**Anomaly Detection in Cloud network Using Machine Learning Techniques**

**Abstract**

The popularity and usage of Cloud computing is increasing rapidly. Several companies are  
investing in this field either for their own use or to provide it as a service for others. One of the results of Cloud development is the emergence of various security problems for both industry and consumer. One of the ways to secure Cloud is by using Machine Learning (ML). ML techniques have been used in various ways to prevent or detect attacks and security gaps on the Cloud.

We have defined 11 Cloud security areas. Moreover, distributed denial-of-service (DDoS), Probe, R2L, U2R and data privacy are the most common security areas. On the other hand, we found 30 ML techniques used, some used hybrid and others as standalone. The most popular ML used is SVM in both hybrid and standalone models. Furthermore, 60% of the papers compared their models with other models to prove the efficiency of their proposed model. Moreover, 13 different evaluation metrics were enumerated.

Additionally, we do make attack with its features and name of the attack, which is been stored in the cloud. Later we apply various machine learning algorithms to compare and get the performance. The algorithms we can use were SVM, KNN, Naïve Bayes, Decision Tree etc.

The most applied metric is true positive rate and least used is training time. Lastly, from 20 datasets found, KDD and KDD CUP’99 are the most used among relevant studies.

**Chapter 1**

Introduction

**1.1 Background Study**

Although Cloud computing is seen as a significant and positive IT infrastructure shift, much security work is still needed to minimize its deficiencies. Since a significant amount of personal and corporate information is placed in the Cloud data centers, those Cloud security issues and vulnerabilities need to be identified and prevented.

Cloud infrastructure runs through standard Internet protocols and uses virtualization techniques, it may be vulnerable to attacks. Those attacks may come from traditional sources  
such as Address Resolution Protocol, IP spoofing, Denial of Service (DoS) etc. They may also come from other sources. Zero-day attacks, for example, referred to as unknown attacks, are seen as a significant challenge in the cyber security domain. Traditional techniques used for detection and prevention are not efficient enough to handle those attacks while also dealing with a large data flow.

In the world of rapidly developing technology, networks are facing threats like viruses, worms, Trojan horses, spyware, adware, root kits, etc. These intrusions need to be identified before any type of loss to the organizations. Even internal Local Area Network (LAN) is also seriously struggling with intrusions. This is affecting productivity of computer networks in terms of bandwidth and other resources. Hackers use advance features like dynamic ports, IP address spoofing, encrypted payload etc., to avoid detection. This type of intrusions can be detected by discovering patterns in network traffic dataset.

Soft computing based IDS embraces several computational intelligence methodologies, including artificial neural networks, fuzzy logic, evolutionary computation, probabilistic computing, artificial immune systems, belief networks etc.

Table below shows the Network Traffic Attacks

|  |  |
| --- | --- |
| Attack group | Attacks |
| Probe | ipsweep, mscan, nmap, portsweep, saint, satan |
| DoS | apache2, back, land, mailbomb, Neptune, processtable, pod, udpstorm, smurf, teardrop |
| U2R | buffer\_overflow, httptuneel, loadmodule, perl, rootkit, xterm, ps, sqlattack |
| R2L | ftp\_write, imap, guess\_passwd, named, multihop, phf, sendmail, snmpgetattack, snmpguess, spy, warezclient, worm, warezmaster, zsnoop, xlock |

**Driving Force behind the idea**

Due to huge and imbalanced dataset machine learning based Intrusion Detection System (IDS) faces problem to process entire data. So, it is necessary to identify intrusions through network traffic behavior. IDS is designed to defend the network from malicious activities. Anomaly based IDS learn normal behavior from network traffic dataset to detect attacks.

**1.2 Objectives**

**The objectives set as follows**

* + To detect security issues and vulnerabilities.
  + To find a novel supervised machine learning system is developed to classify network traffic whether it is malicious or not.

**1.3 Problem Statement**

Considering anomaly pattern detection asdetecting a point in time where the behavior of the system isunusual and significantly different from past behavior. Inthis context, anomaly (pattern) detection mean detecting the behaviors that deviate fromnormal behaviors.

