Kim and Jeong *Hum. Cent. Comput. Inf. Sci.* *(2018) 8:11*
https://doi.org/10.1186/s13673-018-0136-7

Human-centric Computing and Information Sciences

RESEARCH Open Access

Secure Authentication-Management human-centric Scheme for trusting personal resource information on mobile cloud computing with blockchain

Hyun-Woo Kim and Young-Sik Jeong*

*Correspondence:
ysjeong@dongguk.edu
Department of Multimedia Engineering, Dongguk University, Seoul 04620, Republic of Korea

Abstract

The recent advances in information technology for mobile devices have increased the work efficiency of users, the mobility of compact mobile devices, and the convenience of location independence. However, mobile devices have limited computing power and storage capacity, so mobile cloud computing is being researched to overcome these limitations in mobile devices. Mobile cloud computing is divided into two methods: the use of external cloud services and the use of mobile resource management without a cloud server (MRM), which integrates the computing and storage resources of nearby mobile devices. Because mobile devices can freely participate in MRM, it is critical to have authentication technology to determine the correctness of information regarding resources. Conventional technologies require strong authentication techniques because they have vulnerabilities that can easily be tampered with via man-in-the-middle (MITM) attacks. This paper proposes the Secure Authentication Management human-centric Scheme (SAMS) to authenticate mobile devices using blockchain for trusting resource information in the mobile devices that are participating in the MRM resource pool. The SAMS forms a blockchain based on the resource information of the subordinate client nodes around the master node in the MRM. Devices in the MRM that have not been authorized through the SAMS cannot access or falsify data. To verify the SAMS for application with MRM, it was tested for data falsification by a malicious user accessing the SAMS, and the results show that data falsification is impossible.

Keywords: Human-centric Authentication Management, Mobile cloud computing, Blockchain, Resource management for human-centric

Introduction

The performance of mobile devices is advancing with recent developments in information technology. Mobile devices, including smartphones, notebooks, netbooks, and tablets, have improved user efficiency by increasing mobility, providing the convenience of location independence, and making better use of leisure time through a variety of applications. The mobility of these devices is dependent on batteries, and batteries drain fast

when high computing is required. Mobile cloud computing is being researched to overcome the limited computing power and storage capacity of mobile devices [1–10].

Mobile cloud computing is divided into two methods: the use of external cloud services and the use of mobile resource management without a cloud server (MRM), which integrates the computing and storage resources of nearby mobile devices. The use of external cloud services is subdivided based on the development method: a service-oriented architecture, in which mobile devices depend on the Internet to connect to the cloud, and an agent-client architecture, in which mobile devices connect to the cloud through agents such as FemtoCell and Cloudlet [8, 10–17, 28].

MRM should be able to provide the trustworthy resource metadata necessary to integrate computing and storage resources because the infrastructure is composed only of mobile devices. Authentication technology is critical in determining the correctness of the resource information since mobile devices can participate freely in the MRM. Conventional authentication techniques, such as knowledge-based, possession-based, and biometric-based authentication methods, are vulnerable to data falsification via man-in-the-middle (MITM) attacks. Therefore, more powerful authentication technology is required. While 2-factor and multi-factor authentication methods are being used, they require the input of users, which is cumbersome, and they are vulnerable to exposure by shoulder surfing attacks and smudge attacks [18–33].

In this paper, the Secure Authentication Management human-centric Scheme (SAMS) is proposed, which uses blockchain to authenticate mobile devices and trust the resource information in the mobile devices participating in the MRM resource pool. The SAMS creates blocks based on the hash value of the master node and the hash value of the resource information in the subordinate client nodes in the MRM, and it then forms blockchain by connecting the hash values and blocks when the client nodes are added. Devices that have not been authenticated through the SAMS in the MRM cannot access or falsify data. This research evaluates the performance of the SAMS by applying the SAMS to the MRM and verifying the impossibility of data falsification by malicious users accessing the SAMS for human-centric aspect.

