Proceedings of the 18th Conference on Computer Science and Intelligence Systems pp. 1295–1299
DOI: 10.15439/2023F2385
ISSN 2300-5963 ACSIS, Vol. 35

Tackling Variable-length Sequences with High-cardinality Features in Cyber-attack Detection

Chang Lin
State Key Laboratory of Information Photonics and Optical Communications
Beijing University of Posts and Telecommunications
Beijing, China
bupt.ipoc@yandex.com

***Abstract*—Internet of Things (IoT) based systems are vulnerable to various cyber-attacks and need advanced and smart techniques in order to achieve their security. In the FedCSIS 2023 big-data competition, participants are asked to construct scoring models to detect whether anomalous operating systems were under attack by using logs from IoT devices. These log files are variable-length sequences with high cardinality features. Through in-depth and detailed analysis, we find out concise and efficient methods to handle these huge volumes, variety, and veracity of data. On the basis of this, we create detection rules using the fundamental knowledge of mathematical statistics and train gradient boosting machine (GBM) based classifier for attack detection. Experimental and competition results prove the effectiveness of our proposed methods. Our final AUC score is 0.9999 on the private leaderboard.**

***Index Terms*—Internet of Things; Cyber security; Machine Learning; Variable-length Sequences; High-cardinality Features**

I. Introduction

INTERNET of Things (IoT) plays an essential role in remote monitoring and control operations. IoT based systems are widely used in the fields of environment, home automation, healthcare, smart grid, transportation, agriculture, military, surveillance, etc. In 2023, the number of devices connected to networks is expected to be 3 times higher than the global population [1]. With the IoT, sensors collect, communicate, analyze, and act on information. This offers new ways for technology, media and telecommunications businesses to create value. But it also creates new opportunities for that information to be compromised. The IoT connect systems, applications, data storage, and services become a new gateway for cyber-attacks as they continuously offer services but lack of adequate security protection. In 2020, nearly 1.5 billion cyber-attacks on IoT devices were reported [1]. These attacks may steal important and sensitive information that causes economic and societal damages. To address critical challenges related to the authentication and secure communication of IoT, many people (such as Jarosz et al.[2]) have developed various authentication and key exchange protocols for IoT devices. But software piracy and malware attacks remain high risks to compromise the security of IoT. This brings with it a particular challenge: securing IoT based systems against cyber-attacks.

In the FedCSIS 2023 challenge: Cybersecurity Threat Detection in the Behavior of IoT Devices [3], participants are asked to construct scoring models to detect whether anomalous operating systems were under attack by using logs from IoT devices. This competition has important theoretical and practical value for increasing IoT cyber security. It provides rich and detailed data for participants to analyze cyber-attacks from various perspectives and to train and test their models. Thereby we can understand attacker's intent, learn their behavior, and track the tactics, techniques, and procedures that they utilize to achieve their goals. We believe that all predictive models thoughtfully and elaborately constructed by each participant will definitely help to detect attacks as early as possible, determine the scope of the compromise rapidly and predict how they will progress, and eventually empower organizations to better respond to attacks.

In the past decade, traditional machine learning techniques (such as Support Vector Machine, Decision Tree, K-Nearest Neighbor, Random Forests, Naive Bayes, etc.) have been widely used by the cyber security community to automatically identify IoT attacks. Many papers (such as [4]) have provided various reference implementation on state-of-the-art machine learning methods for data preprocessing, feature engineering, model fitting, and ensemble blending. And paper [5] discusses in detail the existing machine learning and deep learning solutions for addressing different security problems in IoT networks. However, with the continuous expansion and evolution of IoT applications, attacks on these IoT applications continue to grow rapidly.

