JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 2, 2022, p. 3638-3647
https://publishoa.com
ISSN: 1309-3452

A Malware Detection Method for Health Sensor Data Based on Machine Learning and Genetic Algorithm

P Devabalan, Prof. CSE:BVCE, Email:devabalanme@gmail.com

Chandra Mouli VSA, Prof. CSE:BVCE. Email: mouliac@yahoo.co.in

Anumala Sai Kumar, CSE:BVCE

Appana Pavan Kumar, CSE:BVCE

Ashok Kumar Kusumanchi, CSE:BVCE

E Vamsi Trivikram, CSE:BVCE

Received 2022 March 25; **Revised** 2022 April 28; **Accepted** 2022 May 15

Abstract:

Small modifications in the virus code are easily detected by conventional signature-based malware detection techniques. The majority of malware programmes today are modifications of other programmes. They therefore have various signatures yet share certain similar patterns. Instead than just noticing slight changes, it's important to recognise the virus pattern in order to protect sensor data. However, we suggest a quick detection strategy to find patterns in the code using machine learning-based approaches in order to quickly discover these health sensor data in malware programmes. To evaluate the code using health sensor data, XGBoost, LightGBM, and Random Forests will be specifically utilised. The codes are either supplied into them as single bytes or tokens or as sequences of bytes or tokens (e.g. 1-, 2-, 3-, or 4-grams). Terabytes of labelled programmes, both virus and benign ones, have been gathered. Choosing and obtaining the features, modifying the three models to train and test the dataset, which comprises of health sensor data, and evaluating the features and models are the challenges of this assignment. When a malware programme is discovered by one model, its pattern is broadcast to the other models, effectively thwarting the infiltration of the malware programme.

Keywords: Random Forests algorithm, LightGBM, and XGBoost.

I. INTRODUCTION

All kinds of sensors are being used to gather health sensor data as we enter the Internet of Things Era. Eventually, malicious software or programmes that are hidden in health sensor data and are regarded as intrusions in the target host computer are executed in accordance with a hacker's predetermined logic. Computer viruses, worms, Trojan horses, botnets, ransomware, and other types of malicious software are examples of data from health sensors that is malicious. Malware assaults can harm computer networks and systems while stealing sensitive data and core data. It poses one of the biggest risks to the security of computers today. categories of analysis

i) Static evaluation:

It is typically done by analysing each component and illustrating the many resources of a binary file without actually using it. A disassembler can also be used to disassemble (or redesign) binary files (such as IDA). Humans are able to read and comprehend assembly code, which can occasionally be converted from machine code. Malware analysts are able to decipher assembly instructions and visualise the program's intended behaviour. Some contemporary malware is developed utilising unclear methods to thwart this kind of examination, such introducing grammatical flaws in the code. Although these mistakes can be perplexing to the disassembler, they are nonetheless functional during execution.

ii) Dynamic analysis:

It involves analysing how the malware behaves when it is actually running on the host machine. Modern malware may employ a wide range of misleading strategies to evade dynamic analysis, such as testing active debuggers or virtual environments, delaying the execution of harmful payloads, or requesting interactive user input.

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 2, 2022, p. 3638-3647
https://publishoa.com
ISSN: 1309-3452
We primarily concentrated on static code analysis in this work. The primary feature matching or broad-spectrum signature scanning techniques used in early static code analysis. Broad-spectrum scanning examines the feature code and employs masked bytes to separate the sections that need to be compared from those that do not, while feature matching simply uses feature string matching to complete the detection. The hysteresis issue is critical since both approaches must get malware samples and extract features before they can be detected. In addition, when malware technology advances, the number of malware variants suddenly rises and malware starts to change during transmission in an effort to escape being detected and eliminated. It is challenging to extract a fragment of code to serve as a virus signature because the shape of the variations varies greatly.

