Cluster Comput (2019) 22:S1985–S1990
https://doi.org/10.1007/s10586-017-1107-x

Verifying a secure authentication protocol for IoT medical devices

Woo-Sik Bae[1]

Received: 10 April 2017 / Revised: 19 July 2017 / Accepted: 9 August 2017 / Published online: 30 August 2017

Abstract The advancement of Internet of Things (IoT) technology has made medical equipment smaller and smarter, while computing environment has shifted from server-client wire/wireless communication networks toward diverse portable laptops, smartphones, tablet PCs, and PDAs. The communication between smaller IoT devices has added to the accuracy and convenience of distance healthcare services. However, security issues in communication sessions resulting from the leakage of personal medical information, infringements of privacy and improper management of medical information are looming large. Since personal medical information is transmitted between wire/wireless devices, the threats to secure distance medical service could be detrimental to further advancement of IoT in healthcare. Hence, this paper proposed a method of addressing the vulnerabilities to a range of attacks in the communication between medical devices. The proposed IoT-based communication protocol used random numbers and session keys to transmit hashed and encrypted data, and underwent a formal verification, where the transmitted data remained intact against data extraction and other attacks.

Keywords IoT authentication · Protocols · Radiofrequency identification · IoT security · Security protocol · Authentication protocol · Internet of Things

1 Introduction

Conventional server-client wire network computing environment has increasingly been replaced by a wide range of portable small devices such as laptops, smartphones, tablet PCs and PDAs. Yet, the very diversity and portability as well as the ease-of-use of such devices have increased the risk of information being extracted, copied and leaked. In comparison to wire communication, wireless communication is vulnerable to data wiretapping, forgery and alteration, which justifies the need to take some countermeasures to prevent data from any leakage [1,2]. Internet of Things (IoT) devices are mostly portable and small, which adds to the hardware constraints on security measures [3,4]. Therefore, relatively complex software-based encryption systems take up much CPU and memory space, imposing additional burdens on devices. Due to the constraints inherent in most of IoT devices, they have to adopt simple security modules or less sophisticated security systems [5,6]. However, expanding IoT technology and its growing user bases require security systems safe against attacks, e.g. hacking, on security loopholes. In healthcare, IoT technology has made simple medical services available anytime anywhere for the benefit of patients whose conditions need be continuously monitored or those in remote regions where doctors are less accessible [7–9]. Unfortunately, however, the wireless sessions in the communication between IoT devices are vulnerable to attacks from intruders, who might tamper with patients' information, causing serious challenges against personal healthcare information and medical consultation. The present paper proposed a security protocol to address the security vulnerabilities in medical IoT communication. The proposed software-based communication protocol used inter-device cross-authentication and encryption to deter diverse attacks, and was verified with Casper/FDR [10,11]

✉ Woo-Sik Bae
drbws@daum.net

[1] Department of AIS Center, Ajou Motor College, Boryeong, Chungnam, Korea

widely used for the formal verification of processes. Taken together, the proposed protocol proved itself to ensure the security and safety of wireless communication between medical devices.

This paper covers the following chapters. Section 2 describes relevant research on IoT healthcare service and CASPER/FDR(Compile for the Analysis of Security Protocols/Failure Divergence Refinements). Section 3 proposes a medical IoT authentication protocol, which is in turn tested with Casper/FDR verification tool. Section 4 discusses the safety of the verified proposed protocol. Finally, Sect. 5 presents the conclusion.

2 Literature review

2.1 IoT healthcare

IoT technology for healthcare service has been widely explored as an alternative to enable patients having difficulties in activities of daily living to see doctors from home or work, or in transit. A broad range of sensors and terminals are used to screen and measure patients' health conditions before sending their health records to healthcare centers or medical equipment systems. Subsequently, the health records are analyzed by doctors and/or other medical staff to provide feedback for patients. In the process, patients engage in the distance consultation with doctors via wire/wireless communication involving biometrics and video consultation. The accessibility of service based on the analysis of the patterns manifest in collected patients' information will significantly contribute to the healthcare sector in the foreseeable future. Thus, the security of relevant systems should be verified, in that patients' personal medical information is crucial for their life and privacy [12–16].

2.2 Requirements for secure IoT communication in healthcare

Secure IoT communication in healthcare requires the following, which are comparable to wireless communication security [12,13,15].

