Borys LYPA[1], Oleh IVER [2], Viktor KIFER[3]

Supervisor: Nataliya ZAGORODNA[4]

APPLICATION OF MACHINE LEARNING METHODS FOR NETWORK INTRUSION DETECTION SYSTEM

Summary: The article is devoted to the application of the machine learning algorithms for intrusion detection in networks. Creating and training intrusion detection system (IDS) using machine learning is mainly limited by the out-of-date open available datasets. The most popular machine learning classification models like decision tree, random forest, and linear support vector classification will be researched based on the CSE-CIC-IDS2018 dataset.

Keywords: Intrusion detection, cybersecurity, machine learning, CSE-CIC-IDS2018 dataset

ZASTOSOWANIE METOD UCZENIA MASZYNOWEGO DO BUDOWY SYSTEMU WYKRYWANIA CYBERWŁAMAŃ

Streszczenie: W artykule omówiono zastosowanie algorytmów uczenia maszynowego do detekcji cyberwłamań w sieciach. Tworzenie oraz uczenie systemów detekcji cyberwłamań (IDS) z zastosowaniem uczenia maszynowego jest ograniczone głównie poprzez dostępność do zbiorów/zestawów danych, które są zdezaktualizowane (out-of-date). W pracy badano najbardziej popularne modele klasyfikacji oparte o uczenie maszynowe – takie jak: drzewa decyzyjne, losowe lasy (w sensie teorii grafów), a także liniową klasyfikacje wektorową. Badania te przeprowadzono na specjalnych zbiorach danych CSE-CIC-IDS2018.

Słowa kluczowe: :detekcja cyberwłamań, cyberbezpieczeństwa, uczenie maszynowe, zbiór danych CSE-CIC-IDS2018

[1] Ternopil Ivan Puluj National Technical University, Ukraine; Faculty of Computer Information Systems and Software Engineering; Cybersecurity; boryslypa1@gmail.com
[2] Ternopil Ivan Puluj National Technical University, Ukraine; Faculty of Computer Information Systems and Software Engineering; Computer Engineering; olehiwer@gmail.com
[3] PhD student, Ternopil Ivan Puluj National Technical University, Ukraine; Faculty of Computer Information Systems and Software Engineering; Cybersecurity department; kifervictor@gmail.com
[4] PhD, Ternopil Ivan Puluj National Technical University, Ukraine; Faculty of Computer Information Systems and Software Engineering; Cybersecurity department; zagorodna.n@gmail.com

1. Introduction

Internet and social networks are increasingly changing our life, but they also expose us to serious security threats.

Due to the fact that in the past years computer networks, services and systems faced more and more threats security in cyberspace has become a very important problem. Identification of various network attacks, especially not previously seen attacks is a key issue to be solved urgently.

IDS are an important part of cybersecurity. They detect intrusions and abnormal behavior in networks or other information systems. IDS usually apply one of two detection principles: signature-based or anomaly-based algorithms [2]. Signature-based approach means usage of manually created rules that detect intrusions, whereas anomaly-based systems try to profile normal behavior and detect abnormal situation dynamically. It is also possible to combine these approaches to form a hybrid IDS. In this case signatures are created automatically and can be periodically updated [3].

Nowadays almost all computer systems generate big data. Classical IDS cannot effectively process huge datasets and response to new threats. So it was a need to find a way of fast data analysis and effective search of anomalies. Machine learning (ML) and Data Mining algorithms came in handy and offers many benefits for intrusion detection. In general machine learning is an approach of artificial intelligence (AI) that uses a system which capable to learn from experience. In other words, ML is a system that can recognize patterns by using examples rather than by programming them. However, it has some restriction of usage which should be considered.

In the research, we use decision trees, random forest and support vector machine algorithms as they belong to the popular classification methods and work fast enough.

The performance of a machine learning algorithm largely depends on the dataset it is trained on. Since intrusion methods are improving rapidly, a valid dataset is required to construct adequate defense model [4].

