


Review

Survey on Prominent RFID Authentication Protocols for Passive Tags

Rania Baashirah * and Abdelshakour Abuzneid * 

Department of Computer Science and Engineering, University of Bridgeport, Bridgeport, CT 06604, USA

* Correspondence: rbaashir@my.bridgeport.edu (R.B.); abuzneid@bridgeport.edu (A.A.);

Tel.: +1-(203)-576-4113 (A.A.)

Received: 1 September 2018; Accepted: 19 October 2018; Published: 22 October 2018



Abstract: Radio Frequency Identification (RFID) is one of the leading technologies in the Internet of Things (IoT) to create an efficient and reliable system to securely identify objects in many environments such as business, health, and manufacturing areas. Recent RFID authentication protocols have been proposed to satisfy the security features of RFID communication. In this article, we identify and review some of the most recent and enhanced authentication protocols that mainly focus on the authentication between a reader and a tag. However, the scope of this survey includes only passive tags protocols, due to the large scale of the RFID framework. We examined some of the recent RFID protocols in term of security requirements, computation, and attack resistance. We conclude that only five protocols resist all of the major attacks, while only one protocol satisfies all of the security requirements of the RFID system.

Keywords: RFID; security; privacy; authentication; passive tag; security threats; security attacks; IoT; lightweight protocol

1. Introduction

The wireless sensor network has expanded recently to employ new technologies in the Internet of Things (IoT). The purpose of this evolution is to create a low-cost, reliable, and secure communication network for current and future applications using radio waves in the most convenient way. Radio Frequency Identification (RFID) is a technology where the detection of the electromagnetic signals in the wireless sensor network identifies objects or people. Hundreds and thousands of RFID applications have been used to improve business efficiency and productivity in a variety of business operations, including supply chain management, access control limitation, product tracking, merchandise allocation, toll collection, and so on. It is also considered an integral part of daily life where its applications not only are limited to business activities, but also daily life activities that are integrated into cell phones, household, automobile, etc.

Although the basic concept of RFID is similar to barcodes in identifying the items using the data stored in barcodes, RFID technology has vital benefits over barcodes. It does not require physical contact with the objects, allows scanning multiple and different types of barcodes using one signal, has the ability to read and write on the tag multiple times [1], and enables identifying objects in different climates such as fog and snow, and packaging conditions such as ice, perishable food, and liquids [2].

RFID is considered a significant structure for future market development. Many business enterprises and manufactures nowadays in the supply chain, including banks, transportation, government, agriculture, food safety, health care, and mass production, are using RFID to automate their product identification faster in different conditions to improve their business efficiency and customer service experience.

2. System Architecture and Communication Model

The basic system of RFID includes a receiver (reader), transponder (tag), and back-end database (server) to store and manage data. The RFID tag is a label that is placed into the object to be identified and located among hundreds and thousands of objects. It consists of a small antenna attached to a microchip with a small memory to store the object's identity and data [3]. The RFID reader is a scanner placed in a fixed location to interrogate the tag whenever the tag exists in the scanning environment. The back-end database server operates as a data processor that manages, controls, and stores the data from the tag and reader. An RFID system is depicted in Figure 1 [4].

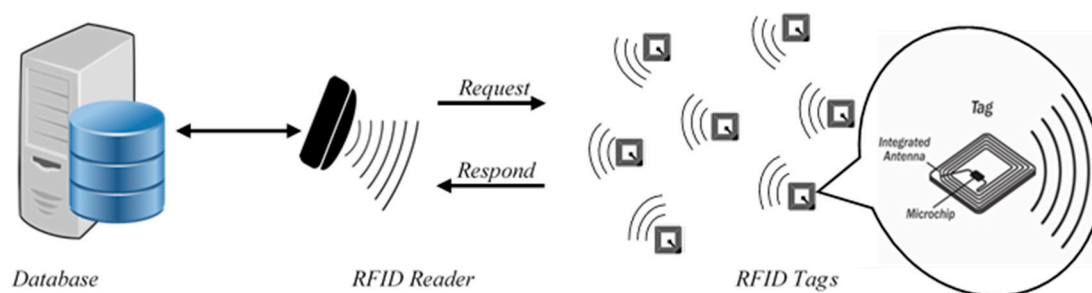


Figure 1. Basic Radio Frequency Identification (RFID) Model.

RFID tags can be classified into three categories based on the storage memory, cost, and battery requirements: passive tags, semi-passive tags, and active tags [5,6].

- A passive tag operates without battery, as the tag is energized when the reader interrogates it by sending a signal to request tag information. It has a short transmission range in communication, and has limited resources in term of storage. It is considered the lowest in cost and has a higher lifespan.
- A semi-passive tag has a battery for its internal chip circuit; however, it is also energized by the reader interrogation, as in the passive tag.
- An active tag runs with battery and can have two-way communication between tag and reader. It is larger due to the larger storage capacity and battery. The transmission range is also larger compared to passive tags. It is more expensive and has a limited life depending on the battery lifespan [2].

Table 1 provides some comparison of the three types of RFID tags.

Table 1. Classification of RFID Tags [7,8].

	Passive Tags	Semi-Passive Tags	Active Tags
Power	Surrounding signal	Internal chip battery	Integrated battery
Storage	Read memory	Read/write memory	Read/write memory
Distance	5 m	100 m	1000 m
Application	Identification	Real-time tracking	Environmental and logistic
Cost	Low	High	High
Size	Small	Large	Large
Lifespan	Unlimited	10 years	10 years
Tag Signal	Low	High	High
Required Signal	High	Low	Low

The basic communication session between an RFID reader and a tag starts when the reader broadcasts radio waves to interrogate the tag. The tag receives the signal and responds corresponding to the reader's request. Since the communication channel between the reader and tag is assumed to be insecure, it is important to maintain a secure system during communication to avoid information

leakage or forgery by unauthorized users. Efficient RFID concerns about system security, cost, and liability are essential factors for future adoption in the IoT.

3. Security Requirements and Threats

3.1. Security Requirements

The basic entities in the RFID system are the tag, reader, and database server. The communication channel between a tag and a reader is insecure and vulnerable to different security threats. Security requirements are the ability features that enable the system to avoid security threats. There are several security requirements to evaluate the security level of an RFID system:

- *Mutual Authentication*: the main requirement in a simple scenario of RFID communication session is the authentication between the reader and tag before exchanging or transmitting any secret or valuable information. Both tag and reader have to prove their legitimacy to each other to start a secure communication.
- *Confidentiality*: all of the transmitted messages have to be secure in which secret information and values that are used to execute communication cannot be obtained by an unauthorized user.
- *Integrity*: the transmitted data has to maintain its accuracy and not to be altered or changed during communication.
- *Availability*: the communication should be successfully executed by maintaining a synchronous state between the RFID entities. Communication values have to be updated after every successful session to provide system availability.
- *Privacy*: all of the secret information such as tag identity has to be secured in order to provide anonymity and avoid tracing the tag or its location.
- *Forward Security*: the transmitted data during communication have to be independent and updated for every session, and cannot be used or related to another authentication session. If a tag or any information is compromised, it is impossible for an adversary to pass the authentication on or violate the system.

3.2. Security Threats

A secure RFID system must be able to resist different types of attacks. Messages in RFID communication are transmitted in clear, and thus are vulnerable to eavesdrop; hence, secret information is disclosed. Many RFID protocols are proposed to defend against different attacks such as:

- *Replay Attack*: an adversary tries to capture the tag response and resend it to the reader to start a successful communication with the reader or obtain any secret information.
- *Man-In-The-Middle*: an adversary intercepts the message between two legitimate entities tag/reader to modify it and send it back.
- *Impersonate Attack*: an adversary obtains either the reader or tag identity information to create a forged entity. As a result, the adversary acts as a legitimate entity to pass the authentication and proceed with the communication.
- *Traceability*: an adversary traces the tag to find its location and revoke the tag's privacy. This attack violates the private information of RFID users, which is an instance where the privacy is important.
- *Desynchronization Attack*: communication session between tag and reader starts using the synchronous values stored in both the tag and reader to authenticate each other. A desynchronization attack occurs when an adversary breaks the synchronous state between the tag and server by blocking the update messages, causing the communication values stored in both server and tag to be different.
- *Denial of Service*: an adversary sends multiple signals simultaneously to the server as responses to make the system unavailable for further communication, which could further lead to a desynchronization attack.

- *Cloning*: an adversary uses a malicious device to obtain the reader or tag secret information and create a fake entity that can be used to perform a successful communication.
- *Disclosure*: an adversary identifies the secret information of the tag and the secret keys used in the communication to fully compromise the security of the protocol.

Many other security threats have been identified for RFID systems. A secure RFID system is created to defend against various threats that are related to the application in use.

4. Review of Recent RFID Authentication Protocols

Several articles are proposed to create a secure RFID protocol that improves the security measures of RFID systems. The modern advancement in technology helps discover many gaps in the proposed protocols presented in the literature. The aim of this work is to review some of the recent RFID authentication protocols that specifically use passive tags. We aim to present an adequate comparison between the protocols in terms of performance and security.

Since a passive tag is a very small chip with scarce resources, it is able to do only low computations. Hence, RFID protocols are classified in this paper into four categories based on the *complexity of the algorithm* that is used to compute the tag responses: heavyweight, simple weight, lightweight, and ultra-lightweight [9]. Heavyweight algorithms use symmetric and public key cryptography that is beyond the scale of the passive tag ability to process. Simple-weight algorithms use hash functions that are also not feasible for passive tag resources. Lightweight algorithms use simple one-way hash functions, cyclic redundancy checks, and pseudo-random number generators [10]. Finally, ultra-lightweight algorithms use bitwise operations, which can be performed at low cost.

4.1. Heavyweight Protocols

Wang and Sarma [11] proposed two session-based authentication protocols, *SB-A* and *SB-B*, for reader–tag authentication based on symmetric key encryption to ensure privacy and access control using two types of passive tags. The protocols are based on a symmetric cryptography algorithm to provide low-cost authentication such as the Advanced Encryption Standard (AES) and Data Encryption Standard (DES). Protocol *SB-A* in Figure 2 includes two processes. The first phase involves mutual authentication between server and tag according to the three-pass mutual authentication protocol according to the International Organization of Standardization and the International Electrotechnical Commission—ISO/IEC 9798-2 [12]. The second phase is for generating a session key between reader and tag according to the Otway–Rees protocol and updating the pseudo tag identity (PID). Protocol *SB-B* in Figure 3 uses tags with no memory or ID so that all of the tag’s information is stored in the server. A physical tag operation is mapped with the digital virtual tag in the server that can do all of the tag’s executions. The protocol time to keep synchronization is controlled by the tag nonce and counter, and not the server, because of the limited power of the tag to keep synchronization. The protocols proved to be secure against major types of attacks; however, the protocols are considered to be heavyweight, since DES and AES are expensive operations that require a lot of computational overhead.

Server S	Reader R	Tag T
<p>Step 4: Use PID to search the tag K_{TS}</p> <p>Step 5: Send $E_{K_{TS}}(N_T, N_S, PID_n)$ to R →</p> <p>Step 9:</p> <ul style="list-style-type: none"> - Verify OP_R - Generate K_{RT} - Update PID_n to PID_{n+1} <p>Step 10:</p> <p>Send to R</p> <ul style="list-style-type: none"> - $E_{K_{RS}}(N_R, PID_n, RID, OP_R, K_{RT})$ → - $E_{K_{TS}}(N_T, PID_{n+1}, RID, OP_R, K_{RT})$ → 	<p>Step 1: Send RID, OP_R to T →</p> <p>Step 3: Send PID_n, N_T to server ←</p> <p>Step 6: Send $E_{K_{TS}}(N_T, N_S, PID_n)$ to T →</p> <p>Step 8: Send to server ←</p> <ul style="list-style-type: none"> - $E_{K_{TS}}(N_S, N_T), PID_n$ - RID, OP_R, N_R <p>Step 11:</p> <ul style="list-style-type: none"> - Retrieve K_{RT} - Send $E_{K_{TS}}(N_T, PID_{n+1}, RID, OP_R, K_{RT})$ → - If OP_R is (write), encrypt info with K_{RT} and send it to T → 	<p>Step 2: Send PID_n and nonce N_T to R ←</p> <p>Step 7:</p> <ul style="list-style-type: none"> - Verify N_T to authenticate S - Send $E_{K_{TS}}(N_S, N_T), PID_n$ to R ← <p>Step 12:</p> <ul style="list-style-type: none"> - Retrieve $K_{RT}, PID_{n+1}, RID, OP_R$ - Verify $OP_R = OP_R$ in Step1 - Check the on-tag counter - Decode OP_R and execute it - Update PID_n to PID_{n+1} - If OP_R is (read), encrypt info with K_{RT} and send it to reader ←
<p>K_{RS}: server/reader shared key; K_{TS}: server/tag shared key; K_{RT}: reader/tag shared key; N_T: nonce generated by tag; N_R: nonce generated by reader; N_S: nonce generated by server; RID: reader ID; OP_R: operation of reader; PID_n: pseudo-ID of tag in current session; $E_K(M)$: message encrypted by key K.</p>		

Figure 2. Session-Based Authentication Protocol (SB-A) by Wang and Sarma.

