An Efficient Approach for Data Security and Integrity in Multi Cloud Computing

N.M. Mallika* & B. Srinivasan**

*Assistant Professor, Department of Computer Science, Sri Vasavi College (SFW), Erode, Tamil Nadu, INDIA.
E-Mail: mallika.scholar[at]gmail[dot]com

**Associate Professor, PG & Research Department of Computer Science, Gobi Arts & Science College, Erode, Tamil Nadu, INDIA.

Abstract—There is considerable increase in the significance given to Cloud Computing (CC) and it has been getting a rapidly developing attention in the communities of science and industries. It is easy on privileges, convenient on-demand access to a repository of resources and services that are configurable. The services and resources observed on the cloud could be reliably implemented with lesser maintenance expense, effort and better efficiency. Still, the cloud is regarded to be a computing platform with insecurity from the user perspective owing to the absence of security with regard to confidentiality and integrity. Shielding techniques for cloud systems has to be developed so as to protect the sensitive information employing cryptographic methods and in addition keep up the user information with intactness saved on the remote storage by safeguarding it against dangerous entities or behaviors. In order to deal with this issue, the available system proposed a multilevel firefly Algorithm Threshold (MFAT) based secret sharing schema. The generation of the secret key is done by making use of the firefly algorithm. The key that is generated, is utilized for encrypting the file. Thereafter the user divides the secret key into several shares and then distributes them between all resource providers for the purpose of decryption. For improving the level of security and integrity the newly introduced system presented a Multilevel Dragonfly Optimization Threshold schemes (MDOT). In this technical work, Dragonfly Optimization algorithm is employed for the generation of key. After this, MDOT does the encryption of the file exploiting the secret key prior to the distribution of the key shares among the participating resource providers who are assumed to be trustworthy. At last, Adaptive Neuro Fuzzy Inference System (ANFIS) based classifier is applied for performing the check on the cloud user data integrity. Results obtained from experimentation proves that the newly introduced MDOT based secret sharing mechanisms yield high security in comparison with the available state of art techniques.

Keywords—ANFIS; Cloud Computing; Dragonfly Optimization; Integrity.

Abbreviations—Adaptive Neuro Fuzzy Inference System (ANFIS); Cloud Computing (CC); Multilevel Dragonfly Optimization Threshold schemes (MDOT); Multilevel Firefly Algorithm Threshold (MFAT).

I. INTRODUCTION

Many trends are emerging in the era of Cloud Computing [Xiao & Xiao, 1], an Internet-based development and usage of computer technology. The most cheap and mighty processors, along with the software as a service (SaaS) [Sakr et al., 2] computing architecture, are changing the data centers into pools of computing service on a greater scale. The rising network bandwidth and robust still flexible network connections render it even practical that now the users can subscribe to high quality services from data and software, which are stored only on remote data centers. Moving the data into the cloud provides a huge convenience to the users as they don’t need to bother about the sophistications of hardware management directly [Zhang et al., 3; Deyan & Hong, 4].

More and more number of users save their data in clouds, which can be remotely accessed over the Internet. Although the benefits of cloud computing are high, a lot of security issues exist too. One primary security problem is assuring the integrity of the data that is remotely stored. In terms of computer security, data integrity can be described as “the state that exists when computerised data is the same as that in the source documents and has not been exposed to accidental or malicious alteration or destruction” [Talib, 5].

In simple words, integrity indicates the prevention of unauthenticated modification to data. It consists of both intentional modification like insertion or deletion of harmful data and unintentional modification like random transmission error. There is no guarantee for the integrity of the data stored at the untrusted cloud server. For instance, the cloud service providers might make a decision to conceal the data errors...
from the client for the purpose of business profits. They can even delete the data, which are accessed rarely by the client. So the client requires knowledge regarding the remote data by continuous monitoring of the integrity of the stored data. In the case of a remote data integrity checking protocol, initially the data owner (client) stores the data and metadata in the cloud storage (server); afterwards, an auditor (the data owner or another client) can demand the server to show that it can generate the data, which was actually stored by the client; then the server produces a proof of data possession on the basis of the stored data and metadata [Cong et al., 6]. Many researches have been carried out on the design of security and remote data integrity checking techniques that let the data integrity to be checked without having the entire data downloaded.