2. Dataset

We built a classification model on a realistic cyber defense dataset provided by Canadian Institute for Cybersecurity (CIC) on AWS (Amazon Web Services) [5]. The final dataset includes seven different attack scenarios: Brute-force, Heartbleed, Botnet, DoS, DDoS, Web attacks, and infiltration of the network from inside. The attacking infrastructure includes 50 machines; the victim organization has 5 departments and includes 420 machines and 30 servers. The dataset includes the captures network traffic and system logs of each machine, along with 80 features extracted from the captured traffic using CICFlowMeter-V3 [6,7].

The description attacks are given below according to information provided on CSE-CIC-IDS2018 dataset`s page [9].

Brute force attacks consist of submitting many passwords or passphrases with the hope of eventually correct guess. They are very common against networks, as they tend to break the accounts with weak username and password combinations. There were two modules, FTP and SSH on the Kali Linux machine as the attacker and an Ubuntu 14.0 system as the victim machine. A large dictionary that contains 90 million words was used as a list of passwords.

The Heartbleed Bug is a serious vulnerability in the popular OpenSSL cryptographic software library. This weakness allows stealing the information protected under normal conditions by the SSL/TLS encryption which used to secure the Internet. SSL/TLS provides communication security and privacy over the Internet for applications such as web, email, instant messaging (IM) and some virtual private networks (VPNs) [8]. The Heartleech is one of the most famous tools to exploit Heartbleed which was used in this dataset.

In botnet scenario machines are infected with two different botnets (Zeus and Ares), every 400 seconds screenshots are also requested from the zombies. Zeus is a Trojan horse malware package that runs on some versions of Microsoft Windows. While it can be used to carry out many malicious and criminal tasks, it is often used to steal banking information by man-in-the-browser keystroke logging and form grabbing. It is also used to install the Crypto-Locker ransomware. Zeus is spread mainly through drive-by downloads and phishing schemes. Also, it is used as a complement Ares botnet which is an open-source botnet.

In the Denial-of-Service (DoS) a Slowloris Perl-based tool is used to take down the web server. Slowloris is a type of denial of service attack tool invented by Robert Hansen which allows a single machine to take down another machine's web server with minimal bandwidth and side effects on unrelated services and ports.

In the Distributed Denial-of-Service scenario the High Orbit Ion Cannon (HOIC) tool was used to conduct DDoS attack by using 4 different computers. The High Orbit Ion Cannon is an open source network designed as stress testing and denial-of-service attack application to attack as many as 256 URLs at the same time.

In the web application attacks scenario, Damn Vulnerable Web App (DVWA) was used which was developed to be an aid for security professionals to test their skills, as victim web application. In the first step, the website is scanned through a web application vulnerability scanner and then different types of web attacks are conducted on the vulnerable website, including SQL injection, command injection, and unrestricted file upload.

In the infiltration of the network from inside scenario, a vulnerable application (such as Adobe Acrobat Reader 9) should be exploited. First the victim receives a malicious document through the email. Then, after successful exploitation using Metasploit framework, a backdoor will be executed on the victim's computer. Now different attacks can be conducted on the victim's network include IP sweep, full port scan and service enumerations using Nmap.

The brief summary of dataset is CSV file with more than 80 features is given in Table 1.

