Peer-to-Peer Netw. Appl. (2016) 9:539–545
DOI 10.1007/s12083-015-0396-1

# Designing and verifying a P2P service security protocol in M2M environment

**Woo-Sik Bae**[1]

Received: 16 March 2015 / Accepted: 14 July 2015 / Published online: 26 July 2015


**Abstract** Multifunctional high-performance electronic systems in M2M(Machine-to-Machine) industry have been evolving substantially in tandem with the advancement of IT. M2M, standing for machine-to-machine communication, replaces people in cases where human intervention is hardly viable or in such fields as weather, environment or disasters where long-term monitoring is required. Yet, due to the nature of M2M devices involving wireless communication, they are exposed to intruders' attacks. Thus, the overriding concern in M2M communication is mutual authentication and security. In this context, security communication protocols are considered worth exploring. This paper concerns designing a safe communication protocol by applying hash locks, random numbers and session keys. Instead of arguing for the security of the protocol based on mathematical theorem proving as most previous studies did, the present paper demonstrates the proposed protocol is safe against a variety of intruders' attacks by formally verifying it using Casper/FDR. In short, the proposed protocol is verified in terms of safety, deadlock and livelock.

**Keywords** P2P · Security protocol · Authentication protocol · RFID security · Casper/FDR · M2M authentication · Model checking

✉ Woo-Sik Bae
drbws@daum.net

[1] Department of AIS Center, Ajou Motor College, 106, Daehak Gil, Jupo-Myeon, Boryeong-Si, Chungnam 355-769, Korea

## 1 Introduction

M2M, or machine-to-machine, communication has been drawing much academic attention in recent years. M2M communication operates automatically without human intervention, being used in the fields where human intervention is not viable or where works are simple and repetitive as well as in dangerous industries. M2M communication systems transmit and exchange information, and thus can be used and applied across industries for fault diagnosis, repairing, monitoring and data collection [1–5]. M2M devices rely on wire or optical communication. Yet, in some sections where wireless communication is employed, M2M devices are vulnerable to every kind of attacks from intruders. In general, intruders can eavesdrop on, arbitrarily alter and send or delete data being transmitted, causing privacy issues [6, 7]. This indicates many risks are inherent in system operation. Hence, many researchers are committed to exploring M2M communication network protocols. Still, most of previous studies failed to go further than the mathematical theorem proving, and their proposed protocols revealed initially unanticipated vulnerabilities [8–14]. Therefore, it is time-consuming to apply their protocols to systems in practice, which warrants further studies. The present paper makes use of formal methods to address many issues found in previous studies [15–17]. Specifically, this paper verifies the proposed protocol by performing a formal verification involving the process of logical expression and proving if specified content meets security requirements. This paper draws on Casper and FDR, both of which are widely recognized as techniques for testing models, to verify the security of the proposed protocol. The following chapters constitute this paper. Chapter 2 discusses M2M communication based on relevant literature. Chapter 3 specifies the proposed protocol and describes its logic. Chapter 4 verifies the security of the proposed protocol. Finally, chapter 5 presents the conclusion.

## 2 Literature review

### 2.1 M2M communication architecture and its security threats

M2M communication technology involves equipping diverse systems and devices with wire and wireless communication modules to extend the communication, broadcasting and internet infrastructures from human-to-human to human-to-machine and machine-to-machine approaches. M2M communication technology collects, processes and exchanges information without human intervention through machine-to-machine communication, with its applications expanding owing to the advancement of RFID/USN technology.

Figure 1 shows the underlying functional architecture of oneM2M, comprised of Application Entity, Common Services Entity and Network Services Entity [18, 19].