"Related works" examines the conventional authentication methods and the blockchain for determining the reliability of mobile devices. "Secure Authentication Management human-centric Scheme (SAMS)" describes the block creation process and authentication procedure for securing the mobile devices in the proposed SAMS. "Design of the SAMS" describes the design for applying the SAMS to the MRM. "Implementation of the SAMS" describes the implemented verification for the authentication of the SAMS with human-centric. "Performance evaluation" analyzes the elapsed time for the authentication of the SAMS and verifies the impossibility of falsifying resource information by artificially attempting to access and falsify data. Finally, "Conclusion" provides the conclusions and suggests future research plans.

Related works

Conventional authentication methods

Conventional authentication methods, such as knowledge-based, possession-based, and biometric-based authentication methods, are outlined in Table 1.

Table 1 Authentication methods and their advantages and disadvantages

Authentication method		Advantages	Disadvantages
Knowledge-based authentication [22]	Static knowledge-based authentication	No need for hardware Low implementation cost High user convenience	Less secure than the other authentication methods Vulnerable to various attacks, such as shoulder surfing attacks and smudge attacks
	Dynamic knowledge-based authentication	Better security than the static knowledge-based authentication Questions and answers based on the user's personal information; no need to set password	Users must memorize their own records because they will not know the questions in advance Malicious users can access via the exposed personal information of other users
Possession-based authentication [23]	Hardware type	Better security than the knowledge-based authentication method	Users must possess separate hardware, such as a One Time Password (OTP) terminal If the terminal is lost, it can lead to security threats
	Software type	Better security than the knowledge-based authentication method Higher portability and convenience compared to the hardware type of possession-based authentication	High risk of leakage because it is stored in a logical storage medium
Inherence-based authentication [22] (Biometric-based authentication)	Based on the user's physical characteristics	Authentication based on various parts of the user's body, such as face recognition, iris recognition, fingerprint recognition, vein recognition, and heart rate and ECG recognition High security High convenience	Difficult to implement and manage High cost Data loss due to physical recognition error
	Based on the user's physical behaviors	Authentication based on the recognition of the user's behaviors, including voice, typing rhythm, signature pattern, signature pressure, and user motion High security High convenience	Difficult to implement and manage High cost Vulnerable to a recorded voice Difficult to set the recognition tolerance range
Multi-factor authentication [22]		Higher security compared to single-factor authentication Reduced masquerade threat	Vulnerable to man-in-the-middle attacks Difficult to implement and manage High cost

Blockchain

A blockchain, which is a type of distributed database, uses distributed ledger technology to prevent the falsification of data records through arbitrary manipulation. Compared to a conventional centralized system, a blockchain has a number of advantages, including advantages in efficiency, security, resilience, and transparency [13, 14].

- Efficiency: A blockchain is easy to manage and can track complex data logs. Even if diverse mobile devices participate, the complex processes of system integration can be bypassed.
- Security: A blockchain has better security than centralized data management. Centralized data management faces the possibility of catastrophic damage due to hacker intrusions. Data falsification is almost impossible with a blockchain because it would require simultaneous control of all the mobile devices in which the data is distributed and then changing all the data stored in the devices.
- Resilience: A blockchain does not have a single point of failure (SPOF), as with centralized data management, because all information is shared equally among the participating mobile devices. For this reason, even if some mobile devices subject to errors or performance degradation, an MRM infrastructure with a blockchain is unlikely to receive malicious threats and can easily recover.
- Transparency: A blockchain transparently opens all the resource status and usage data by default because it shares the resource metadata with all the participating mobile devices. The exclusive occupation of resources by specific mobile devices inside the MRM infrastructure was prevented in this research.

Secure Authentication Management human-centric Scheme (SAMS)

The SAMS is an MRM infrastructure created by the resource integration of mobile devices based on a blockchain, as shown in Fig. 1. The SAMS creates and connects blocks to achieve reliability among the mobile devices in the MRM infrastructure, and it authenticates the resources of the mobile devices.

