Content is available at: CRDEEP Journals
Journal homepage: http://www.crdeepjournal.org/category/journals/global-journal-of-current-reseach-gjcr/

Global Journal of Current Research

(ISSN: 2320-2920) (Scientific Journal Impact Factor: 6.122)

UGC Approved-A Peer Reviewed Quarterly Journal

Review Paper

A Study of Malware Detection Approach based in the Cloud Environment: a Preliminary Review

S M Firdaus Zaki Rizvi[1] and Dr. Amar Nath Chatterjee[2]

[1]-Research Scholar in Computer Science, Magadh University, Bodh Gaya, India
[2]-Assistant Professor, Department of Mathematics, K.L.S. College, Nawada, India.

ARTICLE DETAILS

Corresponding Author:
S. M Firdaus Zaki Rizvi

Key words:
Cloud computing, Machine Learning, Cloud malware detection, Malware file ,cyber-physical system, Legitimate file.

ABSTRACT

Cloud computing has an important role in allaspects of storing information and providing services online. It brings several advantages over traditional storing and sharing schema such as an easy access, on-request storage, scalability and decreasing cost. Using its rapidly developing technologies can bring many advantages to the protection of Internet of Things (IoT), Cyber-Physical Systems (CPS) from a variety of cyber-attacks, where IoT, CPS provides facilities to humans in their daily lives. Since malicious software (malware) is increasing exponentially and there is no well-known approach to detecting malware, the usage of cloud environments to detect malware can be a promising method. Now a days the cloud environment has seen a sudden growth in its users and about 52% of the organisations have stepped up to use the cloud infrastructures just because of its flexible resources and economic scale it provides. When it comes to malware, malware is any type of software or a bug which tries to self-replicate or harm the hardware or a software of a system. These types of attacks are not known to the human eye as they are built with malicious intent to harm any system in use. So overall to overcome the problems faced and make a flexible solution, we propose a framework where Machine learning algorithms are used to find the best features from the data set provided by us and give an accuracy report, and the classifiers used among them best algorithm prediction will be used to extract the best features and use them.

I. Introduction

In recent years, cyber-related attacks to the word economy have been increasing exponentially. According to Steve Morgan, cyber-attacks damage the world economy about $6 trillion in 2021. According to the researchers, these days' approximately more than 1 million malicious software files are created every day and the cost of the malware especially to cyber-physical systems (CPS) and critical systems is rising as well. McAfee report shows that there is an outrageous increase in backdoors, banking Trojans, and fake applications for mobile devices. When it comes to initiating a cyber attack in the context of information security, malware is one of the main risks. Malware is any program, including viruses, worms, root kits, backdoors, and ransom ware, that installs itself on a victim's computer and carries out destructive tasks with or without the owner of the system's permission. Malware has been increasing at an alarming rate over the last ten years, and there is currently no reliable method for identifying all malware that exists in the public. This new breed of malware evades detection by employing sophisticated packaging and obfuscation techniques. This means that using a conventional method to identify complicated malware is almost difficult.

The method of identifying the existence of malware by examining program executables is known as malware detection. Numerous methods, both conventional and cutting-edge, have been put forth to identify malware. For almost ten years, conventional methods such as behavior-, heuristic-, model-checking and signature-based detection approaches have been employed. A variety of methodologies, such as cloud computing, edge computing, machine learning, and deep learning, constitute the foundation of advanced techniques. It is commonly known that the signature-based detection method works well in terms of memory use and time, but it is unable to identify malware that is not yet known. Heuristic,

[1] *Author can be contacted at: Research Scholar in Computer Science, Magadh University, Bodh Gaya, India*
Received: 15-9-2024; Sent for Review on: 18-09-2024; Draft sent to Author for corrections: 28-09-2024; Accepted on: 10-10-2024; Online Available from 20-10- 2024
GJCR-7895/

behaviour, and model checking-based techniques are capable of detecting a large percentage of malware; nevertheless, they are unable to identify a fraction of zero-day malware. Similar techniques to those used in the signature, heuristic, and behavior-based approaches are also employed in deep learning and edge computing (mobile devices) based detection approaches; however, these approaches are unable to identify sophisticated and zero-day malware. The direction of malware detection schemas is shifting from conventional to novel ones. Cloud-based detection is among the most successful new methods of detection. In cloud computing, it consists of a client and a server. The client sends a suspicious file via the internet, and the server analyses it and determines whether or not it contains malware. To enhance performance, the server employs several detecting agents throughout the analytic process. Strings, system calls, static and dynamic features, application traces, API traces, and hybrid features are all employed in the feature extraction step. According to recent research, using a cloud-based detection technique improves the rate of both known and unknown malware detection and offers a more thorough examination of each malware sample. The use of cloud-based detection techniques has several benefits over conventional methods. Larger datasets and more processing capacity are available in cloud environments for malware detection. It is possible to compile many execution traces of the same virus. It also enhances CPS, mobile devices, and desktop computers' detecting capabilities. However, there are also certain disadvantages, such diminished control over data, increased overhead between the client and server, a deficiency in real-time monitoring, and restricted resource utilisation. In addition to providing a thorough analysis of cloud-based malware detection techniques, this review paper adds the following features:

