International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:14, No:4, 2020

A Review of Ultralightweight Mutual Authentication Protocols

Umar Mujahid, Greatzel Unabia, Hongsik Choi, Binh Tran

***Abstract*—**Radio Frequency Identification (RFID) is one of the most commonly used technologies in IoTs and Wireless Sensor Networks which makes the devices identification and tracking extremely easy to manage. Since RFID uses wireless channel for communication, which is open for all types of adversaries, researchers have proposed many Ultralightweight Mutual Authentication Protocols (UMAPs) to ensure security and privacy in a cost-effective manner. These UMAPs involve simple bitwise logical operators such as XOR, AND, OR & Rot, etc., to design the protocol messages. However, most of these UMAPs were later reported to be vulnerable against many malicious attacks. In this paper, we have presented a detailed overview of some eminent UMAPs and also discussed the many security attacks on them. Finally, some recommendations and suggestions have been discussed, which can improve the design of the UMAPs.

***Keywords*—**RFID, UMAP, SASI, IoTs.

I. Introduction

THE Internet of Things (IoTs) is the system of inter-connected computing devices that are provided with unique identification and capability to exchange information with each other without human intervention. The IoTs can be used to improve many existing manual systems, e.g. Sensors based Vehicle diagnostic systems, Weather monitoring and reporting systems etc. The RFID empowers the IoT's sensors/computing devices by providing unique identification and capability of data exchange autonomously. The RFID mainly involves three components; tag, reader and the back-end database.

The tag is the small electronic chip which is embedded onto the object that needs to be identified. The reader acts as a scanner that inquires all the tags which enter in its vicinity and performs authentication. The back-end database contains the detailed information about all the associated tags and the readers. The readers are directly connected to the back-end database to pull the information of the legitimate tags. Usually, it is assumed that the channel between the reader and the backend database is secure because it could be a wired transmission media and also there is no power constraint at the reader/database side. However, the channel between the reader and the tag is wireless which is accessible for all types of adversaries and hence, it needs to be secure.

In 2006, Peris-Lopez [3]-[5] laid the foundation of ultra-lightweight cryptography for low cost RFID systems. The authors used simple bitwise logical operators and designed three UMAPs; Lightweight Mutual Authentication Protocol (LMAP), Extremely Lightweight Mutual Authentication Protocol (EMAP) and Minimalist Mutual Authentication Protocol (M2AP). The design and working of these protocols are discussed in section 2. However, the term ultra-lightweight was first used by Chien [1] in 2007.

Chien [1] categorized the authentication protocols into four types:

a) Full Fledge Protocols: can use traditional cryptographic suites/Algorithms such as AES, Blowfish, RSA or ECC etc.

b) Simple Mutual Authentication Protocols: can incorporate Random Number Generators (LCG, LFSR etc.) and Hash Functions for integrity check.

c) *LMAPS:* designed for low cost applications and can involve lightweight security functions and random number generators but not hash functions.

d) *Ultra-lightweight Mutual Authentication Protocols:* can use only bitwise logical operators and ultralightweight non-triangular primitives.

In this paper, our focus will be on Ultra-lightweight Mutual Authentication Protocols. The rest of the paper is organized as follows: Section II presents the detailed working of some eminent UMAPs. Section III highlights the previously proposed security attacks on UMAPs and recommendations for avoidance. Finally, Section IV concludes the paper and presents the future directions.

II. Survey of UMAPs

In this section, we have discussed the detailed working of some well-known (most cited) UMAPs; LMAP, EMAP, M2AP, SASI and KMAP.

A. LMAP

In 2006, Peris-Lopez et al. introduced the LMAP [3], a real LMAP that use simple cryptographic bitwise functions instead of complex operations such as hashing and typical security algorithms, etc. The simple operations in LMAP allow the protocol to only need around 300 logic gates to perform sufficient security functions [22]. The LMAP protocol is as follows: LMAP involves two entities, the reader and the tag. The communication between the server and the reader is assumed to be secure.

1. *Tag Identification:* The reader will first identify the tag by sending a *hello* message. The tag will then reply with its current *index-pseudonym (IDS).* Only an authentic reader

Umar Khokhar is with the Department of Information Technology, Georgia Gwinnett College, Lawrenceville, United States (e-mail: ukhokhar@ggc.edu).

