

A hybrid authentication model for secure mobile agent migration using Euler’s theorem and machine learning

Pradeep Kumar *

Department of Computer Science and Engineering

JSS University

Noida 201301

Uttar Pradesh

India

Anuradha Singh [†]

Department of Information Technology

JSS University

Noida 201301

Uttar Pradesh

India

Nishant Gupta [§]

Department of Computer Science & Engineering

ITS Engineering College

Greater Noida 201308

Uttar Pradesh

India

Dharm Raj [‡]

Department of Computer Science & Engineering

Sharda University

Greater Noida 201310

Uttar Pradesh

India

* E-mail: pradeep8984@gmail.com (Corresponding Author)

[†] E-mail: anuradha.singh@jssaten.ac.in

[§] E-mail: nishantlira@gmail.com

[‡] E-mail: dharmraj4u@gmail.com

Anand Pandey[@]

*Department of Computer Science & Applications
Sharda University
Greater Noida 201310
Uttar Pradesh
India*

Abhishek Jain[#]

*Department of Information Technology
Guru Ghasidas Viswavidyalaya (A Central University)
Bilaspur 495009
Chhattisgarh
India*

Abstract

The autonomous migration capability of mobile agents makes them highly susceptible to various security threats during their movement across distributed platforms. Consequently, ensuring the security of both mobile agents and their execution environments has become a critical concern in distributed network systems. However, most existing protection mechanisms in mobile agent-based paradigms primarily focus on safeguarding the host platforms, while comparatively less attention has been devoted to protecting the agents themselves. To address this limitation, this paper proposes a hybrid authentication model for secure mobile agent migration that integrates a mathematical key management technique based on Euler's Theorem with a machine learning mechanism for enhanced agent security. The proposed approach enables secure hopping of mobile agents between different platforms through a double encryption mechanism applied to the secret key shared among n mobile agents. Furthermore, the proposed model incorporates machine learning to strengthen authentication and detect potential malicious behavior during mobile agent migration. The proposed technique is evaluated and compared with existing mobile agent security approaches. Experimental results demonstrate that the proposed model outperforms conventional methods in terms of improved agent security and reduced execution time. To validate the effectiveness of the proposed approach, a comparative analysis is conducted with existing key management schemes based on the Chinese Remainder Theorem (CRT) and Shamir's Secret Sharing. The Outcomes show that the time complexity of the proposed method is 0.0071 seconds, which is significantly lower than that of the existing techniques for the same number of mobile agents thereby demonstrating the efficiency and practicality of the proposed hybrid authentication model.

Subject Classification: *Primary 68P20, Secondary 68T99.*

Keywords: *Mobile agent, Secret share, Euler theorem, Tree parity machine, Double encryption.*

[@] E-mail: anandpandey29@gmail.com

[#] E-mail: ajain.nit@gmail.com

1. Introduction

Mobile agents [4]-based architecture is an advancement to distributed computing. It is hyper threaded software which is deployed across malicious communication network in order to accomplish designated tasks. Basically, mobile agents can be considered as intelligent pieces of software that assess the task assigned to it and accordingly migrate to the location where resources for execution of task are available. Thus, a significant load of computational independence is allowed with approach of mobile agents. Figure 1 demonstrates the computation process of mobile agents [2].

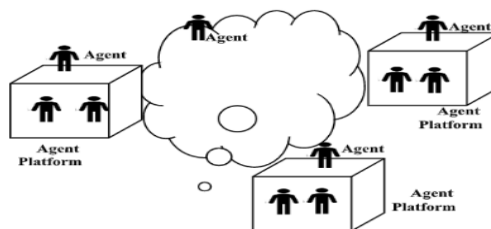


Figure 1
Execution of operation using mobile agent paradigm

As can be depicted from Figure 1, instead of sharing entire information from one platform to another, a particular procedure containing commands for execution is migrated. This procedure consists of the commands required for execution of tasks assigned by the source platform. Every single mobile agent follows a predefined lifespan for accomplishment of allocated operations [3].

The lifecycle of mobile agent [8] follows various stages that are Creation, Cloning, Dispatch, Deactivation, Activation, Interaction, and Communication with platform and Disposal. Now, the primary focus of former agent systems has been on determining the working of agents only and not the protection. However, since mobile agents need to operate in malicious environments, the security of these agents becomes a primary issue. Confidentiality, data integrity and availability of resources are the three major concerns of security that are essential to ensure while functioning within the network.

