DS Journal of Cyber Security
Volume 1 Issue 1, 28-38, Jul - Sep 2023
ISSN: 2584-0665 / https://doi.org/10.59232/CYS-V1I1P103

Original Article

STAR-D: Multiclass SVM-Based Smart TV Attack Ransomware Detection via DLL/API File Features

M. Thangamani

Department of Information Technology, Kongu Engineering College, Tamilnadu, India.

manithangamani2@gmail.com

Received: 25 May 2023; Revised: 6 June 2023; Accepted: 20 June 2023; Published: 6 July 2023;

Abstract - Recent ransomware attacks have been expensive due to the tremendous harm and disruption they caused in various ways, including health, industries, business, education and insurance. The idea that a backup can guard against a hacker stealing an organisation's digital data has been dispelled by recent ransomware outbreaks like WannaCry and Not Petya. Numerous malware detection techniques have been put forth to identify various virus families. However, the issue has not yet been resolved because malware is always developing. In this study, a unique Smart TV Attack Ransomware Detection (STAR-D) method based on Multiclass SVM and DLL/ Application Programming Interface (API) file features has been proposed. In this framework, machine learning is used to examine the ransomware at many levels, including its Dynamic Link Library (DLL) and APIs. Term Frequency and Inverse Document Frequency (TFIDF) were used to further process the raw data from the malware and smart TV to produce the final feature sets. Finally, a multiclass SVM is provided with these attributes as inputs to classify the assault. Although a general-purpose computer can potentially be used, it employs the Apache Spark computing environment for speedier processing. For evaluating the efficacy of the suggested model, the accuracy, specificity, parameters sensitivity, precision and F1 score are taken into account. The suggested approach outperforms Naive Bayes, Random Forest, and Decision Tree in terms of overall accuracy by 0.82%, 1.32%, and 3.57%, respectively.

Keywords - Ransomware detection, Reverse engineering, Machine Learning, Dynamic Link Library, Application Programming Interface (API).

1. Introduction

Public displays have typically been static, impersonal devices that indifferently transmit information to a large audience on smart TVs. However, modern approaches demand a new generation of ubiquitous public displays, which would be a new generation of general public displays, which would be able to offer users engagement possibilities in addition to customised content [1]. Although many options have been considered to upgrade public displays, the smart TV project seems the most exciting. Smart TVs and set-top boxes are a new generation of devices that have better internet integration and more processing capability. Attacks with ransomware are on the rise right now. Particularly in the areas of insurance, health, research, business, and education, numerous government and non-government organisations have been impacted.

Though smart TVs are aimed to make your viewing experience smarter, an ugly incident occurred on christmas day to show the flip side. It is an LG Smart TV model with a tweaked android version that has been

found to be caught by a ransomware attack. The notorious ransomware reportedly reached the Android-powered TV through a multimedia app. It is worth noting that the infected TV was based on the Google TV platform that was completely discontinued way before the official release of Android TV. Likewise, LG also moved apart and is currently offering its WebOS over Google TV or Android TV platform [2]. This is not the first time when ransomware hit smart TVs. In November 2015, a Symantec researcher posted a blog post highlighting how ransomware can easily hit smart TVs. Security company Trend Micro reported in june that ransomware regularly attacks smart TVs, and the most active and common one is cyberplace. An example of a ransomware attack on smart TV is depicted in Figure 1.

Fig. 1 Ransomware attack on smart TV

Malware, known as ransomware, prevents victims from accessing their resources until a ransom is paid [3]. The Internet of Things (IoT), a growing collection of embedded gadgets with internet access, has recently drawn the attention of malware writers. IoT ransomware is still not widely used by attackers, who prefer instead to deploy malware on affected devices to mine cryptocurrencies or launch denial-of-service assaults. Smart TVs, cameras, wearable technology, and automobiles are a few examples of IoT gadgets. While it is possible to claim that any device with the ability to connect to the internet qualifies as an "IoT device."

The two main categories of ransomware are Crypto ransomware and Locker Ransomware [4]. An infected computer system's files are encrypted by crypto-ransomware until a bitcoin payment is made, which holds them prisoner. The infected person's identity is concealed when payments are made using bitcoin. The majority of the

time, users may unlock encrypted files and get the original, readable data back by paying a ransom. As opposed to this, locker ransomware just encrypts the data. Users could access the secured data by shifting the hard drive physically to a safe place or system.

In general, there has been a lot of research done on smart TV file security, but as far as we are aware, none of it has particularly addressed modern threats and challenges, including ransomware. Additionally, a number of additional significant security-related factors that are covered in the current study have not been disclosed [5]. In ransomware research, dynamic analysis is utilised for a range of tasks, such as running an executable file within a virtual box or else a sandbox environment. Some ransomware samples do not function properly in virtual environments and do not exhibit their true characteristics. These restrictions can render ransomware analysis and detection useless. To solve these problems, a novel DLL/API file features-based Multiclass SVM-based STAR-D Smart TV assault ransomware detection approach has been developed.