Greatzel Unabia, Hongsik Choi, and Binh Tran are with the Department of Information Technology, Georgia Gwinnett College, Lawrenceville, United States.

can use the received IDS to access the tag's secret key, which is 96 bits long divided into four subkeys *(K = K1 || K2 || K3 || K4)* [2].

2. *Mutual Authentication:* For reader authentication, the reader must first generate two random numbers n1 and n2. The reader will calculate the value of submessage *A* with the *IDS,* K_1, and n_1. Submessage *B* will be calculated with the *IDS,* K_2 and n_1. And submessage *C* will be calculated with *the IDS,* K_3 *and* n_2 [2]. For tag authentication, the tag will use submessage *A* and submessage *B* to authenticate the reader and to extract n_1. The second random number n2 will be extracted from submessage *C*. The *IDS* and key updating will use the values of n_1 and n_2. After verifications are finished, the tag will generate submessage *D* using the tag's static *ID, IDS,* n_1 and n_2. Generating submessage *D* allows for secure transmission of the tag's static *ID* [2].
3. *Index-Pseudonym and Key Updating:* Once mutual authentication between the tag and the reader have occurred, the *IDS* and the key updating stage must be performed. Since tags have very limited computational capabilities, only simple operations such as, bitwise XOR ($\oplus$), bitwise OR ($\vee$), bitwise AND ($\wedge$), and the addition mod 2^m. The next IDS will be calculated using the current *IDS,* n_2 , K_4 and the static *ID*. Subkeys K_1, K_2, K_3, and K_4's next values must also be calculated. The next K_1 will be calculated using the current K_1, n_2, current K_3 and static *ID*. The next K_2 will be calculated using the current K_2, n_2, current K_4 and static *ID*. The next K_3 will be calculated using the current K_3, n_1, current K_1 and static *ID*. Lastly, the next K_4 will be calculated using the current K_4, n_1, current K_2 and static *ID*.

B. EMAP

In the same year, Peris-Lopez et al. proposed another protocol [4], the Efficient Mutual Authentication Protocol (EMAP), an extremely efficient mutual authentication protocol, which provides adequate security and only needs around 150 logic gates to operate security functions. Similar to LMAP, EMAP does not need to perform complex cryptographic functions such as PRGNs or hash [1], [3]. The EMAP protocol is as follows: EMAP involves two entities, the reader and the tag. The communication between the server and the reader are assumed to be secure.

1. *Tag Identification:* The reader will first identify the tag by sending a *hello* message. The tag will then reply with its current *index-pseudonym (IDS)*. Only an authentic reader can use the received IDS to access the tag's secret key, which is 96 bits long divided into four subkeys (*K = K1 || K2 || K3 || K4*) [3].
2. *Mutual Authentication:* For reader authentication, the reader must first generate two random numbers n_1 and *n2*. The reader will calculate the value of sub-message *A* with the *IDS, K1*, and *n1*. The submessage *B* will be calculated with the *IDS, K2* and *n1*. And submessage *C* will be calculated with the *IDS, K3* and *n2* [3]. For tag authentication, the tag will use submessage *A* and submessage *B* to authenticate the reader and to extract *n1*. The second random number *n2* will be extracted from submessage *C*. The *IDS* and key updating will use the values of *n1* and *n2*. After verifications are finished, the tag will generate two response messages submessage *D* and submessage *E*. Submessage *D* is calculated using the tag's current *IDS, K4*, and *n2*. Generating submessage *D* allows for tag authentication. Submessage *E* will be calculated using current *IDS, n1, n2*, static *ID*, and *KI*. Generating submessage *E* allows the secure transmission of the static *ID* [3].
3. *Index-Pseudonym and Key Updating.* Once mutual authentication between the tag and the reader has occurred, the *IDS* and the key updating stage will be performed. Since tags have very limited computational capabilities, only simple operations such as, bitwise XOR ($\oplus$), bitwise OR ($\vee$), bitwise AND ($\wedge$), and the addition mod 2^m [3] will be used for the purpose of efficiency. The next *IDS* will be calculated using the random number *n2*, *K4*, current *IDS*, and static *ID*. The next *K1* will be calculated using the random number n2, *K1*, *K3,* and static *ID*. The next *K2* will be calculated using the random number n2, *K2*, *K4,* and static *ID*. The next *K3* will be calculated using the random number n1, *K1*, *K3,* and static *ID*. And lastly the next *K4* will be calculated using the random number n_1, *K1*, *K3,* and static *ID*.

