Sensors and Materials, Vol. 35, No. 6 (2023) 2081–2100
MYU Tokyo

S & M 3308

Applying Blockchain Technology and Facial Recognition to Unmanned Stores

Pi-Yun Chen, Yu-Cheng Cheng, Neng-Sheng Pai,* and Yi-Hsuan Chiang

Department of Electrical Engineering, National Chin-Yi University of Technology,
Taichung City 41170, Taiwan (ROC)

(Received December 27, 2022; accepted June 8, 2023)

Keywords: blockchain, sidechain, scalability, image recognition, FaceNet, encryption, decryption

In this study, we utilize blockchain to design an information transmission system for unmanned stores. It is aimed to create a network for exchanging information that does not rely on a third party, a so-called decentralized system. To achieve it, a peer-to-peer (P2P) network replaces the master–slave architecture and uses asymmetric encryption to identify the system users. To authenticate the identities of the stores' customers, we use the face recognition network FaceNet developed by Google. FaceNet has several advantages that make it suitable for performing identity authentication in an unmanned store, including its high accuracy and convenience. A database of face images is required to perform facial recognition, but to address privacy concerns, every pixel in all images in the database is encrypted using the Rivest-Sharmir-Adleman (RSA) algorithm. The system proposed in this paper has three identity endpoints: the host end, the recognition end, and the client end. Messages are transmitted through a P2P network, and a directed acyclic graph is used to achieve message broadcasting while avoiding infinite loops when sending and receiving messages. A sidechain is used to change the structure and consensus mechanism of a traditional blockchain so that they can apply to more scenarios, thereby increasing scalability. The simulation results are displayed via the user interface.

1. Introduction

A blockchain is, in essence, a model based on the integration of cryptography and network topology. It can also be referred to as a distributed ledger that can reach a consensus. The main goal is to remove third-party dependence when performing electronic payments. Blockchain technology imposes no restrictions on the choice of network communication protocols. Still, it adds some rules and concepts to traditional data transmission methods to achieve decentralization, ensuring that data is anonymous, secure, and cannot be tampered with.(1)

Blockchain development is currently at a stage where countries and enterprises are still exploring its applications, and many think it will play a leading role in emerging industries. Some even believe that blockchains will cause a huge disruption in revolutionizing current

*Corresponding author: e-mail: pai@ncut.edu.tw

https://doi.org/10.18494/SAM4289

industries. For instance, there are considerable opportunities for application in the fields of finance,[2] the Internet of Things,[3] supply chain management,[4] and digital assets. However, despite it being a popular topic, some also question whether blockchains are just a fad. For one, the blockchain architecture is very different from the existing architecture and rules that have already been operating for many years. For example, centralized banks have absolute dominance in the financial field, and the technology that banks currently use is mature and widely adopted. Therefore, decentralized finance (DeFi), which is based on blockchain technology, is considered a tool that creates hype and not a fundamental financial institution.

In view of this, many researchers and enterprises have begun to explore whether it is possible to reduce cost and increase efficiency by changing the infrastructure. For instance, there is research on whether Ethereum, which needs significant time and computation power for its proof of work (PoW) mechanism, can be improved considerably by converting its consensus to proof of stake. The hope is to increase the transaction throughput to more than twice that of bitcoin and provide important ideas for the subsequent improvement of the blockchains.[5] Owing to its comprehensive development environment, Ethereum is one of the developers' most common development platforms. Regarding blockchain architecture, we believe that the traditional method of PoW is functional but needs improvements. Sidechains[6] can reduce the number of nodes that perform calculations to reduce resource consumption and improve transaction efficiency. They are also suitable for balancing centralization and decentralization, allowing validators to perform PoW while ordinary users can still acquire symmetrical information, thereby expanding blockchain applications. For example, in this study, we used sidechains as an information transmission system between store owners and customers in unmanned stores. In this paper, the application is "unmanned stores," and there are three endpoints: host end, recognition end, and customer end. This semi-centralized model gives key members reasonable authority and responsibilities to speed up information processing.

2. Related Work

2.1 Blockchains

The origin of the blockchain comes from an article published by Satoshi Nakamoto in 2008 titled "Bitcoin: A Peer-to-Peer Electronic Cash System".[7] This paper is referred to as the Bitcoin White Paper. In it, Nakamoto states that businesses on the Internet rely almost entirely on third-party financial institutions to complete information communication or data storage. This mode of transmission is simple and convenient, yet the problem lies in the power and burden held by third-party institutions. If the third party's database fails to function normally due to an accident or hacking, the rights and interests of countless users will be affected. Therefore, Nakamoto hopes to build a fair electronic currency system that does not rely on third-party organizations. The whitepaper also established the core concepts of blockchain technology, including decentralization, incorruptibility, consensus, anonymity, and encryption.

2.2 Scalability

Scalability generally refers to the design indicators of the computing and processing capabilities of the software. In the context of blockchains, it refers to indicators such as transaction throughput and confirmation latency.(8) Some problems of blockchains include a time delay when confirming transactions and transmitting messages and asynchronous network information. These problems can result in insufficient transaction throughput and the blockchain taking too long for block transactions and consensus. This phenomenon is also referred to as the impossible triangle of the blockchain,(9) also known as the blockchain trilemma.(10) The three factors are decentralization, security, and scalability. If one wants to improve the transaction processing speed of the blockchain, that is, to improve scalability, one must reduce decentralization or security, but reducing security in transactions may cause the overall system to collapse. Therefore, there is always a trade-off between centralization and scalability.

2.3 Sidechains

Sidechains change the consensus mechanism from being fair to being authority-based. In short, the authority to verify blocks is handed over to a small number of people. Sidechains are used in this article because this study applies to unmanned stores with a centralized model, reducing decentralization in exchange for scalability.(8)

2.4 Encryption

Encryption refers to the specific process of converting plaintext to ciphertext. The ciphertext must be difficult to solve and have a corresponding relationship with the plaintext. In this study, encryption will be applied in two places: the images in the face database and the user password, block content, and block verification for the blockchain.

3. Methodology

3.1 Blockchain architecture

In this study, we utilized the blockchain to design an information transmission system for unmanned stores in which sidechain technology is used to improve the scalability problems of traditional blockchains under centralized architectures. Figure 1 shows how information is transmitted in each endpoint. The figure shows that when customer A enters the store, the camera recognition endpoint performs facial recognition on the customer and transmits the information to the host end. The host stores but does not broadcast the message. Similarly, after the customer selects a product, the recognition end sends a message to the host. Finally, after customer A leaves the store, the host assembles the information into a complete block and sends

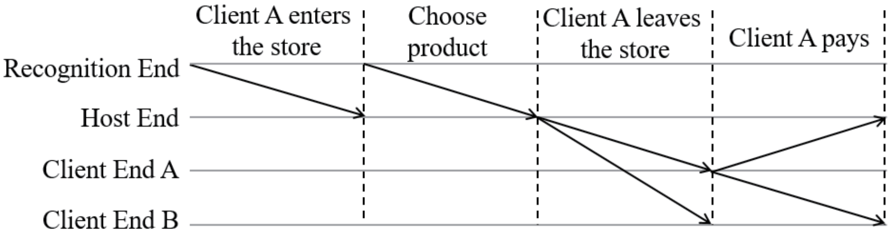


Fig. 1. Information transmission of endpoints.

it to all endpoints except for the recognition end. This represents that the transaction has gone through for customer A and other endpoints can store or verify this information.