The framework uses machine learning methods to examine the ransomware at several levels, such as DLL and API files. The original smart TV data was obtained using a Linux machine, and the malware WannaCry and NotPetya were further processed using our specially developed TF-IDF processing. The results of experiments with various TF-IDF language model parameters demonstrate the accuracy of ransomware detection. The experimental test samples were taken live input files from a smart TV. The extracted features are classified using the Multiclass SVM method for numerical representation and decision-making. The results obtained from Multiclass SVM are more accurate and of higher quality than those obtained from traditional classifiers.

The remainder of this study will be demonstrated as follows. The literature review is described in Section 2. The proposed approach, an explanation, and the associated algorithm are displayed in Section 3. The performance outcome and its analysis are provided in Section 4. The conclusion and future scope were included in Section 5.

2. Literature Survey

In 2017, Tran, T.K. and Sato, H. [6] This study proposed a method for classifying malware that uses API call sequences as classifier inputs. For the sake of categorising malware, the three suggested methods-TF-IDF, paragraph vector with distributed memory, and paragraph vector with distributed bag of words-are categorised into three different groups. Each of them provides us with extremely accurate information.

In 2017 Hasan, M.M. and Rahman, M.M. [7] proposed that a machine learning-based approach will include integrated techniques from static and dynamic analysis to discover ransomware. The experimental test samples were created with almost every ransomware family, including the most recent "WannaCry." The results also suggest that combination analysis, rather than solo analysis, is more effective at detecting ransomware.

In 2022, Zahoora, U. et al. [8] proposed for the purpose of detecting ransomware attacks, propose a newly developed Deep Contractive Autoencoder (DCA) oriented attribute learning approach and an IS strategy based on heterogeneous voting ensemble. Furthermore, using these essential features considerably enhanced attack detection (recall=0.95) and reduced False Negatives (FN=6).

In 2021, Poudyal S. et al. [9] proposed a framework for reverse engineering that effectively detects ransomware by combining feature generation tools and Machine Learning (ML). According to experimental results, the effectiveness of ransomware sample detection ranged from 76% to 97%, depending on the machine learning method utilised.

In 2020, Khan, F. et al. [10] DNA act-Ran was suggested as a ransomware detection technique. The ML algorithm was successfully used to detect ransomware. The proposed DNA act-Ran method's efficacy and efficiency were verified using the real-time dataset. The test provides some indication that active learning classifiers are more effective at detecting ransomware.

3. Proposed STAR-D Methodology

This section proposes a novel STAR-D Smart TV attack ransomware detection via DLL/API file features based on multiclass SVM. Figure 2 illustrates the training and testing phases of the detector engine's operation. The framework uses machine learning methods to examine ransomware at several levels, such as Dynamic Link Library (DLL) and API file. The original smart TV data was obtained using a Linux machine, and the malware WannaCry and NotPetya were further processed using our specially developed TF-IDF processing. Figure 2 depicts the overall structure of the proposed technique.

