Received August 30, 2019, accepted October 1, 2019, date of publication October 11, 2019, date of current version October 31, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2946978

Digital Forensics Architecture for Evidence Collection and Provenance Preservation in IaaS Cloud Environment Using SDN and Blockchain Technology

MEHRAN POURVAHAB[1], **(Member, IEEE), AND GHOLAMHOSSEIN EKBATANIFARD**[2]

[1]Department of Computer Engineering, Islamic Azad University, Rasht Branch, Rasht, Iran
[2]Department of Computer Engineering, Islamic Azad University, Lahijan Branch, Lahijan, Iran

Corresponding author: Gholamhossein Ekbatanifard (ekbatanifard@liau.ac.ir)

This work was supported by the MehranNet ISP (Internet Service Provider), Langarud, Guilan, Iran.

ABSTRACT Cloud forensics is an intelligent evolution of digital forensics that defends against cyber-crimes. However, centralized evidence collection and preservation minimizes the reliability of digital evidence. To resolve this severe problem, this paper proposes a novel digital forensic architecture using fast-growing Software-Defined Networking (SDN) and Blockchain technology for Infrastructure-as-a-Service (IaaS) cloud. In this proposed forensic architecture, the evidence is collected and preserved in the blockchain that is distributed among multiple peers. To protect the system from unauthorized users, Secure Ring Verification based Authentication (SRVA) scheme is proposed. To strengthen the cloud environment, secret keys are generated optimally by using Harmony Search Optimization (HSO) algorithm. All data are encrypted based on the sensitivity level and stored in the cloud server. For encryption, Sensitivity Aware Deep Elliptic Curve Cryptography (SA-DECC) algorithm is presented. For every data stored in the cloud, a block is created in the SDN controller and the history of data is recorded as metadata. In each block, the Merkle hash tree is built by using Secure Hashing Algorithm-3 (SHA-3). Our system allows users to trace their data by deploying Fuzzy based Smart Contracts (FCS). Finally, evidence analysis is enabled by constructing Logical Graph of Evidence (LGoE) collected from the blockchain. Experiments are conducted in an integrated environment of java (for cloud and blockchain) and network simulator-3.26 (for SDN). The extensive analysis shows that proposed forensic architecture shows promising results in Response time, Evidence insertion time, Evidence verification time, Communication overhead, Hash computation time, Key generation time, Encryption time, Decryption time and total change rate.

INDEX TERMS Software-defined networking, blockchain, evidence collection, cloud forensics, security.

I. INTRODUCTION

In this high-tech era, an increase in demands of cloud infrastructure among industries, governments, and individuals results in a lack of security. With the cloud environment, private data of everyone becomes vulnerable against cyber-attacks [1]. According to the National Institute of Standards and Technology (NIST) [2], digital forensics is an applied study to identify an incident, collection, and examination of evidence data. Likewise, cloud forensics is defined as the application of digital forensic science in the cloud computing environment [3]. Reliable evidence collection in the cloud environment is done by CURE, which is a cloud forensic architecture [4]. Security aspects focused by CURE are time heartbeat, anti-forging mechanisms, and key management. Forensic architecture is proposed for Software-defined Networking (SDN) based Internet of Things (IoT) using blockchain [5]. Here linear homomorphic encryption scheme is adapted in the blockchain. Evidential data collection is also carried out in Software Defined Networking (SDN) platform [6].

The digital evidence collected and preserved by OpenFlow switches in which additional forensic tools are adapted for forensic analysis [7]. A provenance-aware data monitoring system (PDMS) is introduced and build upon the existing provenance tracking framework [8]. In the IaaS cloud,

The associate editor coordinating the review of this manuscript and approving it for publication was Luis Javier Garcia Villalba.

Mchain that is the blockchain-based integrity management method is proposed [9]. Thus many research efforts have been held on the SDN cloud environment using blockchain technology. In this work, we use blockchain technology for digital forensics in the cloud environment.

A. PRELIMINARY KNOWLEDGE ON BLOCKCHAIN FOR DIGITAL FORENSICS

However, in centralized forensic architecture, data integrity is a significant issue to be addressed. In order to cope with this specific issue, blockchain and smart contracts will be a better solution [10], [11]. Blockchain is the tamper-resistant timestamp based distributed ledger that often adapted for sharing and storing data [12]–[14]. The fundamental elements of blockchain are listed as follows:

- Decentralization: In blockchain architecture, the control is distributed among peers in the chain instead of given under a centralized authority. In the chain, each node is free to join or leave the blockchain network.
- Collective verification: All transactions made on blockchain are publicly verifiable, i.e., each transaction is verified by all other nodes in the chain. Further, it also provides tamper assistance, which means the data recorded in the blockchain cannot be modified or deleted.
- Security and integrity: All transactions are verified and stored in blocks under strong cryptographic functions. The involvement of digital signature preserves data security and integrity.

The general architecture of Blockchain is illustrated in Fig. 1. Each block contains a list of transactions and a hash value of its own and previous block along with a timestamp.

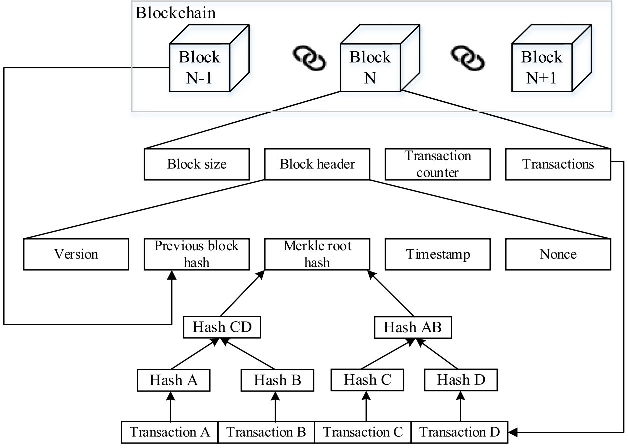


FIGURE 1. **Blockchain structure.**

To ensure tamper-proof records, a blockchain-based data provenance scheme known as ProvChain is introduced [15], which is the primary motivation behind our work. A privacy-preserving model for secure data storage is proposed using blockchain [16]. Blockchain-based forensic architecture is also utilized for vehicular network environment [17] to analyze the cases regarding accidents. In SDN, blockchain and smart contract are used to distributed denial-of-service (DDoS) attack detection and mitigation [18]. In addition, blockchain has been used in many IoT applications such as agricultural [19], industries [20], electronic voting [21], and smart grid applications [22].

B. MOTIVATION

User authentication is the foremost process of cloud forensic to ensure high-level security [23]. Here the significant aim is to prevent evidence from unauthorized users. For effectual authentication, email verification and one-time password (OTP) are utilized. Perhaps blockchain-based cloud forensic architecture is secure; there is also strong authentication is required for evidence provenance [24]. Elliptic Curve Cryptography (ECC) algorithm works better in terms of encryption time and key size. However, the efficiency of the ECC algorithm depends upon the initial key generation process. Inappropriate key generation impacts the security level provided by the ECC algorithm. Thus homomorphic computations based signature scheme is proposed for blockchain applications to improve the security level [25]. Strengths, weakness, opportunities, and threats (SWOT) analysis was conducted on current forensic networks in the cloud [26]. The analysis concludes that a robust forensic system is required in the cloud environment. Forensic analysis in IoT devices [27] and cloud [28] was also performed for evidence identification. In the blockchain, the smart contracts are an autonomous entity that automatically executes under some conditions. Data provenance is another challenging issue in the cloud environment [29]. Data provenance is defined as the source or the historical information of the data which is stored in cloud infrastructure. In digital forensics, data provenance plays a pivotal role. Data provenance clearly state the valuable information of an object (data) including when the data is accessed, who accessed the data, and how it was changed [30].

Therefore the primary motivation of this research is to design digital forensics architecture with the use of SDN and blockchain technology in the cloud environment. Furthermore, we also intend to adopt a strong authentication scheme, digital signature algorithm, and smart contracts for evidence collection and provenance.

C. MAIN CONTRIBUTIONS

In this paper, we made the following contributions to include additional knowledge to the digital forensics.

- Digital forensic architecture is designed for evidence collection,analysis, and provenance in the Infrastructure-as-a-Service (IaaS) cloud environment. For evidence collection, blockchain and SDN technologies are utilized.
- Evidence and the data are protected from unauthorized users by using Secure Ring Verification based Authentication (SRVA) scheme driven by an authentication server (AS). The involvement of the SRVA scheme allows users who have successfully completed the

secure verification process through circular theorem and secret key (SK).
- Sensitivity Aware Deep Elliptic Curve Cryptography (SA-DECC) algorithm is proposed for encryption and digital signature generation. To generate strong secret keys, Harmony Search Optimization (HSO) algorithm is utilized for key generation in SA-DECC. The main contribution of the SA-DECC algorithm is that the proposed algorithm is adaptive based on the sensitivity level of data.
- For each data stored in the cloud server, a block is created by the SDN controller and distributed over the blockchain network. For more security, Secure Hashing Algorithm-3 (SHA-3) is proposed for has computations in the blockchain. Data provenance is maintained by using Fuzzy based Smart Contracts (FSC) to track the activities of data throughout its lifecycle.
- In the forensic system, the investigator performs evidence identification, evidence collection, evidence analysis, and report generation. Evidence collection is supported by the SDN controller and analyzed by the investigator with the support of the Logical Graph of Evidence (LGoE) analysis method.
- Proposed research work preserves the chain of custody (CoC), proof of ownership (PoO) and evidence integrity to improve the reliability of evidence collection.

D. ORGANIZATION

The rest of this article is organized as follows: Section II reviews previous research works held on cloud forensics to identify the research gap. In section III, the major problems discovered from existing works are highlighted. In Section IV, proposed digital forensic architecture is detailed with necessary algorithms. In section V, we evaluate our proposed forensic architecture with previous research work based on experimental findings. In section VI, our contributions are concluded.

II. RELATED WORKS

In this section, we survey significant research works held on digital forensics in the cloud environment. In the past few years, many researchers have focused on digital forensics and blockchain technology to ensure security against cyber-attacks in the cloud environment.

Evidence collection in cloud forensic was concentrated in [31]. This article was attempted to mitigate the issues in evidence collection under the cloud service provider (CSP) control. To do this, all evidence was collected outside of CSP, i.e., forensic monitoring plane. In monitoring plane, a forensic server is deployed to collect and preserve all evidence. Perhaps, all evidence is protected from untrusted CSP; maintaining evidence at a single forensic server leads to a single point of failure. Thus attacker only needs to affect forensic server to alter and delete the evidence. Secure-Logging-as-a-Service (SecLaaS) model was presented to build cloud forensics architecture [32]. The SecLaaS was attempted to collect various logs without loss in integrity. For integrity preservation, hash chain scheme was employed and also proofs of past logs were published to cloud providers periodically. Log collection in a centralized manner increases the vulnerability of logs. To resolve the issue of dependency on CSP, the forensic acquisition and analysis system (FAAS) model was presented [33]. FAAS was an agent-assisted system in which all recorded evidence was controlled by agent coordinator and agent manager. FAAS fails to preserve data provenance which is the significant element of forensics. In addition, the involvement of various agents increases the complexity of the system.

