Preserving Integrity of Evidence with Blockchain Technology in Cloud Forensics for Immigration Management

Sahadev Maruti Shinde and Venkateswara Rao Gurrala

GITAM School of Engineering, Visakhapatanam, Andhra Pradesh, India

https://doi.org/10.26636/jtit.2023.164522

Abstract — As the popularity of cloud computing increases, safety concerns are growing as well. Cloud forensics (CF) is a smart adaptation of the digital forensics model that is used for fighting the related offenses. This paper proposes a new forensic method relying on a blockchain network. Here, the log files are accumulated and preserved in the blockchain using different peers. In order to protect the system against illegitimate users, an improved blowfish method is applied. In this particular instance, the system is made up of five distinct components: hypervisor (VMM), IPFS file storage, log ledger, node controller, and smart contract. The suggested approach includes six phases: creation of the log file, key setup and exchange, evidence setup and control, integrity assurance, agreement validation and confidential file release, as well as blockchain-based communication. To ensure efficient exchange of data exchange between the cloud provider and the client, the methodology comprises IPFS. The SSA (FOI-SSA) model, integrated with forensic operations, is used to select the keys in the best possible way. Finally, an analysis is conducted to prove the effectiveness of the proposed FOI-SSA technique.

Keywords — cloud computing, cloud forensics, FOI-SSA model, improved blowfish

Tab. 1. Abbreviations and terms used in this paper.

Abbreviation	Description
1	2
AES	Advanced encryption standard
BlockSLaaS	Blockchain assisted secure logging as-a-service
BES	Bald eagle search
CF	Cloud forensics
CC	Cloud computing
CSP	Cloud service provider
CFI	Cloud forensic investigator
CADF	Cloud auditing data federation
DMTF	Distributed management task force
DBO	Dynamic butterfly optimization
EB	Ethereum blockchain
FCS	Fuzzy based smart contracts
FIO	Forensic investigation optimization
HSO	Harmony search optimization
ECC	Elliptical curve cryptography
IPFS	Interplanetary file system
LGoE	Logical graph of evidence
LA	Lion algorithm
RSA	Rivest-Shamir-Adleman
SRVA	Secure ring verification based authentication
SIEM	Security data information and event management
SA-DECC	Sensitivity aware deep ECC
SSA	Sparrow search algorithm
SSO	Salp swarm optimization
SRVA	Secure ring verification based authentication
TPS	Transaction per second

1. Introduction

When individuals leave their countries and move to other states, we are dealing with migration [1]–[5]. Such persons go through immigration-related processes in order to become permanent residents of the their new country. Usually, the procedure is very complicated. The applicant needs to get a visa [1] and then apply for a permanent residency permit which may later be converted into citizenship [6]–[11]. This process becomes easier if the applicant is backed by a company or if their family member is already a resident of the country concerned [12]–[14]. Due to the strict immigration laws in effect in some countries, people revert to illegal practices and attempt to infiltrate states without permission [15], [16]. This leads to illegal immigration [17]–[19] – an issue faced by almost every other country in the world [20]–[22].

The proposed work keeps track of immigrants by storing several relevant pieces of information in the form of immutable [6] and unique blockchain records [23], [24]. When a person is suspected of illegal immigration, their official documents are

compared with the record stored using blockchain [25]–[27], i.e. a distributed ledger [28], [29].

Blockchain is a network of fault-tolerant and distributed servers that contain shared, duplicated, and distinct content [30]. The management of a blockchain is cheap and quick, since it is immutable and cannot contain false or duplicate information [31]–[33]. Blockchain can process fingerprints, facial recognition, and retinal scanning biometric data [17], [18]. Blockchain-based reactive data can be secured using protective confidentiality encryption, limiting its use to authorized entities only [34].

The novelty of this work lies in the fact that a novel blockchain-based CF scheme is proposed, where an improved blowfish mechanism is deployed for encryption purposes, and in exploiting the FOI-SSA algorithm for creating an optimal key.

The paper is set up as follows. After the related works review given in Section 2, in Section 3 the project is presented and the model created is described. The FOI-SSA model recommended for generating the best key is described in Section 4. Sections 5 presents the conclusions.

2. Literature Review

This section surveys the eight existing blockchain-dependent evidence integrity preservation methods used in CF. Rane and *et al.* [1] proposed the forensic-aware BlockSLaaS model to steadily process and store logs by addressing multi-stakeholder collusion issues and facilitating confidentiality and integrity. CFI was capable of accessing logs for forensic purposes, using BlockSLaaS to protect the logs' privacy. However, the method failed to validate whether the service provider guaranteed precise logs. Jain *et al.* [2] presented a blockchain method for preserving the integrity of log files. IPFS and the blockchain technology were combined to transform a centralized storage system into a decentralized one. The integrity of log files was preserved by storing log files in blockchain. Thanks to such an approach, the system was used for storing huge log files at a minimal cost. However, the method failed to maximize CSP trust by minimizing CSP dependencies.

Pourvahab *et al.* [3] presented an SRVA scheme for protecting the system against unauthorized users. To ensure even better protection of the cloud platform, the secret keys were optimally produced by utilizing the HSO technique. In this case, the server stored the data after they had been encrypted with the use of the SA-DECC algorithm. By modifying FCS, such a strategy allowed the user to track down data and LGoE collected with the use of blockchain enabled the evidence to be studied. However, the proposed method failed to improve the digital forensics model. Dalezios *et al.* [4] proposed the DMTF with CADF standard for CF. The authors improved the Apache Cloud Stack platform by employing CADF activity tracking adopted in an Open Stack and made it more forensically reverberant. Stelly *et al.* [5] developed a method relying on automated container deployment and orchestration platforms to attain improved performance in digital forensics. The results showed that the distributed container-based approach offered a workable technical foundation for addressing the increased data volumes in digital forensic investigations.

Park *et al.* in [6] presented a permission blockchain-based data integrity management system for CFs. Such a method was capable of certifying the integrity of data while processing more transactions. However, there is an issue that the evaluation of performance cannot be made on anticipated data dimension. However, the model can be utilized as one of the methods for addressing security-related issues in cloud platforms. It failed, however, to accumulate network data by performing simulations concerned with computing precise TPS. Dasaklis *et al.* [7] described a CF method relying on the available blockchain-based technologies. The approach provided a detailed review of the various advantages and shortcomings of the mutually beneficial relationship between blockchain technology and the current digital forensics approach. Unfortunately, the method failed to identify different research issues in digital forensics. Irfan *et al.* [8] presented a model using SIEM to address the problem of effective evidence collection in CFs. The method shared evidence with cloud users, whenever needed. The proposed method helped perform detailed CF by adapting evidence, but failed to improve the performance of the solution by applying advanced optimization techniques.

