International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 12, No.1 (Oct-2022)

https://dx.doi.org/10.12785/ijcds/120185

Detection and Prevention of Cyber Attacks on Cloud-Based Data Centers using Machine Learning

A. Mahendar[1] and Dr. K. Shahu Chatrapati[2]

[1]*Research Scholar, JNTUH Hyderabad and Assistant Professor, Department of CSE, C M R Technical Campus, Hyderabad, Telangana, and India*

[2]*Professor & Head, Department of CSE, JNTUH-CEM, Manthani, Telangana, and India*

Received 22 Jan. 2021, Revised 15 Jul. 2022, Accepted 23 Jul. 2022, Published 31 Oct. 2022

Abstract: The security issues with cloud computing have seen very high growth in recent times. This growth is certainly due to the increase in usage and adaptation of cloud-based services. The benefits from the cloud-based services or the data centers have improved the application building and application management life cycle. Hence, in the last 10 years, significant growth in data and data-driven applications can be seen on the cloud. This has indirectly increased the possibility of attacks on the cloud data. The data on the cloud is primarily hosted on virtual machines or virtualized storage devices. These devices are remotely manageable; thus, the attackers can also manipulate the device securities remotely or can generate cyber-attacks. In recent times, a good number of parallel research outcomes can be seen aiming to reduce the security issues on cloud services. Nonetheless, these parallel research outcomes are highly criticized for a variety of reasons. Firstly, these methods are highly time complex to detect any attack. Thus, cannot be considered reactive attacks, which is the most desirable type of security solution due to less complexity. Secondly, these methods are not applicable to detect a newer type of attack on the cloud or distributed network architectures. Henceforth, this work aims to solve these existing challenges using machine learning methods. This work proposes a novel algorithm to reduce the size of the network information without losing the critical information for detecting the attacks. This ensures timely detection of the attacks and can be preserved as reactive security protocols. Further, this work deploys another novel algorithm for detecting the anomalies based on the network connectivity characteristics using the rule mining methods. This will ensure the detection of the newer types of attacks also based on the deviation of the network characteristics. The proposed algorithms demonstrate nearly 99% accuracy in detecting cyber-attacks on cloud-based services.

Keywords: Dimensionality Reduction, Track File Analysis, Dynamic Rule Engine, Cloud Security, Influence Score Analysis

1. INTRODUCTION

The application building industry is showing a lot of traction towards the application hosting and deployment over the seamless services as cloud-based services. The primary driving force for this adaptation is the easy deployment of the applications on the cloud data centers. The integration of the other various services makes it easy for applications to seamlessly connect and manage various tasks due to easy modeling languages. The work by A. P. Achilleos et al. [1] has demonstrated the benefits of application deployment on the cloud using generic and standard application modeling. Hence, it is very evident that the adaptation of cloud computing is the most appropriate adaptation for application development and deployment. Further, these deployed applications continuously generate data and few of these applications are completely driven by data, which is again placed on the cloud. Thus, the new generation applications have increased the risk of attacks on the cloud. The recent review conducted by P. Kumar et al. [2] has demonstrated that the detection of the attacks for cloud services can be done using parametric analysis. Many research attempts can be seen, which use the parametric analysis strategies on network characteristics for attack detection. Thus, the aim of this work, to produce the reactive security protocols for attack detection using the network characteristic is valid to carry out the research. Also, it has been observed in the recent outcomes from various researchers that, the security protocols to be deployed on the cloud for cloud security must be designed as a service or broker. The work by T. Halabi et al. [3] is one of the significant proofs of this claim. Nonetheless, the parallel research outcomes are primarily criticized due to two issues:

1) Firstly, the model complexities of these existing methods are high, thus, critical to be adapted for real-time higher-performing cloud data centers.
2) Secondly, the existing methods are designed and dedicated to detecting specific types of attacks and as

E-mail address: adapamahendar.phd@gmail.com, shahujntu@gmail.com

these methods are not designed to detect the newer types of attacks, these methods become vulnerable.

Henceforth, this literature aims to solve these two problems primarily. Further, the rest of the literature is furnished to elaborate on the fundamental attack detection method in Section – II, with the understanding gained in Section – III, the recent improvements to the basic methods are critically analyzed, in Section – IV, the problem is formulated using mathematical notations in Section – V, the proposed solution is furnished and based on the proposed solution, the algorithms are discussed in the Section – VI. The obtained results from the proposed algorithms are formulated in Section – VII and finally in Section – VIII the final research conclusion is stated.