In addition to the parameters mentioned, authentication of platform as well as agent identities is required for assuring the security within mobile agent-based network [6]. In this paper, a tree parity-based control strategy for secure operation of mobile agents is proposed. The suggested model includes following vital parts:

- Information shares generation using Euler's theorem.
- Synchronized key generation at sender and receiver end applying neural cryptography.
- Generated share encryption by sender making use of synchronized key.

- Employing synchronized key for deciphering ciphered shares that were received.

Basically, a two-level security measure is applied to carry out the present research work ensuring authenticity and data integrity.

1.1 Euler Theorem

Euler theorem [11] emerges as Fermat's little theorem generalization that handles integer powers modulo to positive integers. With Euler theorem, application of fundamental number theory can be increased. Hereby, Euler theorem establishes that for all relatively prime a and n Equation 1 holds true.

$$a^{\varphi(n)} \equiv 1(\text{mod } n) \quad (1)$$

where, $\varphi(n)$ represents totient function of Euler. This function quantifies positive integers less than the value of n and at the same time coprime to its value. Now, $\varphi(n)$ returns $n-1$ in case the value of n is a prime number. However, if it is a composite number then $n(1 - \frac{1}{p_1})(1 - \frac{1}{p_2})(1 - \frac{1}{p_3}) \dots (1 - \frac{1}{p_n})$ is returned where $p_1 < p_2 < p_3 \dots p_n$ are prime numbers.

1.2 Tree Parity Machine (TPM)

Tree parity machine or TPM is a peculiar kind of neural network which is multilayered and feed forward. As shown in Figure 2, it comprises of $K \times N$ neurons for input, K number of hidden neurons and single output neurons.

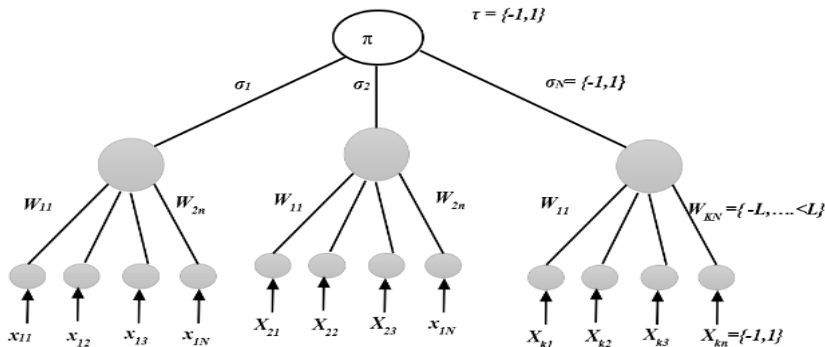


Figure 2
Structure of Tree Parity Machine

Now, inputs to this network can take 3 values indicated in Equation 2.

$$X_{ij} \in \{-1, 0, +1\} \quad (2)$$

However, weights among the input neurons and hidden layer of neurons take values represented in Equation 3.

$$W_{ij} \in -L, \dots, 0, \dots, +L \quad (3)$$

Further, output of every hidden neuron is computed by adding all multiplications attributed to input neuron values denoted in Equation 2 and weights mentioned in Equation 3. Equation 4 [10] signifies the value of output for every hidden neuron.

$$\sigma_i = sg_n(\sum_{j=1}^N W_{ij} * x_{ij}) \quad (4)$$

sg_n in Equation 4 denotes signum which is a very simple function that returns one of three values indicated in Equation 5.

$$Sg_n = \begin{cases} -1 & \text{if } x < 0 \\ 0 & \text{if } x = 0 \\ 1 & \text{if } x > 0 \end{cases} \quad (5)$$

For ensuring binary output, the case where scalar product comes out to be 0, outcome of hidden neuron gets mapped to value-1. In such scenario, output is computed by multiplying all the values generated through hidden elements. Equation 6 signifies that output.

$$\prod_{i=1}^k \sigma_i \quad (6)$$

Ultimately, binary output is generated through TPM. This binary output is then exchanged between 2 parties, so that the weights of Tree Parity Machine can be synchronized by using one of the following rules.