- Provides an overview of recent scholarly research on cloud-based malware detection techniques.
- Offers a perspective on how cloud computing might shield cyber-physical systems from viruses.
- Describes the current trends in malware production and concealment methods.
- Talks about the difficulties of today and offers fresh methods for detecting malware.
- Offers a platform for cloud-based malware detection that combines heuristic, behaviour, deep learning, and signature-based techniques. The rest of this paper is organized as follows. Section II describes trends in malware creation and hiding techniques. Section III explains an overview of cloud-based malware detection systems. Related work on cloud-based malware detection approach is summarized in section IV. Discussion and evaluation of cloud-based malware detection. System Design approach is presented in this section V. presents the proposed Discussion and evaluation of modules in this Section VI.Proposed the methodology in section VI. Finally, conclusion and future work is given in section VIII.

2. Trends In Malware Creation And Hiding Technologies

Malware is described as software that infects a victim's computer with harmful programs. Malware comes in several forms, such as viruses, worms, backdoors, rootkits, and ransomware. Tables I and II show the many forms of malware, their salient features, and well-known malware families. Malware, which takes advantage of mistakes, weaknesses, and malfunctions in current systems including buffer overflows, security misconfigurations, and defects in computer networks protocols, is being used by hackers to conduct cyberattacks. Malware classification is become increasingly challenging these days since many cases of the virus exhibit traits from several classifications at once. The first kind of malware to arise in the wild is a virus. different kinds of viruses that quickly surface in computer systems. Malware was first developed for naive reasons, including breaking into friends' computers or making a little money. But over time, it was replaced with a sophisticated virus that caused financial harm to major corporations, sectors of the economy, and government properties. Malware falls into two categories: next generation malware and traditional malware. Next generation malware is more damaging and more challenging to identify and remove from computer systems than traditional malware, which is common malware that is simple to identify and remove from computer systems. Furthermore, malware of the future generation can conceal itself and easily evade security programs that operate in kernel mode. Cyberattacks that are both persistent and targeted can be initiated with the use of next-generation malware. Various malware kinds are employed in the attacks. Common obfuscation techniques are most often used by next generation malware to evade detection systems. Table III lists common obfuscation strategies along with an explanation of each. Next-generation malware is almost hard to find with a single detection method. Thus, it is imperative to identify malware using a variety of techniques and increased computing capacity.

Table 1. : Types of malware and primary characteristics

Malware Types	Main Characteristics
Virus	Common and well-recognized malware
Worm	Spreads by using networks
	Allows unauthorized access to CPS systems
Trojan Horse	Appears to be a normal software
	Sends secret information to other parties
Backdoor	Bypasses security systems
	Opens systems to remote access
Rootkits	Provide privileged access
	Hide their suspicious codes from the host system
Ransomware	Encrypts the data on infected system
Obfuscated malware	Uses concealing techniques to hide itself in the systems

3. Overview of cloud-based malware detection system

A new paradigm for obtaining a range of services, such as storage, computation, data management, communications, media services, machine learning and artificial intelligence, developer tools, and security, is quickly taking shape in the form of cloud computing. The many cloud deployment models, services, and users are displayed in Figure 1. Data may be accessed at any time, from any location, and on any device thanks to cloud computing services. But this advantage in access can also pose a severe risk by providing malware with simple access. A number of grave security risks, such as the theft of identity, virtual machine hijacking, syphoning off of sensitive data, and damage to systems, can be brought on by cloud malware. Malware has the ability to operate within virtual computers on the cloud and can be the source of user data theft. Malware detection techniques based on behaviour, signatures, and machine learning have all been proposed for cloud environments. Leading cloud service providers (CSPs) including Google, Azure, and Amazon Web Services (AWS) [7] also offer security services for cloud-level malware detection. Google has also introduced a new tool to identify contemporary threats. Amazon GuardDuty is a threat detection service that analyses unauthorised behaviour to detect malware.

