Development of a Threat Detection System for Network Attacks

G. Krishna Kishore, Suresh Babu Dasari , S. Ravi Kishan

Abstract. ***In today's world the structure and dynamic interactions in the large network systems has become substantially complex. The threats and security attacks are currently spread everywhere and are tend to increase significantly in the future with the Internet of Things (IoT). The late detection of security threats causes a significant increase in the risk of irreparable damages, disabling any defense attempt. In this new era of security, information security professionals must deliver a very effective, real-time defense that can predict inherent threats to all the critical assets. All attacks will leave detectable traces, even though most of them will be complex and very hard to analyze. Threat monitoring systems, must have the capacity to observe activities in big data collected from networks and detect the threats. In order to provide the most secured network environment and network traffic monitoring threat detection systems must handle the real-time data. An accurate and reliable TDS will be automated that will be able to improve the traditional methods in order to fulfill the goals quickly and detect malicious activity and act accordingly. We focus on a robust classification method that includes an efficient SVM classifier will be used to handle network security concerning big network traffic.***

Keywords: Threat Detection System (TDS), Network Attacks, Support Vector Machine (SVM), Network Security

I. INTRODUCTION

The unfaltering pattern of expanding digital dangers has made digital security foremost in ensuring individual and private licensed innovation. So as to give the most exceedingly anchored system condition, organize traffic checking and risk recognition frameworks must deal with ongoing information from differed and stretching places in big business systems. At whatever point an interruption occurs, the security and estimation of a PC framework is imperiled. System based assaults make inconvenience for genuine clients to get to different system benefits by intentionally involving or disrupting system assets and administrations. This should be possible by sending a lot of system traffic, misusing surely understood blames in systems administration administrations, and by over-burdening system has [2]. Security is an essential issue in the web log information where the stream of bundles contains various gatecrashers. Shielding systems from PC security assaults is an imperative misgiving of PC security as system traffic may prompt assortment of data trade and touchy information exchange. In spite of the fact that it is likewise notable that the reliance of system are additionally developing quickly. Because of this the system condition are extremely essential now a days and it will turn out to be increasingly convoluted in pending time. This traffic may prompt enormous harm of system framework and its related assets [3].A TDS is a computer security system that monitors the . bundles on the system wire and endeavors to find if a programmer or wafer is endeavoring to break into a framework to cause a refusal of administration assault. A common precedent can be a framework that looks for expansive number of TCP association demands (SYN) to a wide range of ports on the objective machine, in this way finding in the event that somebody is endeavoring for a TCP port output or not.Security and privacy of a system is vulnerable, when an intrusion happens. Threat detection mainly focuses on individual platforms, networks, systems, endpoints or almost any other IT resource. It can also cover a wide range of functions, including intrusion detection, remediation of viruses, malware or spyware, or simply observing the unauthenticated and unauthorized use of programs or networks. The fact is that more and more major security breaches are occurring where the traditional approaches to security no longer work. More than ever before, we see that the internet is changing computing as we know it. The conceivable outcomes and openings are boundless as are the dangers and odds of malevolent system dangers. It is imperative that the security instruments of a framework ought to be planned with the end goal that it ought to avoid unapproved access to framework assets and information. Totally averting security ruptures at present seem unlikely. We can, in any case, endeavor to identify these Network Threat endeavors with the goal that move might be made to fix the harm later. This field of research is called Network Threat Detection. Therefore methods that are specifically taken from data science and machine learning should be coupled with an increased computational power (i.e, at lower cost) and the availability of relevant data will be making inroads into automated threat detection that identifies new patterns, detects events that may not match a specific signature, determines behavioral abnormalities, and subsequently acts upon the possibility of compromise. So, an accurate and reliable TDS should be automated that will be able to improve the traditional methods in order to detect the threats and malicious activities.

Revised Manuscript Received on March 20, 2019.

G. Krishna Kishore, Department of Computer Science and Engineering, V.R. Siddhartha Engineering College, Vijayawada.

Suresh Babu Dasari , Department of Computer Science and Engineering, V.R. Siddhartha Engineering College, Vijayawada.

