# Cryptanalysis of RFID Ultra-lightweight Protocols and Comparison between its Solutions Approaches

Muhammad Zubair, Engr.Umar Mujahid, Najam-ul-Islam and Jameel Ahmed

**Abstract- Radio Frequency Identification (RFID) is one of the rapidly growing technologies in field of universal computing, because of its cost effectiveness and small size. Ultra lightweight protocol consists of simple bitwise operation such as XOR, OR, AND, Left Rotate (x, y) are required on tag, which make it cost effective and simple. To make the system practically cost effective and robust against various security attacks, Ultra light weight authentication protocols are used that involve logical operators like XOR, AND, OR and Rot functions as compare to Hashing and other cryptographic algorithms. In this paper we will perform cryptanalysis of ultra lightweight protocols (SASI and Gossamer) and find vulnerabilities in these protocols. On the basis of proposed attacks we will design a new mutual authentication protocol, which overcome these attacks, and will provide better security feature using simple bitwise operation.**

**Index Terms: RFID Attacks, Ultra lightweight protocols, vulnerabilities, proposed solution**

## I. INTRODUCTION

RFID consist of tags, readers, and a backend system as shown in Fig.1. The reader accesses the information which is contained within the tag using radio transmission. By using index, the reader can therefore get back the matching record from the database. RFID tag can be Active or passive, if tag broadcast a new signal then tag is active tag. Active tag consists of on-board battery that broadcast the signal. Passive tag doesn't contain on-board battery. RFID technology is a widely adopted computerization technology for manufacturing industry, supply chain management, transportation payment and even passport identification.

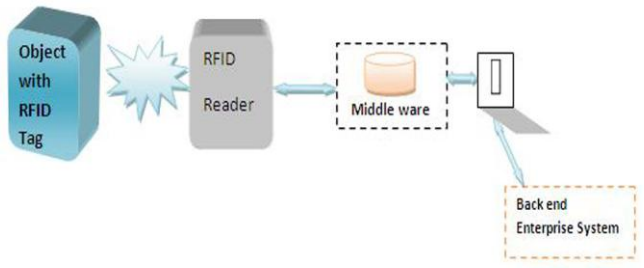


Fig. 1. RFID Architecture

Muhammad Zubair, Engr.Umar Mujahid,.Najam-ul-islam and Jameel Ahmed are affiliated with Bahria University Islamabad campus zubair3377@yahoo.com. Manuscript received February 28, 2012; revised April 30, 2012 and December 5, 2012

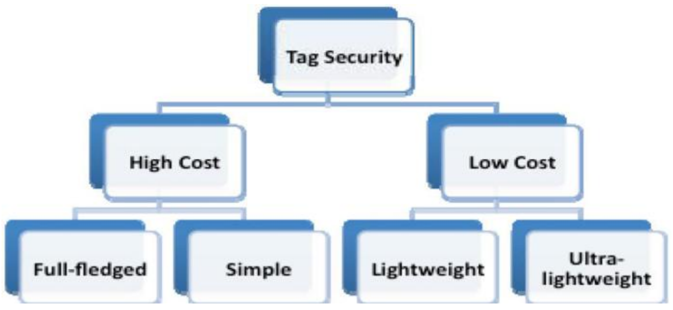


Fig. 2. Classification of security solution for tag [1]

Fig. 2 shows that the tags are divided into two categories that are high-cost and low-cost. Chien [3], further divides the high-cost tags into two subcategories: "full- ledged" and "simple". Full-fledged tags support on-board straight cryptography such as symmetric encryption, cryptographic one-way hash functions and even public key cryptography. On the other side, simple tags can support random number generators and one-way hash functions.

Similarly, low-cost tags can be categorised into "lightweight" and "ultra lightweight". Lightweight tags can only carry out simple functions like Cyclic Redundancy Check (CRC) but not the hash function. Ultra-lightweight tags only perform simple bitwise operations like XOR, AND, OR, etc. Class1 Gen2 tags fall under the lightweight tag category.

Ultra-lightweight authentication protocols on the other hand, one of the efficient protocols because of cost effective and uses simple bitwise operation. A series of ultra-lightweight mutual authentication protocols, referred to as the UMAP family, concerning only bitwise operations and modular addition has been proposed in [7], [8], [9] and [14]. Unfortunately, after few months, the vulnerabilities of these protocols have been showed (e.g., [10], [11],[2]).