3. Blockchain-based Protocol Developed for Maintaining Integrity in CF

Anti-tampering and privacy protection are two critical security requirements in cloud computing environments. Figure 1 shows the outline of the proposed architecture. In judicial forensics, maintaining privacy is a top priority. The suggested technique adopts an appropriate mechanism for maintaining confidentiality and anonymity, ensuring that no private data is released during the derivation function of a blockchain-based process. The system incorporates eight elements, including hypervisor, virtual machine, node controller, log ledger, IPFS file storage, blockchain network, CFI, and smart contract.

3.1. Initialization Step

The start-up phase involves launching virtual machines, hypervisors, node controllers, IPFS cloud storage, smart contracts, blockchain networks, CFI, and log ledgers. The following is a more detailed depiction of each entity. A virtual machine (VM) is a computational source that runs programs maps using software, rather than a real computer. The hypervisor is software that creates and operates a collection of virtual machines, allowing one host to handle several guest VMs, by sharing resources virtually. The nodes controller gathers logs from all virtual platform sources via log libraries and creates log entries for each log. IPFS cloud storage is a file transfer mechanism depending on cryptography hashes, that can be readily stored on the blockchain and regulated to effectively store and transfer large files, while smart contract acts as a set

of applications that are kept on the blockchain and continue to run when specific conditions are satisfied.

The blockchain network, in turn, offers ledger and smart contract functions to varied apps, and if questionable actions on the cloud take place, the CFI is tasked with gathering and reviewing evidence. The last resource component is the log ledger which contains a set of recorded results with a timestamp. Therefore, it serves as a helpful proof for initiating legal action against a suspect. The ledger aids in the preservation of the chronology of created logs.

Table 1 contains all acronymous and abbreviations used in this paper, while Tab. 2 summarizes the symbols used.

Tab. 2. Symbols used in of proposed evidence integrity preservation mechanism.

Symbol	Description
M_{PWD}	Password of node controller
M_{ID}	User ID of node controller
K_M	Key of node controller
T	Time stamp
Hd	Host ID
L	Log file
P	Node controller program to record log file
pk	Public key
$\otimes$	Interpolation
$\oplus$	Ex-or operation
R_{Hd}	Requested ID
$\mathrm{en}(.)$	ECC encryption
S_M	Service name
$K(.)$	Kernel transform
r	Random number
REQ	Integrity assurance request message
$h(.)$	Hashing
S_{PWD}	Session password
K_L	Hypervisor key
A	Acknowledgement message
Q_{PWD}	CFI Password
Q_{Hd}	CFI ID
IA	Integrity assurance

3.2. Creation of Log File

At this stage, the user password M_{Hd} and user ID M_{PWD} are formed by the user, which accumulating logs from every each resource of the virtual podium. M_{Hd} and M_{PWD} are saved in the hypervisor as M^*_{Hd} and M^*_{PWD}. The node controller key is produced after obtaining the user's credentials. The key is created by XOR-ing the public key and the modulus of a random key. After combining the resulting product with the encrypted user ID, the modulus is used to generate the node controller key as:

$$K_M = \left|\mathrm{en}(M^*_{Hd}) \bmod (r) \oplus \mathrm{pk}\right|. \tag{1}$$

The Log L is formed in the hypervisor with the requested ID of user R_D, time stamp T, and service name S_M for Hd, as:

$$L = \langle Hd, T, R_D, S_M \rangle. \tag{2}$$

Similarly, the node controller key K_M, L, and encoded node controller programmes for recording the log files are provided to the node controller as:

$$K_M, L, \mathrm{en}(P). \tag{3}$$

Thus, the node controller key is saved as K^*_M and the node controller program is set to trace the log file.

3.3. Key Setup and Exchange Process

Once the log file generation procedure is has been completed, three entities: the node controller, hypervisor, and log ledger, are used to start the key setup and exchange process. The stored node controller keys are given to the hypervisor and saved as K^{*R}_M. The hypervisor key is created by adding the encrypted node controllers programs and the kernel transformation of an arbitrary integer with the stored key. The final hypervisor key is generated by XOR-ing the acquired result with the hash timestamp Ts, which is modelled as:

$$K_L = h(Ts) \oplus \mathrm{en}(p) || K(r). \tag{4}$$

The hash of the finalized hypervisor key is proceeded here to generate the session password S_{PWD}, which is then supplied to the node controller and saved as S^*_{PWD}. As a result, the session password is:

$$S_{\mathrm{PWD}} = h(K_L). \tag{5}$$

Next, the acquired hypervisor key is saved as K^*_L in the log ledger. The hash of the stored hypervisor key is saved in M_1 and passed to the node controller, where it is saved as M^*_1:

$$M_1 = h(K^*_L). \tag{6}$$

If $M^*_1 = S_{\mathrm{PWD}}$, the ledger is confirmed and the confirmation occurs in the node controller.

3.4. Evidence and Contract Phase

For the sake of achieving privacy, the CFI and smart contract are used. The CFI ID Q_{Hd} and password Q_{PWD} are created during this phase and the credentials are passed into a the smart contract that is saved as Q^*_{Hd} and Q^*_{PWD}. Furthermore, the acknowledgment packet containing an encoded node control program, an the encoded ID, the cloud hashing forensic researcher ID, and the header are given to CFI and saved as A^*:

$$A = \langle \mathrm{en}(P), \mathrm{en}(R), h(Q_{Hd}), \mathrm{Header} \rangle. \tag{7}$$

The evidence is formed by XOR-ing the hash message and secure CFI ID, which is kept in the smart contract as EP^* and modelled as:

$$\mathrm{EP} = h(A^*) \oplus \mathrm{en}(Q, D). \tag{8}$$

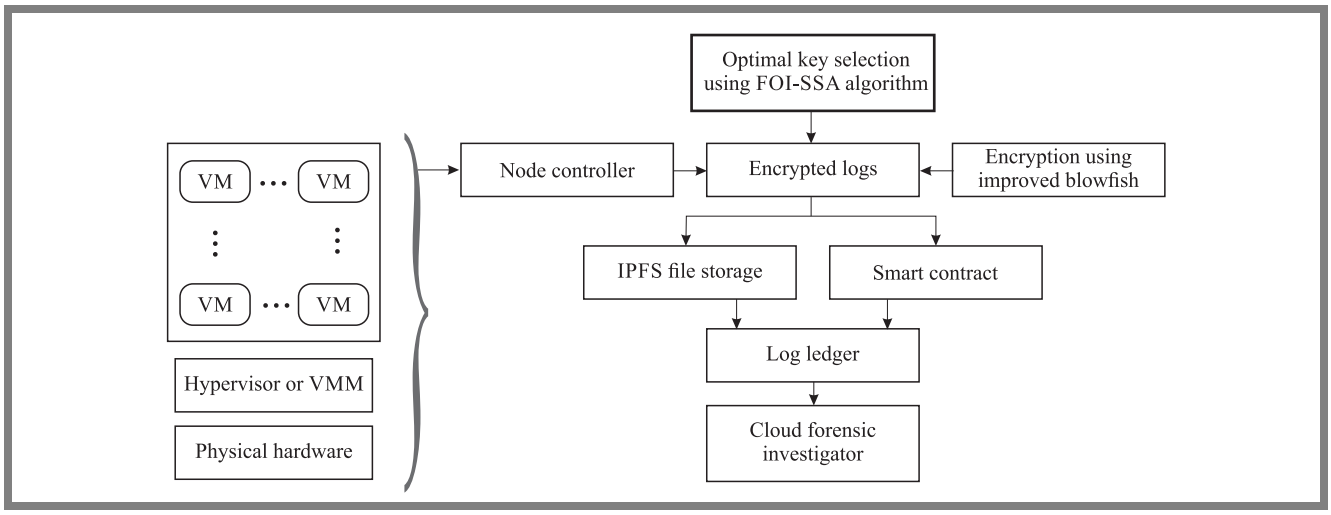

Fig. 1. Overall system model block diagram for evidence integrity preservation for cloud forensics.

The confirmation message is created by XOR-ing the saved evidence with the hashed message that is written as:

$$V = \mathrm{EP}^* \oplus h(A). \tag{9}$$

If $V = \mathrm{en}(Q_{Hd})$, then the validation was is considered accomplished and the information was is sent to CFI.

3.5. Integrity Assurance

Three entities are involved in this process: node controller, hypervisor, and CFI. The user ID, the request message and the stored key of the node controller are initially verified by the hypervisor that is expressed as:

$$M_{Hd}, \mathrm{REQ}, K_M^*. \tag{10}$$

The user ID and the saved node controller key are verified by the hypervisor. Then, the message is created by XOR-ing the hash node controller key, the encoded shared key, and the hash hypervisor key, such as:

$$M_1 = h(K_N^*) \oplus \mathrm{en}(\mathrm{pk}) \oplus h(K_L). \tag{11}$$

The login password is formed by XOR-ing the hashed timestamp with the encoded public key as:

$$S_{\mathrm{PWD}} = h(Ts) \oplus \mathrm{en}(\mathrm{pk}). \tag{12}$$

The resultant message and session passcode are given into the CFI, which performs integrity assurance operations by XOR-ing the message and session passwords and storing the result in the hypervisor as:

$$\mathrm{IA} = (M_1^* \oplus S_{\mathrm{PWD}}). \tag{13}$$

Level 2 verification is calculated based on integrity security operations, using the saved integrity assurance and the saved message as:

$$V_2 = I_A^* \oplus M_1^*. \tag{14}$$

If $V_2 = S_{\mathrm{pwd}}$, the guaranty is permitted. The required message and the CF ID are then sent to the node controller where the CF ID is saved.

3.6. Agreement and Confidential File Release

For agreement and private file release, the node controller, the CFI, and the blockchain network are considered. The CF ID and the request message are supplied to the node controller in this phase. Validation of the CF ID is carried out here in order to determine which one is genuine. Furthermore, two messages are produced, the first being formed by XOR-ing the encoded log and the encoded public key as:

$$R = \mathrm{en}(\log) \oplus \mathrm{en}(\mathrm{pk}). \tag{15}$$

The other message is created by combining the public key with a value, and then combining the result with the hash Req message as:

$$R_G^* = H(\mathrm{IREQ})||(\mathrm{pk})\Theta r. \tag{16}$$

The last message is created by XOR-ing the first and second messages, and then feeding them to the CFI and storing them as M_1^*, as shown in:

$$R^R = M_1^* \oplus R_G^Q. \tag{17}$$

The CFI creates two messages, the first of which is created by XOR-ing the last message with the second message. The second message is created by combining the public key with a randomized value, and then combining the result with the hash Req packet as:

$$R_G^Q = H(\mathrm{IREQ})||(\mathrm{pk}\Theta r). \tag{18}$$

By XOR-ing the initial message with the encoded public key, the log is created. If $\langle S, P\rangle = I(\log)$, then it is found to be satisfactory and is sent again, and then a contact between the CFI and the blockchain network is created.

3.7. Improved Blowfish Algorithm

The blowfish scheme is highly efficient and is suitable for hardware implementation and related modeling [35]. However, to enhance the key management mechanism, a modified version of blowfish is introduced, as follows:

– the input includes 64 bit data,

– it includes 64-bit block ciphers with irregular key lengths,
– it includes four 32-bit **S** arrays and P boxes. The **S** array has 18 of 32-bit subkeys, while each P box comprising 256 entries,
– it comprises two elements: a key-expansion part and a data-encryption part.

The F operation employs four substitution boxes, each consisting 256 32-bit entries [36]. Conventionally, if block XL is divided to 8-bit blocks a, b, c, d, then the operation $F(\mathrm{XL})$ is shown as in Eq. (19). As per the modified blowfish model, $F(\mathrm{XL})$ is modelled as in Eq. (20):

$$F(\mathrm{XL}) = [(P_{1,a} + P_{2,b}\, 2^{32}) \oplus P_{3,c}] + P_{4,d}\, 2^{32}, \quad (19)$$

$$F(\mathrm{XL}) = [(P_{1,a} \oplus P_{3,c}) + (P_{2,b} \oplus P_{4,d})\, 2^{32}]. \quad (20)$$

3.8. Blockchain-based Communication Phase

The blockchain-oriented communication stage is used to securely store and process logs, allowing for effective control of access to CFI and log integrity checking. An attack on the cloud can be carried out by a malevolent employee or an outside attacker. The functions that take place on the cloud platform generate logs for each VM operation, such as network interaction and VM setup logs. These logs are not power-independent, which means that if the VM is turned off, the data stored therein are lost. The suggested technique retrieves logs from Internet platforms and stores them in secondary memory storage to ensure data security and integrity.

The keys denoted by pk are optimally chosen via the FOI-SSA model. Figure 2 shows solutions in which nn indicates the overall count of keys. The objective Obj is to raise the key breaking time kbr as:

$$\mathrm{Obj} = \max(\mathrm{kbr}). \quad (21)$$

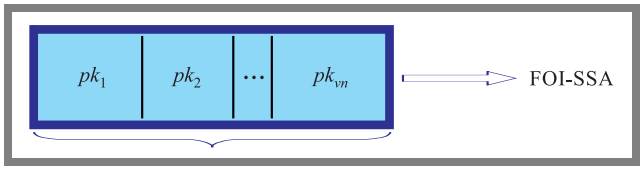

Fig. 2. Solution encoding scheme.