2. CLOUD-BASED ATTACK DETECTION FUNDAMENTALS

In this section of the literature, the foundational model for cloud-based attack detections is formulated. The generic dataset or characteristics of the attack detection is achieved from the notable characteristic dataset called KDD [4]. Assuming that, the network characteristics set, $NC[]$ is a collection of the KDD characteristics for a total of n number of observations. Thus, this relation can be formulated as,

$$NC[] =< KDD_{1,n} > \tag{1}$$

The fundamental attack detection method using the network characteristics is carried out using the thresholding method for each attribute. Assuming that any instance of the NC[] set can be considered as KDD[x], for which the threshold, , must be calculated as,

$$\lambda_x = \frac{\sum_i KDD_x[i]}{CountKDD_x} \tag{2}$$

Further, the threshold for each attribute must be calculated and a threshold collection, $\lambda[]$ must be formulated as,

$$\lambda[] = \frac{\sum_i KDD_{1..n}[i]}{CountKDD_{1..n}} \tag{3}$$

Further, assuming that, any random network characteristics, $NC[K]$ need to be validated for attack or normal instance of connection. Thus, this validation can be formulated as,

$$\exists NC[K] \rightarrow NC[i] > \lambda[i] \tag{4}$$

In this case, the assumption is either of the characteristics are higher than the threshold of that respective attribute, thus, that can be detected as an anomaly or attack. Henceforth with the foundational understanding of the attack detection mechanism, in the next section of this literature, the recent improvements over this method are analyzed.

3. RECENT RESEARCH OUTCOME

After the foundational analysis of the standard attack detection method, in this section of the literature, the recent improvements are discussed. From the traditional method, the attack detection methods have come a long way. The cloud services have enabled multiple advantages such as replication-controlled fault-tolerant architecture to build adaptive software services. The work by M. R. Chinnaiah et al. [5] has demonstrated the benefits of using such services to build higher-performing and less redundant applications. Not restricted to the standard services for increasing the fault tolerance, the increase in flexibility and increase in performance is also additional driving factors for cloud adaptations as reported by R. Moreno-Vozmediano et al. [6]. Hence, adaptability has increased extensively over the last few decades. As already mentioned, this higher rate of adaptations has also increased the risk factors. These risk factors are well summarized in the work by H. Tabrizchi [7], where the guidelines for building security solutions are well furnished and produce significant arguments to design such solutions using machine learning. The strong argument towards building machine learning-driven strategies is to ensure the detection of the newer types of attacks. This survey is an extension of the work carried out by R. Kumar et al. [8].

Some of the existing security solutions have demonstrated the use of encryption and decryption strategies with the higher order of the security keys. One such example can be seen in the work produced by X. Ma et al. [9]. Nonetheless, these strategies are primarily proactive security strategies and have always been criticized due to higher deployment complexities over cloud services.

Few of the standard case studies have showcased that the intensity of the information is the driving force for security solutions. The work by M. Marwan et al. [10] has demonstrated the effectiveness of security solutions over healthcare data and applications. These data-driven security protocols are often designed from the data security aspects and cannot demonstrate the overall security of the applications. This claim can be seen in the work by K. Vijayakumar et al. [11].

Nonetheless, a good number of research attempts are designed using machine learning-driven strategies. The machine learning-driven strategies not only reduced the time and model complexity but also due to the heuristic nature, can detect small variations in the characteristics, which can again lead to attacks. The work by O. A. Kwabena et al. [12] is one such claim. Nevertheless, for any machine learning algorithm to be highly effective, the training of the algorithm with a large past data is a basic requirement. During the design of security applications, the management of the historical data from the network characteristics is one of the critical aspects, which is highlighted in the work by Z. Chkirbene et al. [13]. The historical information from the network characteristics can also drive rule engine building. The work by H. Abbasi et al. [13] confirms the same thought by analyzing the trace files. This literature also builds the proposed solutions using the trace file analysis, which is already mapped to the KDD characteristics. Further, this section of the work raises more questions than answers as the parallel research outcomes cannot address

