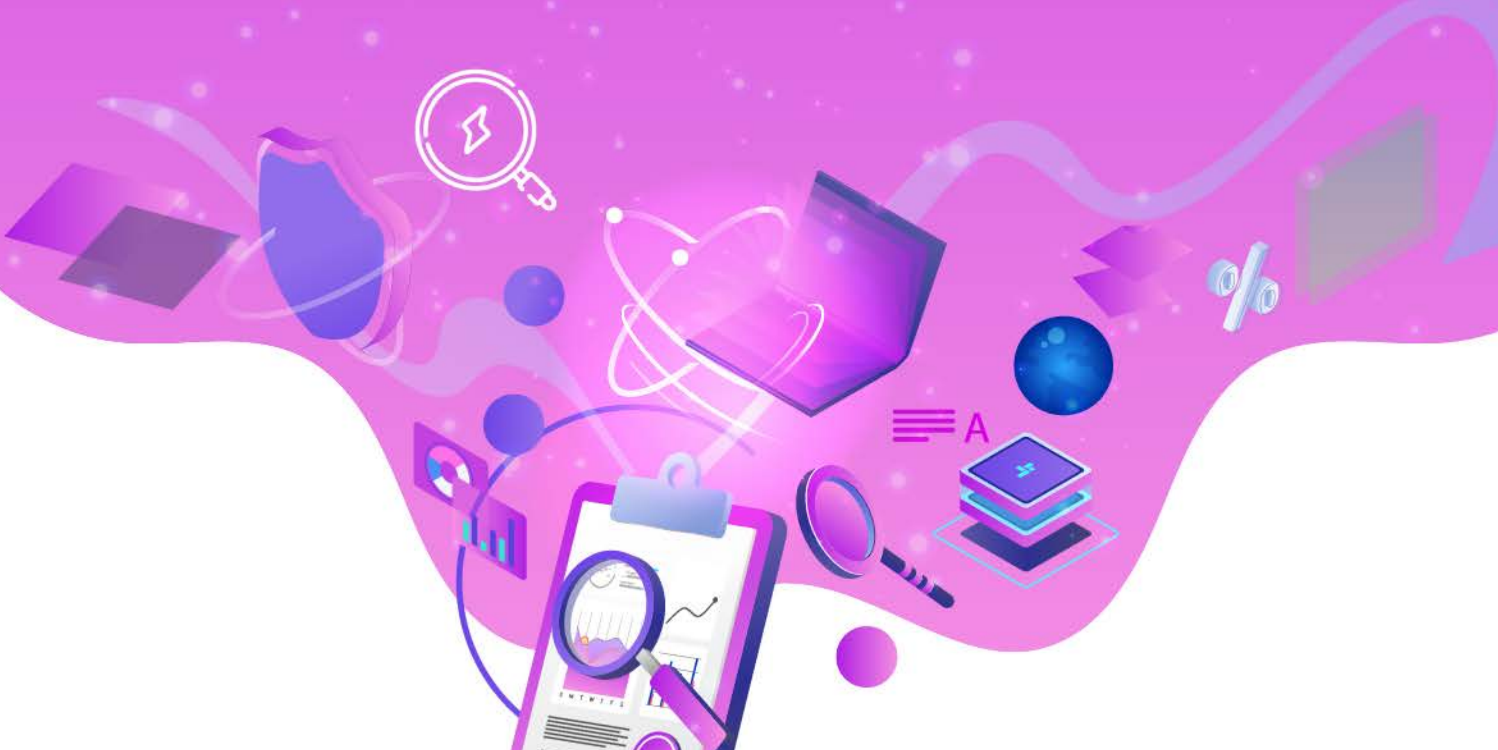
National Centre of Excellence
CYBERSECURITY TECHNOLOGY AND ENTREPRENEURSHIP

DSCI
PROMOTING DATA PROTECTION
A NASSCOM® Initiative

Ministry of Electronics & Information Technology
Government of India
सत्यमेव जयते

SERI
Security Education, Research, & Innovation Conference
Promoting Cybersecurity Education, Research and Innovation

SERI Conference Proceedings



PARTNER – TIER 1
Microsoft

PUBLICATION PARTNER
Taylor & Francis
Taylor & Francis Group

SHORTLISTED PAPERS

Theme	Authors	Institute/ Organization
Secure Data Aggregation Process Using Memetic Algorithm in IoT Enabled Wireless Sensor Networks	Dr.D.Nethra Pingala Suthishni Mr.K.S.Senthil Kumar	Avinashilingam Institute for Home Science & Higher Education for Women, Coimbatore Hindusthan College of Arts & Science, Coimbatore
Detection of Malicious Insider in Cloud Environment based on Behavior Analysis	Dr. G. Padmavathi Dr. D. Shanmugapriya Asha S	Avinashilingam Institute for Home Science & Higher Education for Women, Coimbatore
Evaluation of Supervised Machine Learning Classifiers to Detect Mobile Malware	Dr. G. Padmavathi Dr. D. Shanmugapriya Roshni	Department of Computer Science Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore Department of Information Technology Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore Center for Cyber Intelligence Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore

Secure Data Aggregation Process Using Memetic Algorithm in IoT Enabled Wireless Sensor Networks

DR. D. NETHRA PINGALA SUTHISHNI

MR. K.S. SENTHIL KUMAR

Secure Data Aggregation Process using Memetic Algorithm in IoT Enabled Wireless Sensor Networks

Dr.D.Nethra Pingala Suthishni[a] and K.S.Senthil Kumar[b]

[a]*Assistant Professor, Department of Information Technology,*
Avinashilingam Institute for Home Science and Higher Education for Women,
Coimbatore, India.
nethra_it@avinuty.ac.in

[b]*Assistant Professor, PG & Research Department of Computer Science*
Hindusthan College of Arts & Science
Coimbatore, India
senth4u4m@gmail.com

Dr.D.Nethra Pingala Suthishni, working as Assistant Professor in the Department of Information Technology, Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, India with a teaching experience of around 7 years. She has completed her Doctorate in Computer Science in March 2021and her research areas include network security and soft computing.

K.S.Senthil Kumar, working as Assistant Professor in PG & Research Department of Computer Science, Hindusthan College of Arts & Science, Coimbatore, India with a teaching experience of 10+ years. He is pursuing his doctorate in Computer Science in network security. His research areas include artificial intelligence and machine learning.

Secure Data Aggregation Process using Memetic Algorithm in IoT Enabled Wireless Sensor Networks

Abstract: Over the last 10 years, the Internet of Things (IoT) acts as a backbone for entirely connected sensor devices to achieve integrated communication settings and platforms, both virtual and real-world, in terms of distributed systems. Wireless Sensor Networks (WSNs) tends to be a critical component of the Internet of Things (IoT). The Internet of Things (IoT) monitors the surroundings, collects information, and sends it to a Base Station (BS). WSN routing protocols are suited for IoT environments. However, due to the heterogeneity of nodes, WSNs do not work optimally. Because the Internet of Things is a de-centralised network, the network senses the information and transmits it to the base station. From this network, small sensor nodes consume more energy, which appears to be a serious issue. They are susceptible to a variety of security breaches because of wireless transmission channels and the possibility of deployment in harsh settings or unsupervised areas. In addition, the installed security systems in these contexts have inherent drawbacks. As a result, such systems are susceptible to cyber security threats. To improve the network's performance and to overcome cyber security risks, a new algorithm called Memetic algorithm is proposed in this research. Memetic algorithm is one of the best algorithms in terms of security breaches. To avoid network partitioning, the algorithm is based on a routing mechanism and uses a mobile sink for data gathering. The NS2 Simulator is used to simulate the proposed approach. The experimental findings are compared to existing algorithms to show that the suggested technique is effective against common security threats like traffic interception and ransomware. Additionally, the suggested approach improves throughput, network longevity, packet loss, end-to-end delay, and energy consumption. Node authentication, data integrity, anti-compromise, and traffic analysis resistance are all features of the proposed system.

Keywords: Wireless Sensor Networks (WSNs), IoT, Cyber Security, Security Breaches, Memetic Algorithm.

1. Introduction

WSNs are a crucial component of Internet of Things (IoT), which uses IoT equipments to monitor and provide users with useful data about their environment. Smart home technology, forest surveillance, medicare, satellite agriculture, and digital city are all examples of IoT applications. IoT devices or sensors in the monitoring region sense physical characteristics like pressure, humidity, and temperature, and transmit them to the Base Station (BS) via single-hop or multi-hop communication. Sensor nodes have energy restrictions in WSNs for IoT.

The clustering mechanism has several shortcomings, including an energy hole and network segmentation which severely reduces network longevity and subsequently leads to possibilities of cyber threats. In this scenario, the need for securing the sensor network from various cyber threats arises. These threats may either harm the individual nodes on the network or the entire sensor network. More data packets are relayed by Cluster Head (CH) neighboring sensor nodes than faraway sensor nodes, resulting in the clustering mechanism's early mortality of the cluster head neighboring sensor nodes. Under these circumstances, faraway sensor nodes require enough energy for data transmission, but they are powerless to transfer data packets to the cluster head and base station due to a lack of network structure, resulting in an energy hole. The network has been separated into several independent sections because of the lack of energy. Dividing the network into partitions thus leads to cyber security threats and hence the cyber risks grow in complexity. Due to the lack of appropriate communication links, certain portions are unable to communicate to the Base Station.

Encryption and decryption are two data transformations defined by a cryptosystem. To generate cipher text, unencrypted text, i.e. the plain text to be sent, is encrypted using an encryption key. The decryption key is used to transform cypher text to plain text, which is the original data. Symmetric cryptography is defined as encryption and decryption keys that are the same or can be deduced from one another.

One key, referred to as the private key, is kept private, while the other, referred to as the public key, is made public. The public key encrypts the communication, which can only be decrypted with the private key. As a result, anyone with the public key cannot decrypt the encrypted message, allowing for safe communication. The most prominent public key algorithm called RSA (short for Rivest, Shamir, and Adleman) [8] is a cryptographic algorithm that is adapted.

The deployed nodes are usually immobile, and the sink position in traditional data collection algorithms is usually fixed. Because of the high overhead of relaying messages, Sensor nodes that are closer to the static sink consume more energy than those that are farther away. Because of the issue, Sensor nodes that are close to a static sink expire more faster than other nodes, lowering network longevity [1]. And, this concern eventually increases the possibility of cyber threats.

Heterogeneous nodes offer much more data processing and communication capabilities than common nodes. Heterogeneous nodes, on the other hand, are expensive, thus it's critical to figure out how to maintain a healthy energy balance and increase network longevity [2].

Data redundancy keeps track of every transfer in the network and helps to prevent Denial-of-Service attacks. Authentication, risk assessment, and data security are application layer security requirements for the safeguard of cybernated data that is critical for environmental security. To allow access to data and information, external authentication is required.

The remainder of this article is organised as follows: Section II examines the associated work flow of IoT enabled Wireless Sensor Networks, the Evolution of Cyber Threats and Network Security. The Problem Statement is explained in Section III. Section IV proposes the proposed work and algorithm details. Section V elucidates the Simulation Specifications and Performance Metrics details. The results and discussion are presented in Section VI, and the paper's conclusion is explained in Section VII.

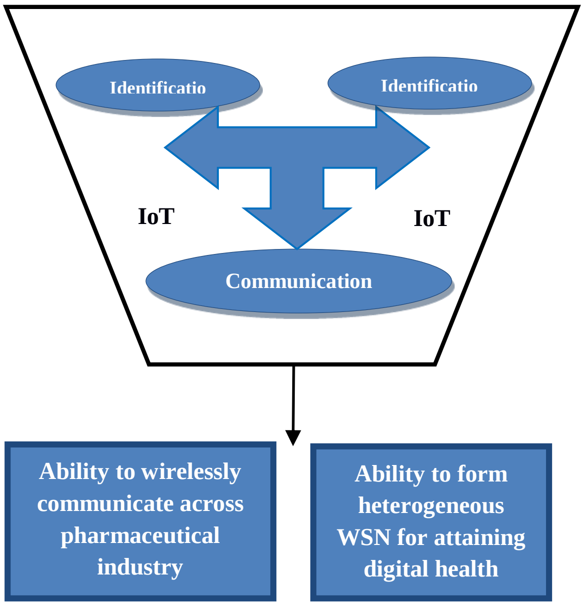


Figure 1 IoT in WSN

2. Related Work

Chuan Zhu et. al [1] explain with mobile sinks, provide a data collection technique that is both high-availability and location-predictive. Sensor nodes use time synchronisation to detect the location of mobile sinks, which minimises sensor node energy usage for sink location updates. When the network uses high-available data collecting technique, and if a few of the mobile sinks are inaccessible, it can continue to collect data. Furthermore, the energy consumption of nodes near resident places can be balanced by changing the moving trajectory of a mobile sink. Data uploading to the mobile sink is substantially slowed when there is an issue at the resident point.