When designing a real lightweight authentication protocol for low cost RFID tags, a number of Challenges and Attacks arise due to limited computational as show in Fig. 3. We focus our attention on a new ultra-lightweight authentication protocol, recently proposed in [3], to provide strong authentication and strong integrity data protection. It is a good representative of the ultra-lightweight methodology design. We start by showing some weaknesses in the SASI and Gossamer protocol, and then, we describe how such weaknesses, through a sequence of simple steps, can be used to compute in an efficient way all secret data used for the authentication process.

One of the aims of SASI's design was to achieve struggle to tracking and claimed to offer more security than earliest ultra-lightweight RFID protocols like LMAP, M2AP and EMAP and Later on, Phan [4] has described a simple probabilistic passive attack against the un-traceability property of SASI. Finally, in [5], Cao et al. have proposed a de-synchronization attack. Gossamer protocol is finally published ultra-lightweight mutual authentication protocol. The two main strengths of Gossamer protocol were:

a) *ROTbits* (Left Rotation of Bits) function and b) *MIXbits* (Mixing of Bits) function. Earlier protocols had common vulnerabilities that they were based only on triangular (T-functions) functions. Whereas, these two functions are non-triangular functions providing strong authentication and integration properties and these both are also implemented as economical operations. Therefore, Gossamer is believed to provide better security in ultra-lightweight protocol.
The organization of the rest of the paper is as follows, Section 1 for literature Survey. Section 2 reviews of the SASI and Gossamer protocol. Section 3 attacks on this protocol. Section 4 proposed solution and Section 5 Result and compare solution.
.

## II. LITERATURE SURVEY

In [7], [8], [9] protocols, which is proposed by peris-Lopez al. simple operator such as XOR, OR, AND, and addition mod $2^m$ rather than to use costly operator such as multiplication and hash assessment are not necessary to use, and random number are generated by the reader. These protocols have main three steps:

1. Tag identification.
2. Mutual authentication.
3. Key updating.

In [11], Li and Wang and in [10], Li and Deng explain the de-synchronization attack and full-disclosure attack on these protocol, while they use just simple bitwise operation for the intention of authentication. They discover De-synchronization attack and full-disclosure attack against their protocol. So we discover that the preceding protocols [7], [8], [9] only present weak authentication and weak integrity defence, which creates vulnerabilities.

In [3], Chien state that the SASI protocol overcomes to the de-synchronization attack, man-in-the-middle attack and provide forward security using simple bitwise operation. But, we will show that the SASI protocol is still vulnerable to DoS attacks. In our attacks, a man-in-the-middle can destroy the synchronization between the database and the tag. Thus, the tag cannot be further authenticated by the database. The RFID system will be involved in DoS state and unable to guarantee availability. However, if we assume that an attacker compromises a tag, the attacker can infer the previous secret data and keys of the same tag and trace the past communication. Thus, the SASI protocol does not provide forward anonymity.

In [6], Paolo D'Arco and Alfredo De Santis, tells about weaknesses in the SASI protocol, and then, they describe vulnerabilities and using these vulnerabilities, through a sequence of simple steps, How all secret data used for the authentication process. Specifically, they describe three attacks:

1. A de-synchronization attack, through which an attacker can break the synchronization between the Reader and the Tag.
2. An identity disclosure attack, through which an attacker can compute the identity of the Tag.
3. A full disclosure attack, which enables an attacker to retrieve all secret data stored in the Tag.

Among All the protocols mentioned above, none of the protocol overcomes these vulnerabilities, so at the end in [12], Gossamer ultra-lightweight protocol was proposed. This protocol used two functions ROT Bits and Mix Bits using these functions author claimed that he can avoid the problem of past. This protocol is consisting of three stages as mentioned above. But we will do comprehensive security analysis of Gossamer protocol, because still Gossamer protocol is vulnerable to DOS, memory and computation exhaustive, de-synchronization, replay attacks.

In [13], the author analysed the security of Gossamer protocol. He do various types of attacks such as, DOS attack, memory and computation exhaustive attack, de-synchronization and reply attack. This attack shows the weak spot of the Gossamer protocol. He proposed a protocol and claims that this protocol is more efficient compare to Gossamer protocol.

In [14], the author describes the Gossamer protocol and claimed it is weak to one particular type of DoS, namely, DoS by de-synchronization. Additionally, author presents a novel method that extends the Gossamer protocol to avoid DoS attacks in general, and the de-synchronization DoS attack.