A log aggregation model with the following processes: log extraction from the client-side, log acquisition from CSP side, log indexing, log normalization, log correlation, log sequencing, and presentation was proposed for digital forensic architecture [34]. All collected logs were stored in an evidentiary log repository for further analysis. However, the log repository is a centralized database that can be easily compromised by attackers. Security information and event management (SIEM) framework were designed for cloud forensics [35]. Here all evidence was distributed among instead of stored in CSP. For further security, Rivest Shamir and Adelman (RSA) encryption algorithm was utilized. All evidence is shared among users who are unauthenticated. This method increases the involvement of unauthorized users and the evidence may be shared with them too. Cloud forensic architecture was implemented with SDN controlled network as Forensic Controller (ForCon) [36]. The network environment was monitored and the evidence was collected by dislocated agents. Here again, evidence integrity, use of agents are major issues to be concentrated. A fuzzy-based data mining approach was introduced for forensic acquisition [37]. This fuzzy-based expert system was proposed for forensic monitoring, analysis, and evidence generation for cloud logs. All evidence was stored under the control of CSP. However, in the cloud environment, CSP could not be fully trusted, which decreases the reliability of evidence.

An adaptive evidence collection mechanism was proposed to handle dynamic configuration of cloud architecture [38]. To make evidence collection as adaptive, three different scenarios such as vulnerable database, security breaches, and cloud configuration are considered. Based on these configurations, the evidence collection process is adaptively updated. Perhaps this method is adaptive; this method is not able to provide data provenance and evidence integrity. The smart contracts based access control mechanism was introduced to track the behavior of data [39]. For this purpose, the user layer, data query layer, data structuring provenance layer, and existing database infrastructure layer were included in the architecture. The Smart contract, authenticator, processing, smart contract permissioned database, blockchain network, and consensus nodes were included in the data structuring layer and provenance layer. In this method, latency is increased with an increase in the number of users due to large tuple size and processing time.

From our critical survey, the major research challenge identified is centralized evidence collection and analysis for cloud forensics. Furthermore, the majority of the researchers have concentrated only on evidence collection and fail to ensure integrity for collected evidence. Thus centralized forensic architecture, data provenance, and evidence integrity preservation are still major issues in cloud forensics.

III. PROBLEM STATEMENT

Cloud Forensics log (CFLOG) framework was introduced for secure log collection under the control of CSP [40]. Perhaps, the logs are authenticated before collection; CFLOG fails to maintain the integrity of logs. CSP controls all logs which are not trustworthy to maintain the reliability of logs. The centralized framework is vulnerable to many security threats since the attacker only needs to compromise a single entity, i.e. CSP.

Fog enabled SDN architecture was introduced in the cloud environment to provide security in a distributed manner [41]. In both fog and cloud layer, blockchain is maintained and each request was processed in both layers. Thus the processing and response time are literally large in this system. Without effectual authentication, unauthorized users are also allowed into the system, which increases the vulnerability of the system. In an efficient and secure data provenance scheme (ESP), all users were authenticated based on ID and security was ensured by blockchain technology [42]. However, the considered authenticated credentials (ID and password) are easily cracked by attackers and not sufficient for strong authentication. For security, timestamp verification is used, which could be not accurate in the blockchain system.

The block-secure method was introduced for secure cloud storage and involved with ECC based signature scheme, SHA-256^2 based integrity verification [43]. Here SHA-256 is poor in security but rich in time consumption, i.e., use of SHA-256 algorithm twice increases time consumption to attain reasonable security level [44]. In addition, ECC based signature generation fully depends upon the prime number generation, which makes it not suitable to provide high-level security.

In this section, we highlighted the following problems,

- Centralized evidence collection and preservation
- Lack of integrity and security
- Unauthorized user access
- Issues in the hash generation and digital signature generation

All aforesaid problems are considered and resolved in our proposed cloud forensic architecture.

IV. PROPOSED CLOUD-FORENSIC USING BLOCKCHAIN

In this section, proposed forensic architecture termed as DFeSB is elaborated with necessary algorithms. Proposed cloud forensic utilizes SDN and blockchain technology for evidence collection and analysis.

A. SYSTEM OVERVIEW

The major objective of this research work is to collect reliable evidence from the cloud environment and to preserve data provenance for cloud data. The overall forensic system encompasses the following entities:

1) CLOUD USERS (U)

In our system 'm' number of cloud users ($U_1, U_2, .., U_m$) are participated. Cloud users are allowed to upload and download data from the cloud server.

2) AUTHENTICATION SERVER (AS)

Initially, all cloud users are registered with AS in order to avoid unauthorized user access. Major responsibilities of AS is key generation and authentication.

3) CLOUD SERVICE PROVIDER (CSP)

All data outsourced by cloud users are stored in cloud servers hosted by CSP. Blockchain is created for each data stored under CSP.

4) OPENFLOW SWITCHES ($OFSs$)

In this research work, SDN is utilized to collect evidence from CSP. Thus we have used multiple OFSs to forward users data to CSP. The major responsibility of OFSs is to transmit users data based on flow rules deployed by the controller. Flow rules are deployed and modified by the SDN controller only.

5) SDN CONTROLLER ($SDN - C$)

SDN controller is responsible for deploying flow rules according to the network status and for collecting all evidence from CSP. In SDN-C, blockchain is maintained for evidence collection and for each data stored in CSP, a block is created at CSP.

The overall system architecture is illustrated in Fig. 2. The major objective of our forensics architecture is to collect reliable evidence from CSP and to preserve data provenance. Initially, we establish an effectual authentication scheme to protect the system from unauthorized users. Data stored under CSP are encrypted based on the sensitivity level to preserve security in the cloud environment. Decentralized evidence collection was proposed based on blockchain technology. In order to track data history and to preserve data provenance, smart contracts are deployed. For efficient evidence analysis, the graph-based analysis method is proposed.

B. USER AUTHENTICATION

At first, all cloud users are registered with AS. The user credentials considered during registration are user ID (ID) and password (PW). For each registered U, AS generates a secret key (SK) using the HSO algorithm. Then all users are authenticated at each time using ID, PW, SK, and secret code (SC) that are generated by the circular theorem.

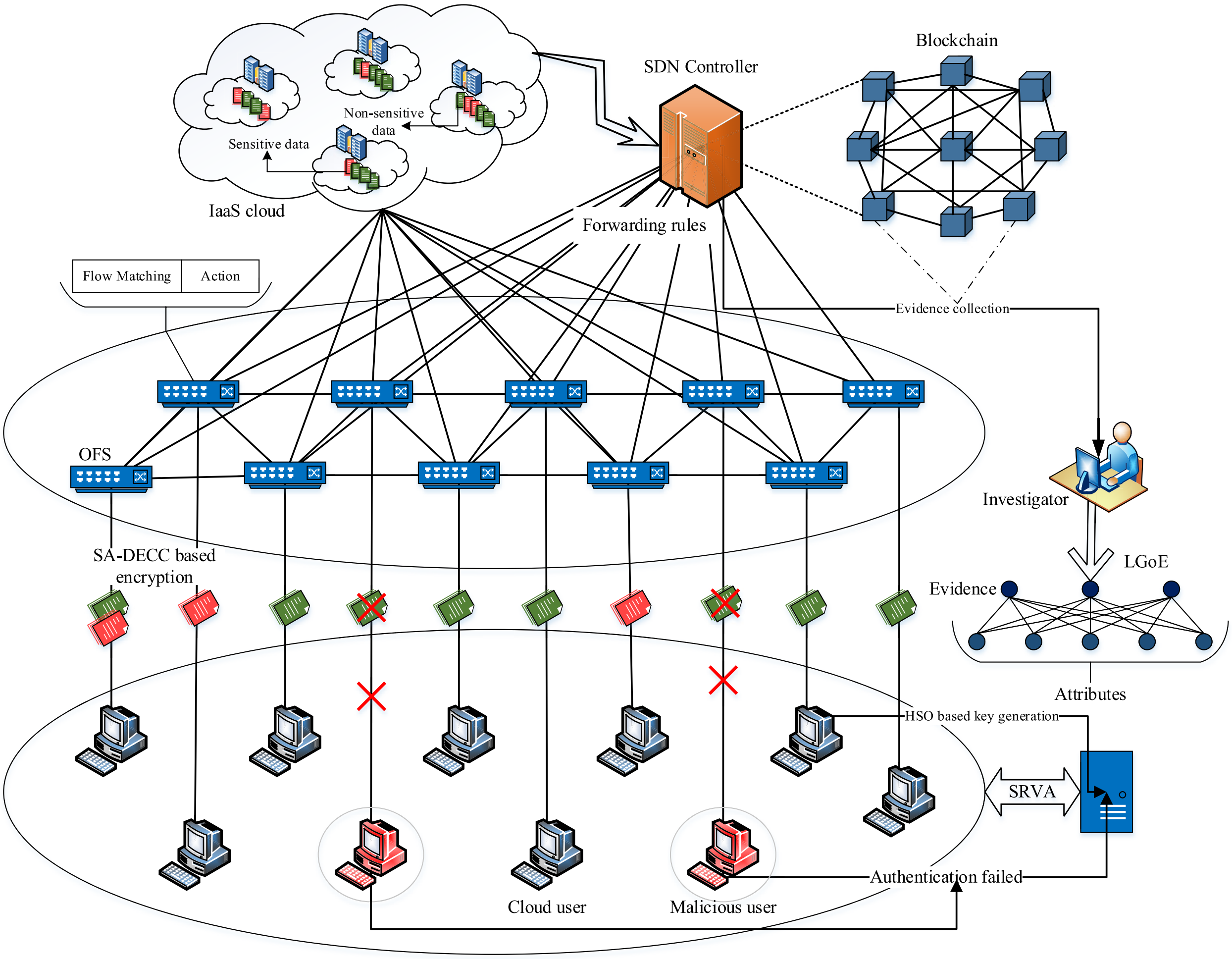


FIGURE 2. The proposed digital forensic DFeSB architecture.

1) KEY GENERATION BY HSO ALGORITHM

HSO is a recently developed meta-heuristic algorithm that follows the music improvisation process performed by musicians to obtain improved harmony [45], [46]. It has been applied in a variety of fields. In this research, the HSO algorithm is utilized for the cryptography key generation process. In general, the equation of ECC is given as,

$$y^2 = x^3 + ax + b \tag{1}$$

where (x, y) represents the point on the curve and a, b represents the values that define the curve.

On this curve, the base point 'P' is selected and the random number '$Pr(SK)$' is selected within the specified range. Then the public key is generated as follows,

$$Pu\,(SK) = Pr(SK) \times P \tag{2}$$

Here, we can see that the private key ($Pr(SK)$) is generated randomly, which can be easily cracked by attackers. In order to improve the key generation process, the HSO algorithm is utilized. In the HSO algorithm, all possible random numbers within the range of $[1, R]$ are initialized as harmony memory (HM) that contains harmony vectors where R represents the maximum limit for random numbers.