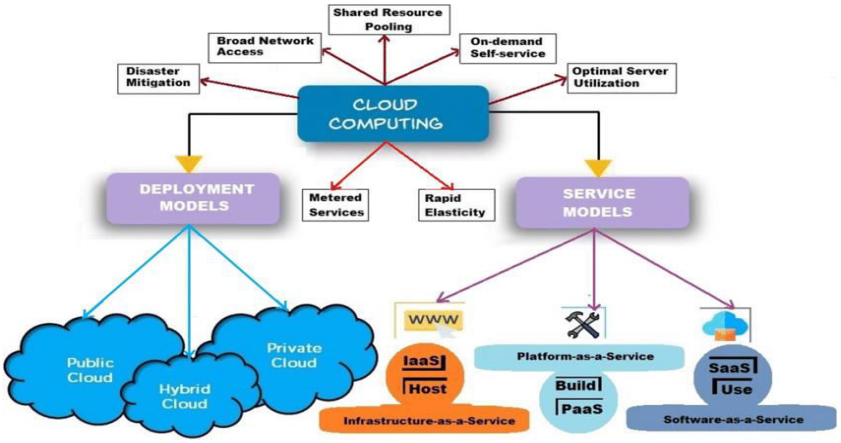

Fig. 1.Cloud Computing technologies and deployment models.

The flow of malware detection at the cloud level is depicted in Fig. 2. Several people, utilising PCs, smartphones, and the Internet of Things, obtain files through email, HTTP, media, instant messaging, P2P, and other channels. These users obtain the file report after uploading their files to the cloud. Signature-based detection techniques in the cloud identify malware by comparing patterns that are saved on the cloud. The signature-based method is said to be very quick and precise for malware that is already known, but it is not able to identify newly discovered malware. Anomaly-based detection techniques identify new malware and identify it by observing its behaviour; nevertheless, they can produce false alarms. Machine learning-based detection techniques have been studied in the past to identify malware, and the performance and efficiency of these techniques are promising. LSTM, support vector machines, decision trees, and other algorithms are used in machine learning-based malware detection techniques. This method only functions well if there is a enough amount of data and processing capacity available to train the models. Nevertheless, the scalability problem also affects malware detection techniques based on machine learning. LSTM, support vector machines, decision trees, and other algorithms are used in machine learning-based malware detection techniques [8]. This method only functions well if there is a enough amount of data and processing capacity available to train the models. Nevertheless, the scalability problem also affects malware detection techniques based on machine learning.

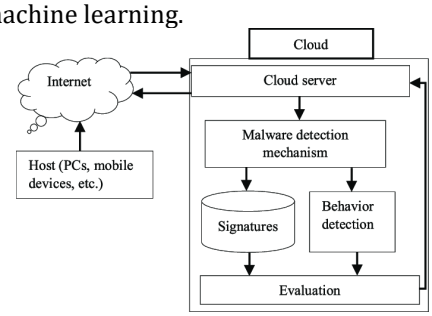

Fig. 2. Cloud-based malware detection system.

Table 2: Well-Known Malware Families

Malware MD5	Malware Type	Malware Family
f3c0c179d69ea7bcede7d908ca2c4cd0	Worm	Worm.Win32, Worm.Vobfus.Gen
3c1a983e85b56ed3c7304e7446fc5 10	Dropper	Trojan.Win32,TrojanDropper
f3c68eb51119fabbb0538b3ab6711c43	Adware	Generic Adware

f3cbd0045a27c7736f78788c2321d66c	Backdoo	Hacktool.Win32, Backdoor.Win32
f3d3a9cdbb62f8d9bb5e15eaa414597d	Ransomware	Trojan.Win32,Ransom:Win32
87f88e2e48c0f45ba32b9535e8a9c2ab	Packed Malware	Gen:Heur.Cridex,TR/Crypt.XPACK.Gen
48cd89827939b3a8976d9bb0993bc338	Spyware	Win.Spyware.Zbot,
9085a7dff20d6a5c287d3056d3ed1cc4	Rootkit	Dropper.Genericr .AC,Win32 : Rootkit gen

Table 3: Common Obfuscation Techniques

Malware Types	**Main Characteristics**
Encryption	It hides malicious code blocks in its entire code
Oligomorphic	It uses different keys while encrypting and decrypting
Polymorphic	It uses layered encryption and encrypted portion contains various decoder copy
Metamorphic	It hides malicious code blocks and change malware for every iteration Each version of malware is different
Stealth	It performs various hiding techniques in order to escape from the detection systems
Packaging	It compresses malware to disguise from the detection systems To analyze correctly, malware must be unpacked