Hebbian rule: $W_i^* = W_i + \sigma_i x_i \theta(\sigma_i \tau \theta \tau^A \tau^B)$ (7)

Anti-Hebbian rule: $W_i^* = W_i - \sigma_i x_i \theta(\sigma_i \tau \theta \tau^A \tau^B)$ (8)

Random walk: $W_i^* = W_i + x_i \theta(\sigma_i \tau \theta \tau^A \tau^B)$ (9)

Once synchronization of weights is done completely, these weights can be used by the two parties as key. Figure 3 represents the entire workflow that Tree Parity Machine follows.

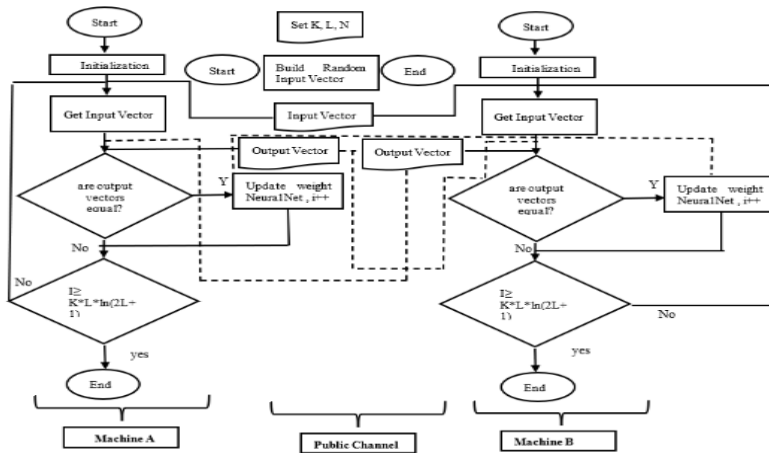


Figure 3
Flowchart of Tree Parity Machine

2. Background Study

Studies centered around the functionality of mobile agents under malicious environment are discussed in this section.

Gupta, Gupta, and Deshmukh [7] applied neural encryption and Shamir's method in their work. There were two steps to the encryption process. Shamir's technique was used to create shares in the first stage, while neural cryptography was used to encode shares in the stage 2. The suggested study considered safety of confidential information via open line. Chourasia et al. [4] offered a vectorized adaptation of the Tree Parity Machine's Neuro Cryptography technique. Additionally, it was demonstrated that employing vectorized tree parity machines significantly reduced execution time even for huge simulations. Joseph and Mariappan [10] suggested a Trust Scoring System to assess a platform's dependability using a parameter called a Trust Score. The five components of Persistence, Competence, Reputation, Credibility, and Integrity were considered to be the makeup function on which the Trust Score was built.

Hsiao et al. [9] had put forward an effort to reduce the amount of time the host spent corresponding with the system's numerous participants. The mobile agent functioned as a host-to-host data converter in the proposed research. The article suggested utilization of a function defined by the Lagrange polynomial to compute the decryption key. Each host was provided with a decryption key to access the private document by entering a secret key into an interpolation function created via Lagrange interpolation. Fong, Parr, and Morrow [5] suggested the VDPS and VBAS, the two security protocols. By encrypting and encasing management data in SDBs, the VDPS was used to safeguard the data. The scheme defined six kinds of Visible Domains (VDs), and it suggested implementation for each class, together with accompanying cryptographic methods and key specifications. Key initialization, distribution, and management procedures with reference to a key environment for the VDPS were also covered.

Ametller, Robles, and Ortega-Ruiz [1] identified some of their drawbacks and suggested a new architecture for secure mobile agents that took care of the two unresolved problems of agent code protection and inter-platform portability. Existing solutions sometimes used cumbersome frameworks, which drastically restricted portability and required extensive re-engineering of legacy systems.

2.1 Research Gap

On the basis of literature, it was identified that security of mobile agents still needs attention of research. The major issue identified is that prevailing algorithms focus only on the platform security and do not take channel insecurity into consideration [1]. Thus, securing the data carried by mobile agent during transmission from one platform to another is still a potential area.

2.2 Aim of Research

The aim of this research is to allow only authentic flow of data among mobile agents in malicious environments. The primary goal of this research is to

determine criteria that allow transmission of only authentic data. The suggested authentication model generates a synchronized key which assures authenticity of data received by a mobile agent. The two-level encryption mechanism proposed in current research work ascertains inaccessibility of data to unauthorized agents, hence maintaining the integrity.