Customer A needs to pay for the product after receiving the message. In this study, each person will have an initial fixed amount to help the system perform the transmission of payment messages. The payment message of Customer A is also sent to all endpoints on the chain except the recognition end so that proof of payment is recorded. It can be seen from the above process that this system can also be used for electronic payments. Unlike the existing invoice system, transaction details are recorded by someone other than the store owner and the customer, but these details are encrypted.

Figure 2 shows the two-chain architecture of this study. In Fig. 1, the main actions need to be completed by a centralized organization, and the sender and receiver of the messages are fixed. In this paper, tasks such as verifying transaction details and packaging blocks are separated and executed by the host, allowing the hosts (A, B, C) to form groups on the sidechains, as shown in the above figure. The purpose of groups is to return the verified results after all group members have completed the verification transaction. If the transaction reaches 2/3 approval, it represents a verified and correct transaction. Confirmed as legal, the transaction is then packaged into blocks and sent to the main chain. The main chain is for storing blocks and consists of all members (A, B, C, D, E) on the chain except the recognition endpoint. This architecture makes it possible for blocks to be verified twice by everyone and ensures that transactions are immutable. The following is an introduction to the three endpoint types in the architecture.

(1) Host end: The host end is the endpoint maintained by the unmanned store. It receives and assembles each node’s messages and sends transaction details and blocks. Figure 3 is the operation flow chart of a single host endpoint. The channel, which is required for the operation of the program, is designed to open the socket of the network module so that it can exchange messages with others. Once the socket is open, it starts to receive messages. If there is no message, it waits; if there is, it executes the corresponding action according to the received message identification character.

When information about the transaction is obtained, the transaction is put into the transaction pool. At this point, if the transaction volume of complete transactions (including entry, commodity name, and departure) is greater than or equal to the threshold, the transactions are verified, ensuring the details of each transaction are correct and complete. Since all hosts are involved in the verification, feedback is received after verification is completed. All hosts need

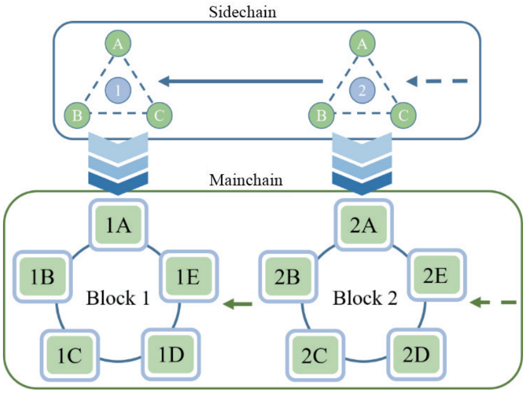


Fig. 2. (Color online) Two-chain architecture.

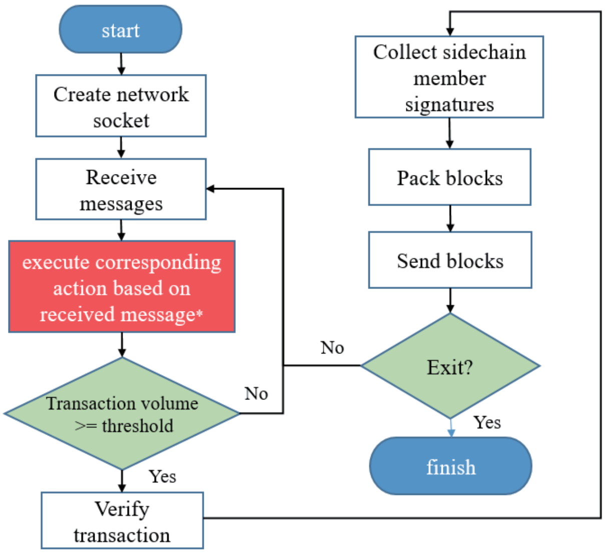

start
Create network socket
Receive messages
execute corresponding action based on received message*
Transaction volume >= threshold
No
Yes
Verify transaction
Collect sidechain member signatures
Pack blocks
Send blocks
Exit?
No
Yes
finish

Fig. 3. (Color online) Operation flow chart of the host endpoint.

to send back messages after verification to carry out the task of the sidechain. Finally, the blocks are packaged with the PoW mechanism and sent to other hosts and clients for storage.

(2) Recognition endpoint: The recognition endpoint is a communication port specially used to transmit the recognition results of customers and goods by using cameras in unmanned stores. It can be seen from the propagation diagram in Fig. 1 that this endpoint only sends messages to the host endpoint. This is because the results obtained by the recognition endpoint are exclusive controllable data and belong to the centralized organization, so no communication is necessary with the clients. Its functional components are a customer-facing button and a face recognition system. The architecture is shown in Fig. 4. Before each function is executed, preprocessing establishes network channels, loads faces, identifies the weights used by the FaceNet network,

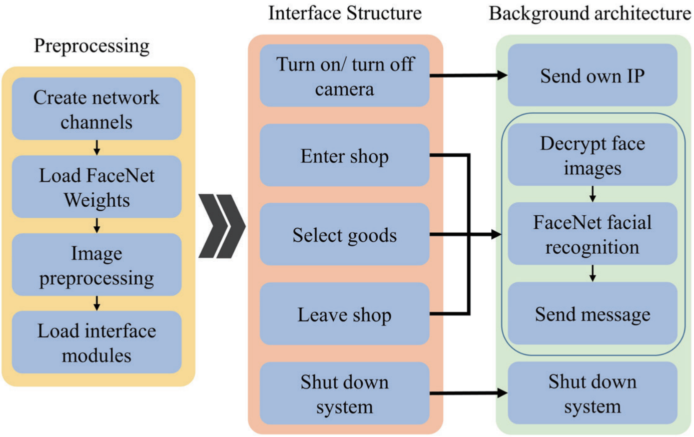


Fig. 4. (Color online) Recognition endpoint architecture.

adjusts parameters such as the image size, and loads the program modules required for the interface. The interface is designed using the PyQt5 framework. The framework rules are followed to load and arrange the required subsequent interfaces and to set the position and display the content of the interface.

The interface structure refers to the buttons displayed and usable on the user interface (UI). After pressing the button, a program in the background is triggered. For instance, turning the camera on and off indicates the status of the recognition endpoint. The program enters the chain by sending its own IP to the host. If the camera is turned off, it sends a command to delete the IP. To identify the customers' faces when they are entering the store, leaving the store, or selecting products, the recognition endpoint decrypts the face images in the database in order to identify the customers' information. The endpoint can inform the host who the client is when sending the transaction message. Finally, when the system is shut down, it closes the UI and FaceNet and sends a message to the host to delete its IP.