Server S	Reader R	Tag T
<p>Step 4: Use PID to search the tag K_{TS}</p> <p>Step 5:</p> <ul style="list-style-type: none"> - Update PID_n to PID_{n+1} - Send $E_{K_{TS}}(N_T, N_S, PID_{n+1})$ to R → <p>Step 9:</p> <ul style="list-style-type: none"> - Verify reader authorization for OP_R <p>Step 10:</p> <ul style="list-style-type: none"> - If $OP_R = \text{read}$, send the message - If $OP_R = \text{kill}$: <ul style="list-style-type: none"> • Send $E_{K_{TS}}(N_T, PID_{n+1}, RID)$ to R → • Kill V_{tag} 	<p>Step 1: Send RID, OP_R to T →</p> <p>Step 3: Send PID_n, N_T to S ←</p> <p>Step 6: Send $E_{K_{TS}}(N_S, N_T, PID_{n+1})$ to T →</p> <p>Step 8:</p> <ul style="list-style-type: none"> - Send $E_{K_{TS}}(N_S, N_T, RID, OP_R), PID_n$ to S ← - Send RID, OP_R, N_R to S ← <p>Step 11: Send $E_{K_{TS}}(N_T, PID_{n+1}, RID)$ to T →</p>	<p>Step 2: Send PID_n and nonce N_T to R ←</p> <p>Step 7:</p> <ul style="list-style-type: none"> - Verify N_T to authenticate S - Send $E_{K_{TS}}(N_S, N_T, RID, OP_R), PID_n$ to R ← - If OP_R is not (kill), update PID_n to PID_{n+1} <p>- Retrieve N_T, PID_{n+1}, RID</p> <ul style="list-style-type: none"> - Verify $RID = RID$ in step1 - Check on-tag counter with time limit - Perform physical kill operation
<p>K_{RS}: server/reader shared key; K_{TS}: server/tag shared key; K_{RT}: reader/tag shared key; N_T: nonce generated by tag; N_R: nonce generated by reader; N_S: nonce generated by server; RID: reader ID; OP_R: operation of reader; PID_n: pseudo-ID of tag in current session; $E_K(M)$: message encrypted by key K; V_{tag}: virtual tag in the server.</p>		

Figure 3. Session-Based Authentication Protocol (SB-B) by Wang and Sarma.

For traceability issues in RFID, Ryu et al. [13] proposed elliptic curve cryptography-based untraceable authentication protocol (ECU) using the Schnorr signature scheme. The elliptic curve cryptography is considered to be a public key cryptography for RFID systems with low constrained tags. It is used to solve the issues of three recent elliptic curve-based untraceable RFID authentication protocols: Strong Privacy-preserving Authentication protocol (SPA) [14], Efficient Mutual Authentication protocol EMA [15], and ECC-based authentication protocol PII [16].

Ryu's protocol generates a digital signature with an appendix on the binary message of arbitrary length, and requires a cryptographic hash function, as shown in Figure 4. The sender's session key is combined with the receiver's public key to provide privacy, in which the message can be verified by only the receiver's private key. Ryu's protocol is secure against replay attacks, impersonate attacks, traceability attacks, and it maintains forward security. It requires two scalar multiplications, two hash functions, a message total size of 544 bits, and two communications between tag and reader. Even though this protocol requires complex computations associated with scalar multiplications and a hash function, it does not authenticate the reader.

Server S	Reader R	Tag T
Setup Phase: - Generate elliptic group G of prime order q . - Choose generator P of group G . - Server private/public keys $(y, Y = yP)$ - Store tag verifier $X = xP$ (public key) Authentication Phase:	Step 1: Send random c to $T \rightarrow$ Step 3: To authenticate tag - Compute $R' = y^{-1} Z$ - Derive $X' = \text{eid} \oplus H(R', s)$ - Check $X' = X$ registered verifier - Compute $v' = H(R', c)$ - Authenticate the tag as $H(sP - v' X, c) = v'$	Store x, X, Y (server public key) Step 2: - Pick r as session secret - $R = rP$ - $v = H(R, c)$ - schnorr sign $Z = rY, s = r + x * v$ - Encrypted verifier $\text{eid} = X \oplus H(R, s)$ - Send (eid, Z, s) to $R \leftarrow$
<small>G: Cyclic additive group; P: Generator of group G; q: Order of group G; x: Tag's private key; \oplus XOR; X: Tag's public key; y: Server's private; Y: Server's public; H: Hash function.</small>		

Figure 4. Elliptic Curve Cryptography-Based Untraceable Authentication Protocol (ECU) by Ryu.

To reduce the tag's overhead in heavyweight protocols, Yao et al. [17] introduced The Reviving-UNder-DoS (RUND) authentication protocol to defend against denial of service (DoS) and preserve user privacy by powering up the tag to do complex computing for symmetric and public key cryptography. It leverages the power in DoS scans to enable the tag to respond in two ways: either using simple encryption when the tag is activated by low signals from a reader, or using public encryption (higher security) when the backscattered signals are high in an insecure environment. The more signals there are in communication, the more power charges the tag. The option of using public key encryption in RUND protocol is to overcome the problem of breaking up the synchronization state between the reader and tag in symmetric key encryption. The protocol is secure because secret information is not sent in clear, so no useful information can be gained if any message is compromised. Moreover, the parameters used in communication are changed and updated in every session, as shown in Figure 5, to prevent replay attack, maintain forward security, and resist tracking. Even though the overall efficiency of RUND is $O(1)$, it is still not compliant with the Electronic Product Code Class1 Generation2 (EPC C1 G2) standard [18], which is defined by EPCGlobal Inc. for RFID data communication.

Server: S	Reader R: $PU_R, PR_R, \text{shared } K_i$	Tag T: $PU_R, \text{shared } K_i, ID$
<p>Initialization Phase:</p> <p>Mutual Authentication Phase:</p> <p>Updating Phase:</p>	<p>Step 1: Precompute and store in S: $f(K_i, c, \text{pad}_i) \leftarrow$ Where pad is padding length for $f()$</p> <p>Step 2: Send power waves last for T_{pw} with energy E_c. Send PRN r_1 in l length to tag \rightarrow</p> <p>Step 5: If response with symmetric: - Check counter c and search database for $f(K', c', \text{pad}_i) \leftarrow$ - Check r_1 for replayed msg. - If matches: tag is authenticated.</p> <p>If response with public key: - Check and search database for (ID, K) pair \leftarrow - Check r_1, r_2 for replayed msg. - If matches: tag is authenticated</p> <p>Step 6: Generate r_3 and compute $I_3 = r_3 f(K, r_3 I_i, \text{pad}_i)$ - Send I_3, r_3 to tag \rightarrow - Update $K = f(K, r_3, \text{pad}_i)$ - Update precomputed $f(K_i, c, \text{pad}_i)$ with updated key. - Preserve old key of tag.</p>	<p>- Counter c is set to 0.</p> <p>Step3: Compute: If E_c energy: - $I_1 = f(K, c, \text{pad}_i)$ - $I_2 = r_1 f(K, r_1 I_i, \text{pad}_i)$ - $I = I_1 I_2$ - Update $c = c + 1$ - Energy consumed E_{sk}</p> <p>If E_{pk} energy: - $E(PU_R, K, r_1 r_2, ID, c)$ in l length - Energy consumed E_{pk}</p> <p>Step4: Send I to reader \leftarrow</p> <p>Step 7: Check I_3 using r_3 by computing I'_3 - If matches: reader is authenticated. - Update $K = f(K, r_3, \text{pad}_i)$ - $C = 0$</p>
<p>PU_R: Public key of reader; ID: Tag's ID; K_i: Shared symmetric key; c: Counter for current key lifecycle; PR_R: Private key of reader; pad_i: Padding for $f()$; E_c: The initial power the tag is charged; T_{pw}: Time for the power waves to last; E_{sk}: Energy consumption for hash function; E_{pk}: Energy consumption for public key.</p>		

Figure 5. The Reviving-Under-Denial of Service Authentication Protocol (RUND) by Yao.

4.2. Simple-Weight Protocols

To better improve the performance of RFID protocols and reduce the power that is needed for complex operations in ECC-based protocols, Farash [19] proposed a mutual authentication protocol (IECC) based on the elliptic curve. The protocol enhances Chou's authentication protocol (EMA) [15], which does not fulfill the security requirement of forward security, mutual authentication, tag privacy, and security against location tracking, impersonating attacks, and tag cloning attack for an RFID system. The main idea behind the protocol is to use the server's public key to create the authentication message to avoid breaking the system privacy, as depicted in Figure 6. The IECC protocol is secure against major attacks, even though the computation cost is the same as in Chou's protocol that needs to be reduced for practical implementation.

Server S: $\{X_i, yP, P\}$	Reader R	Tag T: $\{X_i, Y, P\}$
<p>Setup phase:</p> <ul style="list-style-type: none"> - Generate an elliptic group G of prime order q - Choose generator P of group G - Choose random no. y as private key - Public key $Y = yP$ - Choose random X from G as tag identifier - Store X_i, Y, P in each tag. <p>Authentication phase:</p> <p>Step 1:</p> <ul style="list-style-type: none"> - Choose a prime random no. r - Compute $C0 = rP$ - Send C0 to tags → <p>Step 3:</p> <ul style="list-style-type: none"> - Obtain $K' = y^{-1}C1$ - Obtain $X_i' = C2 - h(C0, C1, K')$ - Find a match for X_i' in DB - If found: $C3 = h(X_i', K')$ and tag authenticated - Send C3 to tag → 		<p>Step 2:</p> <ul style="list-style-type: none"> - Choose a prime random no. k - $K = kP$ - $C1 = kY$ - $C2 = X_i + h(C0, C1, K)$ - Send C1, C2 to server → <p>Step 4:</p> <ul style="list-style-type: none"> - Validate $C3 = (X_i, K)$ - Server is authenticated
<p>G: A additive group of prime order q; P: Generator of group G; h: One-way hash function; y: Server's private; Y: Server's public; X_i: Identifier of ith tag which is a random point in G.</p>		

Figure 6. Mutual Authentication Protocol Based on Elliptic Curve Cryptography (IECC) by Farash.

Zhang and Qi [20] also proposed another protocol (EECC) to withstand the security weaknesses of Chou's protocol, EMA [15]. EECC protocol enhances patient medication safety by also using elliptic curve cryptography. In comparison to EMA protocol, EECC protocol resulted in better performance and security resistance to impersonate and forward security attacks.

B.Chen [21] proposed a role-based access control (RBAC) protocol for mobile RFID to enable user privacy, role, and access control through the back-end server based on a certification mechanism. RBAC assigns role classes as keys to control the information and the number of times each reader can read a tag. RBAC authorizes readers, assigns role classes to control the reader's authority to request tag information, and updates time stamps using random numbers and different shared keys between the database server and reader and tag ad, as depicted in Figure 7. Traceability and replay attacks are prevented using updated random numbers in every session; access control is provided using shared keys to prevent unauthorized readers to request or read any tag's information, and integrity is ensured using timestamps. However, RBAC uses one encryption mechanism that is excessive for low-cost passive tags.

Server: k_x, k_y keys	Reader: k_y keys	Tag: k_x keys
<p>1- Reader Authorization and role class:</p> <ul style="list-style-type: none"> - Request role-class command, read tag command, TID, and RID from RBAC - RBAC sends role-class. - $M_3 = E_{k_y}(RID, r1, TS_1, Cert_r, \text{role-class})$ - $M_4 = E_{k_x}(TID, r2, TS_1, \text{role-class})$ - Send M_3, M_4 to reader \rightarrow <p>2- Assign No. of reads and update timestamps:</p> <p>Step 7: Retrieve $Cert_r, r2$ from M_7</p> <ul style="list-style-type: none"> - If $Cert_r$ is verified, retrieve TS_2, TC_{n-1} from M_6. - $M_8 = E_{k_y}(TS_2, TC_{n-1}, r2)$ - Send M_8 to reader \rightarrow 	<p>Step 1: Reader sends Hello to tag \rightarrow</p> <ul style="list-style-type: none"> - Create random no. $r2$. - $M_2 = E_{k_y}(M_1, r2, RID, \text{Command})$ <p>Step 3: Send M_2 to server \leftarrow</p> <p>Step 4:</p> <ul style="list-style-type: none"> - Retrieve $r1, TS_1, Cert_r, \text{role-class}$ from M_3. - $M_5 = H(TS_1 \oplus r2)$ - Send M_4, M_5 to tag \rightarrow <p>Step 6:</p> <ul style="list-style-type: none"> - Receive M_6 - $M_7 = E_{k_y}(Cert_r, r2, M_6)$ - Send M_7 to database server \leftarrow <p>Step 8:</p> <ul style="list-style-type: none"> - Retrieve $TS_2, TC_{n-1}, r2$ from M_8 - Verify $r2$ 	<ul style="list-style-type: none"> - Create random no. $r1$ - $M_1 = E_{k_x}(TID, TS, r1)$ <p>Step 2: Sends M_1 to reader \leftarrow</p> <p>Step 5: Verify M_5 using TS_1 from M_4 and its $r1$ to authenticate reader</p> <ul style="list-style-type: none"> - Calculate number of reads $TC_{n-1} = TC_n - 1$ - if TS_1 is verified, it's updated to TS_2 - $M_6 = E_{k_x}(TS_2, TC_{n-1})$ - Send M_6 to reader \leftarrow
<p>TID: Tag ID; K_y: Server/Reader shared key; r: random number; TC_n: number of times a reader request information; K_x: Server/Tag shared key; TS: Timestamp; $Cert_r$: Reader security certificate; RBAC: role-based access control.</p>		

Figure 7. Role-Based Access Control Protocol (RBAC) by B. Chen.