S. Ravi Kishan, Department of Computer Science and Engineering, V.R. Siddhartha Engineering College, Vijayawada.

Published By:
Blue Eyes Intelligence Engineering & Sciences Publication

II. RELATED WORK

The field of system interruption location and system security has been around since late 1990s. From that point onwards number of structures and approaches have been proposed and numerous devices are worked to identify organize interruption. Different strategies, for example, rule based calculation, arrangement, grouping, hereditary calculations, bolster vector machines, cross breed order and others have been utilized to identify organize interruptions [4]. Antonio Gonzalez Pastana Lobato [1] et. al., proposed and actualized a precise continuous danger recognition framework, utilizing open source stages. The incorporated framework permits enormous information examination in a stream preparing way. The framework utilizes machine learning for both assault classification and abnormality location. Moreover, a dataset with named classes is made for the framework assessment, containing ordinary system utilization and a few assaults from parcel catches, preoccupied in flows. The framework engineering depends on the idea of the lambda design, which consolidates customary group handling over a verifiable database with ongoing stream preparing investigation. A framework model was created and assessed. In this way, the proposed framework can recognize both known and multi day assaults through automatized classification and oddity recognition techniques. Zhijiang Chen [2] et. al., proposed and built up a spilling based risk discovery framework that can quickly examine profoundly serious system traffic information progressively. In particular, the framework depends on Cloud time stage. To manage constant information frameworks, a flume module is executed and the investigation modules are created dependent on start which is a proficient usage of Map Reduce. Horng et al. proposed a SVM-based TDS, which joins a various leveled grouping and the SVM. The various leveled bunching calculation was utilized to give the classifier less and higher quality preparing information to lessen the normal preparing and testing time and enhance the arrangement execution of the classifier. Probed the adjusted marks KDD Cup 99 dataset, which incorporates some new assaults, the SVM-based TDS scored a general exactness of 95.75% with a false positive rate of 0.7%. Proposed and actualized an exact constant danger recognition framework, utilizing open source stages. The incorporated framework permits enormous information examination in a stream handling way. The proposed framework utilizes machine learning for both assault order and irregularity identification. Moreover, a dataset with named classes for the framework assessment, containing typical system use and a few assaults from parcel catches, preoccupied in streams. The framework design depends on the idea of the lambda engineering, which joins conventional group handling over a chronicled database with continuous stream preparing investigation. Along these lines, the proposed framework can identify both known and multi day assaults through mechanized grouping and irregularity recognition strategies. A framework model was produced and assessed. Besides, the proposed framework quickly recognizes security dangers, the stream preparing is related with the verifiable database handling to adjust the recognition calculations progressively. Trials were directed on surely understood TDS dataset. This is vital in assessing the execution of TDS since KDD dataset is obsolete and does not contain most novel assault designs in it. What's more, these datasets are much of the time utilized in the writing to assess the execution of TDS. Also, these datasets have different example sizes and diverse quantities of highlights, so they give significantly more difficulties to exhaustively testing highlight determination calculations. Unique in relation to the identification system suggested that structures just for double arrangement, we plan our proposed system to consider multiclass order issues. This is to demonstrate the adequacy and the practicality of the proposed strategy.

3. Proposed System and Architecture

The proposed system is as follows:

1. Analyzing the big packet stats data to deduce malicious packet flows.
2. Classification of network packets in a voluminous network data set (Kdd) will be done using a data mining approaches which is a stochastic machine learning process that evolves over time in a probabilistic manner.
3. A significant amount of research has been conducted to develop intelligent intrusion detection (IDS) techniques, which help achieve better network security while processing huge amounts of data.
4. Methods proposed such as Support Vector Machine (SVM) are used to classify network traffic patterns that do not match normal network traffic.
5. These systems were equipped with classifiers (such as SVM) to detect normal traffic and two different types of attacks (i.e., DoS, Probing).
6. Such data can be used for malicious packet estimations for extensive applications when direct measurements of packets are not available.
7. In addition, it demonstrates that the proposed method Classification has superior performance with respect to Big Data Network Traffic.

Figure 1 presents the architecture of the system. The architecture diagram represents mainly flow of requests between user and simulation system. The overall system is designed in three tiers separately using three layers called presentation layer using applets, business logic layer using java and data link layer at network center or mining initialize.