Then each harmony vector is evaluated based on fitness function $f(x)$ which is obtained by the run test as follows [39],

$$f\,(x) = a - {}^{\mu_a}\!/_{\sigma_a} \tag{3}$$

where a = number of runs, μ_a = mean, and σ_a = variance.

Here the run test is adapted for fitness evaluation in order to determine the randomness of the solution. As stated earlier, the strength of the secret key depends upon the random number selected. The random number with high randomness will improve the strength of the generated key. Thus each random number is evaluated based on randomness. The worst vector (x^{worst}) is determined based on fitness value.

Then the new solutions are generated based on the harmony memory consideration rate (HMCR) and pitch adjustment rate (PAR). For memory consideration step, a random number r_1 is selected within [0, 1]. If $r_1 < HMCR$, then the new harmony vector is produced as follows,

$$x_{ij}^{new} = x_{ij}, \quad x_{ij} \in \{x_{1j}, x_{2j}, .., x_{HMSj}\} \tag{4}$$

where *HMSj* represents harmony memory size. The variables, i.e., new solutions obtained by Eq. (4) is further examined by PAR based on random number r_2 which is selected within [0, 1]. New solutions based on PAR are generated as follows,

$$x_{ij}^{new} = x_{ij} \pm r_2.BW \tag{5}$$

Here BW is the bandwidth factor that controls the local search around the new vector. Then the generated new vectors are evaluated based on $f(x)$ and compared with x^{worst}. Here x^{worst} defines the solution with lower $f(x)$ in the previous iteration. If ($x^{new} < x^{worst}$), then HM is updated as follows,

$$x^{worst} = x^{new} \tag{6}$$

Over iteration, a vector with better $f(x)$ is selected by the HSO algorithm, and it is assigned to $Pr(SK)$. Determining the generated secret key is hard for attackers since the random number is chosen more optimally by the HSO algorithm.

2) AUTHENTICATION BY SRVA SCHEME

For all registered users, AS generates secret key and origin points. Origin points are (O_x, O_y) coordinates of a circle that is different for each user. At AS, for each user corresponding credentials $\{ID, PW, SC\}$ are stored. At each time of authentication, all credentials are verified. The secret code generated by AS is random for each user, which is difficult for an attacker to guess the code. A circle is defined by following the equation,

$$(A_x - O_x)^2 + (B_y - O_y)^2 = R^2 \tag{7}$$

By using origin points, each user generates SC that is composed of (A_x, B_y). The user selects a SC that satisfies the equation of the circle in order to complete authentication successfully. When a user needs to access the cloud, the user must submit all credentials along with timestamp (T_S).

In algorithm.1, the process of SVRA based authentication is explained. A user who provides valid credentials can complete authentication successfully. Considering SC along with T_S improve the security level of the SVRA scheme. Since the SC is varied with time, the attacker is not able to crack the SC. Even if the attacker cracks SC at a time, the attacker is not able to use that SC for the next authentication without knowing origin points.

C. SENSITIVE AWARE DATA ENCRYPTION

In the proposed forensic system, the users who have completed the authentication process successfully are allowed to access the cloud environment. In the cloud environment, users store their data in the form of ciphertext with the digital signature. Here sensitivity level of data is decided by users. For example, confidential data such as bank details, identity details are often known as sensitive data and other data such as funny videos, movies are non-sensitive data. As stated in the previous subsection, secret keys are generated with the HSO algorithm. By using generated strong secret key, data is converted into ciphertext in SA-DECC algorithm. In the SA-DECC algorithm, the ECC algorithm is combined with deep structure. Deep learning is a fast-forwarding method that is incorporated for the encryption and decryption process through multiple hidden layers [47]. In the DECC algorithm, data to be encrypted and $Pu(SK)$ are initialized in the input layer and the encryption process is taken place at hidden layers.

However, SA-DECC is sensitivity aware, and it performs the following processes for data encryption.

Algorithm 2, explains the overall process of SA-DECC algorithm using the strong secret key. The illustration of the proposed SA-DECC algorithm is depicted in Fig. 3.

Algorithm 1 Pseudocode for SRVA Based Authentication

```
Input: User credentials
Output: Authentication status
Begin
For $\forall U$            //Registration
   Register $ID, PW \rightarrow AS$
   $AS$ generates $SK$ using HSO
   $AS$ provides $SKs, OriginPoints \rightarrow U$
End for            //Registration is completed
If $U_i$ needs to access cloud   //Authentication
    Compute $SC$ using Eq. (7)
    $U_i$ submits $ID_i, PW_i, SC, T_S \rightarrow AS$
    $AS$ verifies credentials
    If (Credentials are true)
        $U_i = AUthorizeduser$
    Else
        $U_i = Unauthorizeduser$
    End if
  Else
    End process
  End if
  End
```

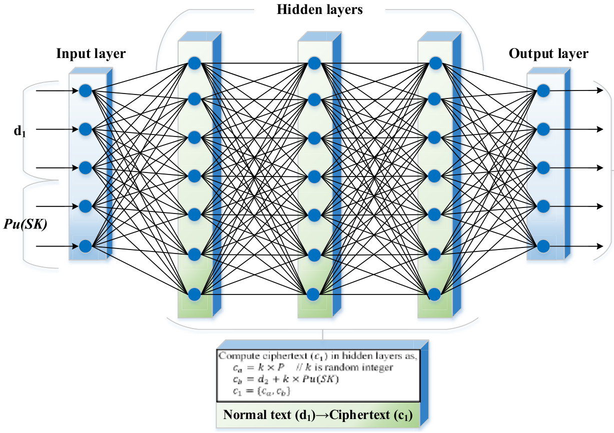

Hidden layers
Input layer
Output layer
d1
Pu(SK)
c1
Compute ciphertext (c_1) in hidden layers as,
$c_a = k \times P$ // k is random integer
$c_b = d_2 + k \times Pu(SK)$
$c_1 = \{c_a, c_b\}$
Normal text (d_1)→Ciphertext (c_1)

FIGURE 3. Proposed SA-DECC based encryption.

Likewise, when data has decrypted, the ciphertext initialized at the input layer and the original text is obtained at the

Algorithm 2 Pseudocode for SA-DECC

```
Input: Data and Public Key
Output: Ciphertext
1.  Initialize data ($d$) and public key ($Pu(SK)$)
2.  If ($d = Sensitive$)
3.    Divide $d \rightarrow d_1, d_2$
4.    For $d_1$
5.      Compute ciphertext 1($c_1$) as,
6.      $c_1 = d_1 \oplus d_2$
7.    End for
8.    For $d_2$
9.      Initialize $Pu(SK)$, $d_2$ at input layer
10.     Compute ciphertext 2 ($c_2$) in hidden layers as,
11.     $c_a = k \times P$  // $k$ is random integer
12.     $c_b = d_2 + k \times Pu(SK)$
13.     $c_2 = \{c_a, c_b\}$
14.   End for
15.   Obtain ciphertext ($c$) as,
16.   $c = \{c_1, c_2\}$
17. Else
18.   For $d$
19.     Do steps (8-13)
20.   End for
21. End if
21. End
```

output layer. The involvement of deep learning algorithm in encryption improves the security level of data. In order to preserve proof of ownership, data must be signed by the user before outsourcing to the cloud environment. By using the ECC algorithm, the digital signature is generated as follows,

At first, the hash value is generated for data to be signed as

$$HV = HASH(d) \tag{8}$$

Then the digital signature is generated as,

$$Sign = \frac{HV + \Pr(SK).k_2}{k_1} \tag{9}$$

where k_1 and k_2 are random numbers. At each time data modified or ownership is changed, the data must be signed by the current owner of the data.

D. RELIABLE EVIDENCE COLLECTION BY BLOCKCHAIN

In the case of cyber-crimes, digital evidence is substantial sources for investigation. The suspects can hide their data in various areas of the IaaS cloud system and can delete the evidence. The major challenging issue in the IaaS cloud system is that data processing is distributed on a large scale of computing resources. In addition, the cloud users have more control than investigators which makes evidence collection and preservation as a challenging issue. In order to defend against all these issues, the proposed digital forensic system uses SDN and blockchain technology to collect and preserve the forensic evidence from the cloud. The evidence is stored in blockchain under the control of the SDN controller. Some significant definitions in cloud forensics are,

- ***Chain of Custody (CoC):*** It can be described as the process of maintaining and documenting the sequential history of handling data as digital evidence. In digital forensics, evidence can be passed through different levels of hierarchy, i.e. from a first responder, investigators (one or more), and judge. During this lifetime, the evidence is handled by these temporary owners. Our proposed work maintains CoC since each action taken on evidence is stored in the blockchain.
- ***Proof of Ownership (PoO):*** In this work, PoO is described as proof of current ownership in digital evidence. Data can be controlled by many owners during their lifetime. Whenever ownership of data has changed then the data must be signed by the current owner to preserve PoO in the cloud environment. In the proposed system, PoO is preserved since the ownership change also stored as the history of data in the blockchain.
- ***Smart contracts:*** It is a computer program that runs to trace the history of data automatically. The smart contract is activated and executed when the necessary conditions are met. In this work, fuzzy rules are deployed to optimize smart contracts.
- ***Data provenance:*** It records the history of ownership and the process of the document throughout its lifecycle. In other words, provenance is defined as the sequence of records that show the actions made on the data. We preserve data provenance with the support of blockchain, i.e., in our work, each modification made on data is stored in the blockchain and traced by FSC.

In the blockchain, which is the distributed ledger, the evidence is stored with the hash value. For hash value generation, we propose the SHA-3 algorithm, which is better in terms of security level. In SHA-3, the hash value for each block is computed as follows,

$$Hash = sponge[g, pad, q](T, L) \tag{10}$$

Here hash value is generated for input, i.e., transaction (T) with padding function pad, permutation function g, rate q, and output length L. In the Merkle tree, the hash value is generated at each time using the 'sponge construction' process in SHA-3 as in Eq. (10) instead of using (SHA-256)^2. Adapting SHA-3 for hash computation brings many advantages over the existing method in time consumption and security level. Let consider, a user U_1 stores data d_1 at time t_1 in cloud. Then the block is created for d_1 and the hash value is generated by SHA-3. From the time of block created, each transaction, i.e., modification held on d_1 is traced by FSC deployed in the system. Each modification is stored as evidence in blockchain and distributed among the peers in the blockchain network. The log of evidence includes the user ID who made a transaction on the data, IP address, accessing time and other hardware details (virtual machine logs, deletion of file, etc.) of the evidence. For each modification held on data, the history of data is preserved as evidence in the blockchain. The history of data may include provenance records that define the modifications, ownership transfer and other actions taken on data stored in the cloud environment.

In algorithm.3, the process of evidence collection is explained. Here the evidence is collected and preserved in blockchain for each data stored in the cloud. In addition, FSC tracks and controls the accessibility of data stored by users in the cloud environment.