Table 1. List of executed attacks and duration [9]

Attack	Tools	Duration	Attacker	Victim
Bruteforce attack	FTP – Patator SSH – Patator	One day	Kali linux	Ubuntu 16.4 (Web Server)
DoS attack	Hulk, GoldenEye, Slowloris, Slowhttptest	One day	Kali linux	Ubuntu 16.4 (Apache)

DoS attack	Heartleech	One day	Kali linux	Ubuntu 12.04 (Open SSL)
Web attack	• Damn Vulnerable Web App (DVWA) • In-house selenium framework (XSS and Brute-force)	Two days	Kali linux	Ubuntu 16.4 (Web Server)
Infiltration attack	• First level: Dropbox download in a windows machine • Second Level: Nmap and portscan	Two days	Kali linux	Windows Vista and Macintosh
Botnet attack	• Ares (developed by Python): remote shell, file upload/download, capturing • screenshots and key logging	One day	Kali linux	Windows Vista, 7, 8.1, 10 (32-bit) and 10 (64-bit)
DDoS+PortScan	Low Orbit Ion Canon (LOIC) for UDP, TCP, or HTTP requests	Two days	Kali linux	Windows Vista, 7, 8.1, 10 (32-bit) and 10 (64-bit)

List of extracted traffic features are described in table 2.

Table 2. List of extracted traffic features [10]

Feature	Description	Type
Label	indicate whether the traffic is malicious or not, e.g., benign, SQLInjection, etc.	String
Dst Port	Destination port number	Integer
Protocol	Protocol	Integer
TimeStamp	Time Stamp of the flow	String
Flow duration	Flow duration	Integer
Tot Fwd/Bwd Pkts	Total packets in forward/backward directions	Integer
TotLen Fwd/Bwd Pkts	Total size of packets in forward/backward directions	Integer
Fwd/Bwd Pkt Len Max/Min/Mean/Std	Maxi/Mini/Average/Std. Dev. size of package in forward/backward directions	Integer
Flow Byts/s & Flow Pkts/s	Flow byte rate, i.e., number of packets per seconds	Float64
Flow IAT Mean/Std/ Max/Min	Average/Std. Deviation/Maxi/Mini time between two flows	Float64
Fwd/Bwd IAT Tot/Mean/ Std/- Max/Min	Total/Average/Std. Deviation/Maxi/Mini time between two packets in forward/backward directions	Float64
Fwd/Bwd PSH/URG Flags	Number of times the PSH/URG flag was set in packets in forward/backward direction	Integer
Fwd/Bwd Header Len	Total bytes used for header in forward/backward direction	Integer
Fwd/Bwd Pkts/s	Number of forward/backward packets	Float64

	per second	
Pkt Len Min/Max/Mean/Std	Maxi/Mini/Average/Std. Dev. length of a flow	Integer
Pkt Len Var	Mini inter-arrival time of packet	Float64
FIN/SYN/RST/PUSH/ACK/ URG/CWE/ECE Flag Cnt	Number of packets with FIN/SYN/RST/PUSH/ACK-/URG/CWE/ECE	Integer
Down/Up Ratio	Download/upload ratio	Integer
Pkt Size Avg	Average size of packets in forward/backward direction	Float64
Fwd/Bwd Seg Size/Byts/b/Blk Rate Avg	Average number of bulk rate/bytes bulk rate/packets bulk rate in forward/backward directions	Float64
Subflow Fwd/Bwd Pkts/Byts	The average number of bytes/packets in a sub flow in forward/backward direction	Integer
Init Fwd/Bwd Win Byts	Number of bytes sent in initial window in forward/backward directions	Integer
Fwd Act Data Pkts	Number of packets with at least 1 byte of TCP data payload in forward	Integer
Fwd Seg Size Min	Minimum segment size observed in forward	Integer
Active Mean/Std/Max/Min	Maxi/Mini/Average/Std. Dev. a flow was active before becoming idle	Float64
Idle Mean/Std/Max/Min	Maxi/Mini/Average/Std. Dev. a flow was idle before becoming active	Float64

Moreover, before starting training a model we preprocess data, included in the CSE-CIC-IDS2018 dataset. Some steps were done to work out missing values and reduce the size of dataset:

- Delete features which do not affect the performance of ML model, for example, “TimeStamp” column;
- Replace “Infinity” and “NaN” values with mean value for each column;
- Format data into standard datatype;

Let us consider an example where the ML model takes into account a total number of packets in forward direction per second or flow byte rate per second to detect malicious traffic. It is quite reasonable that these features can vary from network to network due to differences in networks bandwidth. While running the model on test dataset we observed reduction in precision, so to prevent such noises we normalized data. Normalization makes training less sensitive to the scale of features, so we can find better solution. Most of the numerous data was replaced with its standard deviation and then rescaled from -1 to 1 by using `sklearn.preprocessing.MinMaxScaler`.

