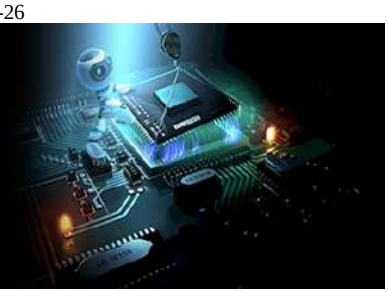
International Journal of Engineering in Computer Science



E-ISSN: 2663-3590
P-ISSN: 2663-3582
IJECS 2021; 3(2): 22-26
Received: 02-12-2021
Accepted: 13-12-2021

Mayank Parashar
Assistant Professor, HMR Institute of Technology and Management, New Delhi, India

Tanvi Dhingra
Student, HMR Institute of Technology and Management, New Delhi, India

Rajat Nagar
Student, HMR Institute of Technology and Management, New Delhi, India

Pawan Kant Tiwari
Student, HMR Institute of Technology and Management, New Delhi, India

Deepika Singh
Student, HMR Institute of Technology and Management, New Delhi, India

Correspondence
Mayank Parashar
Assistant Professor, HMR Institute of Technology and Management, New Delhi, India

Malware detection using machine learning with cloud support

Mayank Parashar, Tanvi Dhingra, Rajat Nagar, Pawan Kant Tiwari and Deepika Singh

DOI: https://doi.org/10.33545/26633582.2021.v3.i2a.55

Abstract
This paper "*malware detection using machine learning with cloud support*" aims to present the functionality and accuracy of five different machine learning algorithms to detect whether an executable is infested or clean. It starts by prompting the user to login into the system by entering valid credentials. Upon successful validation of the details, the user is asked to upload the binary that he/she wants to store in the cloud database. After this, when the user uploads a file from his system, the underlying machine learning algorithms would analyse that the file is benign or infested by means of various classification algorithms. The output provided by the algorithm with highest accuracy is considered and the corresponding result is displayed to the user. The user thus becomes aware if the file uploaded by him/her contains malware or not. In case the file is found to be containing malware, it is discarded, and on the contrary if it is found to be legitimate it is stored on the Cloud by making use of Amazon S3(Simple Storage Service) and thus can be accessed by the user anytime in future. This is how malware is detected successfully by making use of Web, Machine Learning and Cloud Services.

Keywords: Security, malware detection, machine learning, pe files, amazon simple storage service, legitimate, infested

1. Introduction
As a result of the Internet's fast expansion, malware has emerged as one of the most severe cyber threats in recent years. As previously said, malware is any application that performs destructive actions such as data theft, espionage, and so on. Malware is described as "a type of computer programme designed to infect and destroy a legitimate user's machine in a number of ways." Anti-virus scanners are unable to keep up with the increasing diversity of malware, resulting in millions of hosts becoming infected.

As a result, malware protection for computer systems is one of the most critical cyber security jobs for both individuals and enterprises, as even a single attack may result in data breaches and significant losses. As a result, in these frightening times of escalating cybercrime, confronting the problem and taking serious and urgent action has become critical. Our team opted to utilise the Flask Framework, a micro web framework developed in Python that is usually used for constructing machine learning models, to create a login-based website that provides real-time detection of any executable (.pkl). The system would first ask the user to check in with their credentials, and then, following successful verification, it would ask the user to upload any executable file that he or she desired to save in the cloud database. The underlying machine learning algorithms will then utilise several categorization methods to assess if the file is benign or contaminated when the user uploads a file from his PC. Jinja would be used as a full-featured template engine for the complete website's operation, including automated page rendering, static file URL creation, and machine learning prediction results retrieval. For the frontend design, we'd utilise bootstrap and j Query to build HTML, CSS, and JavaScript.

2. Objective
The purpose of this research is to show that malware classification can be done using machine learning and sophisticated classifiers. Anti-virus scanners are unable to keep up with the rising diversity of malware, resulting in millions of hosts being infected. As a result, malware protection of computer systems is one of the most critical cyber security jobs for both individuals and organisations, as even a single assault may result in data breach and significant losses. As a result, approaches based on machine learning can be applied.

From all of the supplied characteristics, the best features are extracted. The approach with the lowest error rate is capable of detecting a malicious file. Both the accuracy of assessing whether a file is harmful and the accuracy of categorising it into a malware family will be evaluated. The project would feature a simple user interface for uploading files and scanning them for viruses before posting.

