symmetry

MDPI

Article

How to Design a Secure Anonymous Authentication and Key Agreement Protocol for Multi-Server Environments and Prove Its Security

Yun-Hsin Chuang [1], Chin-Laung Lei [1] and Hung-Jr Shiu [2,*]

[1] Department of Electrical Engineering, National Taiwan University, Taipei 10617, Taiwan; ntueeyun@gmail.com (Y.-H.C.); cllei@ntu.edu.tw (C.-L.L.)
[2] Department of Computer Science, Tunghai University, Taichung 407224, Taiwan
* Correspondence: hjshiu@thu.edu.tw

Abstract: An anonymous authentication and key agreement (AAKA) protocol provides anonymous members symmetric authentication and establishes a symmetric session key for secure communication in public networks. Today, numerous popular remote services are based on multi-server architecture, such as the internet of things (IoT), smart cities, cloud services, vehicular ad hoc networks (VANET), and telecare medicine information systems (TMIS). Many researchers have attempted to design AAKA protocols in multi-server environments for various applications. However, many of these have security defects, even if they have so-called "formal" security proofs. In this paper, we analyze related AAKA protocols to identify the common design defects, expound the process of designing secure AAKA protocols, and explain why the present AAKA protocols still suffer attacks, despite having security proofs. We instruct readers on how to design a secure AAKA protocol and how to prove the security. This paper will therefore be helpful for the design of new AAKA protocols, and for ensuring their security.

Keywords: anonymity; authentication; biometric; key exchange; multi-server; privacy; three-factor authentication

check for updates

Citation: Chuang, Y.-H.; Lei, C.-L.; Shiu, H.-J. How to Design a Secure Anonymous Authentication and Key Agreement Protocol for Multi-Server Environments and Prove Its Security. *Symmetry* **2021**, *13*, 1629. https://doi.org/10.3390/sym13091629

Academic Editor: Antonio Palacios

Received: 9 July 2021
Accepted: 30 August 2021
Published: 4 September 2021

1. Introduction

An authentication and key agreement (AKA) protocol enables users to log in to remote servers over insecure channels to confirm their symmetric authenticity with each other and create a symmetric session key, which is used to securely communicate in the session. The first AKA protocol, which is password-based, with the server verifying a user by username and the corresponding password, was proposed by Lamport [1] in 1981; however, a password-based authentication protocol requires password tables and is vulnerable to password replay attacks, where an intruder replays the previously intercepted password to successfully log in to the server. In this case, Hwang [2] proposed the first two-factor AKA protocol in 1990, which uses the smart card as the second factor to avoid password replay attack. Three-factor AKA protocols have recently attracted attention, as they can withstand smart card loss attack. In a ***three-factor*** AKA, the authenticity of the user is confirmed by three distinct factors, which are typically the password, the smart card, and a form of biometric identification.

Traditional AKA protocols are only suitable for single-server architectures; today, however, lots of commercial services are built on multi-server architectures. Recently, many ***multi-server*** AKA protocols have been proposed, in which the servers are regarded as independent entities with distinct secret keys.

The following are some general security defects for a three-factor AKA protocol:

- (A1) Replay attack: this is a type of man-in-the-middle attack, where the attacker maliciously repeats or delays a valid data transmission.

- (A2) Privileged insider attack (user/server impersonation attack): a legal user or a legal server has the ability to impersonate another user or server.
- (A3) Smart-card-loss-attack (offline password/identity guessing attack): when an attacker steals the smart card of a user, the attacker can guess the password or identity of the user offline.
- (A4) Failure to ensure forward secrecy: the session keys are compromised because the long-term secret keys are compromised.
- (A5) Failure to provide user anonymity: the real identity of an anonymous user is disclosed to the third party.
- (A6) Side-channel attacks: adversaries obtain partial information of ephemeral or permanent secret keys involved in the computation operations of cryptographic protocols or schemes [3–5].

