A Light-Weight Mutual Authentication Protocol for IoT Systems

Muhammad Naveed Aman, Kee Chaing Chua, and Biplab Sikdar
Department of ECE,
National University of Singapore, Singapore

***Abstract*—One of the most important and critical requirements for Internet-of-Things (IoT) based systems is security under limited resources. The simple and low-cost nature of many IoT devices makes them a prime target for physical, side-channel, and cloning attacks. To address this issue, this paper presents an efficient protocol for mutual authentication in IoT systems. The proposed protocol uses a Physical Unclonable Function to provide the desired security characteristics. An analysis of the protocol shows that it is not only robust against different kind of attacks, but also very efficient in terms of memory, computations, energy, and communication overhead.**

I. Introduction

The IoT has been envisioned as a technology that will enable networking between people and things, generating data that can be used to intelligently control systems and optimize resource utilization. A large fraction of the devices in the IoT will use wireless channels to connect to the network. The use of a wireless channel makes it easier for an adversary to eavesdrop and launch attacks, exposing the devices to a wide range of cyber-threats. The most important security requirements in the IoT include secure booting, authentication, access control, data integrity, privacy, and digital forgetting [1].

Traditional cryptography based techniques and protocols for the Internet were designed for nodes with no power and memory constraints. Moreover, these protocols were designed with the assumption that the devices are physically well-protected. The case of IoT systems is different. Many IoT devices are small sized and have limited power and computing capabilities, and once installed in the field they can easily be captured by an adversary. Thus these systems are prone to physical and side channel attacks. To address security challenges in the IoT, we consider the use of Physical Unclonable Functions (PUFs). PUFs have emerged as a promising technique for hardware based security primitives. PUFs eliminate the need for storing secrets in a memory, making them one of the prime contenders for security protocols for IoT systems. PUFs exploit the inherent variability in integrated circuit (IC) manufacturing to implement challenge-response functions whose output depends on the input and the physical micro structure of the device. The variations in the physical factors during the fabrication process make it practically impossible to replicate the micro structure, making PUFs unique at a device level.

This paper addresses the problem of authentication for IoT systems. The major contributions of this work are as follows: (i) We propose a PUF based authentication mechanism with very low communication overhead, computation and memory requirements. Moreover, the authentication protocol does not require any secrets to be stored locally; (ii) An authenticated key establishment procedure without any extra overhead is proposed. The proposed protocol is safe from physical and side channel attacks.

The rest of the paper is organized as follows. Section II discusses the related work. Section III presents a brief introduction to PUFs. Section IV discusses the network model and assumptions for our IoT system. Section V presents the proposed PUF based mutual authentication protocol. We present security and performance analysis in Sections VII and VIII, respectively. Finally, Section IX concludes the paper.

II. Related Work

Xu et al. [1], describe the different security challenges for designing IoT systems. They proposed the use of Computer Aided Design techniques and hardware security primitives such as PUFs for the design of highly optimized IoT devices.

Several schemes for authentication in IoT systems have been proposed in literature [4], [5], [6]. However all these schemes rely on complex computations and require some secrets to be stored in the IoT device. Both of these properties make these mechanisms unsuitable for IoT devices.

Several PUF based authentication protocols for wireless sensor networks and RFID systems have been proposed in literature [7], [8], [9]. However, these either require a large number of challenge-response pairs (CRPs) for each device to be stored in the server or require storing some kind of initial secret in the device's memory. These requirements make these protocols not suitable for IoT systems because of two reasons. Firstly, they are not scalable, and secondly, they render the PUF useless, exposing the device to physical attacks.

The authors of [10] use zero-knowledge proof of knowledge (ZKPK) of discrete logarithm for authentication. They use a combination of a PUF and user password to provide physical security. Although their proposed protocol does not require any stored secrets, the requirement of a user password for authentication each time makes it less effective for IoT devices. Moreover, their proposed protocol is more computationally complex due to the use of ZKPK of discrete logarithm.