To simulate real-life traffic we downloaded .pcap files with similar attacks logs from Stratosphere Lab [11] containing normal and malicious traffic, extracted the same features as in training dataset by using CICFlowMeter-V4 [6,7]. We took the same preprocessing steps to the test dataset as to the training one.

3. Machine Learning Methods

Our research was conducted in order to identify whether flow is benign or malicious, based on learning on a set of labeled in advance data. In this case our problem belongs to a supervised classification problem.

We have selected the popular machine learning models:
- Decision tree classifier;
- Random forest classifier;
- Linear Support Vector Classification.

Decision tree is a tree structure, used as a predictive model, in which each node is created to test one feature, and a branch is a test output with each leaf node representing a category.

Random forest (RF). A random forest is a set of decision trees which considers the output of each tree before providing a unified final response. Each decision tree is a conditional classifier: the tree is analyzed from the top. A given condition is checked against one or more features of the analyzed data at each node. These methods are efficient for large datasets and especially for multiclass problems, but deeper trees might lead to overfitting [12].

Linear Support Vector Classification (LinearSVC) is a Support Vector Classification (SVC) with linear kernel, but have differences in implementation, and works better with large numbers of samples [13]

We implemented these models using Anaconda 3 and the latest Scikit learn version 0.21.3 [14] and Pandas version 0.25.3 [15] in Jupyter Notebook. For each evaluated model we found best parameters using sklearn.model_selection.GridSearchCV.

4. Evaluation

The evaluation of experiments include cross validation of the training CIC-AWS-2018 Dataset on each of the attack types and the prediction using the model on test dataset. The result of classifiers accuracy are presented in the following tables, grouped by type of attack. True Positive (TP) means the percentage of positive samples correctly classified by the model and True Negative (TN) means the percentage of negative samples correctly classified by the model Accuracy: (TP + TN)/(all instances = TP + TN + False Positive + False Negative). Ratio of the number of correctly classified samples to the total number of samples for a given test data set. In tables, we present two values: Training which represent results achieved on training CIC-AWS-2018 Dataset and Test which represent result on the test dataset, described above.

Table 3. Evaluation of Machine Learning methods for DoS attack

	Random forest		Decision Tree		LinearSVC	
	Train	Test	Train	Test	Train	Test
TP	0.99	1	0.99	0.51	0.94	0.02
TN	0.99	0.01	0.99	0.91	0.99	0.97
Accuracy	0.99	0.97	0.99	0.52	0.97	0.02

Table 4. Evaluation of Machine Learning methods for Botnet attack

	Random forest		Decision Tree		LinearSVC	
	Train	Test	Train	Test	Train	Test
TP	0.99	0	0.99	0.03	0.8	0.01
TN	0.99	1	0.99	0.86	0.89	0.75
Accuracy	0.99	0.02	0.99	0.04	0.8	0.02

Table 5. Evaluation of Machine Learning methods for Brute force attack

	Random forest		Decision Tree		LinearSVC	
	Train	Test	Train	Test	Train	Test
TP	0.8	-	0.8	-	1	-
TN	1	-	1	-	1	-
Accuracy	0.99	-	0.99	-	1	-

We didn`t manage to find logs with similar attack type to test the model on, so only results of train dataset are presented.