(3) Client endpoint: The client endpoint receives transaction details and blocks and executes payments sent by the host. The client-side architecture is shown in Fig. 5. A network channel needs to be established during preprocessing. The interface module is loaded, and the message-receiving mode is always turned on. After launching the system, the interface first requires the user to log in with a private key. The system sends the client IP to the host during the login for message transmission. After the host obtains the client's IP, it provides other hosts and clients feedback. The IP allows the client to communicate with others. The purpose is to verify the blockchain's correctness and uniqueness, which is only possible when all endpoints communicate with each other.

If this is the first time the customer has logged into the system, the registration button is shown, and a picture of the customer's face is required for subsequent facial recognition. After the customer uploads a picture of his or her face, the system uses the Rivest-Sharmir-Adleman (RSA) algorithm to encrypt every pixel in the picture. The encrypted data is then sent to the host, and the system generates a new private key, public key, and address for the user for subsequent identity authentication attempts. On the interface, the user can see overall

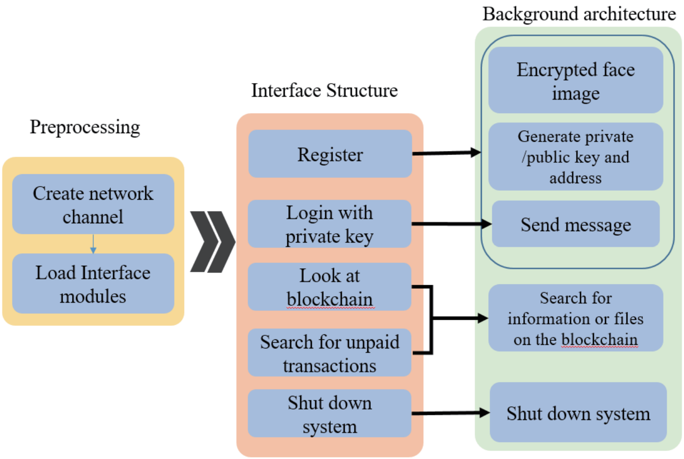


Fig. 5. (Color online) Client endpoint architecture.

information about the blockchain as well as relevant details. When the user presses the button, the system obtains data from temporary memory and system files and displays it on the interface.

(4) Network topology: In this study, changes in the network topology are also required to meet the needs of the architecture. Figure 6 is a diagram of an example network topology of the overall system. The figure assumes that the host (H) has five endpoints labeled from 01 to 05. Each host is equipped with a recognition terminal and an unequal number of customers (Cus). For instance, H01 has three customers, and H03 has one customer. The recognition terminal in Fig. 6 sends a message to the host terminal when the customer enters the door, selects the product, and goes out. Still, the recognition terminal does not receive any information. It only acts as the output terminal once recognition information is identified. When all hosts complete the decision and send the block, each host sends the same block to its clients, and the client sends the payment message to the host. This is a two-way transmission.

Figure 6 shows the flow of network messages in this system. First, all the hosts get a number according to the order in which they entered the chain. The hosts are numbered in the figure from 01 to 05, which indicates the main flow of information between terminals, as shown in Fig. 7(a). When H02 wants to send a block to other hosts for verification and storage, H02 broadcasts the message to other hosts using peer-to-peer (P2P) (denoted using a green dotted line in the figure). However, only H03 receives the message. In the figure, the red arrow points from H02 to H03. H03 broadcasts the message using P2P, and the same procedures repeat until the next member to receive the message is the same as the member who generated the message. In the figure, H01 ends the broadcast when it attempts to send the message to H02 because H02 is the member who first generated the message. This prevents the same message from spreading endlessly on the blockchain. Figure 7(b) shows the content of the transmission block. The first two digits of the message contain the "Previous" number, and the next two digits are the "Generate" number, followed by the message content.

(5) Block structure: The block structure in this article is shown in Fig. 8. Unlike traditional blocks, the transactions used in our system are not unspent transaction output (UTXO), but a

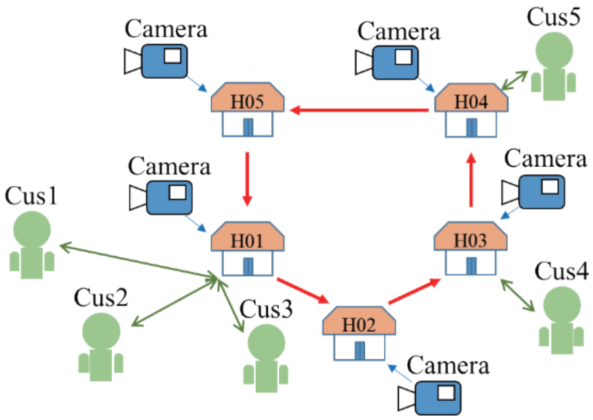
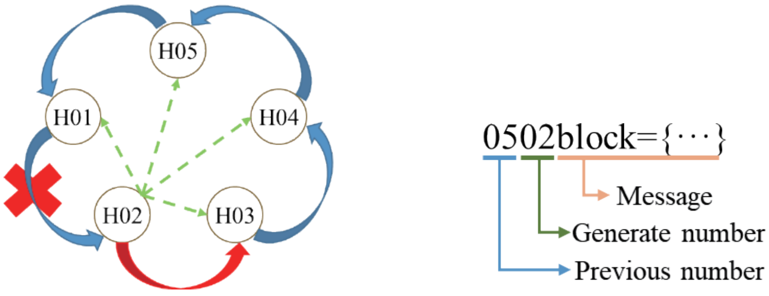


Fig. 6. (Color online) Example network topology of the overall system.



(a) (b)

Fig. 7. (Color online) Message flow topology for hosts.

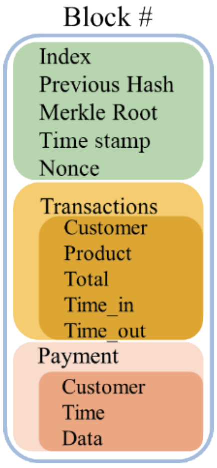


Fig. 8. (Color online) Block structure.

simple accounting method to record customer addresses (Customer), the product (Product), the total cost (Total), the time that the customer entered the store (Time in), and the time that the customer left the store (Time out). In addition, the blocks also record payment information (Payment), which includes the customer's address, payment time (Time), and payment amount (Data).

In our study, it is not the average user that initiates transactions, performs verification, and packages blocks. Therefore, it is unnecessary to use a difficulty target to increase the computational difficulty when packing blocks with PoW. Thus, the version problem and difficulty target of different nodes is ignored in the block header (green square). In addition, to facilitate the calculation of the number of blocks and the serial number, an index value (Index) field has been added to the structure, making it easier to find the block serial number.

3.2 Encryption and decryption algorithm

In this study, we used the RSA encryption algorithm to generate public and private keys and to encrypt face images. Below, we introduce the RSA encryption algorithm and the process and evaluation method of image encryption.