Here, the M2M communication security requirements are defined as below:

1) Authentication: Each entity engaging in M2M communication should identify itself as a legitimate owner. To this end, every entity should have a unique ID.
2) Confidentiality: Confidentiality should be maintained so that intruders cannot access data being transmitted.
3) Anonymity: Lack of anonymity in M2M communication leads to the risk of intrusion on privacy. Thus, unless privacy protection measures are provided, any exposure of information to intruders will cause serious problems. Any actual information in M2M environment should not be exposed to intruders and other devices.
4) Non-repudiation: Any entity sending data should not repudiate, by proving, the fact that it has sent the data. As a rule, digital signatures are used for non-repudiation in M2M communication.
5) Data trust: Data trust is the required trust in data being transmitted in respect of any falsification and alteration of sequence.
6) Privacy: Privacy is required to prevent ID, video data and location information from being exposed to intruders. Privacy protection need be disabled at the request of government agencies for criminal investigation or in case of incidents arising in system monitoring. Therefore, conditional privacy need be supported in M2M communication.
7) Unlinkability: If any information is recursively used in M2M communication, locations of particular devices may be exposed. Thus, unlinkability should be provided so that devices cannot be identified.
8) Traceability: In case of crimes and incidents arising, relevant information should be traceable, whilst trace information should be thoroughly secured [20–22].

### 2.2 Casper/FDR

Casper (Compiler for the Analysis of Security Protocols) [23, 24] is a compiler developed by Gavin Lowe with a view to sequential representation of security attributes and easy specification of protocols. Casper specifies the operation of protocols and the system to be verified in two parts. First, it defines messages, data types, functions, variables and operating orders exchanged between hosts. Second, it defines the real system to verify each host's roles, function statements and attackers' status information.

In Casper, specification and classification are made into eight specific items with their header part that starts with # and they are as follows.

- Free variables: Defines the agents, variables, and functions in the protocol
- Process : Represents each agent as a process
- Protocol description: Shows all the messages exchanged between the agents

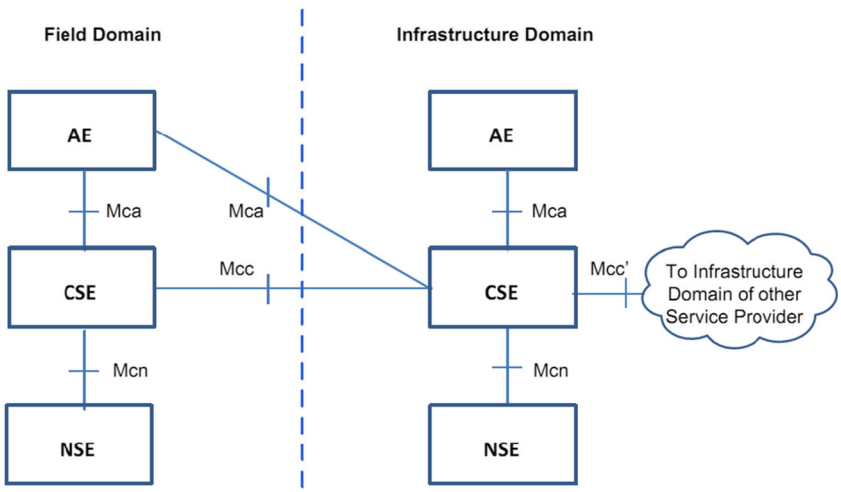


**Fig. 1** oneM2M functional architecture

- Specification: Specifies the security properties to be checked
- Actual Variables: Defines the real variables, in the actual system to be checked
- Functions: Defines all the functions used in the protocol
- System: Lists the agents participating in the actual system with their parameters instantiated
- Intruder Information: Specifies the intruder's knowledge and capabilities

FDR(Failure Divergence Refinements) is a model checker that receives CSP(Communicating Sequential Processes) [25, 26] as the input language and tests if model properties are met. Unless model properties are met, FDR shows counter examples so that security vulnerabilities can be analyzed easily. It supports 3 verification methods, i.e. safety verification, deadlock verification and live-lock verification. It checks whether the requirements of security protocols, i.e. confidentiality, integrity, authentication and non-repudiation properties, are met and supports trace models, failure models and failure/divergence models [27].

