Promize - Blockchain and Self Sovereign Identity Empowered Mobile ATM Platform

Eranga Bandara[1], Xueping Liang[2], Peter Foytik[1], Sachin Shetty[1], Nalin Ranasinghe[3], Kasun De Zoysa[3], and Wee Keong Ng[4]

[1] Old Dominion University, Virginia USA
{cmedawer, pfoytik, sshetty}@odu.edu
[2] University of North Carolina at Greensboro, NC, USA
{x_liang}@uncg.edu
[3] University of Colombo School of Computing, Sri Lanka
{dnr, kasun}@ucsc.cmb.ac.lk
[4] School of Computer Science and Engineering
Nanyang Technological University, Singapore
{awkng}@ntu.edu.sg

Abstract. Banks provide interactive money withdrawal/payment facilities, such as ATM, debit and credit card systems. With these systems, customers could withdraw money and make payments without visiting a bank. However, traditional ATM, debit and credit card systems inherit several weaknesses such as limited ATM facilities in rural areas, the high initial cost of ATM deployment, potential security issues in ATM systems, high inter-bank transaction fees etc. Through this research, we propose a blockchain-based peer-to-peer money transfer system "Promize" to address these limitations. The Promize platform provides a blockchain-based, low cost, peer-to-peer money transfer system as an alternative for traditional ATM system and debit/credit card system. Promize provides a self-sovereign identity empowered mobile wallet for its end users. With this, users can withdraw money from registered banking authorities (e.g shops, outlets etc) or their friends without going to an ATM. Any user in the Promize platform can act as an ATM, which is introduced as a mobile ATM. The Promize platform provides blockchain-based low-cost inter-bank transaction processing, thereby reducing the high inter-bank transaction fee. The Promize platform guarantees data privacy, confidentiality, non-repudiation, integrity, authenticity and availability when conducting electronic transactions using the blockchain.

Keywords: Blockchain ; Self-Sovereign Identity ; Smart Contract ; Electronic Payments ; Cloud Computing

1 Introduction

Automatic Teller Machine (ATM) and debit/credit card systems are popular electronic transaction methods provided by banks. With these systems, customers can do money withdrawals to cash, payments, access their bank account

information, check balance, etc without visiting a bank teller. It is a pay-now payment system which is primarily used for small and macro transactions [40]. Today ATM's are a common feature in most banks revealing that the ATM is a successful technology that humans have adapted to. Even though these systems are popular electronic transaction systems, they inherit weaknesses and loopholes. Below is a list of some of them.

1. The initial cost of setting up an ATM system is very high, requiring separate internet connections, security systems, surveillance camera systems, and buildings to host.
2. ATMs are targets for various frauds and attacks [30]. To cope with these scenarios banks need to invest more money, time and well-trained staff members.
3. ATMs are not evenly scattered in the country. Most of the ATMs are located in urban areas, due to this reason customers have to travel long distances to use ATM facilities.
4. High inter-bank ATM transaction fees exist. All inter-bank transactions go through a third party centralized clearing agent. So for each inter-bank ATM transaction, customers need to pay a high transaction fee. For instance, in Sri Lanka, it costs LKR 20 per transaction.
5. High ATM transaction fee. For each transaction, customers need to pay a relatively high transaction fee. For example, Sri Lankan banks charge LKR 5 per single ATM transaction.
6. Anyone can steal the card number or the CVV number of debit or credit cards and make electronic payments. Physical cards are not required to make electronic payments.
7. Debit and credit cards are easily clone-able. Attackers can clone the card of a user and use it.
8. In many developing countries, national ATM switches are not implemented. Therefore, inter-banking ATM facilities are not available.

With this research, we propose a blockchain and self-sovereign identity [36]-based low-cost peer-to-peer money transfer system, "Promize" to address the above-mentioned challenges in traditional ATM systems and Debit/Credit card systems. Promize can be used as an alternative for traditional ATM systems(inter-bank ATM systems) and debit/credit card systems. With Promize, users are given the ability to withdraw funds using registered authorities or even through their friends without having to go to an ATM. Any user in the Promize platform can act as an ATM, which is introduced as a mobile ATM. All the Promize transactions are done through the self-sovereign identity empowered Promize mobile wallet application with using QR codes. Users need to link their bank account to the Promize wallet application. The Promize platform automatically links the bank accounts using the core-banking APIs. Once the wallet is linked with a bank account, the Promize wallet can be used as an alternative for traditional debit/credit cards. Users can purchase goods from shops and pay via the Promize wallet instead of using a debit or credit card. Promize does not provide a traditional crypto-currency application with its blockchain. Instead,

it provides a low-cost inter-bank money transfer and payment system with a blockchain. The blockchain system in the Promize platform is used to link multiple banks in the network. The Promize platforms' low-cost peer-to-peer money transfer system guarantees, non-repudiation, confidentiality, integrity, authenticity and availability of the transactions [26] with the help of blockchain integration. We have deployed a live version of the Promize platform at the Merchant Bank of Sri Lanka (MBSL) [33]. MBSL's ATMs are located only in urban areas, with no coverage in the rural areas of the country. Due to this, they provide ATM facilities to their customers in rural areas through the Promize platform. The main contributions of "Promize" are as follows.

1. Blockchain-based low-cost peer-to-peer money transfer system "Promize", introduced to address the challenges in traditional ATM systems and Debit/Credit card systems.
2. Self-Sovereign identity empowered mobile wallet(for Android/IoS) has been introduced to facilitate peer to peer money transfer and payment transactions. The transactions can be done with QR code scanning.
3. Promize platform based ATM system is introduced to MBSL bank customers in Sri Lanka to facilitate their day-to-day money withdrawal requirements.
4. Blockchain-based low-cost inter-banking transaction system has been introduced to Sri Lankan banks.

The rest of the paper is organized as below. Section 2 discusses the architecture of the Promize platform. Section 3, the functionality of the Promize platform. Section 4 describes the production deployment of the Promize platform in the MBSL bank, Section 5 discusses the evaluation of the introduced platform. Section 6 surveys related work. Section 7 concludes the Promize platform with suggestions for future work.

2 Promize Platform Architecture

2.1 Overview

Promize is a blockchain-based peer-to-peer money transfer platform. It can be used as an alternative for traditional ATM systems as well as traditional electronic payments. The architecture of the Promize platform is described in Figure 1. It contains four main components.

1. Core bank layer - Each bank in the Promize platform has its own core banking system. The core bank maintains user bank accounts. It provides API's for account management, fund transfers etc.
2. Distributed ledger - All users' decentralized identities (DID) and transaction records are stored here.
3. Money transfer layer - Electronic transactions and verification happen here.
4. Communication layer - Data exchanges between Promize wallets happen in this peer-to-peer communication layer.