Algorithm 3 Pseudocode for Evidence Collection

```
Input: user data
Output: digital evidence
Begin
For all $U_i \in U$
  Create $FSC$ for users
End for
For each data
  $U_1$ stores $d_1$ in IaaS
  Create block for $d_1$
  Compute $Hash\ (d_1)$ using Eq. (10)
  Track $d_1$ and update evidence
End for
For each transaction on $d_1$
  Store source IP, timestamp, action made,
  transaction hash, VM server, etc.
  If ($Fuzzyrulesareviolated$)    //FSC
    Generate report
  Else
    Do not generate the report
  End if
End for
End
```

In this work, smart contracts are deployed to report the actions to a cloud server when it satisfies a fuzzy rule, which is also included as an evidence log in the blockchain. Many authorized users can access the data stored in the cloud environment. In this work, smart contracts are derived by fuzzy logic that works upon a sensitivity level of data. The smart contract is executed using fuzzy rules deployed in the system.

In Fig. 4, a pictorial representation of FSC is illustrated. The involvement of FSC traces all significant actions made on the data stored in the cloud server. Thus, all reliable evidence of data stored in the cloud server is collected, and the integrity of evidence is preserved in our proposed forensic architecture using blockchain technology.

In Table 1, the fuzzy rules deployed in FSC are depicted. Based on these rules, the report is generated and stored as an evidence log.

The past risk is defined as the data alteration made in previous access. If the past risk is low and the data is non-sensitive, then the access evidence log is ignored, and the report is not generated. Otherwise, the generated report is considered as significant evidence and stored in the blockchain.

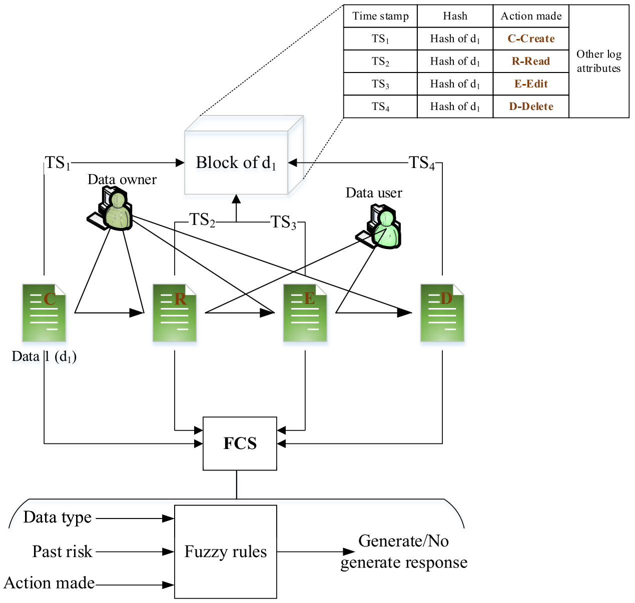


FIGURE 4. Proposed FSC.

TABLE 1. Fuzzy rules for FSC.

Data type	Past risk	Action performed	Fuzzy Value	Report generation
Non-sensitive	Low	Read	0-0.5	No
Sensitive			0-0.5	No
Non-sensitive	Low	Edit	0-0.5	No
Sensitive			0.51-0.1	Yes
Non-sensitive	Low	Delete	0.51-0.1	Yes
Sensitive			0.51-0.1	Yes
Non-sensitive	High	Read	0-0.5	No
Sensitive			0.51-0.1	Yes
Non-sensitive	High	Edit	0-0.5	No
Sensitive			0.51-0.1	Yes
Non-sensitive	High	Delete	0.51-0.1	Yes
Sensitive			0.51-0.1	Yes

E. CLOUD FORENSIC INVESTIGATION

When a cyber-crime is identified, then the authorized investigator (polices, lawyers) must analyze the digital evidence regarding that crime. Before the investigation, the investigator is also authenticated by AS. For example, if a suspect check-in a hotel then her/his details are stored in a hotel database. The suspect is expected to hack the database to remove her/his check-in histories, i.e., attempted to trash the digital evidence. In such a case, our proposed forensic architecture will support effectively since all evidence logs are maintained in the blockchain, which is a distributed ledger. In addition, she/he must pass the strong authentication before entering into the system. In the view of the investigator, the following steps are to be followed for evidence analysis.

1) EVIDENCE IDENTIFICATION

In a digital forensic investigation, the first step is to identify the potential evidence source which has reliable evidence. So the investigator must get appropriate permission from legal authorities.

TABLE 2. Sample evidence with attributes.

Evidence ID	Timestamp	Source IP	User uploaded	User accessed	Action	TxHash	BlockHash	Location	VM server	OFS
001	T_{s1}	212.95.136.xx	U_A	U_A	Upload	m-bits	n-bits	YYY	abcdi	01
002	T_{s2}	212.95.137.xx	U_A	U_B	Read	m-bits	n-bits	YYY	abcdi	02
003	T_{s3}	212.95.136.xy	U_A	U_X	Edit	m-bits	n-bits	yyy	Abcdims	03
004	T_{s4}	212.95.136.xx	U_A	U_X	Edit	m-bits	n-bits	yyy	abcdimjk	03
005	T_{s5}	212.95.136.xx	U_A	U_A	Update	m-bits	n-bits	YYY	abcdi	01
006	T_{s6}	212.95.136.xx	U_A	U_A	Update	m-bits	n-bits	YYY	abcdi	01
007	T_{s7}	212.95.136.xx	U_A	U_A	Delete	m-bits	n-bits	YYY	abcdi	01

2) EVIDENCE ACQUISITION

With approval from a legal authority, the investigator can collect all evidence logs from the blockchain. In this research work, the evidence log contains both user credentials and hardware-oriented evidence. In this stage, the investigator must follow judicial constraints without violating SLA agreements.

3) EVIDENCE ANALYSIS

Then the investigator analyzes entire evidence logs to generate a report regarding digital evidence. For better analysis, LGoE is proposed in this paper. LGoE is constructed upon the evidence with corresponding log attributes. For the same example, the suspect check-in a hotel, the check-in history, i.e., initial data is uploaded to cloud by hotel administration, i.e., authorized user. At this time, the evidence is created at blockchain with all log attributes (source IP, timestamp, action made, transaction hash, VM server, OFS ID, etc.)

Let consider, at t_2 administrator updated the check-in history of suspect. Then the next log is updated in a corresponding block with log attributes. Likewise, when the suspect attempts to hack this data or delete this data from the cloud, this event is also considered to be evidence and updated in the corresponding block. For LGoE construction, the investigator has to perform the following processes:

- Order the evidence sequentially based on the timestamp
- Initialize all evidence with its log attributes
- Construct LGoE based on evidence sequence and log attributes

The sample evidence set is illustrated in Table II with attributes. By using this data, LGoE can be built as in Fig. 5.

From, LGoE, the investigator can see that the evidence is edited (modified) by the suspect (U_X). But the location and IP addresses differ from the authorized user.

4) EVIDENCE REPORTING

In the evidence analysis stage, all evidence present in LGoE is validated through the digital signature, which is maintained along with data and hash value. In our proposed work, data must be signed before outsourcing to the cloud. Thus, when an attacker modifies this data, then the attacker must produce a digital signature.

The hash value of the current transaction is stored in blockchain for all evidence. The root value of the Merkle tree in a block must be matched with the hash value of data stored in the cloud. Based on these analyses, the investigator prepares a report to submit it as digital evidence to the court. Algorithm.4 explains the evidence collection process from acquisition until submission to the court.

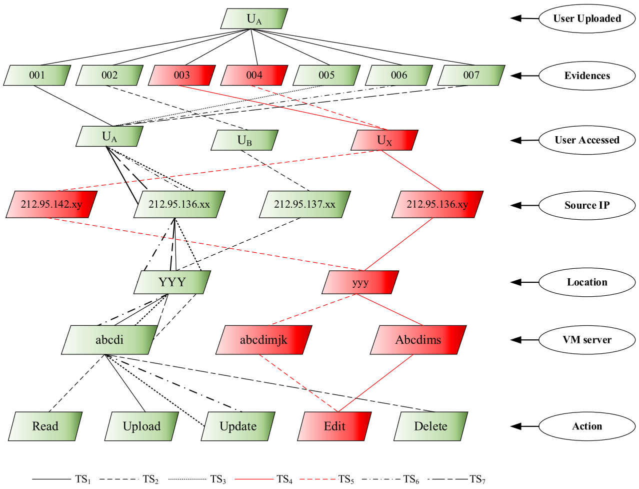


FIGURE 5. LGoE for evidence analysis.

Therefore, our proposed digital forensic architecture using SDN and blockchain technology supports reliable evidence collection from the cloud environment. The involvement of a strong authentication process prevents unauthorized users to access the cloud environment, whereas sensitivity aware encryption process strengthens the security level of data. Utilizing blockchain and SDN for evidence collection is an intellectual solution for distributed evidence preservation. Our proposed forensic architecture supports the entire investigation from evidence collection until evidence reporting to the court.

V. EXPERIMENTAL EVALUATION

In this section, we experimentally analyze proposed forensic architecture with prior research work according to performance metrics. In this section, we first introduce our simulation environment then compare our proposed work with the previous centralized log collection method.

A. SIMULATION SETUP

We configure our proposed forensic architecture in a combined simulation platform. We implemented the IaaS cloud

Algorithm 4 Pseudocode for Forensic Investigation

```
Input: Evidence
Output: LGoE
1.  Begin
2.  Authenticate investigator by SVRA scheme
3.  Identify evidence regarding case
4.  Collect evidence from blockchain as
      { Evidence ID, Timestamp, Source IP, Useruploaded,
        User accessed, Action, TxHash, BlockHash,
        Location, VM server, OFS ID }
5.  Plot LGoE using evidence attributes
6.  For all evidence
7.    Verify $\{BlockHash\&\&SourceIP\}$
8.    If $(Verification = True)$
9.        Verify the signature          //Evidence validation
10.         If $(Signatureisvalid)$
11.             Prepare Valid Evidence
12.         Else
13.             Prepare Invalid Evidence
14.         End if
15.     End if
16.   End for
17.   Prepare the digital evidence and submit it to the court
18.   End
```

environment in java platform using CloudSim. For data stored in the IaaS cloud, blockchain is created in Java as in [42]. For developing Java programs, NetBeans IDE is used.

All the experiments are simulated on Intel Core i7 CPU 2.80 GHz, 16 GB memory, and 128 GB SSD on Ubuntu OS. Further, the cloud and blockchain environment is integrated with the network simulator version.3 (ns-3) simulator, which is dedicated to SDN network simulation. The output obtained from the Java platform (in JAR format) is combined with ns-3 to obtain full-fledged simulation.

In Table 3, we provide the simulation tools used in our with their purpose. Entire work is supported by the Ubuntu operating system.

TABLE 3. Simulation tools we have used.

Tool	Purpose
NetBeans-8.2	Blockchain configuration using Java
NS-3.26	SDN simulation
CloudSim	IaaS Cloud Implementation

In Table 4, the significant simulation parameters considered to implement our forensic architecture is explained. Before getting into the analysis, we provide a practical use case of the proposed forensic system.

In Fig. 6(a), the simulation environment of proposed forensic architecture in ns-3.26 is shown. This screenshot illustrates the secret key generation and SDN simulation. In Fig. 6(b) and Fig. 6(c), the analysis of blockchain is illustrated. Miner is deployed for validating the blockchain and the Proof-of-Work concept is used. For every data stored by the user in the cloud environment, a corresponding block is created, and the hash values are stored.

