com/

Table 1. Number of examples for each class inside the KDD and NSL-KDD, proofing the unbalance between them

Data set	Legitimate	DoS	Probe	U2R	R2L	Total
KDD	70217	5728	520	50	381	76896
NSL-KDD	53561	12517	2976	50	379	69483

Table 2. Summary of research contributions on IDS

Data set	Model	Accuracy (%)
KDD	J48 [7]	74.60
	Naive Bayes [7]	74.40
	NB Tree [7]	75.40
	RF [7]	74.00
	Random Tree [7]	72.80
	Multi-layer Perceptron [7]	78.10
	SVM [7]	74.00
	SVM + K-means [13]	97.75
	Genetic algorithm [13]	90.00
	SVM + BIRCH clustering [13]	95.70
	MOGFIDS [13]	93.20
	Association rules [13]	92.40
	Multi-class SVM [13]	92.46
	Winning the KDD99 [13]	93.30
	DNN [4]	96.3
	Multi-CNN [7]	86.95
NSL-KDD	DNN [4]	91.5
	AE [14]	87.00
	LSTM [14]	80.67
	MLP [14]	81.43
	L-SVM [14]	81.40
	Q-SVM [14]	83.65
	LDA [14]	83.17
	QDA [14]	79.47

3. Training phase

The ANIDINR training phase is designed through *epicycle* and *exploratory data analysis* methodology framework guidelines for training the models [15] and validating the use of two data sets. They are focused and limited to the description of *network protocol*, *connection flags*, *mean attack duration* and *application layer protocols*[3]. From the training data it can be observed that the amount of samples for Internet Control Message Protocol (ICMP) and User Datagram Protocol (UDP) is smaller than the amount of Transmission Control Protocol (TCP) samples. Indeed, a vast number of application layer protocols run over TCP protocol, such as Hypertext Transfer Protocol (HTTP) and File Transfer Protocol (FTP). In addition, the information on connection flags allows to mitigate protocol vulnerabilities, such as *TCP Flags Invalid Combinations*, *TCP Fragment*, *Syn Flood* and *TCP Session Hijacking*. The flag information describes in detail the status of the connection and is not limited to a connection attempt without reply (flag

[3] http://kdd.ics.uci.edu/databases/kddcup99/task.html

S0), a connection established and not terminated (flag *S1*), a connection refused (flag *REJ*), a connection established and terminated from source/destination due to no reply from destination/source (respectively flags *S2* and *S3*) [16]. The mean duration attack analysis shows that Denial of Service (DoS) attacks were performed in the shortest time possible. On the contrary, the mean duration of probe (Probe) attacks had underlined a large amount of time to gather all necessary information. The long mean duration of Probe attacks could be attributed to the *stealth scan*[4] to bypass IDS, for discovering network topology, host operating system and the host's open ports. In conclusion, the data sets have sufficient number of records at the application layer protocols on the destination, not limited to HTTP, FTP, Simple Mail Transfer Protocol (SMTP), Internet Message Access Protocol (IMAP), Domain Name System (DNS), Lightweight Directory Access Protocol (LDAP), Network News Transfer Protocol (NNTP), Secure Shell (SSH), Post Office Protocol (POP), Media Transfer Protocol (MTP), Internet Relay Chat (IRC), Telnet (Telnet) Gateway Protocol (BGP). The aforementioned information allows to create network profiles encapsulating application protocol vulnerabilities, such as HTTP Flooding, FTP Flooding, Telnet DoS and DNS Flood. In addition, NSL-KDD is an improved version of the KDD, where the number of duplicated examples is significantly reduced[5]. To improve the quality of classification models, *duplicated records and null values* were removed, reducing the complexity of the MDL models.

Lastly, *grouping examples by attack types* allowed to categorise the attacks by its type. Furthermore, the examples in the data sets were grouped into the following malicious classes[6] DoS (back, land, neptune, pod, smurf and teardrop), Probe (ipsweep, nmap, portsweep and satan), Root to Local (R2L) (ftp_write, guess_passwd, imap, multihop, phf, spy, warezclient and warezmaster) and User to Root (U2R) (buffer_overflow, loadmodule, perl and rootkit), reformulating the MDL problem into multi-class classification. Thus, the Training phase allows to generate computer network traffic prediction data structure based on legitimate and four malicious classes. Specifically, ANIDINR will be able to classify the network traffic in legitimate, DoS, Probe, R2L and U2R through Prediction module. In addition, the zero-day attack network traffic profiles can be classified as legitimate or one of the malicious classes. However, the classification of zero-day attack could not be accurate. The FAR and the UA perform misclassification due to the similarities of network traffic profiles with legitimate or malicious traffic.

In conclusion, three Machine Learning models (SVM, RF and XGBoost (XGBoost)) and two Deep Learning (DL) models (Neuralnet (Neuralnet) and Keras (Keras)) were trained on the cleaned data sets. The results are illustrated in Tab. 3.