Chunlin Li, et. al [2] The cluster routing approach for WSNs is introduced for explaining how to stabilize energy and lengthen network lifespan. It is taken into account a group of heterogeneous nodes and cluster heads. To begin, construct a layout of optimum node placement for varied nodes. Second, a cluster routing approach for WSNs is suggested, that combines Heterogeneous Routing Algorithm (HRA) with the LEACH-C cluster heads selection mechanism. Finally, detailed testing is done to compare the efficacy of our suggested routing approach to that of numerous previous traditional routing methods. The routing algorithm can significantly extend the lifetime and stability of a network. It can also drastically reduce energy consumption.

Gurbinder Singh Brar et. al [3] explain a PEGASIS-DSR optimised routing protocol (PDORP) based on hybrid optimization, this merged the proactive and reactive routing systems' cache and directed transmission ideas. The simulation results for our proposed protocol show a decrease in end-to-end transmission time and bit error rate without sacrificing energy efficiency. To obtain a fast and damage-free path with reduced transmission delay,, both proactive and reactive routing methodologies were applied in PDORP.

Jianhua Huang et. al [4] proposes ASGRP a circular segment grid clustering-based low-energy multiple hop routing algorithm for WSNs. The recommended approach enhances the formation of clusters in a WSN. The main concept is to divide the network into circular sector grids, with the BS serving as the central point of each circular zone. The nodes of each circular sector grid are grouped into clusters. In comparison to a four-sided grid, the distance between grid nodes and the BS can be kept close to the same using a circular sector grid. The grid is created by calculating the inclination between the border where the base station is located and the route from the nodes to base station. To improve data transmission efficiency amongst the base station and CH nodes, we devised an intermediate level multiple hop routing approach. The recommended routing approach minimises transmission energy usage while uniformizing energy use. Multi-hop ASGRP, EEBCDA, CAMP, and EEMRP achieves more consistent energy utilisation, greatly extend network lifetime, and have greater scalability in networks of varying numbers and sizes..

Jianhua Huang et. al [5] explain to extend the network lifetime, We presented a grid clustering-based energy-efficient multi-hop routing system. The proposed protocol separates the network region across unequal grids to produce clusters of varying levels. Because grids located further away from the sink are larger, cluster distribution is better acceptable, and the consumption of energy is spread evenly among the functional nodes. Sensor nodes in the grids nearest to the sink send data directly to the sink, but sensor nodes in the succeeding grids send data via multi-hop transmission. To decrease the complexity of the election, a management method based on CM nodes is proposed, which may eliminate the randomness of electing CH nodes, optimise the position of CH nodes, and lower the communication cost of member nodes within clusters. In terms of energy efficiency, network lifetime, and scalability, the proposed protocol surpasses current protocols.

Mbanaso. U. M et.al [6] The risks and hazards of IoT systems are studied, and a new policy-driven requirement for overcoming reliance, privacy, and confidentiality difficulties in disseminate scenarios is offered. In a tenure where policies must pool resources and interrelate without a glitch to solve difficulties across various areas, digital entities should become more trustworthy, dependable, and secure, assuring dynamic security and safety from end to end. It devises a framework that permits Internet of Things (IoT) entities to express their abilities and specifications in a fine-tuned strategy construct for mutual and rapid negotiation of proven qualities and resources. It also enables inspection and hassle resolution, that are acute realistic factors in IoT atmospheres, as well as unified trust, privacy, and secrecy resolution.

Moosa Ayati, et. al [7] explain reducing energy usage in WSNs is critical since it extends the network's lifetime. Clustering is a powerful technique for extending the life of a network. LEACH is the most widely used grouping method currently available. In a wireless sensor network with minimal energy consumption, data disbursement in the BS is reduced. One of the most critical factors that impacts network longevity and raises the danger of data loss is data overhead. Data Packets collide with one another when data overhead develops, and some of them may be lost. As a result, the missing packets must be retransmitted. The nodes lose energy as a result of these retransmissions. To control data overhead, a reliable method for WSNs is necessary. In the suggested SCHFTL, the super CH is liable for data transmission to the BS. The recommended solution reduces data overhead, forfeiture, and relaying, resulting in a longer network lifetime.

Preetha. M et. al [8] explains the encryption as a minor variation on the well-known and widely used RSA algorithm-OAEP. Even in the multi-query context, the security of the RSA problem remains significantly tied to the complexity of the RSA problem, according to this scheme. The RSA gives the business application the highest level of security. Furthermore, without using hybrid or symmetric encryption, this approach can be utilised to encrypt large messages.

Ranida Hamidouche et. al [9] describes wireless sensor networks, which are employed in a range of critical applications like health care and military monitoring, have a restricted energy capability. To accomplish effective energy utilisation, LECR-GA, a networked protocol based on genetic algorithms, is described. Using the suitable chromosomal exemplification, fitness function, and Genetic Algorithm operations, we were capable of obtaining with least complexity, longer system functionality and highest data rate. From the experimental results, the suggested algorithms beat GAEEP and GABEEC concerning with energy consumption and throughput.

Se Ra Oh et. al [10] demonstrate how single M2M (i.e., Mobius) O-Auth 2 based security module is designed to offer privacy and authorisation, two crucial security objectives for security in IoT and protected meshing amongst IoT platforms. Examples include a block of secure components, a credential transfer, and a security component reply. A resource request from an unauthorised user will be blocked by the one M2M security module, whereas a resource demand from an authorised user will be granted through.

Yiqun Zhang, et. al [11] explain the limited computing resources and required flexibility, IoT security presents numerous issues. ASICs and coprocessors on the market today have a number of drawbacks. In this study, we offer recryptor, a new architecture that effectively supports huge vector calculations for crypto algorithms by leveraging in memory and near memory computing. When compared to baseline CPU architecture, it preserves programmability and saves approximately 80% of runtime and energy. Recryptor is a worthy transition in terms of balanced region, energy, throughput, and configurability.

3. Problem Statement

The clustering process causes a number of issues, including energy hole problem and network segmentation, which drastically increases the possibilities of threats in the sensor networks and reduces the network lifetime. The sensor nodes relay more data packets with nearby Cluster Head (CH) than far away sensor nodes, resulting in the early mortality of the CH. The energy hole problem in the network is divided into numerous distinct pieces. Due to a shortage of adequate communication links, the segments are unable to interconnect with the Base Station (BS). Sensor node has enough energy for transmitting data, but they are powerless to transmit data packets to the cluster head and BS due to a lack of network structure. To tackle the energy hole problem and network partitioning issue, the clustering approach uses a lot of extra energy. Rather, they overburden the system with hardware resources.

4. Proposed Work and Algorithm

4.1 Proposed Work

There are three types of the phases explained in IoT-enabled wireless sensor networks. The first phase is Memetic algorithm based cluster creation. The second phase is data collection phase based mobile sink based data collecting and third phase is RSA algorithm based cryptography process.

Memetic Algorithm is used to elect CHs during the cluster building phase using a strict memetic approach. Data collecting describes a cluster creation method based on a memetic algorithm, followed by a data collection scheme based on a mobile sink. RSA based cryptography process algorithm uses key-based encryption and decryption mechanism to improve cryptography

4.2 Algorithm

4.2.1 RSA Algorithm

A single integer is encrypted and decrypted using this algorithm. Converting larger or various bits of information into (possibly big) numbers is the first step towards encoding them. Because RSA is a slow method, it is generally used to encrypt the key of a quicker algorithm. This supplemental technique uses the key decrypted by RSA to decode the rest of the message.

To solve the factoring problem and decipher the algorithm analytically, one must first resolve the factoring problem (identify the two prime numbers that provide the given result when multiplied). The problem is difficult to solve by brute force when the chosen numbers are large enough, and there is currently no easier analytic solution.

4.2.2 Memetic Algorithm

The purpose of this phase is to identify network CHs that can significantly lower deployed node energy consumption. As a result, in a heterogeneous WSN environment, we apply a memetic algorithm, an intelligence-based optimization mechanism that produces a better optimal solution, for optimal CH selection.

A genetic algorithm is being used to introduce a local search strategy. It's an evolutionary computation meta-heuristic algorithm. It's a search-based optimization task inspired by nature. It gives solutions that are close to ideal. The term “optimization” refers to the process of maximizing or minimizing objective functions based on input parameters. They offer a variety of options. These solutions are then mated and mutated, resulting in offspring, a process that occurs over several offsprings.

Each person is graded on their fitness, and the fittest are picked to be parents. As a result, genesis will continue until it reaches the termination condition. A genetic algorithm optimises continuous and discrete functions while also providing a group of solutions rather than a individual solution which improves with time. When a big number of factors are involved, it is an excellent option. It is a good fit for NP-hard problems.

The cluster head is found inside the network's nodes using a memetic technique. It is split into two sections. A node can participate in CH selection if its energy is greater than network's usual energy. The initial bit series of the chromosome is created, with cluster head receiving a cost of '1' and the remaining nodes receiving a value of '0.' When the aforementioned criteria is no longer met, the CH is chosen using a memetic algorithm. The steady state phase occurs before crossover and mutation and is defined as the application of a fitness function.

Initial Process: The parameters that affect network performance are selected. After that, the parameters are assigned a starting value. Several parameters include the amount of sinks, the location of every node, the network size, the sum of chromosomes, the population dimensions, the crossover ratio, the rate of mutation, and the generation number. Weight quantities are also initialised.

Fitness Function: The objective functions which aid in application and optimization of the solution to the intended outcomes. Then, using an iterative fitness function, chromosomes are converged to superior solutions generation after generation. It needs to get to the solution rapidly and be related to the goal.

Cluster Formation: Initially generate the x and y co-ordinate value of the nodes. Initiate the Sink node. Sink node is divided into no of grid cells. The grid cell takes the midpoint to form a group of the nodes for grid formation. Create the grid formation to assign a grid id and node id to all nodes. In the network area, grids are four-sided and immobile in nature. During data collection at mobile sink, it could be movable or inactive. The midpoint of grid cell is first picked as CH among all the other nodes in the grid cell (CH). CH is a node with the shortest distance to the midway. Node-id and grid ids are allocated to each node in the network.

Data Collection Phase: A mobile sink follows the trajectory to a rendezvous position and then During the data gathering phase, transmits a data broadcasting message in communication range 2R. To upload all of the network's information, the MS sends this data uploading message to the CHs. A mobile phone ID and location information are included in the transmission. This message will only be received by CHs that are all within the mobile sinks' communication range. The CH examines its buffer state after receiving the data upload message. If the cluster head’s existing buffer is free, the MS's data uploading message is simply ignored. The CH sends a response to the Mobile Sink (MS) if the CH's current buffer state is not empty. The CH ID, remaining energy, and position information are all included in the reply message. When MS receives a response message from the cluster head, it assigns each replying CH a specific time window. Every CH communicating with the mobile sink has their own time window. The CH can only respond during the time frame allotted to it. Among the CHs in the queue, suitable scheduling is carried out. In transmission range 2R, a scheduling communication is broadcast through mobile sinks. The individual CH data uploading plan is included in the scheduling message.

Path Updating: Initialize Source Node as S and Distance Node as D values. The source node and destination nodes check the present Cluster head of the group nodes. Source node is not up to destination node. The algorithm verifies the CH group once the results have been sent. The CH is included in the source input, which is the value of the source node. Finally, CH is sent to the Destination Node.

4.3 Algorithm Steps

4.3.1 Enhanced RSA Algorithm Steps

Step 1: Generate two different primes k and y

Step 2: Give a key to present the file location

Step 3: Read the input key value and split character wise

Step 4: Calculate modulus a = k + y

Step 5: Calculate totient $(n) = (k-1) * (y-1)

Step 6: Select the pu integer of the public exponent, so that 1 < pu < $(a) and gcd ($(a), pu) = 1

Step 7: Calculate a value for pr for a private exponent such that pr = pu-1 mod $(n)

Step 8: Separated characters are arranged in ascending order and with new characters write new line

Step 9: Public key = [pu, a] e

Step 10: Private key = [pr, a] d

4.3.2 Enhanced Memetic Algorithm Based Cluster Formation Steps

Step 1: WSN is formed by deploying BS and M_l.