III. Preliminary Background

From [7]: "A Physical Unclonable Function (PUF) is a function that maps a set of challenges to a set of responses based on an intractably complex physical system". A PUF can

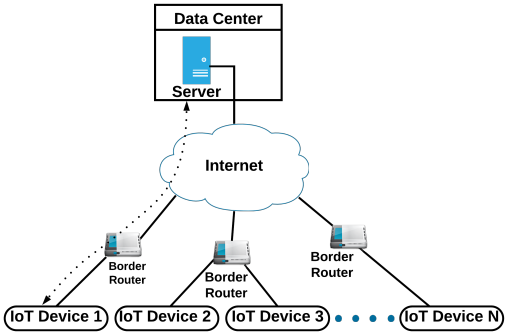

Fig. 1. Network Model

be characterized by a challenge-response pair, where *C* is the challenge and *R* is the response for that specific challenge. A PUF can be represented as follows:

$$R = P(C). \tag{1}$$

The authors of [2], describe a PUF as "an expression of an inherent and unclonable instance-specific feature of a physical object". The property of a PUF that makes it attractive for security is that due to its physical basis, it is immensely hard or even impossible to produce a physical clone of a PUF [3]. Therefore, the output of a PUF is unique for each physical instance and cannot be characterized using an invasive or a side-channel attack. Additional properties of a PUF include:

- Simple to construct and evaluate.
- Output is indistinguishable from a random function.
- A PUF always outputs the same response with the same challenge with high probability. However, if the same challenge is input to a different PUF, it will produce a response far apart with high probability.

IV. Network Model and Assumptions

A. Network Model

Figure 1 shows the network model. In this model, each IoT device is equipped with a PUF. IoT devices are connected using border router elements (e.g 6LoWPAN) and send information over the Internet to a server in a data center.

B. Assumptions

We make the following assumptions for the network model and proposed protocol:

a. Each IoT device is an embedded system. Any attempt to remove the PUF from the device will result in destruction of the PUF [10].
b. It is not possible to eavesdrop or tamper with the communication between a device's microcontroller and PUF [11].
c. The IoT devices have limited resources, while the data center acts as the trusted party. Moreover, the data center has no limitation of resources.
d. An adversary can eavesdrop any message passing through the network, can initiate a session with any other user, inject packets into the network, impersonate, and replay older messages.

TABLE I
Notations

Notation	Description
ID_j	Identity (ID) of the IoT device
$\oplus$	XOR operation
$H(X)$	Hash of X
$\|$	Concatenation operator
$[Ex]_{Mem}$	Expression Ex is evaluated using the device's memory
$[Ex]_{Rec}$	Expression Ex is evaluated using the values from the received message
C^i	Challenge for the i-th iteration
R^i	Response of the respective PUF for C^i

V. Proposed PUF based Mutual Authentication Protocol

In this section we describe the proposed mutual authentication protocol for IoT systems based on PUFs. The set of notations used to describe the protocol are given in Table I.

The proposed authentication protocol is shown in Figure 2. The server in our network model is considered the trusted party. It is assumed that the server has a secure way to obtain the initial CRP in offline mode. For example, the CRP may be manually entered or electronically copied from a manufacturer supplied list at the server. The server then stores the initial CRP against an $H(ID_j)$, i.e., the ID of a device is not stored in plain text for user identity protection.

