Indian Institute of Information Technology, Dharwad

Machine Learning based attack detection systems in cloud computing

Cloud Security Case Study – 1

By

Daulat Kumar Jha – 20BCS037

Under the Supervision of

Dr. Malay Kumar, Asst. Professor, CSE

Contents

1. Aim and Objectives

The main aim of a Machine Learning (ML) based attack detection system in cloud computing is to improve the security of cloud computing environments by detecting and mitigating potential cyberattacks in real-time. The objectives of such a project includes:

- Developing a comprehensive understanding of the cloud computing environment, including the various types of attacks that may occur.
- Collecting and analyzing large amounts of data generated by cloud services, including network traffic, system logs, and user behavior data, to identify patterns and anomalies that may indicate a potential attack.
- Designing and implementing ML algorithms and models that can learn from historical data and detect new attacks as they occur.
- Developing a system that can automatically respond to detected attacks, such as by isolating affected systems, blocking malicious traffic, or alerting security personnel.
- Evaluating the effectiveness of the system and continuously refining it based on feedback and new data.

2. Introduction

Machine learning-based attack detection systems are commonly used in cloud computing environments to detect and respond to security threats. These systems use machine learning algorithms to analyze data and identify patterns that indicate potential attacks.

There are several types of machine learning-based attack detection systems that can be used in cloud computing environments, including:

- The Distributed Denial of Service (DDoS) attack
- Malware injection attack
- Economical Denial of sustainability(EDoS) attack
- Side-channel attack
- Man-in-the-Middle (MITM) attack
- Wrapping attack
- Flooding attack

3. Identification of security concerns

The classic cloud service model consists of three layers of architectural construct:

- Infrastructure-as-a-Service (IaaS)
- Platform-as-a-Service (PaaS)
- Software-as-a-Service (SaaS)

Each layer is vulnerable to a myriad of associated, various security threats. The threats can be either of malicious intent of the Cloud User (CU) or the Cloud Service Provider (CSP) or can be a simple unexpected misconfiguration. These security threats can affect the four cloud security principles i.e., Confidentiality, Integrity, and Availability and Accountability.

3.1 Taxonomy of Security Threats

I. **Malware attacks:** Malware attacks are the most common security threats in cloud computing. Malware can spread across the cloud infrastructure, infecting multiple machines and disrupting cloud services. Ransomware can encrypt important files and data, demanding a ransom in exchange for the decryption key.

II. **Distributed Denial-of-Service (DDoS) attacks:** DDoS attacks are designed to overwhelm cloud infrastructure by generating a large volume of traffic. This type of attack can cause cloud services to become unavailable or slow, causing inconvenience to users.

III. **Insider threats:** Insider threats are security threats caused by authorized users with malicious intent or unintentional mistakes. This type of threat can cause data breaches, service disruption, and intellectual property theft.

IV. **Brute force attacks:** Brute force attacks are designed to gain access to cloud infrastructure by guessing passwords or encryption keys. This type of attack can be successful if the attacker is able to guess the correct password or key.

V. **Man-in-the-middle (MITM) attacks:** MITM attacks occur when an attacker intercepts communication between cloud services and users, allowing the attacker to eavesdrop or manipulate the communication.

VI. **Advanced Persistent Threats (APT's):** APT's are complex, targeted attacks that are designed to gain access to sensitive data or systems over an extended period of time. APT's can be difficult to detect and can cause significant damage to cloud infrastructure.

VII. **Social engineering attacks:** Social engineering attacks are designed to exploit human behavior to gain access to cloud infrastructure. This type of attack can include phishing, pretexting, and baiting.

4. Related Work

There is a vast amount of literature on how Machine Learning enabled security mechanisms could contribute to cloud security. We categorize our literature review based on the below-mentioned key aspects:

4.1. A Fast KNN Based Intrusion Detection System For Cloud Environment

The term paper proposes a K-nearest neighbor (KNN) based interruption location framework (IDS) for cloud situations. The objective of these IDS is to distinguish different sorts of assaults on cloud administrations in a quick and exact way.

The proposed IDS uses2 a subset of the foremost important highlights to diminish the highlight space dimensionality and progress the location precision. The highlight choice preparation is based on the relationship between highlights and the lesson name (i.e., normal or assault). This guarantees that, as it were, the foremost significant highlights are considered amid the location preparation, which diminishes the computational complexity and moves forward the general execution of the framework.