3. Proposed Approach

Figure 4 represents the proposed model. The shares indicated as share₁, share₂, ..., share_n in figure are produced using Euler's theorem discussed in Section 1.2. However, neural cryptography mentioned in Section 1.3 is used to generate keys denoted as key₁, key₂, ..., key_n. Problem that arises now is the secure communication of these shares to their respective shareholders. Thus, to ensure this secure transmission one more security level is added for this research. The shares that are generated at final stage are then encrypted making use of neural keys addressed in Section 1.1.

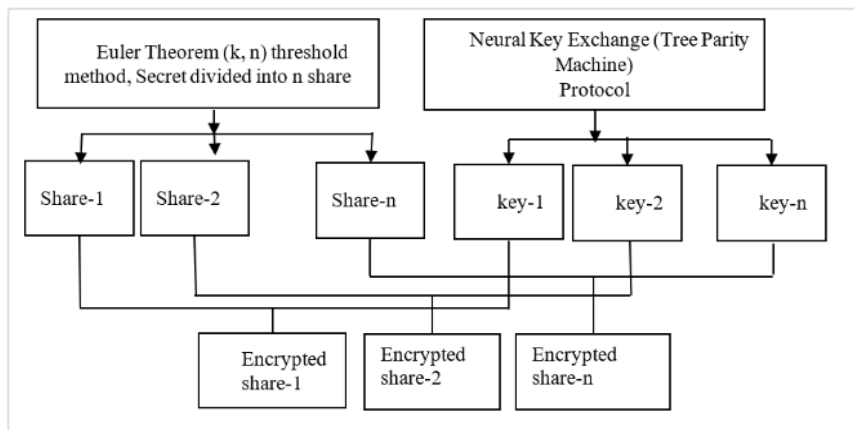


Figure 4
Proposed Model

3.1 Modified Euler's Theorem

Algorithm 1 represents the implementation of Euler's Theorem for current research. It is implemented in three phases of Initialization, creation of share followed by share reconstruction.

Algorithm 1 Euler Initialization (t, n) Threshold Scheme

- 1: Select secret key denoted by S , $(0 < S < p_0)$ in order to authenticate the mobile agents at time of code execution as well as transformation.
- 2: Select positive prime integer $p_0 < p_1 < \dots < p_n$
- 3: $\gcd(p_i, p_j) = 1, \forall i \neq j \leq n$
- 4: $\prod_{i=1}^t p_i > (p_0 + 1) \prod_{i=1}^{t-1} p_{n-t+i+1}$

Algorithm 2 Share creation

- 1: $S_i = (S + \alpha p_0) \bmod p_i \forall \text{ mobile agent } 0 < i < n$
- 2: $\alpha = (\frac{\prod_{i=1}^t p_{n-i+1}}{p_0}, \prod_{i=1}^t \frac{p_i}{p_0} - 1)$

3.2 Modified Tree Parity Machine

Algorithm 3 demonstrates the implementation of Tree Parity Machine i.e., TPM. Tree Parity Machine produces a binary output for any arbitrary vector X given as input. Following parameters can be used for describing a TPM.

Algorithm 3 Tree Parity Machine

- 1: Given: $X[m]$
- 2: Given: (h, m, W)
- 3: $M \leftarrow \text{randit}(-W, W + 1, [K, n])$
- 4: $\sigma_p = \text{sgn}(\sum_{q=1}^n W_{pq} X_{pq})$
- 5: Output: $\tau \leftarrow \prod_{p=1}^h \sigma_p$

For every machine h , n and W are initialized to same values. X denotes n size input vector which is random. M corresponds to $h \times n$ matrix that is initialized with arbitrary integers. σ represents inner activation with $[h]$ dimension. Output of machine is indicated by τ which is computed by multiplying elements belonging to σ . $(-1, 0, +1)$ form the possible output set. However, if output comes out to be 0, it is converted to -1. Thereby, binary outputs $(-1, 1)$ are generated. If two machines produce same output, then these machines synchronize their corresponding weights.

3.3 Weight Synchronization

Weight synchronization plays a very key role in this research because this mechanism prevents the sharing of key over insecure channels. Synchronization of weights at two ends is the only way of attaining the original message, hence ensuring the integrity of data. For current research work, all three synchronization rules are applied distinctly and compared on basis of time and number of updates required for generation of synchronized key.

3.3.1 Hebbian Learning Rule

Hebbian learning rule works on the principle that there exists a positive correlation between the two outputs if they are correlated. Algorithm 4 mentions the steps of Hebbian learning rule applied for weight synchronization. Further, Equation 10 indicates θ function while Equation 11 specifies fclip function.