4.3. Lightweight Protocols

Successful businesses demand an efficient RFID system that is mainly based on low computation for a low cost. Many recent RFID protocols use low-cost operations that are handled by low-cost passive tags for practical implementations.

Fernando and Abawajy [22] proposed a mutual authentication protocol for Networked RFID Systems NRS, which is a lightweight mutual authentication scheme for an RFID system using low operations such as exclusive or operation (XOR) and one-way hash functions. However, Alagheband and Aref [10] reported NRS to be vulnerable to major attacks and specifically a full disclosure attack that compromises the whole RFID system. Alagheband and Aref improved NRS protocol and proposed NRS+ by adding three more hash functions to the authentication message to increase the system security. X. Chen et al. [23] noted that the NRS+ protocol is exposed to desynchronization and traceability attacks by using one random number for the tag and reader. Thus, X. Chen proposed NRS++ to improve the security flaws in the previous versions of NRS by generating two different random numbers, $r1$ and $r2$, for the tag and reader using a pseudo-random number generator (PRNG) to defend against replay attack. In Figure 8, the authentication message $M3$ is encrypted using the tag's random number $r1$ and reader's random number $r2$ to provide message integrity, so any modified message cannot be verified by the tag. NRS++ uses fewer hash functions, which resulted in less computation overhead and storage space than the other versions, with more security power.

Server S	Reader R	Tag T
<p>- Update secrets in Database $ID_{new} = ID \oplus (r_{2right} K_{1left})$ $K_{1new} = H[(K_{1right} r_{1left}) \oplus r_2]$</p>	<p>Step 1: - Generate random no. r - Calculate $M_1 = H(EPC \oplus K_1 r)$ $M_2 = r \oplus K_1$ - Send to tag $M_1 M_2 \rightarrow$</p> <p>Step 3: - Extract $r_1 = N \oplus K_1$ - Compute $C_2 = H(EPC \oplus K_1 r_1 r_2)$ - Verify $C_2 = M_3$ If equal: Generate random no. r_2 $M_4 = r_2 \oplus K_1$ $M_5 = H(EPC \oplus K_1 r_1 r_2)$ If not equal: terminate - Send $M_4 M_5 \rightarrow$</p>	<p>Step 2: - Extract r as $r = M_2 \oplus K_1$ - Compute $C_1 = H(EPC \oplus K_1 r)$ If $C_1 = M_1$, generate r_1 $N = r_1 \oplus K_1$ $M_3 = H(EPC \oplus K_1 r r_1)$ Else termination - \leftarrow Send $M_3 N$ to reader</p> <p>Step 4: - Extract r_2 as $r_2 = M_4 \oplus K_1$ - Compute $C_3 = H(EPC \oplus K_1 r_1 r_2)$ - Verify $C_3 = M_5$ If equal: Update the secrets. If not equal: terminate</p>
<p>ID, EPC: Tag identifier; H(): one-way hash function; K_1: Server/Tag shared key; r, r_1, r_2: random No; \oplus: XOR and concatenation operation.</p>		

Figure 8. Mutual Authentication Protocol for Networked RFID Systems (NRS++) by X. Chen.

C. Chen [24] proposed Anti-Counting Security Protocol (ACSP) as another lightweight protocol for RFID systems to defend from a counter attack, which is defined as the attacker's ability to count the number of objects in a system. Safkhani et al. [25] reported ACSP to be vulnerable to major attacks, including the forward/backward traceability attack. Safkhani further proposed ACSP+ to improve Chen's protocol. Later, X. Chen [23] pointed out that ACSP protocol is not secure, and proposed ACSP++ to withstand DoS and forward/backward traceability attacks. ACSP++ enhances the session identifier (SID) update, which is used to verify the current session, and tag identification phases that suffer from different attacks in ACSP and ACSP+ versions. In ACSP++ as depicted in Figure 9, a tag identifier (TID) is added to the identification message as $(\overline{IDENT}, R4, R5, TID)$ instead of $(\overline{IDENT}, R4, R5)$, and the authentication message $(\overline{AUTHEN}, R4, R5, TID)$ is replaced with $(\overline{AUTHEN}, R5, TID)$ to overcome DoS attack and modifying the TID in the identification phase. The update phase of every key is associated with two separate nonce values to avoid forward and backward traceability. Even though the protocol improved the security weaknesses of all of the ACSP versions, it did not lower the computation overhead nor the storage space.

Reader R	Tag T
<p>(SID Update Phase) Step 1:</p> <ul style="list-style-type: none"> - Generate nonce R1 - Send the following to tag: $\overline{UPDSID}, R1 \oplus SID, H(\overline{UPDSID}, R1, SID) \rightarrow$ <p>Step 3:</p> <ul style="list-style-type: none"> - Extract R2 and verify $H(\overline{UPDACK}, R2, R1, SID)$ - Update SID as $SID_{new} = H(SID R2 R1)$ 	<p>Step 2:</p> <ul style="list-style-type: none"> - Extract R1 to verify $H(\overline{UPDSID}, R1, SID)$ - Generate R2 - Update SID $SID_{new} = H(SID R2 R1)$ $SID_{old} = SID_{cur}$ - \leftarrow Send to reader confirmation: $\overline{UPDACK}, R2 \oplus SID, H(\overline{UPDACK}, R2, R1, SID)$
<p>(Tag Identification Phase) Step1:</p> <ul style="list-style-type: none"> - Generate R3, R4 - Send the following messages to tag \rightarrow a) $\overline{SELECT}, SID1 \oplus R3, H(\overline{SELECT}, R3, SID)$ b) $\overline{QUERY}, SID \oplus TID \oplus R4, H(\overline{QUERY}, R4, SID, TID)$ <p>Step 4:</p> <ul style="list-style-type: none"> - Authenticate tag - Extract R5' to verify $H(\overline{IDENT}, R4, R5, TID)$ - If not verified: stop the session and send $\overline{QUERY REP} \rightarrow$ - If verified: update TID as $TID_{new} = H(TID R4 R5)$ $TID_{old} = TID$ - Send $(\overline{AUTHEN}, H(\overline{AUTHEN}, R5, TID)) \rightarrow$ 	<p>Step 2:</p> <ul style="list-style-type: none"> - Extract R3' to verify $H(\overline{SELECT}, R3, SID)$ If not verified: wait until next run. If verified: respond with step3. <p>Step 3:</p> <ul style="list-style-type: none"> - Extract R4' to verify $H(\overline{QUERY}, R4, SID, TID)$ - Generate R5 - \leftarrow Send $(\overline{IDENT}, TID \oplus R5, H(\overline{IDENT}, R4, R5, TID))$ <p>Step 5:</p> <ul style="list-style-type: none"> - Calculate and verify $H(\overline{AUTHEN}, R5, TID)$ - If not verified: stop the session. - If verified: update the tag identifier as $TID_{new} = H(TID R4 R5)$
<p>R1, R2, R3, R4, R5: nonce; $\overline{SELECT}/\overline{QUERY}$: Select/ query commands; SID_{cur}/SID_{new}: Current/ New session identifier; $\overline{UPDSID}/\overline{UPDACK}$: SID update/ Update knowledge message; TID_{cur}/TID_{new}: Current/ New unique identifier; $\overline{IDENT}/\overline{AUTHEN}$: Identification/ authentication messages.</p>	

Figure 9. Anti-Counting Security Protocol (ACSP++) by X. Chen.

Chien and Huang [26] presented LAP, which is a lightweight authentication protocol to solve the vulnerabilities in the authentication protocol of Li et al. [27], and enhance the computational cost from $O(n)$ to $O(1)$ in identifying tags in RFID systems. The security of LAP protocol is based on a synchronized PRNG between reader and tag using a secret key, secret ID, and index pseudonym. In Figure 10, LAP protocol uses the rotate operator on the message and left/right operator for the divided rotation during the messages that were exchanged to form a secure permutation. Random numbers are used to shift the secret values of the tag to be used safely in communication. Then, the random number is XORed with the shifted secret value to securely retrieve a tag by the server. The server uses the index pseudonym (IDS) to quickly identify the tag in the database instead of computing $PIDL \oplus PIDR$ for every tag to make the computation $O(1)$. LAP protocol is resistant to replay attack, DoS, and forward security. It can be employed easily by different standards such as EPC Gen2 and ISO 15693 [28] for practical implementation. However, the protocol was noted as being partially secure against traceability and synchronization attacks, since a tag can be traced between two successful sessions if the tag could not update its IDS.

Server S: flag, X_{old} , X_{new} , IDS_{old} , IDS_{new} , SID	Reader R	Tag T: {SID, IDS, X}
<p>Step 2:</p> <ul style="list-style-type: none"> - Search IDS_i - If $IDS = IDS_{old}$: flag = 0, $X = X_{old}$ - If $IDS = IDS_{new}$: flag = 1, $X = X_{new}$ - $g' = g(R_1 R_2 X)$ - $SID' = rotate(SID, g')$ - Verify R' as $R' = left(SID' \oplus g')$ - Compute $R'' = right(SID' \oplus g')$ - If flag = 1 <ul style="list-style-type: none"> • $IDS_{old} = IDS_{new}$ • $X_{old} = X_{new}$ - Else <ul style="list-style-type: none"> • $IDS_{new} = g(IDS SID')$ • $X_{new} = g(X g')$ - Send R'' to reader \rightarrow <p>Step 4:</p> <ul style="list-style-type: none"> - When OK is received, send SID to R \rightarrow 	<p>Step 1:</p> <ul style="list-style-type: none"> - Generate R_1. - Send Query R_1 to T \rightarrow - Forward $R_1 R_2 R' IDS$ to S \leftarrow - Forward R'' to T \rightarrow - Forward ACK to S \leftarrow 	<ul style="list-style-type: none"> - Generate R_2 - Compute $g' = g(R_1 R_2 X)$ - $SID' = rotate(SID, g')$ - $R' = left(SID' \oplus g')$ - Send $R_2 R' IDS$ to R \leftarrow Step 3: - Verify $R'' = right(SID' \oplus g')$ - Update: <ul style="list-style-type: none"> • $IDS = g(IDS SID')$ • $X = X_{new} = g(X g')$ - Send ACK to R \leftarrow
<p>SID: Secure ID; PID: Partial ID; IDS: Index pseudonym; g(): Random No. generator; X: 1-bit secret key; R_1, R_2: Random numbers; Rotate(): Rotation function; Left(s): Left half of s; Right(s): Right half of s; ACK: Acknowledgement.</p>		

Figure 10. Lightweight Authentication Protocol (LAP) by Chien.

Burmester and Munilla [29] proposed a lightweight mutual authentication protocol called Flyweight that is based on exchanging messages using only PRNG. Their protocol is based on a shared PRNG algorithm between the tags and back-end server that takes the same seed to produce the same output. The concept of the protocol is to use three consecutive numbers—RN1, RN2, and RN3—generated by the same PRNG in the server, and the tags of five numbers if an active adversary is presented, such as in Figure 11. Furthermore, RFID tags precompute the values to the server challenging the response, so an adversary can be detected based on the response time from the tag. The protocol is able to provide mutual authentication, integrity, confidentiality, and forward and backward security. In addition, it provides strong synchronization, since the server keeps a record for the current and next response value of the tag.

S. Lee et al. [30] proposed a lightweight protocol (MASS) for RFID systems using XOR and a one-way hash function to conform to the scarce resources of RFID tags. The concept of the MASS protocol is to challenge the tag with a fresh random string every session, and the tag responds using the reader's value and its own random key to authenticate the reader ad, as depicted in Figure 12. The secret key is shared between entities, and all of the messages are encrypted during transmission. However, Zuo [31] conducted a survivability experiment on the authentication protocol proposed by S. Lee et al. and defined the vulnerability of the protocol to replay, desynchronize, and impersonate attacks. Zuo concluded from his experiment that the system could employ two different values for the keys (old, new) to recognize the tag and overcome the desynchronization problem.