The complexity and quantity of attacks push for more efficient detection methods. In the recent years, deep learning techniques have been used in an attempt to build more reliable systems. For example, Martin Kodys et al. proposed a novel solution which deployed two CNN architectures (ResNet-50 and EfficientNet-B0) on the same data to observe how their performance differs to detect the intrusion attacks against IoT devices [6]. Kumar Saurabh et al. developed Network Intrusion Detection System (NIDS) models based on variants of LSTMs (namely, stacked LSTM and bidirectional LSTM and validated their performance) [7]. Compared with traditional machine learning, the deep learning brings an end-to-end approach combining feature selection and classification which can speed up the defense response against the fast-evolving cyber-attacks. However, some authors declare that deep learning methods have proved far better than the traditional machine learning models in terms of accuracy, precision with the ability to handle

large amounts of data, and the inability to scale the data poses a large limitation to the extensive use of any conventional machine learning model [7]. This is not always the case.

In this competition, we apply basic data processing approaches, and leverage the feature selection and model building methods mentioned in our ICME2023 paper [8], combined with fundamental knowledge of mathematical statistics for cyber security threat detection. Our methods are fast and accurate, and achieve near-perfect prediction results. Our work provides examples for processing large scale data and extracting effective features to get better detection accuracy with less computational cost.

The paper is structured as follows: Section II introduces data analysis and processing methods. Section III applies basic knowledge of mathematical statistics to construct rules for attack detection. Section IV discusses how to perform feature selection and build binary classification models for threat prediction. Section V explains the experiment design and presents the results of the experiments. Section VI discusses the pros and cons of our proposed approaches and suggests future research directions.

II. Data Analysis and Processing

The available training data and test data in this competition contain 15027 and 5017 log files respectively. Each log file includes 40 fields and it contains 1-minute logs of all related system calls. There are a total of 28,339,158 and 10,060,209 lines of records in the training and test sets, respectively. The size of the data set is over 21.4 gigabytes. Therefore, one of the main tasks of this competition is to analyze and process these data efficiently and thereby construct effective features for attack detection.

In the training set, 522 files were identified as being under attack. Therefore, the chance of cyber-attack is 3.47375%. After the end of the competition, the organizer published the labels of the test set for the participants to do further research. There are 176 files which are under attack in the test set. It seems that the data set is divided by a "stratified K-Fold" manner to let the test set has the same proportion of target variable as the entire data set.

Among the 40 fields, 17 fields only have unique values (for example, 'SYSCALL_arch' always equals to 'aarch64'). These fields are useless. Besides, (1) 'SYSCALL_timestamp' indicates the number of milliseconds relating to the datetime 2023-04-12-00:00:00. It should not be used for prediction, otherwise it will cause over-fitting. (2) Column 'SYSCALL_exit' and 'SYSCALL_exit_hint' take almost similar values, so column 'SYSCALL_exit_hint' can be ignored.

For the remaining columns, we split their values into the smallest units. We call these smallest units "basic items". It is obvious that every unique combination of basic items represents a new kind of attribute, which can dramatically increase the amount of data. These will cause the high cardinality problem which means that there can be many possible values for a single column. To solve the high cardinality problem, a simple and straightforward approach is to subdivide the content of each column, and left the problem of digging the correlations between these basic items to the subsequent algorithms. Take the following line of record as an example:

53824,aarch64,openat,yes,6.0,ps,/usr/bin/ps,>systemd>/usr/sbin/cron>/usr/sbin/cron>/bin/sh>/usr/bin/bash,"['/proc/uptime', '/proc/meminfo', '/proc', '/proc/647524/status']", "['/lib/aarch64-linux-gnu/libprocps.so.8', '/lib/aarch64-linux-gnu/libdl.so.2', '/lib/aarch64-linux-gnu/libc.so.6', '/lib/aarch64-linux-gnu/libsystemd.so.0', '/lib/aarch64-linux-gnu/librt.so.1', '/lib/aarch64-linux-gnu/liblzma.so.5', '/lib/aarch64-linux-gnu/libzstd.so.1', '/lib/aarch64-linux-gnu/liblz4.so.1', '/lib/aarch64-linux-gnu/libgcrypt.so.20', '/lib/aarch64-linux-gnu/libpthread.so.0', '/lib/aarch64-linux-gnu/libgpg-error.so.0', '/lib/aarch64-linux-gnu/libnss_files.so.2']",root,root,6,649165,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,[],,,,,/usr/bin/ps,,