Traditional AKA protocols do not provide user privacy protection, such as user anonymity and user untraceability. With the rapid development of privacy preservation, a three-factor multi-server AAKA protocol with ***user privacy protection*** is urgently required. In the following, we survey present three-factor multi-server AAKA protocols. In 2014, Chuang and Chen [6] proposed a three-factor multi-server AAKA protocol, which uses biometric authentication as the third factor. However, Lin et al. [7] found that the Chuang–Chen protocol [6] is defenseless against servers impersonating attackers (A2) and fails to provide user anonymity (A5), for which they proposed an improvement in 2015. Two three-factor multi-server AAKA protocols [8,9] were proposed in the same year. Later on, Odelu et al. [10] found that He and Wan's protocol [8] cannot withstand user impersonation attacks (A2) and proposed an improvement that additionally achieves user untraceability. The security defects of Hsieh and Leu's two-factor AAKA protocol [11] were found by Amin and Biswas [12] in 2015, for which they developed an improvement; however, the security defects of the Amin–Biswas protocol [12] (A2, A3) were found by Chandrakar and Om [13], and they proposed an improvement [13,14].

In 2016, Park and Park [15] identified the security defects of the three-factor multi-server AAKA protocol proposed by Chang et al. [16] and Choi et al. [17] and improved the biometric-based AAKA protocol proposed by Yoon and Kim [18] to achieve user anonymity. Irshad et al. [19] have also developed an AAKA protocol; however, the servers in their protocol have to store all users' public keys. Reddy et al. [20] developed a multi-server AAKA protocol in 2017, which may suffer insider attacks (A2) and fails to provide user untraceability (A5), as demonstrated by Xu et al. [21]. Qi et al. [22] developed a three-factor AAKA for multi-server telecare medical information systems (TMISs) using ECC in 2018, and Ali and Pal [23] developed a three-factor multi-server AAKA protocol in the same year. Chuang et al. [24] exposed the security defects of four present AAKA protocols: they found that the Ali–Pal protocol [23] and the Chandrakar–Om protocols [13,14] cannot withstand insider attacks, while Choi et al.'s protocol [17] fails to provide user anonymity. Table 1 summarizes the security properties of relevant three-factor multi-server AAKA protocols. Some authenticated key exchange protocols that are resilient to continuous key leakage were proposed to withstand the side-channel attacks [25,26].

1.1. Contribution

Many AAKA protocols have been found to have security defects, despite a lot of them having so-called "formal" security proofs. No research article expounds the process of designing a secure AAKA protocol and provides instructions on how to prove the security of an AAKA protocol. In this paper, we analyze present AAKA protocols to identify the common design errors and the common proof errors. We expound the reasons why a present AAKA protocol with a security proof would still suffer several attacks, and we explain how to make sure that a protocol is secure based on some mathematical assumptions. We instruct readers on how to design a secure AAKA protocol and how to prove the security. This paper will therefore be helpful for the design of new AAKA protocols and for ensuring their security.

Table 1. Relevant three-factor-based AAKA protocols for multi-server environments.

Protocol	Security Defect
Chuang–Chen protocol [6]	(A2, A5) [7]
Lin et al. [7]	-
He–Wang [8]	(A2) [10]
Jiang et al. [9]	-
Odelu et al. [10]	-
Amin–Biswas [12]	(A2, A3) [13]
Chandrakar–Om [13]	(A2) [24]
Chandrakar–Om [14]	(A2) [24].
Park–Park [15]	(A5) [24]
Choi et al. [17]	-
Irshad et al. [19]	-
Reddy et al. [20]	(A2, A5) [21]
Qi et al. [22]	-
Xu et al. [21]	-
Ali-Pal [23]	(A2) [24]

1.2. Organization

The threat models are presented in Section 2. We discuss the occurrence of security defects in Section 3. We instruct readers on how to design a secure AAKA in Section 4, and how to prove the security of an AAKA protocol in Section 5. The conclusion is drawn in Section 6.

2. Threat Model

We assume that an adversary might have the following attack capabilities in an AKA protocol:

- (AC1) They have the ability to replay, eavesdrop, modify, or delete the transmission over an insecure channel [27].
- (AC2) They may be an outsider or any one of the legitimate members [27].
- (AC3) They might try all of the (identity *ID*, password *PW*) pairs offline within probabilistic polynomial time [28].
- (AC4) They might obtain a user's smart card and extract the sensitive information from it [29,30].
- (AC5) They have the ability to fake biometric authentication [31].
- (AC6) They have the ability to individually fake biometric authentication, obtain the (identity, password) pair, and obtain the sensitive information in a smart card. However, the adversary cannot break them all within probabilistic polynomial time.