1.1 Malware Samples Gathered:

The foundation for code analysis is the efficient acquisition of malware samples. The classification model can perform more accurate detection functions when integrated with machine learning techniques, but only after proper training using the sample data. Malware samples can be obtained in a variety of methods.

i) User-side sampling: The majority of anti-virus software companies use this as their primary technique. Antivirus software users that transmit malware samples to providers. This strategy performs well in real-time, but it is challenging to get the data directly because security providers frequently decide not to release their data in an open manner.

ii) Open network databases, such as Virus Bulletin, Open Malware, and VX Heavens, among others. The open online sample systems are currently constrained in comparison to the speed at which malicious code is updated, and the websites have issues such being subject to attacks. Therefore, the development of a malware sharing mechanism has demonstrated its significance more and more.

iii) Additional technological strategies: A particularly fragile system is created to entice attackers to attack in order for the system to get malware samples through collection utilising a capture tool like a honeypot (such as the Nepenthes honeypot). Additionally, some Trojans and Internet backdoors can be acquired via spam traps or security discussion forums. But the size of the capture sample using the aforementioned technological methods is quite little.

1.2 Motivation:

As there isn't a single paper that discusses the predictions made in this, the motivation behind this study is to determine how machine learning and boosting algorithms will aid in better malware detection and to understand how the combination of these models works in a better way than the existing one. To know and comprehend how these models might compare and contrast one another in terms of data prediction.

1.3 Problem Proposition:

We employ a gradient framework for high performance because the running speed is too sluggish and the performance is inadequate. Other issues include the need to repeatedly traverse the whole training set for each iteration. Each split node requires a split-gain calculation, which takes a long time as well.

Size of the project:

In order to train and test the dataset, which comprises of health sensor data, this work's scope is to choose and obtain the features and adjust the three models.

Review the specifications and models.

This may also apply to medical gadgets in intensive care units, hospital wards, doctor's offices, lab equipment, dental offices, and goods for in-home care. give an attacker remote access to a compromised machine, Send spam to gullible recipients from the compromised device, Investigate the local network of the affected user.

II. SUGGESTIVE SYSTEM

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 2, 2022, p. 3638-3647
https://publishoa.com
ISSN: 1309-3452

Malware detection essentially boils down to a classification issue that determines whether a sample is legitimate software or malicious software. Therefore, the key processes of a machine learning algorithm drive host malware detection technology, and the primary research steps of this study are as follows: Amass enough samples of both legitimate software and malicious code. Effectively process the sample's data, then extract the characteristics. Select the classification's primary features further. Create a classification model by combining the training data with machine learning methods. Utilizing the trained classification model, find unknown samples.

The models XGBoost, LightGBM, and Random Forest were used in this study. Prior to using these 3 models, we evaluated the SVM (Support Vector Machine), but the performance was insufficient and the running speed was too slow.

2.1 Methods:

2.1.1 Machine learning algorithms

The ability of machine learning (ML) algorithms to solve huge non-linear problems on their own while utilising information from many sources is one of their key advantages. In real-world situations, ML enables superior decision-making and informed action with no (or little) human involvement. To create a comparable malicious code classifier, the machine learning algorithm can be trained using the distinctive data that are gleaned from the static and dynamic analysis of the harmful code. Some of the ML models that were employed in this include:

SVM:

A supervised machine learning approach called Support Vector Machine (SVM) can be applied to problems involving classification and regression. However, classification issues are where it's most frequently employed. Finding a hyperplane in an N-dimensional space that clearly classifies the data points is the goal of the SVM method.

Simple Bayes:

In comparison to more complex algorithms, the Naive Bayes classifier can be extremely quick. Each class distribution can be individually assessed as a one-dimensional distribution thanks to the separation of the class distributions.

Given the goal value, it is assumed that each attribute value P(d1, d2, d3|h) is conditionally independent, and its values are computed as P(d1|h) * P(d2|H), and so on.