most of the challenges. Using a meta-heuristic technique known as binary particle swarm optimization, the VMShield collects system call characteristics using the Bag of n-gram approach and picks significant ones. An benefit of the Random Forest (RF) classifier is that it is able to distinguish between benign and malignant processes, making it more effective than a traditional signature-matching technique [14], [15]. Even more important is the capacity to identify malicious attacks early on so that essential mitigation may be carried out to minimise interruption and restore cloud operations and their live migration procedures [16]. In comparison to the current detection load distribution strategies, our system increases attack detection, reduces false positives and negatives, and reduces CPU, memory, and bandwidth usage during DDoS assaults [17]. DDoS assaults, after data theft, are the most common sort of distributed denial-of-service (DDoS) attack. With only a few DDoS TCP flood assaults, a cloud project may be completely depleted of all of its resources, bandwidth, and functionality. It is critical for cloud initiatives, particularly eHealth ones, to be able to identify and mitigate such assaults in real time [18]. Cloud-based image detection models, which are more complicated than classifiers but have not received enough attention yet [19], have the potential to have similar security concerns to deep learning classification models that have been extensively demonstrated to be vulnerable to adversarial examples. Thus, in the next section of this work, the research problems are formulated.

4. PROBLEM IDENTIFICATION

With the detailed realization of the parallel research outcomes, in the previous section of this work, the formulation of the problem is furnished here. As already assumed that the total number of attributes or characteristics or parameters are n for the $NC[]$ set. Thus, for the calculation of the threshold the time complexity, $t1$, can be formulated as,

$$t1 = n.t \tag{5}$$

Assuming that t is the unit time to calculate the threshold for each attribute. Further, the time complexity, $t2$, for analyzing any given characteristics, $NC[i]$, for detection of the attack can be formulated as,

$$t2 = NC][i] : KDD[i] \rightarrow n.n.t \tag{6}$$

Thus, the total time complexity can be formulated as,

$$t1 + t2 = n.t + n.n.t \tag{7}$$

or

$$t1 + t2 = n.t + n^2t \tag{8}$$

Finally,

$$t1 + t2 = O(n^2) \tag{9}$$

Considering $n^2 >> n$, thus n can be ignored, and t is a constant. Thus, it is natural to realize that the time complexity is significantly higher for a reactive security solution. Henceforth, in the next section of this work, the proposed solution is furnished.

5. PROPOSED SOLUTION

After realizing the problems in the existing system, this section of the literature is dedicated to furnishing the proposed solutions. This literature solves two problems in the existing parallel research outcomes. Firstly, the reduction of the character set using the influence factors is furnished here. Continuing from Eq. 1, assuming that the class variable, KDD[C], is the deciding factor of the connection types as normal or anomalies. Thus, the impact factor, IF, for any characteristics or attribute, KDD[X], can be formulated as,

$$IF_{C,X} = \frac{\sum\{(KDD[X] - \lambda_x).(KDD[C] - \lambda_C)\}}{\sqrt{(KDD[X] - \lambda_x)^2.(KDD[C] - \lambda_C)^2}} \tag{10}$$

Further, assuming that for all the attributes, the impact factors are calculated as impact factor set, $IF[]$ as,

$$IF[] = IF_{KDD[C]:\forall KDD[X]} \tag{11}$$

Furthermore, the threshold, θ , of the impact factor set is calculated as,

$$\theta = \frac{\sum IF[]}{count(IF[])} \tag{12}$$

With the comparison with the threshold, , the impact of each attribute is considered and the attributes having more impact factors are considered in the reduced attribute set, KDD' as,

$$KDD'[] = KDD[x], IFIF[X] > \theta \tag{13}$$

Secondly, the rule engine-based detection of the attacks is realized. Assuming that, the threshold set, $TH[]$, of the reduced dataset for the attack denoted instances as,

$$TH[] = \prod_{KDD'[C]="Attack"} KDD'[] \tag{14}$$

Further, the rule engine is designed using the tree formation, T, as,

$$\forall TH[] \rightarrow BuildTree(T) \tag{15}$$

Finally with the tree formation, T, the rule engine, R[], can be constructed as,

$$R = \prod_{node=leaf} T[] \cup \prod_{node=leaf-1} T[] \tag{16}$$

The immediate benefit of this proposed solution is, the reduced dataset, $KDD'[]$ can contain m number of attributes. Thus, for the calculation of the threshold the time complexity, $t1$, can be formulated as,