(1) RSA encryption algorithm: The RSA encryption algorithm is an asymmetric encryption algorithm jointly published by three scholars, Rivest, Shamir, and Adleman, in 1978.(11) The core of the algorithm is factoring. The mathematical formula of the RSA encryption algorithm is shown in the following.(12)

Step 1. Choose any two prime numbers and multiply them, as shown in Eq. (1), where N_m is the product of the two primes P_{n1} and P_{n2}. P_{n1} cannot be equal to P_{n2}.

$$N_m = P_{n1} \times P_{n2}, P_{n1} \neq P_{n2} \tag{1}$$

Step 2. Calculate Euler's totient function, where $\varphi(N_m)$ is a positive integer smaller than N_m. The term $\varphi(N_m)$ represents the number of integers that is smaller than or equal to N_m and co-prime with N_m.

$$\varphi(N_m) = (P_{n1} - 1) \times (P_{n2} - 1) \tag{2}$$

Step 3. Choose a random positive integer e that satisfies the following conditions: e needs to be between 1 and $\varphi(N_m)$, and be co-prime with $\varphi(N_m)$. In other words, e and $\varphi(N_m)$ have a greatest common denominator of 1.

$$\gcd\left(e, \varphi(N_m)\right) = 1, 1 < e < \varphi(N_m) \tag{3}$$

Step 4. Calculate an integer d that satisfies Eq. (4). The product of e and d is co-prime with $\varphi(N_m)$.

$$e \cdot d \bmod \varphi(N_m) \equiv 1 \tag{4}$$

Step 5. The mathematical formula for encryption is shown in Eq. (5). First, exponentiate (e) the plaintext M and divide it with a large number N_m. Then, take the remainder to obtain the ciphertext C.

$$C = M^e \left(\bmod N_m\right) \tag{5}$$

Step 6. Finally, the mathematical formula for decryption is shown in Eq. (6). The plaintext M can be deduced by dividing the ciphertext C to the power of the private key d by N_m and taking the remainder.

$$\varphi(N_m) = (P_{n1} - 1) \times (P_{n2} - 1) \tag{6}$$

Following these steps, a pair of keys can be obtained, where (e, N_m) is the public key, and d is the private key. The public key is the information sent by the sender of the message to the receiver. After the receiver obtains the encrypted message, it can use Eq. (8) to decrypt the message. However, a deceptively simple mathematical concept is behind the RSA algorithm; in practical applications, N_m is at least 1024 bits long. Considering the computing power of a normal computer, it is almost impossible for someone to decipher the prime factor of the number. Therefore, the security of the algorithm mainly comes from the bit length of the password. The longer it is, the less likely it is to be cracked. Therefore, the RSA encryption algorithm is suitable for identity verification.[12]

(2) Database encryption/decryption: In this study, we performed encryption and decryption on the face images in the database required for image recognition to protect the clients' privacy and the security of the database. Therefore, the host does not decrypt the images until the face recognition needs to be performed. During this process, the complete photo is not stored, and instead is directly used for calculation. This way, the host endpoint cannot obtain the complete image of the user's face.

Figure 9 shows how the RSA algorithm functions in the image encryption/decryption architecture. First, the user encrypts his or her own photos with the RSA encryption algorithm, as shown in Eqs. (1) to (5). After the calculation is completed, the range is limited to within 255 to obtain the encrypted image, as shown in Fig. 9. Subsequently, the encrypted image data (data not bounded by the 255 range) and the public key (e, N_m) are sent to the host. The host uses the image as the identity file of the face database. Then, when face recognition is required, the host decrypts the image using Eq. (6), and the decryption image in Fig. 9. can be obtained.

(3) *SSIM*: To evaluate whether the decrypted image is distorted from the original image, we used structural similarity (*SSIM*)[13] as the standard to measure the level of image distortion. Compared with the objective peak signal-to-noise ratio, *SSIM* is symmetrical and has clear upper and lower bounds, and thus the results yielded from *SSIM* are more in line with the results produced from human vision. The following shows the calculation formula of *SSIM*.

$$SSIM\left(x, y\right) = l\left(x, y\right)^{\alpha} c\left(x, y\right)^{\beta} s\left(x, y\right)^{\gamma} \tag{7}$$

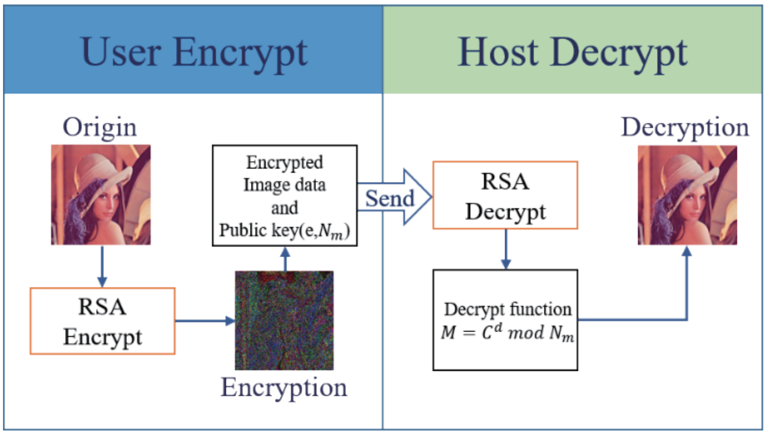

User Encrypt
Host Decrypt
Origin
Encrypted Image data and Public key(e,N_m)
Send
RSA Decrypt
Decryption
RSA Encrypt
Encryption
Decrypt function $M = C^d \bmod N_m$

Fig. 9. (Color online) Image encrypt/decrypt architecture.

$$l(x,y) = \frac{2\mu_x\mu_y + e_1}{\mu_x^2 + \mu_y^2 + e_1} \tag{8}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + e_2}{\sigma_x^2 + \sigma_y^2 + e_2} \tag{9}$$

$$s(x,y) = \frac{\sigma_{xy} + e_3}{\sigma_x\sigma_y + e_3} \tag{10}$$

In Eq. (7), two images are provided: $l(x, y)$ compares the luminance of the two images, $c(x, y)$ compares the contrast, and $s(x, y)$ compares the structure. SSIM is defined by the three indicators. The term α, β, and γ are important parameters that adjust l, c, and s, respectively, and all three must be greater than 0. The functions for calculating the three indicators are Eqs. (8) to (10), where μ_x and μ_y are the means of x and y, σ_x and σ_y are the standard deviations, σ_{xy} is the covariance, and e_1, e_2, and e_3 are constants of l, c, and s, respectively. The constants prevent the denominator from being equal to 0.[14]

Covariance can be negative, so the range of *SSIM* can be from −1 to 1. When the two images are identical, the result of *SSIM* is 1; if *SSIM* < 0.95, the difference between the two images is so large that there is no point in comparing the two images. SSIM is not suitable for evaluating the quality of the entire image. This is because the features of images can easily be unevenly distributed, and *SSIM* results are affected if image distortion occurs. Therefore, in this study, we used a sliding window with a step size of 1. The average of all *SSIM*s is used as the result that is most in line with the judgment of human vision.