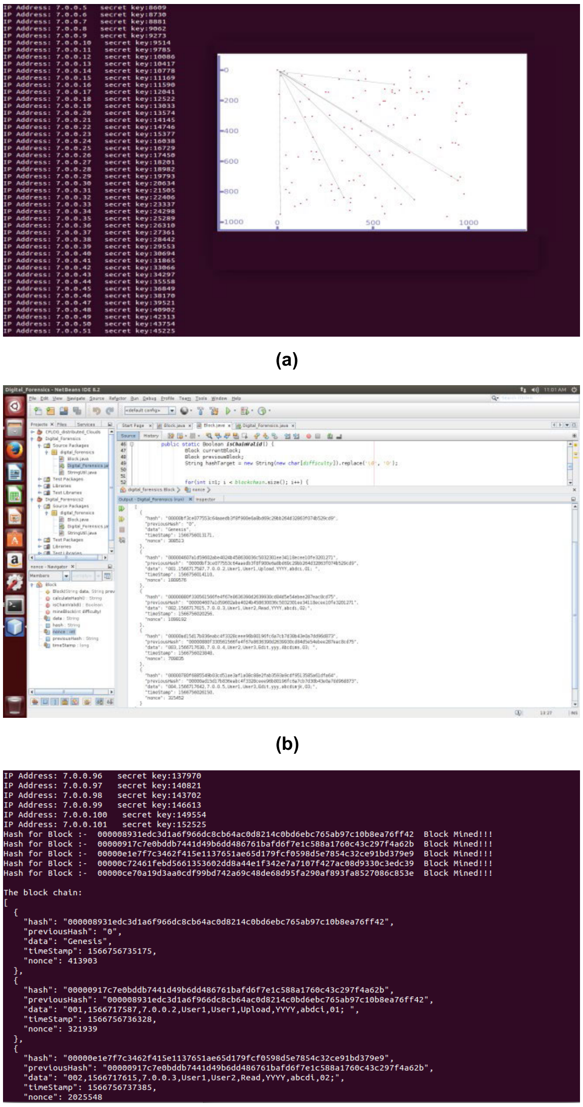
TABLE 4. Simulation settings.

Parameter		Value
Number of users		100
Number of OFSs		7
Number of controllers		1
Number of AS		1
Number of keys generated		100
HSO	HMS	100
	HMCR,PAR	0.995, 0.9
	Maximum iteration	100
DECC	Number of hidden layers	3
	Key size	256 bits
SHA-3	Block size	576 bits
	Word size	64 bits
	Number of rounds	24
	Customized Contract	FSC
	Maximum handles	2048
Cloud	Number of VMs	34
	Average random access memory	512 MB
	Average bandwidth	1000,000 MB
Simulation time		100s

1) USE CASE OF PROPOSED FORENSIC DFeSB ARCHITECTURE

IaaS is a highly-scalable cloud environment that can be utilized by any growing organization. Our proposed digital forensic architecture in the IaaS cloud environment can be applicable to many real-world applications. Here we analyze one use case of proposed work in crime detection applications. Let us consider some hotels which maintain their data (including guest register, finance detail, maintenance details, staff register, and surveillance data) in the IaaS cloud. As per our work, all data are encrypted based on the sensitivity level of data before outsourced the cloud. Besides, the admins of each hotel must be registered with AS. Evidence for all data stored in the cloud environment are collected by the SDN controller and maintained at blockchain. Further, each admin can trace its data via FSC.

The illustration of the proposed use case is depicted in Fig. 7. Consider a criminal stayed at hotel A for a couple of days. Then the details about the suspect can be found in the guest register of hotel A. Furthermore, the data collected from surveillance cameras also will have footage of the suspect in the hotel. This might be helpful for investigators to track the suspect as quickly as possible. Each modification made on guest register and surveillance data is collected as evidence in the blockchain. Without our efficient forensic architecture, the suspect can delete or modify the guest register and



```
IP Address: 7.0.0.5   secret key:8609
IP Address: 7.0.0.6   secret key:8730
IP Address: 7.0.0.7   secret key:8881
IP Address: 7.0.0.8   secret key:9062
IP Address: 7.0.0.9   secret key:9273
IP Address: 7.0.0.10   secret key:9514
IP Address: 7.0.0.11   secret key:9785
IP Address: 7.0.0.12   secret key:10086
IP Address: 7.0.0.13   secret key:10417
IP Address: 7.0.0.14   secret key:10778
IP Address: 7.0.0.15   secret key:11169
IP Address: 7.0.0.16   secret key:11590
IP Address: 7.0.0.17   secret key:12041
IP Address: 7.0.0.18   secret key:12522
IP Address: 7.0.0.19   secret key:13033
IP Address: 7.0.0.20   secret key:13574
IP Address: 7.0.0.21   secret key:14145
IP Address: 7.0.0.22   secret key:14746
IP Address: 7.0.0.23   secret key:15377
IP Address: 7.0.0.24   secret key:16038
IP Address: 7.0.0.25   secret key:16729
IP Address: 7.0.0.26   secret key:17450
IP Address: 7.0.0.27   secret key:18201
IP Address: 7.0.0.28   secret key:18982
IP Address: 7.0.0.29   secret key:19793
IP Address: 7.0.0.30   secret key:20634
IP Address: 7.0.0.31   secret key:21505
IP Address: 7.0.0.32   secret key:22406
IP Address: 7.0.0.33   secret key:23337
IP Address: 7.0.0.34   secret key:24298
IP Address: 7.0.0.35   secret key:25289
IP Address: 7.0.0.36   secret key:26310
IP Address: 7.0.0.37   secret key:27361
IP Address: 7.0.0.38   secret key:28442
IP Address: 7.0.0.39   secret key:29553
IP Address: 7.0.0.40   secret key:30694
IP Address: 7.0.0.41   secret key:31865
IP Address: 7.0.0.42   secret key:33066
IP Address: 7.0.0.43   secret key:34297
IP Address: 7.0.0.44   secret key:35558
IP Address: 7.0.0.45   secret key:36849
IP Address: 7.0.0.46   secret key:38170
IP Address: 7.0.0.47   secret key:39521
IP Address: 7.0.0.48   secret key:40902
IP Address: 7.0.0.49   secret key:42313
IP Address: 7.0.0.50   secret key:43754
IP Address: 7.0.0.51   secret key:45225
```

(a)

Digital_Forensics - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
<default config>
Projects
Files
Services
Start Page
Block.java
Block.java
Digital_Forensics.java
Source
History
Output - Digital_Forensics (run)
Inspector
11:01 AM
13:27
INS

```
public static Boolean isChainValid() {
        Block currentBlock;
        Block previousBlock;
        String hashTarget = new String(new char[difficulty]).replace('\0', '0');


        for(int i=1; i < blockchain.size(); i++) {
```

```
[
  {
    "hash": "[illegible]",
    "previousHash": "0",
    "data": "Genesis",
    "timeStamp": 1566756013171,
    "nonce": 308513
  },
  {
    "hash": "[illegible]",
    "previousHash": "[illegible]",
    "data": "001,1566717587,7.0.0.2,User1,User1,Upload,YYYY,abdci,01; ",
    "timeStamp": 1566756014110,
    "nonce": 1008576
  },
  {
    "hash": "[illegible]",
    "previousHash": "[illegible]",
    "data": "002,1566717615,7.0.0.3,User1,User2,Read,YYYY,abcdi,02;",
    "timeStamp": 1566756020256,
    "nonce": 1090192
  },
  {
    "hash": "[illegible]",
    "previousHash": "[illegible]",
    "data": "003,1566717630,7.0.0.4,User2,User3,Edit,yyy,abcdims,03; ",
    "timeStamp": 1566756023848,
    "nonce": 709835
  },
  {
    "hash": "[illegible]",
    "previousHash": "[illegible]",
    "data": "004,1566717642,7.0.0.5,User1,User3,Edit,yyy,abcdimjk,03;",
    "timeStamp": 1566756026150,
    "nonce": 315452
  }
```

(b)

```
IP Address: 7.0.0.96   secret key:137970
IP Address: 7.0.0.97   secret key:140821
IP Address: 7.0.0.98   secret key:143702
IP Address: 7.0.0.99   secret key:146613
IP Address: 7.0.0.100   secret key:149554
IP Address: 7.0.0.101   secret key:152525
Hash for Block :-  000008931edc3d1a6f966dc8cb64ac0d8214c0bd6ebc765ab97c10b8ea76ff42  Block Mined!!!
Hash for Block :-  00000917c7e0bddb7441d49b6dd486761bafd6f7e1c588a1760c43c297f4a62b  Block Mined!!!
Hash for Block :-  00000e1e7f7c3462f415e1137651ae65d179fcf0598d5e7854c32ce91bd379e9  Block Mined!!!
Hash for Block :-  00000c72461febd5661353602dd8a44e1f342e7a7107f427ac08d9330c3edc39  Block Mined!!!
Hash for Block :-  00000ce70a19d3aa0cdf99bd742a69c48de68d95fa290af893fa8527086c853e  Block Mined!!!

The block chain:
[
  {
    "hash": "000008931edc3d1a6f966dc8cb64ac0d8214c0bd6ebc765ab97c10b8ea76ff42",
    "previousHash": "0",
    "data": "Genesis",
    "timeStamp": 1566756735175,
    "nonce": 413903
  },
  {
    "hash": "00000917c7e0bddb7441d49b6dd486761bafd6f7e1c588a1760c43c297f4a62b",
    "previousHash": "000008931edc3d1a6f966dc8cb64ac0d8214c0bd6ebc765ab97c10b8ea76ff42",
    "data": "001,1566717587,7.0.0.2,User1,User1,Upload,YYYY,abdci,01; ",
    "timeStamp": 1566756736328,
    "nonce": 321939
  },
  {
    "hash": "00000e1e7f7c3462f415e1137651ae65d179fcf0598d5e7854c32ce91bd379e9",
    "previousHash": "00000917c7e0bddb7441d49b6dd486761bafd6f7e1c588a1760c43c297f4a62b",
    "data": "002,1566717615,7.0.0.3,User1,User2,Read,YYYY,abcdi,02;",
    "timeStamp": 1566756737385,
    "nonce": 2025548
```

(c)

FIGURE 6. **(a) Simulation environment of proposed forensic architecture created in ns-3.26. (b) Blockchain created in java. (c) Blockchain analysis.**

surveillance data stored in the cloud. However, with our proposed forensic architecture, all evidence is stored in the blockchain, which is a distributed ledger. Also, we collect the VM logs as evidence in the blockchain. Thus even if the suspect modifies the data in the cloud, the investigator can collect the evidence from the blockchain. Plotting LGoE for the collected evidence log will show if there is any variations are presented among the evidence. From the evidence collected from blockchain, the investigator can transfer the digital evidence with CoC to court.