A. Mutual Authentication

Consider a scenario where an IoT device ID_A wishes to authenticate with the server. The device may have authenticated with the server in the past and let this be the i-th authentication attempt. The authentication process consists of the following steps:

1) The IoT device first sends the hash of its ID and a randomly generated nonce N_1 to the server as shown in message 1 in Figure 2.
2) The server searches its memory for $H(ID_A)$ and if it does not find a match, authentication fails. Otherwise, the server selects the CRP (C^i, R^i) stored against $H(ID_A)$, and generates a random nonce N_2. The server then sends the challenge C^i along with $H(ID_A \parallel R^i \parallel N_1 \parallel N_2)$ to IoT device ID_A as shown in message 2.
3) IoT device ID_A inputs the received challenge C^i to its PUF and obtains the response R^i. It then verifies the integrity and freshness of the received message by evaluating

$$\big[H(ID_A \parallel R^i \parallel N_1 \parallel N_2)\big]_{Mem} \doteq \big[ID_A \parallel R^i \parallel N_1 \parallel N_2)\big]_{Rec} \tag{2}$$

i.e., it regenerates the hash function using the parameters in its memory and the PUF output, and compares it with the received hash function. If the two are not equal the

authentication fails. Otherwise, the IoT device generates a random number N_A, and sends message 3 in Figure 2 to the server. In this message T represents the current time and is used to ensure freshness of messages.

4) The server first verifies the following expression

$$\left[H(H(ID_A) \parallel R^i)\right]_{Mem} \doteq \left[H(H(ID_A) \parallel R^i)\right]_{Rec} \quad (3)$$

If (3) does not hold the authentication fails. Otherwise, the server computes N_A using the following

$$N_A = R^i \oplus \left[N_A \oplus R^i\right]_{Rec} \quad (4)$$

The server compares the current time at its end T^* with the time stamp of the received message T. This evaluation is done using

$$(T^* - T) \geq \Delta T \quad (5)$$

where ΔT represents the expected latency of the message. ΔT also accounts for any clock drifts between the IoT device and the server. If (5) fails validation, the authentication is rejected. Otherwise, the server uses R^i, N_A and T to verify

$$\left[H(ID_S \parallel N_A \oplus R^i \parallel T \parallel N_2 \| N_A)\right]_{Mem} \doteq \left[H(ID_S \parallel N_A \oplus R^i \parallel T \parallel N_2 \| N_A)\right]_{Rec}. \quad (6)$$

If (6) holds, the server generates a random number N_B and sends message 4 to the IoT device. Otherwise, the server refuses the authentication request.

5) IoT device ID_A computes N_B using the following:

$$N_B = N_A \oplus \left[N_B \oplus N_A\right]_{Rec}. \quad (7)$$

It then uses R^i, N_A and N_B to verify

$$\left[H(N_B \oplus N_A \| N_B)\right]_{Mem} \doteq \left[H(N_B \oplus N_A \| N_B)\right]_{Rec}. \quad (8)$$

If (8) does not hold, authentication fails. Otherwise, the IoT device inputs $H(N_A \parallel N_B)$ into its PUF to obtain the new CRP for any future authentications, i.e., $R^{i+1} = P_j(H(N_A \parallel N_B))$. The device then deletes all the temporary variables including N_A, N_B, R^i, C^{i+1}, and R^{i+1}, and sends message 5 to the server as shown in Figure 2.

6) The server verifies

$$\left[H(R^i \parallel N_B)\right]_{Mem} \doteq \left[H(R^i \parallel N_B)\right]_{Rec}. \quad (9)$$

If verification fails, authentication is rejected. Otherwise the server computes the new CRP as follows:

$$C^{i+1} = H(N_A \parallel N_B) \quad (10)$$

$$R^{i+1} = N_B \oplus \left[R^{i+1} \oplus N_B\right]_{Rec}. \quad (11)$$

The server then verifies

$$\left[H(R^{i+1} \oplus N_B \| R^{i+1})\right]_{Mem} \doteq \left[H(R^{i+1} \oplus N_B \| R^{i+1})\right]_{Rec}. \quad (12)$$

If (12) holds, authentication is complete and the server replaces (C^i, R^i) with the new CRP (C^{i+1}, R^{i+1}).