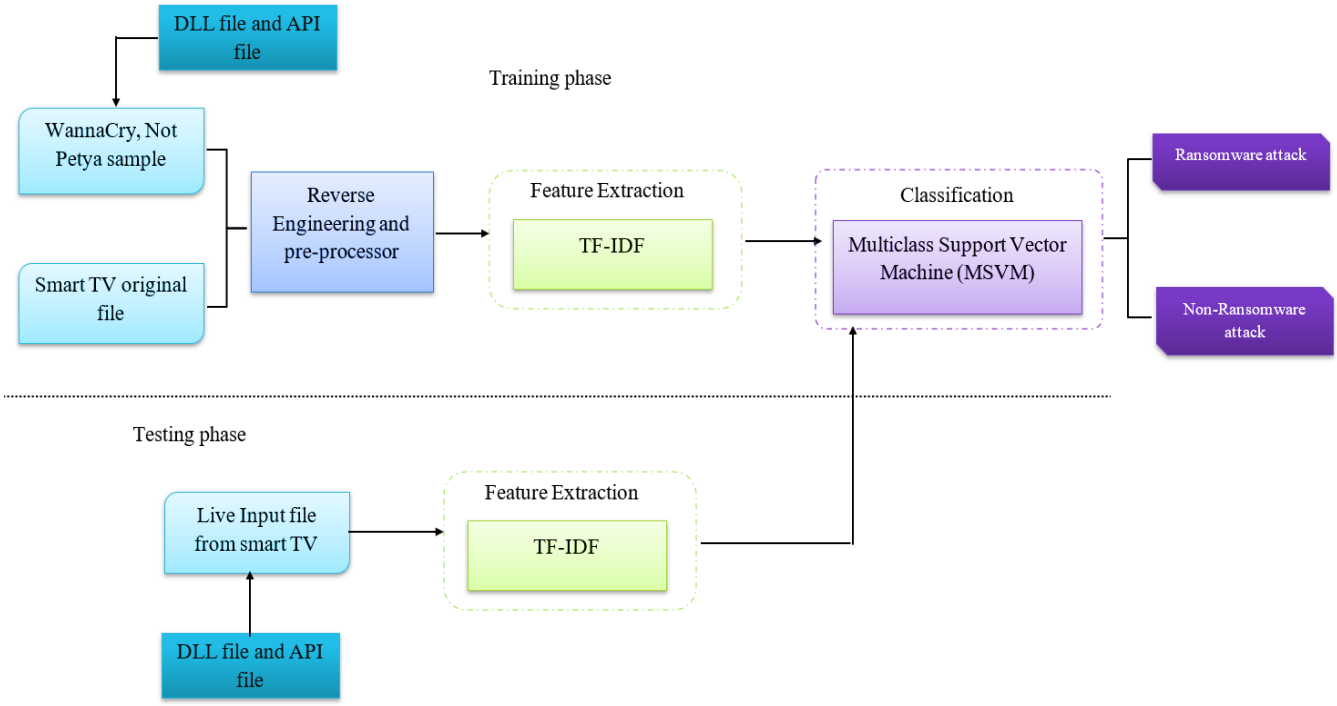


Fig. 2 Overall block of proposed STAR-D methodology

3.1. API and DLL Description

Using the detection engine, the DLL file assesses the DLLs in a given binary and determines the accuracy of categorisation. If the accuracy is higher than or equal to the predetermined threshold value, the detection counter is increased by one. The security team or an expert user sets the threshold value. In our trial, 80% was the threshold. A modest collection of classes and interfaces was created to simplify communication handling for outside developers. A class diagram for the DLL and API is shown in Figure 3.

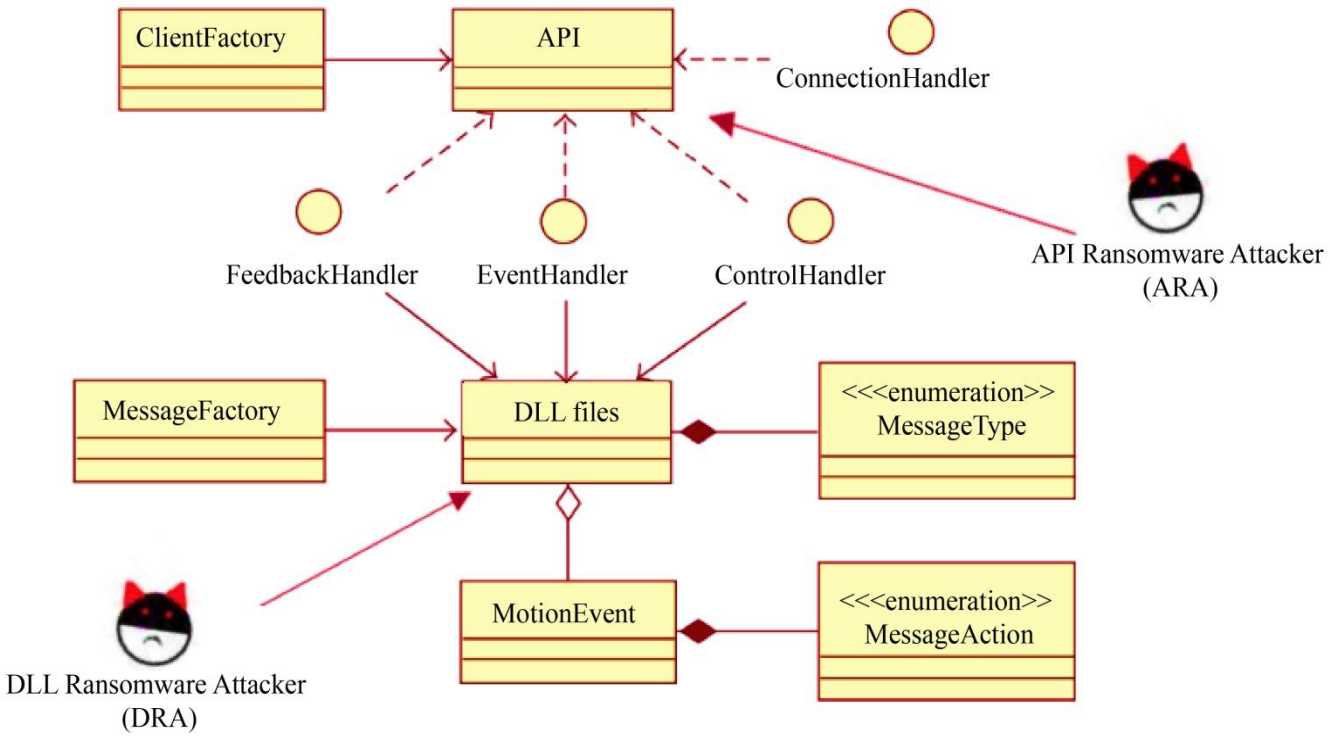


Fig. 3 Smart TV operating system corruption

3.2. Reverse Engineering and Pre-Processor

To obtain the assembly opcodes, a binary executable must be disassembled and study it for the purpose of achieving some important goals, known as reverse engineering. Figure 3 depicts the life cycle of a binary programme. Compilation of the programme source code C is written, or another language programming involves processes like a lexical analyser and a code optimiser. A binary file is connected to the newly produced object files. To create a final executable, the loader, which consists of dynamic link libraries and OS loaders, resolves the references to the code.