(1) *New ownership privacy* Once the ownership of a tag is transferred to a new owner, only the new owner can identify the tag and access the information in the tag. The old owner cannot access the tag the moment the tag ownership is transferred.
(2) *Old ownership privacy* Once the ownership of a tag is transferred to a new owner, the new owner cannot trace the old owner's history of tag use.
(3) *Restoration of authority* In such a case as the exchange of tagged products, the current owner needs to temporarily transfer the ownership to the old owner so that the latter can access the information in the tag.
(4) *Safety against denial-of-service attacks* The public key authentication of a tag sends diverse requests to the authentication protocol on the server, using up the memory space and system resources, which slows down or halts the service. Prior to accessing the server, the tag should always support the resources for the authentication protocol, while the server should allot its resources after checking if the tag has been authenticated.
(5) *Safety against replay attacks* It is necessary to deter intruders from eavesdropping on the messages exchanged between readers and tags and from deceiving the readers or tags using the eavesdropped messages.
(6) *Safety against man-in-the-middle attacks* It is necessary to deter intruders from using fake messages or altered messages between tags and readers to do what they want.
(7) *Secrecy* The secrecy of data exchanged between communication devices in wireless IoT communication should be maintained even on unauthenticated devices.
(8) *Anonymity and privacy* Any failure to meet the anonymity in IoT communication will lead to the risk of infringements of privacy. In case intruders extract personal health information, serious issues may arise including the leakage of medical information.

2.3 CASPER/FDR

CASPER/FDR is a compiler developed to represent the sequence protocol in communication sequential process (CSP). Casper is a highly complicated specification method to the protocol designers, who are not adept at the formal design of CSP-based specification of the process, which is prone to errors and mistakes in design and analysis. Cssper is a program developed to facilitate the design of the transmission in security protocols and to simplify the complicated specification.

As for the method of specification, #Actual variables, #Intruder Information, #System, #Processes, #Free variables, #Protocol description and #Specification are specified so that the program will convert them into a CSP document. #Free variables is a function that defines the variables and function types used for running the protocol. #Processes indicates the protocol parameters and defines a certain image of the function. #Protocol description defines the order of messages in the protocol. The method of marking is comparable to the stepwise marking of the protocol. #Specification specifies the requirements of the protocol, where the lines starting with Agreement are the specification of authentication, which means A has been exactly authenticated to B and both agents have agreed on the data values, na and nb. #Specification defines the variable types to be used on the actual system in a similar manner to that used to define the free variables. Then, three agents and three nonces are basically specified per system, with the agents' public keys and private keys being defined under the Functions. Upon the completion of specification, the CSP document converted by Casper is verified with the FDR program in terms of whether it meets the security and authentication attributes. Here, FDR checks the safety verification, deadlock verification and livelock verification, and displays any likely attack scenarios in case any security vulnerabilities are detected, making it easy to analyze and rectify the loopholes.

Table 1 Symbols and definition

Symbols	Definition
Reader_BOB	Reader_Agent
Tag_ALICE	Tag_Agent
DBS	Database Server
H(x)	Hash Function
sek_a, sek_b	SessionKey
nx1, nk2	Nonce

3 Proposed safe protocol for IoT healthcare

The proposed protocol is designed for wireless communication to exchange the information in the communication sessions between implantable medical devices among other IoT equipment and other tagged devices using the communication method of readers. Given the inter-device wireless communication session is exposed to various threats against security, the present paper proposed a protocol intended to provide a safe communication environment against any intruder's hacking. Nonce, session keys and hash function were used to design the proposed security protocol.

Table 1 shows the symbols and their meanings used in the proposed security protocol for communication between IoT medical devices.

3.1 Casper specification

Figure 1 partially shows the Casper specification code for the proposed protocol for the verification of the wireless communication between implantable medical devices. #Free variables defines the integral variables and function types. InverseKeys = (nk2, nk2), (sek_a, sek_a), (sek_b, sek_b), (nx1, nx1), (Tag_ALICE, Tag_ALICE), (Reader_BOB,