A) Trace model

A process is represented as a finite order set by its actions. When P process includes all actions of Q process, it is marked as $P \subseteq TQ$

$$P \subseteq TQ \triangleq traces(Q) \subseteq traces(P)$$

B) Failure model

A failure is a set of (s, X), where s refers to the trace in P, while X means a set of all events that the process refuses after s. That is, it means a deadlock, which is marked as below.

$$P \subseteq TQ \triangleq failures(Q) \subseteq failures(P)$$

C) Failure/divergence model

The divergence of a process means a livelock. In other words, the failure/divergence model involves both a deadlock and a livelock, which are marked as below.

$$P \subseteq FTQ \triangleq failures(Q) \subseteq failures(P) \wedge divergences(Q) \subseteq divergences(P)$$

## 3 Proposed prorocol

In M2M communication, devices automatically communicate with each other to take appropriate actions or measures, and send and receive information using wire and wireless communication where human intervention is unavailable. This paper employs TimeStamp values varying per session, hash function operations and hidden secret keys for communication, whilst using security agents. Also, the proposed protocol is designed based on SessionKeys and PublicKeys.

### 3.1 Casper specification

Figure 2 is the Casper specification code of the proposed protocol. 3 important domains used in security protocols are listed, i.e. declaration of variable types, operation procedures and attacker models. Variables and function types are defined under #Free

**Fig. 2** Casper specification in the protocol

```
#Free variables

Alice, Bob : Agent
DB : DatabaseServer
pkdb : PublicKey
skdb : SecretKey
Secret : Agent -> ServerSecret
SkeyBob, SkeyAlice : SessionKey
Ta, Tb : TimeStamp
H : HashFunction
InverseKeys = (pkdb,skdb),(SkeyBob,SkeyBob),(SkeyAlice,SkeyAlice)
#Processes

INITIATOR(Bob, DB, pkdb,SkeyBob)knows Secret(Bob)
RESPONDER(Alice, Bob, DB, pkdb,SkeyAlice)knows Secret(Alice)
SERVER(DB, skdb) knows Secret

#Protocol description

0.      -> Bob : Alice
1.   Alice -> Bob : H(Bob),Ta,{Bob, Secret(Alice), SkeyAlice}{pkdb}%enc
[Bob !=Alice]
2.   Bob -> DB :
H(Bob),Tb,{Alice,Secret(Bob),SkeyBob}{pkdb},enc%{Bob,Secret(Alice),SkeyAlice}{pkdb}
3.   DB -> Bob : H(DB,Bob),SkeyAlice(+)SkeyBob
4.   Bob -> Alice : {Bob}{SkeyAlice}(+)Bob,H(DB)
5.   Alice -> Bob : H(SkeyAlice,Alice),Ta
```

variables. ALICE and BOB are Agents. DB represents the server. As the session keys varying per session, SkeyBob and SkeyAlice are declared. As TimeStamps used to check any attacks based on time, Ta and Tb are declared. In addition, for the security of the server, agents are used for the server security data. (pkdb,skdb), (SkeyBob,SkeyBob) and (SkeyAlice,SkeyAlice) are declared to mean that each function returns a reverse key. #Protocol description defines the messages transmitted in the protocol. Integers 0, 1 and 2 signify the steps of message transmission. The proposed protocol involves hash encryption of messages with values varying per session before transmission. Thus, it is safe against a range of attacks as encrypted messages cannot be decrypted, should they be exposed to intruders in wireless sections.

### 3.2 Operations

The proposed protocol operates following the steps below.

Step ①: ALICE → BOB

Alice receives the Query from Bob and generates a hash function HashFunction(BOB), TimeStamp Ta and the values for Bob and secret agents. Then, through a concatenation with the value of SkeyAlice, Alice saves the generated value in the variable %Enc, while at the same time checking if [Bob!=Alice]. Next, Alice transmits the calculated H(Bob),Ta, {Bob, Secret(Alice), SkeyAlice}{pkdb}%enc to BOB. Here, the generated values are eigenvalues that cannot be generated by another Alice. The data values are yielded by hashing the fixed-length data.