C. M2AP

The M2AP [5] was also proposed in 2006 by Peris-Lopez et al. The M^2AP is a mutually authenticating minimalist lightweight protocol that is similar to LMAP, where although the *index-pseudonym* updating differs, the key updating operations remain the same. Both protocols provide security against replay attacks and man in the middle attacks, while also ensuring anonymity [1]. The protocol is as follows:

1. *Tag Identification:* The reader will first identify the tag by sending a *hello* message. The tag will then reply with its current *index-pseudonym* (IDS). Only an authentic reader can use the received IDS to access the tag's secret key, which is 96 bits long divided into four subkeys (*K = K1 || K2 || K3 || K4*) [4].
2. *Mutual Authentication:* For reader authentication, the reader must first generate two random numbers *n1* and *n2*. The reader will calculate the value of submessage *A* with the *IDS,* K_1, and *n1*. Submessage *B* will be calculated with the IDS, K2 and n1. And submessage *C* will be calculated with the *IDS, K3* and *n2*, this value will be used in the *index pseudonym* and key updating phase [4]. For tag authentication, the tag will use submessage *A* and submessage *B* to authenticate the reader and to extract *n1*. The second random number *n2* will be extracted from submessage *C*. The *IDS* and key updating will use the values of *n1* and *n2*. After verifications are finished, the tag will generate two response messages submessage *D* and submessage *E*. Submessage *D* is calculated using the tag's current *IDS, K4*, and *n2*. Generating submessage *D* allows for tag authentication. Submessage *E* will be

calculated using current *IDS, n1*, and static *ID*. Generating submessage *E* allows the secure transmission of the static *ID* [3].

3. *Index-Pseudonym and Key Updating:* Once mutual authentication has occurred between the tag and the reader, the *index-pseudonym* and key updating phase will begin. The next *IDS* will be calculated using the current *IDS*, n1, n2 and the static *ID.* Subkeys *K1, K2, K3,* and *K4*'s next values must also be calculated. The next *K1* will be calculated using the current *K1, n2*, current *K3* and static *ID*. The next *K2* will be calculated using the current *K2, n2*, current *K4* and static *ID*. The next *K3* will be calculated using the current *K3, n1*, current *K1* and static *ID*. Lastly, the next *K4* will be calculated using the current *K4, n1*, current *K2* and static *ID*.

D. SASI

In 2007, Chien proposed an Ultralightweight RFID authentication that provides Strong Authentication and Strong Integrity (SASI) [1]. The protocol involves simple bitwise operation such as bitwise XOR ($\oplus$), bitwise OR ($\vee$), bitwise AND ($\wedge$), left rotate (Rot(x,y)), and the addition mod 2^m. In SASI, security can be achieved by incorporating current secret key values *K1* and *K2*, two random numbers *n1* and *n2*, and the potential next key values $K'1$ and $K'2$in calculating the values of message *C* and message *D* [6]. The protocol is as:

1. Tag Identification: During tag identification, the reader first sends a "*hello*" message to the tag, and upon receiving the message, the tag responds with its next *IDS*. The reader will then compare the received *IDS* with the server's stored *IDS* and once there is a match, then the protocol will proceed to the mutual authentication phase. Otherwise, the reader will probe again, and the tag will respond with its old *IDS* [6].
2. Mutual Authentication: The reader will calculate values *A* || *B* using the values of the *IDS, K1, K2* and *2* random numbers. The reader will use the *IDS, K1, K2* and the two random numbers to calculate the next $K1$ and *K2* values, then use those values to calculate the value of *C*. The reader sends *A* || *B* || C to the tag, and the tag extracts the two random numbers from *A* and *B*. The tag then calculates the next $K'1$ and $K'2$, then uses the values of the next $K'1$, $K'2$, and the extracted two random numbers to verify the value of *C*. Once *C* is verified, a response value *D* will be sent from the tag to the reader, the reader will then verify *D*, then the pseudonym and key updating phase will begin [6].
3. Pseudonym and Key Updating: The new *IDS* will be calculated by using the following operations:

$$IDS = (IDS+ID) \oplus (n_2 \oplus K'1)$$

The new *K1* and *K2* will be updated with values from $K'1$ and $K'2$.