3. Discussion of the Occurrence of Security Defects

Here, we discuss and analyze the relevant three-factor multi-server AAKA protocols [7–10,12–15,17,19–23], which are surveyed in Section 1. The comparisons of the unsecure ones [8,12–14,17,20,23] and the secure ones [7,9,10,15,19,21,22] are listed in Tables 2 and 3, respectively. We explore the reasons why an AAKA protocol with a formal security proof has security defects, the common design defects of present AAKA protocols, the instructions for designing a secure AAKA protocol, and the instructions to prove the security of an AAKA protocol.

Table 2. Comparisons of the unsecure three-factor multi-server AAKA protocols.

Protocol	Type of Proof	Security Defect
He–Wang [8]	BAN	(A2) [10]
Amin–Biswas [12]	BAN + random oracle	(A2, A3) [13]
Chandrakar–Om [13]	BAN + random oracle	(A2) [24]
Chandrakar–Om [14]	BAN	(A2) [24]
Choi et al. [17]	N/A	(A5) [24]
Reddy et al. [20]	BAN + AVISPA	(A2) [21]
Ali–Pal [23]	BAN + random oracle	(A2) [24]

Table 3. Comparisons of the secure three-factor multi-server AAKA protocols.

	Lin [7]	**Jiang et al. [9]**	**Odelu et al. [10]**	**Park–Park [15]**	**Irshad et al. [19]**	**Xu et al. [21]**	**Qi et al. [22]**
User Anonymity	Y	Y	Y	Y	Y	Y	Y
User Untraceability	N	N	Y	Y	Y	Y	Y
Public key announcement free	Y	Y	Y	Y	N [2,3]	N [3]	N [3]
Table free	N [1,2]	Y	N [1]	N [1]	Y	N [1,2]	Y
Independent AKA	Y	N	N	N	Y	Y	N

[1] registration center, [2] users, [3] servers.

3.1. *Why Does an AAKA Protocol with a Formal Proof Still Have Security Defects?*

All of the relevant unsecure protocols [8,12–14,17,20,23] do not have the "proper" formal proofs. Many present AAKA protocols are not secure on account of the lack of formal proofs (R1). We have summarized the types of proofs of relevant unsecure protocols in a multi-server environment in Table 2. Choi et al. [17] did not provide the formal proof of their protocol. He and Wang [8] and Amin and Biswas [12] used Burrows–Abadi–Needham logic (BAN logic) [32] to prove the security of their protocol. Reddy et al. [20] used BAN logic [32] and an automated validation of internet security protocols and applications (AVISPA) [33] simulation tool to prove the security of their protocols. Chandrakar and Om [13,14] and Ali–Pal [23] claimed that they gave formal security analysis using a random oracle, and that the security of their protocols is based on BAN logic.

Indeed, a random oracle is not a security model, and the security of their protocol is not based on existing difficult problems. A random oracle is just an oracle that responds to every query with a random response [34], which is uniformly chosen from its output domain, and which gives the same response for the same query (R2). The results shown in Table 2 indicate that even though BAN logic [32] ensures the correctness of a protocol, it may fail to ensure the security of a protocol when the security assumptions are no longer present in the applied circumstance (R3). Even though an AVISPA simulation tool [33] ensures that a protocol prevents outsider attacks, it cannot ensure that a protocol prevents insider attacks (R4).

3.2. *Common Design Defects*

In this subsection, we identify the common design errors of present unsecure AAKA protocols to show the process of designing a secure AAKA protocol.

3.2.1. Failure to Withstand Malignant Server Attacks in a Multi-Server Environment

In the Ali–Pal protocol [23] and Chandrakar–Om protocol 2 [14], the server can compute the user's secret keys after the user login; hence, their protocols cannot withstand insider (malignant server) attack in a multi-server environment. Therefore, in order to avoid insider (malignant server) attacks in multi-server environments, the servers have to be regard as independent entities that have distinct secret keys and that cannot obtain any user's secret key (R5).

3.2.2. Failure to Withstand Malignant User Attacks

In Chandrakar–Om protocol 1 [13], a registration center verifies the validity of users by their A_i, and these A_i are all identical; hence, their protocol is vulnerable to insider (malignant user) attacks because of the poor design of the shared secrets. Once the sensitive secret has been leaked to someone who should not know the secret, then attacks may occur. Therefore, the validity of the members cannot be confirmed by the duplicate value, and a member cannot obtain any other member's secret key (R6).

3.2.3. Failure to Provide User Anonymity

In Choi et al.'s protocol [17], each user uses the duplicate values $h(x \mid \mid y)$ to mask their real identity; therefore, their protocol does not achieve user anonymity. Thus, we have the result that the identities of users cannot be masked by a duplicate value (R7).