$$t1 = m.t \tag{17}$$

Assuming that t is the unit time to calculate the threshold for each attribute. Further, the time complexity, $t2$, for analysing any given characteristics, NC[i], for detection of the attack

can be formulated as,

$$t2 = NC[i] : KDD'[i] \rightarrow m + m.t \quad (18)$$

Thus, the total time complexity can be formulated as,

$$t1 + t2 = m.t + m + m.t \quad (19)$$

or

$$t1 + t2 = m.t + 2m.t \quad (20)$$

Finally,

$$t1 + t2 = O(m) \quad (21)$$

Considering $2m = m$, thus n can be ignored, and t is a constant. Henceforth, based on the proposed mathematical models for the solutions, the proposed algorithms are furnished and elaborated in the next section of this literature.

6. PROPOSED ALGORITHMS

After the formulation of the proposed solutions, in this section of the work, the proposed algorithms are furnished. Firstly, the dimensionality reduction algorithm is formulated.

Algorithm 1 Influence Based Characteristics Reduction Algorithm (IBCR)

Input: The KDD Dataset
Output: Reduced Set as KDD'
Process:
Step - 1. Load the KDD dataset
Step - 2. For each attribute in the KDD dataset KDD[i]
a. Calculate the impact as $IF[i]$ using Eq. 10
Step - 3. Calculate the threshold, TH using Eq. 12
Step - 4. For each attribute in $IF[i]$ as $IF[j]$
a. Build the $KDD'[]$ set using Eq. 13
Step - 5. Return $KDD'[]$

The vitally direct method for dimensionality decreases, head part examination, plays out straight planning of the information to a lower-dimensional space so that the fluctuation of the information in the low-dimensional portrayal is augmented. Practically speaking, the covariance (and at times the connection) network of the information is built and the eigenvectors on this grid are processed. The eigenvectors that compare to the biggest eigenvalues (the foremost parts) would now be able to be utilized to remake a huge part of the difference of the first information. Also, the initial not many eigenvectors can frequently be deciphered as far as the huge scope actual conduct of the framework, since they regularly contribute by far most of the framework's energy, particularly in low-dimensional frameworks. In any case, this should be demonstrated dependent upon the situation as not all frameworks display this conduct. The first space (with measurement of the number of focuses) has been decreased (with information misfortune, however ideally holding the main difference) to the space crossed by a couple of eigenvectors.

Algorithm 2 Dynamic Rule Engine for Attack Detection Algorithm (DRE-AD)

Input: Reduced Set as KDD'
Output: Rule Set as R
Process:
Step - 1. Load the KDD' dataset
Step - 2. For each attribute in the KDD dataset $KDD[i]$
a. Calculate the threshold, $TH[]$ using Eq. 14
Step - 3. Build the tree nodes, $T[]$ as Eq. 15
Step - 4. For each element in $T[]$ as $T[k]$
a. If $T[node] == Leaf$
b. R = T[node_left] & T[node_right]