**1.4 Existing System**

In the Existing, Traditional techniques used for detection and prevention are not efficient enough to handle those attacks while also dealing with a large data flow.

The man drawback we found as less accuracy and not good efficiency to find the attacks in the data flow in cloud networks.

**Drawbacks**

The man drawback we found as less accuracy and not good efficiency to find the attacks in the data flow in cloud networks

**1.5 Proposed System**

We proposed a systematic review of the ML techniques used to solve, detect or prevent Cloud security issues and vulnerabilities.

In the proposed work, we make an attack by adding the features and its attack name. we do classify the attack name and its category (R2L, U2R, DDOS, PROBE)

Machine Learning (ML) techniques are very helpful for identifying attacks, compared to traditional or zero-day attacks

**Advantages**

Machine Learning (ML) techniques are very helpful for identifying attacks, compared to traditional or zero-day attacks.

**Chapter 2**

**LITERATURE SURVEY**

**2.1 Introduction**

A literature review is a type of academic writing that provides an overview of existing knowledge in a particular field of research. A good literature review summarises, analyses, evaluates and synthesises the relevant literature within a particular field of research.

**2.2 Related Work**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | TITLE | AUTHOR | OBSERVATIONS | REFRENCES |
| 1 | Network Intrusion Detection using Supervised Machine Learning Technique with Feature Selection | Kazi Abu Taher, Billal Mohammed Yasin Jisan, Md. Mahbubur Rahman | In this study, it is found that Artificial Neural Network (ANN) based machine learning with wrapper feature selection outperform Support Vector Machine (SVM) technique while classifying network traffic. To evaluate the performance, NSL-KDD dataset is used to classify network traffic using SVM and ANN supervised machine learning techniques. | 2019 IEEE |
| 2 | Network Intrusion Detection Using Hybrid Machine Learning Model | A K M Mashuqur Rahman Mazumder, Niton Mohammed Kamruzzaman, Nasrin Akter, Nafija Arbe and Md Mahbubur Rahman | In this paper, a hybrid machine learning model with a new feature selection method is proposed for better performance of the Intrusion Detection System. In this proposed model, the Intrusion Detection System is built with a combination of supervised and unsupervised machine learning models. A brief comparison between the proposed model and the other machine learning models such as AdaBoost, XGBoost, Random Forest, Gaussian Naive Bayes, LGB is narrated in this paper | 2021 IEEE |
| 3 | Network intrusion detection system using supervised learning paradigm | J. Olamantanmi Mebawondu, Olufunso D. Alowolodu, Jacob O. Mebawondu, Adebayo O. Adetunmbi | In this paper presents a light weight IDS based on information gain and Multi-layer perceptron Neural Network. Gain ratio was used in selecting relevant features for attack and normal traffic prior classification using Neural Network. Empirical results from the UNSW-NB15 intrusion detection dataset on thirty selected attributes is a highly ranked decision, thus, the light weight IDS is suitable for real time intrusion detection | 2020Elsevier |
| 4 | Packet Sniffing and Sniffing Detection | Ruchi Tuli | This paper discusses the basic working of a packet sniffer, network protocols that are vulnerable to sniffing, various software that can be used for sniffing. This paper also describes possible defensive techniques used to defend against sniffing attacks. Finally the papers ends with describing some sniffing detection techniques. Sniffers are not hacking tools but they can help a hacker to launch further attacks such as session hijacking, DOS attacks, MITM attacks etc | 2020 IJIET |
| 5 | A new method for flow-based network intrusion detection using the inverse Potts model | Camila Pontes, Manuela Souza, João Gondim, Matt Bishop and Marcelo Marotta | Authors propose a new algorithm, called Energy-based Flow Classifier (EFC). This anomaly-based classifier uses inverse statistics to infer a statistical model based on labeled benign examples. We show that EFC is capable of accurately performing binary flow classification and is more adaptable to different data distributions than classical ML-based classifiers. Given the positive results obtained on three different datasets (CIDDS-001, CICIDS17 and CICDDoS19), we consider EFC to be a promising algorithm to perform robust flow-based traffic classification. | 2021, IEEE |
| 6. | Challenges In Cloud Anomaly Detection Using  Machine Learning Approaches | Prathibha S, Vinay S | This paper provides details about anomaly  detection techniques, models or frameworks, their merits,  and demerits so that one can easily decide their  applicability in different circumstances. | 2022,IEEE |
