Hindawi
Security and Communication Networks
Volume 2020, Article ID 8848389, 15 pages
https://doi.org/10.1155/2020/8848389

WILEY | Hindawi

Research Article

A Novel Machine Learning-Based Approach for Security Analysis of Authentication and Key Agreement Protocols

Behnam Zahednejad, Lishan Ke, and Jing Li

Institute of Artificial Intelligence and Blockchain, Guangzhou University, Guangzhou, China

Correspondence should be addressed to Lishan Ke; kelishan@gzhu.edu.cn

Received 2 May 2020; Revised 29 July 2020; Accepted 25 September 2020; Published 16 October 2020

Academic Editor: Xiaolong Xu

The application of machine learning in the security analysis of authentication and key agreement protocol was first launched by Ma et al. in 2018. Although they received remarkable results with an accuracy of 72% for the first time, their analysis is limited to replay attack and key confirmation attack. In addition, their suggested framework is based on a multiclassification problem in which every protocol or dataset instance is either secure or prone to a security attack such as replay attack, key confirmation, or other attacks. In this paper, we show that multiclassification is not an appropriate framework for such analysis, since authentication protocols may suffer different attacks simultaneously. Furthermore, we consider more security properties and attacks to analyze protocols against. These properties include strong authentication and Unknown Key Share (UKS) attack, key freshness, key authentication, and password guessing attack. In addition, we propose a much more efficient dataset construction model using a tenth number of features, which improves the solving speed to a large extent. The results indicate that our proposed model outperforms the previous models by at least 10–20 percent in all of the machine learning solving algorithms such that upper-bound performance reaches an accuracy of over 80% in the analysis of all security properties and attacks. Despite the previous models, the classification accuracy of our proposed dataset construction model rises in a rational manner along with the increase of the dataset size.

1. Introduction

Security protocols (cryptographic protocols) are widely used to transport application-level data in a secure manner. These protocols usually apply a sequence of cryptographic primitives such as (a)symmetric encryption, digital signature, and hash function. The most important goals of security protocols include key agreement or establishment, entity authentication, message authentication, and nonreputation [1]. For instance, Transport Layer Security (TLS) [2] is a well-known cryptographic protocol that is used to provide secure web connections (HTTPS). To prove the correctness of security protocols, various methods were developed over the last decades. These methods can be divided into two main categories.

Model-checking methods refer to the set of automated tools and methods that try to find attacks which violate security goals, rather than proving their correctness. ProVerif [3], Scyther [4], AVISPA [5], CryptoVerif [6], and so on are among the most well-known tools. *Theorem-proving* methods are less automated methods that consider all possible protocol behavior to check whether the security goal is achieved or not. Although they cannot give a security attack, they provide a proof of the correctness of the protocol. BAN logic [7], Dolev-Yao model [8], and strand space [9] are examples of these methods.

1.1. Motivation and Goal of This Paper. The goal of this paper is to develop a novel machine learning-based protocol analysis scheme with much better efficiency that can discover more security attacks and vulnerabilities. Previously, the application of machine learning in security analysis has been mainly limited to side-channel attack [10, 11] and symmetric cryptoanalysis [12, 13]. Our motivation for

applying machine learning in protocol analysis is described as follows:

(1) The most important limitation of classical methods is the fact that, to a large extent, analysis results rely on the prior knowledge and experience of the analysts. It frequently happens that even if a security protocol is found to be correct by a model-checking or theorem-proving method, another more experienced researcher discovers a new attack against the same protocol. For example, Tingyuan et al. [14] proved the security of the Otway-Rees protocol. Later, Liu et al. [15] also used BAN logic to point out that this protocol is vulnerable to man-in-the-middle attack and typing flaw attack. Therefore, researchers are trying to discover other methods to guarantee security in cyberspace.