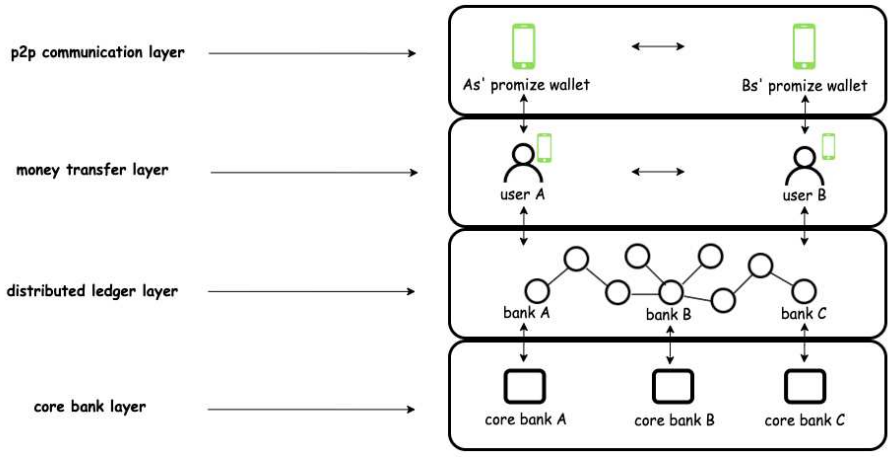


Fig. 1: Promize platform architecture. The core bank API maintains user bank accounts and supports fund transfers etc. A distributed ledger is used to store user identities and transactions. Electronic money transfer transactions occur via the Money transfer layer. The peer-to-peer communication layer is used to exchange transaction information.

2.2 Core Bank Layer

Each bank in the Promize platform has its own core banking system. The core bank maintains user bank account information, account balances etc. It provides an API to search for user accounts and make fund transfers. The blockchain nodes deployed in the banks interact with their respective core banking APIs to search for user accounts, validate user accounts and perform fund transfers between accounts (for example, the blockchain node deployed in Bank A interacts with Bank A's core banking API). There are some common core banking systems which most banks use(e.g Finacle core banking system [31]). These core banking systems expose JSON-based REST APIs or SOAP web service-based APIs to facilitate banking functions such as account creation, account validations, fund transfers etc. The blockchain smart contracts in the Promize platform interact with these core banking APIs to perform user account validations and transfer funds between user accounts.

2.3 Distributed Ledger

The distributed ledger is the blockchain-based peer-to-peer storage system in the Promize platform. The blockchain can be deployed among multiple banks. Each bank in Promize can run their own blockchain node. These blockchain nodes are connected as a ring cluster, Figure 2. Customers of each bank connect to their respective blockchain peer in the network. Blockchain stores all Promize platform users' digital identity (which is identified as DID or decentralized identity proof [6,36]) information as self-sovereign identity. It also stores all the Promize

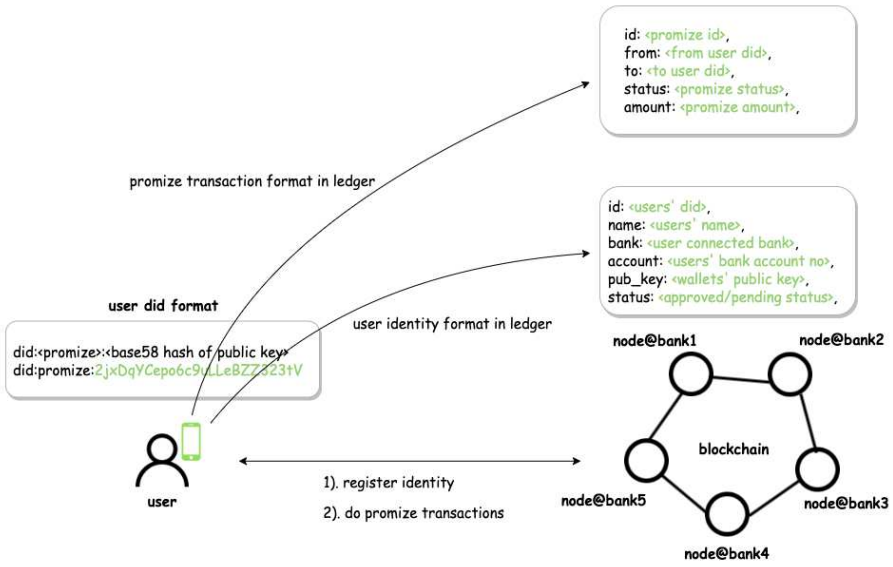


Fig. 2: Promize platform distributed ledger stores user identities and promize transactions. Promize mobile wallet interacts with the blockchain to create user identities and do Promize transactions.

transaction information in the blockchain. The user identity and transaction information which is stored in one banks' blockchain node will be synced with all other bank blockchain nodes by using the underlying blockchain consensus algorithm. The smart contracts running on the blockchain interact with the core bank API to facilitate customer bank account validations and fund transfers.

We have used Rahasak blockchain [7] ; a highly scalable blockchain targeted for big data as the distributed ledger of the Promize platform. Rahasak comes with real-time transaction enabled "Validate-Execute-Group" blockchain architecture [5, 8] with using Paxos [29] and ZAB [25] based consensus. It introduced concurrent transaction enabled functional programming [22] and actor [20]-based "Aplos" smart contract platform to facilitate blockchain functions [8]. All blockchain functions of the Promize platform are implemented with Aplos smart contracts. "Identity smart contracts" are used to handle user identities and the"Promize smart contracts" are used to handle the Promize money transfer transaction. With Rahasak blockchain we were able to support real-time transactions [16, 34], high transaction throughput, high scalability, backpressure operation handling features on Promize platform.

2.4 Money Transfer Layer

All peer-to-peer money transfers take place in the "Money transfer layer". The users of the Promize platform will be given the Promize mobile wallet applica-

tion. Peer-to-peer money transfers occur via this mobile wallet. Users first need to register on the Promize platform and link their bank account to the Promize mobile wallet. To use Promize, users need to have a bank account in any bank which participates in the Promize platform. When registering, the app will generate a public and private key pair which corresponds with the user/mobile wallet. The private key will be saved on the Keystore of the mobile application. The public key and base58 [17] hash of the public key will be uploaded to the blockchain along with other user attributes (e.g. account number, bank name etc). The base58 hash of the public key will be used as the digital identity (DID) [36] of the user on the Promize platform. Figure 2 shows the format of the DID which is generated by the mobile wallet. This DID will be embedded to the QR code in the mobile app, which the user can show to any relevant parties (e.g cashier, friend, bank branch officer etc) when performing peer-to-peer money transfers via the Promize platform.