3.9. Proposed FOI-SSA Model for Optimal Key Generation

To achieve better convergence, FIO [37] is combined with the SSA model [38] to form FOI-SSA. Self-improvement of the optimization schemes results in better accuracy [39]– [42]. The behavior of the sparrows and formulated corresponding rules are described as:

1) The producers are typically highly energized. Assessment of each person's fitness values generates information about their energy reserves;
2) As individual sparrows start to chirp, the producers are required to direct all scavengers to the safe area when the alert value exceeds the safety level;
3) Every sparrow proceeds to production in accordance with how often it seeks out larger food sources, but the ratio of scavengers to producers becomes higher across the board. The producers would act as he sparrows with maximum energy levels. Numerous starving scavengers are inclined to fly towards different locations in search for food, to gain energy;
4) Scroungers look for food by emulating a farmer who actually produces the healthiest food. To increase their predation rate, certain scavengers may keep a tight eye on the producers and engage in food wars;
5) Sparrows in the center of the group haphazardly walk to be close to others when the sparrows at the group's periphery are aware of danger and quickly go into the safe area to take a better position.

As per FOI-SSA, the chaotic-based OBL is performed to generate opposite solutions that ensure a good convergence rate.

The location of the sparrows is represented by:

$$Y = \begin{bmatrix} Y_{1,1} & Y_{1,2} & \dots & Y_{1,a} \\ Y_{2,1} & Y_{2,2} & \dots & Y_{2,a} \\ \vdots & \vdots & \vdots & \vdots \\ Y_{s,1} & Y_{s,2} & \dots & Y_{s,a} \end{bmatrix}. \quad (22)$$

This implies the sparrow count and the size of the optimized variable. The fitness of the sparrow is defined by Eq. (23), which also addresses the fitness of the individuals.

$$F_Y = \begin{bmatrix} f\,[(Y_{1,1} & Y_{1,2} & \dots & Y_{1,a})] \\ f\,([Y_{2,1} & Y_{2,2} & \dots & Y_{2,a})] \\ \vdots & & \vdots & \vdots \\ f\,([Y_{s,1} & Y_{s,2} & \dots & Y_{s,a})] \end{bmatrix}. \quad (23)$$

The locations of producers are updated as per rules 1–2 and:

$$Y_{c,d}^{r+1} = \begin{cases} Y_{c,d}^{r}\, \mathrm{e}^{\frac{-c}{\alpha.\mathrm{it}_{\max}}} & \text{if } C_2 < \mathrm{st} \\ Y_{c,d}^{r} + P.M & \text{if } C_2 \geqslant \mathrm{st} \end{cases}. \quad (24)$$

In Eq. (24), r represents the iteration, $_{\max}$ implies the maximum iteration, α is an arbitrary integer, st and C_2 are the safety threshold and the alarm value, and P is an arbitrary integer. $\mathbf{M}$ denotes a matrix of $1 \times d$ with element 1.

The scroungers follow rules 4–5. As stated earlier, various scroungers track producers. In FOI-SSA, the scrounger's position is updated using FIO as:

$$Y(i)_{\mathrm{new}} = Y_{B_{ij}} + ra_{10}^{*}(Y_{B_{ij}} - Y_{B_{rj}}) + \\ + ra_{11}^{*}(Y_{\mathrm{best}} - Y_{B_{ij}}), \quad (25)$$

where ra_{10} and ra_{11} are arbitrary integers (0 and 1), Y_{best} implies the best position, B_i denotes the agent. In addition, in FOI-SSA, Cauchy's mutation is performed as:

$$Y(i)_{\mathrm{new}} = Y_{\mathrm{best}} + Y_i^{*}\mathrm{cauchy}(0, 1). \quad (26)$$

In addition, the model as per rule (6) is:

$$Y_{c,d}^{r+1} = \begin{cases} Y_{\text{best}}^{r} + \gamma . \left| Y_{c,d}^{r} - Y_{\text{best}}^{r} \right| & \text{if } f_c < f_u \\ Y_{c,d}^{r} + Z . \left(\frac{\left| Y_{c,d}^{r} - Y_{\text{worst}}^{r} \right|}{(f_c - f_w) + \varepsilon} \right) & \text{if } f_c = f_u \end{cases}, \quad (27)$$

where γ indicates the step size control parameter with a variance of 1 and a mean value of 0, R_{host} denotes the current global optimal location, $Z \in [-1, 1]$ denotes the route of the sparrow, f_c stands for the fitness value of the current sparrow, f_w and f_u are the worst fitness value and the current global best, ε is a small constant for avoiding the zero-division-error, and Y_{host} denotes the position at the center of the population.

4. Results and Discussion

The proposed CF integrity management plan has been created using Java and CloudSim. The performance of the FOI-SSA system was computed over EB [2], AES, ECC, RSA, El-Gamal, Signcryption, ECC + IPFS [43], DBO, BES, SSO, FIO, and SSA, taking into consideration such metrics as memory, detection rate, etc. In this case, the assessment was performed by altering the key size from 64 to 128 and the user count from 200 to 400.

4.1. Detection Rate Analysis

The detection accuracy of the proposed FOI-SSA algorithm is evaluated in comparison with traditional methods, for various key sizes of 64 and 128. Estimates concerning the FOI-SSA scheme, made over EB [2], AES, ECC, RSA, El-Gamal, Signcryption, ECC + IPFS [34], DBO, BES, SSO, FIO and SSA approaches are presented in Figs. 3 and 4 for user counts of 100, 200, 300 and 400. Here, the proposed FOI-SSA model showed an enhanced detection rate over EB, AES, ECC, RSA, El-Gamal, Signcryption, ECC + IPFS, DBO, BES, SSO, FIO and SSA. In Fig. 3, a higher detection rate is observed for FOI-SSA, with the user count of 100 for a key size of 64. With an increase in user count, the detection rates for FOI-SSA decreased for a key size of 64. This progression is the result of the enhanced blowfish concept and the integrated optimal key creation. Thus, the benefit of FOI-SSA is recognized over EB, AES, ECC, RSA, El-Gamal, Signcryption, ECC + IPFS, DBO, BES, SSO, FIO, and SSA.

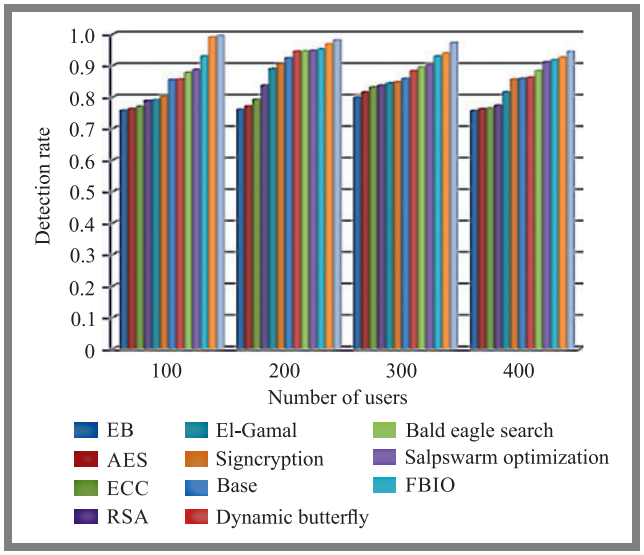

Fig. 3. Detection rate of FOI-SSA vs. other approaches, for a key size of 64.