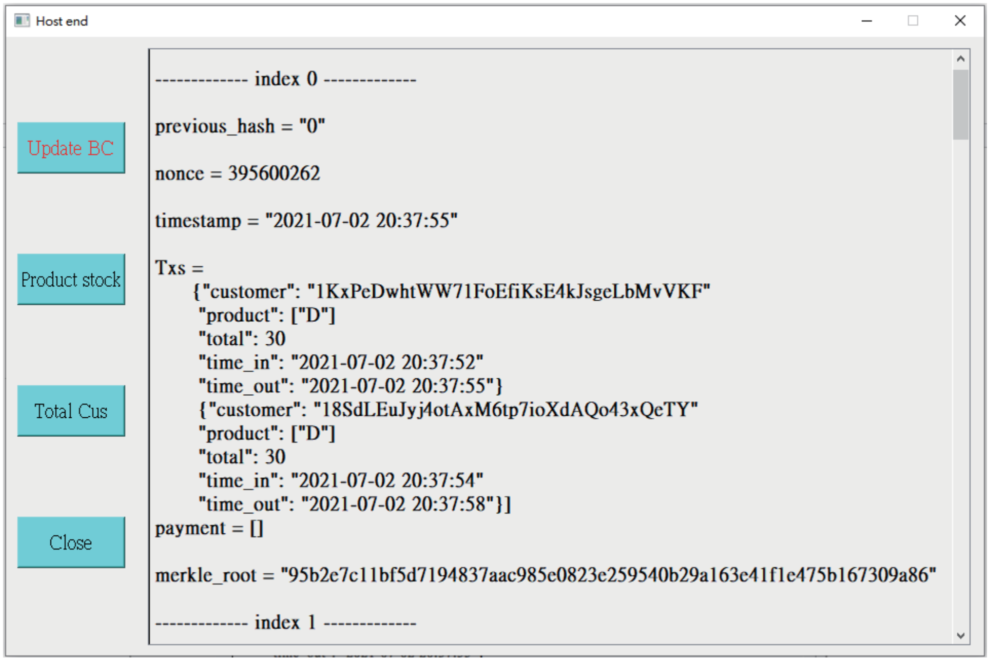
4. Experiment and Results

In this study, we designed a blockchain system for unmanned stores. Python is used as the main programming language for development. Python is chosen to develop the whole system using the same programming language, thereby reducing the programmatic discrepancies between the blockchain architecture and FaceNet. PyQt5 is used to create a UI to assist system users in operating and viewing related information so that many views, including block information, public ledgers, and functional operations, can be displayed more intuitively. The experimental results are described below, including the host endpoint, recognition endpoint, client endpoint, and image encryption/decryption algorithms. The first three use the UI of the terminal to display the simulated transaction results, and the encrypted image is used for image encryption and decryption. The parameters obtained in the RSA algorithm are used for demonstration.

4.1 Host endpoint

The host endpoint is responsible for receiving identification information, compiling customer information, verifying and packaging blocks, communicating with customers, and recording product inventory. In the following, we demonstrate the results of each function.

Figure 10 showcases the UI that contains the buttons to update the blockchain on the host endpoint. The host is the message integration and verifier in the system, so the functions of receiving, sending, responding, verifying, and packaging blocks are forced to be set to passive.



```
------------ index 0 ------------

previous_hash = "0"

nonce = 395600262

timestamp = "2021-07-02 20:37:55"

Txs =
     {"customer": "1KxPeDwhtWW71FoEfiKsE4kJsgeLbMvVKF"
      "product": ["D"]
      "total": 30
      "time_in": "2021-07-02 20:37:52"
      "time_out": "2021-07-02 20:37:55"}
      {"customer": "18SdLEuJyj4otAxM6tp7ioXdAQo43xQeTY"
      "product": ["D"]
      "total": 30
      "time_in": "2021-07-02 20:37:54"
      "time_out": "2021-07-02 20:37:58"}]
payment = []

merkle_root = "95b2e7c11bf5d7194837aac985e0823e259540b29a163e41f1e475b167309a86"

------------ index 1 ------------
```

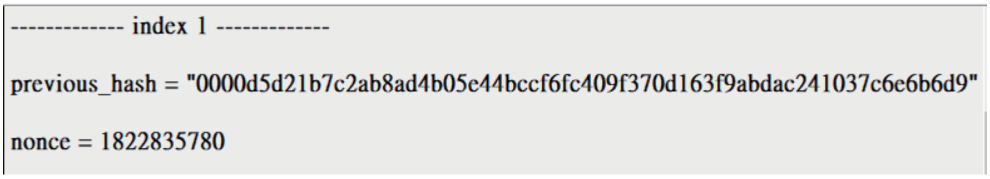
Fig. 10. (Color online) Buttons for updating the blockchain.

This means that the operator cannot make changes to the message; rather, the background program handles the operations automatically. This ensures that the host, although with the most authority, remains neutral to the greatest extent possible. The controllable components include Update Blockchain (Update BC), view inventory (Product stock), total customer (Total Cus), and shutting down the system (Close).

The Update BC button is used to read the data stored on the blockchain. The data is the result obtained after the consensus mechanism. The block shown in Fig. 10 contains the first transaction that is successfully packaged and stored, so it is given an index of 0. Since it is the first block, the previous hash is 0. The PoW result, or the nonce, is 395600262. This means that when the nonce of this block is equal to 395600262, the block hash value is “0000d5d…”. Figure 11 shows the information of the next block. The previous hash value is the same as the block’s hash value with index 0. This demonstrates that the consensus mechanism of index 0 has been correctly implemented. The timestamp records the time at which the first host packages the block. Since all hosts are independently verified nodes, a consensus mechanism is needed to determine how all nodes can recognize a block produced by a single node. The timestamp is an important parameter to help confirm this procedure. Transaction (Txs) is a field that records the detailed information of each transaction. The figure shows two transaction records. This means that the verification process starts when the host’s number of complete transactions reaches 2. The threshold is only set to 2 for display and testing purposes; the threshold is set to a higher number in real scenarios. The payment column records the information relevant to how much the customer paid for the product. The Merkle root is the result of calculating the transaction with the Merkle tree. This parameter is used to quickly confirm whether the packaged transactions are identical and is also one of the parameters that help realize the consensus mechanism. The length of the Merkle root in the figure is 64 bits. Since the value is calculated using the hash function, the total number of bits is 256.

4.2 Recognition endpoint

Figure 12 shows the UI of the recognition endpoint. The camera needs to be turned on in order to operate the endpoint since customer identity authentication via face recognition is required when sending a message. If the face is not recognized in the database, the buttons do not react when pressed by the user. In Fig. 12, the figure in the frame was successfully identified and marked using a red border, and the white text “16qNZ…” is the address of the marked figure. Three message windows can be seen on the left side of Fig. 13: Enter, Beverage A, and Exit.



```
------------- index 1 -------------

previous_hash = "0000d5d21b7c2ab8ad4b05e44bccf6fc409f370d163f9abdac241037c6e6b6d9"

nonce = 1822835780
```

Fig. 11. Previous hash value.