Server S	Reader R	Tag T
<ul style="list-style-type: none"> - Check if $RN1 = RN1^{cur}$ <ul style="list-style-type: none"> • $cnt = 1$ • Generate $RN2$, send $RN2$ to $R \rightarrow$ - If $RN1 = RN1^{next}$ <ul style="list-style-type: none"> • $cnt = 0$ • Update values in DB • Send updated $RN2$ to $R \rightarrow$ <p>Step 4:</p> <ul style="list-style-type: none"> - If $RN = RN3$, and $cnt = 0$ <ul style="list-style-type: none"> • Tag is authenticated - If $RN = RN4$ <ul style="list-style-type: none"> • Send $RN3$, store $RN5$ • Update values • Send $RN3$ to R <p>Step 6:</p> <ul style="list-style-type: none"> - If $RN5$ is correct <ul style="list-style-type: none"> • Authenticate T • Update values - Else terminate 	<p>Step 1:</p> <ul style="list-style-type: none"> - Send Query to $T \rightarrow$ <p>Step 2:</p> <ul style="list-style-type: none"> - Forward $RN1$ to $S \leftarrow$ <ul style="list-style-type: none"> - Forward $RN2$ to $T \rightarrow$ <ul style="list-style-type: none"> - Forward $RN4$ to $S \leftarrow$ <ul style="list-style-type: none"> - Forward $RN3$ to $T \rightarrow$ <ul style="list-style-type: none"> - Forward $RN5$ to $S \leftarrow$ 	<ul style="list-style-type: none"> - $RN1 = g_{tag}(\text{state})$ - Set alarm $cnt = 1$ - Send $RN1$ to $R \leftarrow$ <p>Step 3:</p> <ul style="list-style-type: none"> - If $RN2$ is correct to authenticate S <ul style="list-style-type: none"> • Generate $RN3, RN4, RN5$ • $Cnt = 0$ - If $cnt = 0$, send $RN3$ to $R \leftarrow$ - If $cnt = 1$, send $RN4$ to $R \leftarrow$ <p>Step 5:</p> <ul style="list-style-type: none"> - If $RN3$ is correct and $cnt = 1$ <ul style="list-style-type: none"> • Send $RN5$ to $R \leftarrow$ - Else terminate
RN: Random numbers output of the same generator function		cnt: l-bit flag

Figure 11. Flyweight Mutual Authentication Protocol by Burmester and Munilla.

Server S	Reader R	Tag T
	<p>Step 1:</p> <ul style="list-style-type: none"> - Generate l-bit string str - Send str to tag \rightarrow <p>Step 3:</p> <ul style="list-style-type: none"> - Search database to match key K_i - If found proceed to update key - Retrieve rB from rC - $K_i = h(K_i)$ - $r'C = h(rB \oplus K_i \oplus str)$ - Send $r'C$ to tag \rightarrow 	<p>Step 2:</p> <ul style="list-style-type: none"> - Generate l-bit string rA - $rB = h(rA \oplus K_i \oplus str)$ - $rC = h(rB \oplus K_i \oplus str)$ - Send rB, rC to reader \leftarrow <p>Step 4:</p> <ul style="list-style-type: none"> - Verify $r'C = rC$ - If verified, update key
Ki: Tag/server shared secret key; h(): One-way hash function		

Figure 12. Lightweight Protocol based on Synchronized Secret (MASS) by S. Lee.

To reduce the communication time during the authentication session, K. Lee et al. [32] proposed Efficient Passively-Untraceable Authentication Protocol (EP-UAP). The concept of EP-UAP is that the system precomputes all of the necessary computations before the system initialization, so only low computation overhead is required on the tag side during the process phase. The protocol is based on Randomized Hash-Lock protocol, which uses a static identifier, and its strong security against traceability depends mainly on PRNG to randomize the responses, as explained in Figure 13. Since precomputing all of the possible random numbers and responses requires a storage memory for all of the precomputed data in the database, EP-UAP is preferred for small to medium networks,

as the storage memory increases when the number of tags increases. The protocol shows a huge improvement over the randomized hash lock protocol in terms of computation time, in that only requires 40 ms for authentication; this is similar to LRMAP, which is the most efficient one in stateful protocols. However, it requires 100 MB of database storage memory. The protocol provides integrity due to the two randomly generated nonce values that are used from both tag and reader, and is secure against passive attacks and traceability due to the random responses. However, the EP-UAP protocol seems to be vulnerable to active attacks such as impersonate and replay attacks, since the random responses depend on the database/reader. It also requires high storage capacity in the database side.

Reader	Tag
<p>Step 1:</p> <ul style="list-style-type: none"> - Generate R_R - Send Query, R_R to tag \rightarrow <p>Step 3:</p> <ul style="list-style-type: none"> - Search for ID_{iR} - Verify $H(ID_{iR} R_R) = m_{TR}$ to authenticate the tag. - Compute $m_{RT} = H(ID_{iR} R_T)$ - Send m_{RT} to tag \rightarrow 	<p>Step 2:</p> <ul style="list-style-type: none"> - Generate R_T - Compute $m_{TR} = H(ID_{iT} R_R)$ - Send m_{TR}, R_T to reader \leftarrow <p>Pre-compute $c_T = H(ID_{iT} R_T)$</p> <p>Step 4:</p> <ul style="list-style-type: none"> - If $m_{RT} = c_T$, reader is authenticated.
<p>H: One-way hash function; ID: Tag identifier; R_R, R_T: nonce generated by reader/tag; m: Authentication challenge; c: Authentication challenge response.</p>	

Figure 13. Efficient Passively-Untraceable Authentication Protocol (EP-UAP) by K. Lee.

To defend against a desynchronization attack, Rahman and Ahamad [33] proposed a Desynchronization attack-resistant Robust Authentication Protocol (DRAP) in the wireless identification and sensing platforms (WISP), where RFID technology is combined with sensor nodes. Their protocol mechanism is to decrease the tag collision that leads to DoS attack, as shown in Figure 14. The technique is to decrease the collision rate at the link layer and maintain the system's efficiency. The protocol also detects the DoS attack and recovers the synchronization state of the system. It has higher resources than passive tags, which allow higher security implementation. Yet, it has a short distance limitation, where tags can only function less than 1–2 m away from readers.

Authentication in most RFID protocols is executed between one reader and one tag at a time. Liu et al. [34] proposed a grouping proofs-based authentication protocol (GUPA) to enable authenticating multiple tags and multiple readers simultaneously, such that multiple readers can authenticate a single tag, and multiple tags can be authenticated by a single reader in large-scale RFID. GUPA protocol is based on hierarchical identification between independent subgroups in a distributed RFID system, and the use of an asymmetric denial mechanism to resist denial-of-proof attack (DoP). For the anonymous authentication of a new entity, GUPA deploys a ring signature using a lightweight cryptography (elliptic curve). It also uses lightweight bitwise operations for readers and tags secret information updates, PRNGs, one-way hash functions, timestamps for session freshness, and access lists for each legal reader/tag during system initialization as identity flags to prevent forgery and tracking attack, as fully explained in Figure 15. Since the flags are chosen randomly from the pseudonym index, queries and responses are independent for each session to resist DoP attack; hence, illegal proofs are eliminated during authentication.

Server S	Reader R: ID: $K_{i_{prev}}$, K_i , $D_{i_{prev}}$	Tag T: K_i , ID $_i$, Δ
	<p>Step 1:</p> <ul style="list-style-type: none"> - Generate random n_r. - Send n_r to tag \rightarrow <p>Step 3:</p> <ul style="list-style-type: none"> - Generate $P(K_i \oplus n_r n_i)$ for all tags to verify α_i. <u>If there is a match:</u> - Decrypt α_i and β_i - Retrieve D - If D_{new_i} is not equal to D_{new} then update: <ul style="list-style-type: none"> $K_{i_{prev}} = K_i$ $X = h(K_i)$ $\alpha_j = P(X \oplus n_r n_i)$ $K_i = h(x)$ $D_{i_{prev}} = D_{i_{new}}$ - Else ignore the message and $\alpha_j = rand$ <u>If there is no match:</u> - Generate $P(K_{i_{prev}} \oplus n_r n_i)$ for all tags to verify α_i - If correct: - Decrypt α_i and β_i <ul style="list-style-type: none"> - If D_{new_i} is not equal to D_{old_i} then update: <ul style="list-style-type: none"> $\alpha_j = P(h(K_{i_{prev}}) \oplus n_r n_i)$ $D_{i_{prev}} = D_{i_{new}}$ - Else ignore the message and $\alpha_j = rand$ Else ignore the message and $\alpha_j = rand$ - Send α_j to tag \rightarrow 	<p>Step 2:</p> <p>If ($\Delta \leq D_{new} - D_{old}$)</p> <ul style="list-style-type: none"> - Generate random n_i - $\alpha_i = P(K_i \oplus n_r n_i)$ - $\beta_i = E_{K_{wti}}(h(ID_i) \oplus D_{new})$ - Send α_i, β_i, n_i to reader \leftarrow <p>Step 4:</p> <ul style="list-style-type: none"> - $Y = h(K_i)$ - Generate $P(Y \oplus n_r n_i)$ to verify α_j - if correct: $K_i = h(Y)$
<p>P(): Pseudorandom No. generator; Δ: Activity threshold; D: Sensor value; K_i: Secret number; ID: Tag identifier; h(): One-way hash function.</p>		

Figure 14. Desynchronization Attack-Resistant Robust Authentication Protocol (DRAP) by Rahman and Ahamad.

Database: DB	Reader: R_j	Tag: T_a
<p>Initialization Phase:</p> <ol style="list-style-type: none"> 1- Generate PRN r_{DB} 2- Send r_{DB} to tag \rightarrow <ol style="list-style-type: none"> 6- Verify H_1 in database for match 7- $H_1 = (\Delta R_j L_R r_{T_y})$ 8- PRNG (ΔR_j) 9- Send $H_1 PRNG(\Delta R_j)$ to tag \rightarrow <p>Authentication Phase:</p>		<ol style="list-style-type: none"> 3- Generate r_{T_y} 4- $H_1(L_R r_{DB})$ 5- Send $r_{T_y} H_1(L_R r_{DB})$ to DB \leftarrow <ol style="list-style-type: none"> 10- $PRNG^{-1}(\Delta R_j)$ to obtain ΔR_j 11- $H_1 = (\Delta R_j L_R r_{T_y})$ to authenticate DB 12- Add ΔR_j to L_R
<p>L_R: Local access list; ΔR_j: Reader’s information; H(): One-way hash function.</p>		

Figure 15. Grouping Proofs-Based Authentication Protocol (GUPA) by Liu for a Single-Reader—Single-Tag Case.

Since tag collision is a major problem in the large-scale networks, Rahman and Ahamad [35] proposed two probabilistic batch authentication protocols to determine the valid tags efficiently and accurately in large-scale systems. FTest is a protocol based on Frame Slotted Aloha algorithm that is used to reduce the probability of collision slots. The other protocol is GTest, which is a protocol

based on group batch authentication that is used to reduce the cost of detecting counterfeit tags. Their protocols use simple lightweight operations such as XOR and cyclic redundancy checks (CRC) with a shared key for each group of tags. The theory in both protocols is not to send the tag ID when responding, but rather accept or reject a tag by estimating the number of fake tags. In the FTest protocol that is depicted in Figure 16, a counterfeit threshold parameter is used in the system to reduce the number of rounds in the detection process and response time of the protocol, so that the entire tag responses do not need to be checked. Instead, the detection will stop if the percentage of counterfeit tags exceeds the counterfeit threshold. In GTest, the reader randomly selects a population of tags to authenticate. If one counterfeit tag is detected, the batch of tags will be considered invalid. The reader needs to read a large amount of data to identify the validity of a batch in GTest, so the reader still consumes time through the computation overhead from the tag search. Both FTest and GTest protocols are proved to be secure against tracking and privacy attacks, since tags responses are based on dynamic frame size, random numbers, and ID that is not transmitted during communication. However, the FTest shows less execution time and better performance over GTest.

Server S	Reader R	Tag T: Shared group key k_i
Group Identification Phase:	1- Send nonce n_r to tag \rightarrow 3- Find a group key to decrypt the message. 4- Identify the group of tags based on the group key.	2- Respond by $h(k_i n_r)$ \leftarrow
Authentication Phase:	1- Send to server "Start authentication" \leftarrow 2- Receive (f, r) from server 3- Broadcast frame size and random no. 6- Generate RV based on responses 0, 1, coll. 7- Turn collision slot into singleton by removing one tag (removed tags remain silent until next phase) 8- Send RV to server for verification.	4- Each tag compute its slot position $SP = h(id, r) \bmod f = 0$ or 1 5- Send SP to reader with random bits \leftarrow
Counterfeit Detection Phase:	1- Send random n_r from server to rem tags \rightarrow 3- Forward RV to server \leftarrow	2- Respond $h(id n_r)$
4- Reconstruct RVs as only valid tags can compute correct $h()$ 5- Accept valid tags if $RV_s = RV$		
n : Nonce value; k_i : Shared group key; $h()$: One-way hash function; SP: Slot position within frame; id: Tag ID; f : Frame size; r : Random N; RV: Response vector generated by reader; RV_s : Response vector generated by server; rem: Set of tags removed to reduce collision slot.		

Figure 16. Batch Authentication Protocol based on Frame Slotted Aloha (FTest) by Rahman and Ahamad.

Another anti-collision security protocol (ACS) is proposed by Keqiang et al. [36] for a high-efficiency RFID system combining the chaotic sequence generator with the dynamic frame-slotted ALOHA algorithm for fast tag identification. The protocol scheme is based on a logistic mapping structure with XOR operation and spreading operation to generate real-time keys in a chaotic sequence that are used in authentication messages. Keys are updated in each response from tag to reader and reader to tag during the same session using iteration equations that are known only to the server and tag, such as in Figure 17. The protocol is effective against counterfeits and impersonates attacks, as the authentication scheme not only depends on the iterated key, but also on spreading code and random numbers, so faking at least one of them will result in a wrong response. The protocol requires only four message exchanges, low hardware cost, and low computation cost on the tag side. It also has lower energy consumption than other heavy and simple weight protocols, because XOR uses less energy than symmetric encryption and hash functions.