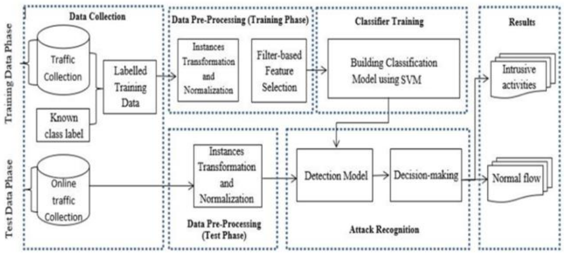

Training Data Phase
Test Data Phase
Data Collection
Traffic Collection
Known class label
Labelled Training Data
Online traffic Collection
Data Pre-Processing (Training Phase)
Instances Transformation and Normalization
Filter-based Feature Selection
Classifier Training
Building Classification Model using SVM
Instances Transformation and Normalization
Data Pre-Processing (Test Phase)
Detection Model
Decision-making
Attack Recognition
Results
Intrusive activities
Normal flow

Figure 1. Threat Detection System Architecture

The architecture diagram represents mainly flow of requests between user and simulation system. The overall system is designed in three tiers separately using three layers called presentation layer using applets, business logic layer using java and data link layer at network center or mining initialize. This project was developed using 2-tier architecture.

STEP 1: KDDCup99 data set is collected.

STEP 2: Training and testing phase is done for the original KDD dataset.

STEP 3: Data set pre-processing in training and testing is done.

STEP 4: Classification model is built using SVM and the classifier training is used in the detection model for attack recognition.

STEP 5: In the Attack recognition stage, decision making is done for the data with the help of detection model.

STEP 6: Results are displayed showing whether it is an intrusion activity or normal flow of data.

3.1 Dataset Description

Network datasets are necessary for many types of network research. With the colossal development of PC systems utilization and the enormous increment in the quantity of uses running over it, organize security is winding up progressively increasingly imperative. The well known KDD Cup 1999 dataset was utilized to assess the proposed framework [5]. Under the sponsorship of Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory (AFRL), MIT Lincoln lab has gathered and dispersed the datasets for the assessment of looks into in PC organize interruption identification frameworks. The KDDCup99 dataset is subsets of the DARPA benchmark dataset [4]. KDDCup99 preparing dataset is around four giga bytes of compacted double TCP dump information from seven weeks of system traffic, handled into around five million associations record each with around 100 bytes. The two weeks of test information have around two million association records. Each KDDCup'99 preparing association record contains 41 includes and is marked as either ordinary or an assault, with precisely one explicit assault type. Every one of the assaults in the dataset fall into four noteworthy classes, to be specific, Denial of Service (Dos), Probing (Probe), Remote to Local (R2L) and User to Root (U2R).

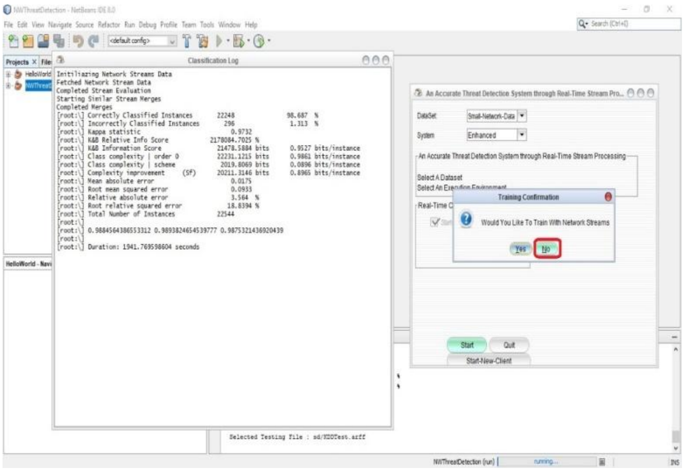
III. RESULTS

In this chapter we are discussing about the output screens that shows the flow of process.

The figure 2. shows two windows.