Furthermore, author built a java base simulation framework to explain the proof of- concept implementation of the proposed technique.

## III. SASI AND GOSSAMER PROTOCOL

In the SASI protocol [3], the channel between the reader and the database server is understood to be safe, but the channel between the reader and the tag is vulnerable to all the possible attacks. The pseudonym IDS is used to find a matched record in the database server.

This protocol compromise of three stages as shown in Fig. 4, tag identification phase, mutual authentication phase, and pseudonym updating and key updating phase.

### A. *Tag identification*

The reader first sends the read request "hello" message to the tag. After getting the read request from the reader, the tag respond with ($IDS_{next}$) to reader. If the reader could find a matched record in the database, it steps into the mutual authentication phase; otherwise, it checks out again and the tag responds with its old IDS ($IDS_{old}$).

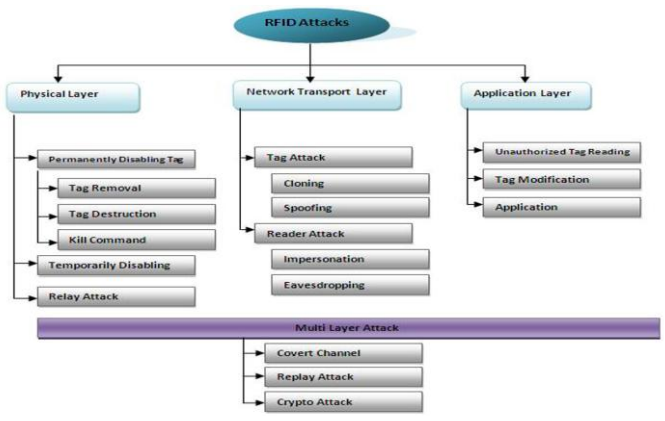
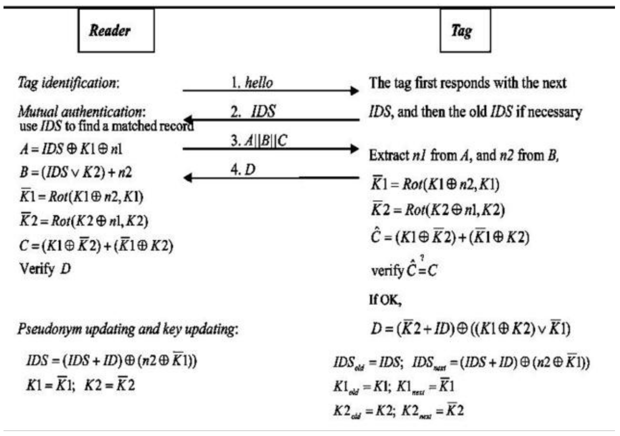


Fig.3 Classification of RFID



Reader
Tag
Tag identification:
1. hello
The tag first responds with the next IDS, and then the old IDS if necessary
Mutual authentication: use IDS to find a matched record
2. IDS
3. A||B||C
4. D
$A = IDS \oplus K1 \oplus n1$
$B = (IDS \vee K2) + n2$
$\bar{K}1 = Rot(K1 \oplus n2, K1)$
$\bar{K}2 = Rot(K2 \oplus n1, K2)$
$C = (K1 \oplus \bar{K}2) + (\bar{K}1 \oplus K2)$
Verify D
Extract n1 from A, and n2 from B,
$\bar{K}1 = Rot(K1 \oplus n2, K1)$
$\bar{K}2 = Rot(K2 \oplus n1, K2)$
$\hat{C} = (K1 \oplus \bar{K}2) + (\bar{K}1 \oplus K2)$
verify $\hat{C} \overset{?}{=} C$
If OK,
$D = (\bar{K}2 + ID) \oplus ((K1 \oplus K2) \vee \bar{K}1)$
Pseudonym updating and key updating:
$IDS = (IDS + ID) \oplus (n2 \oplus \bar{K}1))$
$K1 = \bar{K}1; \quad K2 = \bar{K}2$
$IDS_{old} = IDS; \quad IDS_{next} = (IDS + ID) \oplus (n2 \oplus \bar{K}1))$
$K1_{old} = K1; \quad K1_{next} = \bar{K}1$
$K2_{old} = K2; \quad K2_{next} = \bar{K}2$