3. Material / Methods / Tools

3.1 Technologies to be incorporated and worked upon:

1. Machine Learning (Detecting Software to be Malicious or Legitimate)
2. Web Development (Frontend and Backend)
3. Cloud (AMAZON WEB SERVICES S3)
   Note: S3 stands for Simple Storage Service

3.2 Languages and frameworks used

For Machine Learning

- Python
- Scikit Learn

For Frontend and Backend Web Interface

- Flask - Python Web Framework
- HTML
- CSS
- BOOTSTRAP - Open-Source CSS Framework
- jQuery - JavaScript Library
- Flask
- Jinja
- Werkzeug

For User Authentication

Google Firebase

For cloud file storage

- AWS S3
- Boto3 - A Python SDK for AWS
- AWS Toolkit for debugging

4. Proposed work: Malware detection system

We propose a real-time Login - based web-Interface that would alert the user on detection of any malware, and upon successful classification that a file is benign would store the same on Amazon S3 that provides easy-to-use management features so that users can organize their data and configure finely tuned access controls to meet any of their specific business, organizational, and compliance requirements. Various features such as login, registration for first-time users, and an upload area for detecting files that the user wishes to save in the cloud would all be available on the website.

4.1 Machine Learning

Detection of a file as malware or benign is done through Machine Learning. Dataset is first loaded from a set of 138,000+ legit and malware files from VirusShare.com into a Data frame using Pandas which is a popular data manipulation package in python and the rows corresponding to Name, md5 and legitimate are dropped using the drop method. Moving onto the Feature Selection wherein our goal is to find the best set of features for our classifier and for that we use Extra Trees Classifier which is a type of ensemble learning technique that aggregates the results of multiple de-correlated decision trees collected in a "forest" to output it's classification result. For checking how well the training goes, we split the data into a Pareto ratio of 80:20 and finally once we have our important features, we proceed to create an array of models. Then each model is tested on the dataset using the extracted features as inputs and compare their prediction results. At last, the model having the best results is the one that we used for Malware Detection. In the 'for' loop, we tested each algorithm out, fitting them on the feature set and then scoring the prediction accuracy, each score is then printed out and the winner is calculated by finding the highest prediction accuracy. The Algorithm's weights and features are then saved for later predictions as Pickle files thus marking the end of our classifier's training.

4.2 UI-Design

Starting with the Login/Registration page, there would be three template HTML files to be used for the website – login, index and result. After the successful login of the user, index is loaded when checker.py is executed and it provides an interactive file upload button where users can browse/drag an alien PE file and it would be checked against the Machine Learning model. It is an interactive web page with responsive buttons and links and various resources such as Google font, Bootstrap, Owl Carousel, Magnific popup, CSS and Awesome Icon. The final result would be displayed on the result page i.e the file is malicious or not. The web application also takes into use static files such as JavaScript and CSS, supporting the display of a web page. Usually, the web server is configured to serve them for us, but during the development, these files are served from the static folder in your package or next to your module and it will be available at /static on the application. A special endpoint 'static' is used to generate URL for static files.

4.3 Flask as Framework

The Interface is built upon Flask which is a micro web Framework written in Python that uses the routing technique to help a user remember application URLs. It is useful to access the desired page directly without having to navigate from the home page. The route decorator in Flask is used to bind URL to a function. Entropy is used to measure the amount of data which is present in a selected file. File Entropy is also use in the field of malware protection, in the process of malware analysis as there are all kind of security related tools that we checked on the file to extract all kind of information from the file, to determine if the file is a malware or legit file, and if it is a malware this can be useful on the malware file entropy can be a useful method to quickly check if the malware file had been packed with one of the packed software it is also a good method to check if the file encrypted by one of the encryption algorithm. Storing resources in the dictionary res of our test file where we tested our machine learning model. For this, we call 3 functions all using PE file module

1. def get_resources (peFile) - to Extract resources [entropy, size]
2. def get_version_info (pe) - Returns version information of test.exe file
3. Get all the header files and features in a dictionary called res using def extract_infos(fpath):

After having extracted all the features and header files out of test. Exe, we ran our machine learning model to predict if

test.exe is malicious or legitimate. Here, we upload the alien PE file and run malware analysis on it and finally the result is displayed on the result template.