1. One is the Classification Log window which shows the output.
2. Other one is the User Interface Selection window.

Click on 'START' for the system to start. A dialog box prompt is shown asking whether we would like to train the dataset with the network streams or not. After clicking on 'NO' a Server UI is popped up. First here, we are starting the application without training the dataset. It runs on port number 1123. In the server we can limit the number of clients that can be connected to it.Mention the number of clients to be connected in the box provided and then click 'START' to start the server. Now, the clients can be connected to the server. After limiting the number of clients to be connected, click on 'SET' and then specify the name of the client and then click 'CONNECT' so that the client can be connected to the server.As mentioned, maximum number of 4 clients only can be connected to the server. The server does not allow connection of more than 4 clients. The client which is connected first to the server will act as a leader and here since client 'A' connects first, it acts as leader. To add new clients click on 'START-NEW-CLIENT'.



```
Initiliazing Network Streams Data
Fetched Network Stream Data
Completed Stream Evaluation
Starting Similar Stream Merges
Completed Merges
[root:\] Correctly Classified Instances      22248               98.687  %
[root:\] Incorrectly Classified Instances      296                1.313  %
[root:\] Kappa statistic                         0.9732
[root:\] K&B Relative Info Score           2178084.7025 %
[root:\] K&B Information Score              21478.5884 bits      0.9527 bits/instance
[root:\] Class complexity | order 0         22231.1215 bits      0.9861 bits/instance
[root:\] Class complexity | scheme           2019.8069 bits      0.0896 bits/instance
[root:\] Complexity improvement     (Sf)    20211.3146 bits      0.8965 bits/instance
[root:\] Mean absolute error                     0.0175
[root:\] Root mean squared error                 0.0933
[root:\] Relative absolute error                 3.564  %
[root:\] Root relative squared error            18.8394 %
[root:\] Total Number of Instances           22544
[root:\]
[root:\] 0.9884564386553312 0.9893824654539777 0.9875321436920439
[root:\]
[root:\] Duration: 1941.769598604 seconds
```

DataSet Small-Network-Data

System Enhanced

Training Confirmation

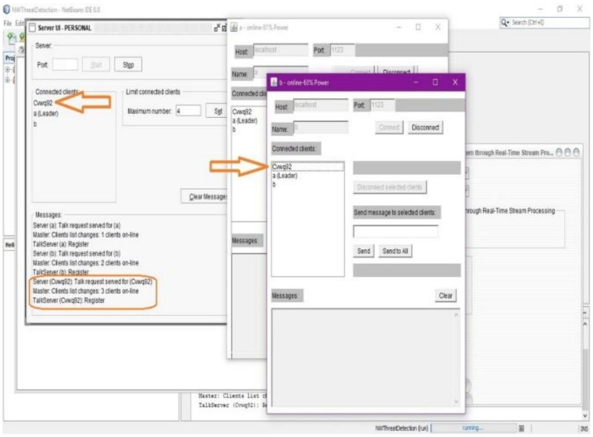
Would You Like To Train With Network Streams

Selected Testing file : sd/KDDTest.arff

NWThreatDetection - NetBeans IDE 8.0
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
<default config>
Search (Ctrl+I)
Classification Log
An Accurate Threat Detection System through Real-Time Stream Pro...
An Accurate Threat Detection System through Real-Time Stream Processing
Select A Dataset
Select An Execution Environment
Real-Time C
Yes
No
Start
Quit
Start-New-Client
NWThreatDetection (run)
running...
INS

Figure 2. Starting the application without training dataset

We can see that 2 clients 'A' and 'B' are connected to the server. But suddenly a new client named 'Cvwq92' is connected to the server automatically without any manual intervention. The server window is showing that there are 3 clients online now as shown in figure 3. As the new client is an unauthorized person who is connected to the server, the leader A has the right to disconnect the selected client as shown in figure 4.



Maximum number: 4

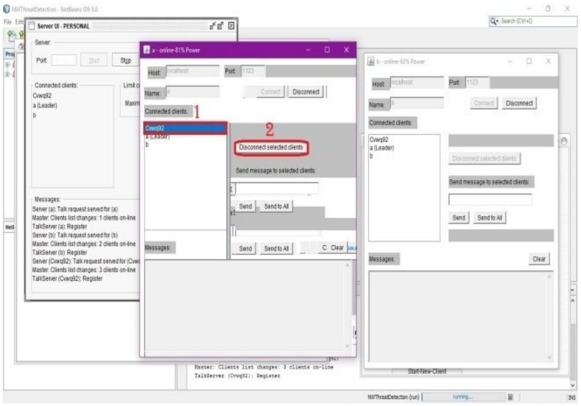
Connected clients:
Cvwq92
a (Leader)
b

Messages:

```
Server (a): Talk request served for (a)
Master: Clients list changes: 1 clients on-line
TalkServer (a): Register
Server (b): Talk request served for (b)
Master: Clients list changes: 2 clients on-line
TalkServer (b): Register
Server (Cvwq92): Talk request served for (Cvwq92)
Master: Clients list changes: 3 clients on-line
TalkServer (Cvwq92): Register
```