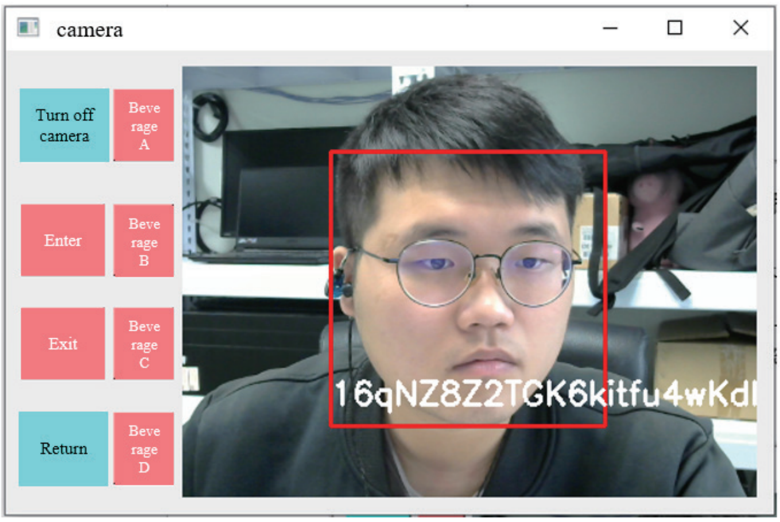
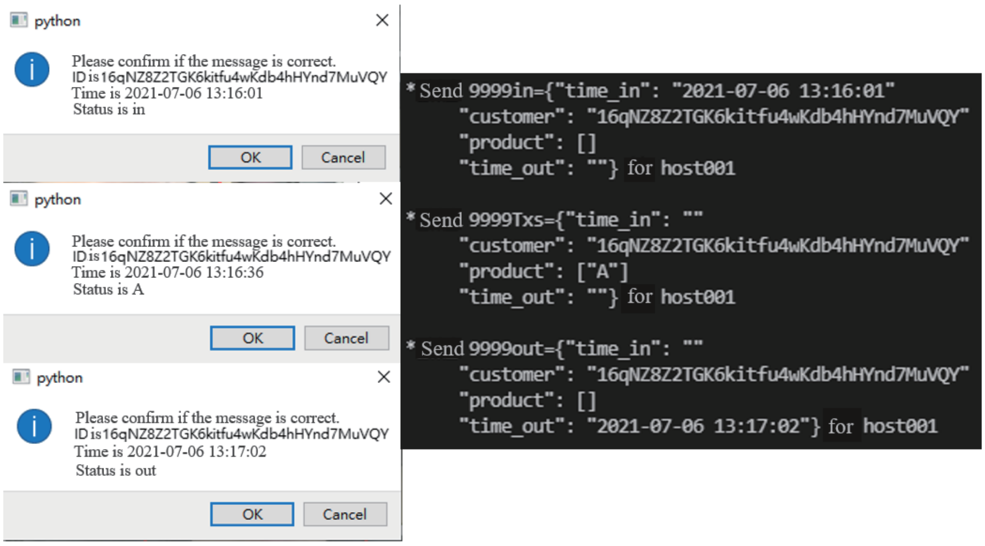


Fig. 12. (Color online) Functionality of the recognition endpoint.



Please confirm if the message is correct.
ID is 16qNZ8Z2TGK6kitfu4wKdb4hHYnd7MuVQY
Time is 2021-07-06 13:16:01
Status is in

Please confirm if the message is correct.
ID is 16qNZ8Z2TGK6kitfu4wKdb4hHYnd7MuVQY
Time is 2021-07-06 13:16:36
Status is A

Please confirm if the message is correct.
ID is 16qNZ8Z2TGK6kitfu4wKdb4hHYnd7MuVQY
Time is 2021-07-06 13:17:02
Status is out

```
* Send 9999in={"time_in": "2021-07-06 13:16:01"
    "customer": "16qNZ8Z2TGK6kitfu4wKdb4hHYnd7MuVQY"
    "product": []
    "time_out": ""} for host001

* Send 9999Txs={"time_in": ""
    "customer": "16qNZ8Z2TGK6kitfu4wKdb4hHYnd7MuVQY"
    "product": ["A"]
    "time_out": ""} for host001

* Send 9999out={"time_in": ""
    "customer": "16qNZ8Z2TGK6kitfu4wKdb4hHYnd7MuVQY"
    "product": []
    "time_out": "2021-07-06 13:17:02"} for host001
```

Fig. 13. (Color online) Results of pressing buttons.

Figure 13 shows the messages sent by the recognition endpoint after the user presses a button. These messages are for testing and display purposes only. The node of the recognition endpoint is set to be 99, and hence, the messages are prefixed with "9999," which represents the recognition endpoint and the node that sent the message. The following "in," "Txs," and "out" indicate the context category in which the message is sent. The transaction is then sent to the transaction pool, where the host endpoint determines whether the transaction already exists in the pool. If it does and the transaction is not yet complete, the fields that are incomplete are updated; if it does not or if the transaction is already complete, the transaction is viewed as a

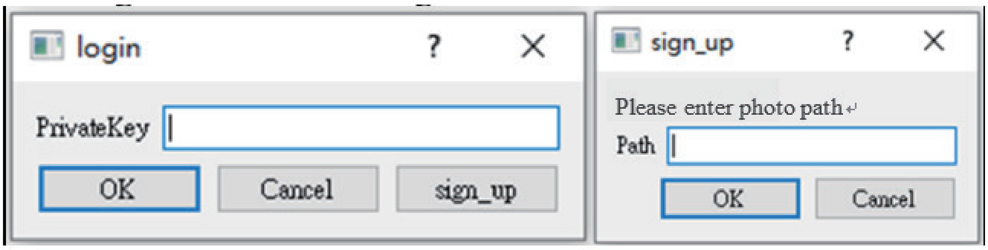
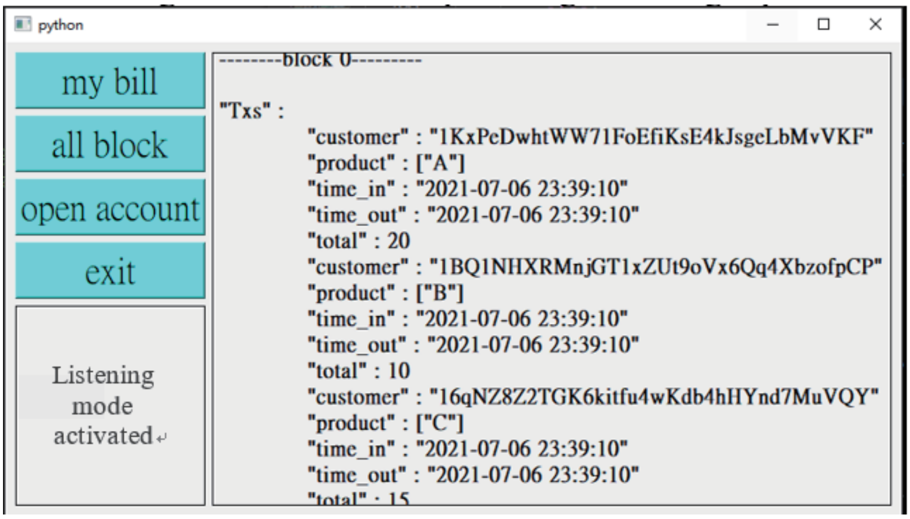


Fig. 14. (Color online) Customer endpoint login and signup.



```
--------block 0---------

"Txs" :
            "customer" : "1KxPcDwhtWW71FoEfiKsE4kJsgcLbMvVKF"
            "product" : ["A"]
            "time_in" : "2021-07-06 23:39:10"
            "time_out" : "2021-07-06 23:39:10"
            "total" : 20
            "customer" : "1BQ1NHXRMnjGT1xZUt9oVx6Qq4XbzofpCP"
            "product" : ["B"]
            "time_in" : "2021-07-06 23:39:10"
            "time_out" : "2021-07-06 23:39:10"
            "total" : 10
            "customer" : "16qNZ8Z2TGK6kitfu4wKdb4hHYnd7MuVQY"
            "product" : ["C"]
            "time_in" : "2021-07-06 23:39:10"
            "time_out" : "2021-07-06 23:39:10"
            "total" : 15
```

Fig. 15. (Color online) Client endpoint user interface.

new transaction and saved. The term host001 is the host's IP address where the message is sent. It is only used for display purposes and does not exist in the content of the transmitted message.