3.2.4. Failure to Provide User Untraceability

Although the protocols proposed by Lin [7] and Jiang et al. [9] provide user anonymity, they fail to provide user untraceability, since a duplicate value is included in the transmitted messages in different sessions, which means that an eavesdropper could use this duplicate value to trace the user. Therefore, no duplicate value can be included in the transmitted messages in different sessions to achieve user untraceability (R8).

3.3. How to Simultaneously Achieve Public Key Announcement-Free, Table-Free, and Independent Authentication

As shown in Table 3, none of AAKA protocol simultaneously achieve public key announcement-free, table-free, and independent AKA; there is a trade-off between these three properties (R9). In the AKA phase of Jiang et al.'s [9], Odelu et al.'s [10], Park–Park's [15], and Qi et al.'s [22] protocols, users and servers need the help of a registration center; that is, these are dependent AKA, and this might cause a traffic bottleneck, while the registration center has additional burden. Users have to maintain key tables in [7,21], and a registration center has to manage verification tables for users in [10,15]; these protocols [7,10,15,21] are not table free. Users encrypt their identities by using the public key of a server in [19,21,22]; however, how to announce the public keys of servers to users and ensure their correctness is a problem. Either the public keys of servers are pre-sharing with users, or users are making online inquiries to a registration center in the login and AKA phase; thus, the protocol is neither table free nor public key announcement free. The solution is identity-based encryption, which uses the identities as the public keys. The user encrypts their real identity by using the server's public key, and the server's public key is exactly the same as the server's identity (R9).

4. How to Design a Secure AAKA Protocol

The steps of designing a three-factor multi-server AAKA protocol are proposed as follows:

Step 1. First, decide the well-known hard problem before designing the protocol. The goal of the design is that if there is an attacker who can successfully attack the protocol, then the administrator can use the ability of the attacker, breaking the security of the protocol to solve the well-known hard problem [35] (R1). The following are several commonly used mathematical hard problems:

Without loss of generality, let G be a multiplicative cyclic group of a large prime order q, and g is a generator of G.

(1) ***Discrete logarithm (DL) problem*** [36]: Given g^a mod q, to find a.

(2) ***Computational Diffie–Hellman (CDH) problem*** [36]: Given g^a mod q and g^b mod q for unknown $a, b \in Z_q^*$, to find g^{ab}.

(3) ***Decisional Diffie–Hellman (DDH) problem*** [35,36]: Given g^a mod q, g^b mod q, and g^c mod q for unknown $a, b, c \in Z_q^*$, to determine whether $g^c = g^{ab}$.

(4) Without loss of generality, let E be an elliptic curve over a finite field F_q, which is a field of integers modulo a large prime number q, and $E(F_q)$ denotes the set of all the points on E. Let G_1 be an additive cyclic subgroup of points on $E(F_q)$, a point P be a generator of G_1, and G_2 be a multiplicative group with the same order q.

(5) ***Elliptic curve discrete logarithm (ECDL) problem*** [37]: Given a point $Q = dP \in G_1$, to determine the integer d.

(6) ***Elliptic curve computational Diffie–Hellman (ECCDH) problem*** [37]: Given P, aP, $bP \in G_1$ for random $a, b \in Z_q^*$, to find $abP \in G_1$.

(7) ***Elliptic curve decision Diffie–Hellman (ECDDH) problem*** [37]: Given $P, aP, bP, cP \in G_1$ for random a, b, $c \in Z_q^*$, to determine whether $cP = abP$. Note that DDH problem in bilinear pairing is easy: it is easy to verify if $\hat{e}\ (aP, bP) = \hat{e}\ (P, cP)$.

(8) ***Bilinear Diffie–Hellman (BDH) problem in*** $[G_1, G_2, \hat{e}]$ [37]: Given P, aP, bP, $cP \in G_1$ for random $a, b, c \in Zq^*$, to find $\hat{e}\ (P, P)^{abc}$.

(9) ***Decisional bilinear Diffie–Hellman (DBDH) problem in*** $[G_1, G_2, \hat{e}]$ [37]: Given P, aP, bP, $cP \in G_1$ and $\hat{e}(P, P)^d \in G_2$ for random a, b, c, $d \in Z_q^*$, to determine whether $\hat{e}\ (P, P)^d = \hat{e}\ (P, P)^{abc}$.