Step 2: Setup Phase

Step 3: for every n=1 to n do

Step 4: while $S_{ei} \leq E_{th}$ do

Step 5: Take part in the election to choose CH

Step 6: if $S_{pun} > S_{pui}$, $1 \leq j \leq m$, $j \neq n$ then

Step 7: S_{pun} Elect CH_n of C_n

Step 8: CH_n bit = = 1 $and\ \forall S_i\ bit == 0$

Step 9: end IF

Step 10: end FOR

Step 11: Steady State Phase

Step 12: Centralized configuration of the cluster by Mobile Sink

Step 13: After determining the node distance and energy value, division of each grid cell by node id and grid id is performed

Step 14: Node selects the CH (Cluster Head) to determine the value to compare the lowest distance and maximum energy

Step 15: Apply MA and CH_n Selection to the next generation population

Step 16: CH_n bit = = 1 and $\forall S_i \ bit == 0$

Step 17: if CH_n energy < E_{th} then

Step 18: Calculate $F_n = (\omega 1 * \ \alpha + \ \omega 2 * \ \beta + \ \omega 3 * \ \gamma + \ \omega 4 * \ \delta)^{-1}$

Step 19: Apply crossover between two leading competitors

Step 20: After a mutation, calculate a new chromosome

Step 21: Perform a local search

Step 22: Choose a new CH_n

Step 23: CH_n bit = = 1 and $\forall S_i \ bit == 0$

Step 24: End

4.3.3 Enhanced Data Collection Phase

Step 1: if MS is at a RP then

Step 2: broadcasting a message containing the mobile sink ID and location information at range 2R

Step 3: Include the source and destination at the start

Step 4: After all nodes have been checked for source node, initialise for loop and then include all nodes

for every i= 0; i < n; i++

Step 5: If CH receives message from MS about uploading

Step 6: if CH buffer is empty then

Step 7: Locate the source node within the cluster's group

if (sn! = d && sn = = CH) then

Step 8: From source to CH, a cluster message is sent

Step 9: if (dn = = CH1 && CH! = CH1) then

Step 10: CH sends a cluster message to CH1, and CH1 sends a cluster message to CH2

Step 11: Discard the message

Step 12: else

Step 13: transmits a message to mobile sink with the cluster head ID, remaining energy, and position information

Step 14: end

Step 15: When MS receives a response message from CH,

Step 16: assign a specific time slot

Step 17: End

4.3.4 Flow Diagram

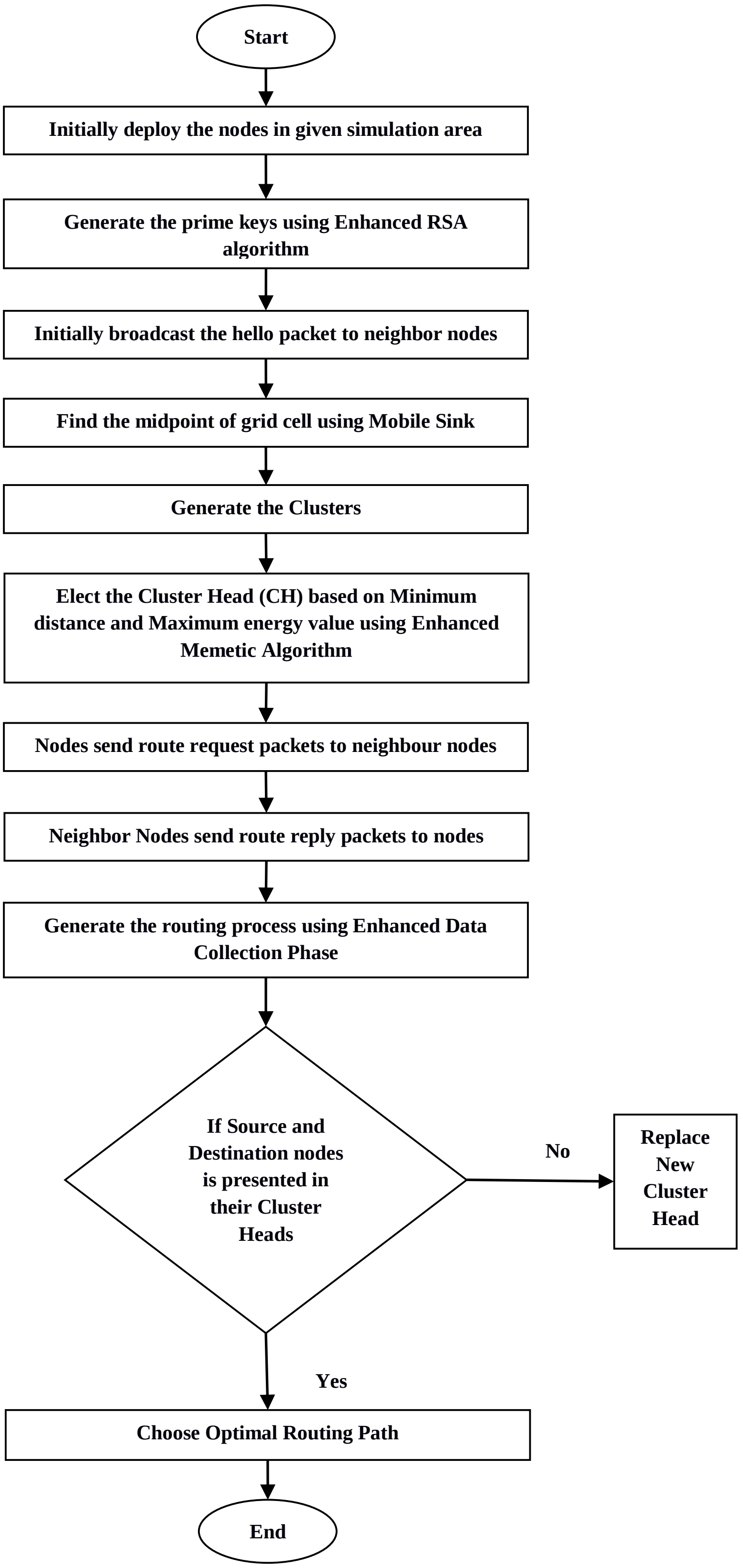


Figure 2 Flow Diagram of Proposed Method

5. Simulation Analysis

5.1 Simulation Specifications

S. No	Specifications	Values
1	Simulator Type	NS - 2
2	Channel Type	Wireless
3	Number of Nodes	100
4	Traffic Model	CBR
5	Simulation Area	2250m * 2250m
6	Transmission range	400m
7	Routing Protocol	DSR
8	MAC Protocol	802.11
9	Simulation Total Time	100ms

Table 1 Simulation Specifications

5.2 Performance Metrics

The simulation performance of the Enhanced Memetic Algorithm for IoT enabled WSN is illustrated in this section. The simulations are performed on a network simulator (NS-2). The network simulator NS2 is discrete event simulation software for performing network simulations. It performs the simulation events such as sending, receiving, forwarding and dropping in the network packets. Some of the protocol's performance measures were discussed. The following are the performance metrics that were utilised to make the comparison.

- End-to-End Delay
- Packet Delivery Ratio
- Network Longevity
- Energy Consumption
- Throughput
- Packet Lost

5.2.1 End-to-End Delay

End-to End-Delay is the duration it acquires for a packet to traverse from its source node to its destination node. The formula takes into account all of the period of time taken up by the router to seek best route in network usage, as well as propagation, processing, and end to end delay for packet pac sent by node k as a source node and successfully received at destination node.

$$\textit{End to End Delay}_{kpac} = \textit{starttime}_{kpac} - \textit{endtime}_{kpac}$$

The start-time kpac is the time when packets sent by node k are successfully received at the target region, and the end-time kpac is the time when packets sent by the node k are successfully received at target area.

5.2.2 Packet Delivery Ratio

The total amount of packets sent among the source node and destination node is referred as the packet delivery ratio (PDR). It's used to figure out how much data is lost in packets during transmission. While being transmitted from source to destination node, few packets could be missing or channelled improperly to alternative nodes; in order to identify this loss, Generally, packet delivery ratio is computed and assesses both the correctness and efficiency of adhoc algorithms for routing. Higher packet delivery ratio is usually anticipated in any network. This is considered to be the best transmission.

PDR = Total no. of packets received / Total of no.of packets sent

5.2.3 Energy Consumption

At the start of the simulation, the node has an initial value, which represents the node's energy level at the start of the simulation. In the equation, this is referred to as Initial Energy. The term "energy" in the simulation environment refers to the amount of energy in a node at any one time, which is provided by battery power or another source. When a node in the simulation environment hits 0 energy, it is no longer capable of transmitting or receiving packets, and it becomes idle.

Energy of Nodes = Current_Energy - Initial_Energy

5.2.4 Network Longevity

The maximum duration of time, the installed sensors in the simulation can observe the phenomena of interest among the nodes. The higher the Network Lifetime range, the better the performance.

Network Longevity = 100 - ΣAi

ΣAi - Average of energy

5.2.5 Throughput

The total number of packets communicated amongst the communication time, or effective data delivery within the scheduled time, is the throughput. The transmission value is calculated using the standard rate of correctly transmitted packets from source node to destination node. It is expressed in bits/bytes per second.

Average throughput = Total no. of packets successfully transferred-Total no. of packets/Transmitting Time

5.2.6 Packet Lost

The discrepancy between the total number of packets transmitted and the total number of packets received is known as packet loss.

Packet Lost = Number of Packets Transmitted – Number of Packets Received

6. Results and Discussions

In NS2, the proposed retrievals' experimental outcomes are evaluated and analyzed. The simulation area is 2250 × 2250 meters. Then, 100 nodes are deployed in the given simulation area. The suggested algorithm is then applied to the metrics propagation data that has been acquired.

The suggested and existing methods are measured with the help of the classification methods like that Enhanced Memetic Algorithm and Enhanced Data Collection Phase and Enhanced RSA Algorithm. These metrics results of the different method are discussed below.

Time (ms)	End-to-End Delay	
	Zigbee	**ZRP**
0	10	10
20	14.6321	12.5533
40	16.7812	14.0445
60	21.4381	19.5593
80	23.8123	21.6712
100	27.3541	24.9813

Table 2 End-to-End Delay

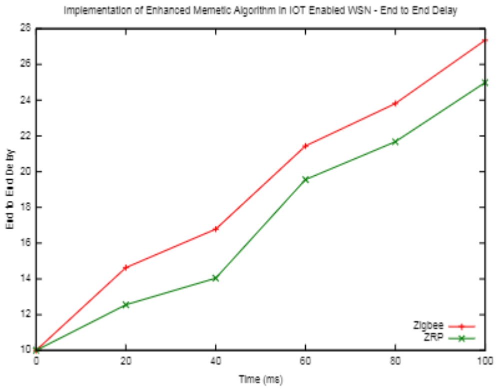


Figure 3 End-to-End Delay

Time (ms)	Energy Consumption	
	Zigbee	**ZRP**
0	100	100
20	80.32	77.80
40	76.43	74.56
60	74.24	71.72
80	71.39	68.79
100	68.34	65.97

Table 3 Energy Consumption

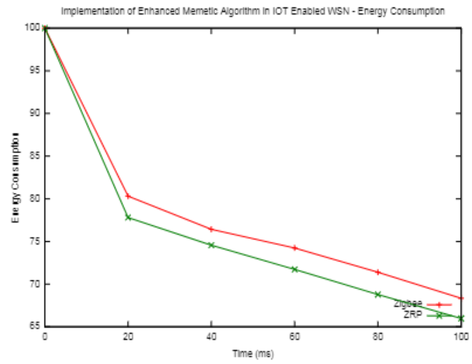


Figure 4 Energy Consumption

Time (ms)	Packet Delivery Ratio	
	Zigbee	**ZRP**
0	1.57	1.57
20	1.7625	1.7824
40	1.8254	1.8640
60	1.9354	1.9640
80	1.9952	2.0532
100	2.1089	2.1358