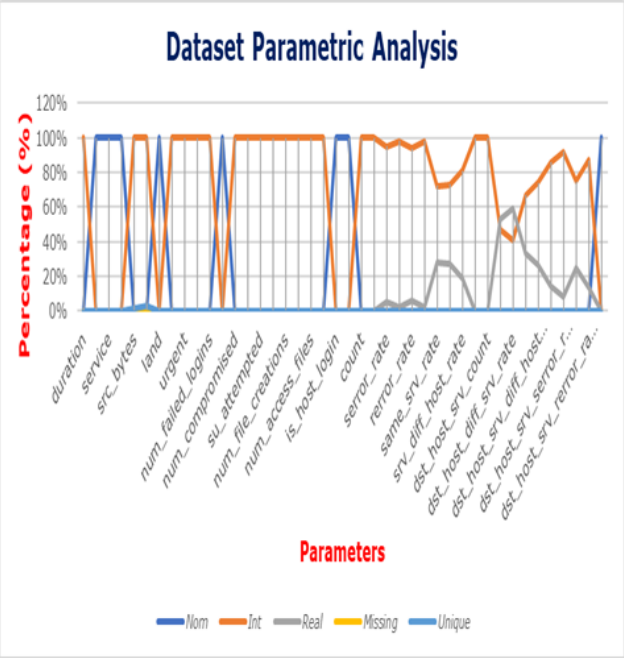

Figure 1. Initial Parametric Analysis

Secondly, the rule engine building algorithm is furnished. Digital warfare employs tactics for safeguarding and attacking internet-connected data and PC networks, sometimes in the form of a long-running digital operation or set of linked missions. It prevents a competitor from being able to accomplish the same by attacking a competitor's core PC frameworks with mechanical war instruments. However, cyberterrorism is the use of PC network instruments to shut down essential public frameworks (such as energy, transportation, and government functions) as well as to pressure and endanger an administration and non-military individual's population. Cyberwarfare and cyberterrorism have a same goal: to destroy the internet's underlying infrastructure and computer systems that are linked to it. Further, in the next section of this work, the obtained results are discussed.

TABLE I. Dataset initial parameters

SNo	Attribute Name	Attribute Type	Nominal Data (%)	Integer Data (%)	Real Data (%)	Missing Value (%)	Unique Value (%)
1	duration	Num	0%	100%	0%	0%	0%
2	protocol_type	Nom	100%	0%	0%	0%	0%
3	service	Nom	100%	0%	0%	0%	0%
4	flag	Nom	100%	0%	0%	0%	0%
5	src_bytes	Num	0%	100%	0%	0%	1%
6	dst_bytes	Num	0%	100%	0%	0%	3%
7	land	Nom	100%	0%	0%	0%	0%
8	wrong_fragment	Num	0%	100%	0%	0%	0%
9	urgent	Num	0%	100%	0%	0%	0%
10	hot	Num	0%	100%	0%	0%	0%
11	num_failed_logins	Num	0%	100%	0%	0%	0%
12	logged_in	Nom	100%	0%	0%	0%	0%
13	num_compromised	Num	0%	100%	0%	0%	0%
14	root_shell	Num	0%	100%	0%	0%	0%
15	su_attempted	Num	0%	100%	0%	0%	0%
16	num_root	Num	0%	100%	0%	0%	0%
17	num_file_creations	Num	0%	100%	0%	0%	0%
18	num_shells	Num	0%	100%	0%	0%	0%
19	num_access_files	Num	0%	100%	0%	0%	0%
20	num_outbound_cmds	Num	0%	100%	0%	0%	0%
21	is_host_login	Nom	100%	0%	0%	0%	0%
22	is_guest_login	Nom	100%	0%	0%	0%	0%
23	count	Num	0%	100%	0%	0%	0%
24	srv_count	Num	0%	100%	0%	0%	0%
25	serror_rate	Num	0%	95%	5%	0%	0%
26	srv_serror_rate	Num	0%	98%	2%	0%	0%
27	rerror_rate	Num	0%	94%	6%	0%	0%
28	srv_rerror_rate	Num	0%	98%	2%	0%	0%
29	same_srv_rate	Num	0%	72%	28%	0%	0%
30	diff_srv_rate	Num	0%	73%	27%	0%	0%
31	srv_diff_host_rate	Num	0%	81%	19%	0%	0%
32	dst_host_count	Num	0%	100%	0%	0%	0%
33	dst_host_srv_count	Num	0%	100%	0%	0%	0%
34	dst_host_same_srv_rate	Num	0%	47%	53%	0%	0%
35	dst_host_diff_srv_rate	Num	0%	41%	59%	0%	0%
36	dst_host_same_src_port_ra	Num	0%	67%	33%	0%	0%
37	dst_host_srv_diff_host_ra	Num	0%	74%	26%	0%	0%
38	dst_host_serror_rate	Num	0%	86%	14%	0%	0%
39	dst_host_srv_serror_rate	Num	0%	92%	8%	0%	0%
40	dst_host_rerror_rate	Num	0%	75%	25%	0%	0%
41	dst_host_srv_rerror_rate	Num	0%	87%	13%	0%	0%
42	class	Nom	100%	0%	0%	0%	0%

7. RESULTS AND DISCUSSIONS

After the realization of the proposed algorithms in the previous section of the work, in this section, the obtained results are discussed. The outcome is visualized graphically [Fig 1]. Firstly, the data set [4] parameters are discussed [Table I]. From the data points, it is evident to mention that the data set is very well balanced and well distributed over the nominal, real, and integer data points. Also, the data set does not contain any missing values, which makes the dataset highly reliable.