4.4 Amazon S3 as Cloud Storage

An S3 object is created when we upload a file to Amazon S3. Objects are made up of data from files and metadata that characterises them. A bucket may hold an infinite number of items. To establish, configure, and administer AWS services, we must first install boto3, which is the AWS SDK for Python (Boto3). We used the IAM dashboard to set up login credentials for our AWS account, and then installed the AWS CLI to utilise the AWS configure command to setup our credentials file before utilising Boto3.

Then we used boto3.client() to create an S3 client and set up the bucket name, and then we used upload file to upload the supplied file using a managed uploader, which would automatically break up huge files and upload pieces in parallel.

4.5 Firebase as Authentication Service

Firebase is used as Email and password-based authentication which authenticate users with their email addresses and passwords. The Firebase Authentication SDK provides methods to create and manage users that use their email addresses and passwords to sign in. We used Firebase Authentication to let the users authenticate with Firebase using their email addresses and passwords, and to manage their app's password-based accounts.

5. Workflow

Overall Project is partitioned into various phases.

Phase 1: Downloading 1,38,000+ dataset of malicious and legitimate PE files.

Phase 2: Performing Malware analysis and classification, choosing best out of 5 Machine Learning Algorithms and feature classification.

To categorise a file as malware or not, we will apply five machine learning methods namely:

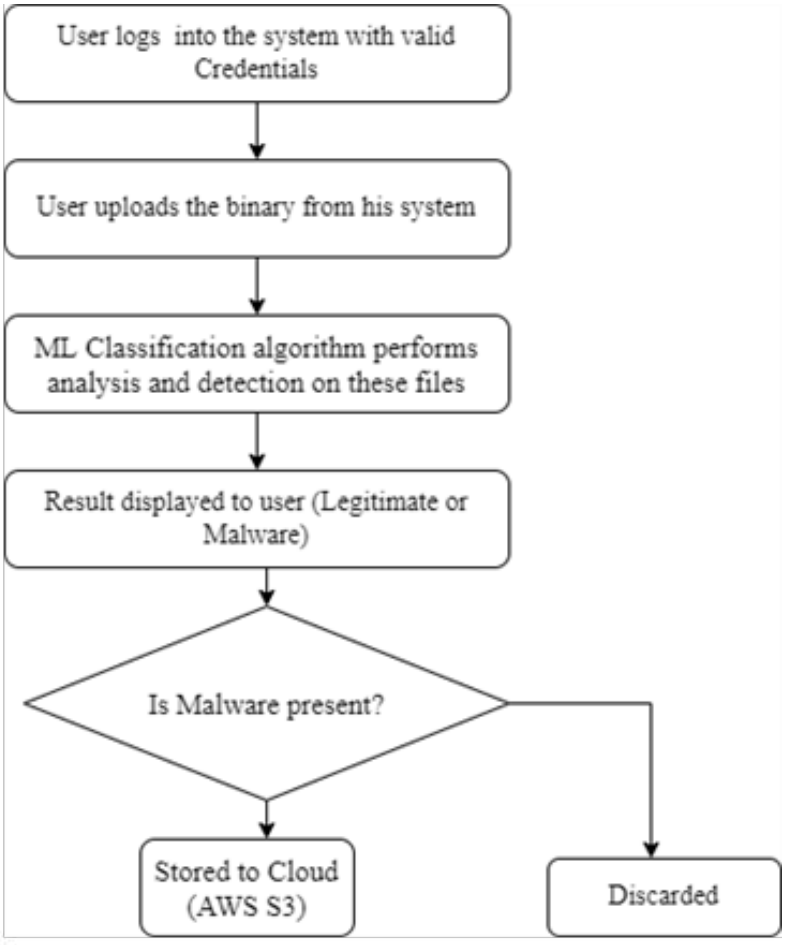
- Decision Trees
- Random Forest
- Gradient Boosting
- Adaptive Boosting
- Gaussian Naive Bayes

We chose the best of them based on maximum accuracy as our final model.

Phase 3: Deploying Full stack Flask framework using HTML, Bootstrap, CSS for static analysis.

Phase 4: Adding User Email-Password authentication using Google firebase.

Phase 5: Once the file is detected malware free, we save the file into Amazon AWS cloud storage.