Algorithm 4 Hebbian Learning Rule

- 1: Input: $\{M, X, \sigma, \tau_1, \tau_2, l\}$
- 2: for each (p, q) in M do

```

3:   $M_{pq} \leftarrow M_{pq} + X_{pq} \times \tau_1(\sigma_p, \tau_1)(\tau_1, \tau_2)$ 
4:   $M_{pq} \leftarrow f_{clip}(M_{pq})$ 
5:  end for

```

$$\theta = \begin{cases} 1 & \text{if } \tau_1 = \tau_2 \\ 0 & \text{otherwise} \end{cases} \quad (10)$$

$$f_{clip} = \begin{cases} W & \text{if } M_{pq} > W \\ -W & \text{if } M_{pq} < -W \end{cases} \quad (11)$$

Anti - Hebbian Learning Rule

Algorithm 5 represents the step implementation of Anti-Hebbian Learning Rule.

Algorithm 5 Anti Hebbian Learning Rule

```

1:  Input: {M,X,σ, τ1, τ2, l}
2:  k, n = M.shape
3:  for each (p,q) in np.ndenumerate(M):
4:     $M_{pq} \leftarrow M_{pq} - X_{pq} \times \tau_1 \times \theta(\sigma_p, \tau_1) \times \theta(\tau_1, \tau_2)$ 
5:     $M_{pq} \leftarrow \text{np.clip}(M_{pq}, -1, 1)$ 
6:  end for

```

3.3.2 Random Walk Learning Rule

Algorithm 6 denotes working of Random Walk Learning Rule.

Algorithm 6 Random Walk Learning Rule

```

1:  Input: {M, X,σ, τ1, τ2, l}
2:  k, n = M.shape
3:  for each (p,q) in np.ndenumerate(M):
4:     $M_{pq} \leftarrow M_{pq} + X_{pq} \times \theta(\sigma_p, \tau_1) \times \theta(\tau_1, \tau_2)$ 
5:     $M_{pq} \leftarrow \text{np.clip}(M_{pq}, -1, 1)$ 
6:  end for

```

Proposed framework for the security of mobile agents works in six phases.

- System Initialization: In the system initialization for the mobile agents select Euler parameters ($p_0, p_1, \dots, p_n$) and Tree parity machine parameter (K, N, L)
- Euler Based Secret Share Generation: Generation of modular shares $S_i = (S + \alpha p_i) \bmod p_i$
- Tree parity machine synchronization Phase: Mutual learning between source and destination using TPM and there is No secret key exchange over the Communication channel.
- Dual Encryption layer: The synchronization of key with two algorithms Euler and Tree parity machine provides dual security.

- e. Secure Agent Migration: Agents migrated securely in malicious environments.
- f. Verification and Re-Keying: TPM weight verification and Dynamic key regeneration for subsequent mobile agents [12].

The proposed framework ensures secure mobile agent migration by Synchronization of Euler's theorem and Tree Parity Machine based secret key sharing. At the Home platform the mobile agent's code and state are protected using Euler based secret generation and TPM mutual learning to establish a synchronized session key without explicit key exchange. The Mobile agent is a secure migration layer preventing unauthorized access during migration. At the destination platform Tree parity machine and Euler theorem generate secret keys to decrypt and restore the agent state. Dynamic rekeying is deployed after each migration to support secure mobile agents' migration, thereby ensuring confidentiality, integrity, and authentication throughout the mobile agent lifecycle.

4. Results and Interpretations

All experiments of proposed framework were executed on a system equipped with an Intel® Core™ i7-12700H processor (2.7 GHz), 16 GB RAM, and 1 TB SSD, running Windows 11 (64-bit). The proposed framework based on Euler and TPM technique was implemented in Python 3.10 using NumPy and SciPy libraries under a single threaded execution environment. A simulated mobile agent framework was used with 50 to 1000 mobile agents migrating across 10 distributed platforms over a malicious communication channel under the Dolev Yao adversarial model. Tree Parity Machine parameters were fixed at $K=3$, $N=64$, and $L=4$, while Euler-based secret sharing employed 64-bit primes with a threshold of $t=3$ out of $n=5$ Secret shares. Time complexity values (e.g., 0.0071 s) represent the average per agent execution time measured using Python's high-resolution timer over 100 independent executions. Performance was evaluated in terms of Secret key generation time, authentication delay, computational overhead, communication cost, and scalability with all baseline techniques executed under same environment to ensure fair and reproducible comparison. Empirical results and interpretation of proposed secret key sharing mechanisms are based on fusion of Euler theorem and neural cryptography is performed. Secret key for the execution and authentication of mobile agents equally depends on both the approaches. Further, the propound approach provides dual level authentication of mobile agents. In Neural key cryptography, secret key generation is based on the synchronization of host and execution of tree parity machines. After 100% synchronization of both machines, a same key is generated that is used as sub key for authentication. Complete key for execution is generated after the combination of this key with the key generated by Euler algorithm. Thereby, combined key is used for authentication of mobile agents and execution of task assigned by host. For empirical analysis, 100 hidden layers and 10 input neurons are considered denoted by K and n ,