2.5 Peer to Peer Communication Layer

When making payments or performing transactions via the Promize platform, users need to exchange transaction details with the blockchain ledger and mobile wallet applications. The peer-to-peer communication layer used to exchange this transaction information between Promize mobile wallet applications and the blockchain. The peer-to-peer communication layer can be implemented with a TCP/WebSocket-based communication service or a push notifications service, similar to Firebase [27] (it works on top of 3G, 4G cellular network). In the Promize platform, we have used the Firebase push notification service to implement the peer-to-peer communication between mobile wallets.

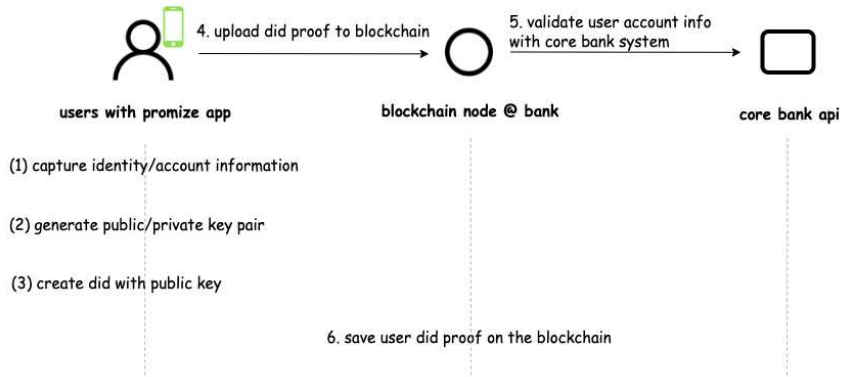
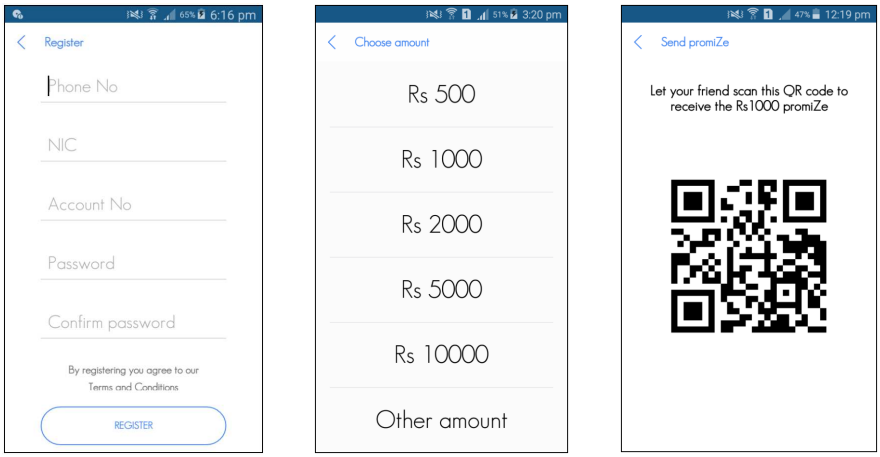


Fig. 3: User captures credentials and account information via the Promize mobile wallet and register on the Promize platform. The blockchain will store a proof of user credentials.

3 Promize Platform Functionality

3.1 Use Case

In traditional scenarios, users need to go to a bank ATM or bank branch to withdraw money. With Promize, instead of going to a bank ATM, users can withdraw money from registered authorities of the Promize platform (e.g shops, outlets etc) or a friend who has the Promize app installed. First, users register with the platform via the Promize mobile application. When a user registers, the mobile application captures user credentials, bank account information (Figure 4) and sends a registration request to the "Identity smart contract" on the blockchain. The registration request contains `user id(DID)`, `user name`, `public key`, `user bank name` and `bank account number`. Upon receiving the registration request, smart contracts validate user credentials, bank account information and create an identity for the user in the blockchain, Figure 3. The format of the user identity in the blockchain ledger can be seen in Figure 2. When validating, it will connect to the corresponding core bank system at the bank and verify the user's bank account details (e.g verify bank account number with identity number). For example, if a user belongs to Bank A, the blockchain node at Bank A will connect with its core bank system and validate the users' account information.

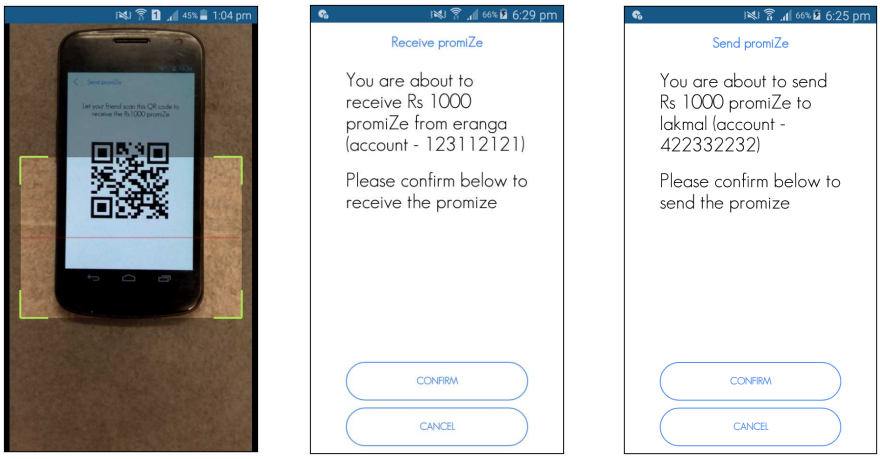


(a) Promize registration (b) Promize amount (c) Promize QR code

Fig. 4: Promize mobile wallet application registration. Promize sending user choose Promize amount and initiate Promize transaction. App generates a QR code embedding DID, account number and transfer amount.

Assume two users (User A and User B) are registered on the Promize platform and link their bank accounts to the Promize mobile wallet. User A wants to withdraw LKR 1,000. User A goes to User B, who has already installed the

Promize mobile app and registered on the Promize platform and asks for LKR 1000. If User B has LKR 1000 at hand, he could navigate to the "Send Promize" section on the mobile application and choose the amount to transfer. Now the app generates a QR code embedding User B's DID, account number and transfer amount, Figure 4(c).



(a) Scan Promize (b) Submit Promize (c) Confirm Promize

Fig. 5: Promize receiver scans the QR code and submits the transaction. A confirmation will be sent to Promize sender via push notification. Once Promize sender confirm it, the transaction will be approved.

Now User A navigates to the "Received Promize" section in his mobile application and scans the QR code from User B's phone 5(a). Once the QR code is scanned, the application will navigate to the Promize receive confirmation screen Figure 5(b), where User A can confirm it and submit a Promize transaction request to the blockchain, Figure 5. When submitting the transaction, it sends a Promize create request to "Promize Smart Contract" on the blockchain, which checks the validity of the transaction(double-spend check) and user accounts. Promize create requests contain `transaction id`, `transaction type(create)`, `User A's did`, `User B's did` and `transaction amount`. If the transaction is valid, the blockchain will generate a corresponding random number (`transaction salt`) and create a new Promize transaction in the blockchain ledger (status of the Promize transaction is marked as `pending`). The format of the Promize transaction in the blockchain ledger is shown in Figure 2. Then it sends this transaction salt and other transaction details to User B's Promize mobile wallet as a notification message (e.g Firebase push notification), Figure 6. This notification message contains `transaction id`, `User A's did`, `User B's did`, `transaction amount` and `transaction salt`.