Table 3. Results of the training phase

Data set	Model	Accuracy (%)	DoS FAR (%)	Probe FAR (%)	R2L FAR (%)	U2R FAR (%)	UA (%)	Testing Time (s)
KDD	SVM	97.15	2.71	0.10	0.26	0.00	1.97	4.16
	RF	98.99	0.99	0.05	0.03	0.00	0.76	18.50
	XGBoost	98.32	0.98	0.80	0.03	0.03	0.59	0.06
	Neuralnet	97.84	1.24	0.26	0.10	0.05	2.34	562.20
	Keras	96.85	3.08	0.36	0.50	0.00	1.02	3.10
NSL-KDD	SVM	93.80	4.91	16.70	17.80	42.86	2.12	16.60
	RF	97.93	0.92	1.16	0.09	0.11	0.62	28.26
	XGBoost	97.67	1.41	1.11	0.15	0.04	0.54	0.08
	Neuralnet	95.28	1.52	1.90	0.09	0.00	4.09	484.08
	Keras	93.82	1.87	3.19	1.31	0.00	2.42	3.90

[4] https://nmap.org/book/subvert-ids.html
[5] https://www.unb.ca/cic/datasets/nsl.html
[6] http://kdd.ics.uci.edu/databases/kddcup99/training_attack_types

4. Improvements and results

To reach better performance, several improvements were implemented to clean further the data sets for reducing the complexity of the MDL models.

The *correlation analysis* allowed to remove highly correlated features from data sets. The correlated features can be rewritten as linear combinations between them. Thus, they do not enrich the information content for the MDL models.

Random Forest importance analysis allowed to understand how the Mean Squared Error (MSE) increases if the features to be randomly permuted. The features, which increased the MSE, were not discarded for keeping important information regarding computer network attacks, such as *duration* and *the exchanged number of byte*.

The operations of the *integer encoding* and the *normalisation with mean zero and standard deviation one* reduced model complexity and misclassification errors of the DL models [17]. After performing the normalisation process, *null values* were removed.

The classes distribution of a data sets was unbalanced, as shown in Tab. 1. Furthermore, the models could incorrectly classify U2R and R2L classes. Thus, *random under-sampling technique* was implemented to balance the class distribution of a data set. The random under-sampling was applied as follows: decreasing up to 7000 the legitimate class examples, rounding to 5000 the DoS class examples and keeping all examples for other class. Although it is a rogue technique, random under-sampling is a widely used operation for balancing data set [18, 19]. However, it hides three drawbacks: loss of information, the increase of the variance and the warping of the posterior distribution [18].

As shown in Tab. 3, the U2R FAR for some models are equal to 0.00%. Thus, the models always predict U2R attacks. Although it could be considered as an excellent result, the aforementioned FAR value can be due to a misclassification problem. Based on the result depicted in Tab. 3, the Neuralnet model trained on the KDD shows a huge testing time. Moreover, the performance of XGBoost models shows better performance than the SVM and RF models. Comparing XGBoost models, the model trained on the KDD achieves higher performance in terms of FAR and testing time than the same model trained on the NSL-KDD. Based on the obtained results, XGBoost model, trained on the KDD, achieves the best performance. Furthermore, its accuracy is also higher than the research community, comparing Tab. 2 and Tab. 3. Finally, the accuracy of the SVM and the Random Forest models, as shown in Tab. 3 and both trained in R, is higher or closer than the research community, as illustrated in Tab. 2.

5. Conclusions and future work

In this paper the step-by-step guidance on methodologies selection and execution is proposed for ANIDINR: an anomaly-based NIDS, designed for predicting near-to-real time well-known and zero-day computer network attacks. The well-known computer network attack examples are used to train MDL model on the network computer attack profiles. By its definition, network traffic associated to zero-day attack can not be described in any data set, before forensic analysis is performed. Furthermore, the ANIDINR should be able to detect a zero-day computer network attacks as anomaly that does not match the legitimate network traffic profiles patterns. However, it could classify wrongly the attack class when attack will be predicted, introducing a misclassification error and erroneous attack mitigation strategies. Moreover, an important issue arises if an adversary launches a zero-day attack, which generates a network traffic profiles closer to the legitimate network profiles. In this case, ANIDINR will not be able to detect the zero-day attack, identifying it as a legitimate network computer traffic. In conclusion, the results prove that XGBoost model, trained on the KDD, reaches the best performance in terms of FAR, testing time and accuracy in comparison to the research community models.

In the future, our efforts will be focused on the following improvements for training phase: *one-hot encoding*, *enlarging data sets' dimension*, *K-fold cross-validation* and *reducing the complexity of Keras model*. After each improvement, the model will be re-trained to re-evaluate the metrics.