| 7. | Cloud Based Intrusion Detection Approach Using Machine Learning Techniques | Hanaa Attou, Azidine Guezzaz\_, Said Benkirane, Mourade Azrour, and Yousef Farhaoui | This paper introduces a cloud-based intrusion detection model utilizing random forest (RF) and feature engineering to enhance accuracy. Results indicate superior performance in accuracy, precision, and recall compared to recent related works.this helped us narrow down on usaing SVM and RF for our model. | 2023,IEEE |
| 8. | Machine learning Approach for Anomaly-based Intrusion  Detection Systems using Isolation Forest Model and  Support Vector Machine | K.Shanthi, R.Maruthi | This paper discusses anomaly-based intrusion detection systems utilizing machine learning algorithms, particularly focusing on the isolation forest model and Support Vector Machine (SVM). The isolation forest algorithm is employed to detect anomalies by recursively partitioning the data using decision trees, while SVM is applied for binary classification to identify outliers. Feature reduction techniques like Principal Component Analysis (PCA) and K-means clustering are utilized alongside SVM to enhance classification performance in detecting anomalous network traffic. | 2023,IEEE |
| 9. | Machine Learning and Deep Learning based Intrusion Detection in Cloud Environment: A Review | A.Vinolia, Dr N.Kanya, Dr V.N.Rajavarman | This research extensively explores various classification approaches for intrusion detection, focusing on recent studies from 2019 to 2022 across multiple disciplines. It highlights the design and capabilities of deep learning and machine learning network types, analyzing the effectiveness of intrusion detection systems based on accuracy and outlining areas for further research to enhance accuracy in intrusion detection. | 2023,IEEE |
| 10. | A Machine Learning-Based Approach for Anomaly  Detection for Secure Cloud Computing Environments | Priya Parameswarappa, Taral Shah, Govinda Rajulu,Lanke. | This research presents a comprehensive methodology for intrusion detection in cloud computing, employing deep learning and machine learning techniques. It utilizes the UNSW-NB15 Dataset for data collection and applies preprocessing techniques like handling missing values. Feature selection is conducted using the K-best Feature Selection (K-FS) method, followed by classification employing various machine learning algorithms such as Logistic Regression, K-Nearest Neighbor, Decision Trees, Extra Trees Classifier, Random Forest, Gradient Boosting, and Multilayer Perceptron layers. | 2023,IEEE |
| 11. | A Cloud-Based Intrusion Detection System for Advanced Threat Detection and Prevention using Machine Learning Techniques | Ravella Abhinav, Kunduru Nikunj Raghav, Singadi Shanthan Reddy, Penugonda Sai Naga Koushik, Senthil Kumar Thangavel, Karthik Srinivasan | This study proposes a hybrid Intrusion Detection System (IDS) combining Isolation Forest (IF) and Feed Forward Neural Network (FFNN) models to address the limitations of signature-based and anomaly-based IDS. By leveraging the strengths of each model and employing a stacking strategy, the hybrid IDS achieves significantly improved accuracy and lower false positive rates compared to individual models. The evaluation results demonstrate its effectiveness in detecting both known and unknown threats, making it a promising solution for modern network security challenges. | 2023,IEEE |

**Chapter 3**

**System Requirements**

Software requirement Specification is a fundamental document, which forms the foundation of the software development process. It not only lists the requirements of a system but also has a description of its major feature. An SRS is basically an organization's understanding (in writing) of a customer or potential client's system requirements and dependencies at a particular point in time (usually) prior to any actual design or development work. It's a two-way insurance policy that assures that both the client and the organization understand the other's requirements from that perspective at a given point in time. The SRS also functions as a blueprint for completing a project with as little cost growth as possible. The SRS is often referred to as the "parent" document because all subsequent project management documents, such as design specifications, statements of work, software architecture specifications, testing and validation plans, and documentation plans, are related to it. It is important to note that an SRS contains functional and nonfunctional requirements only; it doesn't offer design suggestions, possible solutions to technology or business issues, or any other information other than what the development team understands the customer's system requirements to be.