Reverse engineering's objective is to achieve functionality that is as near to that of the original source code as is practical. Using the objdump disassembler and the PE parser tool, reverse-engineered the malware and the typical executable. The DLLs and function calls used by both ransomware and non-ransomware samples are collected using the PE parser tool, and the assembler code for each execution sample is obtained using the systems programme. The PE parser and objdump tool produce code segments processed by the pre-processor component. In this work, we discuss portable executable files for Windows. The Windows Operating System loader needs the information contained in the PE file format in order to handle the programme code. Windows executables, object code, APIs, and DLLs all make use of it.

3.3. Feature Extraction

In the feature extraction phase, the redundant and irrelevant datas are eliminated from the pre-processed data. The features are extracted by using three approaches, namely TF-IDF-based machine learning.

TF-IDF: This technique is used to extract characters, in which the row represents Term Frequency (TF) and Term Frequency-Inverse Document Frequency (TF-IDF), two metrics used in the evaluation process, while the column represents the words. Based on the equation (1), the data is calculated as follows:

$$TF - IDF = tf(t,d) * \log\left(\frac{S}{DF+1}\right) \quad (1)$$

3.4. Classification

Multiclass SVM is used to classify ransomware attacks based on the extracted features. It carries out the classification by projecting the input vectors into a higher-dimensional space and creating a hyper-plane that best divides the data there. The Multiclass SVM is preferred because it can classify a greater variety of classes than the Support Vector Machine (SVM), which can only classify two different kinds of classes. The Multiclass SVM model was trained using the training set in this manner, and the classification accuracy performance was tested using the testing set. One could think of the SVM as a binary classifier. It classifies patterns that belong to two similar classes by abstracting a decision boundary from the data. Finding the best hyperplane over bias, h, and weight vector g is a challenging task for a training set labelled as "left" $\{p_a, q_a\}, a = 1, 2, \ldots t$ and 't'.

$$min\frac{1}{2}||g||^2 + F\sum_{a=1}^{t}\xi_a \quad (2)$$

In this instance, p is the input feature vector with the class label y ∈ {−1, 1}. The initial section aims to enlarge the gap between the two classes. The second term penalises the answers of a big slack variable ξ_a with the cost of the penalty, F., in an effort to lower the training errors. Because limitations are more difficult to disregard with a larger F value, the margin is minimised. The Lagrangian multiplier method and articulating the dual problem using the kernel trick provide a simpler solution to the constrained quadratic programming problem.

$$max\sum_a \pi_a = \frac{1}{2}\sum_a\sum_b \pi_a\pi_b q_a q_b L(p_a p_b) \quad (3)$$

Where, π_a is a constant value. SVM's optimisation problem achieves a global minimum as opposed to the local minimum that may occur in ANN, and the nonlinear feature mapping's rise in the dimensionality of the data aids in class separation that may involve a linear hyperplane. The mapping is accomplished using kernel function $L(p_a, p_b)$ and the most commonly used kernels.

$$d(p) = sign(\sum_a^t \pi_a q_a L(p_a, p)), d(p) \longrightarrow \{-1, 1\} \quad (4)$$

Using a collection of binary SVMs, created for binary classification is successfully applied to the challenge of multiclass classification. The three approaches A Directed Acyclic Graph SVM (DAGSVM), One Against All (OAA), and One Against One (OAO) is the most frequently employed for multiclass classification. In order to do pairwise classification for an l-class problem, the OAO technique builds $L(L-1)/2$ binary models. The OAO technique for multiclass classification is used in this work.

4. Result and Discussion

This section will first address dataset gathering, experimental protocol, and experimental results before analysing the most pertinent findings in accordance with the assessment methodology suggested in the section before this one.

4.1. Performance Metrics

The experimental results were evaluated with Accuracy, Precision, F1 score, and Recall. The statistical evaluation of the parameters is given below,

$$\text{Accuracy} = \frac{TP+TN}{total\ no.of\ samples} \quad (5)$$

$$recall = \frac{TP}{TP+FN} \tag{6}$$

$$f1\ score = 2\left(\frac{precision*recall}{precision+recall}\right) \tag{7}$$

$$Sensistivity = \frac{TP}{TP+FP} \tag{8}$$

$$TRP = \frac{TP}{TP+FP} \tag{9}$$

Where TN denotes the quantity of true negatives, FN denotes the quantity of false negatives, FP signifies the quantity of false positives, TP denotes the quantity of true positives, and FP denotes the quantity of false positives. By increasing the accuracy value, prediction performance can be improved. The ROC generated for two datasets is depicted in Figure 4. The proposed STAR-D model achieved a higher AUC of 0.987 for the data that can be calculated through the TPR and FPR parameters.