Finally, the uniqueness is observed to be less in all the parameters, this demonstrates the least possibilities for containing outliers. Secondly, the influence factor analysis results are furnished for identifying the important parameters to frame the rule engine in [Table II]. The finalized influence scores are also visualized here [Fig 3]. The finalized influence scores are also visualized here [Fig 2].

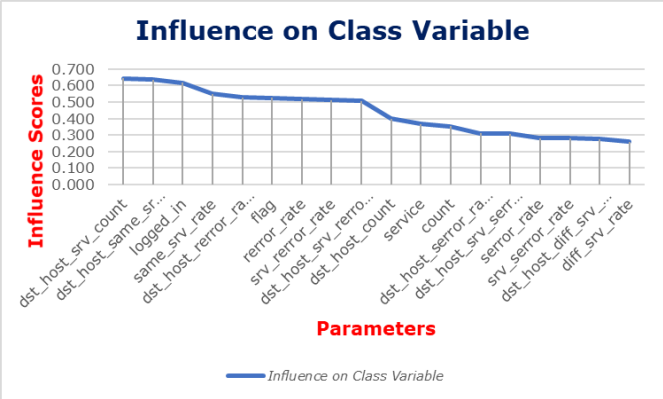

Figure 2. Influence Score Analysis

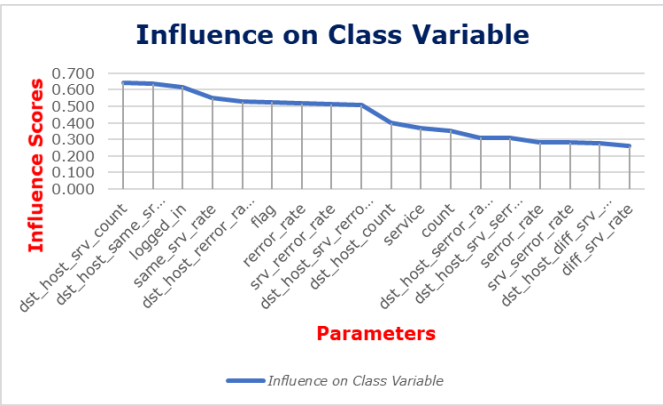

Figure 3. Influence Score Analysis – Selected Parameters with Higher Influence Scores

From this result, it is natural to realize that few of the parameters or attributes demonstrate a higher influence on the class variable. These parameters are to be considered for further analysis and rule engine design. The selected parameters are listed here [Table III]. The finalized influence scores are also visualized in [Fig 4].

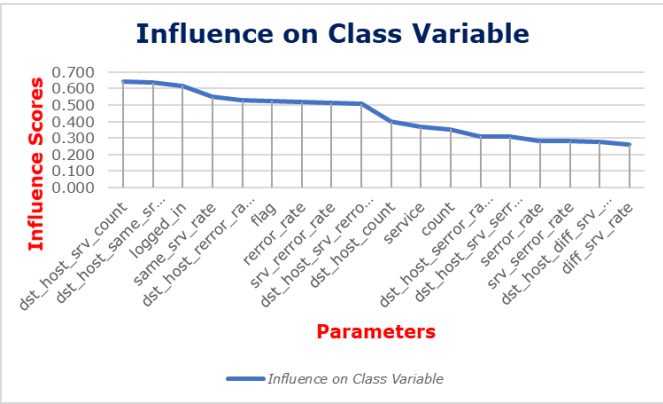

Figure 4. Influence Score Analysis – Selected Parameters with Higher Influence Scores

TABLE II. Influence factor analysis

Attribute Name	Influence on Class Variable
duration	0.1503
protocol_type	0.1125
service	0.3677
Flag	0.5250
src_bytes	0.0159
dst_bytes	0.0973
Land	0.0077
wrong_fragment	0.0387
urgent	0.0086
hot	0.0567
num_failed_logins	0.1348
logged_in	0.6178
num_compromised	0.0208
root_shell	0.0177
su_attempted	0.0217
num_root	0.0214
num_file_creations	0.0158
num_shells	0.0524
num_access_files	0.0703
num_outbound_cmds	0.0000
is_host_login	0.0097
is_guest_login	0.1162
count	0.3529
srv_count	0.0922
serror_rate	0.2819
srv_serror_rate	0.2799
rerror_rate	0.5172
srv_rerror_rate	0.5133
same_srv_rate	0.5502
diff_srv_rate	0.2608
srv_diff_host_rate	0.1923
dst_host_count	0.3988
dst_host_srv_count	0.6450
dst_host_same_srv_rate	0.6357
dst_host_diff_srv_rate	0.2761
dst_host_same_src_port_rate	0.0302
dst_host_srv_diff_host_rate	0.0216
dst_host_serror_rate	0.3117
dst_host_srv_serror_rate	0.3082
dst_host_rerror_rate	0.5278

Finally, the accuracy of the proposed rule engine is tested, and the reports are furnished here [Table IV].