Mobile resource management without a cloud server (MRM)

MRM enables continuous use and management of mobile resources even in problematic situations, such as the failure or disconnection of the cloud server. The mobile device that plays the role of a server in the mobile cloud computing infrastructure works as a scheduler and controller for common jobs. Figure 2① shows the basic conceptual structure, and Fig. 2② explains the three steps for constructing the MRM infrastructure. MRM configuration clusters the mobile resources connected to the network, distinguishes between client and master nodes, and sets and activates an appropriate device as the master node. MRM management continuously checks the connection of the mobile resources in the mobile cloud environment, and it excludes resources that have ended their connection and it adds new resources. MRM service distributes a requested large-scale job and responds to the troubles of client or master nodes. When a large-scale job is distributed in the MRM, only the static resource information (MAC, IP, CPU, and memory) and dynamic information (MAC, IP, CPU, and memory utilization) of the mobile resources are used [30].

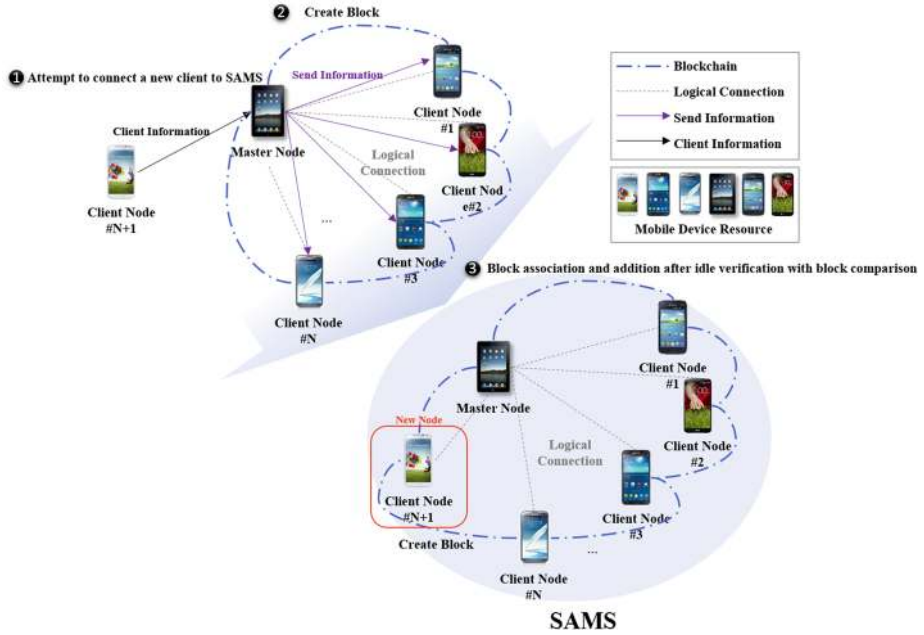

Fig. 1 Resource integration of SAMS

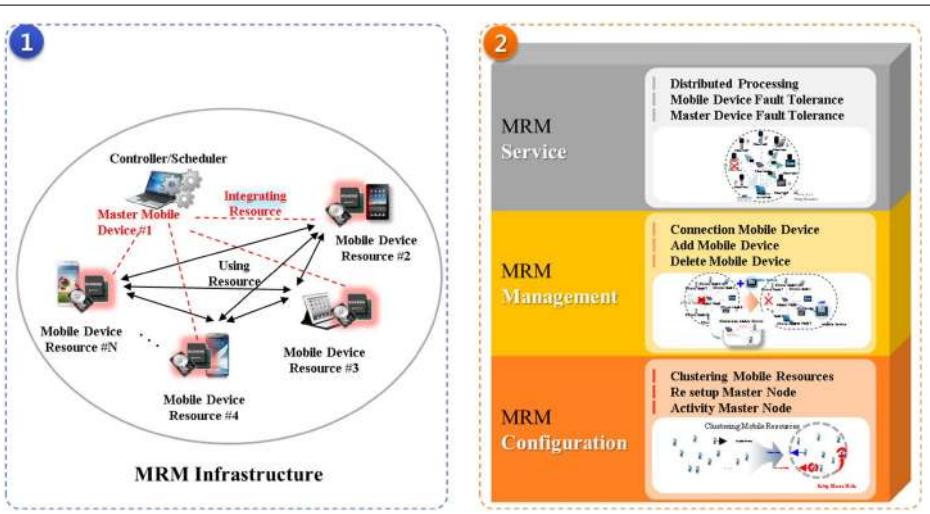

Fig. 2 Basic structure (left) and the three steps of MRM (right)

Block management

The blocks of the SAMS consist of the SAMS Block Header and the SAMS Mobile Information. The block configuration of the SAMS is shown in Table 2.