Table 4 Packet Delivery Ratio

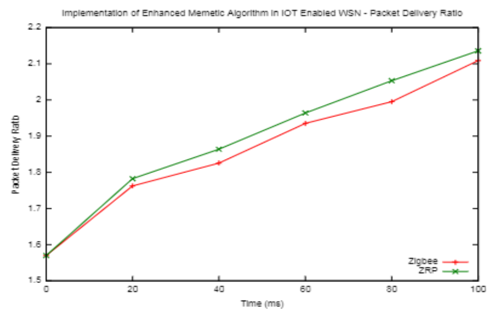


Figure 5 Packet Delivery Ratio

Time (ms)	Throughput	
	Zigbee	**ZRP**
0	27	27
20	43.12	45.19
40	50.36	53.16
60	58.53	60.18
80	63.12	66.78
100	68.29	70.52

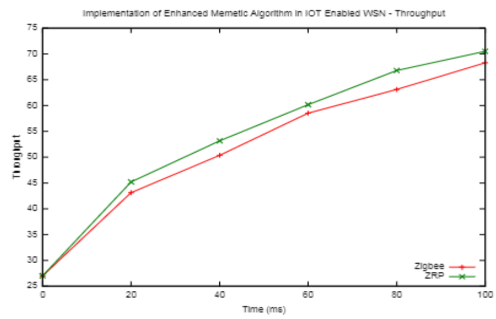


Figure 6 Throughput

Table 5 Throughput

Time (ms)	Network Lifetime	
	Zigbee	ZRP
0	0	0
20	1500	2000
40	5200	6800
60	10250	12000
80	12090	14000
100	15035	16000

Table 6 Network Lifetime

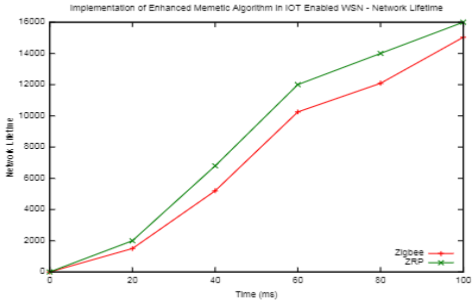


Figure 7 Network Lifetime

Time (ms)	Packet Lost	
	Zigbee	ZRP
0	500	500
20	450.29	425.28
40	382.54	350.17
60	290.26	265.23
80	260.48	210.98
100	190.95	150.28

Table 7 Packet Lost

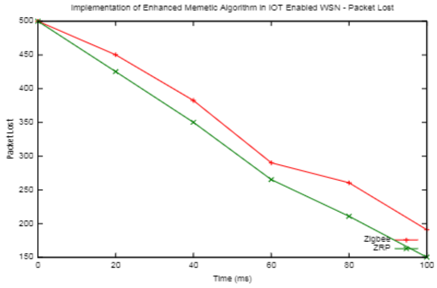


Figure 8 Packet Lost

7. Conclusion

To increase network performance in IoT-enabled wireless sensor networks, a memetic algorithm with a mobile sink-based data gathering technique is used. For CH selection and cluster development, the proposed method contains a unique clustering strategy that combines an algorithm called Memetic with Powell's mechanism for conjugate gradient. The overhead of cluster creation messages is greatly reduced. The memetic method, contrastingly, employs Powell's mechanism for conjugate gradient to determine the ideal amount of cluster heads for reducing data communication loss in energy. In the recommended technique, the mobile sink finds the best data collecting channel to accumulate data from a large number of CHs, resulting in a significant reduction in end-to-end delay. The RSA method encrypts bits and combines them with a public key, reducing decryption time and increasing cryptosystem strength. As a result, the chance of cyber threats or network dangers is decreased. The proposed method's performance has been assessed using a variety of simulation results. The suggested approach surpasses the challenge of throughput, network longevity, consumption of energy, packet delivery ratio, loss of packets, and throughput. The suggested solution outperforms existing methods in terms of network performance and network security as cyber dangers become more sophisticated.

References

[1] **Chuan Zhu, Kangning Quan, Guangjie Han and Joel J.P.C. Rodrigues**. "A High-Available and Location Predictive Data Gathering Scheme with Mobile Sinks for Wireless Sensor Networks", Computer Networks, Vol. 145, November 2018.

[2] **Chunlin Li, Jingpan Bai, Jinguang Gu, Xin Yan and Youlong Luo**. "Clustering Routing Based on Mixed Integer Programming for Heterogeneous Wireless Sensor Networks", Ad Hoc Networks, Vol. 172, April 2018.

[3] **Gurbinder Singh Brar, Shalli Ra,i, Vinay Chopra, Rahul Malhotra, Houbing Song and Syed Hassan Ahmed**. "Energy Efficient Direction-Based PDORP Routing Protocol for WSN", IEEE Access, Vol. 4, June 2016.

[4] **Jianhua Huang, Danwei Ruan and Weiqiang Meng.** "An annulus sector grid aided energy-efficient multi-hop routing protocol for wireless sensor networks", Computer Networks, Vol. 147, December 2018.

[5] **Jianhua Huang, Yadong Hong, Ziming Zhao and Yubo Yuan**. "An energy-efficient multi-hop routing protocol based on grid clustering for wireless sensor networks", Cluster Computing, Vol. 20, June 2017.

[6] **Mbanaso. U. M and Chukwudebe. G. A**. "Requirement Analysis of IoT Security in Distributed Systems", 2017 IEEE 3rd International Conference on Electro- Technology for National Development (NIGERCON), IEEE, Vol. 5, Issue 7, February 2018.

[7] **Moosa Ayati, Mohammad Hossein Ghayyoumi and Atiyeh Keshavarz-Mohammadiyan**. "A fuzzy three-level clustering method for lifetime improvement of wireless sensor networks", Annals of Telecommunications, Vol. 73, March 2018.

[8] **Preetha. M and Nithya. M**. "A Study and Performance Analysis of RSA Algorithm", International Journal of Computer Science and Mobile Computing, Vol. 2, Issue. 6, June 2013.

[9] **Ranida Hamidouche, Zibouda Aliouat and Abdelhak Gueroui**. "Low Energy-Efficient Clustering and Routing Based on Genetic Algorithm in WSNs", Springer, Mobile, Secure, and Programmable Networking, January 2019.

[10] **Se-Ra Oh and Young-Gab Kim.** "Development of IoT Security Component for Interoperability", 2017 13th International Computer Engineering Conference (ICENCO), Vol. 12, Issue. 4, February 2018.

[11] **Yiqun Zhang, Li Xu, Qing Dong, Jingcheng Wang, David Blaauw, and Dennis Sylvester.** "Recryptor: A Reconfigurable Cryptographic Cortex- M0 Processor With In-Memory and Near-Memory Computing for IoT Security", IEEE Journal of Solid-State Circuits, Vol. 9, Issue. 3, February 2018.

Detection of Malicious Insider in Cloud Environment based on Behavior Analysis

PADMAVATHI G
SHANMUGAPRIYA D
ASHA S

Detection of Malicious Insider in Cloud Environment based on Behavior Analysis

Padmavathi G
Department of Computer Science
Avinashilingam Institute for Home Science and Higher Education for Women
Coimbatore, Tamilnadu, India
padmavathi_cs@avinuty.ac.in

Shanmugapriya D
Department of Information Technology
Avinashilingam Institute for Home Science and Higher Education for Women
Coimbatore, Tamilnadu, India
shanmugapriya_it@avinuty.ac.in

Asha S
Department of Computer Science
Avinashilingam Institute for Home Science and Higher Education for Women
Coimbatore, Tamilnadu, India
20phcsf005@avinuty.ac.in

***Abstract*— Insider threat is one of the most stimulating security threats in an organization that possesses sensitive information. In an organization, detecting malicious insider threats is more challenging due to the behavioral changes of malicious insider. To avoid the sensitive information leakage that causes enormous loss, detecting the malicious insider within an organization is necessary. The principal focus of this paper is to find the user's unauthorized activity by analyzing their behavior on website i.e., websearch analysis. To find the user's unauthorized activity by analyzing each user's behaviour, such as the website activity of each individual. The user is classified as a genuine user or malicious user based on user's websearch behavior. This paper proposes an insider threat detection framework to analyze and detect the malicious insider threat within an organization using user's statistical behavior analysis.**

Keywords—Insider threat detection, behavior analysis, Malicious Insider.

I. Introduction

In the rapidly developing world, all business organizations and the corporate sector recommend and enhance the business by possessing the Internet-as-a-solution. Cloud computing is a framework that accomplishes rapid provisioning on-demand charge restricted self-service resources to its user over the Internet. The migration of an organization to the cloud faces some severe threats due to its changing environment. One of the most challenging security threats faced by an organization is Malicious insider or an authorized individual employee who attempts to gain access to confidential information. Recent reports show that 53% of organizations and 42% of U.S. federal agencies suffer from insider threats every year [1]. Insider threat-related activities can be carried out intentionally, such as information system sabotage, intellectual property theft, and disclosure of classified information, as well as unintentionally, such as careless use of computing resources [1]. The primary goal of malicious insiders is to cause economic and reputation loss by leaking sensitive data to the competitive organization. So, it is significant to detect the malicious insider threat in an organization. one of the way for detecting the malicious insider is by analysing the behavior of the user. This paper proposes the detection of malicious insider activity using behavior analysis. This paper aims to explore the insider data using the logging behavior of employees within the organization. The entire paper is organized into four sections. Section II tabulates the literature study on malicious insider detection method. Section III explains the overview of proposed methodology. Section IV discusses the obtained result. Section V concludes with possible scope for future enhancement.

II. Literature Review

The primary concern is to analyze the CERT data to detect the malicious insiders using logging behavior analysis. Table I describes the work done in the field of various Insider Threat detection frameworks.

Table I. Literature Review

S. no	Author	Insider Threat Detection Framework applied	Algorithms applied	Observations
1.	Jiang et al. (2018)	User Behavior Analysis	XGBoost, SVM, Random Forest (RF)	User behaviour analysis using XGBoost outperforms other algorithms based on F-measure up to 99.96% to detect the malicious activity using CERT dataset [5]
2.	Eberle and Holder. (2009)	Graph based anomaly detection	GBAD-MDL, GBAD-P (probability) and GBAD-MPS (maximum partial substructure)	Graph-based anomaly detection using MDL algorithm identifies the graph-based anomalies such as email, phone traffic and business process to detect the insider threat than Probability and MPS algorithm [6]
3.	Liu and et. (2018)	Anomaly-based Insider detection	Deep Autoencoder (AE)	Deep A.E. detects all malicious insider activity with a reasonable false positive rate using US-CERT data [7]
4.	Diop and et. (2019)	Ensemble Learning Behavior Anomaly Detection Framework	IForest, One-Class SVM, Local outlier factor (LOF), Elliptic envelope (EE), artificial neural network (ANN), Gaussian naive Bayes(Gnb)	Ensemble learning behavior using Gbc algorithm outperforms other algorithms with (75%-99%) in both unsupervised learning based testing and supervised learning based testing. An ANN followed this with (60%-99%) result in both tests [8].

S. no	Author	Insider Threat Detection Framework applied	Algorithms applied	Observations
			, Bagging classifiers (Bgc), random forest (RF) and gradient boosting (Gbc)	
5.	Jiang et al. (2019)	Graph Convolutional Network	RF, SVM, Logistic Regression (LR), Convolutional Neural Network (CNN), Graph Convolutional Network (GCN)	GCN performs better than other algorithm based on accuracy, precision and recall to detect malicious insider and fraud activities [9].
6.	Kim et al. (2019)	User Behavior Modeling and Anomaly Detection Algorithms	Gaussian density estimation, Parzen window density, Principal component	User behavior modelling and anomaly detection using Parzen and PCA provided a better result than other algorithms to detect malicious insider threats [10].
7.	Senator et al. (2013)	Detecting Insider Threats in a Real Corporate Database	IP Thief Ambitious Leader Scenario Detector, File Events Indicator Anomaly Detection, Relational Pseudo Anomaly Detection, Repeated Impossible Discrimination Ensemble, Grid-based Fast Anomaly Discovery given Duplicates (GFADD)	The multiple methods detect the malicious insider threat using computer log activity in an actual corporate database [11].
8.	Lv et al. (2018)	Method based on user and role behavior (MURB) and Anomaly Detection (ADAD)	Isolation Forest	MURB outperforms the ADAD with 80% precision and accuracy for detection of the malicious insider threat using CERT data [12].
9.	Gamachchi and et. (2017)	Graph and anomaly detection Framework	Isolation Forest	The combined graph-based anomaly detection framework identifies 79% of individuals as Genuine users and 31% as malicious insiders with suspicious activity [13].
10.	Liu et al. (2020)	Behaviour analysis	Behaviour analysis	The new behaviour analysis framework named Doc2vec simplifies insider threat detection based on spatial and temporal metrics [14].
11.	Le and Heywood . (2021)	Anomaly Detection for Insider Threats Using Unsupervised Ensembles	AutoEncoder, Isolation Forest, Lightweight on-line detector of anomalies (LODA), Local Outlier Factor (LOF)	Unsupervised ensemble-based anomaly detection using Autoencoder outperforms the other algorithm based on voting metrics to detect the malicious insider threat [15].
12.	Legg (2015)	Behavior based malicious insider threat detection	Visual Analytics	Visual analytics is recommended to detect malicious insider threat activity based on profiling behaviour and selected features as a mitigation strategy [16].