Fig .4. SASI Protocol

## *A. Mutual authentication*

Then the reader uses the matched IDS and two generated random numbers n1 and n2 to compute $A=IDS \oplus K_1 \oplus n_1$, $B=(IDS \vee K_2)+n_2$, $K_1'=Rot(K_1 \oplus n_2, K_1)$, $K_2'=Rot(K_2 \oplus n_1, K_2)$, $C=(K_1 \oplus K_2')+(K_1' \oplus K_2)$, and then sends (A||B||C) to the tag. After receiving (A||B||C) from the reader, the tag first extracts $n_1$ from A, extracts $n_2$ from B, computes $K_1'$, $K_2'$ and $C'$. If $C'$ does not match with the received C, the message is aborted; otherwise, that means the reader is authenticated. Then the tag computes $D=(K_2'+ID) \oplus ((K_1 \oplus K_2) \vee K_1')$, and sends it to the reader. Upon receiving D, the reader uses its local values to verify D. If the verification fails, the message is aborted.

## *B. Pseudonym updating and key updating*

After the reader and the tag authenticated each other, the server updates its local values ($IDS=(IDS+ID) \oplus (n_2 \oplus K_1')$, $K_1=K_1'$, $K_2=K_2'$), and the tag then updates its local values as ($IDS_{old}=IDS$, $IDS_{next}=(IDS+ID) \oplus (n_2 \oplus K_1')$, $K_{1old}=K_1$, $K_{1next}=K_1'$, $K_{2old}=K_2$, $K_{2next}=K_2'$).

The SASI protocol [4] is designed for low-cost and very low-cost tags, which do not support random number generation. Because the random numbers for each tag reading are only generated by one end of the communication — the reader, the following attacks are difficult to avoid:

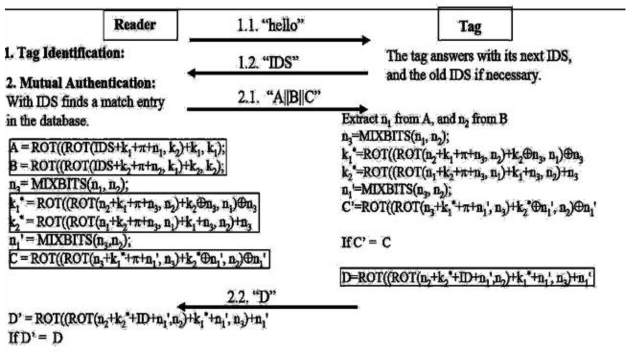
**Denial of Service Attack:** An attacker can first record the message ($A_1||B_1||C_1$) and intercept the message $D_1$ in the 1st round, then the tag updates its local values ($IDS_{old}=IDS_1$, $IDS_{next}=IDS_2$), while the reader does not ($IDS=IDS_1$). After that, the reader and the tag are allowed to communicate without intervening. Because $IDS_2$ cannot be found in the server, the tag responses $IDS_1$ in the next round to pass the reader's authentication. Thus, the tag will then update its local values $IDS_{old}=IDS_1$, $IDS_{next}=IDS3$, and the server will update its IDS from $IDS_1$to IDS3. Finally, the attacker impersonates the legal reader to send "hello message" to the tag, and the tag responses with $IDS_{next}$ (=IDS3).

The impersonated reader probes again then the tag responses with its $IDS_{old}$(=$IDS_1$). The impersonated reader now can replay the recorded ($A_1||B_1||C_1$) previously computed by the legal reader with $IDS_1$ to pass the tag's authentication, and then the tag will update its local values ($IDS_{old}=IDS_1$, $IDS_{next}=IDS_2$). Now, the value of IDS stored in the server is IDS3, which is different with the values stored in the tag. That makes the reader and the tag out of synchronization. Therefore, the SASI protocol is still vulnerable to DoS attacks.

**Tracking attack:** If an attacker impersonates a legal reader to send "hello message" to the tag successively, the tag will response with $IDS_{next}$ in the 1st round and then response with the same old $IDS_{old}$ in the following rounds. The same old value of $IDS_{next}$ can thus be used to track the tag.