Fig. 1 Casper specification in the protocol

```
    #Free variables

    Tag_ALICE , Reader_Reader_BOB : Agent
    DBS : Database Server
    nx1,nk2 : Nonce
    H() : HashFunction
    sek_a, sek_b : SessionKey
    InverseKeys = (nk2, nk2),(sek_a,sek_a),(sek_b,sek_b),( nx1, nx1),(Tag_ALICE,Tag_ALICE),
(Reader_BOB,Reader_BOB)

    #Processes

    INITIATOR(Tag_ALICE, Reader_BOB, DBS, nx1, sek_a)
    RESPONDER(Reader_BOB, DBS, nk2, sek_b)
    SERVER(DBS, Tag_ALICE, Reader_BOB, sek_a, sek_b)

    #Protocol description

    0.      -> Reader_BOB : Tag_ALICE
    1. Tag_ALICE -> Reader_BOB : H( nx1){ nx1}{sek_a}%enc1,H(Reader_BOB)
    2. Reader_BOB -> SBS : {enc1%{ nx1}{sek_a},H(Reader_BOB),sek_b,k}{sek_b}H( nx1)
    3. SBS -> Reader_BOB : {Tag_ALICE, nx1,{
nk2}{sek_a}%enc2}{sek_b},H(SBS,Tag_ALICE)(+)H(Reader_BOB)
    4. Reader_BOB -> Tag_ALICE : enc2%{nk2}{sek_a},h( nx1){ nx1}{nk2}
    5. Tag_ALICE -> Reader_BOB : H(Tag_ALICE),{ nx1}{sek_a}%enc3

    #Actual variables
    TAG_ALICE, READER_BOB, Mallory : Agent
    DBS : DATABASE SERVER
    NX1,NK2 : Nonce
    SEK_A,SEK_B : SessionKey
    InverseKeys = ( NX1, NX1),(NK2,NK2),(M,M),(SEK_A,SEK_A),(SEK_B ,SEK_B ),
    (TAG_ALICE,TAG_ALICE),(READER_BOB,READER_BOB),(Mallory,Mallory)
```

Reader_BOB) means each agent and function return their inverse keys. #Protocol description defines the sequential order of computation transmitted in the protocol. The integers 0, 1 and 2 indicate the steps of the messages transmitted.

3.2 Operation

The proposed protocol for implantable devices operates in the following order and manner.

◎ (Step ① : Tag_Tag_ALICE → Reader_BOB)

Tag_ALICE receives a Query from Reader_BOB, generates { nx1}{sek_a}%enc1 from a hash operation with a Nonce x, and concatenates it with the hashed value. Then, Tag_ALICE saves the value in the variable %enc, performs a hash operation for Reader_BOB, and concatenates each value. Tag_ALICE transmits the computed H(nx1){ nx1}{sek_a}%enc1,H(Reader_BOB) to Reader_BOB. Here, the generated value is the only value computed with the hash operation that cannot be generated by another Tag_ALICE. All the data transmitted is not encrypted but mixed to prevent the attribute of each data transmitted from being used for attacks. The hashed value is computed as follows: $h_a(\mathrm{nx1}) = h_{int}\left(\left(\sum_{i=0}^{\mathrm{nk2}} x_i \cdot a^i\right) \bmod p\right)$. The hash data value in the hash operation involves hashing the fixed-length data.

For the initial vector hash function, an integer 2w applied to $\overline{a} = (a_0, \ldots, a_k)$ yields $h_{\overline{a}}(\overline{\mathrm{nx1}})^{strong} = \left(a_0 \sum_{i=0}^{\mathrm{nk2}} a_{i+1}x_i \bmod 2^{2w}\right) \div 2^w$, which is in turn applied to the string, or the transmitted data value to get $h_a(\overline{\mathrm{nx1}}) = h_{int}\left(\left(\sum_{i=0}^{\mathrm{nk2}} x_i \cdot a^i\right) \bmod p\right)$, where $a \in [p]$ is uniformly random and $h_{int}$is chosen randomly from a universal family mapping integer domain [p] → [m].

◎ (Step ② : Reader_BOB → DBS)

Together with the value of H(nx1){nx1}{sek_a}%enc1,H (Reader_BOB) received from Tag_ALICE, Reader_BOB uses his(Reader_BOB's) own {nx1}{sek_a},H(Reader _BOB),sek_b,k}{sek_b}H(nx1) value to get the following. {enc1%{nx1}{sek_a},H(Reader_BOB),sek_b,nk2}{sek_b} H(nx1) uses H(nx1){ nx1}{sek_a}%enc1,H(Reader_BOB) data from Tag_ALICE for operation and concatenation. Then, Reader_BOB checks the received data and saves it in the variable enc1%. Once {enc1%{nx1}{sek_a},H(Reader _BOB),sek_b,nk2}{sek_b}H(nx1) data to be normally transmitted is generated, Reader_BOB transmits it to DBS.