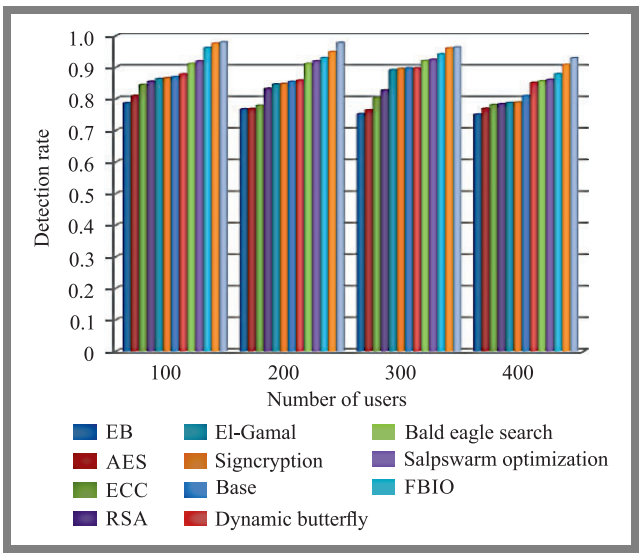

Fig. 4. Detection rate of FOI-SSA vs. other approaches, for a key size of 128.

4.2. Memory Usage Analysis

Figures 5–6 show an analysis of memory usage for FOI-SSA and other algorithms, for 128 and 64 key sizes. 8.5 MB of memory are used FOI-SSA for 100 users and a 64 key size, while other algorithms achieved higher utilization rates. Memory usage grows along with an increase in user count. These data help choose the optimal key and improve the blowfish concept.

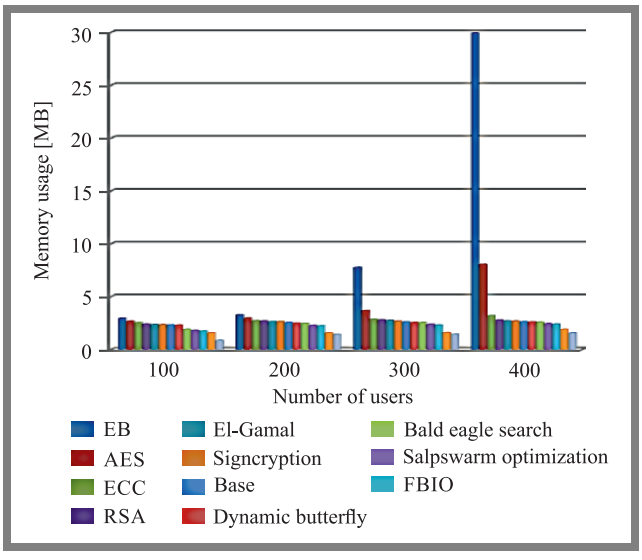

Fig. 5. Memory usage analysis of FOI-SSA vs. other approaches, for a key size of 64.

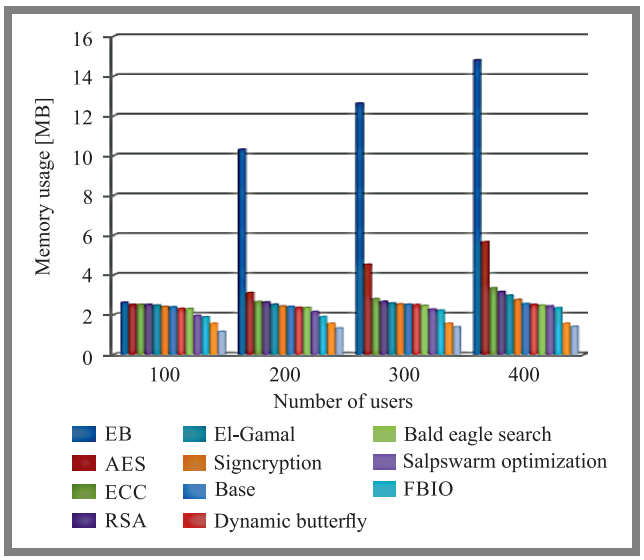

Fig. 6. Memory usage analysis of FOI-SSA vs. other approaches, for a key size of 128.

4.3. Time Analysis

Tables 3 and 4 show computational times for 128 and 64-bit keys. The analysis was performed for various user counts. For all key sizes, the time increases along with the user count. For 100 users, the computational time is shorter, but when the user count reaches 400, the time is longer for all other methods. However, FOI-SSA achieved a shorter time interval than its competitors. These advances are the result of using the blowfish concept and creating the optimal key.

4.4. Encryption and Decryption Time Analysis

The encryption time for various key sizes is summarized in Tables 5–6, while Tables 7–8 show the decoding time. For FOI-SSA, the encryption time is shorter for each key size. For user counts of 100 and 200, the decryption time is shorter with a 64-bit key which also requires less computational time for encryption. Thus, FOI-SSA achieves the shortest decryption and encryption times compared with its competitors, as shown in the tables.

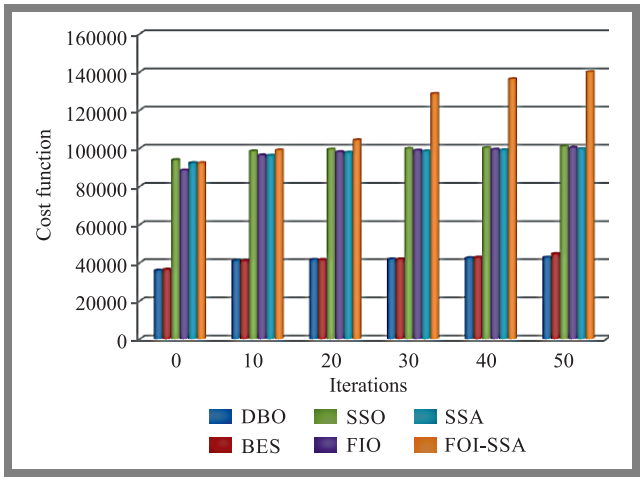

Fig. 7. Convergence analysis: FOI-SSA vs. other schemes.

4.5. Convergence Analysis

Convergence analysis of the proposed FOI-SSA system and the comparison of its performance with former models is shown in Fig. 7. FOI-SSA offers enhanced outcomes – the key break time at the 50-th iteration is 140004, meaning that it is higher than in the case of DBO, BES, SSO, FIO, and SSA. DBO's poor results were disclosed by obtaining a reduced key break time. Therefore, the goal is achieved, as shown in Eq. (21).