E. KMAP

In 2017, Mujahid et al. [12] proposed the pseudo-Kasami code based Mutual Authentication Protocol (KMAP). The proposed protocol avoids using unbalanced logical operators such as OR and AND, instead KMAP introduces a new Ultralightweight primitive pseudocode called the pseudo-Kasami code. The pseudo-Kasami code (K_c) enhances the protocol messages' diffusion properties and allows the hamming weight of the secrets to be unpredictable and irreversible. KMAP offers excellent protocol functionality and well as high resistance to all possible attacks [15]-[18]. The protocol are as follows:

The pseudo-Kasami code only involves the use of two ultralightweight balanced operators, *XOR* and circulator left rotation (*Rot),* this method proves to be extremely lightweight. The pseudo Kasami code (K_c) can be computed as follows:

Pseudo-Kasami code (K_c) Computation; Let *x* be an '*n*' bit string, such that:

$$X = X_1X_2, \ldots, X_n, X_i \in \{0,1\}, i = 1, 2, \ldots, n$$

Two steps are involved in the computation of the pseudo-Kasami code of *X*, $K_c(X)$:

1. The value *P* and *Seed* are calculated. The tag will extract random numbers n_1 and n_2, and use the two values to calculate *P*, then use the hamming weight *P mod K* to calculate the value of *Seed* [5].
2. The *Seed,* which was calculated in the former step, will then select the number of bits of the *X* string to compete the final value of pseudo-Kasami code, $K_c(X)$ [5]:

(a) New shifted X' string is computed using cyclic left rotation of the bit number of *X*.
(b) Perform XOR operation between *X* and X'.

In KMAP, both the tag and reader have access to values static *ID*, pseudonym *IDS* and two keys K_1 and K_2 of the tag [5]. The protocols are as follows:

1. Tag Identification: The reader will first identify the tag by sending a *hello* message. The tag will then reply with its current *pseudonym* (*IDS*). The reader uses the *IDS* as an index for searching a matched entry in the database. If the *IDS* that the reader received is IDS_{old} then the reader will use $K_{1,old}$ and $K_{2,\ old}$ for the computation of values *A, B* and *C* messages that the reader would send to the tag. If, however, the reader received the IDS_{new} then in computing for *A, B* and *C* response messages, the reader will use $K_{1,new}$ and $K_{2,\ new}$.

The message *A* is calculated using *Rot* operator and the values of n_1, *IDS,* K_1 and K_2. The message *B* is calculated by using *Rot* operator and the values of n_2, K_2, *IDS,* K_1 and n_1. *P* and the *seed* are then calculated, and the values will be used to calculate message *C*. Finally, messages *A*||*B*||*C* are sent to the tag. If received *IDS* cannot be found within the database, then the reader will stop the session with the particular tag.

2. Mutual Authentication: The tag will extract n_1 from message *A* by performing the following operations:

$$n_1 = Rot^{-1}(Rot^{-1}(A,K_2), IDS \oplus K_1)$$

The tag will extract n_2 from message *B* by performing the

following operations:

$$n_2 = Rot^{-1}(Rot^{-1}(B, K_1 \oplus n_1), K_1 \oplus IDS)$$

The tag will calculate seed computation of the pseudo-Kasami Code using $P = n_1 \oplus n_2$ and $hw(P) mod K$. The internal Keys K_1^* and K_2^* compute the local value of message C. When it has been determined that the received C message matches with the local value of C, the tag will generate and transmit message D, then pseudonym and Key updating phase will begin.