The above table shows that the various types of insider threat framework utilize different user behavior modelling technique to detect the malicious insider. Hence, the behavior analysis is implemented to improve the precise detection of a malicious insider in an organization.

III. METHODOLOGY

The Following Fig.1 shows the proposed malicious insider threat framework methodology using behavior analysis to detect a malicious insider threat in an organization.

A. Dataset

Data can be obtained from the monitoring process of the organization, where different log files, such as email and weblogs, firewall logs, network traffic captures, and different types of user records are common [2]. The publicly available dataset gathered from U.S. based Computer Emergency Response Team (US-CERT) is used. It consists of information regarding both malicious insider and genuine user activity. This dataset has been collected from https://kilthub.cmu.edu/articles/dataset/Insider_Threat_Test_Dataset/12841247/1 [2].. The above dataset contains the activity information of individual employees in an organization. It comprises the following data input: (i) user activity log data such as web URL, email, file, access log, and removable device connectivity records. It is dynamic and used for the behavior analysis of users. (ii) Structure and information of user and organization. It is considered as meta information for data analysis. For example, Lightweight Directory Access Protocol (LDAP) is considered as metadata in CERT data.

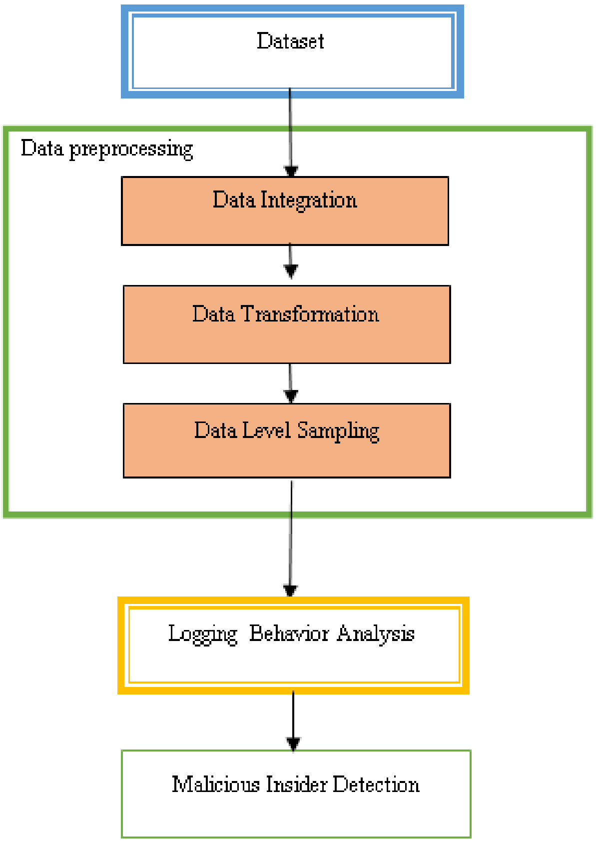


Fig 1: Methodology Overview

In CERT data, the log information of malicious and non-malicious activity is given based on the threat scenario mentioned below.

- Scenario 1: An individual in an organization works after working hours, carries a removable drive and uploads sensitive information to the unauthorized website such as wikileaks.org. Tries to resign from the organization.
- Scenario 2: An individual in an organization visits a job portal website and explores the employment opportunities of a business competitor. An individual's anomalous activity increases the use of a removable device. Resigns the organization in future.
- Scenario 3: Unauthenticated administrator attempts to use unauthorized software to collect sensitive credentials, utilizes the removable device to get sensitive information and tries to access the secure system. Later email the critical information and resigns from the organization.
- Scenario 4: Over three months, an individual often logs in, searches, and forwards sensitive information from other users' computers to personal email addresses.
- Scenario 5: Uploads sensitive information such as documents to Dropbox for personal gain.

Based on the dataset version, the particular scenario is used. The proposed methodology uses the dataset version r3.1. It satisfies the scenario 1 and scenario 2 from the above-mentioned scenarios for further processing.

B. *Data pre-processing*

The primary CERT data contains log details of 516 days, where 4000 users generate 135,117,169 log events. The events are activity including email-based, login-based, device storage-based, HTTP operations, psychometric details, file information and daily log details [3]. The abovementioned five scenarios apply scenario-1 and scenario-2 related malicious insider threat data in this research. Other information is ignored. The selected data undertakes three pre-processing steps to make the data suitable for insider detection. It includes data integration, data transformation and data level sampling.

1) *Data Integration:* The malicious and non-malicious activity information that satisfies the selected scenario are gathered from device connectivity, login status and website operation for detection of malicious insider threat. A simple feature concatenation technique is used to integrate the selected records. Table II demonstrates the details of integrated data.

Table II. INTEGRATED DATA

S. no	Feature name	Explanation
1.	InsiderThreat	Malicious activity or not
2.	Vector	origin of data (HTTP/logon/device)
3.	date	Date
4.	User	The user id of an employee
5.	Pc	Unique identification for each computer
6.	Activity	Actual activity of an employee in the pc

2) *Data Transformation:* The integrated data requires data transformation to encode the absolute value for further processing. The features, namely 'vector', 'pc', 'user' and 'activity' from integrated data is converted into a numerical value. The value of 'date' is converted into a number of epochs. Table III shows the encoded data.

Table III. ENCODED DATA

S. no	Feature	Before Encoding	After Encoding
1.	InsiderThreat	Numerical	Numerical
2.	Vector	Categorical	Numerical
3.	date	Timestamp	Number of epochs
4.	User	Categorical	Numerical
5.	Pc	Categorical	Numerical
6.	Activity	Categorical	Numerical

3) *Data Level Sampling:* Jia et al. (2014) had proposed the solution at the data level for the class imbalance problem is based on sampling methods [4]. It is accomplished using the undersampling technique such as Near-Miss 2. In Near-Miss 2 algorithm, the instance of the majority class was selected if it satisfies the average distance for N outermost instance of a minority class is minimum. In the pre-processed dataset, a feature named 'InsiderThreat' is selected as the target variable where class 0 is majority class non-malicious event and class 1 is

minority class malicious event. After resampling, the majority class instance 0 is restructured and equals the minority class instance 1. Table IV shows the sampled data.

Table IV. SAMPLED DATA

S.no	Training set	Before Sampling	After Sampling
1.	Non-Malicious Majority class instance	(0, 39732)	(0, 268)
2.	Malicious Minority class instance	(1, 268)	(1, 268)

C. Logging Behavioral Analysis

The behavior of each individual in an organization needs to be analyzed to detect the malicious insider threat. Logging behavior analysis using pre-processed data is used to get more insight into the activity of each individual in an organization. Based on the selected scenario, the visited website of each individual is analyzed to identify the malicious user behavior. It is accomplished by analyzing website activity to detect malicious activity. Website activity behavior is an efficient way to collect sensitive evidence from the organization. The website behavior comprises visited website URLs for each individual. Based on the selected scenario, unauthorized access such as www.wikileaks.org and job-related websites is considered an unauthorized malicious activity. Others are considered non-malicious activity. It further conducts an in-depth analysis of user website behavior to detect malicious activity.

IV. RESULTS AND DISCUSSIONS

The following Fig. 2 demonstrates the activity count of genuine users and malicious users using pre-processed data. From the Fig.2, it is observed that the activity of the genuine user (0) is comparatively less than 10 activities per month. In comparison, the total activity count of the malicious user (1) is more than 70 activities per month. Hence, the activity of the malicious user is increasingly high than the average genuine user, and it is required to detect the user who possesses the malicious activity. The users who visit unauthorized websites are categorized as "Malicious" and others as "Non-Malicious". It is required to find the total number of unique activities of each individual in a particular personal computer (pc) for both malicious and genuine users.

This work is supported by the Centre for Cyber Intelligence (CCI), DST-CURIE-AI-Phase II Project, Avinashilingam Institute for Home Science and Higher Education for Women. Coimbatore. Tamil nadu, India.

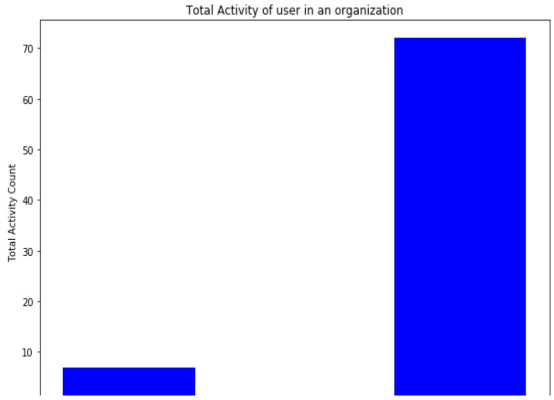


Fig 2: Activity count of Genuine user and malicious user

The following Fig. 3 visualize the total number of activity counts for both malicious and genuine users. The non-malicious insider may possess the malicious activity. So, categorize the user based on malicious activity to foresee the user activity in Fig.4

Fig 3: Total number of activity count for each user.

It pinpoints user's activity based on the frequency of various visited URLs using a particular device. Fig.4 shows that user 162 is an insider who visits the job-related website, namely, http://lockheedmartinjobs.com for 13 times and frequently uses the removable device for connection and disconnection. This is categorized as a malicious insider that satisfies scenario-2 based rule. User 221 is an insider who visits the unauthorized website, namely http://wikileaks.org for 2 times and is categorized as a malicious insider based on scenario-1 based rule.

Fig. 4. Total number of activity count for each user based on category.

The following Fig.5 explains the personal computer used by a malicious user. i.e., user 162 uses the pc, namely 45, to perform the malicious activity. User 221 uses the pc, namely 403, to perform the malicious activity. By converting the numerical value into categorical value, the user id is retrieved. Hence, user CCH0959 and CSF0929 is considered a malicious insider who visits the unauthorized website and performs the malicious activity.

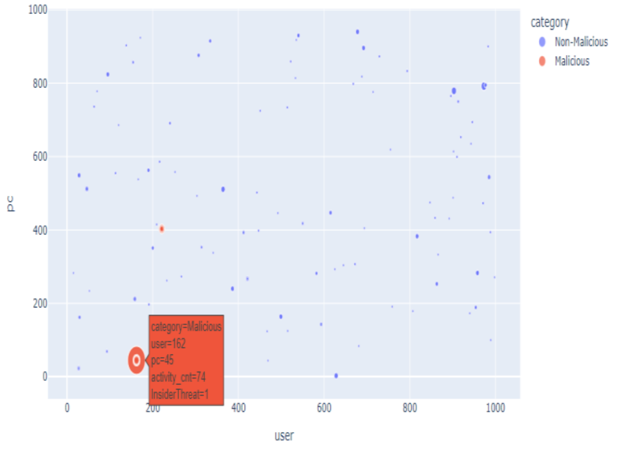


Fig. 5. Personal computer used by Malicious user.

Table V demonstrates the company profile of malicious insiders from LDAP. It shows that the user CCH0959 is Cedric Cyrus Harrison, an Industrial Engineer from the Industrial Engineering department who is considered a malicious insider based on scenario-2. The user CSF0929 is Chaney Sean Fuentes, a Production Line Worker from Assembly Department considered a malicious insider based on scenario-1.