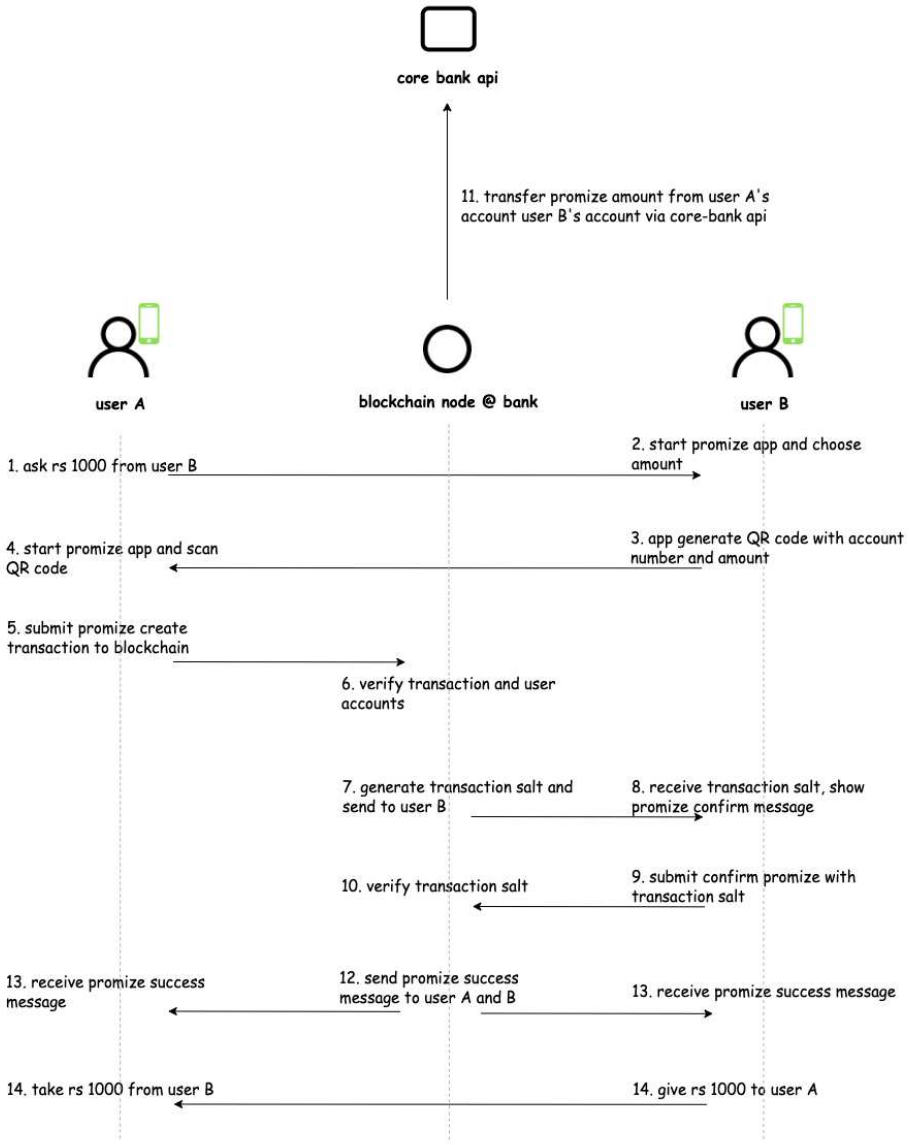


Fig. 6: Promize transaction flow between User A and User B. QR code is used to exchange information between mobile wallets.

Once the notification message is received by User B's Promize application, it will navigate to the Promize send confirmation screen, Figure 5. Then User B confirms it, thereby approving the Promize transaction.At confirmation, a Promize approval request will be sent with the received random number (transaction salt) to the "Promize Smart Contract" on the blockchain. The Promize

approve request contains `transaction id`, `transaction type(approve)`, `User A's did`, `User B's did`, `transaction amount` and `transaction salt`. Then the smart contract will check the validity of the transaction parameters (transaction salt and accounts). When validating, it will compare the transaction salt and account numbers of the Promize transaction saved in the blockchain ledger. If the transaction parameters are valid, it will transfer the requested amount (LKR 1000) from User A's bank account to User B's bank account by communicating with the core banking service. If this transfer is successful, it will update the Promize transaction status to `approved`. Finally, the transaction status will be sent to both users' Promize mobile applications. Then User B physically gives LKR 1,000 to User A, Figure 6.

The idea here is that User A physically receives money from User B which the bank electronically transfers to each other's accounts via Promize transactions. The same function can be used to purchase goods from shops. At the end of the purchase, users can send money to the shop's bank account via Promize transactions. Once the money is sent, the user receives goods.

3.2 Data Privacy and Security

The Promize platform guarantees data privacy, confidentiality, integrity, non-repudiation, authenticity and availability security features. To guarantee privacy, the Promize platform stores users' digital identity in the blockchain as a self-sovereign identity proof. Actual identity data such as ID numbers and signatures are stored on the users' physical mobile phone secure storage. When this information is needed for verification [19, 37], it will be directly fetched via the user's mobile application through push notifications. By using an SSI based approach, Promize platform addresses the common issues in centralized cloud-based storage platforms (e.g. lack of data immutability, lack of traceability). To guarantee confidentiality and integrity we have used RSA cryptography-based digital signature mechanism [24]. All data in the Promize platform are digitally signed by a corresponding party. The random number exchange with the QR code when performing a Promize transaction makes certain users meet in-person to carry out the transaction, which guarantees the non-repudiation. We have used a JWT based auth service to handle the authentication/authorization of the Promize platform. The users' authentication information (username/password fields) of Promize platform are stored in the JWT auth service [23]. Upon login, the user needs to send an authentication request to the auth service. Then, it verifies the credentials and returns the JWT auth token to the user. This auth token contains user permission, token validity time, digital signature etc. All subsequent requests need to be added to the JWT token and into the HTTP request header to perform the authentication and authorization. The token verification is handled by the gateway service in the Promize platform, Figure 7.