The data protection in the form of a service offers the data security and privacy over cloud Platform. These services can be given making use of full disk encryption method but it slows down the data access time [Song et al., 7]. Bessani et al., [8] supposed that making use of secret sharing mechanism allowed them to assure the confidentiality in addition to the stored data without having to use the technique of key distribution that allows sharing the secret key between the readers and writers of a data unit. Moreover, the DepSky scheme is capable of choosing which readers can have access to the data unit by means of the access control mechanism of the cloud provider. In order to enhance the data integrity the new system proposed HAIL, which permits the user to save their data over multiple number of servers such there exists a data redundancy. Simple principal behind this technique is to guarantee to data integrity of file through data redundancy. HAIL makes use of Message Authentication Codes (MACs), the pseudorandom function, and universal hash function for ensuring the process of integrity. The proof generated by this technique is size independent of data and also small in size [Bowers et al., 9].

II. Literature Review

Subashini & Kavitha [10] observed that a multi-tenancy can create issues in data security. This interference can be performed either by means of hacking through the loop holes present in the application or by the injection of the client code into the SaaS system. A client can generate a masked code and then inject into the application. In case the application runs this code without doing any verification, then a high risk of intrusion arises into other’s data. Data access has also been talked of and it is seen that data access problem is chiefly related to the security policies offered to the users when the data is accessed.

Ruj et al., [11] developed a new model for data storage and access in clouds, their mechanism prevents the storage of several encrypted copies belonging to the same data. Here, in this framework, the cloud stores the encrypted data (without being capable of decrypting them) so as to secure the storage of data. The important novelty in this model is the addition of Key Distribution Centers (KDCs). DACC (Distributed Access Control in Clouds) algorithm is proposed by using attribute-based encryption, in which one or more number of KDCs distribute the keys to data owners and users. KDC might yield access to certain fields in all the records.

Mechmet Yilidiz et al., [12] introduced a dynamic security model for the purpose of cloud security. This model is on the basis of eight aspects and comprises of four layers. Network, Storage, Servers and Application layers. It is inclusive of one enterprise level principles at the highest level along with a system management aspect. It also consists of two variations of dynamic security types: horizontal and vertical. The horizontal kind is particular to every layer end to end. Here the horizontal dynamic security policy for storage just covers the security objects corresponding to storage. The vertical kind is developed in order to have the interfaces between layers covered. Few security objects between servers and storage might be belonging partially to every layer. The vertical dynamic policies guarantee that any common object or exception gets covered.

Ram & Sreenivasa [13] an developed a technique for achieving security by means of encryption and decryption of user’s data making use of cryptographic co-processor, a reliable third party service. The cloud provider must enquire the customer if his data requires to be segmented or not. The data of the customer are split into pieces having a constant size known as data chunks; the chunk size is dependent on the bandwidth available. Once the segmentation is done, a header and a tail will get added to every data chunk and it will be randomly distributed in different databases. A log file serving as an index to get the chunks has to be generated and stored in a master database, which can be accessed only by user who is highly privileged, as it is hard for hackers to get the master database, and they cannot get to the data chunks. Through this, the user data security will be improved.

Bohli et al., [14] proposed four various kinds of architectures for multi-cloud computing paradigm for enhancing the security and privacy level to the users and providers. The first approach indicates the replication of application that assists in verifying the integrity of data once the execution in cloud is finished. The second approach aids in the protection of data and logic by isolating them in third approach data and application confidentiality by dividing the application logic into parts and then executing it over different clouds. Same kind of approach is provided in the former architecture in which data is divided into parts and then executed over diverse clouds, thus helping in protection from dangerous cloud service provider. Every architecture has their own advantages and disadvantages, but the integration of that architecture will provide a much better secured approach for the case of multi-cloud systems.

Hao et al., [15] developed a novel remote integrity checking protocol that is on the basis of homomorphic verifiable tags. The new protocol is endowed with the functions SetUp, TagGen, Challenge, GenProof and Check Proof, in addition to the functions for data dynamics. SetUp produces a pair of keys, one openly known to everybody, and the other maintained as a secret by the client. Making use of
TagGen, the client calculates the block tag for every file block. Consider Dm to be the set of block tags. Once the computing of all the block tags are finished, the client transmits the file m to the remote server, and publishes Dm to be openly known to everybody. For the purpose of verifying the integrity of the file m, the verifier performs a challenge generation making use of Challenge function and then transmits it to the server. Once the server gets the challenge, it runs GenProof for the proof computation and then sends it to the verifier. After the verifier gets the response from the server, it does a check for its validity. In case it is valid, the function provides an output as “success,” else the function provides an output “failure”.