Step 2. One must obey the following rules:

(1) To avoid insider (malignant server) attacks in multi-server environments, the servers are regarded as independent entities that cannot have the same secret keys and cannot obtain any user's secret key (R5).

(2) To avoid insider attacks, including malignant users and malignant servers, the authenticity of members cannot be confirmed by the same value, and a member cannot obtain any other member's secret key (R6).

(3) To achieve user anonymity, do not adopt a duplicated value to mask the duplicated value (R7).

(4) To achieve user untraceability, do not transmit duplicate values in different sessions (R8).

(5) The following are three approaches that a user can take to allow the server to secretly obtain the user's identity to achieve user anonymity in an AAKE protocol (R9):

Approach 1: The user encrypts their real identity by using the server's public key [19,21,22].

Approach 2: The user encrypts their identity by using the pre-shared key between the user and the server [7,10,15,21].

Approach 3: The third party, which is usually a registration center or a helper, needs to participate in the AKA phase [9,10,15].

The most preferable design is adopting Approach 1 and letting the identity be the public key. Then the AAKA protocol would simultaneously achieve public key announcement-free, table-free, and independent AKA.

Step 3. Prove the correctness of the protocol:

(1) Prove that the common session key established by each entity is identical;

(2) Burrows–Abadi–Needham logic (BAN logic) [32] may be adopted to prove the correctness of the protocol.

5. How to Prove the Security of an AAKA Protocol

5.1. Misconceptions about the Proof

(1) A random oracle model is one type of security model that adopts random oracles to simulate hash functions [34]. Note that a single random oracle is only a query-response process, which produces random responses; it is not a security model (R2).

(2) BAN logic [32] can ensure the correctness of a protocol, but it may fail to ensure the security of a protocol when the security assumptions are no longer present in the applied circumstance (R3).

(3) An AVISPA simulation tool [33] cannot ensure security against insider attacks (R4).

5.2. The Method to Prove the Security

(1) The security of an AAKA protocol must be given a formal proof in a (formal) security model. The security of an AAKA protocol must be based on a well-known hard problem and mathematical assumptions, meaning that if there is an attacker who can successfully attack the protocol, then the administrator can use the ability of the attacker breaking the security of the protocol to solve the well-known hard problem [36] (R1).

(2) Some trust methods through machine learning [38] might provide another way to confirm the security of an AAKA protocol.

(3) Prove the security by the following steps:

Step 1 Clearly state the well-known hard problem and mathematical assumptions. That is, the protocol is secure only if the hard problem is unsolvable.

Step 2 Construct a security model, which is also called an adversary model. This includes queries that are asked by the adversary and answered by the challenger (protocol). Queries should model all of the possible adversary behaviors (attack capabilities), such as passive attacks, active attacks, insider attacks, and forward secrecy attacks. A random oracle is usually used as a hash function that the challenger can control the output value through [34].

Step 3 Permeate the well-known hard problem through queries, which are asked by the adversary and answered by the challenger. Prove that if the adversary can break the protocol with a non-negligible advantage ε, then the challenger can solve the hard problem with a non-negligible advantage ε', where ε' is related to ε. This is in contradiction to the mathematical assumption. Since the hard problem cannot be solved with a non-negligible advantage in probabilistic polynomial time (PPT), no adversary can break the protocol with a non-negligible advantage in PPT. Q.E.D.

6. Conclusions

In this paper, we discussed the relevant three-factor multi-server AAKA protocols to identify some design defects and some proof defects. We explicitly instructed readers "how to design" and "how to prove" when designing an AAKA protocol. This can help readers avoid producing an unsecure AAKA protocol.

Author Contributions: Conceptualization, Y.-H.C.; methodology, Y.-H.C.; software, Y.-H.C.; validation, Y.-H.C.; formal analysis, Y.-H.C.; investigation, Y.-H.C.; resources, C.-L.L.; data curation, Y.-H.C.; writing—original draft preparation, Y.-H.C.; writing—review and editing, H.-J.S.; visualization, Y.-H.C.; supervision, C.-L.L.; project administration, C.-L.L.; funding acquisition, H.-J.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Ministry of Science and Technology in Taiwan under grants MOST 110-2221-E-029-020-MY3.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: No data is reported.

Acknowledgments: The authors thank the anonymous referees for their valuable comments and constructive suggestions.