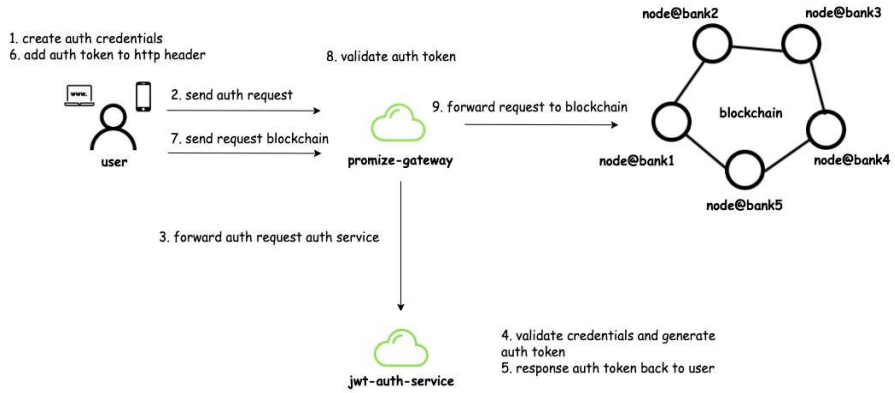


Fig. 7: JSON Web Token(JWT)-based auth service in the Promize platform. Auth tokens will be issued by auth service. Token validation happens at gateway service.

3.3 Low Cost Transactions

With Promize, the cost of electronic transactions can be easily reduced. Since multiple banks can share transaction and account information through the blockchain, it could facilitate low inter-bank transaction costs. Additionally, an inter-bank communication switch or an ATM switch is not required. With Promize, any user can act as an ATM, if they have money they can give it to their friends. If required, banks can appoint official Promize agents to act as ATMs. For example, a bank could appoint a trusted shop in a village as a Promize agent. Users in that village could go to this shop to withdraw money. They could also buy goods and pay via Promize transactions. It would reduce banks' costs to deploy and maintain physical ATMs and the operation costs of credit and debit cards. It's capable of providing services for a wide range of customers in rural areas, securely. In normal ATMs, customers take the money at the bank, which means the bank's money goes out. In Promize, users exchange money between accounts. For example, in above-mentioned scenario User B give's his physical money to User A, and Promize transfers money from User A's bank account to User B's bank account. In this case, the actual money at the bank remains the same. The money does not go out like in ATMs, it is another main advantage for the banks.

4 Promize Platform Production Deployment

4.1 Overview

We have deployed a live/production version of the Promize platform at the MBSL. Their branch network and ATM systems are mainly located in urban

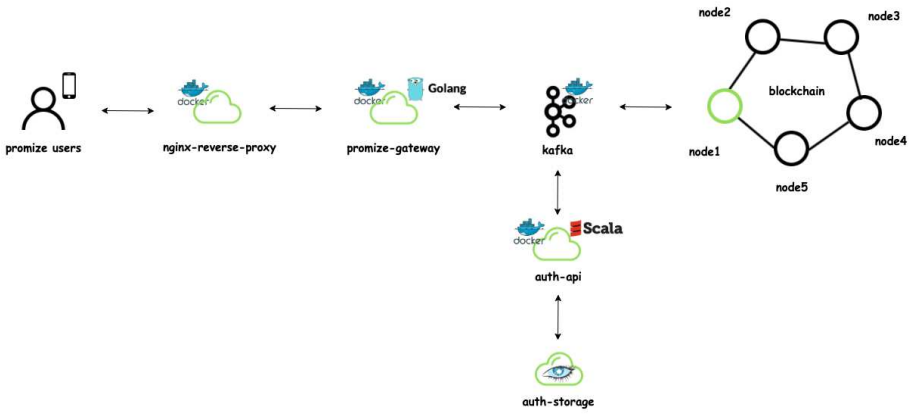


Fig. 8: Promize platform production system architecture which is deployed in MBSL Bank.

areas. This particular bank experiences a lack of ATM systems in rural areas. They use the Promize platform to facilitate banking and financial services (e.g money withdrawals, electronic payments etc) to people in rural villages in Sri Lanka. Promize mobile wallets are used as ATMs. They plan on appointing official bank agents in rural areas with a Promize mobile wallet. For example, supermarket/shops in the village could be bank agents. The agent's Promize mobile wallet will be installed in low-cost Sunmi android tablet devices. There is a built-in Bluetooth printer embedded in the Sunmi android tablet. Customers could go to a shop and withdraw money based on Promize protocol discussed in section 3. At the end of the transaction, the cashier of the shop prints a receipt and gives it to the customer using the receipt printer embedded in the Sunmi tablet. This provides greater transparency and validity to Proimze transactions.

4.2 Service Architecture

We have built the Promize platform to support high scalability and high transaction load in the banking environment. To cope with high transaction load and back-pressure [14] operations, we have adopted reactive streams based approach with using Akka streams [2, 13]. The Promize platform has been built using micro-services architecture [42]. All the services in the Promize platform are implemented as small services (micro-services) with the single responsibility principle. These services are dockerized [4, 35] and deployed using the Kubernetes [9] container orchestration system. Figure 8 shows the architecture of the Promize platform in MBSL. It contains the following services/components.

1. Nginx - Load balancer and reverse proxy service
2. Gateway service - Micro-services API gateway

3. Auth service - JWT based auth service
4. Apache Kafka - Micro-services message broker
5. Rahasak Blockchain - Blockchain ledger in the Promize platform

Nginx load balancer [38] is used as the reverse proxy in the Promize platform. The SSL certificates are set up in the Nginx and expose port 443 to the client. All mobile wallet applications connect with port 443 which is publicly exposed via public IP. The requests coming to Nginx will be redirected to the Gateway service, which is the microservices API gateway of the Promize platform. The gateway service is implemented with golang [3,39]. This service will validate auth tokens in client requests (HTTP header) and redirect them to the blockchain to handle Promize functionalities. Auth services are the JWT [23]-based auth service in the Promize platform. All client credentials(username, passwords) will be stored in auth services' auth storage. The auth service validates client credentials and issues auth tokens to clients. Each bank in the network has its own Gateway, Auth Service and Nginx-proxy. The bank clients(with Promize mobile applications) connect to the blockchain network via their respective banks' Nginx-proxy. Apache Kafka is used as the microservices message broker of the Promize platform. Inter-service communication takes place via Kafka message broker using JSON serialized messages. Every service in the Promize platform has its own Kafka topics to receive messages. We have used the highly scalable Rahasak blockchain to implement the Promize platform at MBSL. All blockchain functions are written with the Scala functional programming and Akka actor-based Aplos smart contract in the Rahasak blockchain. MBSL runs AS400-based core banking system [15]. It maintains customer accounts, account balance etc. There is a core banking API which is exposed to perform account inquiries, fund transfers etc. The blockchain smart contracts of the Promize platform interact with these core banking API perform user account validations and fund transfer between accounts.

5 Performance Evaluation

Performance evaluation of Promize was completed and is discussed. To obtain these results, we deployed the Promize platform with a multi-peer Rahasak blockchain cluster in AWS 2 x large instances (16GB RAM and 8 CPUs). Rahasak blockchain runs with 4 Kafka nodes, 3 Zookeeper nodes and Apache Cassandra [28] as the state database. The smart contracts on the Rahasak blockchain are implemented with Scala functional programming and the Akka actor-based [1, 18] Aplos [8] smart contract platform. The evaluation results are obtained for the following, with a varying number of blockchain peers (1 to 5 peers) used in different evaluations.