◎ (Step ③ : DBS → Reader_BOB)

The Database Server(DBS) checks the value of {enc1% {nx1}{sek_a},H(Reader_BOB),sek_b,nk2}{sek_b}H(nx1) received from Reader_BOB, completes its own authentication for the cross-authentication, and checks the value of Tag_ALICE with reference to the value transmitted by Reader_BOB. Then, the server generates the session keys (sek_a and sek_b), and performs the hash operation on it self(DBS), Tag_ALICE and Reader_BOB. Finally, the server performs the exclusive OR operation to compute the value of {Tag_ALICE, nx1,{nk2}{sek_a}%enc2}{sek_b},H(DBS, Tag_ALICE)(+)H(Reader_BOB), and transmits it to Reader _BOB.

◎ (Step ④ : Reader_BOB → Tag_ALICE)

Reader_BOB checks the value of {Tag_ALICE, nx1, {nk2}{sek_a}%enc2}{sek_b},H(SBS,Tag_ALICE)(+)H (Reader_BOB) received from the Database Server (DBS), and performs his(Reader_BOB's) authentication for the cross-authentication. Then, Reader_BOB checks its value of enc2% saved, performs an operation to generate enc2%{nk2} {sek_a},{nx1}{nk2}, computes the hashed value of $h_a(\mathrm{nx1}) = h_{int}\left(\left(\sum_{i=0}^{\mathrm{nk2}} x_i \cdot a^i\right) \bmod p\right)$ and { nx1}{nk2}, and concatenates each data to generate the value of enc2%{nk2}{sek _a},h(nx1){nx1}{nk2}. Finally, Reader_BOB transmits the generated value of enc2%{nk2}{sek_a},h(nx1){nx1}{nk2} to Tag_ALICE.

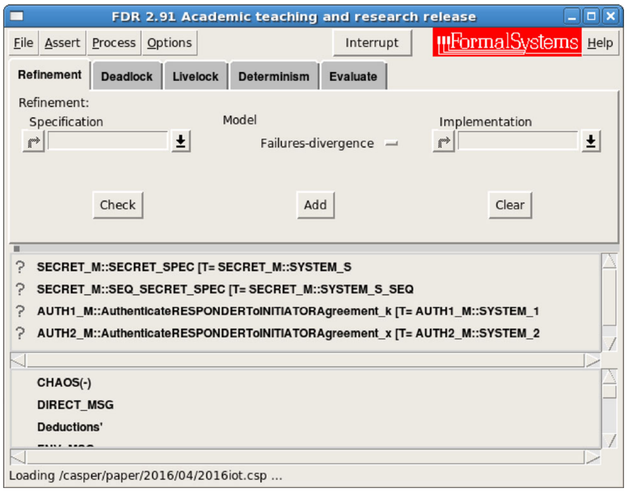
◎ (Step ⑤ : Tag_ALICE →Reader_BOB)

Lastly, Tag_ALICE receives from Reader_BOB the value of enc2%{nk2}{sek_a},$h_a(\mathrm{nx1}) = h_{int}\left(\left(\sum_{i=0}^{\mathrm{nk2}} x_i \cdot a^i\right) \bmod p\right)${nx1}{nk2}, and compares it with the value she(Tag _ALICE) has. Upon confirming the two values, Tag_ALICE performs an operation to get $h_a(Tag\_ALICE) = h_{int}\left(\left(\sum_{i=0}^{\mathrm{nk2}} x_i \cdot a^i\right) \bmod p\right)$,{nx1}{sek_a}%enc3, and transmits it to Reader_BOB, completing her(Tag_ALICE's) authentication session. Reader_BOB transmits the value of $h_a(Tag\_ALICE) = h_{int}\left(\left(\sum_{i=0}^{\mathrm{nk2}} x_i \cdot a^i\right) \bmod p\right)$,{nx1} {sek_a}%enc3 received from Tag_ALICE to DBS. Then, DBS retrieves the value of Tag_ALICE saved earlier and performs its(DBS's) authentication. Once the hash code and Tag_ALICE code are confirmed with a normal authentication, the process continues.