4.3 Client endpoint

The left side of Fig. 14 shows the screen that the customer uses to log in. After the customer enters the private key, the system confirms the input. If the input is correct, the system displays the interface shown in Fig. 15. There is also a sign-up button on the login screen. The interface for signing up is shown on the right of Fig. 14. Path is the parameter associated with the customer's own photo. The image is encrypted and then uploaded to the host to be stored in the face database given a correct input. In Fig. 15, the client-side functions include details related to the user (my bill), the blockchain (all block), outstanding payments (open account), and exit (exit). The box below the button shows whether network transmission and reception functions are opened as a default.

The figure shows the blockchain that has implemented the consensus mechanism and acts as the main chain in the double-chain structure shown in Fig. 2. It is responsible for storing information on the blockchain so that the content of the ledger is stored among decentralized nodes.

4.4 Transaction verification time: testing and analysis

(1)Transaction verification—experiments: To test the change in the time needed for transaction verification with and without a sidechain, as shown in Table 1, we used 3, 5, and 8 computers as nodes for simulation verification and packaging blocks. The verification time of each computer during the test, as well as the time it takes for the transaction to be stored in its own blockchain, is recorded. The latter is taken as the indicator for analyzing the transaction processing speed. In addition, the experiment also records the time difference between each transaction being sent and being fully incorporated into the blockchain. This is the confirmation latency, which is mainly used to evaluate the performance of the blockchain architecture. The transaction threshold is set to 4, the delay time of each message sent is set to 0.5 s, and the total number of transactions sent is 30, 50, and 100. A total of nine different combinations of these factors are evaluated.

(2) Experiment data and results: Figure 16 shows the result of three hosts and a total of 30 transactions. The data of the three hosts from the 1st transaction to the 28th transaction completely overlap, which proves that the information received by all nodes is exactly the same. In this specific sequence of transmissions, there is no fork, no reception errors, and no changes of stored blocks.

However, if the number of transactions increases, different parties can easily produce different times due to the different blocks being used and factors such as receiving delays. For instance, host 03 receives the message the latest, but the time it takes to receive the block is the longest, so when computing the subsequent total blockchain storage time, this value is used as the total transaction processing time.

Table 2 shows a partial result of five hosts and a total of 30 transactions. Next, we explain what happens if a fork occurs or if the stored block is replaced.

Given that the threshold is 4, 12 blocks will usually be generated, including 28 transactions. The remaining 2 transactions that are not packaged into blocks need to wait for subsequent

Table 1
Experiment datasheet.

Number of hosts	Transaction threshold	Delay between messages	Total number of transactions
3	4	0.5	30 50 100
5			30 50 100
8			30 50 100

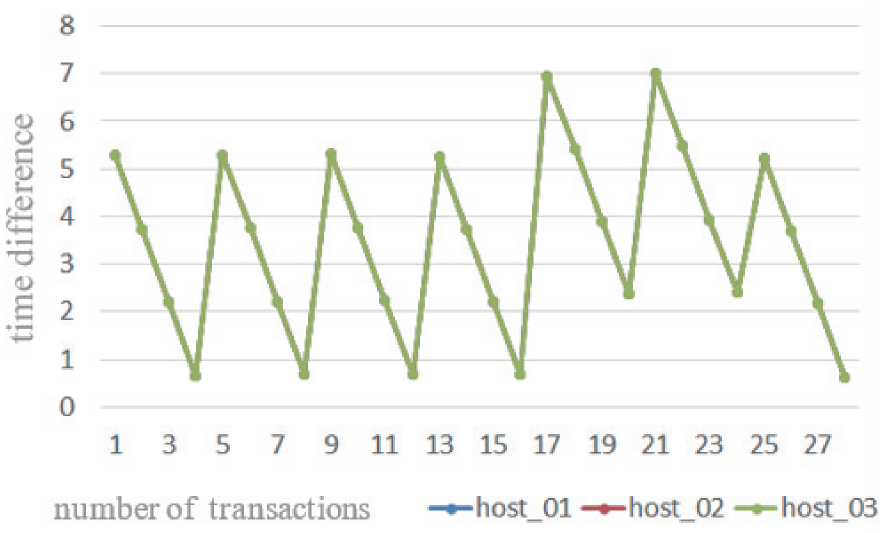


Fig. 16. (Color online) Comparison of the number of transactions and time difference.

Table 2
Partial results.

Number of transactions	Host 01	Host 02	Host 03	Host 04	Host 05
25	0.67935	5.20455	5.20455	5.20455	5.2045
26	5.29906	3.67137	3.67137	3.67137	3.6713
27	3.76773	2.14004	2.14004	2.14004	2.1400
28	2.22065	0.59296	0.59296	0.59296	0.5929
29	0.69035	N/A	N/A	N/A	N/A

transactions to be submitted until the number of transactions in the transaction pool reaches the threshold. Table 2 shows the results of 25 to 29 transactions among five hosts, with 30 transactions in total. Among them, host 01 has one more transaction than the other hosts. This is because the number of transactions in the transaction pool is greater than or equal to the threshold. It is possible that a block contains more transactions than the threshold since transactions can be packaged as long as the number of transactions in the transaction pool is greater than or equal to the threshold. The shorter the delay time between transactions, the higher the probability of this happening. Although it is possible, this scenario also results in forks. Therefore, all nodes must reach consensus, which means more than 2/3 of the nodes agree.