1. Transaction throughput of the blockchain
2. Transaction execution and validation time in Promize platform
3. Transaction scalability of the Promize platform
4. Transaction execution rate in the Promize platform

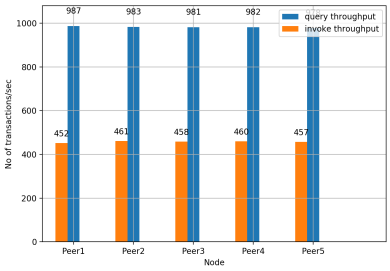


Fig. 9: Transaction throughput in the Promize blockchain

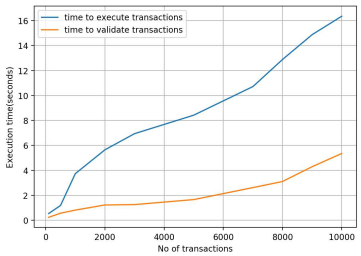


Fig. 10: Time to execute transactions and validate transactions in the Promize platform.

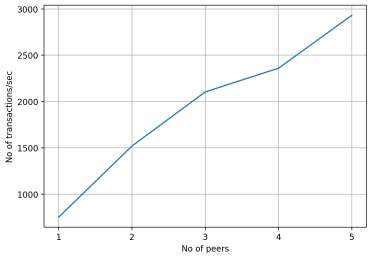


Fig. 11: Transaction scalability in the Promize blockchain.

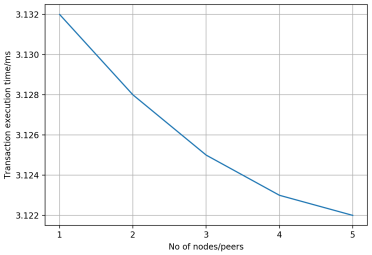


Fig. 12: Transaction latency in the Promize blockchain.

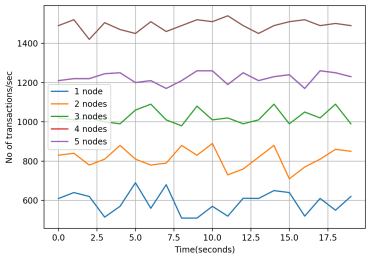


Fig. 13: Transaction execution rate with no blockchain peers in the Promize blockchain.

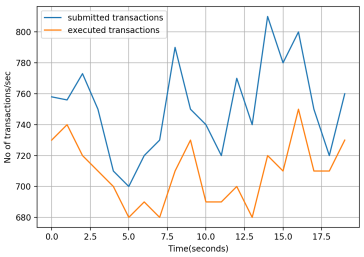


Fig. 14: Transaction execution rate and transaction submission rate in a single blockchain peer.

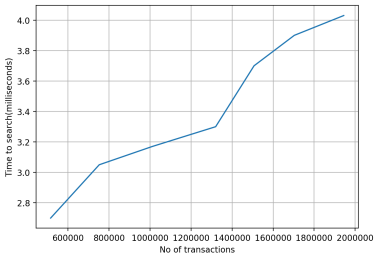


Fig. 15: Search performance of the Promize platform.

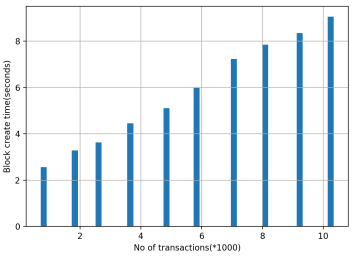


Fig. 16: Block creation time against the number of transactions in the block.

5. Search performance
6. Block creation time

5.1 Transaction Throughput

To complete this evaluation, we recorded the number of Promize create transactions and Promize query transactions that can be executed on each peer in the Promize platform. When creating Promize, an invoke transaction will be executed in the underlying blockchain. The invoke transaction would then create a record in the ledger and update the status of the assets in the blockchain. The query searches the status of the underlying blockchain ledger and can neither create transactions in the ledger nor update the asset status. We flooded concurrent transactions for each peer and recorded the number of completed results. As shown in Figure 9 we have obtained consistent throughput on each peer of the Promize platform. Since query's do not update the ledger status, it has high throughput(2 times) compared to invoke transactions.

5.2 Transaction Execution and Validation Time

We evaluated the transaction execution and transaction validation time and recorded the time to execute and validate the different set of transactions(100, 500, 1000, 2000, 3000, 5000, 7000, 8000, 10000 transactions). The transaction validation time includes the double-spend checking time. Transaction execution time includes the double-spend checking time, ledger update time, data replication time. Figure 10 shows how the transaction execution time and validation time varies in different transaction sets.

5.3 Transaction Scalability

The number of transactions that can be executed (per second) over a number of peers in the network is recorded for this evalutation. We flooded concurrent transactions on each peer and recorded the number of executed transactions. Figure 11 shows transaction scalability results. Each node addition to the cluster has increased the transaction throughput in a nearly linear fashion. This means that the transaction latency will be decreased when adding blockchain peers to the cluster, Figure 12. As peers are added, there is a diminishing return where the performance benefit will degrade if too many peers are added.

5.4 Transaction Execution Rate

Next, we evaluate the transaction execution rate in the Promize platform. We tested the number of submitted transactions and executed transactions in different blockchain peers, recording the time. Figure 13 shows how the transaction execution rate varies according to the different number of blockchain peers in the Promize platform. When the number of peers increases, the rate of executed

transactions also increases, relatively. Figure 14 shows the number of executed transactions and submitted transactions in a single blockchain peer. There is a back pressure operation [14] between the rates of submitted transactions and executed transactions. We have used a reactive streaming-based approach with Apache Kafka to handle these backpressure operations in the Promize platform.

5.5 Search Performance

The Promize platform allows searching it's transaction and account information via the underlying Rahasak blockchain's Lucene Index-based search API. We evaluated this criteria by issuing concurrent search queries into Promize and computing the search time. As shown in Figure 15, to search 2 million records connected it consumed 4 milliseconds.

5.6 Block Generate Time

Finally, we have evaluated the time taken to create blocks in the underlying blockchain storage of the Promize platform. The statistics recorded against the number of transactions in a block. Block generation time depends on a). data replication time b). Merkel proof/block hash generate time c). transaction validation time. When the transaction count increases in the block, these factors will also increase. Due to this, when the transaction count increases, block generation time too increases correspondingly. As shown in Figure 16 to increase a block when having 10k transaction, the platform consumes 8 seconds.