**3.1 Functional Requirements**

Functional Requirement defines a function of a software system and how the system must behave when presented with specific inputs or conditions. These may include calculations, data manipulation and processing and other specific functionality. In this system following are the functional requirements:-

1. Load the dataset.
2. Add Attack patterns to the Model.
3. Train Using ML.
4. Identify the DDoS Detection for unknown traffics.

**3.2 Non-functional Requirements**

Non functional requirements are the requirements which are not directly concerned with the specific function delivered by the system. They specify the criteria that can be used to judge the operation of a system rather than specific behaviors. They may relate to emergent system properties such as reliability, response time and store occupancy. Non functional requirements arise through the user needs, because of budget constraints, organizational policies, the need for interoperability with other software and hardware systems or because of external factors such as:-

1. Product Requirements 2. Organizational Requirements 3. User Requirements 4. Basic Operational Requirements

**3.2.1 Product Requirements**

**Portability:** Since the software is developed in python it can be executed on any platform for which the Python is available with minor or no modifications.

**Correctness:** It followed a well-defined set of procedures and rules to compute and also rigorous testing is performed to confirm the correctness of the data.

**Ease of Use:** The front end is designed in such a way that it provides an interface which allows the user to interact in an easy manner.

**Modularity:** The complete product is broken up into many modules and well-defined interfaces are developed to explore the benefit of flexibility of the product.

**Robustness:** This software is being developed in such a way that the over all performance is optimized and the user can expect the results within a limited time with utmost relevancy and correctness. Python itself possesses the feature of robustness, which implies the failure of the system is negligible.

Non functional requirements are also called the qualities of a system. These qualities can be divided into execution quality & evolution quality. Execution qualities are security & usability of the system which are observed during run time, whereas evolution quality involves testability, maintainability, extensibility or scalability.

**3.2.2 Organizational Requirements**

**Process Standards:** IEEE standards are used to develop the application which is the standard used by the most of the standard software developers all over the world.

**Design Methods:** Design is one of the important stages in the software engineering process. This stage is the first step in moving from problem to the solution domain. In other words, starting with what is needed design takes us to work how to satisfy the needs.

The design of the system is perhaps the most critical factor affecting the quality of the software and has a major impact on the later phases, particularly testing and maintenance. We have to design the product with the standards which has been understood by the developers of the team.

**3.2.3 User Requirements**

## The user must be able to visualize Graphical User Interface Window.

## The user must be able to configure all the parameters.

**3.2.4 Basic Operational Requirements**

The customers are those that perform the eight primary functions of systems engineering, with special emphasis on the operator as the key customer. Operational requirements will define the basic need and, at a minimum, will be related to these following points:-

**Mission profile or scenario:** It describes about the procedures used to accomplish mission objective. It also finds out the effectiveness or efficiency of the system.

**Performance and related parameters:** It points out the critical system parameters to accomplish the mission

**Utilization environments:** It gives a brief outline of system usage. Finds out appropriate environments for effective system operation.

**Operational life cycle:** It defines the system lifetime.

* 1. **Resource Requirement**

**SPYDER**

Spyder, the Scientific Python Development Environment, is a free integrated development environment (IDE) that is included with Anaconda. It includes editing, interactive testing, debugging, and introspection features.After you have installed Anaconda, start Spyder on Windows, macOS, or Linux by running the command spyder.Spyder is also pre-installed in [Anaconda Navigator](https://docs.anaconda.com/anaconda/navigator/), which is included in Anaconda. On the Navigator **Home** tab, click the Spyder icon.For more information about Spyder, see the [Spyder web page](https://www.spyder-ide.org/) or the [Spyder documentation](https://docs.spyder-ide.org/).

**Anaconda** command **prompt** is just like command **prompt**, but it makes sure that you are able to use **anaconda** and conda commands from the **prompt**, without having to change directories or your path.These locations contain commands and scripts that you can run.