4. Related Work

Actually, the idea of "cloud computing" dates back to the 1950s, when businesses and educational institutions began to have access to large-scale mainframes. Furthermore, the cloud model's on-demand computing idea dates back to the 1960s time-sharing period. As a result, it's possible to argue that a large number of the security problems with cloud computing are comparable to those that emerged during the period of Internet proliferation. Nonetheless, the widespread use of virtualisation, service-oriented architecture, autonomic, and utility computing led to the advancement of malware detection in the cloud, or what we now refer to as cloud computing. specifics like the location of a component or infrastructure. Most end users don't know anything about devices, thus they don't need to be knowledgeable about, in charge of, or thorough with the technical infrastructure supporting their computing activities. This research is connected to a number of earlier studies that address the entirety of cloud computing, its architecture, and the detection methods employed for each of the Static analysis, detection: Using a heuristic approach, optimise signature matching for dynamic analysis detection.

5. System design

The general structure construction outline is developed by the framework configuration preparation. Programming diagrams include addressing the way the item system functions in a way that can be altered to fit at least one expectation. The crucial information provided by the final client has to be arranged methodically. A diagram is an imaginative system; the greatest method for achieving a sensible structure is an amazing design. The framework known as "Layout" is described as "The methodology of applying distinctive frameworks and guidelines with the ultimate objective of describing a strategy or a system in sufficient purpose important to permit its physical affirmation" . Various design elements are added to the system thereafter. The system's components, the structure's parts or segments, and how they look to end users are all shown in the design detail.

A. Proposed System

The majority of the clean files in the training database are executable and library files from several well-known programs, as well as system files from various operating system versions. In order to better train and test the system, we also employ clean files that are packed, or that share the same form or geometrical characteristics with malware files (e.g., use the same packer). The training dataset's malware files were extracted from the Virus Share collection. Malware files from the dataset collection and clean files from other operating systems are included in the test dataset (other files that the ones used in the initial database). A sizable dataset was utilised to assess the machine learning algorithms' capacity to scale up.

B. System Architecture

The process of architectural configuration involves constructing a foundational structure for a framework. It entails identifying the actual components of the framework and the interactions amongst these pieces. Construction modelling outline is the first configuration process that identifies these subsystems and establishes a framework for subsystem control and communication. The output of this process is a representation of the product structural planning. The system's suggested architecture is shown below. It illustrates the architecture of the system and its basic operation.

C. Data Flow Diagrams

The functions, or processes, that collect, process, store, and disseminate data between a system and its surroundings as well as amongst system components are visually represented by DFD. It is an effective communication tool between the system designer and the user because of the visual depiction. The structure of DFD enables one to start with a high-level overview and work their way down to a hierarchy of in-depth diagrams. DFD is frequently utilised for the following reasons.:

- Logical information flow of the system
- Determination of physical system construction requirements
- Simplicity of notation
- Establishment of manual and automated systems requirements

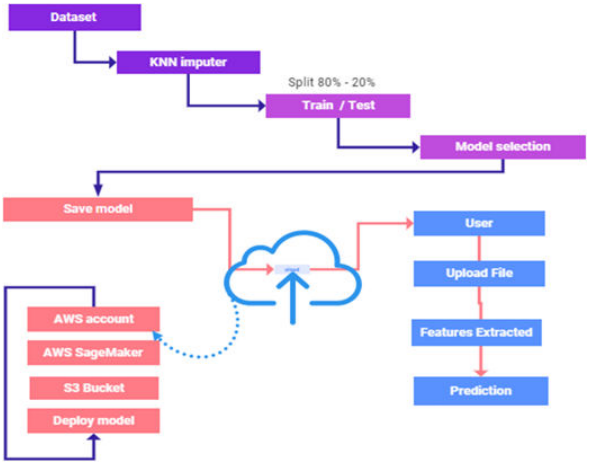

Fig. 3. System Architecture Model

DFD Components - DFD can represent Source, destination, storage and flow of data using the following set of components.
Entities - An external entity is a person, department, outside organization, or other information system that provides data to the system or receives outputs from the system.
Process - any process that changes the data, producing an output. It might perform computations, or sort data based on logic, or direct the data flow based on business rules.
Data Storage - files or repositories that hold information for later use, such as a database table or a membership form. Each data store receives a simple label, such as "Orders."
Data Flow - the route that data takes between the external entities, processes and data stores.