Further, the final results are visualized [Fig 5].

TABLE III. Influence factor analysis

Attribute Name	Influence on Class Variable
dst_host_srv_count	numeric
dst_host_same_srv_rate	numeric
logged_in	nominal
same_srv_rate	numeric
dst_host_rerror_rate	numeric
flag	nominal
rerror_rate	numeric
srv_rerror_rate	numeric
dst_host_srv_rerror_rate	numeric
dst_host_count	numeric
service	nominal
count	numeric
dst_host_serror_rate	numeric
dst_host_srv_serror_rate	numeric
serror_rate	numeric
srv_serror_rate	numeric
dst_host_diff_srv_rate	numeric
diff_srv_rate	numeric

TABLE IV. Attack detection accuracy analysis

Analysis Metric Parameter	Values
Correctly Classified Instances	99.5779 %
Incorrectly Classified Instances	0.4221 %
Kappa statistic	0.9914
Mean absolute error	0.0067
Root mean squared error	0.0579
Relative absolute error	1.37%
Root relative squared error	11.72%
Total Number of Instances	85763

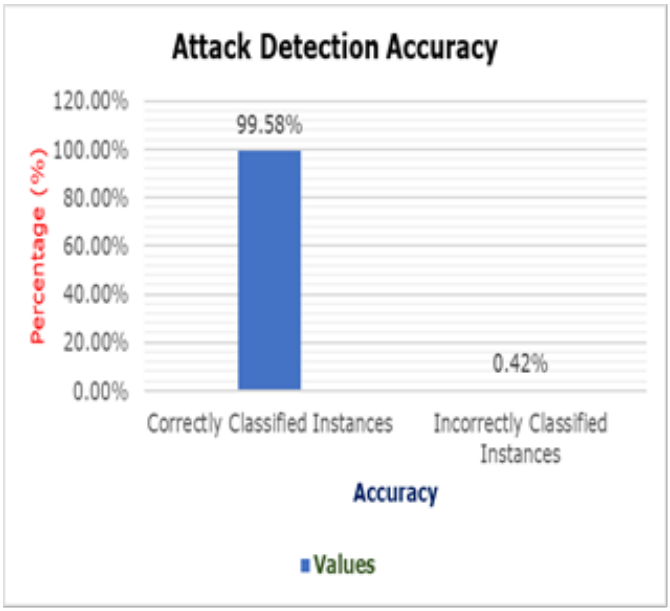

Figure 5. VHDL Synthesis Process

Thus, is it very evident to mention that the accuracy of the proposed rule engines is significantly higher. Henceforth in the next section of this work, the final research conclusion is presented.

8. Conclusions and Future Work

The historical data available from the cloud-based services are significantly important to design any new security mechanisms. However, the size of the parametric data extracted from the service traces is huge and restricts the researchers to adapt to this method. Thus, this work firstly contributed towards the reduction of the parametric information using the proposed influence score analysis. Due to this strategy, the complexity of the proposed algorithms has been reduced to $O(n)$ from $O(n2)$. Further, the existing methods are not capable of detecting newer types of attacks, hence this work proposed the second contribution towards attack detection by designing a dynamic rule engine for attack detection. This proposed rule engine has demonstrated 99.57% accuracy in detecting the attacks, standard or new.

References

[1] A. P. Achilleos, K. Kritikos, A. Rossini, G. M. Kapitsaki, J. Domaschka, and M. Orzechowski, "The cloud application modelling and execution language," *J. Cloud Comput*, vol. 8, pp. 1–20, 2019.

[2] P. Kumar and P. J. A. Alphonse, "Attribute based encryption in cloud computing: A survey gap analysis and future directions," *J. Netw. Comput. Appl*, vol. 108, pp. 37–52, 2018.

[3] T. Halabi and M. Bellaiche, "A broker-based framework for standardization and management of cloud security-slas," *Comput. Secure*, vol. 75, pp. 59–71, 2018.