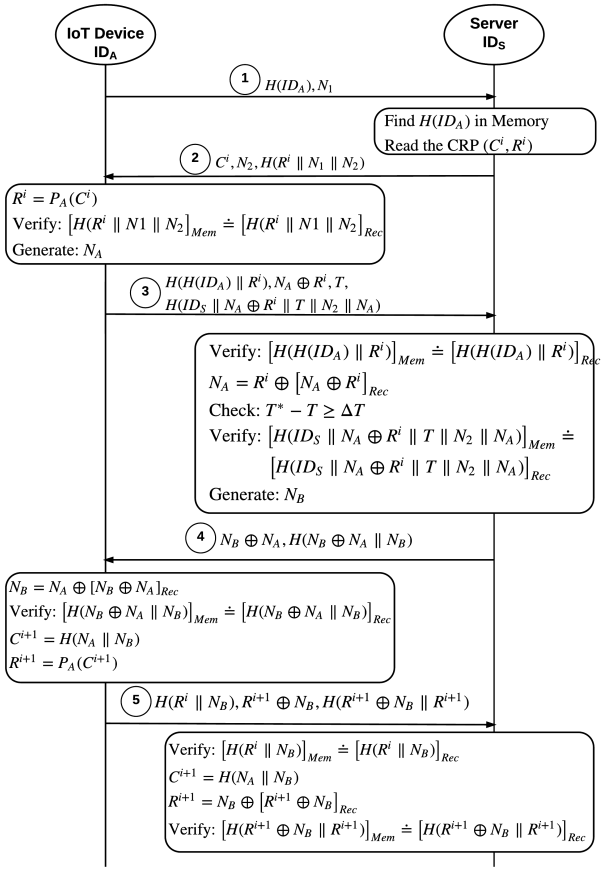

Fig. 2. Proposed Mutual Authentication Protocol

B. Session Key Establishment

To establish the session key, we use a technique similar to that in [12]. The random numbers N_A and N_B can be used to establish a session key. We can use $H(N_A) \oplus H(N_B)$ as the session key. In case the key is revealed by accident or stolen, the system will still remain safe. To verify this, note that if an adversary A gets hold of the secret key, it is still not possible to calculate R^i or construct "valid" data.

VI. Protocol Verification

A formal verification of the correctness of the proposed protocol is presented in this section. We show that the proposed protocols possess the following properties [15]:

1) **Completeness**: The protocol accepts all valid inputs.
2) **Deadlock freeness**: The protocol does not enter a deadlock state, i.e., a state in which it stays indefinitely.
3) **Livelock or tempo-blocking freeness**: The protocol does not contain an infinite loop.
4) **Termination**: Starting from an initial state, the protocol is always able to reach a well-defined final state.
5) **No non-executable interactions**: The protocol only contains realizable (under normal conditions) transmission, reception, and interaction paths.

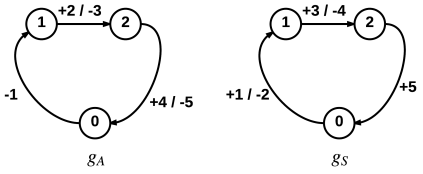

Fig. 3. Directed Graph/FSM for Protocol 1

The correctness of the proposed protocol is proved using techniques proposed in [15]. In this technique, we first represent each entity of a protocol using a directed graph which can also be considered a finite state machine (FSM). The directed graphs for the IoT device ID_A and the server, denoted by g_A and g_S respectively, are shown in Figure 3.

In this section we refer to ID_A as A in the figures. The labeled circles represent the state of a protocol machine. -m (respectively, +m) on the directed arcs represent a transmission (reception) of message m. Moreover, +m/-n denotes the reception of message m followed by the transmission of message n. For example, one run of the proposed protocol corresponds to the following interaction paths for g_A and g_S:

- g_A: [0] -1[1] +2/-3[2] +4/-5[0]
- g_S: [0] +1/-2[1] +3/-4[2]+5[0]

where the numbers in the brackets represents a state in Figure 3. The above sequence of events for IoT device ID_A is interpreted as: ID_A starts in state 0, sends message 1 to the server and enters state 1, receives message 2 and transmits message 3 to enter state 2, and finally receives message 4 and transmits message 5 to enter state 0 again. The sequence of events for the server can be interpreted in a similar fashion. Note that state 0 is considered the final state for both the IoT device ID_A and the server.