4 Test results

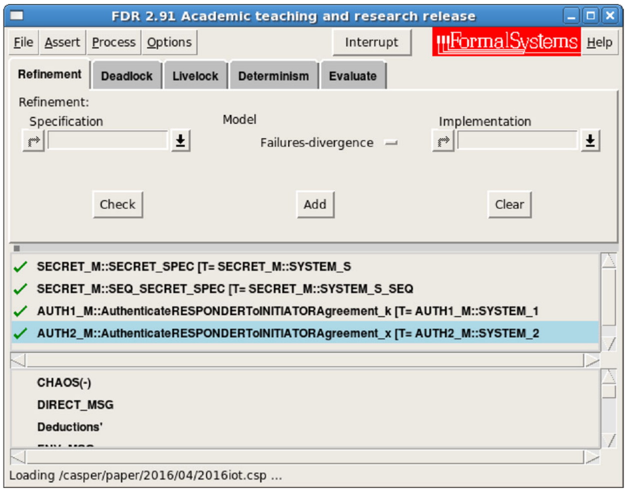
The proposed protocol for medical information transmission and communication was verified in terms of safety, livelock and deadlock with the CASPER/FDR model verification program.

In Fig. 2, the CASPER program successfully converts the proposed protocol into the source codes for verification and loads the protocol to verify its security with the FDR program. ? on the left side indicates the verification has not been proceeded with.



```
? SECRET_M::SECRET_SPEC [T= SECRET_M::SYSTEM_S
? SECRET_M::SEQ_SECRET_SPEC [T= SECRET_M::SYSTEM_S_SEQ
? AUTH1_M::AuthenticateRESPONDERToINITIATORAgreement_k [T= AUTH1_M::SYSTEM_1
? AUTH2_M::AuthenticateRESPONDERToINITIATORAgreement_x [T= AUTH2_M::SYSTEM_2

CHAOS(-)
DIRECT_MSG
Deductions'

Loading /casper/paper/2016/04/2016iot.csp ...
```

Fig. 2 Proposed protocol ready for verification



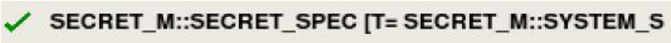
```
✓ SECRET_M::SECRET_SPEC [T= SECRET_M::SYSTEM_S
✓ SECRET_M::SEQ_SECRET_SPEC [T= SECRET_M::SYSTEM_S_SEQ
✓ AUTH1_M::AuthenticateRESPONDERToINITIATORAgreement_k [T= AUTH1_M::SYSTEM_1
✓ AUTH2_M::AuthenticateRESPONDERToINITIATORAgreement_x [T= AUTH2_M::SYSTEM_2

CHAOS(-)
DIRECT_MSG
Deductions'

Loading /casper/paper/2016/04/2016iot.csp ...
```

Fig. 3 Verified medical IoT protocol

In Fig. 3, the designed source file is loaded and run, upon the completion of basic grammar and process checks. The process and security of the proposed IoT protocol was verified with the program. As shown in the Figure, all attributes passed the verification. The verification program displays X if it detects any security vulnerability, runs Debug to identify the issue, and rectifies the identified loophole prior to resuming the verification.

Figure 3 shows 4 facets of the verification, which are discussed below.

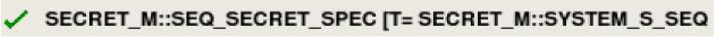
1)



✓ SECRET_M::SECRET_SPEC [T= SECRET_M::SYSTEM_S

The security and performance of the proposed protocol for IoT medical communication were verified. The tick marks before the messages show the protocol proves itself to be safe against various attacks and secure enough not to be exposed to intruders. Also, the security of session keys and inter-agent communication against various attacks was verified. Thus, the proposed protocol proved itself to be safe as shown in the Figure.

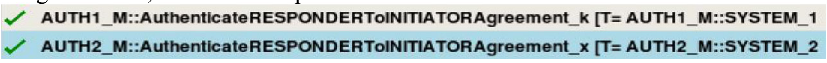
2)



✓ SECRET_M::SEQ_SECRET_SPEC [T= SECRET_M::SYSTEM_S_SEQ

This verifies whether the proposed IoT protocol seamlessly works in a stepwise manner. As shown in the Figure, the proposed protocol proves itself to be safe in each step against a range of errors, attacks and exposures.

3,4)



✓ AUTH1_M::AuthenticateRESPONDERToINITIATORAgreement_k [T= AUTH1_M::SYSTEM_1
✓ AUTH2_M::AuthenticateRESPONDERToINITIATORAgreement_x [T= AUTH2_M::SYSTEM_2

3) and 4) verify whether the Responder and Initiator can perform the cross-authentication without encountering any security challenges via k. The agents perform the safe authentication with each other in the proposed protocol.