Server S: K_0	Reader R	Tag T: K'_0
<ul style="list-style-type: none"> - K_0 = Master key, $x_0 = K_0$ to compute x_i - Verify ChaosSpec using x_i: <ul style="list-style-type: none"> • If there is collision, go to step5. • If no collision, proceed. - Perform one-time iteration to get $x_{i+1} = K_{i+1}$ - Extract R'_0 from $H'(R_0)$ and verify $R'_0 = R_0$ - Tag is authenticated. - Extract ID from $H'(ID)$ - Perform R_1 iteration to get x_j, $j = (r+R_1+R_0)$ - $K_j = x_j$ - $H(R_1) = R_1 \oplus K_j$ - Send to reader $(H(R_1)) \otimes$ ChaosSpec \rightarrow 	<p>Step 1:</p> <ul style="list-style-type: none"> - Generate and send a frame size R_0 to tag \rightarrow <p>Step 3:</p> <ul style="list-style-type: none"> - Send $(H'(R_0) H'(ID) R_1) \otimes$ ChaosSpec, and R_0 to S \leftarrow <p>- Send $(H(R_1)) \otimes$ ChaosSpec \rightarrow</p> <p>Step 5: Collision case</p> <ul style="list-style-type: none"> - Increase tag's slot counter by 1 - Restart identification process in Step2 <p>Step 6: No authentication occurs</p> <ul style="list-style-type: none"> - Issue AdjustQuery command - Adjust R_0 to decide a new frame size - Send search signal to rest of tags \rightarrow 	<p>Step 2:</p> <ul style="list-style-type: none"> - Receive R_0. - Choose slot index with the value in $[1, R_0]$ - Reset time slot counter = slot-index - $r = 20, i = (r+R_0), x'_0 = K'_0$ - $x'_{k+1} = rx'_k (1-x'_k)$ iteration = x'_i - $x'_i =$ ChaosSpec - Perform one-time iteration to get $K'_{i+1} = x'_{i+1}$ - $H'(R_0) = R_0 \oplus K'_{i+1}$ - $H'(ID) = K'_{i+1} \oplus ID$ - Generate random R_1 - Send $(H'(R_0) H'(ID) R_1) \otimes$ ChaosSpec \leftarrow <p>Step 4:</p> <ul style="list-style-type: none"> - Perform the equations to get $K'_j = x'_j$ - Calculate $R'_1 = H(R_1) \oplus K'_j$ - If $R'_1 = R_1$, then $K'_j = K_j$ - R is authenticated
<p>R_0: Frame size; i: Number of iterations; K'_0: Tag key; K_0: Server master key; K'_{i+1}: Real-time key; $H(), H'()$: One-way hash functions; ChaosSpec: Spreading code; ID: Tag's ID; R_1: Random number generated by tag; r: Constant value to put the equation in chaotic state.</p>		

Figure 17. Anti-Collision Security Protocol (ACS) by Keqiang.

Cho et al. [37] proposed a hash-based mutual authentication protocol (HBA) to defend against the brute force attack. This protocol was reported by Chang et al. [6] to be vulnerable to denial of service (DoS) and replay attacks. Later, Chang et al. proposed an improved (HBA+) protocol to avoid DoS and replay attacks using a shared PRNG algorithm between the server and tag to produce the same output that is used in updating the protocol values, as in Figure 18. Also, the confidentiality in the protocol is based on protecting the secret value *datai* using reader ID (*Rid*), which is only known to a legitimate reader and server. The improved protocol of Chang is considered to be efficient and secure against DoS attack, traceability, and forward secrecy.

Server S	Reader R	Tag T
<p>Step 4: Search the database using I: - Found: $I = I_{new} \{EPC_i, Auk_{new}, Ack_{new}, data_i\}$ $I = I_{old} \{EPC_i, Auk_{old}, Ack_{old}, data_i\}$ $M1' = Auk_{new} \oplus I_{new} \oplus PRNG(EPC_k \oplus Ack_{new} \oplus R_t \oplus R_r)$ to authenticate T. - Not Found: termination. - $B = data_i \oplus Rid_k$ - $M2 = PRNG(Auk_{new} \oplus R_t) \oplus Ack_{new}$ - $C = H(data_i \oplus R_r)$ - Update database values and keys: $Auk_{old} = Auk_{new}$ $Auk_{new} = PRNG(Auk_{new})$ $Ack_{old} = Ack_{new}$ $Ack_{new} = PRNG(Ack_{new})$ $I_{old} = I_{new}$ $I_{new} = PRNG(Ack_{new} \oplus I_{new})$ - send $\{B, C, M2\} \rightarrow$</p>	<p>Step 1: - Generate random No. $R_r \rightarrow$</p> <p>Step 3: - $A = H(Rid \oplus R_r)$ - $\leftarrow \{M1, R_t, I, A, R_r\}$</p> <p>Step 5: - Obtain $data_i$ from B - $C' = H(data_i \oplus R_r)$ - send $M2 \rightarrow$</p>	<p>Step 2: - Generate random No. R_t - $M1 = Auk \oplus I \oplus PRNG(EPC \oplus Ack \oplus R_r \oplus R_t)$ - $\leftarrow \{M1, R_t, I\}$</p> <p>Step 6: - Compute $M2' = PRNG(Auk \oplus R_t) \oplus Ack$ - Update tag values and keys</p>
<p>R_r, R_t: Random No. of reader/tag; Auk: Authentication key of tags shared with server; Rid: Reader ID; EPC: Electronic product code of tag; Ack: Access key of tags shared with server; I: Index value of tag; $H()$: One-way hash function; $data_i$: Secret information of the tag's object.</p>		

Figure 18. Hash-Based Mutual Authentication (HBA+) Protocol by Chang.

Z.Liu et al. [38] proposed variable linear shift-based authentication protocol (VLP) to support the implementation of RFID for the new EPC Gen2v2 standard, satisfy its security features of untraceability and access control, and reduce a tag's read range. In Figure 19, the protocol is based on a lightweight encryption function called Variable Linear Feedback Shift Register (VLFSR), which is implemented at the application-specific integrated circuit (ASIC) level. In every session, mutual authentication involves different random numbers from the tag and reader combined with the new secret value SID stored in the database to provide resistance against active attacks.

Another protocol (OMP) is proposed by Niu et al. [39] mainly for passive tag ownership transfer using a lightweight authentication mechanism to support EPC Gen2 standard. Since the ownership transfer is based on transferring the keys, the OMP protocol aims to prove the possession of the shared secret key to a tag and reader without disclosing it using ultra-lightweight permutation operation (Per), as in Figure 20. Yet, the protocol has no mechanism to check the freshness of the message that is sent by a legitimate reader.

Server S	Reader R	Tag T
<p>Step 5: Authenticate and Update</p> <p>1- Find an SID_j match in database based on UID</p> <p>2- Extract R_{t1}, R_{t2}</p> <p>3- $(R_{t2} R_{t1}) \oplus SID_j$</p> <p>4- Find m_j from M_j table based on SID_j</p> <p>5- $B_b = VLFSR (R_{t1} R_r, m_j)$</p> <p>6- If $B_b = B_r$, proceed to update</p> <ul style="list-style-type: none"> $m_{j+1} = (R_{t2} R_{t1}) \oplus m_j$ $SID_{j+1} = VLFSR (R_{t1} R_{t2}, m_j)$ Store new values in M_j and keep the old in M_{j-1} tables. <p>7- If $B_b \neq B_r$, find m_j and SID_j from M_{j-1} table and do step3</p> <p>8- If no match is found, protocol will stop.</p> <p>Step 6:</p> <p>- Send to reader $VLFSR (R_{t1} R_r, SID_j) \rightarrow$</p>	<p>Step 1:</p> <p>- Generate random R_r</p> <p>- Send R_r to tag \rightarrow</p> <p>Step 4:</p> <p>- Send to server $R_r, B_r, (R_{t2} R_{t1}) \oplus SID_j$ and UID \leftarrow</p> <p>Step 7:</p> <p>- Send to reader $VLFSR (R_{t1} R_r, SID_j) \rightarrow$</p>	<p>Step 2:</p> <p>- Generate random R_{t1}, R_{t2}</p> <p>- Secret value = m_j</p> <p>- $B_r = VLFSR (R_{t1} R_r, m_j)$</p> <p>Step 3:</p> <p>- Send to reader $B_r, (R_{t2} R_{t1}) \oplus SID_j \leftarrow$</p> <p>Step 8: Authenticate R and S</p> <p>- Authentication via received msg.</p> <p>- $m_{j+1} = R_{t2} R_{t1}) \oplus m_j$</p> <p>- $SID_{j+1} = VLFSR (R_{t1} R_{t2}, m_j)$</p>
<p>SID: Session secure ID of tag; UID: Unique ID of tag; R_r: Reader random No.; R_{t1}, R_{t2}: Random No. generated by tag; m_j: Secret value used in a session; VLFSR(): Variable LFSR function.</p>		

Figure 19. Variable Linear Shift-Based Authentication Protocol (VLP) by Z. Liu et al.

Server S	Reader R: K, K_M, EPC, RID_1	Tag T: K, K_M, EPC, RID, IDS
<p>Mutual Authentication:</p>	<p>1- Generate random rnd_1, rnd_2 of 96 bits</p> <p>2- $A_i = rnd_{1i} \oplus PRNG(K_i \oplus RID_{1i}) \oplus PRNG(K_i \oplus RID_{2i})$</p> <p>3- $B_i = rnd_{2i} \oplus PRNG(rnd_{1i} \oplus K_i)$</p> <p>4- $C_i = PRNG(rnd_{1i} \oplus RID_{1i}) \oplus PRNG(rnd_{2i} \oplus RID_{2i})$</p> <p>5- Send A_i, B_i, C_i to tag \rightarrow</p> <p>9- Verify D:</p> <p>- $D'_i = PRNG(K_{i+1} \oplus IDS_{i+1})$, where $i = 1$ to 6</p> <p>- If D is verified, tag is authenticated</p>	<p>6- Extract rnd_1, rnd_2</p> <p>- $rnd_{1i} = A_i \oplus PRNG(K_i \oplus RID_{1i}) \oplus PRNG(K_i \oplus RID_{2i})$</p> <p>- $rnd_{2i} = B_i \oplus PRNG(rnd_{1i} \oplus K_i)$</p> <p>- $C'_i = PRNG(rnd_{1i} \oplus RID_{1i}) \oplus PRNG(rnd_{2i} \oplus RID_{2i})$</p> <p>7- If $C = C'$, reader authenticated</p> <p>- $K_{i+1} = Per(rnd_{1i}, K_i) \oplus K(i+1 \text{ mod } 6)$</p> <p>- $IDS_{i+1} = Per(rnd_{2i}, K_i) \oplus K_i$</p> <p>- $D_i = PRNG(K_{i+1} \oplus IDS_{i+1})$, where $i = 1$ to 6</p> <p>8- Send D to reader \leftarrow</p>
<p>K: Secret shared key for owners; K_M: Master key to modify K. EPC: Static ID of a tag. RID: ID of reader owning tag. IDS: Pointer to tag database.</p>		

Figure 20. Passive Tag Ownership Authentication Protocol (OMP) Protocol by Niu.

Dass and Om [40] also proposed an efficient authentication protocol (SEAS) that uses lightweight operations and a pseudo-random number generator (PRNG) for a low computational cost.

Their scheme is based on a secure channel between the back-end server and reader, prestored tags' secret (SIDs) in the tags side, a one-way hash function of the tag ID in the server side, and rewritable memory with a flag indicator in the server side to update the secret values. Any change to the messages transmitted leads to terminate the communication during the verification to resist security attacks, as shown in Figure 21.

Server S	Reader R	Tag T
<p>Step 4: Search the database using $h(\text{ID})$: - Not Found: termination - Found: verify V $V' = \text{PRNG}(S_{\text{new}} \oplus N_R \oplus N_T)$ Send S_{new} to reader \rightarrow Flag = 0</p> <p>$V'' = \text{PRNG}(S_{\text{old}} \oplus N_R \oplus N_T)$ Send S_{old} to reader \rightarrow Flag = 1</p> <p>Step 6: - Flag = 0 $\rightarrow S = S_{\text{new}}$ $U = h(S_{\text{new}} \parallel M)$ $S_{\text{old}} = S_{\text{new}}$ $S_{\text{new}} = S_{\text{new}} \oplus U$</p> <p>- Flag = 1 $\rightarrow S = S_{\text{old}}$ $U = h(S_{\text{old}} \parallel M)$ $S_{\text{old}} = S_{\text{old}}$ $S_{\text{new}} = S_{\text{old}} \oplus U$</p>	<p>Step 1: - Generate random No. $N_R \rightarrow$</p> <p>Step 3: - $\leftarrow \{V, H, N_R, N_T\}$</p> <p>Step 5: - Reader takes S_{new} OR S_{old} - $M = \text{PRNG}(S_{\text{new, old}}, N_R)$ - $N = \text{PRNG}(M)$ - Send N to tag \rightarrow - \leftarrow send M to server</p>	<p>Step 2: - Generate random No. N_T - $V = \text{PRNG}(S \oplus N_R \oplus N_T)$ - $H = h(\text{ID})$ - $\leftarrow \{V, H, N_T\}$</p> <p>Step 6: - To authenticate reader: Calculate $M' = \text{PRNG}(S, N_R)$ Calculate $N' = \text{PRNG}(M')$ Verify $N' = N$ - If equal calculate $U = h(S \parallel M')$ - Update $S = S \oplus U$</p>
<p>$h()$: One-way hash function; N_R, N_T: Random No. generated by reader/tag; S: Secret value of tag; ID: ID pseudonym of tag; $S_{\text{new}}, S_{\text{old}}$: Current and old session secrets of tag.</p>		

Figure 21. Efficient Authentication Protocol (SEAS) by Dass and Om.