For the initial vector hash function, a random integer 2w is applied to $\bar{a}=(a_0,\ldots,a_k)$. Then, it is calculated as $h_{\bar{a}}(\bar{x})^{strong}=\left(a_0\sum_{i=0}^{k}a_{i+1}x_i \bmod 2^{2w}\right)\div 2^{w}$, which is in turn applied to the variable-length string, $h_a(\bar{x})=h_{\text{int}}\left(\left(\sum_{i=0}^{k}x_i\cdot a^i\right)\bmod p\right)$, where $a \in [p]$ is uniformly random and $h_{int}$ is chosen randomly from a universal family mapping integer domain [p]→[m].

Therefore, Ta,{x}{a1}, $h_a(R)=h_{\text{int}}\left(\left(\sum_{i=0}^{k}x_i\cdot a^i\right)\bmod p\right)$, where the hash operation is performed followed by a concatenation operation with other values. Then, Alice transmits the value of H(Bob),Ta,{Bob, Secret(Alice), SkeyAlice}{pkdb}%enc to BOB.

Step ②: BOB → DB

Using the value of H(Bob),Ta,{Bob, Secret(Alice), SkeyAlice}{pkdb}%enc sent by Alice, Bob calculates the value of H(Bob),Tb, {Alice,Secret(Bob),SkeyBob}{pkdb},enc%{Bob, Secret(Alice),SkeyAlice}{pkdb}.

With hash function operation, Bob generates Tb, the value of $h_a(BOB)=h_{\text{int}}\left(\left(\sum_{i=0}^{k}x_i\cdot a^i\right)\bmod p\right)$ TimeStamp, and calculates the data from Alice, followed by a concatenation operation. Then, Bob calculates {Bob, Secret(Alice),SkeyAlice}{pkdb, saves the value in the variable enc1%, performs math operations and generates the value of Tb,H(BOB),{a2,k}{a2}, enc1%{x}{a1}. Once the H(Bob),Tb,{Alice, Secret(Bob),SkeyBob}{pkdb},enc%{Bob, Secret(Alice),SkeyAlice}{pkdb} data are generated to be transmitted normally, Bob sends the data to DB.

Step ③: DB → Bob)

Using the value of H(Bob),Tb,{Alice, Secret(Bob),SkeyBob}{pkdb},enc%{Bob, Secret(Alice),SkeyAlice}{pkdb} received from Bob, DB performs math operations and then mutual authentication before calculating the following hash value using the value transmitted by Bob. Following $h_a(DB,Bob)=h_{\text{int}}\left(\left(\sum_{i=0}^{k}x_i\cdot a^i\right)\bmod p\right)$, DB performs a concatenation operation to generate the value of H(DB,Bob),SkeyAlice(+)SkeyBob. Once the value of H(DB,Bob),SkeyAlice(+)SkeyBob is generated, DB transmits it to Bob.

Step ④: Bob → Alice

Bob checks and authenticates the value of H(DB, Bob),SkeyAlice(+)SkeyBob, received from DB, upon completion of math operations. Bob calculates the hash value of $h_a(DB)=h_{\text{int}}\left(\left(\sum_{i=0}^{k}x_i\cdot a^i\right)\bmod p\right)$ and concatenates it with the value of {Bob}{SkeyAlice}(+)Bob to generate the value of {Bob}{SkeyAlice}(+)Bob,H(DB). Once the value is generated normally, Bob sends it to Alice for authentication.

Step ⑤: Alice → Bob

Lastly, upon receiving from Bob the value of {Bob}{SkeyAlice}(+)Bob,, $h_a(DB)=h_{\text{int}}\left(\left(\sum_{i=0}^{k}x_i\cdot a^i\right)\bmod p\right)$, Alice compares it with the one of her own generating. Upon completion of checking the value, Alice encrypts her own ID with hash operation $h_a(SkeyAlice,Alice)=h_{\text{int}}\left(\left(\sum_{i=0}^{k}x_i\cdot a^i\right)\bmod p\right)$, Ta, transmits it to Bob, and finishes the authentication session in Alice. Then, Bob receives from Alice the value of $h_a(SkeyAlice,Alice)=h_{\text{int}}\left(\left(\sum_{i=0}^{k}x_i\cdot a^i\right)\bmod p\right)$, Ta and transmits it to DB, which in turn searches and checks the value of Alice saved. Upon completion of normal checking, hash codes and Alice code can be checked. Then, follow-up operations ensue before the session is completed.