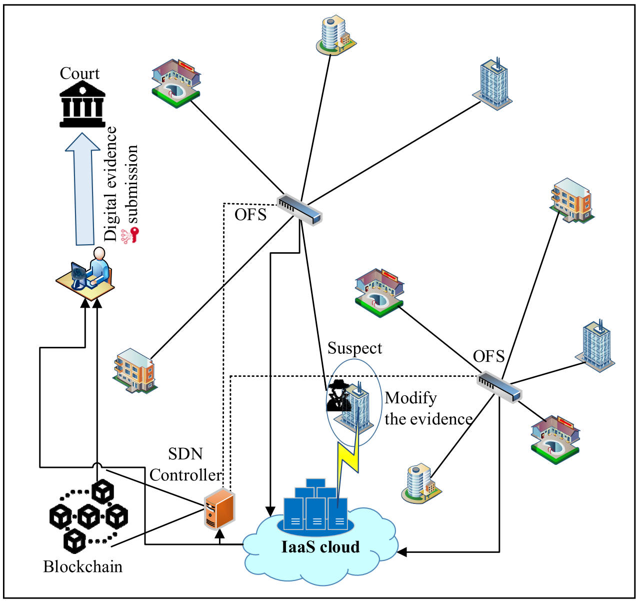


FIGURE 7. **Use case digital forensic in crime investigation.**

B. COMPARATIVE ANALYSIS

In this subsection, we compare our proposed forensic architecture with the existing CFLOG [40] method, which is intended to collect digital evidence securely. The significant difference between proposed forensic architecture and CFLOG is that CFLOG collects and stores evidence in a centralized manner under CSP. This introduces many problems, as stated in section III. To overwhelm these challenges, we proposed a novel forensic architecture by using SDN and blockchain technology, which collects and preserves digital evidence securely.

1) ANALYSIS OF RESPONSE TIME

Response time is the time taken by the users to receive the response for the requested data. This metric is validated based on the number of users involved in the forensic system. In other words, response time is defined as the time taken by the forensic system to respond to the users with the required evidence or data.

In Fig. 8, we compare the response time of the proposed SDN-blockchain based forensic system with the existing CFLOG system, which is centralized architecture. In both works, response time is gradually increased with the increase in the number of users since the number of requests from users is increased with the increase in the number of users. However, even with an increased number of users proposed digital forensic system responds quickly for user requests. The involvement of SDN technology increases scalability, i.e. supports the huge number of users simultaneously. Thus any cloud user can be connected immediately with the cloud server and can retrieve requested data immediately. Likewise, the investigator can collect evidence from the blockchain without a time delay from the SDN controller.

Thus proposed forensic architecture minimizes response time. In CFLOG, both data storage and evidence collection are performed by CSP in a centralized manner, which increases response time in the presence of a huge number of users. In the presence of 100 users, the CFLOG system

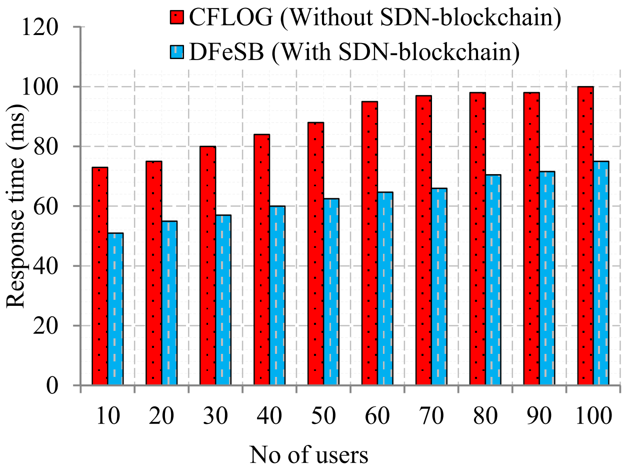


FIGURE 8. Comparison of response time.

requires 100ms to respond whereas the proposed digital forensic system takes 75ms for the same number of users. Thus proposed digital forensic system achieves 25% better results than the CFLOG system.

2) ANALYSIS OF EVIDENCE INSERTION TIME

Evidence insertion time is defined as the time taken to insert (or) create the digital evidence for data stored in the cloud server. In our work, it can be described as the time taken by the SDN controller to create evidence for the data stored in CSP.

In Fig. 9, evidence insertion time is evaluated with respect to number users. When the number of users is increased, then the amount of data to be stored and the number of evidence to be created is also increased. Thus in both works, evidence insertion time is increased with an increase in the number of users. In the CFLOG method, all evidence is collected and stored in a centralized manner under the control of CSP. Thus centralized evidence collection process increases evidence insertion time. Also, in our work, we preserve the history of data, i.e., each modification held on data is considered as evidence and inserted in the blockchain. However, evidence creation and preservation processes are being performed by the SDN controller without the involvement of CSP. Thus evidence insertion in blockchain minimizes time consumption compared to previous work.

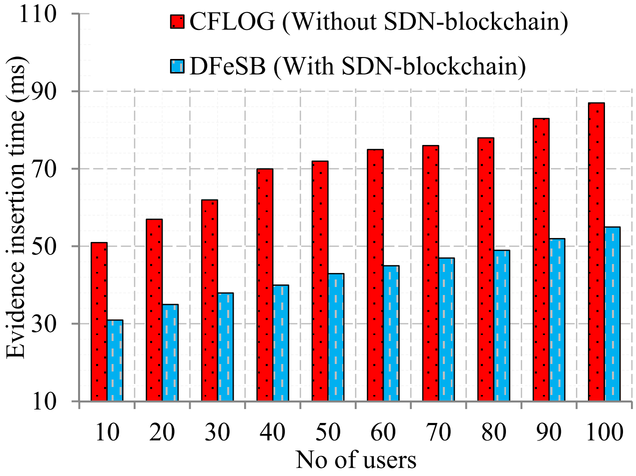


FIGURE 9. Comparison of evidence insertion time.

3) ANALYSIS OF EVIDENCE VERIFICATION TIME

Evidence verification time is defined as the time taken by an investigator to collect and verify the digital evidence from the blockchain. During the investigation, the investigator must collect and verify the digital evidence. For an efficient forensic system, evidence verification time must be as low as possible.

In Fig. 10, evidence verification time required in the CFLOG method and the proposed forensic method is compared. The proposed digital forensic system attains minimum evidence verification time. In the CFLOG method, the investigator must access CSP for evidence collection and verification is performed in the traditional method. However, in the proposed work, the investigator aggregates all evidence from the controller instead of CSP. Also, evidence verification is performed by constructing LGoE for better analysis. Besides, we proposed SHA-3 based hash computation to preserve the integrity of evidence without an increase in time consumption. Therefore, we achieve evidence integrity within minimum time consumption during evidence verification.

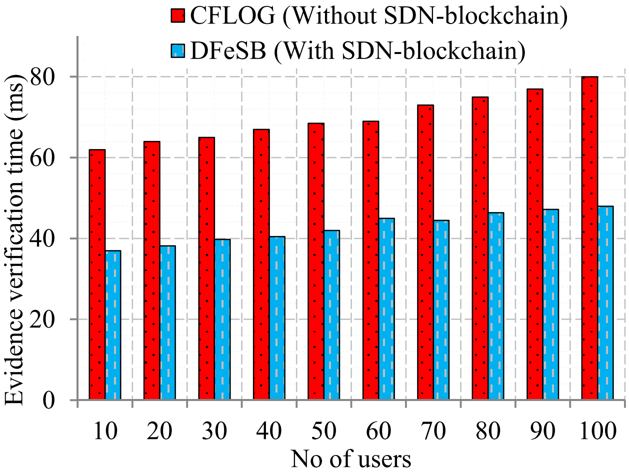


FIGURE 10. Comparison of evidence verification time.

In the presence of 10 users, 62ms is required in CFLOG to collect in verify the digital evidence while only 37ms is required in proposed digital forensics DFeSB, i.e., our system minimizes nearly 50% of verification time.

4) ANALYSIS OF COMPUTATIONAL OVERHEAD

Communication overhead measures the amount of bandwidth spent to perform a specific task (data upload, read, edit, evidence creation, verification) in the forensic system.

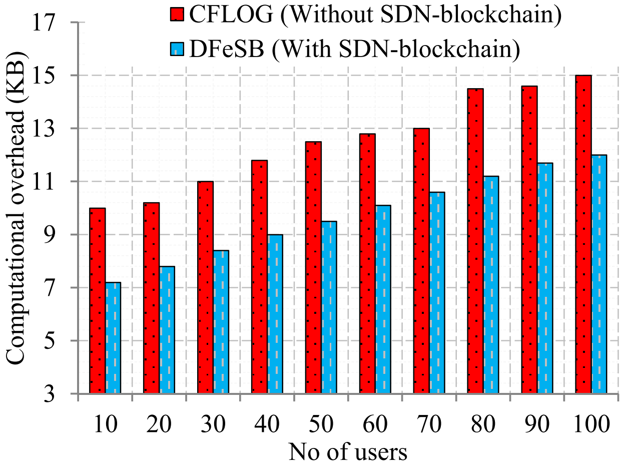


FIGURE 11. Comparison in computational overhead.

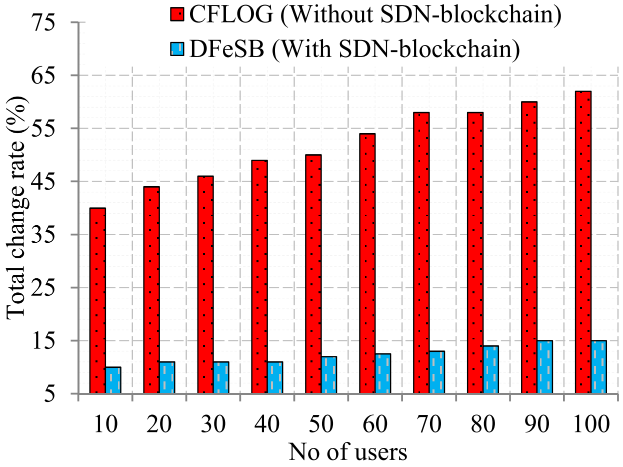


FIGURE 12. Comparison of the total change rate.

In Fig. 11, the comparison of computational overhead is made with respect to the number of users. Here in computational overhead is increased with an increase in the number of users since the amount of data to be processed is also increased. In the absence of blockchain technology, the computational overhead is increased due to centralized system management. In CFLOG, all data processing and evidence processing are carried out in CSP, which increases the overhead.

However, in the proposed forensic system, evidence processing (collection, hash computation, preservation) is held on the SDN controller, which minimizes overall computational overhead. In addition, the involvement of SDN technology improves scalability without an increase in overhead. Thus for ten cloud users, proposed digital forensic architecture introduces 7KB of overhead, whereas the CFLOG system requires 10KB of overhead.

5) ANALYSIS OF THE TOTAL CHANGE RATE

The total change rate is defined as the ratio between the number of modified evidence and the number of total evidence maintained in the forensic system.

In Fig. 12, we compare the total change rate of our proposed work with the previous CFLOG system. The total change rate increases whenever a malicious user modifies the evidence in order to demolish the digital evidence. For an efficient forensic system, the collected evidence must be reliable, and the integrity of evidence should be ensured. In the proposed forensic system, all evidence and data from unauthorized users are denied since it allows only authorized users. Furthermore, we preserve the integrity of evidence by using blockchain technology based on the SHA-3 algorithm.