Table V. Information of Malicious Insider

User id	Name	Email	Roles	Depart ment	Supervis or
CCH095 9	Cedric Cyrus Harrison	Cedric. Cyrus.H arrison @dtaa.c om	Industrial Engineer	1– Industria l Engineer ing	Desiree Claudia Booth
CSF0929	Chaney Sean Fuentes	Chaney. Sean.Fu entes@ dtaa.co m	Productio-n Line Worker	3- Assembl y	Theodor e Upton Barry

V. Conclusion and Future Enhancement

In this proposed research paper, the logging behavior analysis is implemented using pre-processed insider threat data to detect the malicious insider threat in an organization. The user who visits the unauthorized data leak websites and job-related websites are considered malicious insiders based on scenario-1 and scenario-2. The predicted malicious insider who possesses malicious activity is correctly detected. It pinpoints the basic information of malicious insiders from LDAP to further mitigate such activity that could happen in the cloud environment. It is highly beneficial in real-time insider threat detection. In future, the graph-based behavior analysis using deep learning can be proposed to detect the malicious insider threat.

References

[1] Collins, M. Common sense guide to mitigating insider threats. CARNEGIE-MELLON UNIV PITTSBURGH PA PITTSBURGH United States, 2016.

[2] Glasser, J., and Lindauer, B. 2013. Bridging the gap: A pragmatic approach to generating insider threat data. *In 2013 IEEE Security and Privacy Workshops*, pp. 98-104. IEEE.

[3] Meng, F., Lou, F., Fu, Y., and Tian, Z. 2018. Deep learning based attribute classification insider threat detection for data security. *In 2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*, pp. 576-581. IEEE.

[4] Pengfei, J., Chunkai, Z., and Zhenyu, H. 2014. A new sampling approach for classification of imbalanced data sets with high density. *In 2014 International Conference on Big Data and Smart Computing (BIGCOMP)*, pp. 217-222. IEEE.

[5] Jiang, W., Tian, Y., Liu, W., and Liu, W. 2018. An Insider Threat Detection Method Based on User Behavior Analysis. *In International Conference on Intelligent Information Processing*, pp. 421-429. Springer, Cham.

[6] Eberle, W., and Holder, L. 2009. Applying graph-based anomaly detection approaches to the discovery of insider threats. *In 2009 IEEE International Conference on Intelligence and Security Informatics*, pp. 206-208. IEEE.

[7] Liu, L., De Vel, O., Chen, C., Zhang, J., and Xiang, Y. 2018. Anomaly-based insider threat detection using deep autoencoders. *In 2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 39-48. IEEE.

[8] Diop, A., Emad, N., Winter, T., and Hilia, M.. 2019. Design of an Ensemble Learning Behavior Anomaly Detection Framework. *International Journal of Computer and Information Engineering 13*, 10: 547-555.

[9] Jiang, J., Chen, J., Gu, T., et al. 2019. Anomaly detection with graph convolutional networks for insider threat and fraud detection. *In MILCOM 2019-2019 IEEE Military Communications Conference (MILCOM)*, pp. 109-114. IEEE.

[10] Kim, J., Park, M., Kim, H., Cho, S., and Kang, P. 2019. Insider threat detection based on user behavior modeling and anomaly detection algorithms. *Applied Sciences 9*, 4018.

[11] Senator, T. E., Goldberg, H. G., Memory, A., et al. 2013. Detecting insider threats in a real corporate database of computer usage activity. *In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1393-1401.

[12] Lv, Q., Wang, Y., Wang, L. and Wang, D. 2018. Towards a user and role-based behavior analysis method for insider threat detection. *In 2018 international conference on network infrastructure and digital content (IC-NIDC)*, pp. 6-10. IEEE.

[13] Gamachchi, A., Sun, L. and Boztas, S. 2018. A graph based framework for malicious insider threat detection. arXiv preprint arXiv:1809.00141.

[14] Liu, L., Chen, C., Zhang, J., De Vel, O. and Xiang, Y. 2020. Doc2vec-based insider threat detection through behaviour analysis of multi-source security logs. *In 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pp. 301-309. IEEE.

[15] Le, D.C. and Zincir-Heywood, N. 2021. Anomaly detection for insider threats using unsupervised ensembles. *IEEE Transactions on Network and Service Management 18*, 2: 1152-1164.

[16] Legg, P. A. 2015. Visualizing the insider threat: challenges and tools for identifying malicious user activity. *In 2015 IEEE Symposium on Visualization for Cyber Security (VizSec)*, pp. 1-7. IEEE.

Evaluation of Supervised Machine Learning Classifiers to Detect Mobile Malware

DR. G. PADMAVATHI
DR. D. SHANMUGAPRIYA
ROSHNI

Evaluation of Supervised Machine Learning Classifiers to Detect Mobile Malware

Dr. G. Padmavathi
Department of Computer Science
Avinashilingam Institute for Home Science and Higher Education for Women
Coimbatore, India
padmavathi_cs@avinuty.ac.in

Dr. D. Shanmugapriya
Department of Information Technology
Avinashilingam Institute for Home Science and Higher Education for Women
Coimbatore, India
shanmugapriya_it@avinuty.ac.in

A. Roshni
Center for Cyber Intelligence
Avinashilingam Institute for Home Science and Higher Education for Women
Coimbatore, India
roshini_cci@avinuty.ac.in

Abstract – Due to the rapid growth of android applications and mobile users in this technological era, there is a large increasing cyber attacks through mobile phones. During Pandemic period, mobile malware attacks are one of the top most cyber attacks observed in android mobile users to steal the user personal credentials by intrusion of adware, spyware, banking malware, SMS malware, riskware, viruses and so on. Machine learning methods are very useful and amicable to detect mobile malwares. Automation of the mobile malware detection is the need of the hour and it is imperative to identify the most suitable machine learning techniques. This paper investigates the evaluation of supervised machine learning algorithms that are applied to detect and classify the mobile malwares. A systematic method of evaluation of supervised machine learning model to detect the malware data points and to classify them into binary classification as malware or benign is essential. The purpose of evaluating the supervised machine learning algorithms is to identify the best supervised machine learning model for mobile malware detection with high efficacy rate. All important performance measures are applied and the entire experiments are conducted using benchmark dataset. Nine Supervised Machine Learning methods are experimented and the results are discussed.

Keywords – Machine Learning, Malware Classification, Mobile Malware, Supervised Learning, Python.

I. INTRODUCTION

A Malware or Malicious Software is one of the most common types of cyber attack which was highly pre-dominant during the pandemic by the means of intruding as a malicious code to take over the system control by monitoring all the user activities and steal the user personal credentials without the knowledge of the user [2] [3] [4].

Mobile malware are the important threats and android based mobile malware are very significant today and they compromise user's credentials through unauthorized access [2] [3] [4]. The aim of the paper is to identify efficient supervised machine learning algorithms to detect mobile malwares. The applied algorithms classify the dataset into malware data and benign data using a systematic approach which is vital towards automation of mobile malware detection.

Automation is the way of handling the problems without human intervention by incorporating AI methods to provide solution to reduce the processing time [9]. The proposed approach experiments nine different supervised machine learning models, evaluate the models and recommends the most efficient supervised ML model for accurate malware detection. The evaluation of the supervised machine learning models is done in terms of performance metrics such as accuracy, precision, recall, F1 score, R2 score, TPR, FPR and ROC [5] [7] [8].

The major contribution of this paper is to devise a systematic methodology to test the supervised machine learning model suitable for android based mobile malware detection. This paper is divided into different phases of Machine Learning work flow. The first step is to acquire the dataset and analyze the data to fit for the further development process. The malware dataset taken for study contains 1,00,000 records with 35 feature attributes. In the second step, data pre-processing methods are applied to check whether the data contains any null values or irrelevant values, which helps to remove the unwanted data values then split the data into training and test data set as in the ratio of 70:30. Third step, applies the feature selection methods to find out the important features of the dataset. Selecting the significant features from the dataset will help to improve the data processing time and provides better accuracy of the ML algorithms. The methodology uses three different feature selection algorithms namely, (i) Univariate Selection (ii) Feature

Importance and (iii) Recursive Feature Elimination. By applying the above mentioned feature selection methods, out of 35 features top 10 important features are selected for effective malware classification. Fourth step, applies supervised machine learning algorithms to identify and classify the data into malware/benign [1] [2] [6] [9] [10]. Following supervised machine learning algorithms are evaluated in this study.

i. Decision tree
ii. Random forest
iii. K – Nearest Neighbors (KNN)
iv. Support Vector Machine (SVM)
v. Naïve Bayes
vi. AdaBoost
vii. Neural Network (MLP)
viii. Logistic Regression
ix. Linear Discriminant Analysis

Fifth step, evaluate the performance of the supervised machine learning models using various evaluative metrics to provide the best model for mobile malware detection [1] [4].

II. Problem Definition

For automation of mobile malware detection, it is necessary to develop a systematic framework. This study applies supervised machine learning algorithms to detect and classify the mobile malware. Nine different supervised machine learning algorithms are implemented and evaluated for performance and the most suitable supervised machine learning models are identified based on their performance.

III. Methodology

A study has been conducted for dynamic behavior based android mobile malware classification using supervised machine learning techniques [2]. The entire methodology is divided into five different phases. The first phase of the work is to acquire the appropriate malware dataset for the problem. The second phase involves data pre-processing to investigate the quality of the data by removing the duplicate records, noisy data and conversion of null values into well defined format. The third phase is to apply Feature Selection methods to find out the best features of the dataset that strives to detect or classify the malware data points with high efficacy rate with less processing time. The fourth phase is building a nine different Supervised Machine Learning Models which automatically detects and classifies the malware data points based on the training data [1] [2] [6] [9] [10]. The fifth phase is to make a comparative analysis between the nine different Supervised Machine Learning models developed in the previous phase by evaluating them using performance metrics to suggest the best classifier model that can detect and classify the malware data points accurately [1] [4]. Figure 1 illustrates the step – by – step methodology followed in this paper.

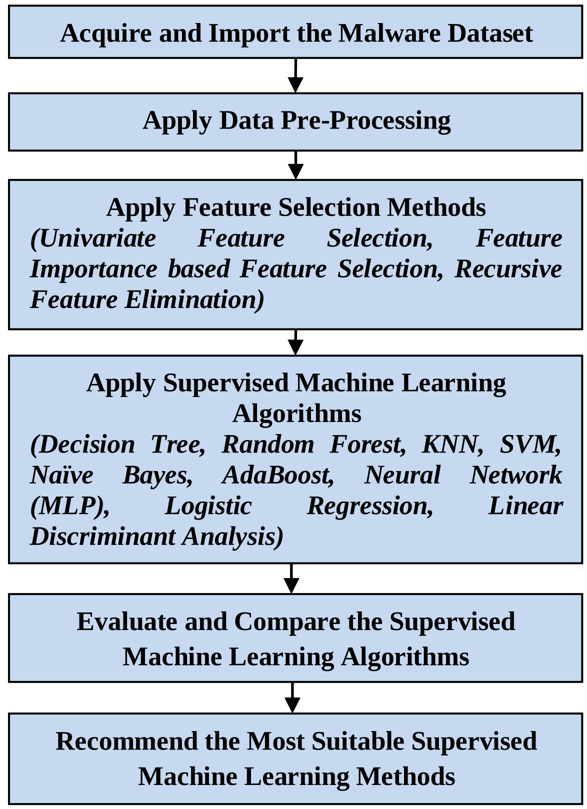


Fig. 1. Proposed Workflow Methodology

The above illustrated methodology is implemented using Python programming in Jupyter Notebook environment.

A. *About the Dataset*

Understanding of the dataset is very important for the accurate prediction and classification. In this work, the benchmark dataset is taken from the kaggle community. The dataset consists of 1,00,000 records and 35 feature attributes. The identification of malware and classification as malware or benign depends on the behavioral features. Table I shows the description of the 35 android kernel attributes involved in the dataset.