The blocks of the SAMS are created through the SAMS Mobile Information in Table 2. The first block is created by the master node alone, while the blocks added later are created by the client node. When the client node creates a block, the client nodes in the SAMS, including the master node, authenticate the block. According to the default authentication setting, the block authentication is performed by at least three mobile

Table 2 Block configuration of the SAMS

Configuration		Description
SAMS block header	Previous block hash	Hash value of the previous block
	Current block hash	Hash value using the previous block hash value and the Merkle Value
	Merkle value	Hash value using the MAC, IP, CPU, STORAGE, and MEMORY of the SAMS Mobile Information
	Time stamp	Creation time of the current block
	Next block hash	The next block hash is added when the next block is created, and the last block value is always zero
	Nonce	Disposable values used in hash functions
SAMS mobile information	MAC	MAC address of the mobile device
	IP	IP address of the mobile device
	Dynamic CPU	CPU usage (%) of the mobile device
	Static CPU	Static CPU capacity (GHZ) of the mobile device
	Dynamic storage	Storage usage (%) of the mobile device
	Static storage	Static storage size (GB) of the mobile device
	Dynamic memory	Memory usage (%) of the mobile device
	Static memory	Static memory size (GB) of the mobile device

devices in the SAMS, and the number of mobile devices performing the block authentication must be at least 51% of all the connected mobile devices. That is, if less than three mobile devices are connected, all the connected mobile devices must perform block authentication. When a client node is connected to the master node first, the mobile device authentication is performed as follows.

Step 1: The master node creates its own block and stores the block.
Step 2: When a new client node wants to connect, the client node creates a block. The client node sends its own information and the created block to the master node.
Step 3: The master node creates a block with the received client information.
Step 4: The master node determines whether the client block that it created is identical to the block received from the client node.
Step 5: If they are identical, the client block is connected to the master block.

If one or more client nodes are connected to the master node, the mobile device authentication is performed as follows.

Step 1: A new client node attempts to connect to the master node, and the client node creates a block. The client node sends its own information and the created block to all the connected mobile devices in the SAMS.
Step 3: All the connected mobile devices in the SAMS authenticate the block received as a new client.
Step 4: If 51% of all the connected mobile devices in the SAMS authenticate it, and the number of such mobile devices is at least three, the block is connected.
Step 5: When a new client is added, Steps 1 to 4 are repeated.

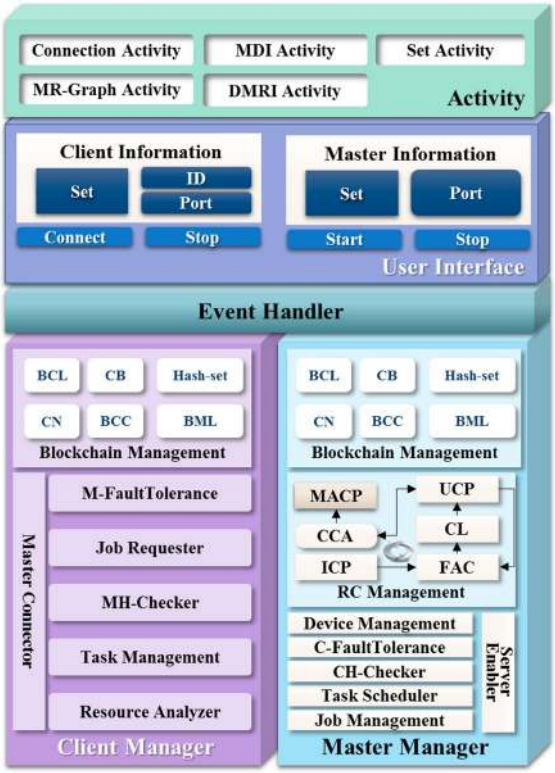

Fig. 3 SAMS architecture

Design of the SAMS

The SAMS consists of the following:

- User interface (for interacting with users)
- Master manager (to work as the master node)
- Client manager (to work as the client node)
- Event handler (to process resources in order to monitor the resource state of the client node, master node, and blockchain and deliver the resources to the activity)
- Activity (to provide the MRM operation status information in the SAMS to the users).