**Python Programming**

[Python](https://www.geeksforgeeks.org/python-programming-language/) is a widely used general-purpose, high level programming language. It was created by Guido van Rossum in 1991 and further developed by the Python Software Foundation. It was designed with an emphasis on code readability, and its syntax allows programmers to express their concepts in fewer lines of code.

Python is a programming language that lets you work quickly and integrate systems more efficiently.

**1) Finding an Interpreter:**

Before we start Python programming, we need to have an interpreter to interpret and run our programs. There are certain online interpreters like [**https://ide.geeksforgeeks.org/**](https://ide.geeksforgeeks.org/), http://ideone.com/ or http://codepad.org/ that can be used to run Python programs without installing an interpreter.

***Windows*:**There are many interpreters available freely to run Python scripts like IDLE (Integrated Development Environment) that comes bundled with the Python software.

**3.4 Hardware Requirements**

We need 1 machine with following minimal requirements

CPU : Intel 2.1 GHZ

Memory : 4GB

Disk : 40GB

Display : 15-inch color

**3.5 Software (Tools &Technologies) Requirements**

The proposed solution will be implemented in python

Coding : Python

Platform : Python 3.7 and above

Tool : Anaconda Spyder / Jupyter Notebook

Libraries : pandas, numpy, sklearn etc

**Chapter 4**

**SYSTEM ANALYSIS AND DESIGN**

**4.1 System Analysis**

Analysis is the process of finding the best solution to the problem. System analysis is the process by which we learn about the existing problems, define objects and requirements and evaluates the solutions. It is the way of thinking about the organization and the problem it involves, a set of technologies that helps in solving these problems. Feasibility study plays an important role in system analysis which gives the target for design and development.

**4.1.1 Feasibility Study**

Depending on the results of the initial investigation the survey is now expanded to a more detailed feasibility study. “FEASIBILTY STUDY” is a test of system proposal according to its workability, impact of the organization, ability to meet needs and effective use of the resources.

Eight steps involved in the feasibility analysis are:

✓Form a project team and appoint a project leader.

✓Enumerate potential proposed system.

✓Define and identify characteristics of proposed system.

✓Determine and evaluate performance and cost effective of each proposed system.

✓Weight system performance and cost data.

✓Select the best proposed system.

✓Prepare and report final project directive to management.

Three key considerations involved in the feasibility analysis are

✓ECONOMICAL FEASIBILITY

✓TECHNICAL FEASIBILITY

✓SOCIAL FEASIBILITY

**4.1.2 ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

**4.1.3 TECHNICAL FEASIBILITY**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**4.1.4 SOCIAL FEASIBILITY**

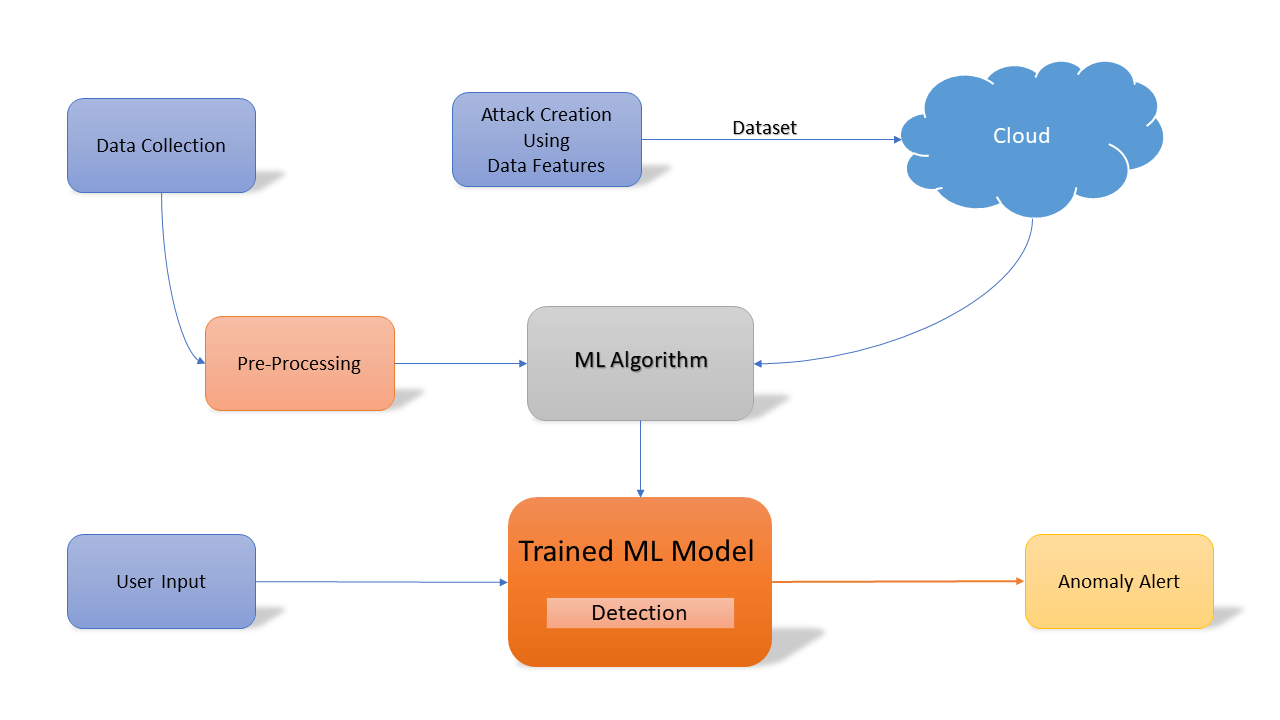
The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system