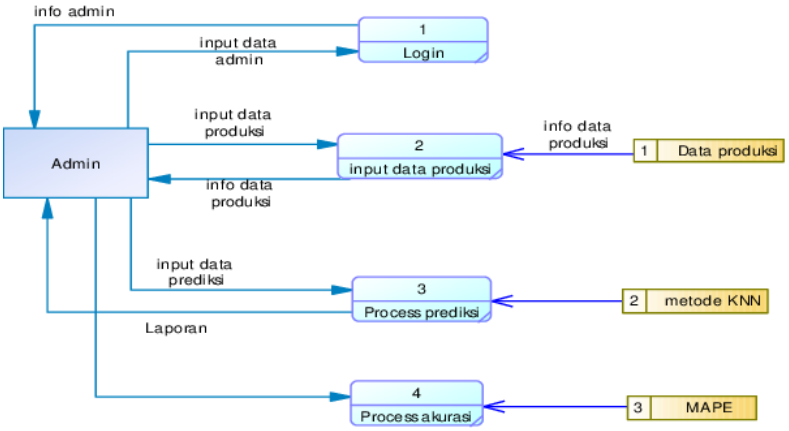

Fig. 4. Data Flow Diagram – L0

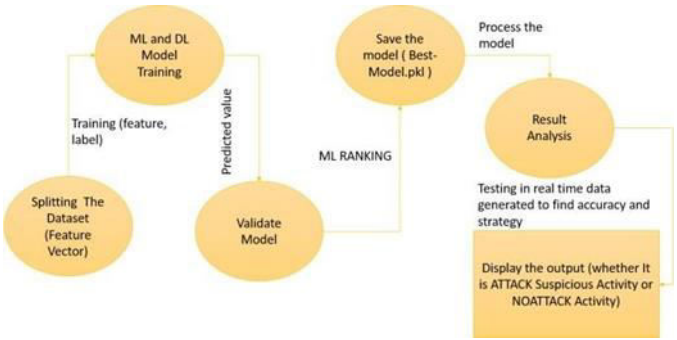

Fig. 5. Data Flow Diagram- L1

D. Use Case Diagrams

The goal of a use case diagram is to illustrate the dynamic nature of a system. However, this explanation is too generic to describe it, as the activity, sequence, collaboration, and State chart are the other four diagrams that serve the same purpose. We'll look into a specific feature that distinguishes it from the other four diagrams.

Use case diagrams are employed to compile a system's needs, taking into account both internal and external factors. The majority of these specifications are design-related. Therefore, use cases are created and actors are discovered during the process of gathering a system's functions through analysis.
Use case diagrams are designed to show the outside perspective when the first job is finished.
In brief, the purposes of use case diagrams can be said to be as follows –

- Used to gather the requirements of a system.
- Used to get an outside view of a system.
- Identify the external and internal factors influencing the system.
- Show the interaction among the requirements are actors.

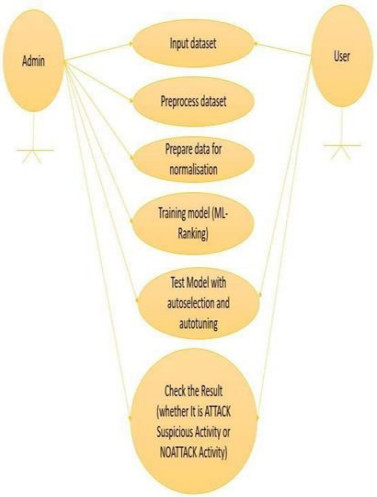

Fig. 6. Use case Diagram of the Model

E. Sequence Diagram

A sequence diagram, also known as an interaction diagram, illustrates the order in which processes interact with one another. It is a message sequence chart construct. An object's interactions are arranged chronologically in a sequence diagram. It shows the classes and objects that are a part of the scenario as well as the messages that are sent between the objects in order for the scenario to work. Sequence diagrams are often referred to as event scenarios or event diagrams.

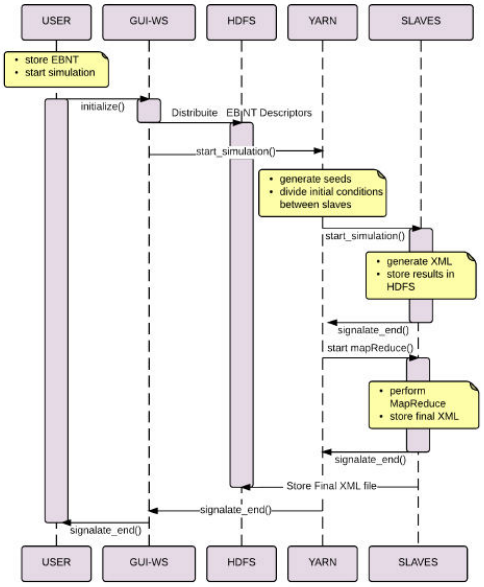

Fig. 7. Sequence Diagram of the Model

6. Modules

The fundamental step in transitioning from the issue space to the course of action space is this one. Thus, the diagram leads us to work towards how to fully fulfil those needs by beginning with what is required. The system map displays every important data structure, record path, yield, and actual modules within the structure together with the selected

specification. This presupposes a crucial role since it will provide the final yield that it was operating on. In our work we are using some modules, these modules are listed below.