Analogous Regression:

As a classifier, logistic regression is used to group observations into distinct classes. The method uses the logistic sigmoid function to translate its output into a probability value and forecasts the goal using the idea of probability. Statistics experts created the logistic function, also known as the sigmoid function, to characterise the characteristics of population expansion in ecology, which rise swiftly and peak at the carrying capacity of the ecosystem. Any real-valued number can be transformed into a value between 0 and 1, but never precisely at those ranges, using this S-shaped curve.

$$1 / (e^{-value} + 1) \quad (1)$$

Where value is the actual numerical value you want to alter and e is the base of the natural logarithms (Euler's number or the EXP() function in your spreadsheet). The logistic function was used to translate the numbers between -5 and 5 into the range between 0 and 1. The results are plotted below.

Algorithm for Random Forests:

Three random principles are used in this model: selecting training data at random when creating trees, choosing specific subsets of features when splitting nodes, and only taking into account a small part of all characteristics when dividing each node in each simple decision tree. Each tree in a random forest learns from a random selection of the data points during training data.

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 2, 2022, p. 3638-3647
https://publishoa.com
ISSN: 1309-3452
Boosting an extremely gradient:

A regularising gradient boosting framework is offered by this open-source software package. Integrated cross-validation, regularisation to prevent overfitting, effective handling of missing data, catch awareness, tree pruning, and parallelized tree building are all features of this technique that are used in this model. Among XGBoost's key attributes are: Parallelization: Multiple CPU cores are used to train the model. Regularization: To prevent overfitting, XGBoost provides a variety of regularisation penalties. Non-linearity: XGBoost can identify non-linear data patterns and learn from them.

The following are XGBoost's drawbacks:

I Each iteration necessitates repeatedly navigating the whole training set.

ii) Each split node requires a split-gain calculation to be performed, which takes a lot of time.

LightGBM:

It is a model for boosting. It is a quick, distributed, high-performance gradient framework built on decision tree algorithms and is used for many different machine learning tasks, including classification and ranking. It is under the purview of Microsoft's DMTK project. It is used for classification, ranking, and other machine learning applications and is based on decision tree algorithms.

It divides the tree leaf-wise with the best fit since it is based on decision tree algorithms, as opposed to other boosting algorithms that divide the tree depth- or level-wise. As a result, in Light GBM, when growing on the same leaf, the leaf-wise method can reduce more loss than the level-wise strategy, which leads to significantly superior accuracy that can only be sometimes attained by any of the existing boosting algorithms.

Additionally, it moves remarkably quickly, hence the name "Light." **Algorithm and Process Design:**

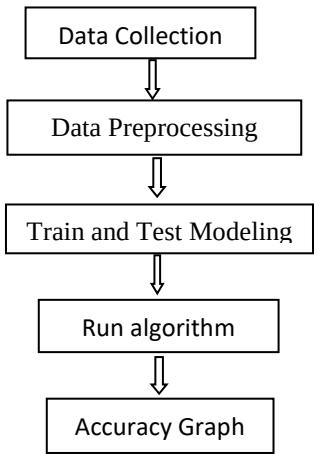

Fig.1 Process Design

III. **Data collection:**
We gathered enough legal software samples and malware code samples to create a health sensor dataset, which we then published.

Data preprocessing: We efficiently processed the sample's data to extract its features.

Data should be divided into train and test data for train and test modelling. The model will be trained using Train, and performance will be evaluated using Test data.

Run SVM, Navie Bayes, Random Forest, and XGboost algorithms. Create a classification model by combining the training data with machine learning methods.

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 2, 2022, p. 3638-3647
https://publishoa.com
ISSN: 1309-3452
Feature selection: Pick out the most important elements for categorization and eliminate the extraneous features.

Accuracy Graph: With the help of this module, we can view a graph of all the accuracy results.

IV. Execution and Results:

4.1 Details of the Data

We extract 27 subdivided features, such as the byte count (256d, where d represents dimensions), opcode 1-gram (150d), opcode 2-4-grams (150, 450, and 750d), segment (150, 450, and 750d), and dll (150, 450, and 750d), and run 81 experiments (we run each feature's libsvn code).