respectively. Further, L represents the weight range taking values 10, 15, 20 and 25. Table 1 and Table 2 shows Number of updates required for the Synchronization and Time Required in Second(s) for the Synchronization respectively.

Table 1
Number of updates required for the Synchronization of Neural Tree Parity Machine

	Weight Range L			
	10	15	20	25
Hebbian	859	835	674	543
Anti- Hebbian	1476	1430	1526	1430
Random Walk	579	660	560	701

Table 2
Time Required in Second(s) for the Synchronization of Neural Tree Parity Machine

	Weight Range L			
	10	15	20	25
Hebbian	23.41	17.35	16.38	13.66
Anti- Hebbian	36.25	32.69	36.22	35.18
Random Walk	13.61	14.15	13.81	16.09

Turnaround time for Euler, CRT and Shamir secret sharing in average and best case is shown in Table 3 and Table 4 respectively. Key generation by Euler theorem is subpart of the proposed mechanism. Besides, it can be analysed from results demonstrated in Table 3 and Table 4 that Euler algorithm takes less time for subpart key generation as compared to CRT and Lagrange interpolation for both average as well as best case.

Table 3
Average Case Analysis of Proposed Mechanism for Turnaround Time in seconds

	Number of Mobile Agents					
	5	10	15	20	25	30
Euler	0.0036	0.0052	0.0061	0.0068	0.007	0.0071
CRT	0.0043	0.0061	0.0068	0.0088	0.0091	0.0096
Shamir	0.0054	0.0058	0.0067	0.0069	0.0077	0.0085

Table 4
Best Case Analysis of Proposed Mechanism for Turnaround Time in seconds

	Number of Mobile Agents					
	5	10	15	20	25	30
Euler	0.003	0.005	0.006	0.007	0.006	0.007
CRT	0.003	0.006	0.006	0.007	0.007	0.007
Shamir	0.004	0.006	0.006	0.008	0.007	0.009

The combined utilize Tree Parity Machines and Euler’s theorem offers advantages over Traditional secret key management methods in terms of speed, synchronization robustness, and key randomness. TPM-based mutual learning enables fast key establishment using simple integer operations and avoids costly modular exponentiation, while Euler-based modular secret generation further reduces computational cost. Synchronization robustness is achieved because only legitimate platforms with identical TPM parameters can successfully synchronize, and active adversaries fail to converge under interference. Both theoretical complexity analysis and empirical results demonstrate lower latency, reduced communication overhead, and higher synchronization success rates compared to Diffie–Hellman, CRT, and Shamir-based schemes shown in Table 5.

Table 5
Performance matrices proposed framework compared to Diffie–Hellman, CRT, and Shamir-based schemes

Metric	Euler -TPM (Proposed)	Diffie Hellman (DH)	CRT-Based Scheme	Shamir Secret Sharing
Key Establishment Method	Neural synchronization + modular arithmetic (no key exchange)	Modular exponentiation	Modular congruence computation	Polynomial interpolation
Key Generation Speed	Fast (linear time, lightweight ops)	Slow (expensive exponentiation)	Moderate	Moderate–Slow
Synchronization Robustness	High (fails under MITM interference)	Moderate	Low	Low
Communication Overhead	Very Low (parity bits only)	High	Medium	Medium
Key Randomness / Entropy	Very High (neural + Euler diffusion)	High	Moderate	Moderate

Computational Complexity	$O(T \cdot K \cdot N)$	$O(\log^3 n)$	$O(n)$	$O(n^2)$
Scalability (Number of Agents)	High	Limited	Moderate	Limited
Resistance to MITM Attacks	Strong (no key transmission)	Vulnerable without authentication	Weak	Weak
Replay Attack Resistance	Yes (dynamic re-keying)	Partial	No	No
Memory Requirement	Low	High	Medium	Medium
Suitability for Mobile Agents	Highly Suitable	Less Suitable	Moderately Suitable	Less Suitable