III. PROPOSED METHODOLOGY

The Distributed cloud is created by making use of the resource rendered by users and can offer resource for free to anybody who is a part of the system. The Distributed cloud uses virtualization for allocating the resources to different users and shares the available resources in an efficient manner and it also prevents single point of failure. In this work, a technique for protecting the secret key employing a multilevel threshold secret sharing scheme in the distributed cloud environment is proposed.

3.1. Secret Key Generation using Dragonfly Optimization

Dragonfly Optimization is introduced for generating the secret key values for every user in the distributed cloud. The user divides the secret key skin to n number of shares Si, where i∈{1, n} and then distributes them among all the resource providers. In this research work, the multilevel cloud users are taken into consideration and the value of threshold is got for offering security and also integrity which cloud users belongs to the corresponding threshold value.

Dragonfly Optimization (DO) is an optimization scheme that is inspired from nature, imitating the static and dynamic swarming activities of the dragonflies. The static swarm having lesser number of dragonflies hunts for preys in a small location, whereas the dynamic swarm having more number of dragon flies moves over huge distances; and they indicate the exploration and exploitation phases of the DO. Initially, DFO generates a swarm of dragonflies that are located in random in the search space. Then the position of every dragonfly in the solution space stands for a possible solution of the problem of optimization.

The swarm comprises of N dragonflies in the form of search agents. The position and step vector of a search agent is X and AX correspondingly with 1 ≤ i ≤ N. In this technical work, time is regarded to be an objective function. The search agent’s movement is modeled by the five separate behaviors in swarm, with their explanation as below:

Separation, is actually the tendency of an individual to maintain a distance away from search agents in the neighborhood, computed as below:

\[ S_i = -\sum_{j=1}^{f} X_i - X_j \]  
where i ∈ {1, 2, ..., N} refers to the index of current individual and j ∈ {1, 2, ..., J} indicates the index of the search agent in the neighborhood.

Alignment, is the tendency of an individual to have its velocity matched with velocity of adjacent search agents, calculated as below:

\[ A_i = \frac{1}{J} \sum_{j=1}^{f} \Delta X_j \]  
where \( \Delta X_j \) is the change in position of search agent j.

Cohesion, is the tendency of an individual to move towards the center of mass of the adjacent search agents, calculated as below:

\[ C_i = \left( \frac{1}{\sum_{j=1}^{f} X_j} \right) - X_j \]

The attraction toward a food source, is the tendency of an individual to move toward a food source, calculated as below:

\[ F_i = F_{loc} X_i \]  
Where \( F_{loc} \) refers to the location of food source.

Distraction outwards an enemy, is the tendency of an individual to move away from an enemy, calculated as below:

\[ E_i = E_{loc} + X_i \]  
Where \( E_{loc} \) refers to the location of enemy.

The food source fitness value \( F_{fit} \) and location \( F_{loc} \) are updated by making use of the search agent having the best fitness value. The enemy fitness value \( E_{fit} \) and location \( E_{loc} \) gets updated employing the search agent having the worst fitness value. During iteration k, the step and location of every search agent are updated as below:

\[ X_i(k+1) = X_i(k) + \Delta X_i(k) \]  
\[ \Delta X_i(k) = sS(k) + aA_i(k) + cC_i(k) + fF_i(k) + eE_i(k) + \omega \Delta X_i(k) \]  
where \( w \) stands for the inertia weight of search agent, \( S_i \), \( A_i \), \( C_i \), \( F_i \), and \( E_i \) refers to the respective behaviors fitness values of search agent i, and w, a, c, f, and e indicates the weights of the behaviors. The location of the search agent is updated as indicated in (7) in case the number of neighboring search agent J is at least one. In case it does not possess an adjacent search agent, then it will update its location by applying Levy flight, with the equation as below:

\[ X_i(k+1) = X_i(k) + Levy(k) \]  
\[ Levy(k) = 0.01 \frac{ri \cdot e^{-\frac{r^2}{|r|^2}}} \]