## 4 Results

To test the proposed authentication protocol, the program was installed in the following environment. To run Casper 2.0,

**Table 1** Trial environment

| Platform | Specification |
|---|---|
| CPU | Intel(R) Core(TM)2 Duo CPU E7300 2.66GHz |
| RAM | 4GB DDR3 RDIMM, 1333MHz, ECC |
| HDD | 1.5TB SATA(7200RPM) |
| Video card | ATI Radeon HD 3450 512Mb Internal DAC(400Mhz) |
| OS | RedHat Linux 6.0 32 BIT |
| Kernel | Linux 2.6.32-71.el6.x86_64 |
| Compiler | Casper version 2.0 |
| Model checking program | FDR 2.91 Academic teaching and research release |

JAVA 1.6.0-openjdk was used in Linux Kernel version 2.6 31 Bit environment. Also, FDR 2.91 Academic teaching and research release version was installed to check the security of the CSP language. Table 1 summarizes the environment used for the trial.

The proposed machine-to-machine communication protocol was verified with the FDR model checking tool in terms of its safety, deadlock and livelock. The FDR tool marks any security vulnerability with X. Also, the Debug menu pinpoints a vulnerable step where any problem arises and makes it easy to rectify a fault, enabling reinforcement and re-testing. Figure 3 shows the designed protocol being read with CasperFDR and transformed into CSP files without trouble for the verification of security aspects. If the source files had any logical issues or errors, the transformation into the CSP files would not be seamless.

Figure 4 is about verifying the security of the protocol and whether the process works safely and normally by running the FDR program once the CSP files are generated normally. When all requirements are met, green tick marks appear. Any problem is marked with X, which indicates a faulty part, which is in turn rectified and re-tested.

Figure 5 shows the state upon completion of verifying several security attributes with FDR by loading the designed source files. This is the state after the safety and security against deadlock and livelock are completely verified. As seen in Fig. 5, the proposed security protocol proves to return satisfactory values of all security attributes

Figure 5 shows 3 verification results, each of which is analyzed as below.

1) SECRET_M::SECRET_SPEC[T=SECRET_M::SYSTEM_S

   The security of the proposed protocol was checked. The tick mark before a message indicates the protocol was not exposed to intruders and its security was verified. The security of the proposed protocol including the inter-agent communication, TimeStamp, hash functions and SessionKey was verified. That is, the protocol remained intact after a range of attacks.
2) SECRET_M::SEQ_SECRET_SPEC[T=SECRET_M::SYSTEM - S_SEQ

   This is about if the process of the proposed protocol operated normally in the system. As seen in Fig. 5, the proposed protocol proved to work in a safe process.
3) AUTH1_M::AuthenticateRESPONDERToINITIATOR-Agreement_k[T=AUTH1_M::SYSTEM_1