Host: localhost Port: 1123

Figure 3. Unauthorized client connected to the server



```
Messages:
Server (a): Talk request served for (a)
Master: Clients list changes: 1 clients on-line
TalkServer (a): Register
Server (b): Talk request served for (b)
Master: Clients list changes: 2 clients on-line
TalkServer (b): Register
Server (Cvwq92): Talk request served for (Cvw
Master: Clients list changes: 3 clients on-line
TalkServer (Cvwq92): Register
```

Figure 4. Leader A disconnecting the selected client from the server

As shown in figure 5 we are starting the application by training our dataset. A training confirmation dialog box is shown which asks us whether we would like to train our dataset with network streams or not. The dataset will be trained with network streams and the output will be shown in classification log as in figure 6

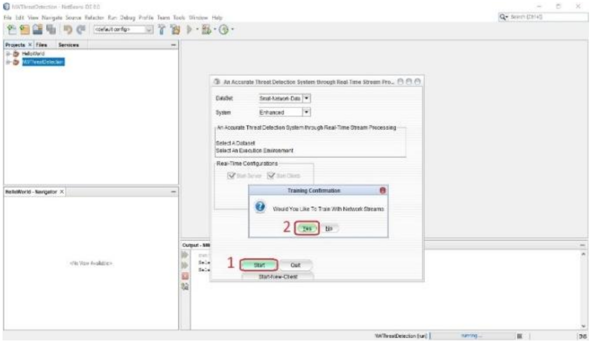
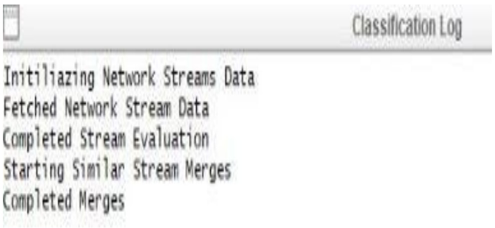

An Accurate Threat Detection System through Real Time Stream Pro...
DataSet
Small-Network-Data
System
Enhanced
Select A Dataset
Select An Execution Environment
Real-Time Configurations
Training Confirmation
Would You Like To Train With Network Streams
2
Yes
No
1
Start
Quit
Start-New-Client

Figure 5. Starting the application by training the dataset

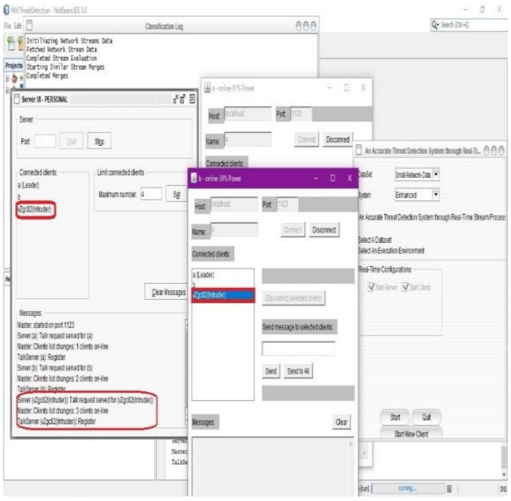
.



```
Initiliazing Network Streams Data
Fetched Network Stream Data
Completed Stream Evaluation
Starting Similar Stream Merges
Completed Merges
```

Figure 6. Classification log output

Now here also without any manual intervention, a client with name 'vZgc82' is connected to the server automatically as shown in figure 7. This time as the dataset is trained with the network streams it is displaying us that it is an intruder who is connected to the server. The leader 'A' can disconnect the intruder from the server. Not only intruder, leader 'A' can disconnect other clients from the server also.



```
Messages:
Master: started on port 1123
Server (a): Talk request served for (a)
Master: Clients list changes: 1 clients on-line
TalkServer (a): Register
Server (b): Talk request served for (b)
Master: Clients list changes: 2 clients on-line
TalkServer (b): Register
Server (vZgc82(Intruder)): Talk request served for (vZgc82(Intruder))
Master: Clients list changes: 3 clients on-line
TalkServer (vZgc82(Intruder)): Register
```

Figure 7. Intruder is connected to the server

Leader 'A' disconnects the intruder from the server. Two clients 'A' and 'B' are present in the server. In figure 8 we can see that client 'B' is sending the message "hiiii" to all the clients in the server. As only one client is left in the server, client 'A' receives the message and after receiving the message the leader 'A' is disconnecting client 'B' from the server. In the figure 9 we can observe that client 'B' has gone offline. It also receives a message that "the leader has forced your disconnection".