4.6. Attack Analysis

Figures 8 and 9 show the outcomes of research on various attack types, including inside and password spoofing attacks, for various key sizes and for varied user counts. Figure 8 shows the average key breakage time for inside and password spoofing attacks. While vulnerable to insider and password spoofing attacks, the proposed FOI-SSA approach has revealed a higher key break time. This is achieved due to a better blowfish concept and optimal key creation in FOI-SSA.

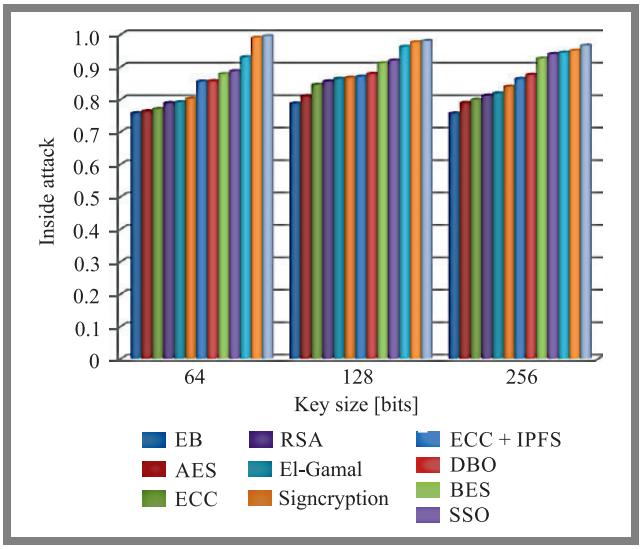

Fig. 8. Inside attack: FOI-SSA vs. other schemes.

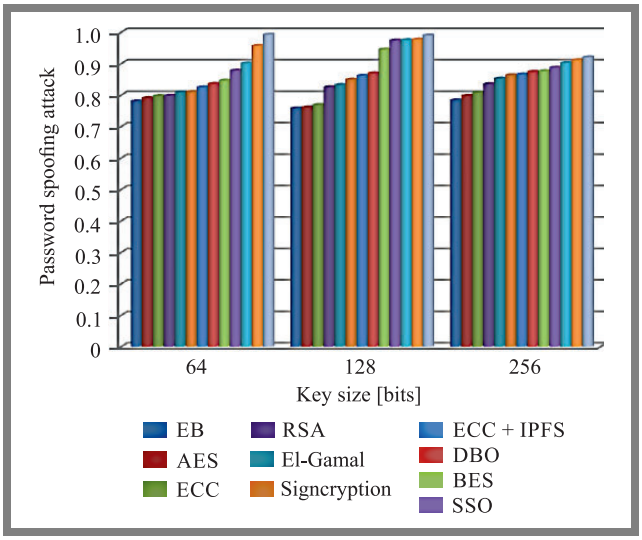

Fig. 9. Password spoofing attack: FOI-SSA vs. competitors.

Tab. 3. Time analysis using FOI-SSA over others algorithms for a key size of 64 [s].

User count	EB	AES	ECC	RSA	El-Gamal	Signcryption	ECC + IPFS	DBO	BES	SSO	FIO	SSA	FOI-SSA
100	505	504	503	402	351	324	327	221	184	146	110	74	35
200	506	505	505	403	351	324	323	222	185	147	110	74	36
300	509	509	508	405	352	325	324	222	185	147	110	74	36
400	712	709	707	504	453	426	425	223	186	149	110	73	36

Tab. 4. Time analysis using FOI-SSA over others algorithms for a key size of 128 [s].

User count	EB	AES	ECC	RSA	El-Gamal	Signcryption	ECC + IPFS	DBO	BES	SSO	FIO	SSA	FOI-SSA
100	613	612	612	407	355	327	326	217	181	145	109	73	35
200	643	642	641	472	414	377	375	223	187	149	110	73	35
300	702	701	701	487	420	393	391	224	187	149	112	73	37
400	790	789	787	598	546	519	518	226	188	150	113	74	37

Tab. 5. Encryption time for FOI-SSA vs. others algorithms for a key size of 64 [s].

User count	EB	AES	ECC	RSA	El-Gamal	Signcryption	ECC + IPFS	DBO	BES	SSO	FIO	SSA	FOI-SSA
100	25	25	17	16	16	16	16	16	28	0.8	0.7	0.3	0.3
200	25	25	17	17	16	16	16	16	28	0.8	0.7	0.4	0.3
300	25	25	17	17	17	16	16	16	29	0.8	0.7	0.4	0.3
400	27	26	17	17	17	17	16	16	29	1	0.8	0.4	0.3

Tab. 6. Encryption time for FOI-SSA vs. others algorithms for a key size of 128 [s].

User count	EB	AES	ECC	RSA	El-Gamal	Signcryption	ECC + IPFS	DBO	BES	SSO	FIO	SSA	FOI-SSA
100	25	25	16	16	16	17	16	16	3	0.8	0.6	0.4	0.3
200	25	25	17	17	17	17	16	16	3	0.9	0.8	0.4	0.3
300	35	26	17	17	17	17	16	16	3	0.9	0.8	0.5	0.5
400	55	27	18	17	17	17	17	16	4	1	1	0.8	0.7

Tab. 7. Decryption time for FOI-SSA vs. others algorithms for a key size of 64 [s].

User count	EB	AES	ECC	RSA	El-Gamal	Signcryption	ECC + IPFS	DBO	BES	SSO	FIO	SSA	FOI-SSA
100	76	75	48	17	16	16	16	16	16	0.6	0.194	0.2	0.1
200	76	75	48	17	17	16	16	16	16	0.7	0.241	0.2	0.1
300	77	76	49	17	17	16	16	16	16	0.7	0.241	0.2	0.1
400	177	177	50	17	17	17	16	16	16	0.7	0.265	0.2	0.1

Tab. 8. Decryption time for FOI-SSA vs. others algorithms for a key size of [s].

User count	EB	AES	ECC	RSA	El-Gamal	Signcryption	ECC + IPFS	DBO	BES	SSO	FIO	SSA	FOI-SSA
100	124	75	49	16	17	16	16	16	15	0.6	0.2	0.2	0.1
200	179	77	49	17	16	16	16	16	16	0.7	0.3	0.2	0.1
300	259	117	49	17	17	16	16	16	16	0.7	0.4	0.3	0.1
400	275	142	68	17	17	17	17	16	16	1	0.5	0.3	0.2

5. Conclusion

This paper proposes a novel forensic method relying on a blockchain network. In order to protect the system against illegitimate users, an improved blowfish method is used. The system is made up of five distinct elements: hypervisor, node controller, log ledger, IPFS file storage, and smart contract. The suggested method entails six phases, including determination of the log file concept, key arrangement and exchange process, setup and control of evidence, assurance of integrity, agreement validation, release of the confidential file, and the blockchain-based communication phase. The proposed FOI-SSA approach offers an enhanced detection rate compared with EB, AES, ECC, RSA, El-Gamal, Signcryption,

ECC+IPFS, DBO, BES, SSO, FIO, and SSA algorithms. A higher detection rate was observed for FOI-SSA for a user count of 100 and a key size of 64. As the user count increases, the detection rate of FOI-SSA decreases for a key size of 64.