The malware sample used for the training set and test set is from Secure Age's malware sample from April 2017, respectively. The experiments consist of 4 sections:

testing each feature's and model's impact on this useful dataset.

comparing the performance of many models for a particular attribute.

determining which feature overall delivers the best performance.

determining which dimension, with relation to a particular attribute, produces the greatest outcomes. We evaluate whether opcode or daf characteristics are superior for 1-gram to 4-gram evolutionary trends. We also evaluate which kind of feature a particular model chooses to use.

Table.1

Class No Label Name	sample count
1 Worm	1541
2 Adware	2478
3 Backdoor	2942
4 Trojan	475
5 Backdoor	42
6 TrojanDownloader	751
7 Backdoor	398
8 obfuscated malware	1228
9 Backdoor	1013

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 2, 2022, p. 3638-3647
https://publishoa.com
ISSN: 1309-3452

4.2 Performance Metrics:

To evaluate them, three metrics are used.

I. Area Under the Curve (AUC)

The AUC indicates the likelihood that, given two randomly chosen samples, the classifier will properly assign the positive sample a higher score than the negative sample. The sorting ability of the model is stronger the higher the AUC value is.

II. Accuracy and Recall

We designate the number of positive samples in this dataset as P and the number of negative samples as N. (malware or legitimate software).

If a sample's prediction is positive and it turns out to be positive, like in the first scenario, we refer to it as a true positive (TP).

In the second scenario, we refer to it as a false positive if the prediction is positive but the actual value is negative (FP).

In the third scenario, a false negative occurs when the forecast is negative but the actual number is positive (FN).

A true negative (TN) is what happens in the last scenario when both the prediction and the actual value are negative.

There can only be one of these four situations for each sample. There isn't any other option. Then, we have the subsequent: Precision-P=TP/TP+FP,

N=TN+FP, P=TP+FN (2)

Precision-N=TN/TN+FN (3)

Recall-P=TP/P; Recall-N=TN/N; (4) (5)

Recall reflects the classification model's capacity to recognise P/N samples. The model's capacity to recognise P/N samples increases with recall. The precision represents the model's capacity to distinguish between N/P samples.

III. Precision:

It displays the classifier's overall accuracy, or the percentage of accurate predictions.

4.3 Result:

The Validation Summary Based on the tested outcomes of our proposed model, which performs better in malware detection for health data, AUC Curve, Precision, Recall, Accuracy metrics of machine learning models, including Random Forest, Naive Bayes, support vector machine, Logistic Regression, and Extreme Gradient Boosting, were employed as predictors.

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 2, 2022, p. 3638-3647
https://publishoa.com
ISSN: 1309-3452