The source code is available at the GitHub repository[7].

[7] https://github.com/marksniper/Network-Intrusion-Detection-System

Acknowledgements

This work has received funding by the European Union's Horizon 2020 Research and Innovation Programme through GHOST project (https://www.ghost-iot.eu/) under grant agreement No 740923 and nIoVe project (https://www.niove.eu/) under grant agreement No 833742.

References

[1] W. Wu, R. Li, G. Xie, J. An, Y. Bai, J. Zhou, K. Li, A Survey of Intrusion Detection for In-Vehicle Networks, IEEE Transactions on Intelligent Transportation Systems PP (1) (2019) 1–15. doi:10.1109/TITS.2019.2908074.

[2] V. L. L. Thing, J. Wu, Autonomous Vehicle Security: A Taxonomy of Attacks and Defences, in: Proceedings - 2016 IEEE International Conference on Internet of Things; IEEE Green Computing and Communications; IEEE Cyber, Physical, and Social Computing; IEEE Smart Data, iThings-GreenCom-CPSCom-Smart Data 2016, 2017, pp. 164–170. doi:10.1109/iThings-GreenCom-CPSCom-SmartData.2016.52.

[3] M.-a. Omer, Implementing security techniques to lower the probability of IoT- devices getting hacked (2019).

[4] S. Choudhary, N. Kesswani, Analysis of KDD-Cup'99, NSL-KDD and UNSW-NB15 Datasets using Deep Learning in IoT, Procedia Computer Science 167 (2019) (2020) 1561–1573. doi:10.1016/j.procs.2020.03.367.
URL https://doi.org/10.1016/j.procs.2020.03.367

[5] I. Homoliak, M. Teknos, M. Ochoa, D. Breitenbacher, S. Hosseini, P. Hanacek, Improving Network Intrusion Detection Classifiers by Non-payload-Based Exploit-Independent Obfuscations: An Adversarial Approach, arXiv preprint arXiv:1805.02684 (2018).

[6] Y. Al-Hadhrami, F. K. Hussain, Real time dataset generation framework for intrusion detection systems in IoT, Future Generation Computer Systems 108 (2020) 414–423. doi:10.1016/j.future.2020.02.051.
URL https://doi.org/10.1016/j.future.2020.02.051

[7] Y. Li, Y. Xu, Z. Liu, H. Hou, Y. Zheng, Y. Xin, Y. Zhao, L. Cui, Robust detection for network intrusion of industrial IoT based on multi-CNN fusion, Measurement: Journal of the International Measurement Confederation 154 (2020) 107450. doi:10.1016/j.measurement.2019.107450.
URL https://doi.org/10.1016/j.measurement.2019.107450

[8] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, H. Janicke, Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study, Journal of Information Security and Applications 50 (2020) 102419. doi:10.1016/j.jisa.2019.102419.
URL https://doi.org/10.1016/j.jisa.2019.102419

[9] J. Akosa, Predictive accuracy: a misleading performance measure for highly imbalanced data, in: Proceedings of the SAS Global Forum, 2017, pp. 2–5.

[10] P. Aggarwal, S. K. Sharma, Analysis of KDD dataset attributes-class wise for intrusion detection, Procedia Computer Science 57 (2015) 842–851.

[11] M. Aldwairi, Y. Khamayseh, M. Al-Masri, Application of artificial bee colony for intrusion detection systems, Security and Communication Networks 8 (16) (2015) 2730–2740.

[12] A. L. Buczak, E. Guven, A survey of data mining and machine learning methods for cyber security intrusion detection, IEEE Communications Surveys & Tutorials 18 (2) (2015) 1153–1176.

[13] W. L. Al-Yaseen, Z. A. Othman, M. Z. A. Nazri, Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system, Expert Systems with Applications 67 (2017) 296–303.

[14] C. Ieracitano, A. Adeel, F. C. Morabito, A. Hussain, A novel statistical analysis and autoencoder driven intelligent intrusion detection approach, Neurocomputing 387 (2020) 51–62. doi:10.1016/j.neucom.2019.11.016.
URL https://doi.org/10.1016/j.neucom.2019.11.016

[15] R. D. Peng, E. Matsui, The Art of Data Science, Skybrude Consulting, LLC, 2015.

[16] J. Song, H. Takakura, Y. Okabe, Description of Kyoto University Benchmark Data, Tech. rep., National Institute of Information and Communications Technology (NICT) (2010).

[17] F. Chollet, J. J. Allaire, Deep Learning with R., Manning Publications, 2017.

[18] A. Dal Pozzolo, O. Caelen, G. Bontempi, When is undersampling effective in unbalanced classification tasks?, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, 2015, pp. 200–215.

[19] C. V. KrishnaVeni, T. Sobha Rani, On the classification of imbalanced datasets, IJCST 2 (SP1) (2011) 145–148.