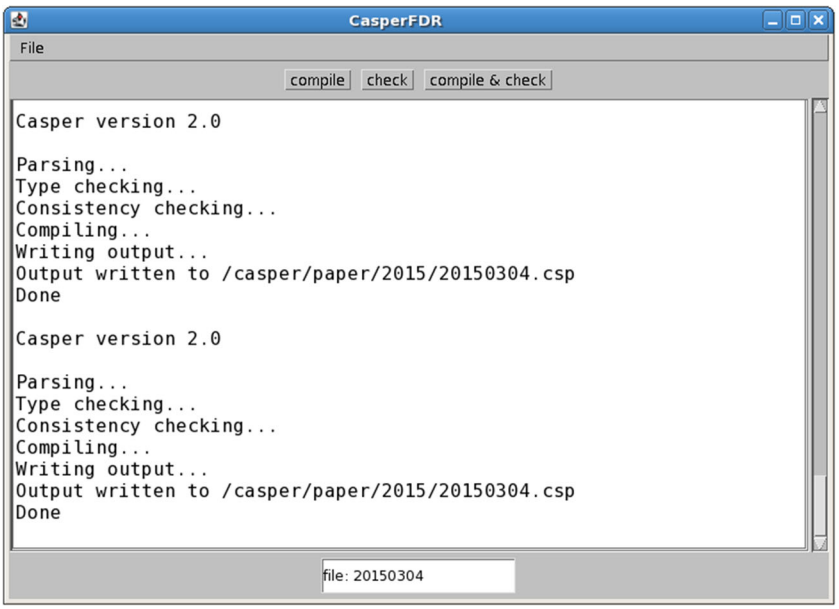


**Fig. 3** Transforming protocol files into CSP format

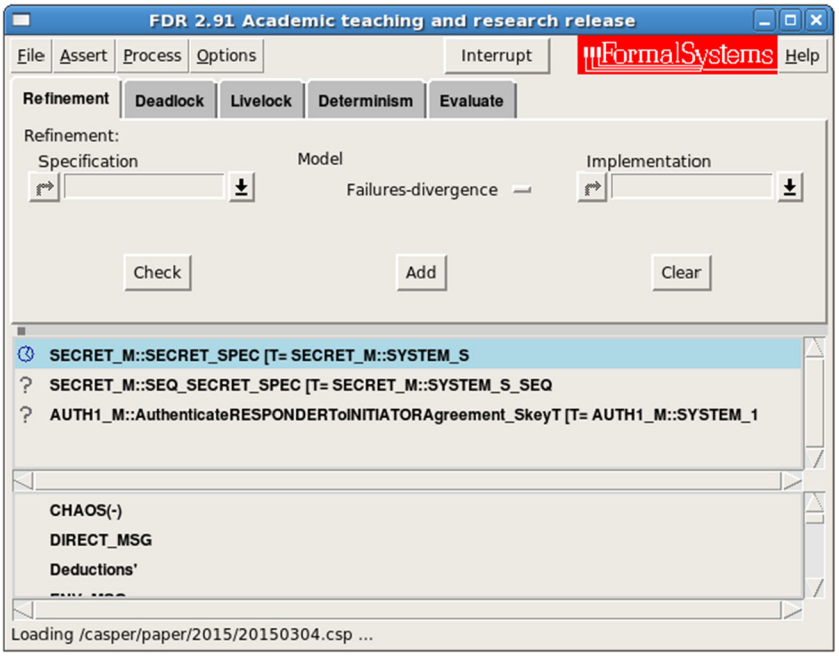


**Fig. 4** Verification in progress

Concerns verifying if the responder and initiator are mutually authenticated via k. The CasperFDR program verified that the proposed protocol enabled a secure mutual authentication between the responder and Initiator.

The inter-session security was satisfactorily verified. In each verification step, the process was fully completed, which kept the system from being in a deadlock. Also, the protocol was completely verified, not causing memory errors, infinite iteration, system shutdown and other system-related issues.

## 5 Conclusion

First of all, this paper defined the concept of M2M communication system, relevant issues, security threats and security requirements. With the advancement of M2M industry and the increase in its application across the latest networks, any attacks from intruders cause serious damages to security, leading to system disruption and malfunction. Thus, reinforcing M2M security system against threats to communication has drawn much attention. In the same vein, extensive research efforts are being made on security measures against attacks including