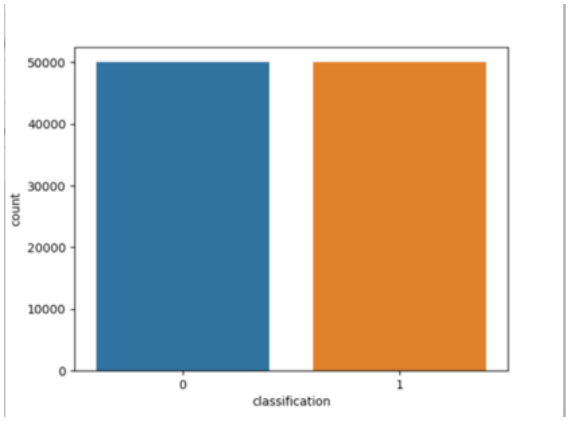

Fig.2.Classification vs count

```
Dataset loaded

Data Information:
                                                hash  millisecond classification  state  usage_counter  ...  utime  sti
me  gtime  cgtime  signal_nvcsw
0  42fb5e2ec009a05ff5143227297074f1e9c6c3ebb9c914...           0        malware      0              0  ...
 380690     4      0        0           0
1  42fb5e2ec009a05ff5143227297074f1e9c6c3ebb9c914...           1        malware      0              0  ...
 380690     4      0        0           0
2  42fb5e2ec009a05ff5143227297074f1e9c6c3ebb9c914...           2        malware      0              0  ...
 380690     4      0        0           0
3  42fb5e2ec009a05ff5143227297074f1e9c6c3ebb9c914...           3        malware      0              0  ...
 380690     4      0        0           0
4  42fb5e2ec009a05ff5143227297074f1e9c6c3ebb9c914...           4        malware      0              0  ...
 380690     4      0        0           0

[5 rows x 35 columns]
Columns Information: Index(['hash', 'millisecond', 'classification', 'state', 'usage_counter',
       'prio', 'static_prio', 'normal_prio', 'policy', 'vm_pgoff',
       'vm_truncate_count', 'task_size', 'cached_hole_size', 'free_area_cache',
       'mm_users', 'map_count', 'hiwater_rss', 'total_vm', 'shared_vm',
       'exec_vm', 'reserved_vm', 'nr_ptes', 'end_data', 'last_interval',
       'nvcsw', 'nivcsw', 'min_flt', 'maj_flt', 'fs_excl_counter', 'lock',
       'utime', 'stime', 'gtime', 'cgtime', 'signal_nvcsw'],
      dtype='object')
Shape of DataSet: (100000, 35)
```

Fig-3 Basic Preprocessing

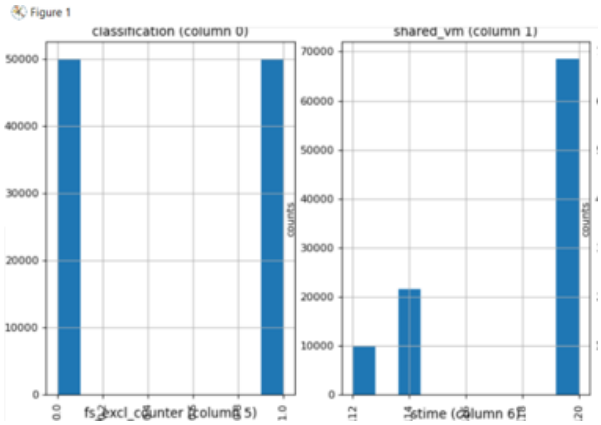

Fig-4 split the data into train and test

Train data will be used for training and to test the performance we are using test data.

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 2, 2022, p. 3638-3647
https://publishoa.com
ISSN: 1309-3452

```
dtype: int64
Output Variables Information:
1    50000
0    50000
Name: classification, dtype: int64
Logistic Accuracy: 0.49833333333333335
Logistic recall_score: 1.0
Logistic precision_score: 0.49833333333333335
Logistic f1_score: 0.6651835372636263
SVC Accuracy: 0.49946666666666667
SVC recall_score: 1.0
SVC precision_score: 0.49889875191884137
SVC f1_score: 0.6656870602903197
GaussianNB Accuracy: 0.6274
GaussianNB recall_score: 0.8965886287625418
GaussianNB precision_score: 0.581871852752214
GaussianNB f1_score: 0.7057336913599748
RandomForest Accuracy: 1.0
RandomForest recall_score: 1.0
RandomForest precision_score: 1.0
RandomForest f1_score: 1.0
XGB Accuracy: 1.0
XGB recall_score: 1.0
XGB precision_score: 1.0
XGB f1_score: 1.0
LGBM Accuracy: 1.0
LGBM recall_score: 1.0
LGBM precision_score: 1.0
LGBM f1_score: 1.0
```