Our results show that 10% of the evidence is modified in the proposed forensic system. However, this modification is also recorded as evidence in blockchain since we assure integrity, CoC, and PoO for evidence. In CFLOG method, nearly 60% of evidence are modified due to (i) centralized architecture since the CSP can be malicious, (ii) single node failure (attacker only needs to crack CSP), (iii) no integrity is preserved, and (iv) involvement of unauthorized user access.

We overwhelm all problems with the support of SDN and blockchain technology, which minimize the total change rate of the system.

In Table 5, average results obtained by the CFLOG method and the proposed forensic system are compared with respect to performance metrics. Here we can see that the proposed digital forensic DFeSB architecture has an improvement in each metric.

TABLE 5. Comparative analysis.

Performance metric	CFLOG	Proposed digital forensic DFeSB
Response time (ms)	88.5	**63**
Evidence insertion time (ms)	71	**43.5**
Evidence verification time (ms)	70	**42.8**
Computational overhead (KB)	12.5	**9.7**
Total change rate (%)	52	**12.3**

6) EFFICIENCY OF SA-DECC WITH HSO ALGORITHM

In blockchain technology, the ECC algorithm is conventionally used for digital signature. However, it involves many problems in key generation, encryption, and decryption. In order to improvise the traditional ECC algorithm, we proposed the SA-DECC algorithm with the HSO algorithm for key generation. Thus we analyze our proposed SA-DECC algorithm with the HSO algorithm against the Paillier encryption algorithm proposed for blockchain technology [16].

From Fig. 13 to Fig. 15, the analysis of the proposed SA-DECC algorithm is presented. In [16], the Paillier encryption algorithm is proposed for secure blockchain architecture. However, the Paillier encryption scheme increases key generation time, encryption time, and decryption time rapidly. The involvement of large homomorphic computations in the Paillier scheme increases time consumption.

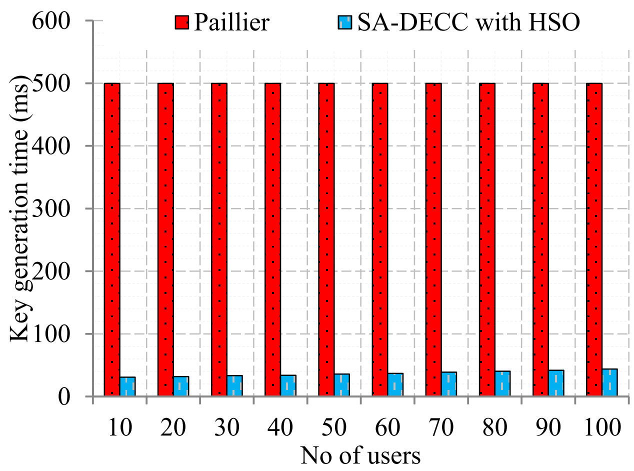


FIGURE 13. Analysis of key generation time.

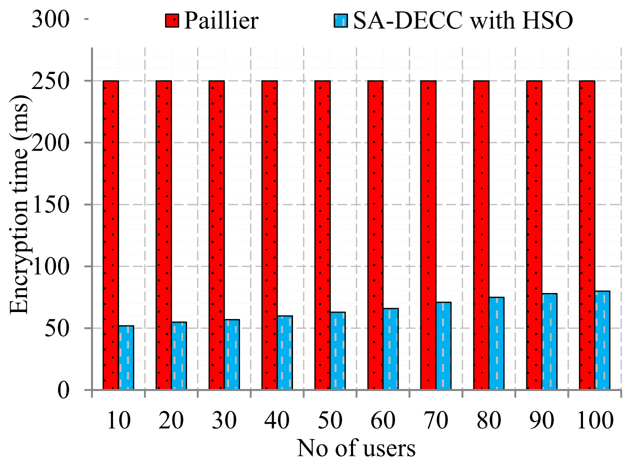


FIGURE 14. Analysis of encryption time.

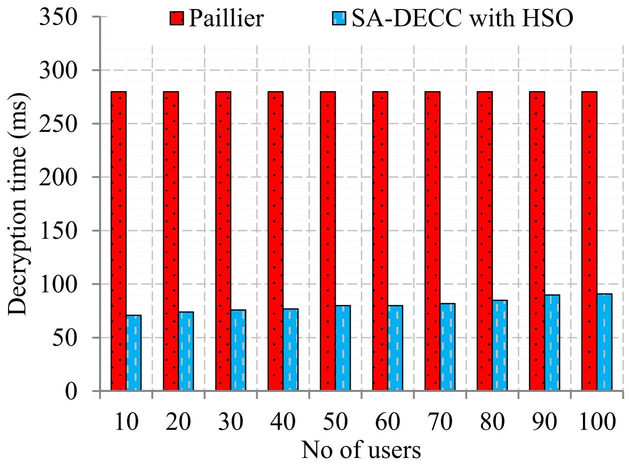


FIGURE 15. Analysis of decryption time.

However, in cloud environment data encryption is significant since there will be numerous users involved. In the Paillier algorithm, 500ms is averagely taken for key generation. Similarly, encryption and decryption also require considerable time constraints, which is not suitable for the cloud environment.

However, in the proposed SA-DECC algorithm key generation time is reduced with the support of the HSO algorithm, which has minimum convergence time. Likewise, the deep architecture of the SA-DECC algorithm minimizes the time required to perform encryption and decryption.

Therefore the proposed SA-ECC algorithm is better than the conventional algorithm to improve security level without an increase in time consumption.

7) EFFECTIVENESS OF SHA-3 ALGORITHM

In blockchain technology, $(\text{SHA-256})^2$ is used for hash generation. In our proposed forensic system, we have used the SHA-3 algorithm for hash computation in order to improve hash computation time and security level [48].

In Fig. 16, the hash computation time of proposed SHA-3 with previous $(\text{SHA-256})^2$ algorithm. This analysis shows that SHA-3 minimizes hash computation time to 16ms for 100 users without loss in the security level. In general, SHA-3 is better than SHA-256 against many security attacks, such as length extension attacks. Thus the involvement of SHA-3 based Merkle tree construction improves the security level without an increase in time consumption.

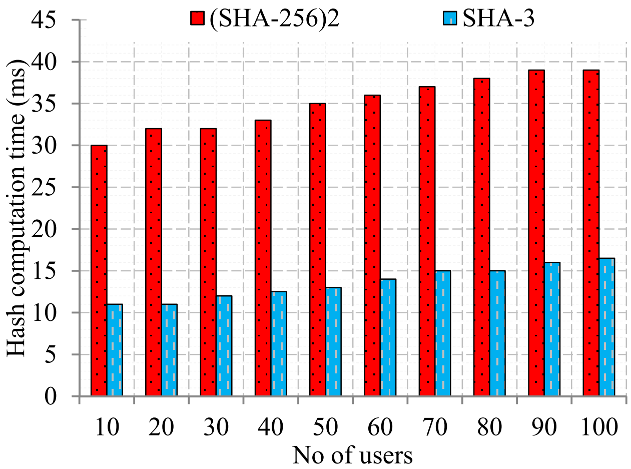


FIGURE 16. Analysis of hash computation time.

Overall analysis shows that the proposed digital forensic DFeSB architecture performs better than the existing CFLOG system. The involvement of blockchain and SDN technology improves system performance and scalability.

VI. CONCLUSION

In this paper, a novel digital forensic architecture is proposed with SDN and blockchain technology to collect and preserve reliable evidence from the IaaS cloud environment. All cloud users are authenticated by AS with secure verification scheme known as the SRVA scheme. For data security, the SA-DECC algorithm is proposed. Before this, optimal keys are generated by the HSO algorithm. For each data stored in the cloud, a block is created at the controller. In each block, SHA-3 based Merkle tree construction ensures the integrity of evidence. All evidence is collected and CoC, PoO is preserved by blockchain technology. In order to trace data activities, FCS is deployed in the system. Finally, evidence analysis is made simplified by using LGoE based analysis. Overall, the forensic system is analyzed in a combined simulation environment that includes java and ns-3.26. Experimental evaluations show that proposed forensic architecture achieves better results than the existing centralized forensic system. In the future, we intend to introduce network forensic (within SDN) along with cloud forensics in order to strengthen the digital forensic system.

APPENDIX

Table 6 demonstrates the list of notations used to proposed forensic architecture.

TABLE 6. Basic notations for system model.

Notations	Definition
AS	Authentication Server
BW	Bandwidth
CoC	Chain of Custody
CSP	Cloud Service Provider
d	Data
FCS	Fuzzy based Smart Contracts
HM	Harmony Memory
$HMCR$	Harmony Memory Consideration Rate
HMS	Harmony Memory Size
HSO	Harmony Search Optimization
ID	User Identification
$LGoE$	Logical Graph of Evidence
$OFSs$	OpenFlow Switches
PoO	Proof of Ownership
PW	Password
PAR	Pitch Adjustment Rate
Pr	Private Key
Pu	Public Key
$SA - DECC$	Sensitivity Aware Deep Elliptic Curve Cryptography
SC	Secret Code
$SDN - C$	SDN Controller
SK	Secret Key
$SRVA$	Secure Ring Verification based Authentication
T	Transaction
TS	Time-Stamp
U	Cloud Users
VM	Virtual Machine

ACKNOWLEDGMENT

This research was supervised in the Rasht Branch, Islamic Azad University, Rasht, Iran.

REFERENCES

[1] S. K. A. Manoj and D. L. Bhaskari, "Cloud forensics-a framework for investigating cyber attacks in cloud environment," *Procedia Comput. Sci.*, vol. 85, pp. 149–154, 2016. doi: 10.1016/j.procs.2016.05.202.

[2] S. Almulla, Y. Iraqi, and A. Jones, "A state-of-the-art review of cloud forensics," *J. Digit. Forensics Secur. Law*, vol. 9, no. 4, p. 2, 2014.

[3] S. Zawoad and R. Hasan, "Trustworthy digital forensics in the cloud," *Computer*, vol. 49, no. 3, pp. 78–81, Mar. 2016.

[4] R. Battistoni, R. D. Pietro, and F. Lombardi, "CURE—Towards enforcing a reliable timeline for cloud forensics: Model, architecture, and experiments," *Comput. Commun.*, vols. 91–92, pp. 29–43, Oct. 2016.

[5] 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.

[6] M. K. Pandya, S. Homayoun, and A. Dehghantanha, "Forensics investigation of openflow-based SDN platforms," in *Advances in Information Security*, vol. 70. Cham, Switzerland: Springer, 2018, pp. 281–296.

[7] T. Chin and K. Xiong, "A forensic methodology for software-defined network switches," in *Proc. IFIP Int. Conf. Digit. Forensics*, 2017, pp. 97–110.

[8] Y. Xie, D. Feng, X. Liao, and L. Qin, "Efficient monitoring and forensic analysis via accurate network-attached provenance collection with minimal storage overhead," *Digit. Invest.*, vol. 26, pp. 19–28, Sep. 2018.