3. Pseudonym and Key updating: The new pseudonym will be calculated using:

$$IDS_{new} = Rot(K_c(IDS) \oplus n_1, K_c(n_2))$$

The new keys will be calculated using the following:

$$K_{1,\,new} = K_c(K_1^*)$$

$$K_{1,\,new} = K_c(K_1^*)$$

From 2006 to 2018, many other UMAPs [1]-[5], [10], [12], [25] have also been proposed, however all of them have the similar mathematical structure (but different non-triangular functions & messages design).

III. Security Attacks on UMAPs

In this section, we will discuss some recently proposed security attacks on UMAPs and also give some recommendations for avoidance. The following attack models have been discussed; Desynchronization and Full Disclosure attacks.

A. Desynchronization Attacks

In desynchronization attacks, the attackers modify the communicated messages (to tamper the concealed random numbers) and create such a chaos between the reader and the tag so, they cannot authenticate each other. Some of the desynchronization attack models are presented as follows:

1. Desynchronization attack on LMAP and M2AP

Since, the mathematical structures of these three UMAPs are almost same therefore same attack scenario can desynchronize them. In 2007, Wang and Li [8], [9] reported a desynchronization attack on LMAP and M2AP. The proposed attack model broke the synchronization between the tag and the reader. The attacker modifies the message Cto C' where $C' = C \oplus I_0 \wedge I_0 = [000 \ldots 001]$ (the first 95 MSBs are 0 and last bit is 1). Similarly, the attacker also modifies the Dmessage as well and makes it D'. Now, by the doing this, the tag will get different random number n_2 (which will be now n_2') and it will result in a different D. Hence, the reader will be unable to authenticate the tag and for the next session (after the tag updates its variables with the tampered random numbers) will try to establish a connection with the legitimate reader, the reader will turn down the request (since it does not have the modified IDS).

2. Desynchronization Attack on SASI

To avoid the above-mentioned desynchronization attack, Chien [1] introduced the back up (previous pseudonyms and keys) storage on the tag side. However, Sun et al. [27] highlighted a desynchronization attack in SASI and challenged its security claims.

The proposed attack model [27] is a three-step model. In step one, the attacker sniffs the messages (A, B, and C) of a legitimate session and blocks the message D (going to reader). Hence, the tag updates its pseudonym (IDS_{i+1}) and keys $(K_{1(i+1)}, K_{2(i+1)})$ while the reader will keep the previous values (IDS_i) and keys (K_{1i}, K_{2i}) (since the protocol did not get completed). In step two, the attacker will allow the protocol to be uninterrupted (complete) between the specified tag and the reader. After the successful completion of the protocol, now both the tag and the reader variables are (IDS_{i+2}) and $(K_{1(i+2)}, K_{2(i+2)})$. Now, if we look at the tag's storage, it has two sets of variables $(K_{1i}, K_{1(i+2)}, K_{2i}, K_{2(i+2)})$ & (IDS_i, IDS_{i+2}). In the step three, the attacker initiates the session with the tag (pretends to be a reader). When the tag responds with IDS_{i+2}, the attacker asks for old IDS and the tag responds with its IDS_i. The attacker now replays the pre-captured messages (A, B & C). Since, these messages were captured from a valid session therefore, tag will respond with message D and update its variables. Now, the tag has the following variables in its memory $(K_{1i}, K_{1(i+1)}, K_{2i}, K_{2(i+1)})$ & (IDS_i, IDS_{i+1}). However, the legitimate reader has $K_{1(i+2)}, K_{2(i+2)}$& IDS_{i+2} and therefore it will not authenticate the legitimate tag at this time.

3. Desynchronization attack on KMAP

In 2016, Mujahid et al. [12] introduced the concept of storage of pseudonym and keys on both sides (Tag and Reader) to avoid existing desynchronization attacks. However, Safkhani and Bagheri [26] proposed an efficient desynchronization attack model that requires five consecutive sessions to fully desync the tag and the reader. The success probability of the attack is almost one.