The proposed framework is assessed utilizing the NSL-KDD dataset, which may be a broadly utilized benchmark dataset for interruption discovery frameworks. The dataset contains different sorts of assaults on arrange administrations, counting DoS, Test, R2L, and U2R assaults. The proposed framework accomplishes a discovery exactness of 99.85%, which is altogether higher than other state-of-the-art strategies. Furthermore, the untrue positive rate of the proposed framework is 0.002%, which demonstrates that the framework can precisely recognize between typical and assault activity.

Generally, it appears that the proposed KNN-based IDS is viable in recognizing diverse sorts of assaults in cloud situations and outflanks other state-of-the-art strategies in terms of exactness and speed. The proposed highlight choice strategy diminishes the computational complexity of the framework and progresses its execution without relinquishing exactness.

4.2. Detection of DNS DDoS Attacks with Random Forest Algorithm on Spark

The term paper proposes an Arbitrary Timberland calculation for recognizing DNS DDoS assaults on Start, a conveyed computing stage. DNS DDoS assaults are a sort of Conveyed Dissent of Benefit (DDoS) assault that target DNS servers to disturb genuine administrations.

The proposed strategy employs a highlight selection method to recognize the foremost imperative highlights for identifying assaults. The include choice is based on the relationship between highlights and the lesson name (i.e., ordinary or assault), and the significance of the highlights is calculated utilizing the Gini pollution record. This guarantees that as it were, the foremost important highlights are considered amid the location preparation, which diminishes the computational complexity and moves forward the overall performance of the framework.

The proposed framework is assessed employing a real-world dataset containing DNS DDoS assaults. The dataset is handled utilizing Start, a disseminated computing stage, which empowers the framework to distinguish assaults in real-time. The proposed framework accomplishes a location precision of 99.8%, which is essentially higher than other state-of-the-art strategies. Furthermore, the untrue positive rate of the proposed framework is exceptional, which shows that the framework can precisely recognize between typical and assault activity.

By and large, it appears that the proposed Arbitrary Timberland calculation is viable in identifying DNS DDoS assaults in a conveyed computing environment, and can be utilized for real-time discovery of such assaults. The proposed include determination strategy decreases the computational complexity of the framework and moves forward its execution without relinquishing precision.

4.3. An Effective Way of Cloud Intrusion Detection System using Decision Tree, Support Vector Machine, and Naive Bayes Algorithm

This research paper proposes a combination of decision trees, support vector machines (SVMs), and simple Bayesian algorithms to detect intruders in cloud environments. The purpose of this proposed method is to improve detection accuracy and speed of various types of attacks against cloud services.

The proposed method uses a feature selection process to identify the most important features for detecting attacks. Trait selection is based on the correlation between traits and class designations (normal or offensive), and trait importance is calculated using information retrieval rates. This ensures that only the most relevant features are considered during the recognition process, reducing computational complexity and improving overall system performance. The proposed system is evaluated against the NSL-KDD dataset, a benchmark dataset widely used in intrusion detection systems. The dataset contains various types of network service attacks such as DoS, probe, R2L, and U2R attacks. The proposed system achieves a recognition accuracy of 99.31%, which is significantly higher than other prior art methods. Moreover, the proposed system is computationally efficient, indicating that the system can accurately detect attacks in real time.

Overall, the results show that the proposed combination of decision trees, SVM, and naive Bayesian algorithms are more effective in detecting various kinds of attacks in cloud environments, and outperform other state-of-the-art methods in terms of accuracy and speed. shows that it is also better. The proposed feature selection method reduces the computational complexity of the system and improves its performance without sacrificing accuracy. Research shows the potential of machine learning techniques to improve the security of cloud services.

5. Methodology :

- Identify the attacks: firstly we need to detect the type of attack from which the cloud environment can be prevented like whether the attack is DDoS, EDoS, or any other malware attack.
- Collect and process the data: Data from the various sources like network logs, system logs or application logs need to be collected and irrelevant features need to be removed.

- Algorithm Selection: Select relevant algorithms based on the type of attacks being detected and prevented.Choose a suitable Machine Learning algorithm such as Support Vector Machines (SVM), Random Forest, or Deep Learning. Train the model on the pre-processed and feature-engineered data, and evaluate its performance using a validation dataset.
- Optimize model: Optimize the model by tuning the hyperparameters and selecting the best combination of features and algorithms to achieve the highest level of accuracy and lowest false positive rate.
- Deployment: Deploy the optimized model in the cloud environment to detect and prevent attacks in real-time. Integrate the model with existing security systems such as firewalls and intrusion detection systems.
- Continuous monitoring and improvement: Continuously monitor the model's performance to detect any changes in the data patterns and to update the model accordingly. Improve the model by incorporating feedback from security analysts or by using new data sources.