Figure 3 shows the full architecture of the SAMS.
User interface consists of the following:

- Set (for setting the resource permission information with client information)
- ID (for entering the IP to access the master node)
- Port (for entering the port)
- Connect (for attempting to connect to the master code)
- Stop (for disconnection).

Master information consists of the following:

- Set (for setting the maximum number of clients accepted in the MRM infrastructure)
- Port (for setting the port through which the client will gain access)
- Start (for activating the master server)
- Stop (for releasing the master server).

Master manager consists of the following:

- Server enabler (for activating the server of the master node)
- Device management (for managing the connected client nodes)
- Resource clustering (RC) Management (for hierarchical management of the resources of the client node)
- Task management (for managing the overall tasks)
- Task scheduler (for allocating tasks to clients)
- Client heartbeat (CH) Checker (for identifying the operation status of the client node)
- Client (C) fault tolerance (for responding to the failure state of the client node detected by the CH Checker).

RC management performs find adjacent client (FAC), which finds an adjacent client node from a random cluster. The client node found through the FAC is added to the cluster list (CL). When the performance of the FAC has been completed for every client node, cluster move to the new center point based on the client node added to each cluster through the Update Center Point UCP). Then cluster move to the optimal center point by repeatedly performing FAC, and the need for additional performance of FAC is determined through the check clustering availability (CCA). If no further performance is required based on the CCA, the client closest to the center point is selected as the center cluster. Furthermore, when a client is added, Blockchain Management creates a block using the corresponding client information and sends it to the connected clients, and the connected clients connect the block if the block is valid. These blocks are continuously created and connected even when the client requests computing and storage resources. Blockchain Management consists of create block (CB) for creating a block, create nonce (CN) for creating a nonce, Hash-set for setting the hash function when the information is hashed, blockchain check (BCC) for sending the created block to each client and authenticating it, and block mobile list (BML), to which clients that have been found to be malicious are added. If a block is found to be reliable by the BCC authentication, it is connected by being added to the blockchain list (BCL).

Client manager consists of the following:

- Master connecter (for connecting a client node to the master node)
- Resource analyzer (for analyzing the resource status of the client node)
- Task management (for allocating tasks from the master node and handling them)
- Master heartbeat (MH) Checker (for checking the operation status of the master node)

- M-Fault tolerance (for detecting and coping with the failure of the master node)
- Task requester (for requesting computing service from the master node).

Furthermore, the blockchain Management plays the same role as that of the master node; that is, it verifies the integrity of the block using the client information sent from the master node when a client is added to the master node.

Blockchain management consists of the following:

- Create block (CB) (for creating a block)
- Create nonce (CN) (for creating a nonce)
- Hash-set (for setting the hash function when the information is hashed)
- Blockchain check (BCC) (for sending the created block to each client and authenticating it)
- Block mobile list (BML), to which clients that have been found to be malicious are added.

If a block is found to be reliable by the BCC authentication, it is connected by being added to the blockchain list (BCL). In the case of Hash-set, the same hash function as the one in the master node must be set so that the same value can be obtained. The default hash function is SHA-2, and it can be changed to SHA-1 depending on the user setting.

Event handler sends information to the Activity to visualize the manual control of the user, the performance process, and the result through the Client Manager and Master Manager.

Activity consists of the following:

- Connection activity (for connecting the user to the master node and client node)
- Mobile device information (MDI) Activity (for visualizing the integrated resources of the mobile device and the blockchain status)
- Set activity (for setting the master node and the client node from the user)
- Mobile resource (MR) graph activity (for visualizing the mobile resource status as a graph)
- Dynamic mobile resource information (DMRI) Activity (for visualizing the dynamic changes of the mobile resources).