## *C. Gossamer Protocol*

The SASI protocol proposed by Chien [3] assures this requirement as it uses the left rotation operation. However, it was observed that after more than a few successive interpretations, a passive attacker can break the secret tag id. The Gossamer protocol proposed by Peris-Lopez [15] to prevent DoS attacks on RFID systems belong to the UMAP family of protocols. These protocols are based on bitwise logical operations such as XOR, OR and AND. The reader needs to produce the pseudorandom number and tags make use of them for generating messages. Gossamer uses all the bitwise operations used by SASI (XOR, AND, OR). However, it makes use of a dual rotation compared to one left rotation in SASI. In addition, the protocol also uses a particularly designed lightweight function known as MiXBits which compromises of bitwise right shift and bitwise additions, to strengthen the security.



Reader
Tag
1.1. "hello"
1. Tag Identification:
1.2. "IDS"
The tag answers with its next IDS, and the old IDS if necessary.
2. Mutual Authentication: With IDS finds a match entry in the database.
2.1. "A||B||C"
$A=ROT((ROT(IDS+k_1+\pi+n_1, k_2)+k_1, k_1);$
$B=ROT((ROT(IDS+k_2+\pi+n_2, k_1)+k_2, k_2);$
$n_3=MIXBITS(n_1, n_2);$
$k_1^*=ROT((ROT(n_2+k_1+\pi+n_3, n_2)+k_2 \oplus n_3, n_1) \oplus n_3$
$k_2^*=ROT((ROT(n_1+k_2+\pi+n_3, n_1)+k_1+n_3, n_2)+n_3$
$n_1'=MIXBITS(n_3, n_2);$
$C=ROT((ROT(n_3+k_1^*+\pi+n_1', n_3)+k_2^* \oplus n_1', n_2) \oplus n_1'$
Extract $n_1$ from A, and $n_2$ from B
$n_3=MIXBITS(n_1, n_2);$
$k_1^*=ROT((ROT(n_2+k_1+\pi+n_3, n_2)+k_2 \oplus n_3, n_1) \oplus n_3$
$k_2^*=ROT((ROT(n_1+k_2+\pi+n_3, n_1)+k_1+n_3, n_2)+n_3$
$n_1'=MIXBITS(n_3, n_2);$
$C'=ROT((ROT(n_3+k_1^*+\pi+n_1', n_3)+k_2^* \oplus n_1', n_2) \oplus n_1'$
If C' = C
$D=ROT((ROT(n_2+k_2^*+ID+n_1', n_2)+k_1^*+n_1', n_3)+n_1'$
2.2. "D"
$D'=ROT((ROT(n_2+k_2^*+ID+n_1', n_2)+k_1^*+n_1', n_3)+n_1'$
If D' = D

Fig .5. Gossamer Protocol

The Gossamer protocol consists of three stages, namely, tag identification phase, mutual authentication phase, and updating phase. Each one tag stores three parameters, a static identifier (*ID*), an index pseudonym (*IDS*), and two keys ($k_1$, $k_2$). These values for all tags are also stored in the back-end database.

***Tag Identification*****:** The tag reacts to the hello from the reader with its possible next *IDS*. If the reader is able to find a match in the database, it starts to the next phase. Otherwise, the step is again attempted with the old IDS.

***Mutual Authentication***: The reader retrieves the keys $k_1$ and $k_2$ linked to the tag, generates nonce's $n_1$ and $n_2$, builds A||B||C and sends it to the tag. From sub messages A and B, the tag extracts nonce's $n_1$ and $n_2$, computes $n3$, $k_1$*, $k_2$*, $n_1'$ a local version of C (C´) and D. $k_1$* and $k_2$* are updated versions of $k_1$ and $k_2$, and $n_1'$ is generated from $n_2$ and $n3$. C´ is compared with the received value of C, which if identical authenticates the reader. The tag then sends message D to the reader. The reader evaluates the received value with its computed value of D, thus verifying the authenticity of the tag.

***Index-Pseudonym and Key Updating*****:** Both the tag and the backend database update the values of the IDS and the keys after the successful pass of the protocol. The Gossamer protocol has been analyzed to be effective in providing data confidentiality, tag anonymity, mutual authentication and data integrity, forward security, robustness against replay attacks, and DoS attack prevention.

### *D. Vulnerabilities in Gossamer*

In 2008, Peris-Lopez et al. [12] developed an ultra-lightweight authentication mechanism, called Gossamer protocol, which is inspired by SASI scheme [3]. The Gossamer protocol is developed to eliminate the security vulnerabilities, i.e. de-synchronization and disclosure attacks [10, 11, 2], on SISA protocol.