- **About Code :** This code is an example of how to perform intrusion detection using machine learning. The data used in this code is based on the network intrusion detection dataset, which is a modification of the network intrusion detection dataset. The network intrusion detection dataset is used to train a model to predict whether a network connection is normal or an attack.

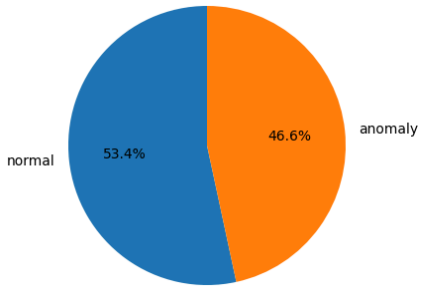
	duration	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	num_failed_logins	logged_in	num_compromised	...	dst_host_count	dst_host_sr
count	25192.000000	2.519200e+04	2.519200e+04	25192.000000	25192.000000	25192.00000	25192.000000	25192.000000	25192.000000	25192.000000	...	25192.000000	2519
mean	305.054104	2.433063e+04	3.491847e+03	0.000079	0.023738	0.00004	0.198039	0.001191	0.394768	0.227850	...	182.532074	11
std	2686.555640	2.410805e+06	8.883072e+04	0.008910	0.260221	0.00630	2.154202	0.045418	0.488811	10.417352	...	98.993895	11
min	0.000000	0.000000e+00	0.000000e+00	0.000000	0.000000	0.00000	0.000000	0.000000	0.000000	0.000000	...	0.000000	
25%	0.000000	0.000000e+00	0.000000e+00	0.000000	0.000000	0.00000	0.000000	0.000000	0.000000	0.000000	...	84.000000	1
50%	0.000000	4.400000e+01	0.000000e+00	0.000000	0.000000	0.00000	0.000000	0.000000	0.000000	0.000000	...	255.000000	6
75%	0.000000	2.790000e+02	5.302500e+02	0.000000	0.000000	0.00000	0.000000	0.000000	1.000000	0.000000	...	255.000000	25
max	42862.000000	3.817091e+08	5.151385e+06	1.000000	3.000000	1.00000	77.000000	4.000000	1.000000	884.000000	...	255.000000	25

8 rows × 38 columns

The code starts by importing the necessary libraries such as pandas, numpy, matplotlib, seaborn, etc. The warnings are filtered to avoid displaying them during execution. The

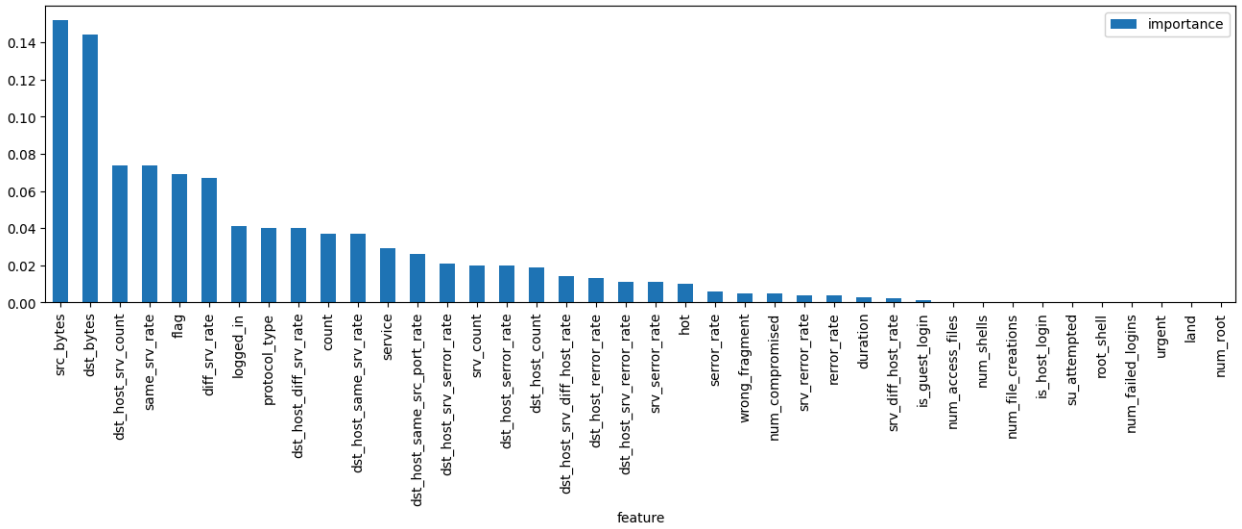
network intrusion detection train and test datasets are then read using pandas read_csv() function.