We use the reachablitiy analysis technique proposed in [16], [15], as the next step to prove the correctness of the proposed protocol. We use the following matrix to denote the overall state of the system (consisting of all entities in the protocol):

$$\begin{bmatrix} A & A \rightarrow Server \\ STATE & CHANNEL \\ Server \rightarrow A & Server \\ CHANNEL & STATE \end{bmatrix}. \tag{13}$$

Each element in the above matrix represents either the current state of the FSM of an entity or the message sent by the entity. For example, all entities are in their initial state, i.e., state 0 at the start of the protocol. Now let us assume ID_A sends message 1 to the server. Figure 3 shows that the FSM of ID_A transitions into state 1 after sending message 1. The overall state of the system can then be represented by the following matrix:

$$\begin{bmatrix} S_1 & 1 \\ E & S_0 \end{bmatrix}. \tag{14}$$

The matrix in (14) shows that the IoT device ID_A is in state 1 denoted by S_1 (element at row 1 and column 1), while the element at row 1 and column 2 shows that message 1 has been sent from the IoT device ID_A to the server. Similarly, in this example the server is currently in state 0 denoted by S_0 and has not sent any message to ID_A, represented by the E at row 2 and column 1. The overall state of the system is represented by SSi, while S_i is used to denote the constituent states of the subsystems (entities).

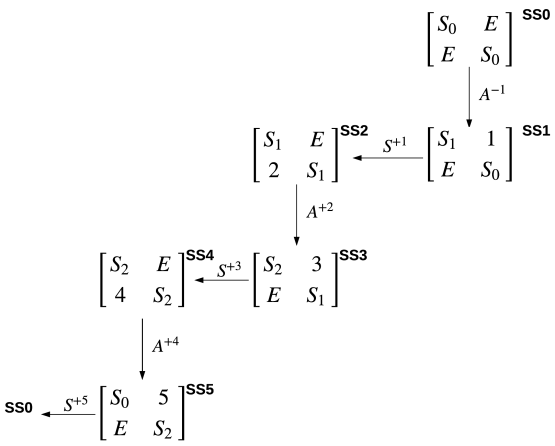

Fig. 4. Reachability Analysis for Protocol 1

Figure 4 shows the result of the reachability analysis of the proposed protocol. All the protocol entities are in their respective initial states, i.e., S_0, and all the channels are empty, i.e., E, at the start of the protocol i.e., system state $SS0$. Moreover, we represent a transition from one system state to another caused by the transmission (reception) of message i by entity X by X^{-i} (X^{+i}) on the respective directed arcs.

Figure 4 shows that the proposed protocol starts with a transition from $SS0$ to $SS1$ when IoT device ID_A sends message 1, followed by subsequent transitions. We observe that the protocol accepts all valid inputs implying the completeness property. Similarly, Figure 4 shows that the proposed protocol does not have any potential deadlock state, implying deadlock freeness. An overall system state which is not an initial or final state and whose channels are all empty is called a potential deadlock state. Moreover, Figure 4 shows that the protocol always ends up in state $SS0$ and does not contain any infinite loops. This proves termination, livelock freeness, and absence of non-executable interactions. Thus, we can consider the proposed protocol as correct.

VII. Security Analysis

A. Simulations

The security of the proposed protocol is established using ProfVerif (PV) [17]. PV is an automated security verification tool which has the ability to simulate and test security protocols against various types of attacks. PV can be used to prove properties such as correspondence assertions, observational equivalence, and reachability properties. We model the IoT

device and the server as separate processes. Moreover, we simulate arbitrarily many sessions between the two parties by instantiating an unbounded number of instances of the respective processes. The simulation script for our mutual authentication protocol can be found at [19].