Implementation of the SAMS

The client connection and resource integration process for the MRM is shown in Fig. 4. When the server of the master node is activated, the master node creates its own block and waits for the client connection. The client node sets the IP and port of the master node to connect and creates its own block. The block created in the client node is sent to the master node. When the connection request is received, the master node creates a block using the received client information and sends its own information to the client node. At this time, the master node compares the block received from the client node with the block that it created to determine whether they are identical. The client node also compares the block received from the master node

Fig. 4 Client connection and resource integration process of the SAMS

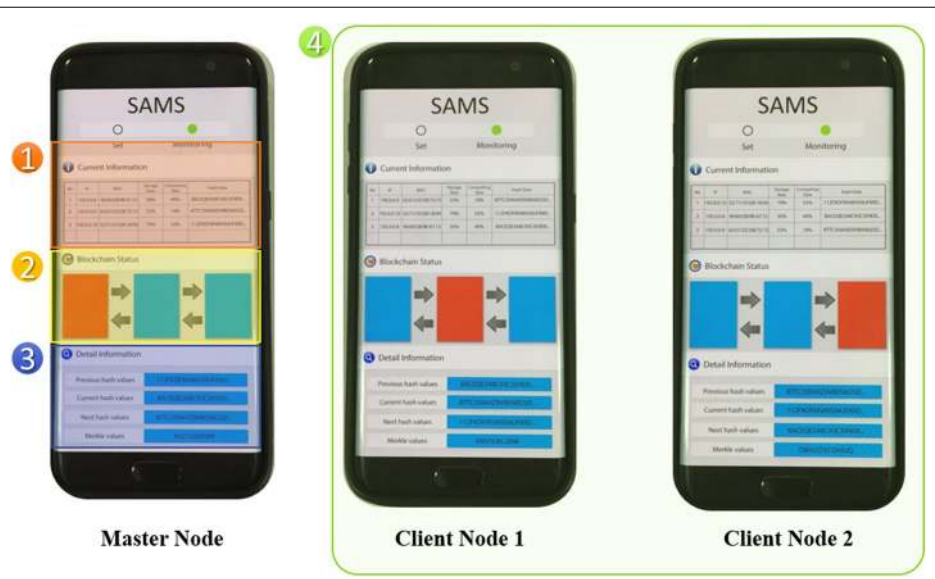

Fig. 5 Blockchain status monitoring in the SAMS

with the block that it created to determine whether they are identical. In addition, the client node creates a block based on the information received from the master node. If the two blocks are identical, the master node permits the client connection and connects the block. The client node sets the storage size in gigabyte (GB) units and the computing power in percent (%) units to be permitted by the SAMS based on the MRM. The master node integrates the permitted resource settings of the client node with the SAMS resources.

Table 3 Various types of authentication in the SAMS

Authentication type	Description
Client node does not create a block with its own information	The client node cannot connect to the blockchain because it attempts to receive authentication by sending its own information and block to all the connected mobile devices
Client node attempts to change an already connected block	Even if one client node changes all of its blockchain, it takes a long time to change the blockchain stored in the other mobile devices
Client node falsifies its own data and spreads it	The connection is impossible because the authentication conditions (at least three of all mobile devices connected in the SAMS and at least 51% of all mobile devices) have not been met

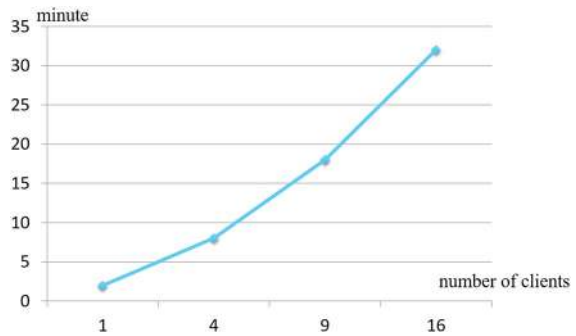

Fig. 6 Measurement of block change time in SAMS

Figure 5 shows the monitoring of the blockchain when one master node and two client nodes are connected. Figure 5① shows the status table of the mobile devices connected to the master node. The master node information shows the IP, MAC, and hash value for unique identification. In Fig. 5①, the connected client node shows the ID and MAC for unique identification, the permitted storage size, the permitted computing power, and the hash value. Figure 5② shows the connected status of the blockchain to the double-linked list, and the connected position is indicated by a red box. Figure 5③ shows the information of the block selected in ③ in detail. For the block information, the hash value connected before, the hash value of the selected block, and the Merkle tree are visualized to the user. Figure 5④ shows the monitoring of the blockchain in the client node. Access and data falsification by malicious clients is impossible due to the mutual validity check between the master node and the connected clients.