The proposed attack model requires five consecutive sessions between the reader and the tag. In the first session, the adversary intercepts the *IDS* and messages. In the second session, the adversary blocks the messages from reaching at the tag side. In the third session, the adversary blocks all the communication going to the tag and impersonates as tag. The adversary sends a random *IDS* to the reader, which the legitimate reader cannot find in its database and sends another hello to the tag. This time adversary lets the tag to receive the messages and complete the protocol session. In the last two sessions, the adversary impersonates as reader and used the pre-captured messages and completes the sessions with the tag. After completion of these five sessions in the same sequence, both the legitimate reader and the tag will be desynchronized (having different values of *IDS* and keys).

B. Full Disclosure Attacks

In the full disclosure attacks, the attacker intercepts the publicly disclosed messages and uses different attack models

to unveil the concealed secrets. Most of the full disclosure models proposed for UMAPs are ad hoc (not applicable to a broader class of UMAPs) and unstructured [7]-[9], [15], [17], [21]. However, there are few formal (structural) cryptanalysis models that can used to validate wide range of UMAPs. The formal cryptanalysis models (with their application on UMAPs) are described as follows:

1. Norwegian Attack

Peris-Lopez et al. [7] introduced a full disclosure attack called: Norwegian attack in 2010. Initially, the proposed attack model targeted the Yeh et al. [28] protocol, however it is applicable to many other UMAPs including LMAP, EMAP & SASI. The proposed attack model based on the following analogy:

$$n_1 mod L = n_2 mod L$$

(both of the nonce have the same module L value). This assumption further provides:

$$n_1 \oplus n_2 mod L = 0$$

After this attacker sniffs the public messages and simplify the equations based on the above equations. After filtering the results, the attacker can guess the $ID_{conjuncture}$. The reader can find more details of the attack in [7].

2. Tango Attack

In 2010, Caesar [20] proposed a formal cryptanalysis model to validate the security claims of the UMAPs. The proposed attack model involves a cumbersome process of guessing and filtering of variables; however, the success probability of the attack is 100%.

The tango attack mainly exploits the inherent weak diffusion properties of t-functions and improper designing of protocol messages. The attack involves two steps:

i) Finding Good Approximation equations for concealed secret keys and ID.
ii) Comparison of equations with a threshold $T_r(A)$:

$$T_r(A) = \begin{cases} A_i \geq \partial assign1 \\ A_i \leq \partial assign0 \end{cases}$$

where $\partial = \left(\frac{1}{2}\right) \times N_A \times N_s$; $N_A =$ Number of Approximations for the secret; $N_s =$ Number of intercepted sessions.

The tango attack can be used to fully disclose the concealed ID and keys for SASI [1], DAVID Prasad [19], LMAP [3], EMAP [4] and M2AP [5]. The advanced form of the tango attack (Genetic Tango attack) has increased the speed of the attack by finding good approximation equations using genetic algorithms.

3. Recursive Linear and Recursive Differential Attacks

The Recursive Linear Cryptanalysis (RLC) [18] also exploits the weak diffusion properties of t-functions and constructs the linear matrix (equations) of the unknown variables (keys and ID). Then the attacker solves this system of equations bit by bit to find the concealed secrets. Similarly, the Recursive Differential Cryptanalysis (RDC) [18] also construct the system of linear equations for all unknown variables. However, RDC is an active attack where attacker block the protocol sessions so, both the tag and the reader use previous variables for communication and this static nature will not bring new variables in the new sessions beside random numbers. The RDC then finds the differential relationships between the new random numbers with the previous ones and after solving the differential matrix, the random numbers can be retrieved.

The RLC and the RDC attack models have successfully attacked SASI, Yeh et al., LMAP, EMAP, M2AP etc.

C. Recommendation to Avoid Attacks

From the above discussion, we can see that most of UMAPs are vulnerable to many desynchronization and full disclosure attacks. Even though, the researchers have introduced many new non-triangular primitives (Rot, Permutation, Recursive Hash etc.) but the attacks performed in [7]-[9], [15]-[18], [20], [21], [23], [24], [26], [27] have shown their vulnerabilities. Following are some recommendations and suggestions that can improve the designs of UMAPs and can avoid many of the highlighted attacks.