Table 6. Evaluation of Machine Learning methods for DDoS attack

	Random forest		Decision Tree		LinearSVC	
	Train	Test	Train	Test	Train	Test
TP	1	0.31	1	0.31	1	0.32
TN	0.99	0.98	0.99	0.98	0.99	0.99
Accuracy	0.99	0.5	0.99	0.5	0.99	0.5

Table 7. Evaluation of Machine Learning methods for Web Application attack

	Random forest		Decision Tree		LinearSVC	
	Train	Test	Train	Test	Train	Test
TP	0.8	0.06	0.8	0.1	0.46	0
TN	1	0.96	0.99	0.95	0.99	0.99
Accuracy	0.99	0.07	0.99	0.11	0.99	0.06

Table 8. Evaluation of Machine Learning methods for Infiltration attack

	Random forest		Decision Tree		LinearSVC	
	Train	Test	Train	Test	Train	Test
TP	1	0.2	1	0.2	0.99	0.1
TN	1	1	1	1	0.34	0.2
Accuracy	1	0.90	1	0.90	0.78	0.16

5. Conclusion

Our research examines three common machine learning classifications models trained on the CIC-AWS-2018 Dataset and tested on datasets that we extracted from .pcap logs created by Stratosphere Lab. During the research, we found out that logs

created with the same attack programs but with different parameters or with some gap in time can have very different values of features. That is why the results on test data is appeared to be worse comparing to train data. To improve them we normalized test dataset values of attributes. Usually, it gave from 10 to 20 percent of improvement.

Although Decision tree showed the best performance and may be used in some situations. We think that trained models are far from ready to use in real-life situations. Much is left to do in the future, for example, finding better way to preprocess and normalize dataset, improvement of developed models and testing new algorithms in order to fit better the statistical data, testing on more types of intrusions dynamically.

REFERENCE

1. AFTERGOOD S.: Cybersecurity: The cold war online, *Nature*, vol. 547, pp. 30-31, Jul. 2017.
2. SCARFONE K, MELL P.: Guide to intrusion detection and prevention systems (IDPS). NIST Special Publication 2007;800(2007):94.
3. ANTTI JUVONEN, TUOMO SIPOLA: Anomaly Detection Framework Using Rule Extraction for Efficient Intrusion Detection, 2014
4. SOMMER R., PAXSON V.: Outside the closed world: On using machine learning for network intrusion detection, in: 2010 IEEE symposium on security and privacy, IEEE, 2010, pp. 305–316.
5. Internet service Registry of Open Data on AWS: *https://registry.opendata.aws/cse-cic-ids2018/*
6. LASHKARI A. H., DRAPER-GIL G., MAMUN ,M.S.I., GHORBANI A.A.: Characterization of Tor Traffic Using Time Based Features, In the proceeding of the 3rd International Conference on Information System Security and Privacy, SCITEPRESS, Porto, Portugal, 2017
7. DRAPPER-GIL G., LASHKARI A.H., MAMUN M., GHORBANI A.A.: Characterization of Encrypted and VPN Traffic Using Time-Related Features, In Proceedings of the 2nd International Conference on Information Systems Security and Privacy(ICISSP 2016) , pages 407-414, Rome , Italy
8. Internet service The Heartbleed Bug: *http://heartbleed.com/*
9. Internet service University of New Brunswick: *https://www.unb.ca/cic/datasets/ids-2018.html*
10. QIANRU ZHOU, PEZAROS D.: Evaluation of Machine Learning Classifiers for Zero-Day Intrusion Detection – An Analysis on CIC-AWS-2018 dataset
11. Internet service Stratosphere Lab: *https://www.stratosphereips.org/datasets-overview*
12. APRUZZESE G., COLAJANNI M., FERRETTI L., GUIDO A., MARCHETTI M.: On the Effectiveness of Machine and Deep Learning for Cyber Security
13. Internet service Scikit-learn: *https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html*
14. Internet service Scikit-learn: *https://scikit-learn.org/stable/index.html*
15. Internet service Pandas 0.25.3 documentation: *https://pandas.pydata.org/pandas-docs/stable/index.html*