References

[1] S. Rane and A. Dixit, "BlockSLaaS: Blockchain assisted secure logging-as-a-service for cloud forensics", *Proceedings of International Conference on Security & Privacy*, pp. 77–88, 2019 (https://doi.org/10.1007/978-981-13-7561-3_6).

[2] P. Jain, "Decentralize log file storage and integrity preservation using blockchain", *International Journal of Computer Science and Information Technologies*, vol. 11, no. 2, pp. 21–30, 2020 (https://ijcsit.com/docs/Volume%2011/vol11issue02/ijcsit2020110202.pdf).

[3] M. Pourvahab and G. Ekbatanifard, "Digital forensics architecture for evidence collection and provenance preservation in IaaS cloud environment using SDN and blockchain technology", *IEEE Access*, vol. 7, pp. 153349–153364, 2019 (https://doi.org/10.1109/ACCESS.2019.2946978).

[4] N. Dalezios, S. Shiaeles, N. Kolokotronis, and B. Ghita, "Digital forensics cloud log unification: Implementing CADF in Apache CloudStack", *Journal of Information Security and Applications*, vol. 54, pp. 102555, 2020 (https://doi.org/10.1016/j.jisa.2020.102555).

[5] C. Stelly and V. Roussev, "SCARF: A container-based approach to cloud-scale digital forensic processing", *Digital Investigation*, vol. 22, pp. S39–S47, 2017 (https://doi.org/10.1016/j.diin.2017.06.008).

[6] J.H. Park, J.Y. Park, and E.N. Huh, "Blockchain based data logging and integrity management system for cloud forensics", *Computer Science & Information Technology*, vol. 149, 2017 (https://doi.org/10.5121/csit.2017.71112).

[7] T.K. Dasaklis, F. Casino, and C. Patsakis, "SoK: Blockchain solutions for forensics", in *Technology Development for Security Practitioners*, B. Akhgar, D. Kavallieros, and E. Sdongos, Eds. Springer Cham, 2021, pp. 21–40 (https://doi.org/10.1007/978-3-030-69460-9_2).

[8] M. Irfan, H. Abbas H, Y. Sun, A. Sajid, and M. Pasha, "A framework for cloud forensics evidence collection and analysis using security information and event management", *Security and Communication Networks*, vol. 9, no. 16, pp. 3790–3807, 2016 (https://doi.org/10.1002/sec.1538).

[9] A.C. Kumar and R. Vimala, "Load balancing in cloud environment exploiting hybridization of chicken swarm and enhanced raven roosting optimization algorithm", *Multimedia Research*, vol. 3, no. 1, pp. 45–55, 2020 (https://doi.org/10.46253/j.mr.v3i1.a5).

[10] R. Battistoni, R. Di Pietro, and F. Lombardi, "CURE-Towards enforcing a reliable timeline for cloud forensics: Model, architecture, and experiments", *Computer Communications*, vol. 91, pp. 29–43, 2016 (https://doi.org/10.46253/j.mr.v3i1.a5).

[11] H. Mahdi, *et. al*, "Vehicular Networks Performance Evaluation Based on Downlink Scheduling Algorithms for High-Speed Long Term Evolution–Vehicle", *International Journal of Interactive Mobile Technologies*, vol. 15, no. 21, 2021 (https://doi.org10.3991/ijim.v15i21.22475).

[12] Chao Lina, *at al.*, "BSeIn: A blockchain-based secure mutual authentication with fine-grained access control system for Industry 4.0", *Journal of Network and Computer Applications*, vol. 116, pp. 42–52, 2018 (https://doi.org/10.1016/j.jnca.2018.05.005).

[13] D.F. Maesa, P. Mori, and L. Ricci, "A blockchain based approach for the definition of auditable access control systems", *Computers and Security*, vol. 84, pp. 93–119, 2019 (https://doi.org/10.1016/j.cose.2019.03.016).

[14] P.T. Duy, H. Do Hoang, N.B. Khanh, and V.H. Pham, "Sdnlog-foren: Ensuring the integrity and tamper resistance of log files for SDN forensics using blockchain", *Proceedings of 2019 6th NAFOSTED Conference on Information and Computer Science (NICS)*, pp. 416–421, 2019 (https://doi.org/10.1109/NICS48868.2019.9023852).

[15] G.G. Dagher, J. Mohler, M. Milojkovic, and P.B. Marell, "Ancile: Privacy-preserving framework for access control and interoperability of electronic health records using blockchain technology", *Sustainable Cities and Society*, vol. 39, pp. 283–297, 2018 (https://doi.org/10.1016/j.scs.2018.02.014).

[16] Hao Gu, L. Wanxin, Mark Nejad, and Chien-Chung Shen, "Access Control for Electronic Health Records with Hybrid Blockchain-Edge Architecture", *2019 IEEE International Conference on Blockchain*, 2019 (https://doi.org/10.1109/Blockchain.2019.00015).

[17] B. Prasanalakshmi., A. Kannammal, and R. Sridevi, "Multimodal biometric cryptosystem involving face, fingerprint and palm vein", *International Journal of Computer Science Issues (IJCSI)*, vol. 8, no. 4. pp. 604, 2011 (URL: http://ijcsi.org/papers/IJCSI-8-4-1-604-610.pdf).

[18] S. Kokilavani, M.D. Vignesh, E. Surya, A. Narendran, and B. Prasanalakshmi, "Enhanced biometric smart key scheme for smart card authentication", *2014 International Conference on Communication and Signal Processing*, pp. 1589–1592. IEEE, 2014 (https://doi.org/10.1109/ICCSP.2014.6950116).

[19] V. Rupapara, F. Rustam, H.F. Shahzadehmood, A. Ashraf, and G.S. Choi, "Impact of SMOTE on imbalanced text features for toxic comments classification using RVVC model", *IEEE Access*, vol. 9, pp. 78621–78634 (https://doi.org/10.1109/ACCESS.2021.3083638).

[20] M. Irfan, H. Abbas, and W. Iqbal, "Feasibility analysis for incorporating/deploying SIEM for forensics evidence collection in cloud environment", *Proceedings of 2015 IEEE/ACIS 14th International Conference on Computer and Information Science (ICIS)*, pp. 15–21, 2015 (https://doi.org/10.1109/ICIS.2015.7166563).