Gathering Data: Once you are clear about your goals and have the necessary tools, you may move on to the first proper stage of machine learning: data collection. This is a very important phase since the quality and quantity of data collected will directly impact the prediction model's accuracy. Following collection, the data is summarised and referred to as training data.

The virus-share dataset: Security researchers, incident responders, forensic analysts, and the morbidly curious may obtain samples of live malicious code through the VirusShare dataset, which is a library of malware samples.

Data Preparation: Following the collection of training data, the next stage of machine learning is data preparation, which involves loading the data into the appropriate location and getting it ready for use in machine learning training. Since the sequence of the data shouldn't have an impact on what is learnt, it is first combined all together and then randomly arranged. This is also an excellent opportunity to visualise the data, as that can assist you in determining whether the various variables have any meaningful links, how to best utilise them, and whether there are any imbalances in the data. Furthermore, the data must now be divided into two sections. The bulk of the dataset will be utilised in the first step to train our model, and the second half will be used to assess how well the trained model performs. At this stage, additional adjustments and manipulations like mistake correction, normalisation, and more are made.

Choosing a Model: Selecting a model among the numerous that researchers and data scientists have developed over the years is the next stage in the workflow. Select the appropriate option that will complete the task.

Training: Following the completion of the previous phases, the training phase begins, during which the data is utilised to gradually increase the model's predictive capacity. This phase is frequently regarded as the majority of machine learning. During the training phase, we initialise some random values for our model's A and B, for example. We then forecast the output using those values, compare it to the model's prediction, and then modify the values to match the earlier predictions. Then, this procedure is repeated, with each update cycle denoted as a single training step.

Evaluation: After training is finished, you use this phase to determine whether it is sufficient. This is when the dataset you previously placed aside becomes useful. Evaluation is designed to be reflective of the model's potential performance in the actual world by enabling testing of the model against never-before-seen training data.

Parameter Tuning: After the evaluation, you may adjust the parameters to see whether your training can be made even better. When the training was completed, a few criteria were taken for granted. The learning rate is an additional parameter that determines the amount that the line is moved in each step, taking into account the data from the preceding training stage. The precision of the training model and the duration of the training are both influenced by these factors. When dealing with more complicated models, the training outcome is heavily influenced by the beginning conditions. Variations may be observed based on whether a model is trained using a distribution of values or with values initialised to zero, which raises the issue of which distribution should be utilised. Determining what constitutes a good model is crucial since there are a lot of factors to take into account at this stage of training. Hyper parameters are the name given to these parameters. The model, training procedure, and dataset all influence how these parameters are tuned or adjusted. You can proceed to the last stage when you have completed and are pleased with these settings.

Prediction: In essence, machine learning uses data to provide answers to enquiries. This is the last phase, where you will be asked a few questions to answer. At this juncture, the true worth of machine learning becomes apparent. This is the final opportunity for you to utilise your model to forecast the desired result. The aforementioned procedures serve as a learning route since they lead you from the point of model creation to the point of output prediction.

7. **Methodology**

Random Forest Algorithm: An approach for supervised machine learning based on ensemble learning is called random forest. In ensemble learning, distinct algorithm types or iterations of the same algorithm are combined to create a more potent prediction model. The name "Random Forest" comes from the random forest algorithm, which mixes several algorithms of the same sort, i.e., several decision trees, to create a forest of trees. The random forest technique is applicable to applications involving both classification and regression.

The following are the basic steps involved in performing the random forest algorithm:

1. Pick N random records from the dataset.
2. Build a decision tree based on these N records.
3. Choose the number of trees you want in your algorithm and repeat steps 1 and 2.
4. In a regression issue, every tree in the forest predicts a value for Y (output) for a new record. The average of all the values that each tree in the forest projected may be used to get the final value. Alternatively, every tree in the forest

forecasts the category to which the new record belongs in the event of a classification challenge. Ultimately, the category with the majority vote is given the new record.