[9] B. Zhao, P. Fan, and M. Ni, "Mchain: A blockchain-based VM measurements secure storage approach in IaaS cloud with enhanced integrity and controllability," *IEEE Access*, vol. 6, pp. 43758–43769, 2018.

[10] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the Internet of Things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

[11] W. Yang, E. Aghasian, S. Garg, D. Herbert, L. Disiuta, and B. Kang, "A survey on blockchain-based Internet service architecture: Requirements, challenges, trends, and future," *IEEE Access*, vol. 7, pp. 75845–75872, 2019.

[12] A. Nayak and K. Dutta, "Blockchain: The perfect data protection tool," in *Proc. Int. Conf. Intell. Comput. Control (I2C2)*, Jun. 2017, pp. 1–3.

[13] L. Zhou, L. Wang, Y. Sun, and P. Lv, "Beekeeper: A blockchain-based iot system with secure storage and homomorphic computation," *IEEE Access*, vol. 6, pp. 43472–43488, 2018.

[14] Q. Xia, E. B. Sifah, A. Smahi, S. Amofa, and X. Zhang, "BBDS: Blockchain-based data sharing for electronic medical records in cloud environments," *Information*, vol. 8, no. 2, p. 44, 2017.

[15] X. Liang, S. Shetty, D. Tosh, C. Kamhoua, K. Kwiat, and L. Njilla, "Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability," in *Proc. 17th IEEE/ACM Int. Symp. Cluster Cloud Grid Comput. (CCGRID)*, 2017, pp. 468–477.

[16] B.-K. Zheng, L.-H. Zhu, M. Shen, F. Gao, C. Zhang, Y.-D. Li, and J. Yang, "Scalable and privacy-preserving data sharing based on blockchain," *J. Comput. Sci. Technol.*, vol. 33, no. 3, pp. 557–567, May 2018.

[17] M. Cebe, E. Erdin, K. Akkaya, H. Aksu, and S. Uluagac, "Block4Forensic: An integrated lightweight blockchain framework for forensics applications of connected vehicles," *IEEE Commun. Mag.*, vol. 56, no. 10, pp. 50–57, Oct. 2018.

[18] Z. A. El Houda, A. S. Hafid, and L. Khoukhi, "Cochain-SC: An intra- and inter-domain Ddos mitigation scheme based on blockchain using SDN and smart contract," *IEEE Access*, vol. 7, pp. 98893–98907, 2019.

[19] K. Salah, N. Nizamuddin, R. Jayaraman, and M. Omar, "Blockchain-based soybean traceability in agricultural supply chain," *IEEE Access*, vol. 7, pp. 73295–73305, 2019.

[20] J. Al-Jaroodi and N. Mohamed, "Blockchain in industries: A survey," *IEEE Access*, vol. 7, pp. 36500–36515, 2019.

[21] B. Shahzad and J. Crowcroft, "Trustworthy electronic voting using adjusted blockchain technology," *IEEE Access*, vol. 7, pp. 24477–24488, 2019.

[22] Z. Zhou, B. Wang, M. Dong, and K. Ota, "Secure and efficient vehicle-to-grid energy trading in Cyber physical systems: Integration of blockchain and edge computing," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published.

[23] P. Dubey, V. Tiwari, S. Chawla, and V. Chauhan, "Authentication framework for cloud machine deletion," in *Information and Communication Technology for Sustainable Development* (Lecture Notes in Networks and Systems), vol. 10. Singapore: Springer, 2018, pp. 199–206.

[24] J. Ricci, I. Baggili, and F. Breitinger, "Blockchain-based distributed cloud storage digital forensics: Where's the Beef?" *IEEE Security Privacy*, vol. 17, no. 1, pp. 34–42, Jan./Feb. 2019.

[25] Q. Lin, H. Yan, Z. Huang, W. Chen, J. Shen, and Y. Tang, "An ID-based linearly homomorphic signature scheme and its application in blockchain," *IEEE Access*, vol. 6, pp. 20632–20640, 2018.

[26] S. Khan, A. Gani, A. W. A. Wahab, S. Iqbal, A. Abdelaziz, O. A. Mahdi, A. I. Abdallaahmed, M. Shiraz, Y. R. B. Al-Mayouf, Z. Khan, K. Ko, M. K. Khan, and V. Chang, "Towards an applicability of current network forensics for cloud networks: A SWOT analysis," *IEEE Access*, vol. 4, pp. 9800–9820, 2016.

[27] D. Quick and K. R. Choo, "Iot device forensics and data reduction," *IEEE Access*, vol. 6, pp. 47566–47574, 2018.

[28] X. Fu, R. Yang, X. Du, B. Luo, and M. Guizani, "Timing channel in IaaS: How to identify and investigate," *IEEE Access*, vol. 7, pp. 1–11, 2019.

[29] S. Haque and T. Atkison, "A forensic enabled data provenance model for public cloud," *J. Digit. Forensics Secur. Law*, vol. 13, no. 3, 2018.

[30] P. M. Trenwith and H. S. Venter, "FReadyPass: A digital forensic ready passport to control access to data across jurisdictional boundaries," *Austral. J. Forensic Sci.*, vol. 51, no. 5, pp. 583–595, Sep. 2019.

[31] M. E. Alex and R. Kishore, "Forensics framework for cloud computing," *Comput. Elect. Eng.*, vol. 60, pp. 193–205, May 2017.

[32] S. Zawoad, A. K. Dutta, and R. Hasan, "Towards building forensics enabled cloud through secure logging-as-a-service," *IEEE Trans. Depend. Sec. Comput.*, vol. 13, no. 2, pp. 148–162, Mar./Apr. 2016.

[33] S. Alqahtany, N. Clarke, S. Furnell, and C. Reich, "A forensic acquisition and analysis system for IaaS: Architectural model and experiment," in *Proc. 11th Int. Conf. Availability Rel. Secur. (ARES)*, Aug./Sep. 2016, pp. 345–354.

[34] M. N. A. Khan and S. Ullah, "A log aggregation forensic analysis framework for cloud computing environments," *Comput. Fraud Secur.*, no. 7, pp. 11–16, Jul. 2017.

[35] M. Irfan, H. Abbas, Y. Sun, A. Sajid, and M. Pasha, "A framework for cloud forensics evidence collection and analysis using security information and event management," *Secur. Commun. Netw.*, vol. 9, no. 16, pp. 3790–3807, Nov. 2016.

[36] D. Spiekermann, J. Keller, and T. Eggendorfer, "Network forensic investigation in OpenFlow networks with ForCon," *Digit. Invest.*, vol. 20, pp. S66–S74, Mar. 2017.

[37] P. Santra, P. Roy, D. Hazra, and P. Mahata, "Fuzzy data mining-based framework for forensic analysis and evidence generation in cloud environment,' in *Advances in Intelligent Systems and Computing*, vol. 696. Singapore: Springer, 2018, pp. 119–129.

[38] L. Pasquale, S. Hanvey, M. Mcgloin, and B. Nuseibeh, "Adaptive evidence collection in the cloud using attack scenarios," *Comput. Secur.*, vol. 59, pp. 236–254, Jun. 2016.

[39] Q. I. Xia, 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.

[40] A. Pichan, M. Lazarescu, and S. T. Soh, "Towards a practical cloud forensics logging framework," *J. Inf. Secur. Appl.*, vol. 42, pp. 18–28, Oct. 2018.

[41] P. K. Sharma, M.-Y. Chen, and J. H. Park, "A software defined fog node based distributed blockchain cloud architecture for IoT," *IEEE Access*, vol. 6, pp. 115–124, 2018.

[42] Y. Zhang, X. Lin, and C. Xu, "Blockchain-based secure data provenance for cloud storage," in *Proc. Int. Conf. Inf. Commun. Secur.* Cham, Switzerland: Springer, 2018, pp. 3–19.

[43] J. Li, J. Wu, and L. Chen, "Block-secure: Blockchain based scheme for secure P2P cloud storage," *Inf. Sci.*, vol. 465, pp. 219–231.

[44] P. Vadhera and B. Lall, "Review paper on secure hashing algorithm and its variants," *Int. J. Sci. Res.*, vol. 3, no. 6, pp. 478–482, 2014.

[45] J. Yi, X. Li, C.-H. Chu, and L. Gao, "Parallel chaotic local search enhanced harmony search algorithm for engineering design optimization," *J. Intell. Manuf.*, vol. 30, no. 1, pp. 405–428, Jan. 2019.

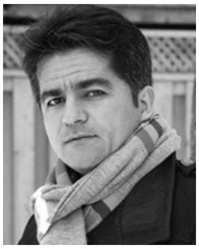
[46] T. Zhang and Z. W. Geem, "Review of harmony search with respect to algorithm structure," *Swarm Evol. Comput.*, vol. 48, pp. 31–43, Aug. 2019.

[47] S. Kalsi, H. Kaur, and V. Chang, "DNA cryptography and deep learning using genetic algorithm with NW algorithm for Key generation," *J. Med. Syst.*, vol. 42, no. 1, p. 17, Jan. 2018.

[48] R. K. Dahal, J. Bhatta, and T. N. Dhamala, "Performance analysis of SHA-2 and SHA-3 finalists," *Int. J. Cryptogr. Inf. Secur.*, vol. 3, no. 3, pp. 1–10, 2013.

MEHRAN POURVAHAB received the B.Sc. degree in software engineering from Islamic Azad University, Bafgh Branch, Iran, in 2001, and the M.Sc. degree in information technology engineering-networking from Guilan University, Rasht, Iran, in 2013. He is currently pursuing the Ph.D. degree in software system engineering with Islamic Azad University, Rasht Branch, Rasht, Iran.

In 2001, he founded the first Internet Service Provider (ISP) in Langrud, Iran. Since 2001, he has been a System/Network Administrator with MehranNet (ISP), more than 3000 subscribers. From September 2006 to November 2018, he was the Manager of Information and Communication Technology with the Azad University of Langarud Branch. Since 2011, he has been a MikroTik Certified Trainer and a Consultant. Since 2016, he has been a Faculty Member with the Department of Computer Engineering, Islamic Azad University, Langarud Branch, Iran. Since August 2018, he has also been the Head of the Security and Network Commission at the Iranian ICT Guild Organization (IIG) of Guilan Province. His main research interests include software-defined networking (SDN), cloud forensics, network security, and blockchain technology. He is a member of the IEEE Computer Society and serves as a Reviewer of the IEEE Access Journal.



GHOLAMHOSSEIN EKBATANIFARD He received the B.Sc. degree in software engineering from Islamic Azad University, Lahijan Branch, Iran, in 2001, and the M.Sc. degree in computer network security and the Ph.D. degree in software systems from the Ferdowsi University of Mashhad, Iran, in 2004 and 2013, respectively. In 2005, he joined the Islamic Azad University. He has been an Assistant Professor with the Azad University of Lahijan Branch, Iran, since 2013. From February 2012 to August 2012, he was a Visiting Researcher with the Delft University of Technology, The Netherlands. From November 2013 to April 2017, he was the Director of the Information and Communication Technology Center, Islamic Azad University, Lahijan Branch. His research interests include design and performance evaluation of communication protocols for wireless networks, network security, and cryptocurrency technologies.