Algorithm Steps
Input: Multilevel Cloud Users MCUs = (mcu1, ... , mcun), 
Random prime positive integer PN = (p1, ... , pn)
Output: Secret Key SK = (sk1, ... , skn)

Objective function f(x), x = (x1, ..., xn) T
1. Initialize swarm (multi cloud users) N
2. initialize food source fitness \( F_{fit} \) and location \( F_{loc} \)
3. initialize enemy fitness \( E_{fit} \) and location \( E_{loc} \)
4. initialize number of neighbor J
5. initialize maximum iteration \( k_{max} \)
6. initialize objective function func (X)
The CRT solution is further split into m number of shares $S_k$ where $k \in \{1, n\}$, and $i \in \{1, m\}$. The number of share pools and have one or two of them needed.

To reconstruct SK at least t number of shares is needed.

1. for each share i of S, where $i \in 1, n$ do
2. Replicate the share $S_i$ into $k \geq 1$ number of replicas.
3. Distribute the replicas among $n$ number of resource providers.
4. If any data loss occurs then call data replication MSSIP.
end for

11. for each resource provider, $R_{p_i}, i \in 1, n$ do
12. Choose a pair $(P_i, N_i)$ by the following steps
   - for $i = 1$ atleast $n$ do
   - Generate a random series of pairwise relatively prime positive integers $P_i = p_{i1}$, $p_{i2}, ..., p_{im}$.
   - Generate a random series of m arbitrary integers $N_i = n_{i1}, n_{i2}, ..., n_{im}$.
   - Place these two series $P_i$ along with $N_i$, indicated as $(P_i, N_i)$.
   end for
13. Cloud User CU saves $(i, P_i)$ and $R_{p_i}$ saves $N_i$
14. Get a distinct solution $m = \chi_i$ from $(P_i, N_i)$
15. Divide the share of secret $S_{k_i}$ into m number of shares, $S_{1i}, S_{2i}, S_{3i}, S_{4i}, ..., S_{mi}$
16. Generate j number of dummy shares $S^D_{ij}$, where $j \geq m$
17. Reconstruct the share of secret $S_i$ from m number of shares,
end for
18. for each resource provider, $R_{p_i}, i \in 1, t$ do
19. Cloud User CU saves $(i, P_i)$ and $R_{p_i}$ saves $N_i$
20. Get a distinct solution $m = \chi_i$ from $(P_i, N_i)$
21. Divide the share of secret $S_{k_i}$ into m number of shares, $S_{1i}, S_{2i}, S_{3i}, S_{4i}, ..., S_{mi}$
22. Generate j number of dummy shares $S^D_{ij}$, where $j \geq m$
23. Reconstruct the share of secret $S_i$ from m number of shares,
end for
24. for each resource provider, $R_{p_i}, i \in 1, t$ do
25. Cloud User CU saves $(i, P_i)$ and $R_{p_i}$ saves $N_i$
26. Get a distinct solution $m = \chi_i$ from $(P_i, N_i)$
27. Divide the share of secret $S_{k_i}$ into m number of shares, $S_{1i}, S_{2i}, S_{3i}, S_{4i}, ..., S_{mi}$
28. Generate j number of dummy shares $S^D_{ij}$, where $j \geq m$
29. Reconstruct the share of secret $S_i$ from m number of shares,
end for
30. for each resource provider, $R_{p_i}, i \in 1, t$ do
31. Cloud User CU saves $(i, P_i)$ and $R_{p_i}$ saves $N_i$
32. Get a distinct solution $m = \chi_i$ from $(P_i, N_i)$
33. Divide the share of secret $S_{k_i}$ into m number of shares, $S_{1i}, S_{2i}, S_{3i}, S_{4i}, ..., S_{mi}$
34. Generate j number of dummy shares $S^D_{ij}$, where $j \geq m$
35. Reconstruct the share of secret $S_i$ from m number of shares,
end for
36. for each resource provider, $R_{p_i}, i \in 1, t$ do
37. Cloud User CU saves $(i, P_i)$ and $R_{p_i}$ saves $N_i$
38. Get a distinct solution $m = \chi_i$ from $(P_i, N_i)$
39. Divide the share of secret $S_{k_i}$ into m number of shares, $S_{1i}, S_{2i}, S_{3i}, S_{4i}, ..., S_{mi}$
40. Generate j number of dummy shares $S^D_{ij}$, where $j \geq m$
41. Reconstruct the share of secret $S_i$ from m number of shares,
20. Gather the share $Sh_i$ from each resource provider
21. end for
22. Reconstruct the secret key $SK$ from $Sh_i$ where $i = 1, \ldots, l$
23. end for