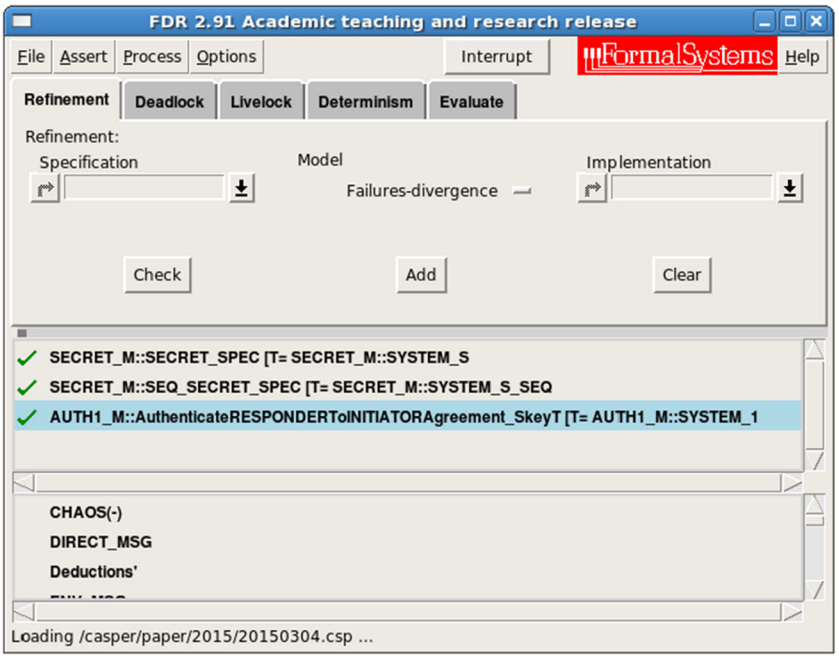


**Fig. 5** Security verification results of the protocol

hardware development, diverse encryption approaches and cryptographic protocols. This paper designed the proposed protocol by drawing on hash functions, real-time secret agents and encryption key values and inserting SessionKey and PublicKey so as to transmit optimized values varying per session. The hash operations reduced the communication data. Inter-agent mutual authentication, TimeStamp and identification of secret agents before transmission of messages virtually precluded intruders, if any, tampering with communication.

To verify the safety of each step in the proposed protocol, it was specified in Casper. Then, whether the proposed protocol met the security attributes of the FDR tool was verified. The proposed protocol proved to meet the security requirements throughout the verification process. The present findings will ensure safer security environment for M2M communication, and be conducive to the reliability of M2M environment. Further studies will pertain to applying complex functions for more efficient and robust security authentication.