**4.2 System Design**

**4.2.1 System Architecture**

System architecture is the conceptual design that defines the [structure](http://en.wikipedia.org/wiki/Structure) and [behavior](http://en.wikipedia.org/wiki/Behavior) of a [system](http://en.wikipedia.org/wiki/System). An architecture description is a formal description of a system, organized in a way that supports reasoning about the structural properties of the system. It defines the [system](http://en.wikipedia.org/wiki/System) components or building blocks and provides a plan from which products can be procured, and systems developed, that will work together to implement the overall system.

The System architecture is shown below.



**Modules Used**

1. Data Collection
2. Data Pre-processing
3. Training with Machine Learning
4. Detection of DDoS Attacks

**IDS:** An intrusion detection system (IDS) is a software application that monitors network or **system** activities for malicious activities or policy violations and produces electronic reports to a management station.

Intrusion detection technique considers various issues such as huge of network traffic dataset, feature selection, low accuracy and high rate of false alarms.

**Preprocessing:** This module takes training data set as input, training data set is helpful for multi stage classifier modelling.

**Training and Detection of Attacks:** This module will train the dataset with machine learning model and classify the DDoS attacks for the unknown traffics.

**4.2.2 Use case Diagram of the system**

A use case diagram is a type of behavioral diagram created from a [Use-case analysis](http://en.wikipedia.org/wiki/Use-case_analysis). Its purpose is to present a graphical overview of the functionality provided by a system in terms of [actors](http://en.wikipedia.org/wiki/Actor_(UML)), their goals (represented as [use cases](http://en.wikipedia.org/wiki/Use_case)), and any dependencies between those use cases.



**4.2.3 Sequence Diagram**



**4.2.3 Data Flow Diagram of the system**

An information stream outline (DFD) is a graphical representation of the "stream" of information through a data framework. DFDs can likewise be utilized for the perception of information handling (organized outline). On a DFD, information things stream from an outside information source or an inner information store to an interior information store or an outer information sink, through an inward process.

**Level 0 Data stream chart**

A connection level or level 0 information stream chart demonstrates the collaboration between the framework and outside specialists which go about as information sources and information sinks. On the connection chart (otherwise called the Level 0 DFD) the framework's associations with the outside world are displayed simply as far as information streams over the framework limit. The connection chart demonstrates the whole framework as a solitary process, and gives no pieces of information as to its inward association.



**Level 1 Data flow diagram**

The Level 1 DFD shows how the system is divided into sub-systems (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole. It also identifies internal data stores that must be present in order for the system to do its job, and shows the flow of data between the various parts of the system.



**Chapter 5**

**Expected Outcome**

We do compute the Intrusion detection on the system, with various type of attacks and find the category of the attacks.