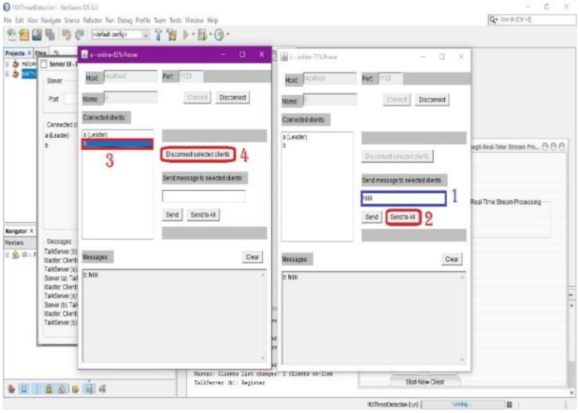


Figure 8. Disconnecting client B from the server

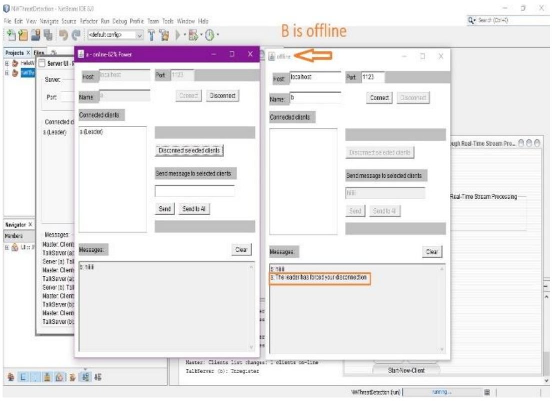


Figure 9. Client B is offline

IV. CONCLUSION

Now a days, intrusion which affects the privacy and security of the system, has become a major concern. Threat detection system is a way of analyzing the intrusion activities and threats that occur in the network. This work proposes a real-time stream processing threat detection system. The proposed system uses the network stream data architecture, which combines stream and batch processing. The off-line batch processing allows automatic updates of machine learning algorithms to be used in real-time. To evaluate the proposed system performance, a dataset was created from real network traffic. The captured packets were abstracted into labeled flows with 24 features. To increase system efficiency, the dataset dimensionality was reduced by using the principal component analysis algorithm. Three machine learning algorithms, decision tree, neural network and support vector machine, were implemented to perform real-time classification and two of them obtained accuracy above 95%. Moreover, we implemented an anomaly detection algorithm to detect zero-day attacks with a good trade-off between false positive and attack detection rates, when adjusting the threshold. The anomaly detection algorithm is also adaptive, since parameters are updated in real-time.

REFERENCES

1. A. Lobato, M. Andreoni Lopez, and O. C. M. B. Duarte, “An Accurate Threat Detection System through Real-Time Stream Processing,” (GTA), Universidade Federal do Rio de Janeiro (UFRJ), Tech. Rep, 2016.
2. Chen, Zhijiang, Hanlin Zhang, William G. Hatcher, James Nguyen and Wei Yu, “A Streaming-Based Network Monitoring and Threat Detection System” IEEE 14th International Conference on Software Engineering Research, Management and Applications (SERA), June 2016.
3. Sonali Rathore, Prof. Amit Saxena, Dr. Manish Manoria, “Intrusion Detection System on KDDCup99 Dataset: A Survey”, International Journal of Computer Science and Information Technologies (IJCSIT), Vol. 6, pg. 3345-3348, 2015.
4. P.Natesan, P.Balasubramanie, “Multi Stage Filter Using Enhanced Adaboost for Network Intrusion Detection”, International Journal of Network Security & Its Applications (IJNSA), Vol.4, No.3, May 2012.
5. Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani, “A Detailed Analysis of the KDD CUP 99 Data Set”, IEEE Symposium on Computational Intelligence in Security and Defense Applications, CISDA 2009.

Dr G Krishna Kishore, M.Tech Ph.D working as an Associate Professor, in V.R.Siddhartha Engineering College has 15 Years of research experience in the area of Mobile Ad-hoc networks and has more than 20 research publications.

D Suresh Babu, M.Tech working as an Assistant Professor in V.R.Siddhartha Engineering College has 3 years of research experience in the area of Data Engineering.

S.Ravi Kishan, M.Tech (Ph.D) working as an Associate Professor in VR Siddhartha Engineering College has 10 years of research experience in the area of Data Analytics with more than 10 research publications.

Published By:
Blue Eyes Intelligence Engineering & Sciences Publication