Gaussian Naïve Bayes: Continuous valued features are supported by Gaussian Naive Bayes, which models each as adhering to a Gaussian (normal) distribution.
Assuming that the data is characterised by a Gaussian distribution with no covariance (independent dimensions) between dimensions is one way to build a basic model. All that is required to establish such a distribution is determining the mean and standard deviation of the points within each label, which makes it easy to fit this model.

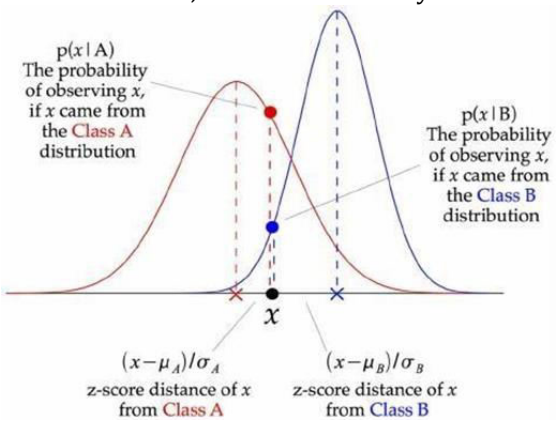

Fig. 8. The above illustration indicates how a Gaussian Naive Bayes (GNB) classifier works.

Each data point's z-score distance—that is, the distance from the class mean divided by the class standard deviation—is computed in relation to each class mean. The likelihood that event A will occur given that event B has already happened is known as the conditional probability of event A given event B. In mathematical notation, P[A|B] = P[A AND B] / P[B] represents the conditional probability of A given B.

Decision Tree Algorithm: The decision tree algorithm is a member of the supervised learning algorithm family. The decision tree approach, in contrast to other supervised learning algorithms, is also capable of handling regression and classification issues. By learning straightforward decision rules from historical data (training data), a Decision Tree may be used to develop a training model that can be used to predict the class or value of the target variable. With decision trees, we begin at the root of the tree when attempting to forecast a record's class label. We contrast the root attribute's values with those of the record's attribute. We follow the branch that corresponds to that value and go on to the next node based on the comparison.

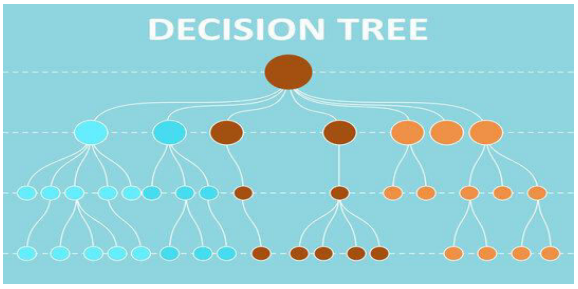

Fig. 9. The above illustration indicates how a Decision Tree Classifier Works

Working procedure: The correctness of a tree is greatly impacted by the choice to make strategic splits. Regression trees and classification trees have various decision criteria. To determine whether to divide a node into two or more sub-nodes, decision trees employ a variety of techniques. The homogeneity of the resulting sub-nodes is increased by the development of sub-nodes. Stated differently, we might assert that the node's purity rises in relation to the target variable. The decision tree finds the split that produces the greatest homogenous sub-nodes after splitting the nodes according to all given characteristics.

8. **Conclusion and future work**

In conclusion, we have put forth a solution that combines cloud computing environments with malware detection systems. All malware and binaries in use are intercepted by sending them to one or more analysis engines, which do a thorough check against a signature database in order to find malware or exploits that have not yet been discovered. As more and more people turn to cloud computing platforms for their computing requirements, we will advocate for a greater reliance on cloud computing. To establish the optimal method for cloud-based malware detection, we examined prior research on both traditional and storage-aware malware detection in this study. Additionally, we make the case for the advantages of dispersing detection throughout the cloud and offer a fresh method for coordinating detection throughout the cloud. Traditional detection methods (optimising patterns) based on static signatures and dynamic detection technologies

(heuristics) are employed in the suggested system. Next, in order to compete with current antivirus software, we have selected faster, more advanced, and safer system techniques. The goal of this effort is to identify the best remedies for anti-virus issues, enhance performance, and identify potential substitutes for problematic working environments that are highly effective and adaptable. We employed the best contemporary and conventional techniques to identify viruses, including heuristics and signatures to identify known and undiscovered viruses. As contrast to signature-based detection, future work in this area will concentrate on the development of detection systems based on memory introspection and heuristic or statistical detection.