To prove secure mutual authentication of the principals ID_A and ID_S we define the following events in PV:

- **event** $beginAfull(ID_A, ID_S, N_A, N_B)$: IoT device ID_A has started a run of the protocol with ID_S where N_A and N_B are the shared secrets.
- **event** $endAfull(ID_A, ID_S, N_A, N_B)$: IoT device ID_A has successfully completed the protocol with ID_S where N_A and N_B are the shared secrets.
- **event** $beginBfull(ID_A, ID_S, N_A, N_B)$:$ID_A$ intends to launch the protocol with ID_S with the given shared secrets.
- **event** $endBfull(ID_A, ID_S, N_A, N_B)$: ID_S has successfully completed the protocol with ID_A with the given shared secrets.

We intend to prove the following authentication properties:

1) **Authentication of ID_S to ID_A:** This property represents the fact that if IoT device ID_A has completed the protocol, she has indeed done so with the server. We use the following correspondence assertion to prove this property in PV:

$$inj\text{–}event(endBfull(\cdots)) ==> inj\text{–}event(beginBfull(\cdots)). \quad (15)$$

The statement $event_A ==> event_B$ verifies the fact that whenever there is an occurrence of the event $event_A$, it must always be preceded by the event $event_B$.

2) **Authentication of ID_A to ID_S:** The server needs to be assured that if he thinks he has completed a run of the protocol with IoT device ID_A then he has indeed done so with the IoT device ID_A. We use the following correspondence assertion to prove this property in PV:

$$inj - event(endAfull(\cdots)) ==> inj - event(beginAfull(\cdots)). \quad (16)$$

We establish the (syntactic) secrecy of N_A, N_B, and R^{i+1} in the proposed protocol using the following queries in PV:

$$query \quad attacker(ANa); attacker(BNa); \quad (17)$$
$$attacker(ANb); attacker(BNb); \quad (18)$$
$$attacker(AR\_new); attacker(SR\_new) \quad (19)$$

where, ANa, ANb, and AR_new are used to prove the secrecy of N_A, N_B, and R^{i+1} on the site of principal ID_A. While, BNa, BNb, and BR_new establish the secrecy of N_A, N_B, and R^{i+1} on the site of principal ID_S.

The communication channels simulated by PV are assumed to be controlled by an attacker with Dolev-Yao capabilities [18] i.e., the attacker may: read, modify, delete, and inject messages. The attacker may also manipulate data e.g., compute a specific element of a tuple and if the attacker has the necessary keys, he/she may also decrypt a message. Moreover, PV can also simulate the behavior of dishonest participants. Therefore, after simulating and verifying the security properties of the proposed protocol in PV, we conclude that the proposed protocol is secure against different types of attacks.

TABLE II
COMPUTATIONAL BURDEN OF THE PROPOSED PROTOCOL

Task	IoT Device	Server
Proposed Protocol	$7N_H + 3N_{\oplus}$	$6N_H + 4N_{\oplus} + 2N_M$
[10]	$2N_H + 2N_{exp} + N_{\times}$	$1N_H + 3N_{exp}$

B. User Identity Protection

IoT devices should not reveal their private information such as name, location, time, and user data. The proposed protocol saves a one way hash function of the ID in the device and the same is also used for communication between the server and the device. Therefore, an adversary cannot extract any kind of information regarding the device's ID.

C. Cloning and Physical Attacks

The use of a PUF in the proposed authentication protocol safeguards it against cloning attacks. It has been shown that it is not possible to clone a PUF, i.e., each PUF has a unique output which can not be recreated [7].

One of the most important security requirements for nodes like IoT devices is that they should not reveal any secrets even if they are captured by an adversary. Most of the authentication protocols based on PUFs discussed in Section II require some kind of secret (e.g. symmetric or private keys) to be stored in the device. This makes them susceptible to physical attacks. The proposed protocol does not require any secrets to be stored in the device. Moreover, PUFs cannot be separated from the device's microcontroller and an attacker cannot eavesdrop on the communication between the chip and the PUF [11]. Therefore, even if an IoT device is captured by an adversary it will not leak any information about the PUF. Thus, physical attacks are rendered useless.