We split the content of field 'PROCESS_PATH' (column 7) into the following small items:

systemd
/usr/sbin/cron
/bin/sh
/usr/bin/bash

and 'CUSTOM_openFiles' (column 8) into these items:

/proc/uptime
/proc/meminfo
/proc
/proc/647524/status

and 'CUSTOM_libs' (column 9) into these items:

/lib/aarch64-linux-gnu/libprocps.so.8
/lib/aarch64-linux-gnu/libdl.so.2
/lib/aarch64-linux-gnu/libc.so.6
/lib/aarch64-linux-gnu/libsystemd.so.0
/lib/aarch64-linux-gnu/librt.so.1
/lib/aarch64-linux-gnu/liblzma.so.5
/lib/aarch64-linux-gnu/libzstd.so.1
/lib/aarch64-linux-gnu/liblz4.so.1
/lib/aarch64-linux-gnu/libgcrypt.so.20
/lib/aarch64-linux-gnu/libpthread.so.0
/lib/aarch64-linux-gnu/libgpg-error.so.0
/lib/aarch64-linux-gnu/libnss_files.so.2

As for the columns 'SYSCALL_exit' (column 4) and 'SYSCALL_pid' (column 13), since 'exit code' and 'pid' take a very large range of values, we use the simple method of data binning to combine every 100 values into one group. After this, we count the occurrence of each basic item in the training log files and in files which are under attack, respectively. Examples of the obtained statistics are shown in Table 1.

Table 1. Example of statistical analysis results of column 2 ('SYSCALL_syscall')

Under attack times	Total occurrence	Items
491	14199	write
132	3521	exit
266	8123	bind
517	14942	exit_group
514	14701	socket
515	14939	execve
0	19	kill
520	14977	openat
501	14406	connect
522	15026	close
517	14956	clone
517	14962	mmap
0	51	listen
519	14963	munmap
0	49	bpf

These results can be further applied to construct detection rules and create features for classification.

III. Building Rules for Attack Detection

List all the basic items with attacked chance equal to 100% and number of occurrences >= 5 and are not included in other items, we can get the following list:

Table 2. Rules used for attack detection

Column index	Occurrence in training set	Occurrence in test set	Items
8	169	18	/proc/647524/stat
8	145	50	/proc/573203/stat
8	106	45	/proc/671015/stat
8	59	54	/proc/600849/stat
13	6	1	[576000000-576099999]
13	5	1	[574500000-574599999]

From the list we can find that basic item "/proc/647524/stat" appears 169 times in different log files, and all these files are identified as being under attack. From this we can infer that if a log file contains "/proc/647524/stat", it definitely indicates a cyber-attack has occurred.

Suppose "/proc/647524/stat" is an ordinary event, then the probability that "/proc/647524/stat" consecutively occurs 169 times in and only in the attacked files is 0.0347^169 = 0. According to the impossibility principle of small probability events, a small probability event is practically impossible to happen in a single trial. And once it does happen, we can reasonably reject the null hypothesis. In fact, it only needs five consecutive occurrences, then we can reasonably infer that an event has close relationship with cyber-attack.

By applying the above 6 rules, we are able to accurately detect 169 compromised files from the test set.

Furthermore, using the same method, we can confirm that 4003 files are secure (i.e., there are no attack events in these log files).

Applying these simple rules for threat prediction yields an AUC = 0.9985 on the test set.

IV. Feature Selection and Model Building

The aforementioned rule-based intrusion detection methods use only a small fraction of the data and cannot take advantage of the complex nonlinear relationships between various features. In this section we apply the sequential floating forward and backward (SFFB) feature selection method [8] for feature selection, and train a binary classification model based on GBM for attack prediction.