3.2. Adaptive Neuro Fuzzy Inference System (ANFIS) based Classification

Information security is the chief concern with IDS, where the aim is the protection of the confidentiality, integrity and availability of data present in the system. With the goal of providing a larger security for the confidential data along with information integrity, the new system proposed an ANFIS classifier. The data integrity is compared based on the classification accuracy and the minimum error got by making use of the ANFIS. The ANFIS needed data to be provided in a specific format aided by the learning process of the network.

ANFIS configuration is actually an oversimplified neural network through a Sugeno fuzzy model, as shown in Figure 1. Nodes at the equivalent layer possess comparable functions. The output of the $i$th node in layer 1 is represented as $O_{1,i}$.

Layer 1: Every node $i$ in this layer is actually an adaptive node having a node function,

$$O_{1,i} = \mu A_i(x) \text{ for } i = 1, 2$$

Or

$$O_{1,i} = \mu B_{i-2}(x) \text{ for } i = 3, 4$$

(10)

Where $x$ (ory) indicates the input to the $i$th node and $A_i$ (or $B_{i-2}$) represents a linguistic label (such as “low” or “high”) corresponding to this node. It means, $O_{1,i}$, indicates the membership grade of a fuzzy set $A = (A_1, A_2, B_1, or B_2)$ and it denotes the degree to which the input $x$ (ory) provided meets the quantifier $A$. The membership functions for $A$ and $B$ are generally defined by generalized bell functions.

Here $\{a_i, b_i, c_i\}$ indicates the parameter set. As the values of these parameters change, the bell-shaped function varies consequently, showing different forms of membership functions on the linguistic label $A_i$. The Figure 1 demonstrates the first order sugeno fuzzy model with two rules. In fact, few continuous and piecewise differentiable functions, like the globally used triangular shaped membership functions, are also competitive candidates for the node functions in this layer. Parameters in this layer are represented as premise parameters. The outputs from this layer comprise the membership values of the premise portion.

Layer 2: This layer consists of the nodes marked $\prod$ that multiply the incoming signals and provides the product out. For instance,

$$O_{2,i} = w_i = \mu A_i(x)\mu B_i(y)i = 1, 2$$

Each node output specifies the firing potency of a rule.

Layer 3: In this layer, the nodes marked $\Sigma$ calculate the ratio of the $i$th rule’s firing power to the sum of the complete rules’ firing strengths,

$$O_{3,i} = \bar{w}_i = \frac{w_i}{w_1 + w_2}i = 1, 2$$

(13)

The outputs of this layer are known as the normalized firing strengths.

Layer 4: The nodes of the layer are adaptive with the following node functions,

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i(p_i x + q_i y + r_i)$$

(14)

Where $\bar{w}_i$ indicates the output of layer 3, and $\{p_i, q_i, r_i\}$ specifies the parameter set. Parameters of this layer are known as resultant parameters.

![Figure 1: Equivalent ANFIS Architecture](image-url)

Layer 5: This single prefixed node of the layer, marked as $\sum$, comes out to be the final output as the sum of the complete incoming signals shown as below,

$$O_{5,i} = \sum_{i=1} w_i f_i = \frac{\sum w_i f_i}{\sum w_i}$$

Consequently, an adaptive network which functionally corresponds to a Sugeno first-order fuzzy inference system gets generated. On the basis of the classification accuracy the verification of the data integrity is done.
IV. PERFORMANCE EVALUATION

The experimentation carried out simulations utilizing a distributed cloud model, which is on the basis of the P2P overlay Kademia. It is realized by making use of the new algorithm for secret sharing on the distributed cloud. It supposed that the resource providers possess a minimum of 2GB RAM up to 16 GB, 2 to 8 cores. At first a node detects the available resource providers close to him and then splits the key into different shares and distributes every share to a resource provider who is available. The resource provider again generates more shares by employing his share to be the secret and saves it. User will then maintain the list of providers who are in store of the secret shares. Whenever a user needs the key, he gets in touch with other resource providers having the shares. When resource providers get the request, they merge the shares in their possession and transmit the original share to the user. At last, integrity, security and replica are checked. The new Multilevel Dragonfly Optimization Threshold schemes (MDOT) is then compared with the available system like Secret Sharing (SS), Multilevel Threshold Secret Sharing (MTSS), Multilevel Differential Evolution Threshold (MDET) –SS schema, Multilevel Bat Threshold (MBT) –SS and Multilevel Firefly Algorithm Threshold (MFAT) –SS.