Fig-5: Mentioned algorithms will be run on the data

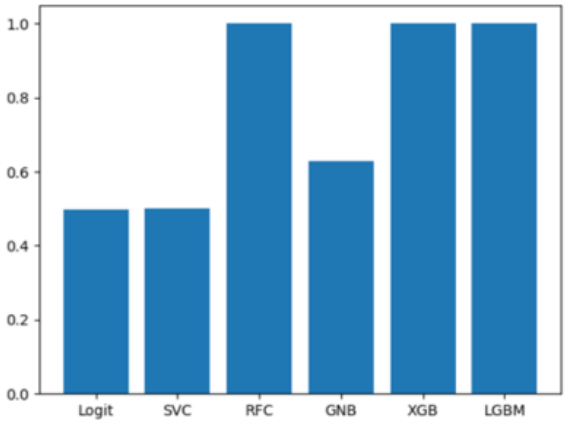

Fig.6: Accuracy Comparison for all the models

Generic Optimization Algorithm

In this study, we used a variety of algorithms to identify malware from health sensor data, but we didn't use any feature interpretation or selection algorithms that explain which crucial features contribute to greater accuracy. Similarly, in the proposed study, many algorithms gave a 100% accuracy rate, but we didn't know which features were most important to achieve that level of accuracy. The features with the highest fitness will be chosen and taken into consideration as crucial characteristics in order to get the highest accuracy, which is why we are using Genetic Method in extension, which will identify fitness of each feature by utilising Logistic Regression algorithm.

There are 35 features or columns in the dataset, and the genetic algorithm will only select those characteristics that have high fitness values. In the paper, the author also states that, as an extension, he will interpret or identify the traits that are most helpful in reaching high accuracy. For reference, see below. from the paper

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 2, 2022, p. 3638-3647
https://publishoa.com
ISSN: 1309-3452

We can see the names of the columns and features in the previous page, and we can see that the dataset has a total of 35 columns. Since we don't know which column contributes the most, we can find out by utilising the extension idea, and we can then execute each button individually.

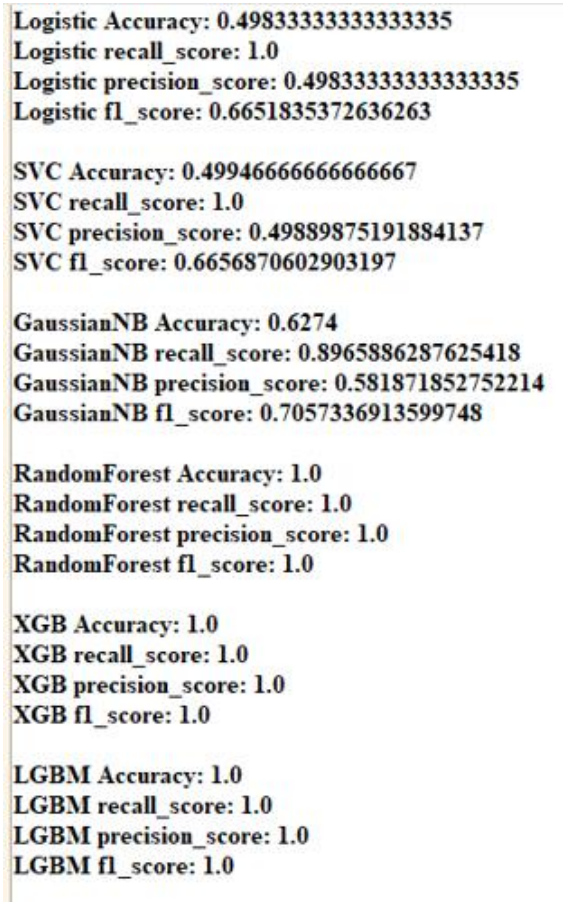

Fig.7 Accuracy comparison

In above screen we can see most of algorithms gave 100% accuracy and which columns/features are contributing most we don't know so by clicking on 'Extension Genetic Algorithm Features' button we can know the names of most important features

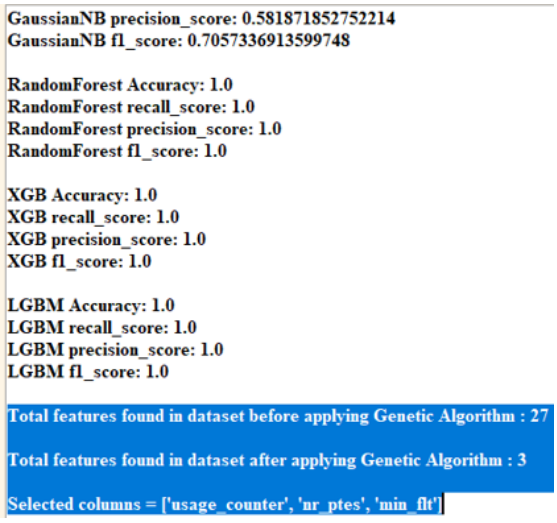

Fig.8 Genetic algorithm operations

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 2, 2022, p. 3638-3647
https://publishoa.com
ISSN: 1309-3452
Out of 35 rows, we can observe in the diagram above. Using a genetic algorithm, 27 columns of data are analysed, and the most important attributes are then selected as three. Using this extension notion, we may identify which dataset columns are the most important for achieving high accuracy.