TABLE I. Mobile Malware Dataset – Attribute Description

S. No	Malware Data Attributes	Description
1	hash	It is a common method to uniquely identify the malware, which acts as a fingerprint for malware detection. This data contains 100 unique hash values. (i.e. unique apk app names)
2	millisecond	It denotes the time in millisecond ranges from 0 to 999 milliseconds.
3	state	It denotes the flag of unrunable or runnable or stopped tasks as ‘0’.
4	usage_counter	It shows the reference count for task_struct of process as ‘0’.
5	prio	It denotes the system task structure with normal priority value from 0 to 99 and real-time from 100 to 140 as ‘3.07b’.
6	static_prio	It holds the processes’ initial priority value ranging from 14.0k to 31.9k.
7	normal_prio	It denotes the priority without taking RT-inheritance into account as ‘0’.
8	policy	It denotes the scheduling policy used for this process as ‘0’.
9	vm_pgoff	It is the page offset of the area in the file. This is the file position of the first page mapped in this area when a file or device is mapped. It is the first page of the file or device marked as ‘0' in the vm_area.
10	vm_truncate_count	It denotes the vm_areatruncate_count values ranging from 9695 to 27.2k.
11	task_size	It represents the current task size as ‘0’.
12	cached_hole_size	It represents the size of the free address space hole as ‘0’.
13	free_area_cache	It represents the first address space hole ranging from 0 to 515.
14	mm_users	It represents the address space of users ranging from 612 to 995.
15	map_count	It denotes the number of mapping areas ranging from 2588 to 28.2k.
16	hiwater_rss	It represents the high watermark of resident set and sets the peak of resident set size as ‘0’.
17	total_vm	It denotes the total number of memory pages ranging from 4 to 2810.
18	shared_vm	It denotes the number of shared pages ranging from 112 to 120.
19	exec_vm	It represents the number of executable memory pages ranging from 92 to 196.
20	reserved_vm	It represents the number of reserved memory pages ranging from 29 to 755.
21	nr_ptes	It represents the number of page table entries as ‘0’.
22	end_data	It represents the end address of data ranging from 112 to 120.
23	last_interval	It denotes the last

		interval time before thrashing ranging from 0 to 9526.
24	nvcsw	It denotes the number of voluntary context switches ranging from 338k to 385k.
25	nivcsw	It denotes the number of in voluntary context switches ranging from 0 to 365.
26	min_flt	It represents the minor page faults ranging from 0 to 256.
27	maj_flt	It represents the major page faults from 112 to 120.
28	fs_excl_counter	It holds the file system exclusive counter value ranging from 0 to 18.
29	lock	It denotes the file lock as '3.20b'
30	utime	It represents the cumulative time spends on user code ranging from 372k to 422k. (i.e. user time)
31	stime	It represents the cumulative time spent executing system code ranging from 3 to 7. (i.e. system time)
32	gtime	It denotes the group time, cumulative resource counter ranging from 0 to 15. (i.e. guest time)
33	cgtime	It denotes the cumulative group time, cumulative resource counter as '0'.
34	signal_nvcsw	It denotes the cumulative resource counter as '0'.
35	classification	It contains binary classification as malware or benign.

B. Data Acquisition and Data Importing

The first and foremost step in developing a Supervised Machine Learning Model is to acquire the appropriate dataset and import them into the working environment. Python is a platform excellence to support enormous Machine Learning algorithms and it is the suitable environment experiments methodology of machine learning model. In this step, the acquired malware dataset is converted into the CSV file format and imported into the Python Jupyter notebook environment.

C. Data Pre-processing

After importing the dataset, the second consequent step to be performed is data pre-processing [4]. Data pre-processing is the significant step to derive the best results from the classifier models. The data pre-processing is carried out by verifying that the taken dataset contains any null values, un-defined values or irrelevant values which may deviate the results and degrade the performance of the models. The null values are replaced into "zero (0)" or "unknown" data values to overcome the processing issue. Label encoding method is applied to convert the attributes into the integer data type format to develop suitable machine learning model. For developing ML models the entire dataset is divided into training set and testing set in the ratio of 70% for training and 30% for testing [1] [2]. After this step, the dataset is now available in a well defined format for further processing.

D. Feature Selection

Feature selection is an important step of Feature Engineering in Machine Learning process [6] [7] [8] [10]. Feature Engineering is the third step, in the Machine Learning process which extracts the features from the raw dataset to provide labels for appropriate classification. Following feature selection methods are used to choose the top 10 important features precisely out of 35 attributes of given dataset.

i. Univariate Feature Selection

The univariate feature selection approach is a statistical technique for identifying relevant features having a strong association to the target variable. The scikit-learn library in Python includes a function called SelectKBestclass. The SelectKBest class is used in this study to choose the top 10 features out of 35 features in a mobile malware dataset using the chi-square statistical test. . It calculates the Chi-square coefficient for each non-negative feature in the target class and chooses the desired top 10 features with the highest Chi-square scores.

Chi – Square Formula:

$$X^2 = \sum \frac{(Observed\ value - Expected\ value)^2}{Expected\ Value}$$

ii. Feature Importance Based Feature Selection

The feature importance property of the ensemble model is used by the python tool feature importance to pick important features of the dataset. Every property of the dataset is given a score. Greater the score, higher the significance or relevance feature towards target variable. Using the python sklearn ensemble model with the Extra tree classifier method, the system calculates the top 10 features and displays the scores of all the calculated features.

iii. Recursive Feature Elimination (RFE)

Recursively deleting characteristics and then developing a classifier model with the remaining attributes is how the recursive feature elimination is done. The model selects the top 10 features of the data by recursively eliminating the smallest features using the python sklearn feature selection library, RFE method, and the logistic regression classifier algorithm, and the selected features are discernible as True in the support array, and the features are ranked using the choice "1" in the ranking array.

After incorporating three different feature selection methods, the top 10 features are selected based on the highest occurring nature of attributes from the three different feature selection methods. This selection helps in malware data point selection in optimum time. Figure 2 indicates the top 10 selected features out of the 35 features shown in Table I. The Top 10 Selected Features are listed below.

- hash
- state
- static_prio
- vm_truncate_count
- map_count
- total_vm
- reserved_vm
- nvcsw
- nivcsw
- utime

	hash	state	static_prio	vm_truncate_count	map_count	total_vm	reserved_vm	nvcsw	nivcsw	utime
0	30	0	14274	13173	6850	150	210	341974	0	380690
1	30	0	14274	13173	6850	150	210	341974	0	380690
2	30	0	14274	13173	6850	150	210	341974	0	380690
3	30	0	14274	13173	6850	150	210	341974	0	380690
4	30	0	14274	13173	6850	150	210	341974	0	380690
...	...	...	...	...	...	...	...	...	...	...
99995	1	4096	13988	10406	3651	40	90	337688	2	371979
99996	1	4096	13988	10406	3651	40	90	337688	2	371979
99997	1	4096	13988	10406	3651	40	90	337688	2	371979
99998	1	4096	13988	10406	3651	40	90	337688	2	371979
99999	1	4096	13988	10406	3651	40	90	337688	2	371979

100000 rows × 10 columns

Fig. 2. The top 10 selected features based on the three different feature selection methods

E. *Supervised Machine Learning Models*

The fourth phase is Model building. After selecting the top 10 best features out of 35 attributes in the dataset, now the data is ready for developing the model based on Supervised Machine Learning algorithms for the function of classification of mobile malware as malware or benign. Following nine supervised machine learning algorithms are implemented to develop a mobile malware classification model [1] [2] [6] [9] [10].

i. Decision Tree
ii. Random Forest
iii. KNN
iv. SVM
v. Naïve Bayes
vi. AdaBoost
vii. Neural Network (MLP)
viii. Logistic Regression
ix. Linear Discriminant Analysis

The above specified supervised learning algorithms are developed using python Scikit learn library. The general steps involved in developing the above mentioned supervised machine learning models for Mobile Malware classification is explained below:

Step 1: Import the necessary Scitkit learn Supervised Machine Learning algorithm library.

Step 2: Fit the Supervised Machine Learning model to the given training set.

Step 3: Based on the training data, forecast the test result.

Step 4: Then, perform test data predictions and calculate accuracy.

Step 5: Finally, the Supervised Machine Learning Classifier model is ready.

Each Supervised Machine Learning algorithm detects and classifies the malware data points depending upon their own mechanism of calculating the attributes thresholds. The distance calculation measures vary for each algorithm and the weights for the model parameters are determined for final classification. Table II describes the criteria used to detect and classify the mobile malware data points.

TABLE II. Supervised Machine Learning Algorithm's Classification Criteria

Algorithm Name	**Classification Criteria**
Decision Tree	Attribute Selection Measure (Entropy or Gini Index)
Random Forest	Attribute Selection Measure (Entropy or Gini Index)
KNN	Euclidian Distance Measure
SVM	Hyperplane Dimensionality
Naïve Bayes	Bayes Theorem
AdaBoost	Ensemble Learning
MLP	Back Propagation
Logistic Regression	Sigmoid Function
LDA	Collinearity / Class Variance

F. Performance Evaluation Metrics

Following performance metrics are used to evaluate the nine Supervised Machine Learning models [1] [4]:

- Accuracy
- Precision
- Recall
- F1 Score
- R2 Score
- Confusion matrix

The error rates of each Supervised Machine Learning models are also calculated based on the following [9]:

- Mean Absolute Error (MAE)
- Mean Squared Error (MSE)
- Root Mean Squared Error (RMSE)

The results are also visualized in Receiver Operating Characteristic **(**ROC) curve form.

Accuracy: It is defined as the percentage of accurate predictions of the test data.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

True Positives (TP) – These are the accurately predicted positive values, indicating that the actual and predicted class values are both True.

True Negatives (TN) - These are the precisely predicted negative values, indicating that the actual and predicted class values are both False.

False Positives (FP) – When the predicted class is True, but the actual class is False.

False Negatives (FN) – When the predicted class is False, but the actual class is True.

Precision: It is the ratio of successfully predicted positive observations to total expected positive observations. It refers to the classifier's ability to avoid mislabeling a negative sample as positive.

Precision = TP/TP+FP

Recall: It is the proportion of accurately predicted positive observations to all observations in the actual class of observations. It refers to the capacity of classifier on its ability to locate all positive samples.

Recall = TP/TP+FN

F1 Score: It is the weighted average, or Harmonic Mean, of Precision and Recall. As a result, this score takes into account both false positives and false negatives.

F1 Score = 2*(Recall * Precision) / (Recall + Precision)
Or
F1 = 2TP / 2TP + FP + FN

R2 Score: The coefficient of determination (sometimes referred to as R-Squared) is a statistical metric used in regression models to determine how much variance in the dependent variable can be explained by the independent variable.

$$R^2 = 1 - \frac{\sum(y_i - \hat{y})^2}{\sum(y_i - \bar{y})^2}$$

Where, $\hat{y}$ is the predicted value of y and $\bar{y}$ is the mean value of y.

Confusion Matrix: It is a matrix that is used to evaluate classification model's performance for a certain set of test data. Only if the true values for test data are known, it can be determined. Since it displays the errors in the model's performance as a matrix, it is also known as an error matrix. Figure 3 shows the confusion matrix.

N = Total Predictions	Actual Positive (1)	Actual Negative (0)
Predicted Positive (1)	TP	FP
Predicted Negative (0)	FN	TN

Fig. 3. Confusion Matrix

Mean Absolute Error (MAE) (L1 Loss): It is the average of the absolute difference in the dataset's actual and forecasted values. It computes the average of the residuals in the dataset.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}|$$

Mean Squared Error (MSE) (Quadratic Loss or L2 Loss): It is the average of the squared difference between the original and forecasted values. It calculates the residuals' variance.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2$$

Root Mean Squared Error (RMSE): It is the standard deviation of the errors that occur when making a prediction on a dataset. This is the same as MSE (Mean Squared Error). Only the root of the number is taken into account when determining the model's accuracy. The square root of Mean Squared Error is termed as RMSE. It calculates the residuals' standard deviation.

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2}$$

Where, $\hat{y}$ is predicted value of y and $\bar{y}$ is the mean value of y.

Receiver Operating Characteristic (ROC) Curve: It is a graph that shows how effectively a classification algorithm works across all the conceivable thresholds. The graph shows both the true positive rate (Y-axis) and the false positive rate (X-axis). As a function of the model's positive classification threshold, it plots the true positive rate (TPR) against the false positive rate (FPR).

AUC (Area under the Curve): It is a statistic used for calculating the classification model's overall performance based on the area under the ROC curve.

G. *Comparative Analysis*

Based on the above given evaluation measures, a comparative analysis is made between the nine different supervised machine learning models to discover the best classifier model with high efficacy rate. Figure 4 shows the overall accuracy comparison of all supervised machine learning model. Table III and Table IV describes the comparative analysis of nine supervised machine learning algorithms based on the evaluation metrics such as accuracy, precision, recall, F1 score, R2 score and the error rate evaluation based on Mean Absolute Error (MAE), Mean Squared Error (MSE) and Root Mean Squared Error (RMSE).