### *E. De-synchronization attack on Gossamer protocol*

Similarly, Gossamer protocol cannot defend against the de-synchronization attack. The corresponding malicious procedures are as follows. First, a given synchronized tag in which the secret information (IDSnew, $k_1$_new, $k_2$_new) maintained at the tag side equals to the values (IDS, $k_1$, $k_2$) stored in the backend database is assumed. Now we suppose the reader intends to query the tag. During a normal operation process of Gossamer protocol, the attacker eavesdrops and records the transmitted messages A||B||C. At the end of the protocol, the attacker interrupts the message D and these results in that the backend database will not update the information (IDS, $k_1$, $k_2$) associated with the tag. However, the tag will update the secret information as follows. For clarity, we denote the old secret information as ($IDS_1$, $k_{1\_1}$, $k_{2\_1}$) and updated information as ($IDS_2$, $k_{1\_2}$, $k_{2\_2}$) at current session. ($IDS_{old}$, $k_{1\_old}$, $k_{2\_old}$) stored in the tag= ($IDS_1$, $k_{1\_1}$, $k_{2\_1}$) (IDSnew, $k_{1\_new}$, $k_{2\_new}$) stored in the tag= ($IDS_2$, $k_{1\_2}$, $k_{2\_2}$) (IDS, $k_1$, $K_2$) stored in the server/database= ($IDS_1$, $k_{1\_1}$, $k_{2\_1}$) Next, the attacker lets the reader and the tag run the Gossamer protocol without being intervened. In this communication process, the tag will utilize the old values, i.e. $IDS_{old}$, $k_{1\_old}$ and $k_{2\_old}$, to communicate with the reader as the IDS stored in the backend database are the old one. After performing all authentication procedures, the database will update the corresponding values of the tag to a new one (IDS3, $k_1$_3, $k_2$_3) due to two new random nonce values generated by the reader. At the tag side, the secret information will update as follows. ($IDS_{old}$, $k_{1\_old}$, $k_{2\_old}$) stored in the tag= ($IDS_1$, $k_{1\_1}$, $k_{2\_1}$) (IDSnew, $k_{1\_new}$, $k_{2\_new}$) stored in the tag= (IDS3, $k_1$_3, $k_2$_3) Finally, the attacker utilizes its own legal reader to inquire the tag. The tag first replies IDSnew, which is IDS3, and then sends $IDS_{old}$, which is $IDS_1$, when the attacker pretends that he/she cannot find the IDS3 in the backend database and requests the $IDS_{old}$ value. The attacker then transmits the previously eavesdropped values A||B||C to the tag. Since these values are computed by the legal reader with $IDS_1$ previously, the tag cannot distinguish whether these values are truly issued from a legal user or not, and accepts these values. After performing the update procedures, the secret information at the tag side will be as follows.

($IDS_{old}$, $k_{1\_old}$, $k_{2\_old}$) stored in the tag= ($IDS_1$, $k_{1\_1}$, $k_{2\_1}$)

(IDSnew, $k_{1\_new}$, $k_{2\_new}$) stored in the tag= ($IDS_2$, $k_{1\_2}$, $k_{2\_2}$)

Obviously, the secret information stored in the tag side and in the backend database side is out of synchronization now.

(IDS, $k_1$, $k_2$) stored in the server/database= (IDS3, $k_1$_3, $k_2$_3)
Other than De-synchronization attack in Gossamer Protocol is:

1. DOS attack.
2. Memory and computation exhaustive on tag.
3. DoS attack on Reader.

Attacker sends a random string of IDS with high frequency

## IV. PROPOSED SOLUTION

A new proposed solution for above vulnerabilities is suggested. This will help avoid the attacks such as DOS attack, De-synchronization attacks, memory and computation exhaustive attacks, Replay attack and Tracing attack. To conquer these vulnerabilities, a counter is used in tag. This may be used with each message and incremented with its reply by the tag while transfer IDS. When counter reaches at 4, protocol should be completed as shown in Fig. 3, but in case of connection problem, here counter is set the value for example 7. When counter will arrived at 7 than it understood that that tag will be access 7 times but complete protocol yet doesn't completed than it will indicate a DOS attack.