When creating features, we use the target encoding method to replace the categorical values with the mean of the target variable, and introduce a smoothing parameter to regularize towards the unconditional mean. We found this to be helpful in improving the predictive performance of the subsequent algorithms. We also find that the "K-fold target encoding" preferred by many people cannot mitigate over fitting risks. In fact, for high cardinality features "K-fold target encoding" method will lead to serious data leakage. This can be easily verified.

After feature encoding, we calculate the maximum, minimum and average chance of being attacked of each field. We also count their number of the contained basic items. Subsequently, these features are concatenated to form a feature set of equal length. We then use SFFB method to select features. The optimal subsets selected by the SFFB method are somewhat random. In most cases, the selected subset will only contain 10 features, such as:

PROCESS_comm_count, PROCESS_exe_count, PROCESS_PATH_mean, CUSTOM_openFiles_max, CUSTOM_openFiles_min, SYSCALL_pid_min, SYSCALL_pid_mean, SYSCALL_pid_count, PROCESS_name_mean, PROCESS_name_count.

*_max, *_min and *_mean means the maximum, minimum and average attacked chance of the fields. *_count means the number of basic items of the fields.

Training a GBM model with these 10-dimensional features leads to a classification result of AUC=0.9997 on the test set. Figure. 1 shows the gain contribution of these features.

V. Experiment Design and Experiment Results

Cybersecurity threat detection always is a majority-minority classification problem. Class imbalance in the dataset can dramatically skew the performance of classifiers. Therefore a reliable cross-validation method is essential to train a good classifier.

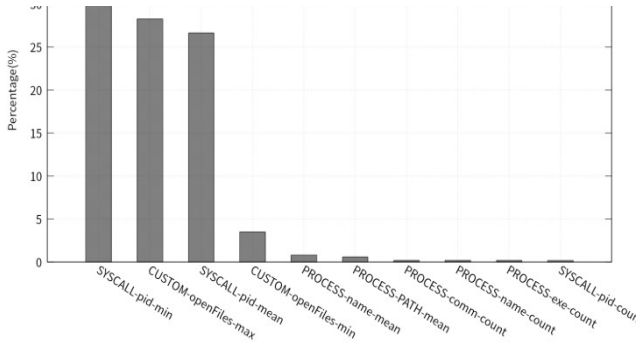

Fig. 1. Gain contribution of the 10 selected features

In our experiments, we estimate the performance of the classifier by using 3-fold cross-validation. At each fold, we completely hide the validation set when processing data and performing feature engineering. The average AUC score of 3-fold cross-validation is 0.9997 in local test. However, the classifiers trained in this way cannot achieve optimal scores on the public leaderboard. In fact, when the score of local CV is greater than 0.998, the changing trends of the local CV score is not consistent with the trends of the public leader-board. To address this problem, we randomly select 2/3 of the data from the training set at a time to train several classifiers, and then weighted averaging the prediction result of each classifier. In this way, we try to eliminate the effects of class imbalance and sample bias.

Finally, we ensemble the results obtained from the rules prediction with those predicted by the GBM model, and achieve an AUC score 0.9999 on the private leaderboard. After the organizer published the labels of the test set, we found that by correctly ensembling the prediction results from sections 3 and 4, we could obtain an AUC score 0.99995 on the test set. This is equivalent to the total accuracy can up to 99.88%. The ensemble method is:

1. If rule-based prediction results are equal to 1, then:

*ensemble results = 0.85 + 0.15*GBM prediction results.*

2. If rule-based prediction results are equal to 0, then:

*ensemble results = 0.15*GBM prediction results.*

3. Otherwise,

ensemble results = GBM prediction results

The total time (includes data processing, feature construction, feature selection, classifier training, and target prediction) required to obtain this result on our i7-10700 desktop is less than 30 minutes.