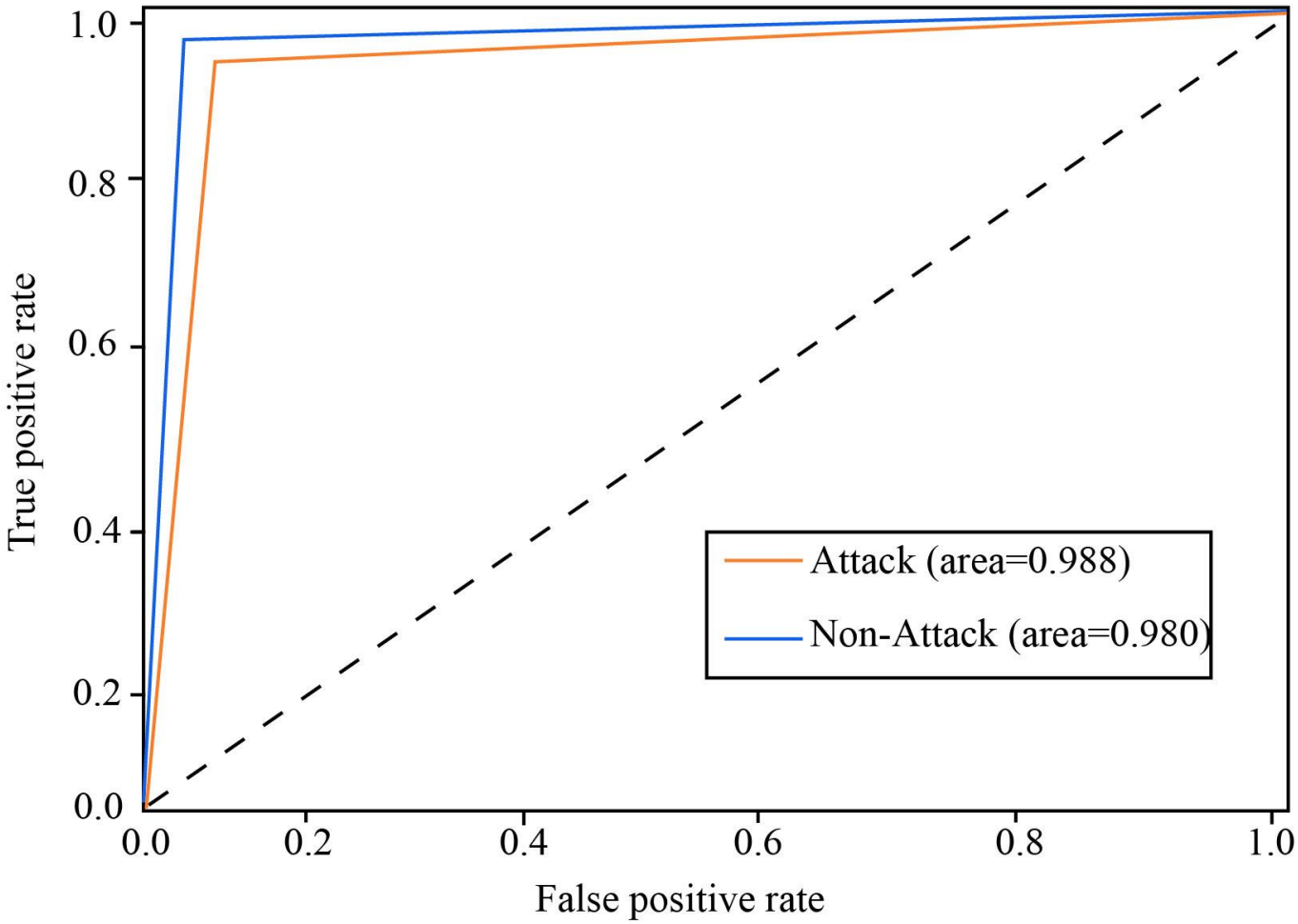
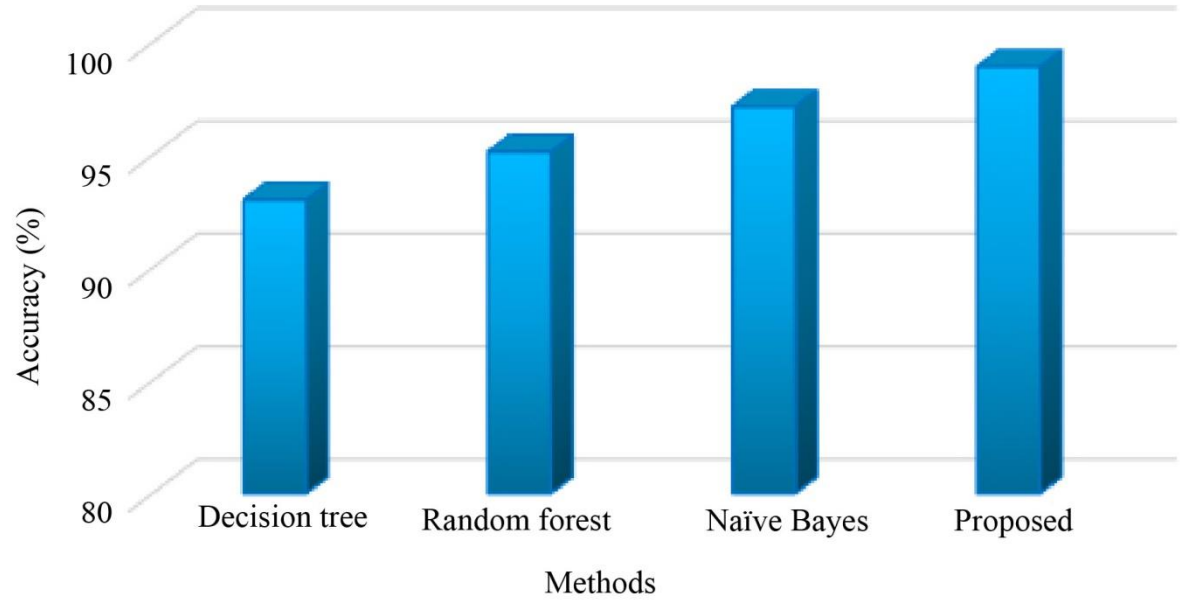