An alternative solution to replace the central database in the RFID system is to use a *serverless* model in which the database server does not maintain a connection with the readers and tags during the communication. Regarding this challenge, Mtita et al. [41] proposed (SAP), a serverless security protocol used for the mass authentication of RFID tags in the presence of untrusted readers. In SAP protocol, the reader and tag do not communicate with the back-end server; instead, they authenticate each other using only ephemeral of the tag's secrets that expire within a given time, as shown in Figure 22. Verification and authentication between reader and tag are done during the authentication phase to exchange the data and generate the session key locally in both tag and reader for their next communication. The protocol has also been proved using the *CryptoVerif* tool [42], which was shown to have low computation overhead and resources.

Server: S	Reader: R _j	Tag: T _i
<p>Initialization Phase:</p> <p>2- Generate K_{ij}, temp_{ij}, AR_{ij} (access right) for each tag derived from time window and start date $K_{ij} = \text{HMAC}_{\text{id}_i}(W_{sj} \parallel \text{AR}_{ij})$</p> <p>3- Build lists of authenticated tags L_j for R_j $L_j = \{(temp_{1j}, K_{1j}), (temp_{2j}, K_{2j}), \dots, (temp_{ij}, K_{ij})\}$</p> <p>4- Send L_j, AR_{ij}, W_{sj} to R_j →</p> <p>Mutual Authentication Phase:</p>	<p>1- Request permission from server S.</p> <p>1- Generate r_j 2- Send to tag A = W_{sj}, AR_{ij}, r_j →</p> <p>6- Verify H'_{ij} = HMAC_{K_{ij}}(r_i r_j) If equal: T_i is authenticated If not equal: K_{ij} is not in the list and tag is not authorized</p> <p>7- Generate timestamp t_j and calculate $V_{ij} = \text{HMAC}_{K_{ij}}(r_i \parallel t_j)$</p> <p>9- Send to tag C = t_j, V_{ij} →</p> <p>12- Generate session key $K_s = \text{HMAC}_{K_{ij}}(t_j \parallel r_i \parallel W_{sj})$</p>	<p>- T_i has Timestamp T_{sys} and id_i</p> <p>3- Generate r_i 4- H_{ij} = HMAC_{K_{ij}}(r_i r_j) 5- Send to reader B = H_{ij}, r_i ←</p> <p>10- Verify V'_{ij} = HMAC_{K_{ij}}(r_i t_j) If equal: R_j is authenticated</p> <p>11- Update T_{sys} = t_j</p> <p>13- Generate session key $K_s = \text{HMAC}_{K_{ij}}(t_j \parallel r_i \parallel W_{sj})$</p>
<p>T_{sys}: Tag static timestamp; t_j: Reader timestamp; id_i: Tag ID; K_{ij}: Tag's key; temp_{ij}: Temporary tag ID; AR_{ij}: Access rights; W_{sj}: Time window; K_s: Session key; L_j: List of authorized tags; r_i, r_j: Reader/Tag random No.</p>		

Figure 22. Serverless Security Authentication Protocol (SAP) by Mtitia.

4.4. Ultra-Lightweight Protocols

As mentioned earlier in this paper, passive tags are small chips with scarce resources that can only support low-cost operations. The goal of ultra-lightweight protocols is to reduce the cost of RFID systems at a minimum and provide strong security for promising future use. In this regard, Sundaresan et al. [43] introduced an ultra-lightweight serverless protocol (STS) using only simple XOR and 128-bit PRNG operations that require less than 2000 gates, three random number generation on the tag, and two message exchanges. In Figure 23, the STS protocol mechanism is to use a blind factor to hide the pseudo-random numbers that are used in communication between readers and tags to overcome impersonation attacks. RFID tag is also able to preserve its location privacy by responding as a noise tag. Moreover, the protocol does not employ a one-way hash function nor any encryption conforming to EPC C1 G2 Standards.

Server: t_s	Reader R	Tag T
<p>Setup Phase:</p> <ul style="list-style-type: none"> - Stores access list (AL) of all n tags: $h(TID_1, t_{s1}) = id_1, rts_1, ctr_1, ctrmax_1$ $h(TID_n, t_{sn}) = id_n, rts_n, ctr_n, ctrmax_n$ - Establish shared rts between a reader and each tag to be searched. <p>Search Phase:</p> <ul style="list-style-type: none"> - Server is offline. 	<ul style="list-style-type: none"> - Precompute and store $id = h(TID, t_s)$ <p>Step 1:</p> <ol style="list-style-type: none"> 1- Check that $ctr \leq ctrmax$ 2- Generate PRN r_r 3- $B = rts_j \oplus id_i$ 4- $M1 = id_i \oplus PRNG(rts_j \oplus r_r)$ 5- $M2 = r_r \oplus B$ 6- Broadcast $M1, M2$ to all tags \rightarrow <p>Step 3:</p> <ol style="list-style-type: none"> 1- Extract t_r from $M4$ 2- Verify $rts_j = M3 \oplus PRNG(id_j \oplus t_r)$ 3- If verified, tag is authenticated 4- Update $rts_j = M3 \oplus PRNG(id_j \oplus t_r)$ $ctr = ctr + 1$ 	<ul style="list-style-type: none"> - Stores $id = h(TID, t_s)$. - Stores for each reader: $rts_1, rts_1^{-1}, ctr_1, ctrmax_1, r_{r1}^{-1}$ $rts_m, rts_m^{-1}, ctr_m, ctrmax_m, r_{rm}^{-1}$ - $ctr = 0$. <p>Step 2:</p> <ol style="list-style-type: none"> 1- $B = rts \oplus id$ 2- Extract r_r from $M2 = B \oplus r_r$ 3- Check r_r: <ul style="list-style-type: none"> - If $r_r = r_r^{-1}$, a replayed msg, exit. - If $r_r \neq r_r^{-1}$, proceed 4- Verify $id = M1 \oplus PRNG(rts \oplus r_r)$: <ul style="list-style-type: none"> - If equal, reader is authenticated - If not equal, repeat using rts^{-1}. - If not equal, respond with λ and exit. 5- If id is verified, check if $ctr < ctrmax$: <ul style="list-style-type: none"> - Generate PRN t_r - $M3 = rts \oplus PRNG(id \oplus t_r)$ - $M4 = t_r \oplus B$ 6- Update $r_r^{-1} = r_r$ 7- If id is verified using rts: <ul style="list-style-type: none"> - Update $rts^{-1} = rts$ - $rts = PRNG(rts)$ - $ctr = ctr + 1$ 8- Send $M3, M4$ to reader \leftarrow
<p>AL: Access list for the reader; t_s: Secret key of tag; rts, rts^{-1}: Shared secrets between reader/tag; B: Blind factor to hide PRN; ctr: Counter value; $ctrmax$: Number of times a reader is pre-authorized to search; TID: Tag ID; id: hashed value of TID; r_r: Random No. of reader in current session.</p>		

Figure 23. Ultra-Lightweight Serverless Authentication Protocol (STS) by Sundaresan.

Aggarwal and Das [44] proposed the CHW+ protocol, which is based on a previous version introduced by Y. Chen, Wang, and Hwang (CWH) [45]. In Figure 24, the protocol CHW+ solves the problem of full disclosure attack due to the simple XOR operation that is used in the authentication message, which uses the bit rotation and shifting operation on the message before transmission to increase the protocol complexity. CWH+ protocol is resistant to replay attack, forge attack, and DoS with a very efficient computation.

Huang and Li [46] proposed and implemented two improved protocols of RFID mutual authentication based on generating the PadGen function in the ISO 18000-6C [47] protocol to protect the memory with a 32-bit access password. The concept of their protocols is to cover up the tag's access password (Apwd) before transmitting the data using a set of 16-bit random numbers such as RTx and RMx. One of the improved schemes, PadGen with XOR (PGX), implements XOR operation between the random number sets and the PadGen function; the other protocol, PadGen with Mod (PGM), implements a Modulo operation (MOD9) in the eight-bit half of the 16-bit random number set (RTx, RMx) to be used in the PadGen function. Both improved schemes conform to the EPC C1 G2 standard, do not require any hash function or key exchange, do not involve synchronization for hash or key values, and also show better efficiency during implementation. The security level of the MOD scheme is higher due to the low-cost implementation, but requires a higher computation cost in PadGen than XOR.

Server S	Reader R	Tag T
<ul style="list-style-type: none"> - $n = \text{weight}(r)$ - $r' = \text{Rot}(r, n)$ - $r'_{\text{prev}} = \text{Rot}(r_{\text{prev}}, n)$ - Retrieve and verify TID to authenticate tag. - $r'' = \text{Rot}(r', n)$ - $r''_{\text{prev}} = \text{Rot}(r'_{\text{prev}}, n)$ - $t2 = (TID + r''_{\text{prev}}) \wedge r''$ - Update $r_{\text{old}} = r_{\text{prev}}, r_{\text{prev}} = r'$ - Send $t2$ to tag \rightarrow 	<p>Step 1:</p> <ul style="list-style-type: none"> - Generate random r. - $s = \text{RID} \oplus r$ - $n1 = \text{weight}(\text{RID})$, $n1 = \text{no. of bit value 1 of RID}$ - $s' = \text{Rot}(s, n1)$ - Send s' to tag \rightarrow <p>Step 3:</p> <ul style="list-style-type: none"> - Forward $t1, r$ to server \leftarrow 	<p>Step 2:</p> <ul style="list-style-type: none"> - $s = \text{Rot}(s', n1)$ - Retrieve $r = \text{RID} \oplus s$ - $n1 = \text{weight}(r)$ - $r' = \text{Rot}(r, n1)$ - $r'_{\text{prev}} = \text{Rot}(r_{\text{prev}}, n1)$ - $t1 = (TID \oplus r'_{\text{prev}}) + (r' \wedge r'_{\text{prev}})$ - Send $t1$ to reader \leftarrow <p>Step 4:</p> <ul style="list-style-type: none"> - Compute r'', r''_{prev} as the server - $t'2 = (TID + r''_{\text{prev}}) \wedge r''$ - Verify $t'2 = t2$ - Update $r_{\text{prev}} = r'$
<p>RID: Reader ID; Rot(): Rotation function; TID: Tag ID; Weight(r): number of 1's in the binary string shifts r to the left for n bits.</p>		

Figure 24. Improved Authentication Protocol (CWH+) by Aggarwal and Dass.

Huang and Jiang [48] proposed an ultra-lightweight reader–tag mutual authentication protocol (MACC) based on Chien and Chen’s protocol [49] to overcome forge attacks, DoS, and forward security attacks. Although the improved scheme uses only lightweight operations such as RNG, PRNG, and XOR, it involves an exhaustive search in the database for tag pseudo-IDs in every session that leads to computational overhead, as shown in Figure 25. It also fails to resist tracking attacks.

Server S	Reader R	Tag T
<p>Step 4:</p> <ul style="list-style-type: none"> - Verify V_R using reader ID and r_1 - Search database for tag PID_i - Verify M_1 as: $r_2 = M_1 \oplus N_i^{\text{old}}$ OR $r_2 = M_1 \oplus N_i^{\text{new}}$ - Verify M_2 as: $M_2 = P(\text{EPC}_i r_1 r_2 K_i^{\text{old}}$ or $K_i^{\text{new}})$ - $M_3 = P(\text{EPC}_i r_2 N_i^x K_i^x)$ $x = \text{old or new}$ - Send M_3 to reader \rightarrow <p>- If $x = \text{new}$, proceed to update $N_i^{\text{old}} = N_i^{\text{new}}, N_i^{\text{new}} = P(N_i^{\text{new}} \oplus r_2)$ $K_i^{\text{old}} = K_i^{\text{new}}, K_i^{\text{new}} = P(K_i^{\text{new}} \oplus r_2)$ $PID_i^{\text{old}} = PID_i^{\text{new}}, PID_i^{\text{new}} = P(PID_i^{\text{new}} \oplus r_2)$</p>	<p>Step 1:</p> <ul style="list-style-type: none"> - Generate r_1 - $V_R = h(\text{RID}_i \oplus r_1)$ - Send to tag $r_1 \rightarrow$ <p>Step 3:</p> <ul style="list-style-type: none"> - Send to server ($M_1, M_2, PID_i, r_1, V_R$) <p>Step 5:</p> <ul style="list-style-type: none"> - Forward M_3 to tag \rightarrow 	<p>Step 2:</p> <ul style="list-style-type: none"> - Generate r_2 - $M_1 = N_i \oplus r_2$ - $M_2 = P(\text{EPC}_i r_1 r_2 K_i)$ - Send to reader (M_1, M_2, PID_i) \leftarrow <p>Step 6:</p> <ul style="list-style-type: none"> - Verify $M_3 = P(\text{EPC}_i r_2 N_i K_i)$ - $N_i = P(N_i \oplus r_2)$ - $K_i = P(K_i \oplus r_2)$ - $PID_i = P(PID_i \oplus r_2)$
<p>P: Access key with reader; PID: Pseudonym ID of tag; Ni: Nonce; EPC: Electronic product code for the tag; Ki: Authentication key.</p>		

Figure 25. Ultra-Lightweight Reader–Tag Mutual Authentication Protocol (MACC) by Huang and Jiang.