9. **References**

[1] Steve Morgan, "cybersecurity almanac: 100 facts, figures, predictions and statistics," Cybercrime Magazine Cisco and Cybersecurity Ventures, 2019.
[2] ÖmerAslan, RefikSamet, and ÖmerÖzgürTanrıöver, "Using a Subtractive CenterBehavioral Model to Detect Malware," Security and Communication Networks 2020, 2020.
[3] Ajeet Singh and Anurag Jain, "Study of cyber-attacks on cyber- physical system," In Proceedings of 3rd International Conference on Internet of Things and Connected Technologies (ICIoTCT). 26–27, 2018.
[4] R Samani and G Davis, "McAfee Mobile Threat Report Q1," 2019. https://www.mcafee.com/enterprise/en-us/assets/reports/rpmobile- threat-report-2019.pdf.
[5] ÖmerAslan and RefikSamet, "A comprehensive review on malware detection approaches," IEEE Access 8, 6249–6271, 2020.
[6] Hao Sun, Xiaofeng Wang, RajkumarBuyya, and Jinshu Su, "CloudEyes: Cloud-based malware detection with reversible sketch for resource-constrained internet of things (IoT) devices," Software: Practice and Experience 47(3), 421–441, 2017.
[7] Deepti Gupta, Smriti Bhatt, Maanak Gupta, OlumideKayode, and Ali SamanTosun, "Access control model for google cloud iot. In 2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity)," IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS). IEEE, 198–208, 2020.
[8] J. Oberheide, E. Cooke, and F. Jahanian "CloudAV: N- Version Antivirus in the Network Cloud", In Proceedings of the 17th USENIX Security Symposium (Security'08). San Jose, CA, 2008.
[9] Jon Oberheide, Evan Cooke and FarnamJahanian "Cloud N- Version Antivirus in the Network Cloud",Electrical Engineering and Computer Science Department, University of Michigan, Ann Arbor, MI 48109 (2007).
[10] Matthias Schmidt, Lars BaumgArtner, Pablo Graubner, David Bock and Bernd Freisleben "Malware Detection and Kernel Rootkit Prevention in Cloud Computing Environments." University of Marburg, Germany (2011).
[11] K. Murad, S. Shirazi, Y. Zikria, and I. Nassar, "Evading Virus Detection Using Code Obfuscation" in Future Generation Information Technology, vol. 6485 of Lecture Notes in Computer Science, pp. 394–401, Springer Berlin , Heidelberg, 2010.
[12] Scott Treadwell, Mian Zhou "A Heuristic Approach for Detection of Obfuscated Malware", Bank of America, 1201 Main St, Dallas, TX 75202, © IEEE 2009.
[13] Carlin, S., & Curran, K. "Cloud computing security", International Journal of Ambient Computing and Intelligence.
[14] "Heuristic analysis in Kaspersky Internet Security" [Online]: http://support.kaspersky.com , ID: 8936 , 2013 Mar 01 2013
[15] AlgirdasAvizienis, "The n-version approach to fault- tolerant software", IEEE Transactions on Software Engineering, 1985.
[16] Rodrigo Rodrigues, Miguel Castro, and Barbara Liskov. Base,
"using abstraction to improve fault tolerance", In Proceedings of the eighteenth ACM symposium on Operating systems principles, New York, NY, USA, 2001.
[17] Lajos Nagy, Richard Ford, and William Allen, "N- version programming for the detection of zero-day exploits", In IEEE Topical Conference on Cybersecurity, Daytona Beach, Florida, USA, 2006.
[18] CarstenWillems and Thorsten Holz.Cwsandbox.[Online]: http://www.cwsandbox.org/, 2007.
[19] HispasecSistemas. "Virus total", [Online]: http://virustotal.com, 2004.
[20] Norman Solutions. Norman sandboxwhitepaper.
http://download.norman.no/whitepapers/whitepaper_Norman_ SandBox.pdf, 2003.
[21] Barracuda Networks. "Barracuda spam firewall",[Online]: http://www.barracudanetworks.com, 2007.
[22] Cloudmark, "Cloudmark authority anti- virus",[Online]:http://www.cloudmark.com, 2007.
[23] Alexander Moshchuk, Tanya Bragin, Damien Deville, Steven D. Gribble, and Henry M. Levy, "Spyproxy: Execution-based detection of malicious web content", In Proceedings of the 16th USENIX Security Symposium, August 2007.
[24] SteliosSidiroglou, AngelosStavrou, and Angelos D. Keromytis, "Mediated overlay services (moses): Network security as a composable service", In Proceedings of the IEEE Sarnoff Symposium