Performance evaluation

The performance evaluation of the SAMS verified that devices that have not been authorized in the MRM cannot access the SAMS infrastructure and falsify data. The results for the attempts by unauthorized users are outlined in Table 3. The data in the SAMS could not be falsified in the various authentication attempts in Table 3.

The time required to change all the blocks through the man-in-the-middle attack when 1 master node and 3 client nodes are connected to the SAMS as shown Fig. 6. It

exhibits the time taken to change all blocks created according to the number of client connections connected to the block chain. If the number of clients is less than 2, it does not take much time to change all the blocks. However, if the number of clients is 3 or more, it takes a lot of time. Computing services in MRM require computing distributed workloads of 5–10 min and require different processing times depending on workload. In this work, if the number of clients is three, the attack that changes all the blocks takes about 30 min, so even if the attack succeeds, the purpose of the attack is lost.

Conclusion

In this research, the SAMS was proposed for authenticating mobile devices using a blockchain in order to trust the resource information of the mobile devices in the MRM resource pool. The SAMS creates blocks based on the hash value of the master node in the MRM and the hash value of the resource information in the subordinate client node, and it forms blockchain by creating and connecting hash values and blocks when client nodes are added. To evaluate the performance of the SAMS, it was applied to the MRM and the connection of unauthorized devices was artificially attempted. The results confirmed that unauthorized devices cannot access and falsify data in the SAMS.

In the future, we aim to minimize the data size of blocks generated in the MRM with block chain based SAMS.

Authors' contributions
All the authors contributed equally to this work. Both authors read and approved the final manuscript.

Acknowledgements
This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2017R1D1A1A09000631).

Competing interests
The authors declare that they have no competing interets.

Availability of data and materials
Not applicable.

Ethics approval and consent to participate
Not applicable.

Funding
Not applicable.

Publisher's Note

Received: 28 March 2018 Accepted: 19 April 2018
Published online: 11 May 2018

References

1. Jeong YS, Kim HW, Jang HJ (2013) Adaptive resource management scheme for monitoring of CPS. J Supercomput 66(1):57–69
2. Gil JM, Park JH, Jeong YS (2013) Data center selection based on neuro-fuzzy inference systems in cloud computing environments. J Supercomput 66(3):1194–1214
3. Sonkar SK, Kharat MU (2015) A survey on resource management in cloud computing environment. Int J Adv Trends Comput Sci Eng 4(4):48–51
4. Kumar K, Lu YH (2010) Cloud computing for mobile users: can offloading computation save energy. IEEE Comput 43:51–56
5. Sun Z, Fox G, Gu W, Li Z (2014) A parallel clustering method combined information bottleneck theory and centroid-based clustering. J Supercomput 69(1):452–467
6. Kim HW, Byun H, Song EH, Jeong YS. Sustainable operation algorithm for high availability with integrated desktop storage based on virtualization. In: Proceedings of the 2015 international symposium on real-time natural user interface and natural user experience, Hanoi, Vietnam, 18–20; 2015, pp 123–128