4.1 *Security Proof: Security of mobile agent using Tree parity machine and Euler Theorem shown in Table 6.*

Table 6
Security of mobile agent using Tree parity machine and Euler Theorem

S. no	Security Property	Validation Process
1	Confidentiality	Dual-layer encryption (Euler + TPM)
2	Authentication	Mutual TPM synchronization
3	Integrity	Encrypted share verification
4	Forward Secrecy	Session-wise TPM re-keying
5	MITM Resistance	Synchronization failure under interference
6	Replay Protection	Dynamic key regeneration

5. Conclusion and Future Scope

The paper presents an extension to Euler theorem and Tree Parity Machine for preventing attacks on mobile agents while moving to different platforms. The proposed approach certain security of key through robust mechanism for key management based on threshold values. The approach monitors secure agent access by limiting the sharing to only legitimate users. Further, results demonstrate that the given approach follows a linear time complexity. Conclusively, the focal point of current research work is to improve mobile agents' security aspects during transmission and in return, new research directions to achieve more security requirements are opened. Thus, future studies of prevailing area may introduce more generalized paradigm for catering the necessity of mobile agent's security along with identification of unreliable hosts.

References

- [1] J. Ametller, S. Robles, and J. A. Ortega-Ruiz, "Self-protected mobile agents," in *Proc. 3rd Int. Joint Conf. Autonomous Agents and Multiagent Systems (AAMAS)*, New York, USA, pp. 362–367 (2004).
- [2] P. Bagga and R. Hans, "Applications of mobile agents in healthcare domain: A literature survey," *Int. J. Grid Distributed. Computer.*, vol. 8, no. 5, pp. 55–72 (2015).
- [3] T. Chattopadhyay and G. Prasad, "Mobile agent security against malicious hosts: A survey," *SN Computer. Sci.*, vol. 3, no. 2, pp. 1–9 (2022).
- [4] S. Chourasia, H. C. Bharadwaj, Q. Das, K. Agarwal, and K. Lavanya, "Vectorized neural key exchange using tree parity machine," *Comp. Soft.*, vol. 8, no. 5, pp. 3140–3145 (2019).
- [5] C.-H. Fong, G. Parr, and P. Morrow, "Security schemes for a mobile agent-based network and system management framework," *J. Network. Syst. Manage.*, vol. 19, no. 2, pp. 230–256 (2011).
- [6] Y. Guo, C. Jiang, T.-Y. Wu, and A. Wang, "Mobile agent-based service migration in mobile edge computing," *Int. J. Commun. Syst.*, vol. 34, no. 3, Art. no. e4699 (2021).
- [7] M. Gupta, M. Gupta, and M. Deshmukh, "Single secret image sharing scheme using neural cryptography," *Multimedia Tools Appl.*, vol. 79, no. 17, pp. 12183–12204 (2020).
- [8] Q. Hong, Q. Tan, J. W. Siegel, and J. Xu, "On the activation function dependence of the spectral bias of neural networks," *arXiv preprint arXiv:2208.04924* (2022).
- [9] T.-C. Hsiao, Z.-Y. Wu, T.-L. Chen, Y.-F. Chung, and T.-S. Chen, "A hierarchical access control scheme based on Lagrange interpolation for mobile agents," *Int. J. Distributed. Sensor Network.*, vol. 14, no. 7, Art. no. 1550147718790892 (2018).
- [10] A. J. John Joseph and M. Mariappan, "A novel trust-scoring system using trustability coefficient of variation for identification of secure agent platforms," *PLoS One*, vol. 13, no. 8, Art. no. e0201600 (2018).

- [11] P. Kumar, N. Singhal, and D. Pandey, "Euler totient function and Fermat–Euler theorem based on an optimized key management scheme for securing mobile agents' migration," *Int. J. Inf. Technol.*, vol. 14, no. 2, pp. 657–665 (2022).
- [12] R. Aragona, R. Civino, N. Gavioli, and M. Pugliese, "An authenticated key scheme over elliptic curves for topological networks," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 25, no. 8, pp. 2429–2448 (2022), doi: 10.1080/09720529.2020.1866298.

Received December, 2025