VIII. PERFORMANCE ANALYSIS

This section discusses the efficiency of the proposed protocol. We compare the performance of the proposed protocol with the PUF based authentication protocol proposed by Frikken et al. in [10].

A. Computational Efficiency

Let us denote the number of Hash operations as N_H, the number of exclusive-or operations as $N_{\oplus}$, the number of modular exponentiation by N_{exp}, and the number of basic arithmetic operations (add, subtract, and comparison) as N_M. The computational requirements of the proposed protocol are shown in Table II.

Let us assume the use of universal hashing. Then the hash operations have a worst case running time of $O(n)$ [20], [21], where n is the message size. Similarly, exclusive-or operations

TABLE III
PARAMETER LENGTHS

Parameter	Size (bits)
ID	8 [26]
N_1, N_2	48 [27]
T	64
N_A, N_B	128 [23]
C, R	128 [23]
H	32/64/96/128 [24]

are also $O(n)$. This shows that the proposed protocol has a complexity of $O(n)$ for the IoT device as well as the server. However, the protocol in [10] requires the computation of modular exponentiation which translates into a worst case running time of $O(n + M(l)k)$ for the IoT device as well as the server where k is the exponent, and $M(l)$ is the complexity of a general modular multiplication with l bit operands. This shows that the proposed protocol has a lower computational burden. The low computational complexity translates into low energy requirements, making it attractive for practical implementation.

B. Storage Requirement

The proposed protocol does not require anything to be stored in the IoT device except for $H(ID_A)$. The temporary variables N_A and N_B are only stored for the duration of the authentication. Moreover, the server needs to store only three variables namely: $H(ID_A)$, and (C, R) for each IoT device. Other PUF based protocols either require the devices to store secret information in the IoT devices resulting in compromised security (e.g. [13], [9]) or the server to store a large number of CRPs resulting in non-scalability (e.g. [7]). On the other hand, the proposed protocol has extremely low storage requirements, making it highly scalable.

C. Communication Overhead

Let us assume the size of the different parameters used in the proposed protocol as given in Table III. Figure 2 shows that the longest message in our protocol is message 3. Assuming the output size of the hash function to be 64 bits, message 3 is 40 bytes long. Note that the maximum transmission unit size for 6LoWPAN is 127 bytes [25]. We observe that the maximum length of messages in the proposed protocol is less than the MTU size of 6LoWPAN. However, the addition of other layer headers such as IP headers may result in fragmentation at the 6LoWPAN adaptation layer [25]. Moreover, [10] has a message length of approximately 68 bytes. Thus, the proposed protocol requires lower communication overhead as compared to other authentication schemes.

IX. CONCLUSIONS

This paper presented a novel mutual authentication protocol for IoT systems. The protocol uses PUFs to carry out authentication through a challenge-response mechanism. The protocol can perform authentication and establish a session key without the need to store any secrets in an IoT device. We showed that the proposed protocol is efficient and provides security against different types of attacks including physical attacks, side-channel attacks, and cloning attacks.

REFERENCES

[1] T. Xu et. al., "Security of IoT Systems: Design Challenges and Opportunities," in *Proc, IEEE/ACM ICCAD,* pp. 417-423, San Jose, CA, November 2014.

[2] R. Maes, "Physically Unclonable Functions: Constructions, Properties and Applications," Katholieke Universiteit Leuven Belgium DEngg Thesis, 2013.

[3] C. Bohm, and M. Hofer, "Physical Unclonable Functions in Theory and Practice," Springer, 2012.

[4] V. Shivraj et. al., "One time password authentication scheme based on elliptic curves for Internet of Things (IoT)," in *Proc. NSITNSW,* pp.1-6, Riyadh, KSA, February 2015.