## References


1. Song JS (2013) M2M standards and technology trends. TTA J 150:84–89
2. Pyo CS (2013) M2M techonolgy and its standardization trends, oneM2M 2013 Seoul International Conference
3. Wu G, TalwReader S, Johnsson K, Himayat N, Johnson KD (2011) M2M: from mobile to embedded internet. IEEE Commun Mag 49(4):36–43
4. Ngo HH, Wu X, Le PD, Srinivasan B (2010) An individual and group authentication model for wireless network services. J Convergence Inf Technol 5(1):82–94
5. ETSI (2011) "Machine to machine communications (M2M); M2M functional architecture," ETSI, TS 102 690
6. Hummen R, Ziegeldorf JH, Shafagh H, Raza S, Wehrle K (2013) "Towards viable certificate-based authentication for the Internet of Things". In: Proc, ACM HotWiSec.'13: 37–42
7. Kalyani P, Chellappan C (2011) Heterogeneous wireless mobile sensor network mobile based routing adapted to dynamic topology. Eur J Sci Res 50(1):143–150
8. Aiash M, Mapp G, Lasebae A, Phan R, Loo J (2012) A formally verified AKA protocol for vertical handover in hetero-geneous environments using Casper/FDR. EURASIP J Wirel Commun Netw 2012:57–80
9. Chao H-C, Zeadally S, Chen Y-S, Martinez G, Wang R-C (2010) Next Generation Networks(NGNs). Int J Commun Syst 23:691–693. doi:10.1002/dac.1144
10. Lowe G, Broadfoot P, Dilloway C, Hui M, Casper, "A compiler for the Analysis of security protocol," 2011. (Available from: http://www.comlab.ox.ac.uk/gavin.lowe/Security/Casper/), Accessed 19, 2011
11. Aiash M, Mapp G, Lasebae A, Nemrat A AL (2012) "Supporting LTE networks in heterogeneous environment using the Y-Comm framework". In: Proceeding of The Fourth International Conference on Networks & Communications (NETCOM-2012), Chennai, India, pp. 125–136
12. He D, Chen C, Chan S, Bu J (2012) Strong roaming authentication technique for wireless and mobile networks. Int J Commun Syst. doi:10.1002/dac.1387, **Early view of an online version**
13. Chen C, He D, Chan S, Bu J, Gao Y, Fan R (2010) Lightweight and provably secure user authentication with anonymity for the global mobility network. Int J Commun Syst 2010 24:347–362. doi:10.1002/dac.1158
14. Stig Fr M, Joe-Kai T (2012) "Computational security analysis of the UMTS and LTE authentication and key agreement protocols". CoRR, abs, pp. 1203–3866
15. Aiash M, Mapp G, Lasebae A, Phan R (2012) A survey on authentication and key agreement protocols in heterogeneous networks. Int J Netw Secur Appl (IJNSA) 2012 4(4):199–214
16. Peris-Lopez P, Hernandez-Castro JC, Estevez-Tapiador JM, Ribagorda A (2010) Vulnerability analysis of RFID protocols for tag ownership transfer. Comput Netw 54(9):1502–1508
17. Song B, Mitchell CJ (2011) Scalable RFID security protocols supporting tag ownership transfer. Comput Commun 34(4):556–566
18. Chen H, Yu S, Shang J etc. (2009) "Comparison with several fuzzy trust methods for P2P-based system". In proceedings of the 2009 International Conference on Information Technology and Computer Science, Washington, DC, USA, pp. 188–119
19. Aringhieri R, Damiani E, Vimercati SDCD, Paraboschi S, Samarati P (2006) Fuzzy techniques for trust and reputation management in anonymous peer-to-peer systems, special topic section on soft approaches to information retrieval and information access on the web. J Am Soc Inf Sci Technol 57(4):528–553
20. Shin K, Reeves DS, Rhee I (2009) "Treat-before-trick: free-riding prevention for bittorrent-like peer-to-peer networks". Proceedings of 23rd IEEE international parallel and distributed processing symposium, pp. 1–12
21. Sarjaz BS, Abbaspour M (2013) Securing BitTorrent using a new reputation-based trust management system. Peer-to-Peer Netw Appl 6:86–100
22. Nam T, Lee H, Jeong C, Han C (2005) A harmful content protection in peer-to-peer networks. Artif Intell Simul 3397:617–626
23. Lowe G (2009) Casper: a compiler for the analysis of security protocols. Oxford University Computing Laboratory, Oxford
24. Kim l-G, Jeon C-W, Kim H-S, Choi J-Y, Kang I-H (2005) Formal methodology for analysis of security protocols. J Korea Inst Inf Secur Cryptol 15:17–27
25. Fromal system(Europe) Ltd (2010) Failures-divergence refinement FDR2 user manual. Oxford University Computing Laboratory, Oxford
26. Pura M-L (2010) Victor valeriu patriciu, ion bica. "Formal verification of G-PAKE using Casper/FDR2-securing a group PAKE protocol using Casper/FDR2," Security and Cryprography Proceedings of the 2010 International Conference. pp. 1–5
27. Bae WS (2014) Formal verification of an RFID authentication protocol based on hash function and secret code. Wirel Pers Commun Int J 79(4):2595–2609


**Woo-Sik Bae** received the M.S. degree in Telecommunication in the field of Information Communication Engineering, Baekseok University, Chungnam, Korea, (2006), He is a Ph.D. degree from the Computer Education, Chungbuk National University, Chungbuk, Korea, (2012). In recent years, He has published more than 48 papers on RFID security in international and Korea journals and conferences Proceedings. His research interest includes: RFID security, authentication protocols, network security and public key cryptography. He supervised and managed conferences including 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 the KCS and the SDPM.