We use AWS S3 buckets as storage, which is helping in real time adding the attack and detecting the intrusions.

We use various machine learning algorithms like SVM, KNN etc.

Performance we can measure in terms of accuracy, precision, recall and its confusion matrix etc.

**Applications**

A "network intrusion detection system (NIDS)" monitors traffic on a network looking for suspicious activity, which could be an attack or unauthorized activity.

A large NIDS server can be set up on a backbone network, to monitor all traffic; or smaller systems can be set up to monitor traffic for a particular server, switch, gateway, or router.

In addition to monitoring incoming and outgoing network traffic, a NIDS server can also scan system files looking for unauthorized activity and to maintain data and file integrity.

The NIDS server can also detect changes in the server core components.

In addition to traffic monitoring, a NIDS server can also scan server log files and look for suspicious traffic or usage patterns that match a typical network compromise or a remote hacking attempt.

The NIDS server can also server a proactive role instead of a protective or reactive function. Possible uses include scanning local firewalls or network servers for potential exploits, or for scanning live traffic to see what is actually going on.

Keep in mind that a NIDS server does not replace primary security such as firewalls, encryption, and other authentication methods.

The NIDS server is a backup network integrity device. Neither system (primary or security and NIDS server) should replace common precaution (building physical security, corporate security policy, etc.)

**Conclusion**

We carried out a systematic literature review to analyze ML techniques used in Cloud security. The review investigated relevant studies that answered 3 RQs; Cloud security area, type of ML techniques used, and the accuracy estimation of the ML model.

Our conclusions are summarized as follows:

• Firstly we should add the attack with its features and store it to the cloud.

• Later We should build the model and train with the respective algorithms.

• Then check with the performance with each Algorithm.

Furthermore, datasets have been used to evaluate models’ performance. From the  
20 datasets found, KDD and KDD CUP ’99 are the most used.

Another important observation is that most of the datasets used are relatively old such as KDD. Researchers are encouraged to use recent datasets such as CICIDS2017, CSE-CICIDS2018 and Kyoto 2006C for intrusion detection.

**References**

1. Taher, K.A., Jisan, B.M.Y., & Rahman, M.M. (2019). Network Intrusion Detection using Supervised Machine Learning Technique with Feature Selection. IEEE.
2. Rahman Mazumder, A.K.M., Kamruzzaman, N.M., Akter, N., Arbe, N., & Rahman, M.M. (2021). Network Intrusion Detection Using Hybrid Machine Learning Model. IEEE.
3. Mebawondu, J.O., Alowolodu, O.D., Mebawondu, J.O., & Adetunmbi, A.O. (2020). Network intrusion detection system using supervised learning paradigm. Elsevier.
4. Tuli, R. (2020). Packet Sniffing and Sniffing Detection. IJIET.
5. Pontes, C., Souza, M., Gondim, J., Bishop, M., & Marotta, M. (2021). A new method for flow-based network intrusion detection using the inverse Potts model. IEEE.
6. S, P., & S, V. (2022). Challenges In Cloud Anomaly Detection Using Machine Learning Approaches. IEEE.
7. Attou, H., Guezzaz, A., Benkirane, S., Azrour, M., & Farhaoui, Y. (2023). Cloud Based Intrusion Detection Approach Using Machine Learning Techniques. IEEE.
8. Shanthi, K., & Maruthi, R. (2023). Machine learning Approach for Anomaly-based Intrusion Detection Systems using Isolation Forest Model and Support Vector Machine. IEEE.
9. Vinolia, A., Kanya, N., & Rajavarman, V.N. (2023). Machine Learning and Deep Learning based Intrusion Detection in Cloud Environment: A Review. IEEE.

[10]. Parameswarappa, P., Shah, T., Rajulu, G., & Lanke. (2023). A Machine Learning-Based Approach for Anomaly Detection for Secure Cloud Computing Environments. IEEE.

[11].Abhinav, R., Raghav, K.N., Reddy, S.S., Koushik, P.S.N., Thangavel, S.K., & Srinivasan, K. (2023). A Cloud-Based Intrusion Detection System for Advanced Threat Detection and Prevention using Machine Learning Techniques. IEEE.