CONCLUSION

The use of machine learning techniques in the identification of dangerous code in health sensor data has been increasingly recognised by the academic community and various security vendors as the complexity of malware programmes increases. Combining several models and discussing static code analysis based on various machine learning algorithms and characteristics is the focus of this study. Malware detection technology for machine learning could benefit from this work's reference value. This sector, however, is still in its infancy.

REFERENCES

1. M. Shen, B. Ma, L. Zhu, R. Mijumbi, X. Du, and J. Hu, "Cloud-Based Approximate Constrained Shortest Distance Queries over Encrypted Graphs with Privacy Protection", IEEE Transactions on Information Forensics & Security, Volume: 13, Issue: 4, Page(s): 940 – 953, April 2018, DOI: 10.1109/TIFS.2017.2774451.
2. P. Dong, X. Du, H. Zhang, and T. Xu, "A Detection Method for a Novel DDoS Attack against SDN Controllers by Vast New Low-Traffic Flows," in Proc. of the IEEE ICC 2016, Kuala Lumpur, Malaysia, 2016.
3. Z. Tian, Y. Cui, L. An, S. Su, X. Yin, L. Yin and X. Cui. A Real-Time Correlation of Host-Level Events in Cyber Range Service for Smart Campus. IEEE Access. vol. 6, pp. 35355-35364, 2018. DOI: 10.1109/ACCESS.2018.2846590.
4. Q. Tan, Y. Gao, J. Shi, X. Wang, B. Fang, and Z. Tian. Towards a Comprehensive Insight into the Eclipse Attacks of Tor Hidden Services. IEEE Internet of Things Journal. 2018. DOI: 10.1109/JIOT.2018.2846624.
5. Z. Wang, C. Liu, J. Qiu, Z. Tian, C., Y. Dong, S. Su Automatically Traceback RDP-based Targeted Ransomware Attacks. Wireless Communications and Mobile Computing. 2018. https://doi.org/10.1155/2018/7943586.
6. L. Xiao, Y. Li, X. Huang, X. Du, "Cloud-based Malware Detection Game for Mobile Devices with Offloading", IEEE Transactions on Mobile Computing, Volume: 16, Issue: 10, Pages: 2742 – 2750, Oct. 2017. DOI: 10.1109/TMC.2017.2687918.
7. L. Xiao, X. Wan, C. Dai, X. Du, X. Chen, M. Guizani, "Security in mobile edge caching with reinforcement learning", IEEE Wireless Communications Volume: 25, Issue: 3, pp. 116-122, June 2018, DOI: 10.1109/MWC.2018.1700291.
8. Y. Wang, Z. Tian, H. Zhang, S. Su and W. Shi. A Privacy Preserving Scheme for Nearest Neighbor Query. Sensors. 2018; 18(8):2440. https://doi.org/10.3390/s18082440.
9. ABOU-ASSALEH T , CERCONE N , KESELJ V ,et al. N-gram-based detection of new malicious code[C] The 28th Annual Int. Computer Software and Applications Conference (COMPSAC). 2004: 41-42.
10. Henchiri O, Japkowicz N. A feature selection and evaluation scheme for computer virus detection[C] Data Mining, 2006. ICDM'06. Sixth International Conference on. Hong Kong, Chian IEEE, 2006: 891-895.
11. Y. Ding , X. Yuan , K. Tang, et al. A fast malware detection algorithm based on objective-oriented association mining[J]. Computers & Security, 2013,39: 315-324.