### *A. ANALYSIS of Proposed Solution*

Above mention proposed result can avoid active attacks, like DoS or de-synchronization. A relative security analysis is presented as follows:

### *B. Analysis of Denial of Service Attacks*

Gossamer protocol does not give security in opposition to a DoS attack, which is an active attack. By addition of a counter in the Tag, DoS, and memory and computation exhaustive attacks can be avoided. In the same way, a kind of DoS attack on the Reader utilizes its weakness to re-communicate lest of backscattered IDS not known, is avoided in the proposed solution given above. To overcome the flaws and as a effect to overcome the DoS attack, a counter is used with every hello message and incremented with its reply by the Tag as sending IDS. An overflow condition can be employed depending on the consistency of the network connection.

### C. *Analysis of De-Synchronization Attacks and Message Replay*

To get the better of de-synchronization attack, it is recommended that a message concerning shared secrets, be sent by the Reader to the Tag. This message will make sure that the Reader received and confirmed the messages D||E sent by the Tag. Shared secrets will make sure that this message is sent by the legal Reader shown in Fig. 7. This will keep away from the Tag updating without ensuring whether messages E were confirmed by the Reader or not. If E is confirmed properly, message F is sent, if not message to abandon protocol is sent and the Tag will not vary its internal condition, hence avoiding de-synchronization. In case D||E does not reach the Reader or it is not confirmed, the Reader will produce a protocol Discard message, than conditions are not changed. If D||E are confirmed than reader sends F, now in case F doesn't reach the rag, the tag will not update its states as shown in Fig..

## V. PERFORMANCE COMPARISON

Table given below briefly sums up the result of security requirements and cost computation. This table shows the comparison among our proposed solution, SASI and Gossamer protocols in according with security and system efficiency requirements. It appears that, our proposed solution is better to SASI and Gossamer protocols by supporting all security requirements with lower computation cost and less tag memory exploiting.

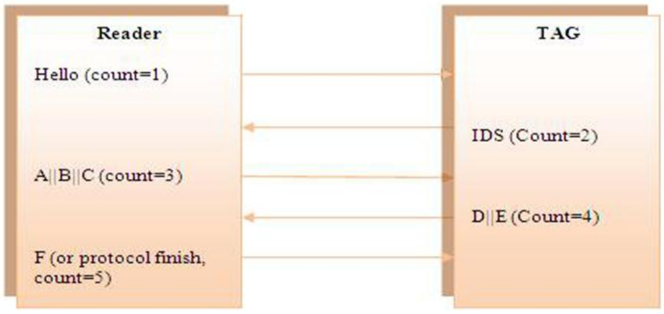


Fig .6. Block Diagram of proposed Solution, Counter in a Tag

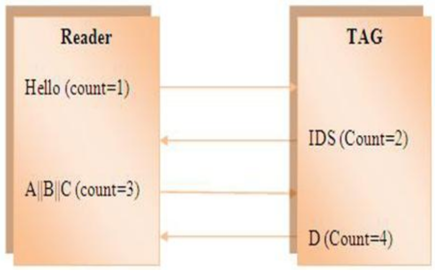
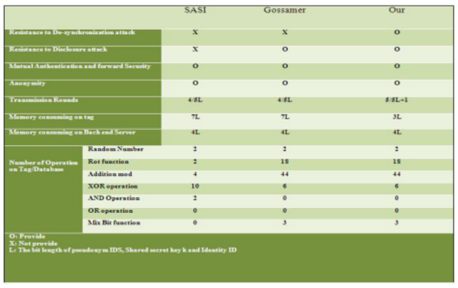


Fig. 8. Performance comparison



| | | SASI | Gossamer | Our |
|---|---|---|---|---|
| Resistance to De-synchronization attack | | X | X | O |
| Resistance to Disclosure attack | | X | O | O |
| Mutual Authentication and forward Security | | O | O | O |
| Anonymity | | O | O | O |
| Transmission Rounds | | 4/5L | 4/5L | 5/5L+1 |
| Memory consuming on tag | | 7L | 7L | 3L |
| Memory consuming on Back end Server | | 4L | 4L | 4L |
| Number of Operation on Tag/Database | Random Number | 2 | 2 | 2 |
| | Rot function | 2 | 18 | 18 |
| | Addition mod | 4 | 44 | 44 |
| | XOR operation | 10 | 6 | 6 |
| | AND Operation | 2 | 0 | 0 |
| | OR operation | 0 | 0 | 0 |
| | Mix Bit function | 0 | 3 | 3 |

O: Provide
X: Not provide
L: The bit length of pseudonym IDS, Shared secret key k and Identity ID

## VI. CONCLUSION

Radio Frequency Identification (RFID) is an automated identification technology; however transmission of data in air is vulnerable. In 2007 Chein proposed a SASI protocol having simple bitwise operation but this protocol is vulnerable to De-synchronization and DoS attack. In [12] Peris-Lopez, Pedro and Hernandez-Castro (2008), Gossamer ultra-lightweight protocol was proposed. This protocol used two function ROT Bits and Mix Bits using these function author claimed that can avoid the problem of past. This paper has proposed alteration of Gossamer protocol, the proposed solution shows that the added alteration increase security level of Gossamer and prevent eavesdropping.