5 Conclusion

Medical equipment has substantially developed with the advancement of IoT technology. IoT medical devices process personal health information and get involved in wireless communication with one another, where protecting personal medical information and privacy is an overarching point because any manipulation or leakage of patients' sensitive medical information could result in very serious issues on the system.

As a means of addressing the security challenges for privacy protection in IoT, security protocols used for safe communication sessions have been widely explored. The present paper proposes a protocol design based on hash function, nonce and session keys to ensure safe IoT communication between medical devices. The proposed IoT protocol proved itself to be safe in all aspects, which was verified with the formal verification tool, FDR program, and safely ended without falling into memory errors and infinite loops. The proposed protocol should be noted on two grounds. First, it is possible to address the vulnerabilities of IoT protocols, which benefits the safety and efficiency of communication in comparison to complicated operations. Second, the formal verification tool reduces mistakes and errors in designing the security protocol and ensures effective verification. Future research will include different functions and operations for the safe and efficient authentication of the sensors used in military, finance and luxury items.

References

1. Ashraf, Q.M., Habaebi, M.H.: Autonomic schemes for threat mitigation in Internet of Things. J. Netw. Comput. Appl. **49**, 112–127 (2015)
2. Aljawarneh, S., Yassein, M.B.: A resource-efficient encryption algorithm for multimedia big data. Multimed. Tools Appl., pp. 1–22 (2017)
3. Rehiman, K.R., Veni, S.: A secure authentication infrastructure for IoT Enabled smart mobile devices: an initial prototype. Indian J. Sci. Technol. **9**(9) (2016)
4. Mahmoud, R., et al.: Internet of things (IoT) security: current status, challenges and prospective measures. Internet Technology and Secured Transactions (ICITST). In: 10th International Conference for IEEE, pp. 336–341 (2015)
5. Kang, A.N., Barolli, L., Park, J.H., Jeong, Y.S.: A strengthening plan for enterprise information security based on cloud computing. Clust. Comput. **17**(3), 703–710 (2014)
6. Bae, W.S.: Function-based connection protocol development and verification for secure communication in vehicle environment. Clust. Comput. **18**(2), 761–769 (2015)
7. Sicari, S.: Security, privacy and trust in Internet of Things: the road ahead. Comput. Netw. **76**, 146–164 (2015)
8. Bamasag, O.O., Youcef-Toumi, K.: Towards continuous authentication in internet of things based on secret sharing scheme. In: Proceedings of the WESS'15: Workshop on Embedded Systems Security, ACM (2015)
9. Park, R.C., Jung, H., Shin, D.K., Kim, G.J., Yoon, Kun-Ho: M2M-based smart health service for human UI/UX using motion recognition. Clust. Comput. **18**(1), 221–232 (2015)
10. Lowe, G.: Casper: a compiler for the analysis of security protocols. User Manual and Tutorial. Version 1.12 (2009)
11. Formal Systems (Europe) Ltd and Oxford University Computing Laboratory: failures-divergence renement—FDR2 User Manual (2010)
12. Gao, Y., Liu, W.: BeTrust: a dynamic trust model based on bayesian inference and tsallis entropy for medical sensor networks. J. Sens. **2014**, 1–10 (2014)
13. Kritika, E., et al.: Multivariate authentication and encryption scheme for data privacy in IoT healthcare monitoring. Imp. J. Interdiscip. Res. **2**(8), 543–550 (2016)
14. Han, K.H., Bae, W.S.: Proposing and verifying a security-enhanced protocol for IoT-based communication for medical devices. Clust. Comput. **19**(4), 2335–2341 (2016)
15. Jung, E.Y., Kim, J., Chung, K.Y., Dong, K.P.: Mobile healthcare application with EMR interoperability for diabetes patients. Clust. Comput. **17**(3), 871–880 (2014)
16. Park, R.C., Jung, H., Shin, D.K., Kim, G.J., Yoon, Kun-Ho: M2M-based smart health service for human UI/UX using motion recognition. Clust. Comput. **18**(1), 221–232 (2015)

Woo-Sik Bae received his Ph.D. degree in Computer Education from Chungbuk National University, Korea in 2012. He has published more than 54 papers on IoT security in international and Korean journals and conferences. His research interest includes: VANET, computer and network security, IoT security, authentication protocol, and Convergence. He steering committees member of the International Conference Convergence Technology (ICCT), International Conference on Digital Policy and Management (ICDPM), and International Conference for Small and Medium Business (ICSMB). He is a member of KCS and SDPM.