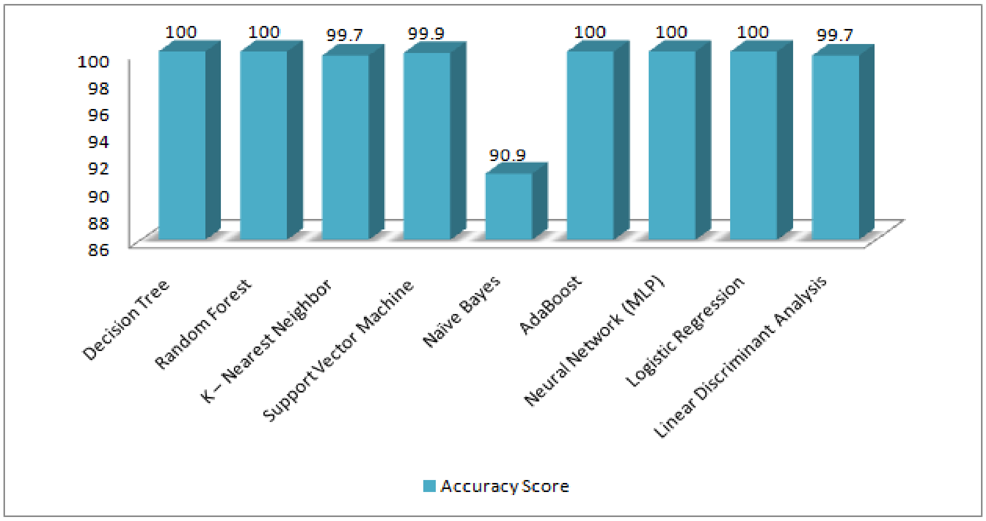


Fig. 4. Overall Accuracy Comparison of all Supervised ML Model

TABLE III. Comparison of Precision, Recall, F1, R2 Score between Supervised Machine Learning Algorithms

S. No	Supervised Machine Learning Algorithms	Accuracy Score	Precision Score	Recall Score	F1 Score	R2 Score
1	Decision Tree Classifier	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
2	Random Forest Classifier	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
3	K – Nearest Neighbor Classifier	0.996	0.996	0.996	0.996	0.986
4	Support Vector Machine Classifier	0.999	1.0	0.999	0.999	0.999
5	Naïve Bayes Classifier	0.909	0.895	0.927	0.911	0.638
6	AdaBoost Classifier	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
7	Neural Network (MLP) Classifier	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
8	Logistic Regression Classifier	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
9	Linear Discriminant Analysis Classifier	0.997	0.999	0.995	0.997	0.990

TABLE IV. Comparison of MAE, MSE, RMSE between Supervised Machine Learning Algorithms

S. No	Supervised Machine Learning Algorithms	MAE	MSE	RMSR
1	Decision Tree Classifier	**0.0**	**0.0**	**0.0**
2	Random Forest Classifier	**0.0**	**0.0**	**0.0**
3	K - Nearest Neighbor Classifier	0.003	0.003	0.057
4	Support Vector Machine Classifier	6.666	6.666	0.008
5	Naïve Bayes Classifier	0.0904	0.0904	0.3007
6	AdaBoost Classifier	**0.0**	**0.0**	**0.0**
7	Neural Network (MLP) Classifier	**0.0**	**0.0**	**0.0**
8	Logistic Regression Classifier	**0.0**	**0.0**	**0.0**
9	Linear Discriminant Analysis Classifier	0.0024	0.0024	0.0496

IV. RESULTS AND DISCUSSION

From the above comparative analysis (Table III and Table IV) made between nine different supervised machine learning algorithms it is evident that the Decision Tree, Random Forest, AdaBoost, Neural Network based Multi Layer Perceptron (MLP), Logistic Regression accurately detect and classify the malware data points with 100% accuracy at 0% error rate for Mobile Malware binary classification. Moreover, KNN algorithm classifies with 99.7%, SVM algorithm classifies with 99.9%, Naïve Bayes algorithm classifies with 90.9%, LDA algorithm classifies with 99.7% accuracy. When compared with Decision tree, Random forest, KNN, SVM, AdaBoost, MLP, Logistic Regression and LDA supervised model classifiers; the Naïve Bayes classifier model performs little low in accuracy for the given dataset.

Figure 5, Figure 6, Figure 7, Figure 8 and Figure 9 shows the results of comparative analysis between the nine supervised machine learning models evaluated based on the performance metrics such as accuracy, train accuracy, test accuracy, precision, recall, F1 score, R2 score, confusion matrix, TPR, FPR and ROC curve.

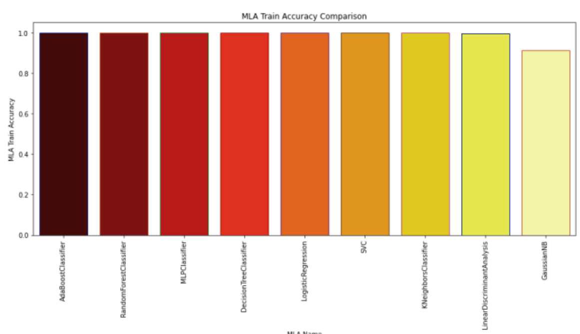


Fig. 5. Train Accuracy Comparison of all Supervised ML Models

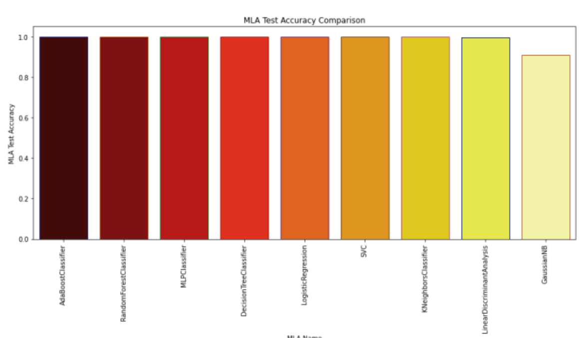


Fig. 6. Test Accuracy Comparison of all Supervised ML Models

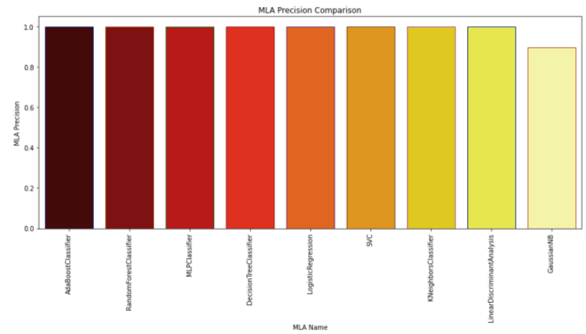


Fig. 7. Precision Comparison of all Supervised ML Models

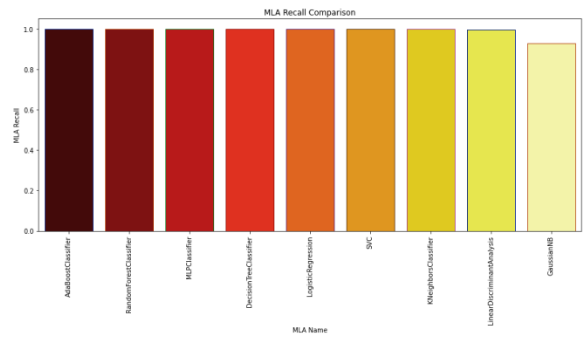


Fig. 8. Recall Comparison of all Supervised ML Models

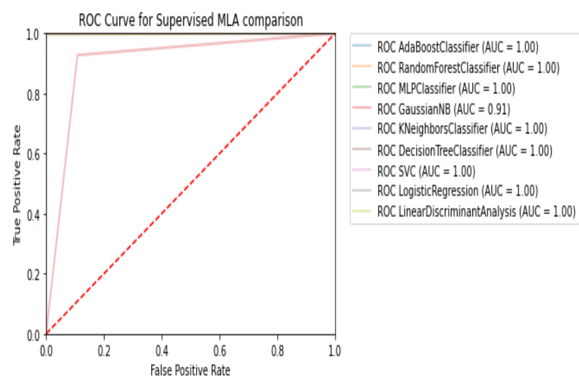


Fig. 9. Overall Comparative Representation of Supervised ML Algorithms in ROC Curve

V. CONCLUSION

Android based mobile malware are of serious threats to the user community due to challenges with user data compromise. Therefore, automatic detection and dynamic analysis of mobile malware are very crucial to develop. The implementation of various Supervised Machine Learning Models using Mobile Malware Dataset, it is observed that, out of nine Supervised Machine Learning algorithms, the Naïve Bayes Classifier Model performs little low when compared with other different eight given Supervised Machine Learning algorithms for the given dataset.

This paper clearly describes the step-by-step implementation procedure to develop an effective Machine Learning Model using a Mobile Malware Dataset in an elaborative way. Moreover, all Supervised Machine Learning algorithms support Malware classification with higher efficacy rate. Thus, this study will be helpful for the learners to understand and implement Supervised Machine Learning algorithms in a systematic way by incorporating mobile malware datasets to arrive at the best results.

VI. References

[1] Prema Agarwa., and Bhushan Trivedi. 2020 (November). Evaluating Machine Learning Classifiers to detect Android Malware. *International Conference for Innovation in Technology (INOCON).* IEEE.

[2] Samaneh Mahdavifar., Andi Fitriah Abdul Kadir., Rasool Fatemi., Dima Alhadidi., and Ali, A. Ghorbani. 2020. Dynamic Android Malware Category Classification using Semi-Supervised Deep Learning. *International Conference on Dependable, Autonomic and Secure Computing, International Conference on Pervasive Intelligence and Computing, International Conference on Cloud and Big Data Computing, International Conference on Cyber Science and Technology Congress*. IEEE.

[3] Muhammad 'Afif Husainiamer., Madihah Mohd Saudi., and Azuan Ahmad. 2020. Classfication for iOS Mobile Malware Inspired by Phylogenetic: Proof of Concept. *IEEE Conference on Open Systems (ICOS).* IEEE.

[4] Jianguo Jiang., Song Li., Min Yu., et al. 2019. Android Malware Family Classification Based on Sensitive Opcode Sequence. *IEEE Symposium on Computers and Communications (ISCC).* IEEE.

[5] Ken, F. Yu., and Richard, E. Harang. 2017. Machine Learning in Malware Traffic Classifications. *Milcom 2017 Track 3 – Cyber Security and Trusted Computing*. IEEE.

[6] Marian Kuhnel., and Ulrike Meyer. 2016. Classification of Short Messages Initiated by Mobile Malware. *11th International Conference on Availability, Reliability and Securit*. IEEE.

[7] Xinjian Ma., Qi Biao., Wu Yang., and Jianguo Jiang. 2016. Using Multi-features to reduce false positive in Malware Classification. IEEE.

[8] George Cabau., Magda Buhu., and Ciprian Oprisa. 2016. Malware Clasiification Based on Dynamic Behavior. *18th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*. IEEE.

[9] Mohammed, S. Alam., and T, Vuong. 2013. Random Forest Classification for Detecting Android Malware. *IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber Physical and Social Computing*. IEEE.

[10] Ivan Firdausi., Charles Lim., Alva Erwin., and Anto Satriyo Nugroho. 2010. Analysis of Machine Learning Techniques used in Behavior-Based Malware Detection. *Second International Conference on Advances in Computing, Control, and Telecommunication Technologies*. IEEE.

VII. Acknowledgement

This work is supported by DST-CURIE-AI project during 2021 – 2023 by Centre for Cyber Intelligence established under Centre for Machine Learning and Intelligence, Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore – 43, Tamilnadu, INDIA.

About National CoE

The National Centre of Excellence for Cybersecurity Technology Development is a joint initiative conceptualized by the Ministry of Electronics & IT (MeitY) and DSCI to catalyse and accelerate cybersecurity technology development and entrepreneurship in India. National CoE works to establish India as a leading hub for cybersecurity capabilities and leverage the expertise to secure Digital India of Tomorrow from cyber threats.

Get in touch with us

Address: 4th Floor, NASSCOM Campus, Plot No. 7-10, Sector 126, Noida, UP -201303
Email: ncoe@dsci.in

Social Media Channels:

@CoeNational nationalcoe nationalcoe