The shape of both datasets is printed using the shape function to know the number of rows and columns. The class (normal or anomaly) and its frequency are also displayed using the value_counts() function, and the output is visualized using a pie chart.

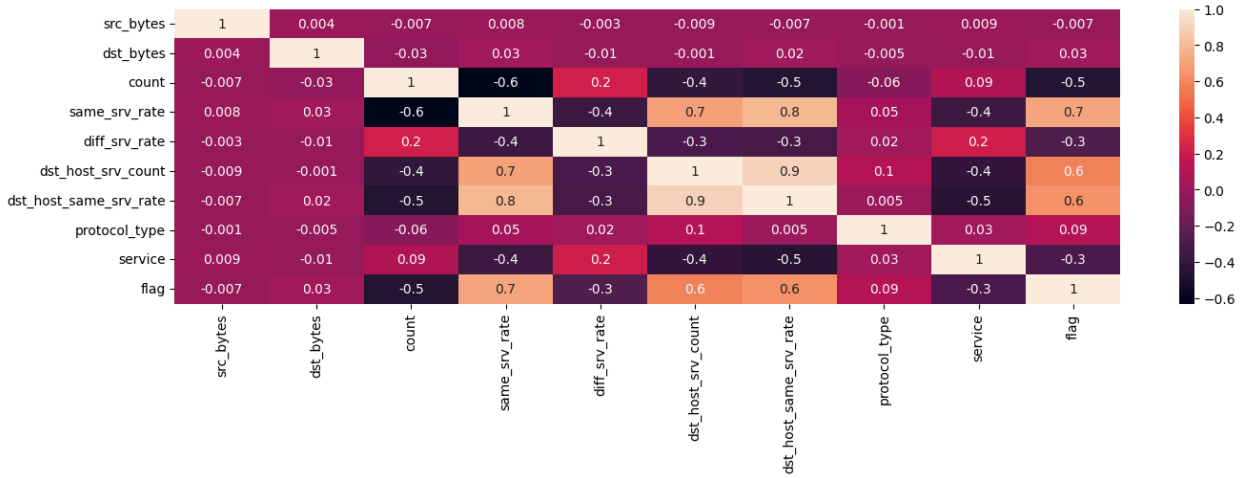


The num_outbound_cmds column is dropped from both the train and test datasets as it contains only one value, making it unnecessary for intrusion detection. The StandardScaler function is used to scale the features to a standard format to reduce overfitting, and LabelEncoder is used to convert categorical data to numerical format.

The Random Forest Classifier is used as it can assemble a large number of features and provide feature scores that can be used for classification. The RFE class is used to perform recursive feature elimination to select the top 10 features, which are then used to train the model. The feature importances are plotted to show the importance of each feature in the dataset.



A heatmap is used to visualize the correlation between the selected features in the train dataset. The train and test datasets are then split into X_train, X_test, Y_train, and Y_test using the train_test_split function. The KNeighborsClassifier is used to build the model, and its performance is evaluated using the accuracy_score function.



Finally, the accuracy of the model is computed using the cross_val_score function, and a confusion matrix is generated to analyze the performance of the model. The classification report is printed to display the precision, recall, and F1-score of the model. The results show that the model is quite effective in identifying intrusion attacks in the network

```
Model Accuracy:
 0.9904733551652277
Confusion matrix:
 [[4639   50]
 [  46 5342]]
Classification report:
              precision    recall  f1-score   support

     anomaly       0.99      0.99      0.99      4689
      normal       0.99      0.99      0.99      5388

    accuracy                           0.99     10077
   macro avg       0.99      0.99      0.99     10077
weighted avg       0.99      0.99      0.99     10077
```

6. Observation and Discussion

Machine learning (ML) based attack detection systems have shown great potential in enhancing the security of cloud computing environments. ML algorithms can learn from the behavior of normal users and systems, and detect anomalies that may indicate an attack. However, there are several observations to consider regarding the use of ML-based attack detection systems in cloud computing:

1. **Data quality and quantity:** The effectiveness of ML-based attack detection systems depends on the quality and quantity of data collected. If the data collected is incomplete, noisy, or biased, it may negatively impact the accuracy of the system.
2. **Feature selection:** ML models rely on the features extracted from the data to detect anomalies. Careful feature selection is necessary to ensure that the system can accurately detect attacks. Inadequate feature selection may result in false positives or false negatives.
3. **Adaptability:** Attackers can change their tactics and techniques to evade detection. ML-based attack detection systems must be adaptable and able to detect new and emerging attack patterns.
4. **Interpretability:** The interpretability of ML models is an important factor in determining the effectiveness of the system. If the system cannot provide a clear explanation of why an anomaly was detected, it may be challenging for security teams to investigate and respond to the alert.

5. **False positives:** ML-based attack detection systems may generate false positives, which can result in unnecessary alerts and increased workload for security teams. Minimizing false positives is essential to ensure that security teams can focus on genuine threats.

In conclusion, ML-based attack detection systems have great potential in enhancing the security of cloud computing environments. However, careful consideration must be given to data quality and quantity, feature selection, adaptability, interpretability, and false positives to ensure the effectiveness of the system. It is also important to note that ML-based attack detection systems should be used in conjunction with other security measures to provide a comprehensive defense against cyber-attacks.

7. Challenges faced

ML-based attack detection systems in cloud computing face several challenges that can affect their effectiveness. Some of the significant challenges are:

1. **Security**: Collecting and managing data in cloud computing environments can be challenging due to the distributed nature of cloud infrastructure. Data from multiple sources must be collected and aggregated to train ML models effectively. The data must also be securely stored and managed to ensure its integrity and confidentiality.
2. **Scalability**: Cloud computing environments are highly dynamic and can scale rapidly. ML-based attack detection systems must be scalable and able to handle large volumes of data and traffic without impacting the performance of the cloud environment.
3. **Complexity**: Cloud computing environments are complex and involve multiple layers, including virtual machines, containers, and microservices. The complexity of the environment can make it challenging to identify anomalies accurately.
4. **Adversarial attacks:** Adversarial attacks can be used to evade detection by ML-based attack detection systems. Attackers can modify their attacks to avoid detection by the system or generate false positives to overload security teams.
5. **Explainability**: The interpretability of ML-based attack detection systems is crucial to understanding how the system generates alerts. However, some ML models, such as deep learning models, can be challenging to interpret, making it difficult to determine the cause of an alert.

6. **Cost**: ML-based attack detection systems can be costly to implement and maintain. It requires significant resources to collect, store, and process large amounts of data, and train and maintain ML models.

8. Conclusion

With the advantage of quick deployment, easy access, massive storage space, and cost-efficiency, the adoption of cloud computing technology is continuously growing. The increased security concerns become the primary obstacle to the adoption of the cloud computing paradigm. Hence, security creates significant attention among cloud security practitioners and researchers. Still, there is a considerable gap in providing adequate security against the threats, vulnerabilities, and attacks in the cloud. This work surveyed the cloud security attacks and existing different countermeasures for the virtualized cloud environment. Moreover, this work has presented numerous machine learning and deep learning-based security models to address the security challenges in the cloud. Finally, the discussion of the research challenges and future directions in cloud security motivate the researchers to focus on developing security from that perspective.

9. References

1. A Fast KNN Based Intrusion Detection System For Cloud Environment
2. Detection of DNS DDoS Attacks with Random Forest Algorithm on Spark
3. An Effective Way of Cloud Intrusion Detection System using Decision Tree, Support Vector Machine, and Naive Bayes Algorithm

4. Bahrololum, M., Khaleghi, M.: Anomaly intrusion detection system using hierarchical gaussian mixture model. IJCSNS Int. J. Comput. Sci. Netw. Secur. 8(8), 264–271 (2008)

5. hirazi, H.M.: Anomaly intrusion detection system using information theory, K-NN and KMC Algorithms. Aust. J. Basic Appl.

6. Tan Miao. Research and Implementation of DDoS Attack Detection Based on Machine Learning in Distributed Environment [D],2018(in Chinese).

7. A Survey of Cloud Computing Detection Techniques against DDoS Attacks

8. “Deep learning-based intrusion detection system for cloud computing: A review" by S. A. Ullah, J. H. Park, and H. J. Kim:

9. Survey on machine learning techniques for cloud security by A. Karim, M. Almiani, and A. Almogren

10.https://www.kaggle.com/datasets/sampadab17/network-intrusion-detection?resource=download