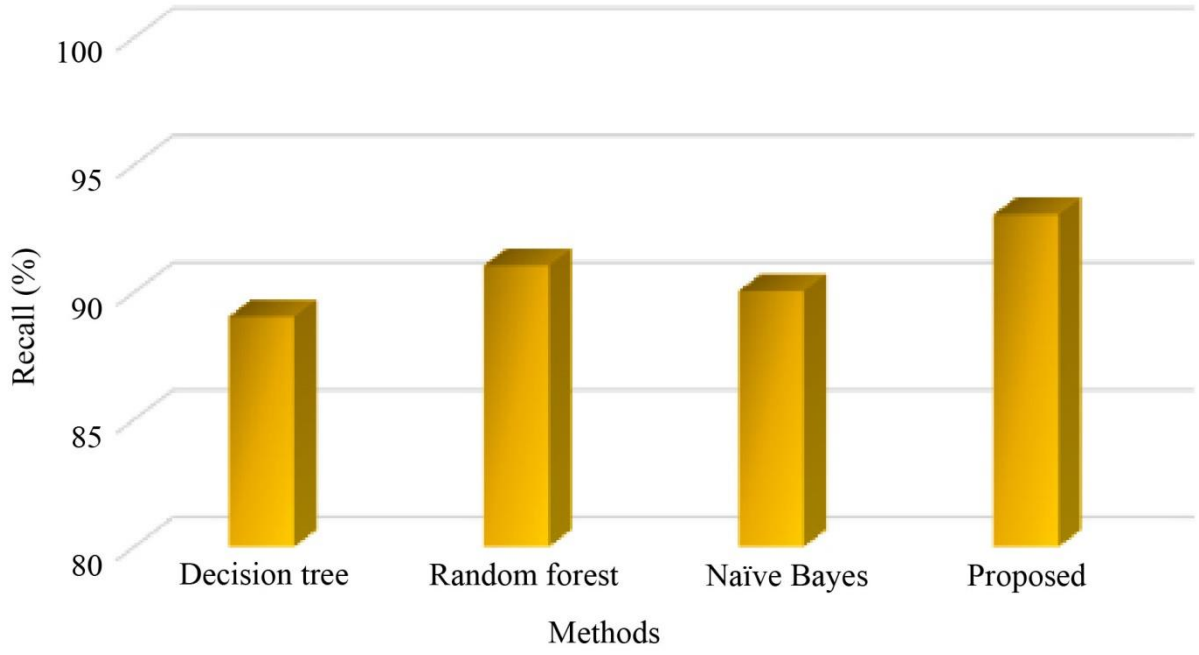
Fig. 4 ROC of the proposed method

4.2. Comparative Analysis

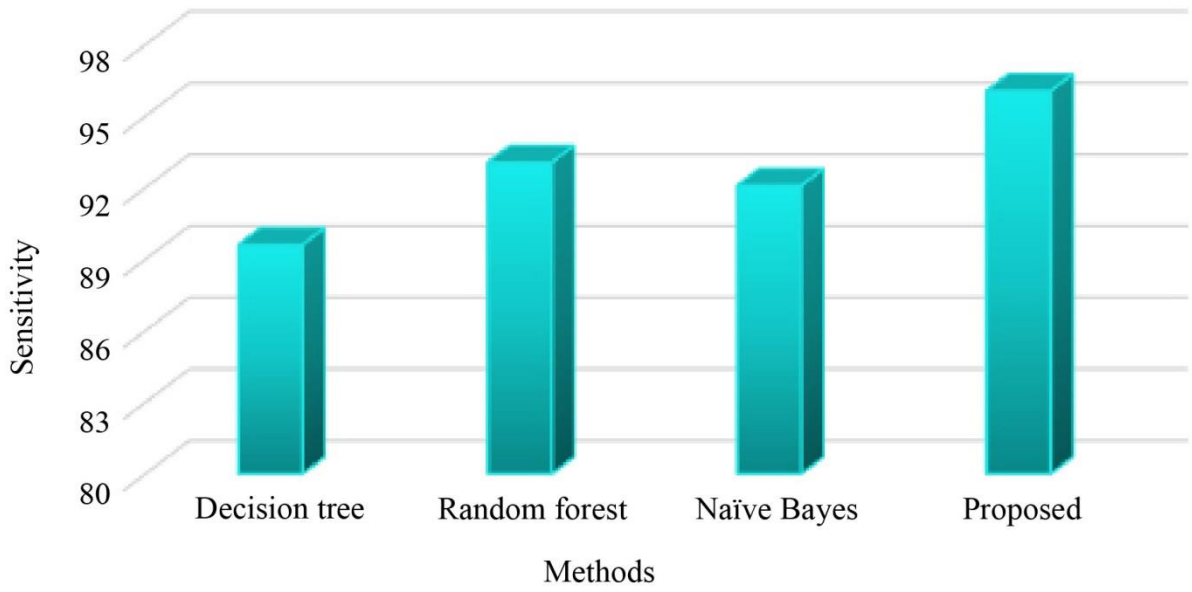
The proposed method was contrasted with existing methods in order to demonstrate that it is more efficient than others. Performance