Table 3 shows the time difference between the complete received transaction and the storage block, obtained using nine combinations. Figure 17 shows the same data visualized as a chart. It can be seen from the table that when the number of hosts is three, the overall processing speed is faster than when the number of hosts is eight, because transmission is faster, and there are fewer forks and fewer transmission errors. However, when the total number of transactions is 100, the

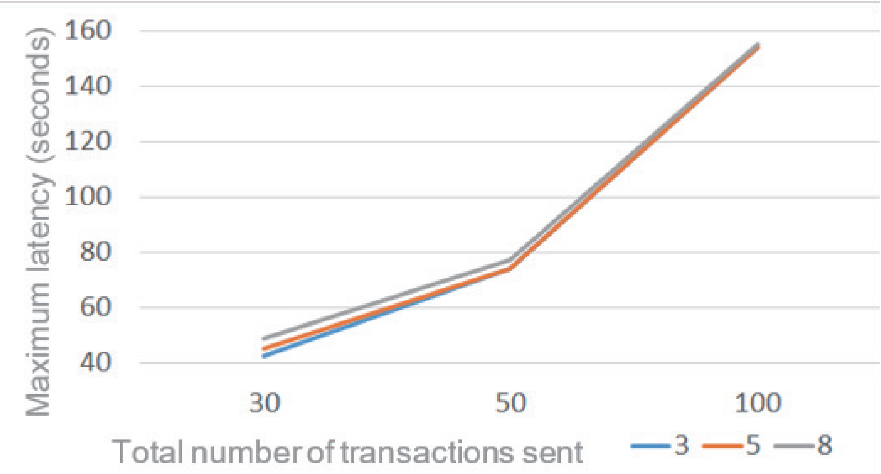


Fig. 17. (Color online) Maximum latency vs. total number of transactions.

Table 3
Time difference between the complete transmission of the transactions and storage of the block.

Number of transactions /Number of hosts	30	50	100
3 units	42.80572	74.07792	153.9889
5 units	45.16756	74.02687	154.0619
8 units	49.08229	77.30071	154.9087

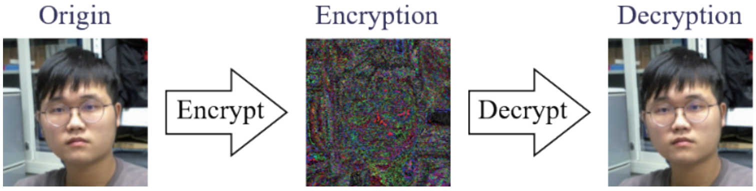


Fig. 18. (Color online) RSA encryption/decryption.

transmission delay time difference is reduced. A fork condition may also occur, so the difference between host 03 and host 08 is not as large as 30 transactions.

(3) Image encryption/decryption: Figure 18 shows the RSA encryption and decryption algorithm on a photo. In the figure, we can see that after the original image (Origin) is encrypted, the encrypted image (Encryption) can be obtained. An image very similar to the original image can be obtained after decryption (Decryption).

Table 4 shows the resulting parameters of RSA encryption and decryption. The random numbers p and q parameters have 40 digits in the experiment. The product N_m of p and q is a 78-digit number. The Euler function $\varphi(N_m)$ has the same number of digits. After the required C, e, N_m is calculated, the value is sent to the host endpoint for decryption.

Table 4
Resulting parameters for RSA encryption/decryption.

Parameters	Results
p	2342627019308745662639243051224167109617
q	2805550489414747409210953265385164573089
e	13
d	25278301463607162780000143949515858148388352325948255598534220680961942814527
$\varphi(N_m)$	65723583805378623228000374268741231185809716047465464556188973770501051316 8018
N_m	65723583805378623228000374268741231186324533798337813863373993402161984485 1063
SSIM	0.9997
Encryption time (s)	3.7019
Decryption time (s)	81.7056

In the process of decryption, the host endpoint uses e to obtain the value of d. Then, each pixel and its related parameters are passed through Eq. (8), after which the plaintext M can be obtained. In this experiment, the two images after encryption and decryption are used for SSIM evaluation, and a value of 0.9997 is obtained. Therefore, it can be considered that the decrypted value is very similar to the original image and can be used in the face recognition system.

5. Conclusion

In this study, we utilized the blockchain architecture to design an information transmission and communication system for unmanned stores to create a network for exchanging information that does not rely on a third party. The basic infrastructure uses concepts such as distributed ledgers and encryption and decryption algorithms. To authenticate the identities of the store's customers, we used the face recognition network FaceNet developed by Google. FaceNet has several advantages that make it suitable for performing identity authentication in an unmanned store, including its high accuracy and convenience. A database of face images is required to perform facial recognition. The database contains encrypted images and performs decryption when needed. RSA is used for encryption and decryption.

The system proposed in this paper has three identity endpoints, namely, the host end, the identification end, and the client end. Messages are transmitted through a P2P network, and a directed acyclic graph is used to achieve message broadcasting while avoiding infinite loops when sending and receiving messages. The system utilizes a sidechain as its main endpoint architecture to reap the benefits of a centralized network while maintaining the impartiality of ledgers belonging to members with lower authority so that they do not have to worry about transaction records being tampered with. In addition, fewer people need to come to a consensus on a sidechain, reducing transaction delay and improving scalability.

References

1 D. Yaga, P. Mell, N. Roby, and K. Scarfone: Blockchain Technology Overview. https://nvlpubs.nist.gov/nistpubs/ir/2018/NIST.IR.8202.pdf (accessed July 2019).

2 P. Treleaven, R. G. Brown, and D. Yang: IEEE Comput. **50** (2017) 14.

3 Z. Li, J. Kang, R. Yu, D. Ye, Q. Deng, and Y. Zhang: IEEE Trans. Ind. Inf. **14** (2018) 3690.

4 S. Sahai, N. Singh, and P. Dayama: IEEE Int. Conf. Blockchain (2020) 134. https://doi.org/10.1109/Blockchain50366.2020.00024

5 V. Buterin: Ethereum White Paper: A Next-Generation Smart Contract and Decentralized Application Platform (2015) 102471. https://github.com/ethereum/wiki/wiki/White-Paper (accessed July 2019).

6 A. Singh, K. Click, R. M.Parizi, Q. Zhang, A. Dehghantanha, and K.-K. R. Choo. J. Network Comput. Appl. **149** (2020) https://doi.org/10.1016/j.jnca.2019.102471

7 S. Nakamoto: Bitcoin A Peer-to-Peer Electronic Cash System (2008). https://bitcoin.org/bitcoin.pdf

8 K. Lei, M. Du, J. Huang, and T. Jin: IEEE Trans. Serv. Comput. **13** (2020) 252. https://doi.org/10.1109/TSC.2019.2949801

9 R. Wang, K. Ye, and C. Z. Xu: 2019 Int. Conf. Blockchain **LNSC, 11521** (ICBC 2019) 171. https://doi.org/10.1007/978-3-030-23404-1_12

10 M. Conti, A. Gangwal, and M. Todero: ARES'19 14th Int. Conf. Availability, Reliability and Security (2019) 26. https://doi.org/10.48550/arXiv.1901.10019

11 R. L. Rivest, A. Shamir, and L. M. Adleman: Commun. ACM **21** (1978) 120. https://doi.org/10.1145/359340.359342

12 A. Karakra and A. Alsadeh: 2016 SAI Computing Conf. (SAI) (2016) 1016. https://doi.org/10.1109/SAI.2016.7556103

13 Z. Wang, A.C. Bovik, H. R. Sheikh, and E. P. Simoncelli: IEEE Trans. Image Process **13** (2004) 600. https://doi.org/10.1109/TIP.2003.819861

14 Y.-H. Chen, “Application of Symmetric Encryption/Decryption: Taking a Chest X-ray Medical Image as an Example”, National Chin-Yi University of Technology Department of Electrical Engineering, Master's thesis, (2020).