6 Related Work

Table 1: Peer-to-peer electronic payment platform comparison

Platform	Architecture	Running blockchain	Scalability level	Payment type	SSI support	Privacy level
Promize	Decentralized	Rahasak	High	Bank	Yes	High
Mobile-ATM [26]	Centralized	N/A	Mid	Bank	No	Low
RSCoin [12]	Decentralized	RSCoin	Mid	Crypto	No	Mid
DTPS [21]	Decentralized	Ethereum	Low	Visa/MasterCard	No	Mid
BPCSS [11]	Decentralized	Bitcoin	Low	Crypto	No	Mid
BCPay [43]	Decentralized	Ethereum	Low	Crypto	No	Mid
TTPayment [32]	Decentralized	Bitcoin	Low	Crypto	No	Mid
Syscoin [41]	Decentralized	Bitcoin	Low	Crypto	No	Mid

Much research has been conducted to improve the privacy/security features of electronic transactions. They seek to provide low-cost alternatives for traditional

banking and inter-banking transactions [26]. In this section, we outline the main features and architecture of these research projects.

Mobile-ATM [26] Mobile-ATM is a simple M-Commerce application that provides ATM services. The traditional ATM network is replaced with the proposed M-ATM system. This system is incorporated of a Bank, Customer and the M-ATM agent. Both, M-ATM agent and the customer in this system are required to have mobile phones set up to perform the functions of M-ATM. The bank has an M-ATM server that it hosts as a front-end interface and uses a back-end transaction management system. This tool proposes a new money withdrawal/deposit system (ATM system), enabling users to perform ATM transactions with their mobile phones with additional security features. The goal of this toot is to reduce barriers of using ATM systems while simultaneously improving security related to ATM transactions.

RSCoin [12], a centrally controlled cryptocurrency framework uses a sharding-based blockchain protocol that allows for better scalability and efficiency of the centrally-banked crypto-currency system. RSCoin utilizes a centralized monetary supply and distributed transaction ledger. A set of authorities are established and called `mintettes` that perform the transaction validation (double-spend checking). With a centralized monetary authority, RSCoin is able to scale better compared to other decentralized crypto-currencies. Currency supply is created and controlled by a central bank, making the cryptocurrency based on RSCoin significantly more palatable to governments. While monetary policy is centralized, RSCoin still provides strong transparency and auditability guarantees. Double spending is checked with a simple and fast mechanism and two-phase commit maintaining the integrity of the transaction ledger.

Delay-Tolerant Payment Scheme(DTPS) [21] proposes a cash-less payment system intended for rural villages where limited internet network connectivity exists. To work in an intermittent network environment, it uses blockchain mining nodes located locally in villages that can handle the transaction processing and verification. Base stations are set up and run in these remote villages using technology such as Nokia Kuha connected to the public Internet via unreliable satellite links. This in turn offers 4g coverage to the village while providing connectivity to the wider internet intermittently. Banks control and initiate deployment of the system utilizing the intermittent connectivity and periodically monitor system operations of the villages. Ethereum [10] blockchain-based smart contracts are used for payment service management including, user account initiation, interactions with credit operator, and management of rewards for blockchain miners. VISA and MAsterCard handle all the local transaction verification through the Ethereum blockchain in which they have implemented payment gateways with.

BPCSS [11] proposes a Blockchain-based Payment Collection Supervision System(identified as BPCSS). Customers and merchandise stores can use this system to help manage transactions when they want to use the pervasive Bitcoin digital wallet. All transaction details are efficiently saved on a cloud database immediately after customers use their NFC-enabled Android smartphone App

to purchase goods in the store using RFID-tags. Both customers and merchants are able to review transaction details in a soveriegn way without the need of a traditional finance system. Results presented for the tool are preliminary using the Bitcoin Testnet but demonstrate the cost-effectiveness of the proposed tool to easily and effectively manage transactions.

BCPay [43] BCPay is introduced to provide secure and fair payment of outsourcing services in general without relying on third-party (trusted or non-trusted). Blockchain-based fair payment framework for outsourcing services on cloud and fog computing. BCPay can provide sound and robust payments without the need to have an organization facilitate it. This is achieved by an all-or-nothing checking-proof protocol that ensures the outsourcing service provider performs their service completely or pays a fee. Performance evaluations show that BCPay is able to process a high number of transactions in a short amount of time without a large computational cost.

Thing-to-Thing payment(TTPayment) [32] is a bitcoin cryptocurrency-based automated micro payment platform for the Internet of Things. It allows devices to pay each other for services without a person required to interact. A proof-of-concept implementation exists of a smart cable that connects to a smart socket and pays for the electricity based on the draw without human interaction. Bitcoin is shown as a feasible payment solution for thing-to-thing payments, but high transaction fees exist when doing micro transactions. This is mitigated using the developed single-fee micro-payment protocol which aggregates multiple small transactions essentially batching them into one large transaction instead of many small transactions.

Syscoin [41] is a permissionless blockchain-based cryptocurrency providing blockchain-based e-commerce solutions for businesses of all sizes. Turing complete smart contracts built on the Bitcoin scripting system are used to control and interact with the system. Coin transactions are controlled by a hardened layer of distributed consensus logic. Each smart contract (Syscoin service) uses this hardened layer while still retaining backwards compatibility with the Bitcoin protocol. Commercial developers want to use the most powerful network available (currently Bitcoin), Syscoin allows for the utilization of this network whilc providing security and efficiency.

The comparison summary of these platforms and the Promize platform is presented in Table 1. It compares the Architecture(Centralize/Decentralized), Running blockchain, Scalability level, Supported payment types(bank payments, Visa/Mastercard payments, cryptocurrency payments), SSI support, Privacy level details.

7 Conclusions and Future Work

With this research, we have proposed a blockchain-based low-cost alternative for traditional ATM systems and credit/debit card systems, “Promize”. With Promize, users can withdraw money from registered authorities or their friends without going to an ATM. Any user in the Promize platform can act as an

ATM. All Promize transactions are done through the QR code-based Promize mobile wallet application. The Promize wallet application can be used as an alternative for traditional debit and credit cards. Users can purchase goods from shops and pay via the Promize wallet instead of using debit/credit cards. The Promize platform reduces the above-mentioned issues in ATM systems as well as debit/credit card systems. The Promize platform provides a secure, low-cost method of doing ATM, debit/credit card transactions while guaranteeing data privacy, confidentiality, integrity, non-repudiation, authenticity and availability security features.

We have proven the scalability and transaction throughput features of the Promize platform with empirical evaluations. Most recently we have deployed the 1.0 version of the Promize platform at the Merchant Bank of Sri Lanka.

Acknowledgements

This work was funded by the Department of Energy (DOE) Office of Fossil Energy (FE) (Federal Grant #DE-FE0031744).