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

References

1. Lamport, L. Password authentication with insecure communication. *Commun. ACM* **1981**, *24*, 770–772. [CrossRef]
2. Hwang, T.; Chen, Y.; Laih, C.J. Non-interactive password authentications without password tables. In Proceedings of the IEEE TENCON'90: 1990 IEEE Region 10 Conference on Computer and Communication Systems. Conference Proceedings, Hong Kong, China, 24–27 September 1990.

3. Boneh, D.; Demillo, R.A.; Lipton, R.J. On the importance of checking cryptographic protocols for faults. In *EUROCRYPT*; Springer: Berlin/Heidelberg, Germany, 1997; pp. 37–51.
4. Brumley, D.; Boneh, D. Remote timing attacks are practical. *Comput. Netw.* **2005**, *48*, 701–716. [CrossRef]
5. Biham, E.; Carmeli, Y.; Shamir, A. Bug attacks. In *Annual International Cryptology Conference*; Springer: Santa Barbara, CA, USA, 2008; pp. 221–240.
6. Chuang, M.C.; Chen, M.C. An anonymous multi-server authenticated key agreement scheme based on trust computing using smart cards and biometrics. *Expert Syst. Appl.* **2014**, *41*, 1411–1418. [CrossRef]
7. Lin, H.; Wen, F.; Du, C. An improved anonymous multi-server authenticated key agreement scheme using smart cards and biometrics. *Wirel. Pers. Commun.* **2015**, *84*, 2351–2362. [CrossRef]
8. He, D.; Wang, D. Robust biometrics-based authentication scheme for multiserver environment. *IEEE Syst. J.* **2015**, *9*, 816–823. [CrossRef]
9. Jiang, P.; Wen, Q.; Li, W.; Jin, Z.; Zhang, H. An anonymous and efficient remote biometrics user authentication scheme in a multi server environment. *Front. Comput. Sci.* **2015**, *9*, 142–156. [CrossRef]
10. Odelu, V.; Das, A.K.; Goswami, A. A secure biometrics-based multi-server authentication protocol using smart cards. *IEEE Trans. Inf. Forensics Secur.* **2015**, *10*, 1953–1966. [CrossRef]
11. Hsieh, W.B.; Leu, J.S. An anonymous mobile user authentication protocol using self-certified public keys based on multi-server architectures. *J. Supercomput.* **2014**, *70*, 133–148. [CrossRef]
12. Amin, R.; Biswas, G. Design and analysis of bilinear pairing based mutual authentication and key agreement protocol usable in multi-server environment. *Wirel Pers. Commun.* **2015**, *84*, 439–462. [CrossRef]
13. Chandrakar, P.; Om, H. Cryptanalysis and improvement of a biometric-based remote user authentication protocol usable in a multiserver environment. *Trans. Emerg. Tel. Tech.* **2017**, *28*, e3200. [CrossRef]
14. Chandrakar, P.; Om, H. A secure and robust anonymous three-factor remote user authentication scheme for multi-server environment using ECC. *Comput. Commun.* **2017**, *110*, 26–34. [CrossRef]
15. Park, Y.; Park, Y. Three-factor user authentication and key agreement using elliptic curve cryptosystem in wireless sensor networks. *Sensors* **2016**, *16*, 2123. [CrossRef] [PubMed]
16. Chang, I.P.; Lee, T.F.; Lin, T.H.; Liu, C.M. Enhanced two-factor authentication and key agreement using dynamic identities in wireless sensor networks. *Sensors* **2015**, *15*, 29841–29854. [CrossRef] [PubMed]
17. Choi, Y.; Nam, J.; Lee, D.; Kim, J.; Jung, J.; Won, D. Security improvement on biometric based authentication scheme for wireless sensor networks using fuzzy extraction. *Int. J. Distrib. Sens. Netw.* **2016**, *2016*, 1–16.
18. Yoon, E.J.; Kim, C. Advanced biometric-based user authentication scheme for wireless sensor networks. *Sens. Lett.* **2013**, *11*, 1836–1843. [CrossRef]
19. Irshad, A.; Sher, M.; Chaudhary, S.A.; Naqvi, H.; Farash, M.S. An efficient and anonymous multi-server authenticated key agreement based on chaotic map without engaging Registration Centre. *J. Supercomput* **2016**, *72*, 1623–1644. [CrossRef]
20. Reddy, A.G.; Yoon, E.J.; Das, A.K.; Odelu, V.; Yoo, K.Y. Design of mutually authenticated key agreement protocol resistant to impersonation attacks for multi-server environment. *IEEE Access* **2017**, *5*, 3622–3639. [CrossRef]
21. Xu, D.; Chen, J.; Liu, Q. Provably secure anonymous three-factor authentication scheme for multi-server environments. *J. Ambient Intell. Humaniz. Comput.* **2019**, *10*, 611–627. [CrossRef]
22. Qi, M.; Chen, J.; Chen, Y. A secure biometrics-based authentication key exchange protocol for multi-server TMIS using ECC. *Comput. Methods Programs Biomed.* **2018**, *164*, 101–109. [CrossRef]
23. Ali, R.; Pal, A.K. An efficient three factor–based authentication scheme in multiserver environment using ECC. *Commun. Syst.* **2018**, *31*, e3483. [CrossRef]
24. Chuang, Y.H.; Lei, C.L.; Shiu, H.J. Cryptanalysis of four biometric based authentication schemes with privacy-preserving for multi-server environment and design guidelines. In Proceedings of the 15th Asia Joint Conference on Information Security, Taipei, Taiwan, 20–21 August 2020; pp. 66–73.
25. Wu, J.D.; Tseng, Y.M.; Huang, S.S. An identity-based authenticated key exchange protocol resilient to continuous key leakage. *IEEE Syst. J.* **2019**, *13*, 3968–3979. [CrossRef]
26. Hsieh, T.C.; Tseng, Y.M.; Huang, S.S. A leakage-resilient certificateless authenticated key exchange protocol withstanding side-channel attacks. *IEEE Access* **2020**, *8*, 121795–121810. [CrossRef]
27. Tiburski, R.T.; Amaral, L.A.; Hessel, F. Security challenges in 5G-Based IoT middleware systems. In *Internet of Things (IoT) in 5G Mobile Technologies, Modeling and Optimization in Science and Technologies*; Mavromoustakis, C., Mastorakis, G., Batalla, J., Eds.; Springer: Cham, Switzerland, April 2016; Volume 8, pp. 399–418.
28. Wang, D.; Wang, P. Two birds with one stone: Two-factor authentication with security beyond conventional bound. *IEEE Trans Dependable Secur. Comput.* **2018**, *15*, 708–722. [CrossRef]
29. Kocher, P.; Jaff, J.; Jun, B. Differential power analysis. In Proceedings of the Annual International Cryptology Conference; Springer: Santa Barbara, CA, USA, 1999; pp. 388–397.
30. Messerges, T.S.; Dabbish, E.A.; Sloan, R.H. Examining smart-card security under the threat of power analysis attacks. *IEEE Trans. Comput.* **2002**, *51*, 541–552. [CrossRef]
31. Rane, S.; Wang, Y.; Draper, S.C.; Ishwar, P. Secure biometrics: Concepts, authentication architectures, and challenges. *IEEE Signal Process Mag.* **2013**, *30*, 51–64. [CrossRef]