is affected by the TPR, sensitivity, recall, F1 score, and accuracy.



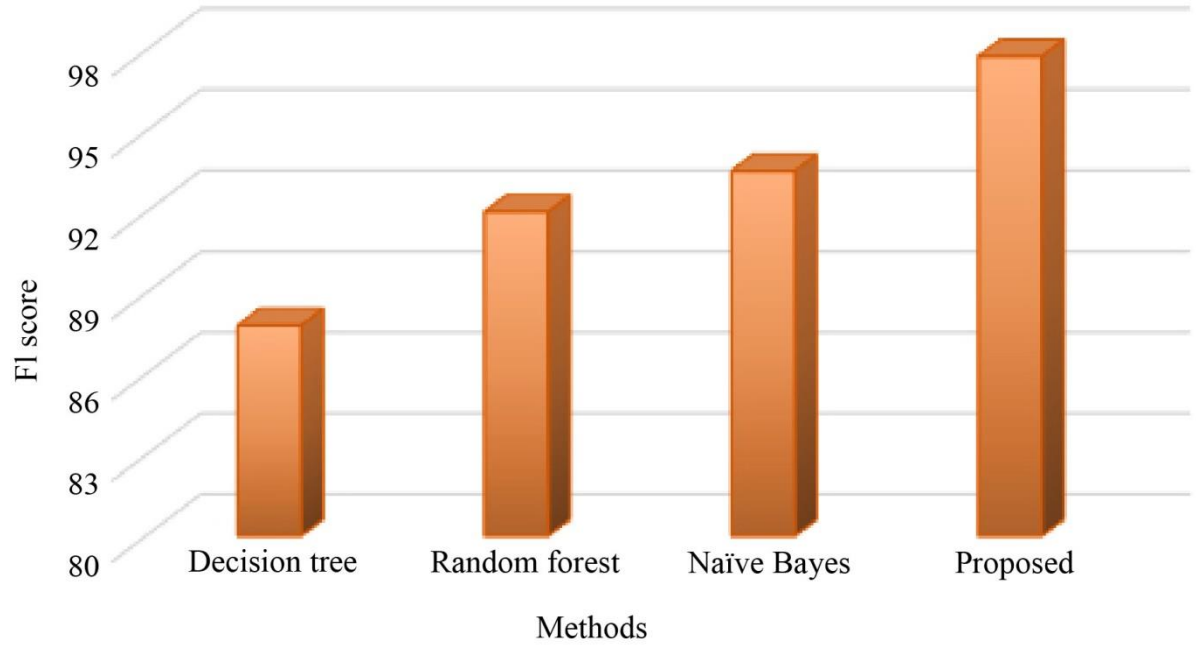
(a)



(b)



(c)



(d)

Fig. 5 Graphical representation of the performance parametric (a) Detection accuracy, (b) Recall, (c) Sensitivity, and (d) F1 score.