References

1. Akka documentation, `https://doc.akka.io/docs/akka/2.5/actors.html`
2. Akka streams documentation, `https://doc.akka.io/docs/akka/2.5/stream/`
3. The go programming language, `https://golang.org/`
4. Docker documentation (Aug 2018), `https://docs.docker.com/`
5. Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Ferris, C., Laventman, G., Manevich, Y., et al.: Hyperledger fabric: a distributed operating system for permissioned blockchains. In: Proceedings of the Thirteenth EuroSys Conference. p. 30. ACM (2018)
6. Baars, D.: Towards self-sovereign identity using blockchain technology. Master's thesis, University of Twente (2016)
7. Bandara, E., NG, W.K., DE Zoysa, K., Fernando, N., Tharaka, S., Maurakirinathan, P., Jayasuriya, N.: Mystiko—blockchain meets big data. In: 2018 IEEE International Conference on Big Data (Big Data). pp. 3024–3032. IEEE (2018)
8. Bandara, E., NG, W.K., De Zoysa, K., Ranasinghe, N.: Aplos: Smart contracts made smart. BlockSys'2019 (2019)
9. Burns, B., Grant, B., Oppenheimer, D., Brewer, E., Wilkes, J.: Borg, omega, and kubernetes. Queue **14**(1), 70–93 (2016)
10. Buterin, V., et al.: A next-generation smart contract and decentralized application platform. white paper (2014)
11. Chen, P.W., Jiang, B.S., Wang, C.H.: Blockchain-based payment collection supervision system using pervasive bitcoin digital wallet. In: 2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob). pp. 139–146. IEEE (2017)
12. Danezis, G., Meiklejohn, S.: Centrally banked cryptocurrencies. arXiv preprint arXiv:1505.06895 (2015)
13. Davis, A.L.: Akka streams. In: Reactive Streams in Java, pp. 57–70. Springer (2019)

14. Destounis, A., Paschos, G.S., Koutsopoulos, I.: Streaming big data meets backpressure in distributed network computation. In: IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications. pp. 1–9. IEEE (2016)
15. Ebadi, Z.: Advance banking system features with emphasis on core banking. In: The 9th International Conference on Advanced Communication Technology. vol. 1, pp. 573–576. IEEE (2007)
16. Eykholt, E., Meredith, L.G., Denman, J.: Rchain architecture documentation (2017)
17. Fisher, J., Sanchez, M.H.: Authentication and verification of digital data utilizing blockchain technology (Sep 29 2016), uS Patent App. 15/083,238
18. Gupta, M.: Akka essentials. Packt Publishing Ltd (2012)
19. Hammudoglu, J., Sparreboom, J., Rauhamaa, J., Faber, J., Guerchi, L., Samiotis, I., Rao, S., Pouwelse, J.A.: Portable trust: biometric-based authentication and blockchain storage for self-sovereign identity systems. arXiv preprint arXiv:1706.03744 (2017)
20. Hewitt, C.: Actor model of computation: scalable robust information systems. arXiv preprint arXiv:1008.1459 (2010)
21. Hu, Y., Manzoor, A., Ekparinya, P., Liyanage, M., Thilakarathna, K., Jourjon, G., Seneviratne, A.: A delay-tolerant payment scheme based on the ethereum blockchain. IEEE Access **7**, 33159–33172 (2019)
22. Hughes, J.: Why functional programming matters. The computer journal **32**(2), 98–107 (1989)
23. Jones, M.B.: The emerging json-based identity protocol suite. In: W3C workshop on identity in the browser. pp. 1–3 (2011)
24. Jonsson, J., Kaliski, B.: Public-key cryptography standards (pkcs)# 1: Rsa cryptography specifications version 2.1. Tech. rep., RFC 3447, February (2003)
25. Junqueira, F.P., Reed, B.C., Serafini, M.: Zab: High-performance broadcast for primary-backup systems. In: Dependable Systems & Networks (DSN), 2011 IEEE/IFIP 41st International Conference on. pp. 245–256. IEEE (2011)
26. Karunanayake, A., De Zoysa, K., Muftic, S.: Mobile atm for developing countries (01 2008). https://doi.org/10.1145/1403007.1403014
27. Khawas, C., Shah, P.: Application of firebase in android app development-a study. International Journal of Computer Applications **179**(46), 49–53 (2018)
28. Lakshman, A., Malik, P.: Cassandra: a decentralized structured storage system. ACM SIGOPS Operating Systems Review **44**(2), 35–40 (2010)
29. Lamport, L.: The part-time parliament. ACM Transactions on Computer Systems (TOCS) **16**(2), 133–169 (1998)
30. Li, Z., Sun, Q., Lian, Y., Giusto, D.D.: An association-based graphical password design resistant to shoulder-surfing attack. In: 2005 IEEE International Conference on Multimedia and Expo. pp. 245–248. IEEE (2005)
31. Luka, M.K., Frank, I.A.: The impacts of icts on banks. Editorial Preface **3**(9) (2012)
32. Lundqvist, T., de Blanche, A., Andersson, H.R.H.: Thing-to-thing electricity micro payments using blockchain technology. In: 2017 Global Internet of Things Summit (GIoTS). pp. 1–6. IEEE (2017)
33. MBSL: Mbsl bank, `https://www.mbslbank.com`
34. McConaghy, T., Marques, R., Müller, A., De Jonghe, D., McConaghy, T., McMullen, G., Henderson, R., Bellemare, S., Granzotto, A.: Bigchaindb: a scalable blockchain database. white paper, BigChainDB (2016)

35. Merkel, D.: Docker: lightweight linux containers for consistent development and deployment. Linux journal **2014**(239), 2 (2014)
36. Mühle, A., Grüner, A., Gayvoronskaya, T., Meinel, C.: A survey on essential components of a self-sovereign identity. Computer Science Review **30**, 80–86 (2018)
37. Othman, A., Callahan, J.: The horcrux protocol: a method for decentralized biometric-based self-sovereign identity. In: 2018 International Joint Conference on Neural Networks (IJCNN). pp. 1–7. IEEE (2018)
38. Reese, W.: Nginx: the high-performance web server and reverse proxy. Linux Journal **2008**(173), 2 (2008)
39. Schmager, F., Cameron, N., Noble, J.: Gohotdraw: Evaluating the go programming language with design patterns. In: Evaluation and Usability of Programming Languages and Tools. p. 10. ACM (2010)
40. Schwiderski-Grosche, S., Knospe, H.: Secure mobile commerce. Electronics & Communication Engineering Journal **14**(5), 228–238 (2002)
41. Sidhu, J.: Syscoin: A peer-to-peer electronic cash system with blockchain-based services for e-business. In: 2017 26th international conference on computer communication and networks (ICCCN). pp. 1–6. IEEE (2017)
42. Thönes, J.: Microservices. IEEE software **32**(1), 116–116 (2015)
43. Zhang, Y., Deng, R.H., Liu, X., Zheng, D.: Blockchain based efficient and robust fair payment for outsourcing services in cloud computing. Information Sciences **462**, 262–277 (2018)