[5] P. Porambage et. al., "Two-phase Authentication Protocol for Wireless Sensor Networks in Distributed IoT Applications," in *Proc. IEEE WCNC,* pp. 2728-2733, Istanbul, Turkey, April 2014.

[6] Y. Kim et. al., "DAoT: Dynamic and Energy-aware Authentication for Smart Home Appliances in Internet of Things," in *Proc. IEEE ICCE,* pp.196-197, Las Vegas, NV, Jan 2015.

[7] G. E. Suh, and S. Devadas "Pysical Unclonable Functions for Device Authentication and Secret Key Generation," in *Proc. IEEE/ACM DAC,* pp. 9-14, San Diego, CA, June 2007.

[8] P. Cotese et. al., "Bernardo, Efficient and Practical Authentication of PUF-Based RFID Tags in Supply Chains," in *Proc. IEEE RFIDTA,* pp. 182-188, Guangzhou, China, June 2010.

[9] Y. S. Lee et. al., "Mutual Authentication in Wireless Body Sensor Networks (WBSN) based on Physical Unclonable Function (PUF)," in *Proc. IEEE IWCMC,* pp.1314-1318, Sardinia, Italy, July 2013.

[10] K. B. Frikken et. al., "Robust Authentication Using Physically Unclonable Functions," in *Proc. Information Security Conference,* pp. 262-277, Pisa, Italy, September 2009.

[11] S. Guilley, and R. Pacalet, "SoCs security: a war against side-channels", *Annals of Telecommunications,* vol. 59, no. 7, pp 998-1009, 2004.

[12] L. T. Liang, and J. Z. Gang, "A New Low Cost One Time ID and Password Authentication Protocol Using Popular Removable Storage Devices," in *Proc. ICINIS,* pp. 213-216, Tianjin, China, November 2009.

[13] K. Yang et. al., "PUF-based Node Mutual Authentication Scheme for Delay Tolerant Mobile Sensor Network," in *Proc. IEEE WiCOM,* pp. 1-4, Wuhan, China, September 2011.

[14] M. J. Hinek, *Cryptanalysis of RSA and its variants.* Taylor and Francis group: CRC Press, 2009.

[15] D. P. Sidhu, "Authentication protocols for computer networks: I", *Computer Networks and ISDN systems,* Vol. 11, pp. 287-310, 1986.

[16] V. Varadharajan, "Verification of network security protocols", *Computers and Security,* Vol. 8, no. 8, pp. 693-708, 1989.

[17] B. Blanchet and B. Smyth, *ProVerif: Automatic Cryptographic Protocol Verier, User Manual and Tutorial*.

[18] D. Dolev and A. C. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory,* Vol. 29, no. 12, pp. 198208, 1983.

[19] https://www.ece.nus.edu.sg/stfpage/bsikdar/scripts/GC17.

[20] M. Babka, "Properties of Universal Hashing," Charles University in Prague, Master Thesis, 2010.

[21] Y. Mansour et. al., "The Computational Complexity of Universal Hashing," *Theoretical Computer Science,* vol. 107, no. 1, pp. 121-133, 1993.

[22] T. Kivinen and M. Kojo, "More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)," IETF RFC 3526, May 2003.

[23] M. Katagi and S. Moriai, "The 128-bit blockcipher CLEFIA," IETF RFC 6114, March 2011.

[24] T. Krovetz, "UMAC: Message Authentication Code using Universal Hashing", IETF RFC 4418, March 2006.

[25] G. Montenegro et. al., "Transmission of IPv6 Packets over IEEE 802.15.4 Networks," IETF RFC 4944, September 2007.

[26] P. Kim, " IoT Specific IPv6 Stateless Address Autoconfiguration with Modified EUI-64," IETF Internet-Draft, July 2015.

[27] D. Whiting et. al., "Counter with CBC-MAC (CCM)," IETF RFC 3610, September 2003.