VI. Conclusion and Future Work

In this cyber security threat detection challenge, we only apply the fundamental methods of machine learning, but achieve near-perfect detection results. Many big-data competition participants like to apply ready-to-use GBM or deep learning frameworks. They prefer the end-to-end approaches that automates data processing, feature selection and classification, and expect to get good answers just by tuning the parameters. But our experiments show that each algorithm has a different application scenario.

In this competition, we conduct in-depth, detailed analysis of the massive-volumes data, and propose concise and efficient methods to process these data. (A significant portion of our work is C++ programming. To master the methodologies and techniques of contemporary C++ in the age of new technologies and challenges, one can start by reading paper [9].) Our proposed approaches are useful for solving variable-length, high-dimensional and high-cardinality problems.

However, our detection method still has obvious limitations: it is good at detecting known attacks but may fail at detecting attacks which have not been seen before. As more and more IoT devices are added, the potential for new and unknown threats grows exponentially. For this reason, an intelligent security framework for IoT networks must be developed that can identify such threats (e.g., detect any anomaly which rises from any deviation from normal behavior of the IoT network, or monitor network traffic to identify potential threats). In these research directions, conventional machine learning methods will still play an important role.

Acknowledgements

Many thanks to Łukasiewicz Research Network, EFIGO, and QED Software for their great efforts in organizing this challenging competition. Thanks to all the participants. They all did an amazing job.

References

[1] IoT Cybersecurity in 2023: Importance & Tips To Deal With Attacks. https://research.aimultiple.com/iot-cybersecurity/

[2] Michał Jarosz, Konrad Wrona, Zbigniew Zieliński. Formal verification of security properties of the Lightweight Authentication and Key Exchange Protocol for Federated IoT devices. Proceedings of the 17th Conference on Computer Science and Intelligence Systems, ACSIS, Vol. 30, pages 617-625 (2022). DOI: http://dx.doi.org/10.15439/2022F169.

[3] FedCSIS 2023 Challenge: Cybersecurity Threat Detection in the Behavior of IoT Devices. https://knowledgepit.ai/fedcsis-2023-challenge/

[4] Eyad Kannout, Michał Grodzki, Marek Grzegorowski. Considering various aspects of models' quality in the ML pipeline - application in the logistics sector. Proceedings of the 17th Conference on Computer Science and Intelligence Systems. ACSIS, Vol. 30, pages 403-412 (2022). DOI: http://dx.doi.org/10.15439/2022F296.

[5] F. Hussain, R. Hussain, S. A. Hassan and E. Hossain. Machine Learning in IoT Security: Current Solutions and Future Challenges. in IEEE Communications Surveys & Tutorials, vol.22, no.3, pp.1686-1721, 2020. DOI: https://doi.org/10.1109/COMST.2020.2986444.

[6] Martin Kodys, Zhi Lu, Kar Wai Fok, et al. Intrusion Detection in Internet of Things using Convolutional Neural Network. https://arxiv.org/pdf/2211.10062.pdf. DOI: https://doi.org/10.1109/PST52912.2021.9647828.

[7] Kumar Saurabh, Saksham Sood, P. Aditya Kumar, et al. LBDMIDS: LSTM Based Deep Learning Model for Intrusion Detection Systems for IoT Networks. https://arxiv.org/pdf/2207.00424.pdf. DOI: https://doi.org/10.48550/arXiv.2207.00424

[8] Chang Lin. Predicting Frags in Tactic Games using Machine Learning Techniques and Intuitive Knowledge (in press). In Proceedings of the 2023 IEEE International Conference on Multimedia and Expo. IEEE, 2023.

[9] Bogusław Cyganek. Modern C++ in the era of new technologies and challenges - why and how to teach modern C++?. Proceedings of the 17th Conference on Computer Science and Intelligence Systems. ACSIS, Vol. 30, pages 35-40 (2022). DOI: http://dx.doi.org/10.15439/2022F308.