The graphical examination of proposed and existing methodologies with various parameters, including recall, accuracy, sensitivity, and F1 score, is shown in Figure 5 the proposed method is clearly superior to all currently used techniques and suited for attack detection, as illustrated by the figure. This comparison shows that the suggested STAR-D outperforms the current methodologies. The proposed STAR-D has a maximum accuracy of 98.95%, whereas existing models like Naïve Bayes have 97.15 %, Random Forest has 95.15 %, and Decision Tree

has 93.2 %. It demonstrates that the proposed approach is effective and yields a highly accurate outcome. The sensitivity of the proposed method is 2.3 %, 4.3 %, and 5.7 %, increased by existing NB, RF and DT methods. The recall of the proposed is 8.7 %, 6.2 %, and 5.6 % better than the existing methods.

Table 1. Comparison of accuracy, TPR & Error rate

Methods	Accuracy	TPR	Error Rate
Decision Tree (DT)	93.08	92.05	6.94
Random Forest (RF)	95.15	95.15	4.84
Naïve Bayes (NB)	97.26	96.25	2.74
Proposed STAR-D	98.92	98.95	1.08

Table 1 shows the ransomware detection accuracy, error rate and TPR using different machine learning algorithms. This enables an evaluation of the results of the suggested methods and existing DT, RF and NB techniques. From the results obtained, it can be seen that the proposed method performs better than those obtained using other algorithms. From the table, it is clear that the accuracy increases error rate is decreased.

5. Conclusion

In this study, a unique Smart TV Attack Ransomware Detection (STAR-D) method based on Multiclass SVM and DLL/Application Programming Interface (API) file features has been proposed. The Dynamic Link Library (DLL) and API are just two of the numerous levels at which the framework examines ransomware using machine learning. TFIDF were used to process further the raw data from the malware and smart TV in order to produce the final feature sets. Using the detection engine, the DLL file assesses the DLLs in a given binary and determines the categorisation accuracy. Tearing apart an executable in binary form to get the assembly opcodes and study it for the purpose of achieving some important goals is known as reverse engineering. In the feature extraction phase, the redundant and irrelevant data are eliminated from the pre-processed data. The features are extracted by using three approaches, namely TF-IDF-based machine learning. Multiclass SVM is used to classify ransomware attacks based on the extracted features. It carries out the classification by projecting the input vectors into a higher-dimensional space and creating a hyper-plane that best divides the data there. The proposed method was compared to existing methods in order to demonstrate that it is more efficient than others. The TPR, sensitivity, recall, F1 score, and accuracy contribute to performance. The suggested approach outperforms Random Forest, Naive Bayes and Decision Tree in terms of overall accuracy by 0.82%, 1.32%, and 3.57%, respectively.

References

[1] Francisco Martinez-Pabon et al., "Smart TV-Smartphone Multiscreen Interactive Middleware for Public Displays," *The Scientific World Journal*, vol. 2015, pp. 1-14, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[2] Syed Rameem Zahra, and Mohammad Ahsan Chishti, "Ransomware and Internet of Things: A New Security Nightmare," *2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pp. 551-555, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[3] Amin Azmoodeh et al., "Detecting Crypto-Ransomware in IoT Networks Based on Energy Consumption Footprint," *Journal of Ambient Intelligence and Humanized Computing*, vol. 9, pp. 1141-1152, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[4] Mamoona Humayun et al., "Internet of Things and Ransomware: Evolution, Mitigation and Prevention," *Egyptian Informatics Journal*, vol. 22, no. 1, pp. 105-117, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[5] Ibrar Yaqoob et al., "The Rise of Ransomware and Emerging Security Challenges in the Internet of Things," *Computer Networks*, vol. 129, pp. 444-458, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[6] Trung Kien Tran, and Hiroshi Sato, "NLP-Based Approaches for Malware Classification from API Sequences," *2017 21st Asia Pacific Symposium on Intelligent and Evolutionary Systems (IES)*, pp. 101-105, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[7] Md Mahbub Hasan, and Md. Mahbubur Rahman, "Ranshunt: A Support Vector Machines Based Ransomware Analysis Framework with Integrated Feature Set," *2017 20th International Conference of Computer and Information Technology (ICCIT)*, pp. 1-7, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[8] Umme Zahoora, "Zero-Day Ransomware Attack Detection Using Deep Contractive Autoencoder and Voting Based Ensemble Classifier," *Applied Intelligence*, vol. 52, no. 12, pp. 13941-13960, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[9] Subash Poudyal, Kul Prasad Subedi, and Dipankar Dasgupta, "A Framework for Analyzing Ransomware Using Machine Learning," *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1692-1699, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[10] Firoz Khan et al., "A Digital DNA Sequencing Engine for Ransomware Detection Using Machine Learning," *IEEE Access*, vol. 8, pp. 119710-119719, 2020. [CrossRef] [Google Scholar] [Publisher Link]