4.1. Time Complexity

The system can be called better if it has lesser time complexity.

Figure 2 illustrates the average total time of SS, MTSS, MDET –SS, MBT, MFAT-SS and MDOT is introduced for security in cloud computing. The total time is inclusive of the time taken to divide the secret into shares at the first level, get the resources, distribute the shares to the resource providers and divide the shares at second level. It can be observed that with the increase in the number of nodes, the time to get the nodes and distribute the shares to neighborhood nodes is reduced. As the distributed cloud is created by several users for secret sharing, it is quite feasible and effective.

4.2. Accuracy (ACC)

Accuracy is defined as the ratio corresponding to the correctly performed classification or the tracking of normal and abnormal scenes in traffic monitoring process.

\[ ACC = \frac{(TP + TN)}{(TP + FN + FP + TN)} \times 100 \]  

(16)

Where,

- TP - True Positive
- TN - True Negative
- FP - False Positive
- FN - False Negative

![Figure 3: Accuracy Comparison](image)

Figure 3 illustrates the accuracy of SS, MTSS, MDET – SS, MBT-SS, MFAT-SS and MDOT-SS mechanisms in cloud computing. The number of nodes is plotted along the X-axis and the accuracy is plotted along the y-axis. The result obtained experimentally indicates that the new system attains a high accuracy in comparison with the available system.

4.3. Precision

Precision indicates the rightness of the classification or the tracking conducted in the traffic monitoring video files.

\[ Precision = \frac{TN}{(TP + FP)} \times 100 \]  

(17)

![Figure 4: Precision Comparison](image)

Figure 4 demonstrates the precision of SS, MTSS, MDET–SS, MBT-SS, MFAT-SS and MDOT-SS mechanisms in cloud computing. The number of nodes is plotted along the X-axis and accuracy is plotted along the y-axis. There is an increase in the precision result with the increase in the number of nodes. The result obtained experimentally reveals that the new system attains a great precision in comparison with the available system.

4.4. Recall

Recall is defined as the completeness of the classification or the tracking performed in the traffic monitoring video files.
Figure 5 demonstrates the recall of SS, MTSS, MDET–SS, MBT-SS, MFAT-SS and MDOT-SS mechanisms in cloud computing. The number of nodes is plotted along the X-axis and accuracy is plotted along the y-axis. There is an increase in the recall result when there is an increase in the number of nodes. The result obtained experimentally indicates that the new system attains a higher recall when compared with the already existing system.

V. Conclusion

Cloud Computing is a significantly novel concept, which offers a good deal of advantages for its users; though, it also can lead to few security issues that may reduce its speed during use. The present work guarantees the deployment of Multilevel secret sharing mechanisms for distributed cloud computing. Multilevel Dragonfly Optimization Threshold schemes (MDOT) based secret sharing mechanism is introduced for security in cloud computing. The chief concept behind these secret sharing mechanisms is that, the data is divided into multiple shares and then these shares are distributed among different users for ensuring the cloud users’ security. An ANFIS based classifier is introduced so as to provide security to the most vital data of the cloud user. Cloud user data integrity checking, makes use of a classification function that considers a cloud user data and then minimizes it to data of predetermined size. The result is known as encrypted data. It sends a specific random key for cloud user in order to accomplish integrity on data from any change or deletion. Moreover the proposed system examined the problem of QoS-aware data replication in distributed cloud computing. For the purpose of solving the problem of data replication, the device heterogeneity is also taken into consideration along with the QoS requirements of applications. Multi-Stage Stochastic Integer Programming (MSSIP) follows the intuitive concept of Quality of Service (QoS) to carry out data replication. Result obtained from experimentation reveals that the newly introduced MDOT based secret sharing schemas identifies related vulnerabilities and malicious attacks with potential solutions in comparison with the available state of art techniques.

REFERENCES