7. Degefa FB, Won D (2013) Extended key management scheme for dynamic group in multi-cast communication. J Converg 4(4):7–13
8. Malkawi MI (2013) The art of software systems development: reliability, availability, maintainability, performance (RAMP). Humancentric Comput Inf Sci 3(22):1–17
9. Kim Hyun-Woo, Park JH, Majigsuren D, Jeong YS (2015) Efficient sustainable operation mechanism of distributed desktop integration storage based on virtualization with ubiquitous computing. Sustainability 7(6):7568–7580
10. Song EH, Kim HW, Jeong YS (2012) Visual monitoring system of multi-hosts behavior for trustworthiness with mobile cloud. J Inf Process Syst 8(2):347–358
11. Lee SH, Lee IY (2013) A secure index management scheme for providing data sharing in cloud storage. J Inf Process Syst 9(2):287–300
12. Shrivastava N, Kumar G (2013) A survey on cost effective multi-cloud storage in cloud computing. Int J Adv Res Comput Eng Technol 2(4):1405
13. Peck M (2016) A block chain currency that beats bitcoin on privacy. IEEE Spectr 53(12):11–13
14. Christidis K, Devetsikiotis M (2016) Blockchains and smart contracts for the internet of things. IEEE Access 4:2292–2303
15. Kim SY, Roh HC, Park CH, Park SH. Analysis of metadata server on clustered file systems. In: Proceedings of the Korea computer congress 2009, KCC, vol 36, No. 1, Seoul, Korea, 1–3; 2009, pp 64–69
16. Gaonkar PE, Bojewar S, Das JA (2013) A survey: data storage technologies. Int J Eng Sci Innov Technol 2(2):547–554
17. Gibson GA, Van Meter R (2000) Network attached storage architecture. Commun ACM 43(11):37–45
18. Dong B, Zheng Q, Tian F, Chao K, Ma R, Anane R (2012) An optimized approach for storing and accessing small files on cloud storage. J Netw Comput Appl 35(6):1847–1862
19. Chattopadhyay M, Dan P, Mazumdar S (2014) Comparison of visualization of optimal clustering using self-organizing map and growing hierarchical self-organizing map in cellular manufacturing system. Appl Soft Comput 22:528–543
20. Silva JS, Campos JC, Harrison MD (2014) Prototyping and analysing ubiquitous computing environments using multiple layers. Int J HumanComput Stud 72(5):488–506
21. Park JH, Kim HW, Jeong YS (2014) Efficiency sustainability resource visual simulator for clustered desktop virtualization based on cloud infrastructure. Sustainability 6:8079–8091
22. Nayak A, Bansode R (2016) Analysis of knowledge based authentication system using persuasive cued click points. Proc Comput Sci 79:553–560
23. Riesen K, Hanne T, Schmidt R (2016) Sketch-based user authentication with a novel string edit distance model. IEEE Trans Syst Man Cybern Syst PP(99):1–13
24. Kim HW, Park JH, Jeong YS (2016) Human-centric stroage resource mechanism for big data on cloud service architecture. J Supercomput 72(7):2437–2452
25. Duan H, Yu S, Mei M, Zhan W, Li L (2015) CSTORE: a desktop-oriented distributed public cloud storage system. Comput Elect Eng 42:60–73
26. Shafer J, Rixner S, Cox AL. The hadoop distributed filesystem: balancing portability and performance. In: Proceedings of IEEE international symposium on performance analysis of system and software (ISPASS 2010), White Plains, New York, 28–30; 2010, pp 122–133
27. Vasilyev NP, Rovnyagin MM (2014) Hybrid clusters for budget supercomputers and cloud computing. Autom Remote Control 75(10):1869–1874
28. Heo YA. A study on mobile resource management scheme based on collaborative architecture. Master Degree Thesis, Dongguk University; 2017
29. Kar J, Mishra MR (2016) Mitigating threats and security metrics in cloud computing. J Inf Process Syst 12(2):226–233
30. Motavaselalhagh F, Esfahani FS, Arabnia HR (2015) Knowledge-based adaptable scheduler for SaaS providers in cloud computing. Humancentric Comput Inf Sci 5(14):1–19
31. Kim S, Lee H, Kwon H, Lee S (2015) Evaluation model of defense information systems use. J Converg 6(3):18–26
32. Kang WM, Moon SY, Park JH (2017) An enhanced security framework for home appliances in smart home. Humancentric Comput Inf Sci 7(6):1–12
33. Suryani V, Sulistyo S, Widyawan W (2017) Internet of things (IoT) framework for granting trust among objects. J Inf Process Syst 13(6):1613–1627