[4] M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," *IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*, 2019.

[5] M. R. Chinnaiah and N. Niranjan, "Fault tolerant software systems using software configurations for cloud computing," *J. Cloud Comput.*, vol. 7, pp. 50–71, 2018.

[6] R. Moreno-Vozmediano, R. S. Montero, E. Huedo, and I. M. Llorente, "Efficient resource provisioning for elastic cloud services based on machine learning techniques," *J. Cloud Comput.*, vol. 8, pp. 51–60, 2019.

[7] H. Tabrizchi and M. K. Rafsanjani, "A survey on security challenges in cloud computing: Issues threats and solutions," *J. Supercomput.*, vol. 76, pp. 9493–9532, 2020.

[8] R. Kumar and R. Goyal, "On cloud security requirements threats vulnerabilities and countermeasures: A survey," *Comput. Sci. Rev.*, vol. 33, pp. 1–48, 2019.

[9] X. Ma, J. Ma, H. Li, Q. Jiang, and S. Gao, "Pdlm: Privacy-preserving deep learning model on a cloud with multiple keys," *IEEE Trans. Services Comput*, vol. 1, pp. 1–18, 2018.

[10] M. Marwan, A. Kartit, and H. Ouahmane, "Security enhancement in healthcare cloud using machine learning," *Procedia Comput. Sci.*, vol. 127, pp. 388–397, 2018.

[11] K. Vijayakumar and C. Arun, "Continuous security assessment of cloud-based applications using distributed hashing algorithm in sdlc," *Cluster Comput.*, vol. 22, pp. 10 789–10 800, 2019.

[12] O. A. Kwabena, Z. Qin, Z. Qin, and T. Zhuang, "Mscryptonet: Multi-scheme privacy-preserving deep learning in cloud computing," *IEEE Access*, vol. 7, pp. 29 344–29 354, 2019.

[13] Z. Chkirbene, A. Erbad, and R. Hamila, "A combined decision for secure cloud computing based on machine learning and past information," *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, vol. 1, pp. 1–6, 2019.

[14] P. M. et al., "Vmshield: Memory introspection-based malware detection to secure cloud-based services against stealthy attacks," *IEEE Transactions on Industrial Informatics*, vol. 17, pp. 6754–6764, 2021.

[15] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, "A survey on deep learning for big data," *Inf. Fusion*, vol. 42, pp. 146–157, 2018.

[16] O. Alkadi, N. Moustafa, and B. Turnbull, "A review of intrusion detection and blockchain applications in the cloud: Approaches, challenges and solutions," *IEEE Access*, vol. 8, pp. 104 893–104 917, 2020.

[17] O. A. Wahab, J. Bentahar, H. Otrok, and A. Mourad, "Optimal load distribution for the detection of vm-based ddos attacks in the cloud," *IEEE Transactions on Services Computing*, vol. 13, pp. 114–129, 2020.

[18] A. Sahi, D. Lai, Y. Li, and M. Diykh, "An efficient ddos tcp flood attack detection and prevention system in a cloud environment," *IEEE Access*, vol. 5, pp. 6036–6048, 2017.

[19] X. L. et al., "Adversarial examples versus cloud-based detectors: A black-box empirical study," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, pp. 1933–1949, 2021.

A. MAHENDAR short biographyResearch Scholar, JNTUH Hyderabad. Currently working as Assistant Professor in Department of CSE, CMR Technical Campus, Hyderabad, Telangana and India.

Dr. K. Shahu Chatrapati received his B.Tech in Computer Science & Engineering from Acharya Nagarjuna University, Andhra Pradesh, India in the year 1999, M.Tech in Computer Science & Engineering from Andhra University College of Engineering Vishakhapatnam, Andhra Pradesh, India in the year 2002, and Ph.D in Computer Science & Engineering (CSE) from Acharya Nagarjuna University in 2010. He is currently working as Professor & Head, Dept. of CSE, JNTUH College of Engineering, Manthani, Telangana, India. He has more than 20 years of teaching experience. Under his guidance, two students got awarded Ph.D and 8 students are currently pursuing Ph.D. He published more than 50 research papers in reputed International and National Journals. His research interests include Cloud computing, Data ware Housing & Data Mining, Artificial Intelligence, Machine Learning and Data Science.