Huang and Jiang [48] proposed another mutual authentication protocol (MACD) based on Chen and Deng’s scheme [50] to overcome forge attacks, DoS, replay attacks, and mainly the tag identification time. It is shown in Figure 26 that the MACD protocol uses ultra-lightweight operations and achieves a lower communication cost between tag and reader than the other improved scheme, MACC.

Server S	Reader R	Tag T
<p>Step 4:</p> <ul style="list-style-type: none"> - Search database for EPC_i match - $r_2 = B \oplus (P_i^{old} r_1)$ OR $B \oplus (P_i^{new} r_1)$ - $A' = r_1 \oplus r_2 \oplus P_i^{old}$ or P_i^{new} - Verify $M_1 = CRC(EPC_i A' B K_i^{old}$ or $K_i^{new})$ - $M_2 = CRC(EPC_i r_2 P_i^x K_i^x)$ $x = old, new$ - Send M_2 to reader \rightarrow <p>- If $x = new$, proceed to update $P_i^{old} = P_i^{new}, P_i^{new} = P(P_i^{new} \oplus r_2)$ $K_i^{old} = K_i^{new}, K_i^{new} = P(K_i^{new} \oplus r_2)$</p>	<p>Step 1:</p> <ul style="list-style-type: none"> - Generate r_1 - Send r_1 to tag \rightarrow <p>Step 3:</p> <ul style="list-style-type: none"> - Send to server (M_1, B, r_1) <p>Step 5:</p> <ul style="list-style-type: none"> - Send M_2 to tag \rightarrow 	<p>Step 2:</p> <ul style="list-style-type: none"> - Generate r_2 - $A = r_1 \oplus r_2 \oplus P_i$ - $B = P(P_i r_1) \oplus r_2$ - $M_1 = CRC(EPC_i A B K_i)$ - Send to reader (M_1, B) \leftarrow <p>Step 6:</p> <ul style="list-style-type: none"> - Verify $M_2 = CRC(EPC_i r_2 P_i K_i)$ - $P_i = P(P_i \oplus r_2)$ - $K_i = P(K_i \oplus r_2)$
<p>P: Access key with reader; K_i: Authentication key; N_i: Nonce; EPC: Electronic product code for the tag; CRC(): Cyclic redundancy check function.</p>		

Figure 26. Mutual Authentication Protocol (MACD) by Huang and Jiang.

Considering the complexity of the authentication protocol, Hopper and Blum proposed the first HB protocol to identify unaided humans to computers [51]. Many authors adopted the idea of HB protocol to identify tags in RFID networks. As a matter of fact, HB family protocols are based on the hard problems of Learning Parity with Noise (LPN), which involves the calculation of inner products of binary vectors and Bernoulli noise bit generation [52]. In this regard, Lin and Song [53] proposed HBROT, which is one of the latest HB protocols that produces the key in each authentication round using the rotation function. The protocol is considered to be secure against most of the RFID attacks.

Another improvement of the HB protocol is proposed by Juels and Weis [54] as (HB+) to overcome the weaknesses of the original HB. The HB+ protocol involves two secret keys, x and y , which are used with shared blind vectors between the reader and tag. The reader and tag verify the values that are computed to perform the mutual authentication. Later, the protocol is reported by Gilbert et al. [55] to be vulnerable to the man-in-the-middle attack (MIM). Hence, Ouaskou et al. [56] proposed a variant of HB protocol based on Permutation function (HBPER). The protocol performs a permutation of the keys x, y during each round of the protocol to update the value of the keys, as shown in Figure 27. This method secures the protocol against the MIM attack that is reported in the HB+ protocol, although both protocols HB+ and HBPER almost have the same complexity.

Reader R	Tag T
$x = x_{k-1}, \dots, x_1, x_0$ $y = y_{k-1}, \dots, y_1, y_0$	$x = x_{k-1}, \dots, x_1, x_0$ $y = y_{k-1}, \dots, y_1, y_0$
Step 1: - Generate random challenge (a) - Send (a) to T →	Step 2: - Compute $y = \text{Per}(y,a)$ - Compute $x = \text{Per}(x,a)$ - Compute $z = a \cdot x \oplus v$ ($v = \text{noise bit}; v = 1 \text{ with probability of } \eta$) - Send z to R ←
Step 3: - Compute $y = \text{Per}(y,a)$ - Compute $x = \text{Per}(x,a)$ - Verify $z = a \cdot x$	
x : Shared key by tag and reader of k-bit; k : Length of secret keys; y : Shared key by tag and reader of k-bit; $\eta = \text{noise level} \in]0,1/2[$.	

Figure 27. A Variant of HB Protocol Based on Permutation Function (HBPER) by Ouaskou et al.

5. Analysis and Security Evaluation

In this section, we compare the different protocols in terms of computation, security requirements, and attacks resistance. Since the passive tag used in the RFID system has limited computation capabilities and resources, it is important to consider the computation and security features for the appropriate application. Table 2 demonstrates the different operations computed by the tag in each protocol, and the communication overhead based on the number of transmitted messages between tag and reader.

Table 2. Comparison of the Computation Cost on Tag.

Protocol	Operations	Tag Passes	Reader Passes	Tag Overhead
SB-A [11]	$1 T_{ENC} + 2 T_{DEC} + 2 T_{PRNG}$	2	3	High
SB-B [11]	$2 T_{ENC} + 2 T_{DEC} + 2 T_{PRNG}$	2	3	High
EMA [15]	$2 T_{SMUL} + 2 T_{CH}$	2	1	High
ECU [13]	$2 T_{SMUL} + 2 T_{CH}$	1	1	High
SPA [14]	$4 T_{SMUL} + 1 T_{CH}$	1	1	High
PII [16]	$4 T_{SMUL} + 3 T_{CH}$	1	1	High
RUND [17]	$2 T_H \text{ OR } 1 T_{ENC} + 1 T_{PRNG}$	1	2	High
IECC [19]	$2 T_{SMUL} + 2 T_H$	1	2	High
EECC [20]	$2 T_{SMUL} + 2 T_H$	1	2	High
RBAC [21]	$2 T_{ENC} + 2 T_{DEC} + 1 T_{PRNG}$	2	2	High
DRAP [33]	$1 T_{ENC} + 3 T_{XOR} + 3 T_H + 1 T_{RNG} + 2 T_{PRNG}$	1	2	High
NRS [22]	$10 T_{XOR} + 3 T_H$	4	5	Medium
NRS+ [10]	$10 T_{XOR} + 6 T_H$	4	5	Medium
NRS++ [23]	$8 T_{XOR} + 4 T_H$	1	2	Medium
ACSP [24]	$3 T_{XOR} + 7 T_H + 4 T_{CRC}$	1	4	Medium
ACSP+ [25]	$4 T_{XOR} + 8 T_H$	2	4	Medium
ACSP++ [23]	$6 T_{XOR} + 8 T_H$	1	2	Medium
MASS [31]	$4 T_{XOR} + 2 T_H + 1 T_{RNG}$	1	2	Medium
EP-UAP [32]	$2 T_H + 1 T_{RNG}$	1	2	Medium
GUPA [34]	$2 T_H + 3 T_{PRNG} + 19 T_{BIT}$	3	3	Medium
HBA [37]	$6 T_{XOR} + 2 T_H + 1 T_{RNG} + 4 T_{MOD}$	1	2	Medium
VLP [38]	$2 T_{XOR} + 2 T_{RNG} + 3 T_{BIT} + 2 T_{VLFPSR}$	1	2	Medium
SEAS [40]	$1 T_{XOR} + 2 T_H + 1 T_{RNG} + 3 T_{PRNG} + 1 T_{BIT}$	1	2	Medium
SAP [41]	$2 T_H + 2 T_{RNG}$	1	2	Medium
LAP [26]	$2 T_{XOR} + 1 T_{RNG} + 2 T_{PRNG} + 1 T_{ROT} + 1 T_{SHIFT}$	2	2	Low
Flyweight [29]	$5 T_{PRNG}$	3	3	Low
FTest [35]	$1 T_{XOR} + 3 T_{CRC}$	3	2	Low
ACS [36]	$3 T_{XOR} + 2 T_{ITER} + 1 T_{SPR}$	1	2	Low
HBA+ [6]	$7 T_{XOR} + 1 T_{RNG} + 5 T_{PRNG}$	1	2	Low
OMP [39]	$12 T_{XOR} + 6 T_{PRNG} + 2 T_{PER}$	1	1	Low
STS [43]	$7 T_{XOR} + 3 T_{PRNG}$	1	1	Low

Table 4. Cont.

	SR1	SR2	SR3	SR4	SR5	SR6	SR7	SR8
LAP [26]	Y	Y	Y	N	Y	*	N	Y
Flyweight [29]	Y	Y	Y	Y	Y	Y	Y	Y
MASS [31]	Y	Y	N	*	Y	*	*	Y
EP-UAP [32]	N	Y	Y	Y	*	*	Y	Y
DRAP [33]	Y	*	*	Y	*	*	Y	Y
GUPA [34]	Y	Y	Y	Y	Y	*	Y	Y
FTest [35]	N	Y	Y	Y	Y	*	Y	Y
ACS [36]	Y	*	*	Y	*	*	Y	Y
HBA [37]	Y	Y	Y	Y	Y	*	Y	Y
HBA+ [6]	Y	Y	Y	Y	Y	*	Y	Y
VLP [38]	Y	Y	Y	Y	Y	*	Y	Y
OMP [39]	Y	Y	Y	Y	Y	Y	Y	Y
SEAS [40]	Y	Y	Y	Y	Y	*	Y	Y
SAP [41]	Y	Y	Y	*	*	*	*	Y
STS [43]	Y	Y	Y	Y	Y	*	Y	Y
CWH+ [44]	Y	Y	Y	*	Y	*	*	Y
PGX [46]	Y	*	*	N	*	*	N	Y
PGM [46]	Y	*	*	N	*	*	N	Y
MACC [48]	Y	Y	Y	N	Y	*	N	Y
MACD [48]	Y	Y	Y	Y	Y	*	Y	Y
HBROT [53]	Y	Y	Y	Y	Y	*	*	Y
HBPER [56]	Y	Y	Y	Y	Y	*	*	Y

SR1: mutual authentication, SR2: confidentiality, SR3: message integrity, SR4: privacy, SR5: forward secrecy, SR6: backward secrecy, SR7: tag anonymity, SR8: conforming to EPC standard, Y: satisfied, N: not satisfied, *: not applicable.

6. Conclusions

Recent RFID authentication protocols are proposed to develop an efficient and secure RFID system. This survey is conducted to review and compare different RFID authentication protocols of low-cost passive tags for better utilization in the appropriate application. We demonstrate in this study the security requirements of an RFID system that must be satisfied, so the system could be able to defend major attacks such as replay, man-in-the-middle, impersonation, desynchronization, DoS, and more. We further identify the category levels of the protocols based on the operation complexity on the tag side, and compare the protocols based on the tag computation cost. Since the RFID passive tag has limited resources to compute complex operations, the heavyweight and simple-weight protocols are not feasible for practical implementation. However, lightweight and ultra-lightweight protocols use only simple operations within the tag computation limits, and show the lowest tag overhead level. Lightweight and ultra-lightweight protocols are considered the most suitable for the current applications. Another vital aspect when considering the appropriate RFID protocol is the security resistance to the attacks. We examined the security threats in each protocol presented in the review. We found out that Chang et al. [6], Farash [19], Zhang and Qi [20], X. Chen et al. [23], Liu et al. [34], Lin and Song [53], and Ouaskou et al. [56] protocols successfully resist all of the major attacks. Although the other protocols could not resist all of the attacks, they could perform better than the fully secure protocols in term of computation cost; examples include the protocols presented in Farash [19], Zhang and Qi [20], X. Chen et al. [23], and Liu et al. [34], which have high computation overhead on the tag side. We encourage researchers to pay attention to the forward and backward security, since most protocols do not reflect on these two types of attacks. Finally, maintaining the basic security requirements for an RFID system is required to achieve protection against the mentioned attacks in this article. Our assessment is that only the protocols of Niu et al. [39] and X. Chen et al. [23] satisfy all of the security requirements to maintain the system in a stable and available state. Even though this review shows security variation among the reviewed protocols, each one could still be a preference over others, depending on the requirements of the application in hand.