[21] K. Kent, S. Chevalier, T. Grance, and H. Dang, "Guide to integrating forensic techniques into incident response", *NIST Special Publication*, vol. 10, no. 14, pp. 800–806, 2006 (https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-86.pdf).

[22] S.K. Manoj and D.L. Bhaskari, "Cloud forensics – a framework for investigating cyber attacks in cloud environment", *Procedia Computer Science*, vol. 85, pp. 149–154, 2016 (https://doi.org/10.1016/j.procs.2016.05.202).

[23] M. Ma, G. Shi, and F. Li, "Privacy-Oriented Blockchain-based Distributed Key Management Architecture for Hierarchical Access Control in the IoT Scenario", *IEEE Access*, vol. 7, pp. 34045–34059, 2019 (https://doi.org/10.1109/ACCESS.2019.2904042).

[24] L. Pasquale, S. Hanvey, M. Mcgloin, and B. Nuseibeh, "Adaptive evidence collection in the cloud using attack scenarios", *Computers and Security*, vol. 59, pp. 236–54, 2016 (https://doi.org/10.1016/j.cose.2016.03.001).

[25] V.K, Netajihole and G.P. Bhole, "Optimal Container Resource Allocation Using Hybrid SA-MFO Algorithm in Cloud Architecture", *Multimedia Research*, vol. 3, no. 1, pp. 11–20, 2020 (https://doi.org/10.46253/j.mr.v3i1.a2).

[26] A.C. Kumar and R. Vimala, "Load Balancing in Cloud Environment Exploiting Hybridization of Chicken Swarm and Enhanced Raven Roosting Optimization Algorithm", *Multimedia Research*, vol. 3, no. 1, pp. 45–55, 2020 (https://doi.org/10.46253/j.mr.v3i1.a5).

[27] M.K. Mahesh, "Workflow scheduling using Improved Moth Swarm Optimization Algorithm in Cloud Computing", *Multimedia Research*, vol. 3, no. 3, 2020 (https://doi.org/10.46253/j.mr.v3i3.a5).

[28] M. Pourvahab and G. Ekbatanifard, "An efficient forensics architecture in software-defined networking-IoT using blockchain technology", *IEEE Access*, vol. 7, pp. 99573–99588, 2019 (DOI: 10.1109/ACCESS.2019.2930345).

[29] P. Santra, P. Roy, D. Hazra, and P. Mahata, "Fuzzy data mining-based framework for forensic analysis and evidence generation in cloud environment", *Ambient Communications and Computer Systems*, pp. 119–129, 2018 (https://doi.org/10.1007/978-981-10-7386-1_10).

[30] S. Aishwarya, R.S. Selvi, and R. Ilakkiya, "Secure public Auditing With Code-Regeneration In Multi Storage Cloud Architecture", 2006 (https://ijartet.com/1370/v3s15ranganathancseit/conference).

[31] S. Wang, Y. Zhang, and Y. Zhang, "A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems", *IEEE Access*, vol. 6, pp. 38437–38450, 2018 (https://doi.org/10.1109/ACCESS.2018.2851611).

[32] X. Qi, E.B. Sifah, K.O. Asamoah, J. Gao, X. Du, and M. Guizani, "MeDShare: Trust-less medical data sharing among cloud service providers via blockchain", *IEEE Access*, vol. 5, pp. 14757–14767, 2017 (https://doi.org/10.1109/ACCESS.2017.2730843).

[33] S. Zawoad, A.K. Dutta, and R. Hasan, "Towards building forensics enabled cloud through secure logging-as-a-service", *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 2, pp. 148–62, 2015 (https://doi.org/10.1109/TDSC.2015.2482484).

[34] P.P. Srivastava, S. Goyal, and A. Kumar, "Analysis of various NoSql database", *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, pp. 539–544, 2015 (Dhttps://doi.org/10.1109/ICGCIoT.2015.7380523).

[35] M. Agrawal and P. Mishra, "A Modified Approach for Symmetric Key Cryptography Based on Blowfish Algorithm", *International Journal of Engineering and Advanced Technology (IJEAT)*, vol. 1, no. 6, 2012 (https://www.ijeat.org/wp-content/uploads/papers/v1i6/F0610071612.pdf).

[36] R.K. Meyers and A.H. Desoky, "An Implementation of the Blowfish Cryptosystem", *2008 IEEE International Symposium on Signal Processing and Information Technology*, 2008 (https://doi.org/10.1109/ISSPIT.2008.4775664).

[37] J-S. Chou and N-M. Nguyen, "FBI inspired meta-optimization", *Applied Soft Computing*, vol. 93, 2020 (https://doi.org/10.1016/j.asoc.2020.106339).

[38] J. Xue and B. Shen, "A novel swarm intelligence optimization approach: sparrow search algorithm", *Systems Science & Control Engineering*, vol. 8, no. 1, pp. 22–34, 2020 (https://doi.org/10.1080/21642583.2019.1708830).

[39] M.M. Beno, I.R. Valarmathi, S.M. Swamy, and B.R. Rajakumar, "Threshold prediction for segmenting tumour from brain MRI scans", *International Journal of Imaging Systems and Technolog*, vol. 24, no. 2, pp. 129–137, 2014 (https://doi.org/10.1002/ima.22087).

[40] T. Renjith and M.J.S. Rangachar, "Hybrid Optimization based DBN for Face Recognition using Low-Resolution Images", *Multimedia Research*, vol. 1, no. 1, pp. 33–43, 2018 (https://doi.org/10.46253/j.mr.v1i1.a5).

[41] J. Devagnanam, N.M. Elango, "Optimal Resource Allocation of Cluster using Hybrid Grey Wolf and Cuckoo Search Algorithm in Cloud Computing", *Journal of Networking and Communication Systems*, vol. 3, no. 1, pp. 31–40, 2020 (https://doi.org/10.46253/jnacs.v3i1.a4).

[42] S.K.M. Shareef and R.S. Rao, "A Hybrid Learning Algorithm for Optimal Reactive Power Dispatch under Unbalanced Conditions", *Journal of Computational Mechanics Power System and Control*, vol. 1, no. 1, pp. 26–33, 2018 (https://doi.org/10.46253/j.mr.v1i1.a5).

[43] P. Purnaye and V. Kulkarni, "BiSHM: Evidence detection and preservation model for cloud forensics", *Open Computer Science*, vol. 12, no. 1, 2022, pp. 154–170 (https://doi.org/10.1515/comp-2022-0241).

Sahadev Maruti Shinde
E-mail: sahadevmaruti@gmail.com
GITAM School of Engineering, Visakhapatanam, Andhra Pradesh, India

Venkateswara Rao Gurrala
Professor in Department of Computer Science and Engineering
GITAM School of Engineering, Visakhapatanam, Andhra Pradesh, India