1. Ultralightweight Random Number Generators at Tag side

In all of the previously proposed UMAPs [1]-[6],[10]-[14], [19], [25], the reader generates the random numbers and sends them to the tags. The tag extracts the random numbers and then can verify the reader. In most of the desynchronization attacks, the attacker modifies the random numbers (transmitted by the reader) and tricks the tag to update its pseudonyms and keys. All the desynchronization attacks can be avoided if we integrate an ultralightweight Pseudo Random Number Generator (PRNG) at the tag side. Umar Mujahid and Pedro Paris have proposed some ultralightweight PRNGs which can used in designing of UMAPs.

2. Comprehensive Security Analysis Model (SAM)

A comprehensive SAM is really important that the researchers can use to test the robustness of their UMAPs before getting it designed. The SAM should cover both of the aspects; Protocol functionalities and analysis of protocol (whether it can withstand against security attack or not).

3. Non-Triangular Primitives and Messages Design

Most of the proposed UMAPs [7]-[9], [15]-[18], [20], [21], [23], [24], [26], [27] are either totally based on T-functions or some have used non-triangular primitives in their designs. However, the poor design of the protocol messages made them fully disclosed or desynchronized. There is an immense need of formal analysis (AVISPA, BAN, GNY etc.) models where the designers can formally validate the protocol messages.

IV. Conclusion

In this paper, we have presented a review of some recently proposed UMAPs. We have discussed the detailed working of four UMAPs and also discussed why there was a need or of

the new UMAP designs. Further, we have discussed the security attacks which can make the UMAPs vulnerable. We have discussed desynchronization attacks and some full disclosure attacks models. In this research paper, we have identified a research gap and clearly established that there is not a single existing UMAP (to the best of authors knowledge) which can resist or avoid the discussed security attacks. We have also discussed some recommendations and suggestions that can be really helpful in designing of a secure and robust UMAP. Our next research goal is to use these guidelines and design a sophisticated UMAP.

REFERENCES

[1] Hung-Yu Chien, "SASI: A New Ultralightweight RFID Authentication Protocol Providing Strong Authentication and Strong Integrity", IEEE Transaction on Dependable and Secure Computing, Vol. 4, No. 4, pp. 337 – 340, 2007.

[2] Tian Yun, Gongliang Chen, and Jianhua Li, "A new ultralightweight RFID authentication protocol with permutation", IEEE Communications Letters, Vol.16, No. 5, pp.702-705, 2012.

[3] Pedro Peris-Lopez, Julio Hernandez-Castro et al., "LMAP: A real lightweight mutual authentication protocol for low-cost RFID tags", 2nd Workshop on RFID Security, Austria, pp.100-112, 2006.

[4] Pedro Peris-Lopez, Julio Cesar Hernandez et al., "EMAP: An efficient mutual-authentication protocol for low-cost RFID tags", 1st International Workshop on Information security (OTM-2006), France, pp. 352-361, 2006.

[5] P. Peris-Lopez, J.C. Hernandez-Castro, J.M.E. Tapiador, A. Ribagorda, "M2AP: a minimalist mutual-authentication protocol for low cost RFID tags", International Conference on Ubiquitous Intelligence and Computing, Wuhan China, pp. 912–923, 2006.

[6] Pedro Peris-Lopez et al., "Advances in ultralightweight cryptography for low-cost RFID tags: Gossamer protocol", 9th International Workshop on Information Security Applications, Korea, pp. 56-68, 2009.

[7] Pedro Peris-Lopez et al., "Quasi-linear cryptanalysis of a secure RFID ultralightweight authentication protocol", 6th International Conference on Information Security and Cryptology, China, pp. 427-442, 2011.

[8] Tieyan Li, and Guilin Wang, "Security analysis of two ultra-lightweight RFID authentication protocols", International Information Security Conference, South Africa, pp.109-120, 2007.

[9] Tieyan Li et al., "Vulnerability Analysis of EMAP-An Efficient RFID Mutual Authentication Protocol", 2nd International Conference on Availability, Reliability and Security (ARES 2007), Vienna, pp. 224 – 231, 2007.

[10] Umar Mujahid, M. Najam-ul-Islam, and M. Ali Shami, "RCIA: A New Ultralightweight RFID Authentication Protocol Using Recursive Hash", International Journal of Distributed Sensor Networks, Vol. 2015, No. 642180, 8 pages, 2015.