Author Contributions: Conceptualization, R.B. and A.A.; Methodology, R.B. and A.A.; Formal Analysis, R.B. and A.A.; Writing-Original Draft Preparation, R.B.; Writing-Review & Editing, A.A., and R.B.; Supervision, A.A.; Project Administration, A.A.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zhao, Z. A Secure RFID Authentication Protocol for Healthcare Environments Using Elliptic Curve Cryptosystem. *J. Med. Syst.* **2014**, *38*, 46. [CrossRef] [PubMed]
2. Roberts, C.M. Radio frequency identification (RFID). *Comput. Secur.* **2006**, *25*, 18–26. [CrossRef]
3. Xie, L.; Yin, Y.; Vasilakos, A.V.; Lu, S. Managing RFID Data: Challenges, Opportunities and Solutions. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 1294–1311. [CrossRef]
4. Pagán Alexander, J.; Baashirah, R.; Abuzneid, A. Comparison and Feasibility of Various RFID Authentication Methods Using ECC. *Sensors* **2018**, *18*, 2902. [CrossRef] [PubMed]
5. Syamsuddin, I.; Han, S.; Dillon, T. A survey on low-cost RFID authentication protocols. In Proceedings of the 2012 International Conference on Advanced Computer Science and Information Systems (ICACSIS), Depok, Indonesia, 1–2 December 2012.
6. Chang, C.C.; Chen, W.Y.; Cheng, T.F. A Secure RFID Mutual Authentication Protocol Conforming to EPC Class 1 Generation 2 Standard. In Proceedings of the 2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Kitakyushu, Japan, 27–29 August 2014.
7. Swedberg, C. Iotera Develops Active RFID Tag with 4-Mile Read Range. Available online: <https://www.rfidjournal.com/articles/view?11374> (accessed on 22 October 2018).
8. Roque, P. Performance Analysis of Effective Range and Orientation for UHF Passive RFID. In *Department of Electrical and Computer Engineering; Air Force Institute of Technology: Dayton, OH, USA*, 2008.
9. Zheng, L.; Xue, Y.; Zhang, L.; Zhang, R. Mutual Authentication Protocol for RFID Based on ECC. In Proceedings of the 2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC), Guangzhou, China, 21–24 July 2017.
10. Alagheband, M.R.; Aref, M.R. Simulation-Based Traceability Analysis of RFID Authentication Protocols. *Wirel. Pers. Commun.* **2014**, *77*, 1019–1038. [CrossRef]
11. Wang, J.; Floerkemeier, C.; Sarma, S.E. Session-based security enhancement of RFID systems for emerging open-loop applications. *Pers. Ubiquitous Comput.* **2014**, *18*, 1881–1891. [CrossRef]
12. ISO. ISO/IEC DIS 9798-2. In *Information Technology-Security Techniques-Entity Authentication—Part 2: Mechanisms Using Authenticated Encryption*; International Organization for Standardization: Geneva, Switzerland, 2017.
13. Ryu, E.-K.; Kim, D.-S.; Yoo, K.-Y. On Elliptic Curve Based Untraceable RFID Authentication Protocols. In Proceedings of the 3rd ACM Workshop on Information Hiding and Multimedia Security, Portland, OR, USA, 17–19 June 2015.
14. Songhela, R.; Das, M.L. *Yet Another Strong Privacy-Preserving RFID Mutual Authentication Protocol*; Springer International Publishing: Cham, Switzerland, 2014.
15. Chou, J.-S. An efficient mutual authentication RFID scheme based on elliptic curve cryptography. *J. Supercomput.* **2014**, *70*, 75–94. [CrossRef]
16. Chen, Y.; Chou, J.-S. ECC-based untraceable authentication for large-scale active-tag RFID systems. *Electron. Commer. Res.* **2015**, *15*, 97–120. [CrossRef]
17. Yao, Q.; Ma, J.; Cong, S.; Li, X.; Li, J. Attack gives me power: DoS-defending constant-time privacy-preserving authentication of low-cost devices such as backscattering RFID tags. In Proceedings of the 3rd ACM Workshop on Mobile Sensing, Computing and Communication, Paderborn, Germany, 5–8 July 2016.
18. EPCglobal, E.P.C. Radio-Frequency Identity Protocols Generation-2 UHF RFID. In *Specification for RFID Air Interface Protocol for Communications at 860 MHz–960 MHz Version 2.0.1 Ratified*; EPCglobal Inc.: Lawrenceville, NJ, USA, 2015.
19. Farash, M.S. Cryptanalysis and improvement of an efficient mutual authentication RFID scheme based on elliptic curve cryptography. *J. Supercomput.* **2014**, *70*, 987–1001. [CrossRef]

20. Zhang, Z.; Qi, Q. An Efficient RFID Authentication Protocol to Enhance Patient Medication Safety Using Elliptic Curve Cryptography. *J. Med. Syst.* **2014**, *38*, 47. [[CrossRef](#)] [[PubMed](#)]
21. Chen, B.-C.; Yang, C.T.; Yeh, H.T.; Lin, C.C. Mutual Authentication Protocol for Role-Based Access Control Using Mobile RFID. *Appl. Sci.* **2016**, *6*, 215. [[CrossRef](#)]
22. Fernando, H.; Abawajy, J. Mutual Authentication Protocol for Networked RFID Systems. In Proceedings of the 2011 IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications, Changsha, China, 16–18 November 2011.
23. Chen, X.; Cao, T.; Zhai, J. Untraceability Analysis of Two RFID Authentication Protocols. *Chin. J. Electron.* **2016**, *25*, 912–920. [[CrossRef](#)]
24. Chen, C.; Qian, Z.; You, I.; Hong, J.; Lu, S. ACSP: A Novel Security Protocol against Counting Attack for UHF RFID Systems. In Proceedings of the 2011 Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, Seoul, Korea, 30 June–2 July 2011.
25. Safkhani, M.; Bagheri, N.; Mahani, A. On the security of RFID anti-counting security protocol (ACSP). *J. Comput. Appl. Math.* **2014**, *259*, 512–521. [[CrossRef](#)]
26. Chien, H.-Y.; Huang, C.-W. A Lightweight Authentication Protocol for Low-Cost RFID. *J. Signal Process. Syst.* **2010**, *59*, 95–102. [[CrossRef](#)]
27. Li, Y.Z.; Cho, Y.B.; Um, N.K.; Lee, S.H. Security and Privacy on Authentication Protocol for Low-cost RFID. In Proceedings of the 2006 International Conference on Computational Intelligence and Security, Guangzhou, China, 3–6 November 2006.
28. ISO. ISO/IEC 15693-2:2006. In *Identification Cards-Contactless Integrated Circuit Cards-Vicinity Cards—Part 2: Air Interface and Initialization*; International Organization for Standardization: Geneva, Switzerland, 2006.
29. Burmester, M.; Munilla, J. Lightweight RFID authentication with forward and backward security. *ACM Trans. Inf. Syst. Secur.* **2011**, *14*, 1–26. [[CrossRef](#)]
30. Lee, S.; Asano, T.; Kim, K. RFID Mutual Authentication Scheme based on Synchronized Secret Information. In Proceedings of the 2006 Symposium on Cryptography and Information Security, Hiroshima, Japan, 17–20 January 2006.
31. Zuo, Y. Survivability Experiment and Attack Characterization for RFID. *IEEE Trans. Dependable Secur. Comput.* **2012**, *9*, 289–302. [[CrossRef](#)]
32. Lee, K.; Nieto, J.M.G.; Boyd, C. Improving the efficiency of RFID authentication with pre-computation. In Proceedings of the Tenth Australasian Information Security Conference, Melbourne, Australia, 31 January–3 February 2012.
33. Rahman, F.; Ahamed, S.I. DRAP: A Robust Authentication protocol to ensure survivability of computational RFID networks. In Proceedings of the 27th Annual ACM Symposium on Applied Computing, Trento, Italy, 26–30 March 2012.
34. Liu, H.; Liu, H.; Ning, H.; Zhang, Y.; He, D.; Xiong, Q.; Yang, L.T. Grouping-Proofs-Based Authentication Protocol for Distributed RFID Systems. *IEEE Trans. Parallel Distrib. Syst.* **2013**, *24*, 1321–1330. [[CrossRef](#)]
35. Rahman, F.; Ahamed, S.I. Efficient detection of counterfeit products in large-scale RFID systems using batch authentication protocols. *Pers. Ubiquitous Comput.* **2014**, *18*, 177–188. [[CrossRef](#)]
36. Keqiang, Y.; Lingling, S.; Xing, Q.; Zhonghua, Z. Design of anti-collision integrated security mechanism based on chaotic sequence in UHF RFID system. *China Commun.* **2014**, *11*, 137–147. [[CrossRef](#)]
37. Cho, J.-S.; Jeong, Y.-S.; Park, S.O. Consideration on the brute-force attack cost and retrieval cost: A hash-based radio-frequency identification (RFID) tag mutual authentication protocol. *Comput. Math. Appl.* **2015**, *69*, 58–65. [[CrossRef](#)]
38. Liu, Z.; Liu, Z.; Liu, D.; Li, L.; Lin, H.; Yong, Z. Implementation of a New RFID Authentication Protocol for EPC Gen2 Standard. *IEEE Sens. J.* **2015**, *15*, 1003–1011.
39. Niu, H.; Taqieddin, E.; Jagannathan, S. EPC Gen2v2 RFID Standard Authentication and Ownership Management Protocol. *IEEE Trans. Mob. Comput.* **2016**, *15*, 137–149. [[CrossRef](#)]
40. Dass, P.; Om, H. A Secure Authentication Scheme for RFID Systems. *Procedia Comput. Sci.* **2016**, *78*, 100–106. [[CrossRef](#)]
41. Mtita, C.; Laurent, M.; Delort, J. Efficient serverless radio-frequency identification mutual authentication and secure tag search protocols with untrusted readers. *IET Inf. Secur.* **2016**, *10*, 262–271. [[CrossRef](#)]
42. Blanchet, B. *CryptoVerif: Cryptographic Protocol Verifier in the Computational Model*; IEEE: Oxford, UK, 2010; pp. 16–30.

43. Sundaresan, S.; Doss, R.; Piramuthu, S.; Zhou, W. Secure Tag Search in RFID Systems Using Mobile Readers. *IEEE Trans. Dependable Secur. Comput.* **2015**, *12*, 230–242. [[CrossRef](#)]
44. Aggarwal, R.; Das, M.L. RFID security in the context of “internet of things”. In Proceedings of the First International Conference on Security of Internet of Things, Kollam, India, 17–19 August 2012.
45. Chen, Y.C.; Wang, W.L.; Hwang, M.S. RFID Authentication Protocol for Anti-Counterfeiting and Privacy Protection. In Proceedings of the 9th International Conference on Advanced Communication Technology, Kobe, Japan, 12–14 February 2007.
46. Huang, Y.J.; Lin, W.C.; Li, H.L. Efficient Implementation of RFID Mutual Authentication Protocol. *IEEE Trans. Ind. Electron.* **2012**, *59*, 4784–4791. [[CrossRef](#)]
47. ISO. ISO/IEC 18000-6:2013. In *Information Technology-Radio Frequency Identification for Item Management—Part 6: Parameters for Air Interface Communications at 860 MHz to 960 MHz General*; International Organization for Standardization: Geneva, Switzerland, 2013.
48. Huang, Y.C.; Jiang, J.R. Ultralightweight RFID Reader-Tag Mutual Authentication. In Proceedings of the 2015 IEEE 39th Annual Computer Software and Applications Conference, Taichung, Taiwan, 1–5 July 2015.
49. Chien, H.-Y.; Chen, C.-H. Mutual authentication protocol for RFID conforming to EPC Class 1 Generation 2 standards. *Comput. Stand. Interfaces* **2007**, *29*, 254–259. [[CrossRef](#)]
50. Chen, C.-L.; Deng, Y.-Y. Conformation of EPC Class 1 Generation 2 standards RFID system with mutual authentication and privacy protection. *Eng. Appl. Artif. Intell.* **2009**, *22*, 1284–1291. [[CrossRef](#)]
51. Hopper, N.J.; Blum, M. Secure Human Identification Protocols. In Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology, Gold Coast, Australia, 9–13 December 2001; Springer: Berlin/Heidelberg, Germany; pp. 52–66.
52. Piramuthu, S. HB and Related Lightweight Authentication Protocols for Secure RFID Tag/Reader Authentication. *COLLECTeR Eur.* **2006**, *2006*, 239.
53. Lin, Z.; Song, J.S. An Improvement in HB-Family Lightweight Authentication Protocols for Practical Use of RFID System. *J. Adv. Comput. Netw.* **2013**, *1*, 61–65. [[CrossRef](#)]
54. Juels, A.; Weis, S.A. Authenticating Pervasive Devices with Human Protocols. In *Annual International Cryptology Conference*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 293–308.
55. Gilbert, H.; Robshaw, M.; Sibert, H. Active attack against HB/sup +/-: A provably secure lightweight authentication protocol. *Electron. Lett.* **2005**, *41*, 1169–1170. [[CrossRef](#)]
56. Ouaskou, M.; Lahmer, M.; Belkasm, M. A variant of HB protocols based on permutation for low-cost RFID. In Proceedings of the 2015 International Conference on Wireless Networks and Mobile Communications (WINCOM), Marrakech, Morocco, 20–23 October 2015.



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).