## REFERENCES

[1] VinodRamachandra ,MusfiqRahman, and SrinivasSampalli*, "Lightweight Matrix-based Authentication Protocol for RFID "Software, Telecommunications and Computer Networks (SoftCOM), 2011 19th International Conference on, September 2011

[2] Chien and C. Hwang, "Security of Ultra-Lightweight RFID Authentication Protocols and Its Improvements," ACM SIGOPS Operating Systems Rev., vol. 41, no. 4, pp. 83-86, July 2007.

[3] H.-Y. Chien, "SASI: A New Ultra-lightweight RFID Authentication Protocol Providing Strong Authentication and Strong Integrity", *IEEE Transactions on Dependable and Secure Computing*, vol. 4, no. 4, pp. 337-340, 2007

[4] R.C.-W. Phan, "Cryptanalysis of a New Ultra-lightweight RFID Authentication Protocol—SASI," IEEE Trans. Dependable and SecureComputing, vol. 6, no. 4, pp. 316-320, Oct.-Dec. 2009.

[5] T. Cao, E. Bertino, and H. Li, "Security Analysis of the SASI Protocol," IEEE Trans. Dependable and Secure Computing, vol. 6,no. 1, Jan.-Mar. 2009.

[6] Paolo D'Arco and Alfredo De Santis, "On Ultra-lightweightRFID Authentication Protocols", IEEE Trans.On Dependable and secure Computing, vol. 8, no. 4, JULY/AUGUST 2011.

[7] P. Peris-Lopez, J.C. Hernandez-Castro, J.M. Estevez-Tapiador, and A. Ribagorda, "LMAP: A Real Lightweight Mutual Authentication Protocol for Low-Cost RFID Tags," In *Proc. Second Workshop RFID Security*, July 2006.

[8] P. Peris-Lopez, J.C. Hernandez-Castro, J.M. Estevez-Tapiador, and A. Ribagorda, "EMAP: An Efficient Mutual Authentication Protocol

forLow-Cost RFID Tags," In *Proc. OTM Federated Conf. and Workshop: IS Workshop*, Nov. 2006.

[9] P. Peris-Lopez, J.C. Hernandez-Castro, J.M. Estevez-Tapiador, and A. Ribagorda, "M2AP: A Minimalist Mutual-Authentication Protocol for Low-Cost RFID Tags," In *Proc. Int'l Conf. Ubiquitous Intelligence and Computing (UIC'06)*, pp. 912-923 2006.

[10] T. Li and R.H. Deng, "Vulnerability Analysis of EMAP-An Efficient RFID Mutual Authentication Protocol," Proc. Second Int'l Conf, Availability, Reliability, and Security (AReS '07), 2007.

[11] T. Li and G. Wang, "Security Analysis of Two Ultra-Lightweight RFID Authentication Protocols," Proc. 22nd IFIP TC-11 Int'l Information Security Conf. May 2007

[12] Peris-Lopez, Pedro, Hernandez-Castro, J. Cesar, Estevez-Tapiador, J. M., Ribagorda, and Arturo, "Advances in Ultra-lightweight Cryptography for Low-cost RFID Tags: Gossamer Protocol," in *Workshop on Information Security Applications*, ser. Lecture Notes in Computer Science. Jeju Island, Korea: Springer-Verlag, September 2008.

[13] Z. Bilal, A. Masood, F. kausar, "Security Analysis of Ultra-light Cryptographic Protocol for Low-cost RFID tags: Gossamer Protocol", international conference on Network-Based information Systems 2009.

[14] Deepak Tagra, Musfiq Rahman and Srinivas Sampalli," Technique for Preventing DoS Attacks on RFID Systems" international conference telecommunication and computer networks, 2010.

[15] Pedro Peris-Lopez. "Advances in Ultra-lightweight Cryptography for Low-Cost RFID Tags: Gossamer Protocol", Lecture Notes in Computer