[11] Umar Mujahid and M.Najam-ul-islam," Ultralightweight Cryptography for Passive RFID Systems", International Journal of Communication Networks and Information Security", Vol.6, No.3, pp.173-181, December 2014.

[12] Umar Mujahid and M. Najam-ul-Islam, "KMAP: A New Ultralightweight RFID Authentication Protocol for passive low cost tags," Wireless Personal Communications (Springer), Vol. 94, Issue 3, pp 725–744, June 2017.

[13] Cai Qingling, Zhan Yiju and Wang, "A minimalist Mutual Authentication Protocol for RFID systems &BAN logic analysis", International Colloquium on Computing, Communication, Control and Management, Guangzhou, pp 449 – 453, 2008.

[14] Hanuage Luo, G. Wen et al., "SLAP: Succinct and Lightweight Authentication Protocol for low-cost RFID system", Wireless Networks (Springer), pp.1-10, 2016.

[15] Raphael C.-W.Phan, "Cryptanalysis of a New Ultralightweight RFID Authentication Protocol-SASI", IEEE Transactions on Dependable and Secure Computing, Vol. 6, No.4, 2009.

[16] Pieter and Michiel, "Analysis of the OpenPGP and OTR protocols", using GNY logic (Online Tutorial), Available from: http://www.ai.rug.nl/mas/ finishedprojects/2007/PieterMichiel/gny.html

[17] Zahra Ahmadian, Mahmoud Salmasizadeh and Mohammad Reza Aref, "Desynchronization attack on RAPP ultralightweight authentication protocol", Information processing letters, Vol.113, No.7, pp. 205-209, 2013.

[18] Zahra Ahmadian, Mahmoud et al., "Recursive Linear and Differential Cryptanalysis of ultralightweight authentication protocols", IEEE Transactions on Information Forensics and Security, Vol.8. No.7, pp. 1140 – 1151, 2013.

[19] Mathieu David and Neeli R. Prasad. "Providing strong security and high privacy in low-cost RFID networks." International conference on Security and privacy in mobile information and communication systems, Italy, pp. 172-179, 2009.

[20] Hernandez-Castro, Julio Cesar et al. "Cryptanalysis of the David-Prasad RFID ultralightweight authentication protocol." Workshop on RFID Security and Privacy, Turkey, pp. 22-34, 2010.

[21] Raphael C.-W.Phan, "Cryptanalysis of a New Ultralightweight RFID Authentication Protocol-SASI", IEEE Transactions on Dependable and Secure Computing, Vol. 6, No.4, 2009.

[22] EPC Global- RFID Identity protocols, Generation-2 UHF RFID Specification for Air Interface Version 2.0.0, 2013.

[23] Julio C. Hernandez et al, "Cryptanalysis of the SASI ultralightweight RFID authentication protocol with modular rotations", ArXiv, Cryptography and Security, Report; Report No. 0811.4257, http://arxiv.org/abs/0811.4257, 2008.

[24] Zeeshan Bilal, Ashraf Masood, and Firdous Kausar "Security analysis of ultra-lightweight cryptographic protocol for low-cost RFID tags: Gossamer protocol", 12th International Conference on Network-Based Information Systems, Indianapolis, USA, pp. 260-267, 2009.

[25] Xu Zhuang · Yan Zhu · Chin-Chen Chang, "A New Ultralightweight RFID Protocol for Low-Cost Tags: R2AP", Wireless Personal Communications, Vol 79, pp.1787-1802, 2014.

[26] Masoumeh Safkhani and Nasour Bagheri, "Generalized Desynchronization Attack on UMAP: Application to RCIA, KMAP, SLAP and SASI+ protocols", eprint.IARC, Article 905, 2016.

[27] Hung Min Sun et al. "On the Security of Chien's Ultralightweight RFID Authentication Protocol" IEEE Transactions on Dependable and Secure Computing 8(2):315-317 · March 2011

[28] K.-H. Yeh, N. W. Lo, and E. Winata, "An efficient ultralightweight authentication protocol for RFID systems," pp. 49– 60, Proceedings of the Workshop on RFID Security and Privacy, Istanbul, Turkey, 2010