32. Burrows, M.; Abadi, M.; Needham, R. A logic of authentication. *ACM Trans. Comput. Syst.* **1990**, *8*, 18–36. [CrossRef]
33. Vigano, L. Automated security protocol analysis with the AVISPA tool. *Electron. Notes Theor. Comput. Sci.* **2006**, *155*, 61–86. [CrossRef]
34. Bellare, M.; Rogaway, P. Random oracles are practical: A paradigm for designing efficient protocols. In Proceedings of the ACM conference on Computers and Communication Security, Fairfax, VI, USA, 3–5 November 1993; pp. 62–73.
35. Boneh, D. The decision Diffie-Hellman problem. In *Third Algorithmic Number Theory Symposium, Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 1998; Volume 1423, pp. 48–63.
36. Bellare, M.; Rogaway, P. *Introduction to Modern Cryptography*; University of California at Davis: Davis, CA, USA, 2005.
37. Boneh, D.; Franklin, M. Identity-based encryption from the Weil pairing. In Proceedings of the 21st Annual International Cryptology Conference (Crypto 2001), Santa Barbara, CA, USA, 19–23 August 2001; Springer: Santa Barbara, CA, USA, 2001; Volume 2139, pp. 213–229.
38. He, Y.; Han, G.; Jiang, J.; Wang, H.; Martinez-Garcia, M. A trust update mechanism based on reinforcement learning in underwater acoustic sensor networks. *IEEE Trans. Mob. Comput.* **2020**. [CrossRef]