6. Result and Discussion

Fig 1: User Login Page

Fig 2: Uploading of File

Fig 3: Project Interface

Fig 4: If the file contains Malware, the following is displayed to the user

Fig 5: If the file is Legitimate, the following is displayed to the user

7. Acknowledgement

Our thanks to the Professors, experts and other faculty members who provided useful resources and background to complete this research paper. The Success and outcome of this project were possible by the guidance and support from many people. We are incredibly privileged to have got this all along with the achievement of this paper. It required a lot of effort from each individual involved in this research paper.

8. Conclusion and future scope

The purpose of this research is to provide a machine learning method to the problem of malware. We need automated solutions to detect infected files due to the rapid rise of malware. The data set is produced in the initial part of the study utilising infected and clean executable, and we utilised a Python script to extract the data required for the data set construction. The data collection must be ready to train machine learning algorithms after it has been created. Decision trees, Random Forest, Naïve Bayes, Gradient Boost, and ADA Boost are the algorithms that were employed. It has a Random Forest algorithm with an accuracy of 99.406012 percent after using the best accuracy methods. This research shows that Random Forest is the best method for identifying malicious applications, and that valid files are successfully detected and saved in Amazon S3 storage. This precision will become more important in the future. This research shows that Random Forest is the best method for identifying malicious applications, and that valid files are successfully detected and saved in Amazon S3 storage. This accuracy can be enhanced in the future by including a significantly higher number of files in the data set used to run the algorithms. Each algorithm contains a number of parameters that may be experimented with to improve accuracy. With the aid of a library called pickle, this project may go to the application level, where we can preserve what the algorithm has learnt and then test a new file to determine whether it is clean or infected. Furthermore, for a user to safely use the Detection programme, the entire system is made more secure by integrating Email-password based authentication. Static analysis has also shown to be more secure and devoid of execution time overhead. The accuracy of this may be improved by adding more datasets. Accuracy may be improved by adding more algorithms with greater performance.

9. References

1. Dragoş Gavriluţ, Mihai Cimpoeşu, Dan Anton, Liviu Ciortuz. Malware detection using machine learning | IEEE Conference Publication, 12-14 October 2009, Proceedings of the International Multi-conference on Computer Science and Information Technology 2009, 735-741.
2. Rieck K, Holz T, Willems C, Ussel PD, Laskov P. Learning and classification of malware behavior, in DIMVA ’08: Proceedings of the 5th international conference on Detection of Intrusions and Malware, and Vulnerability Assessment. Berlin, Heidelberg: Springer-Verlag 2008, 108-125.
3. VirusShare.com VirusShare.com for malware dataset in windows PE format for analysis and prediction.
4. Boto3 documentation — Boto3 Docs 1.18.27 documentation for configuring and managing AWS services using Boto3 (AWS SDK for python).
5. Malware Types and Classifications, Bert Rankin, 28.03.2018, published in Last Line, last accessed 12.09.2018.
6. A Brief History of Malware - Its Evolution and Impact, Bert Rankin, 05.04.2018, published in Last Line, last

accessed 12.09.2018.
7. Detecting malware through static and dynamic techniques, Jeremy Scott, 14.09.2017, published in NTT Security, last accessed 12.09.2018.
8. Hybrid Analysis and Control of Malware, Kevin A. Roundy and Barton P. Miller, International Workshop on Recent Advances in Intrusion Detection 2010, 317-338, Springer.
9. Advanced Malware Detection - Signatures Vs. Behavior Analysis John Cloonan Director of Products, Last line, 11.04.2017, published in Info security Magazine, last accessed 12.09.2018.
10. What is Machine Learning? Daniel Faggella, 12.08.2017, published in tech emergence, last accessed 12.09.2018.
11. Data mining, Margaret Rouse, Search SQL Server last accessed 12.09.2018, the article can be found here. https://searchsqlserver.techtarget.com/definition/data-mining.
12. Supervised and Unsupervised Machine Learning Algorithms, Jason Brownlee, 16.03.2016, published in Machine Learning Algorithms, last accessed 12.09.2018.
13. Decision trees, scikit-learn.org last accessed 12.09.2018.
14. Random Forest Classifier, scikit-learn.org last accessed 12.09.2018.
15. Gradient Boosting Classifier, scikit-learn.org last accessed 12.09.2018.
16. Malware Researcher's Handbook, Resources InfoSec institute, last accessed 12.09.2018.