(2) Inspired by the astonishing results of the application of machine learning in cybersecurity [16, 17], Ma et al. [18] designed a machine learning-based model to master the machine in the security analysis of protocols. They suggested a multiclassification model in which every protocol is either secure or prone to replay attack, lacks key confirmation, or is prone to other attacks. Although they received remarkable results for the first time, their analysis is limited to only replay attack and key confirmation. In addition, it frequently happens that a protocol is prone to two or three attacks at the same time (e.g., replay attack and lack of key confirmation). Therefore, multiclassification is not an appropriate model for this purpose. Further, their dataset size is so small, i.e., less than 100 instances for each category.

1.2. Contributions and Structure of This Paper. This paper has three main contributions:

(1) We use a machine learning framework to analyze more security properties such as strong entity authentication and Unknown Key Share (UKS) attack, key freshness, key authentication, and resistance to password guessing attack.

(2) To analyze every research problem in machine learning, the features of the problem should be first extracted. Ma et al. [18] suggested three models, namely, LCM, TLM, and SLM, to extract the features of every protocol as a weighted matrix. We propose a new model with much less number of features which improves the convergence speed.

(3) We propose a binary classification model for each category in which each instance of the dataset either violates one security property or is secure against that. Further, we develop more than 1000 datasets for each category, which is 10 times more than the previous work [18]. Inspired by Ma et al.'s scheme, we also use XGBoost [19] to estimate the classification accuracy of the analysis. In addition, a dense neural network (DNN) was deployed to integrate the deep learning approach to the protocol analysis problem.

The rest of the paper is organized as follows. In Section 2, we briefly introduce authentication and key agreement protocol along with their security goals and attacks. Section 3 discusses the application of machine learning in the security analysis of protocols. In this section, we propose our model to analyze more security properties, that is, strong entity authentication and Unknown Key Share (UKS) attack, key freshness, and so on. The experimental results of the analysis are described in Section 4. Finally, a conclusion is given in Section 5.

2. Authentication and Key Establishment Protocols

Authentication and key establishment protocols are the backbone of any secure electronic communication. Cryptographic algorithms such as AES and DES [20, 21] cannot be implemented unless common secret keys are preshared (key establishment) and communication parties know who owns such keys (authentication). Authentication and key establishment protocols achieve these goals by using a set of messages consisting of random numbers, identities, timestamps, hash function, and so on. For example, consider the ISO symmetric key two-pass unilateral authentication and key establishment protocol [22] between two parties like Alice and Bob in Figure 1.

Here, Alice sends a random number N_A to Bob. The secret key K_{AB} is preshared between Alice and Bob. When Bob sends message $E_{K_{AB}}(N_A)$ to Alice, she makes sure that it was sent by Bob because he only has the key K_{AB}. In other words, she authenticates Bob. Authentication protocols are widely used in different applications such as wireless networks [23], smart city [24], and Internet of Things (IoT) [25]. Authentication and key establishment are the two main goals of cryptographic protocols [1]. In the following, we describe more detailed security goals based on authentication and key establishments. Then we present the most common attacks that try to violate these goals. Figure 2 shows the set of security attacks and goals of authentication protocols.

2.1. Authentication Goals. According to ISO security architecture [26], authentication is defined as the "assurance that an entity is the one who claims to be." More precisely, two kinds of authentications can be distinguished as follows.

2.1.1. Entity Authentication. Entity authentication is the process whereby one party is assured of the identity of the second party in the protocol and that the second party has actually participated [27]. This definition assures one party (e.g., A) that the other party (B) has participated in the protocol. It does not provide assurance for A that B also recognized A as his/her peer entity. For example, suppose the protocol of Figure 3(a).

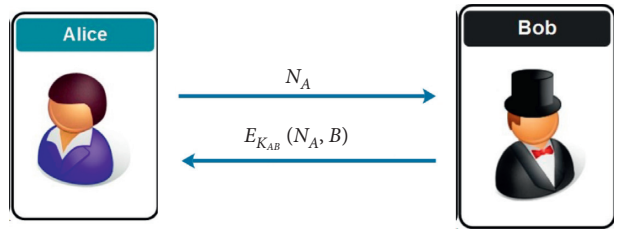

Figure 1: ISO symmetric authentication protocol.

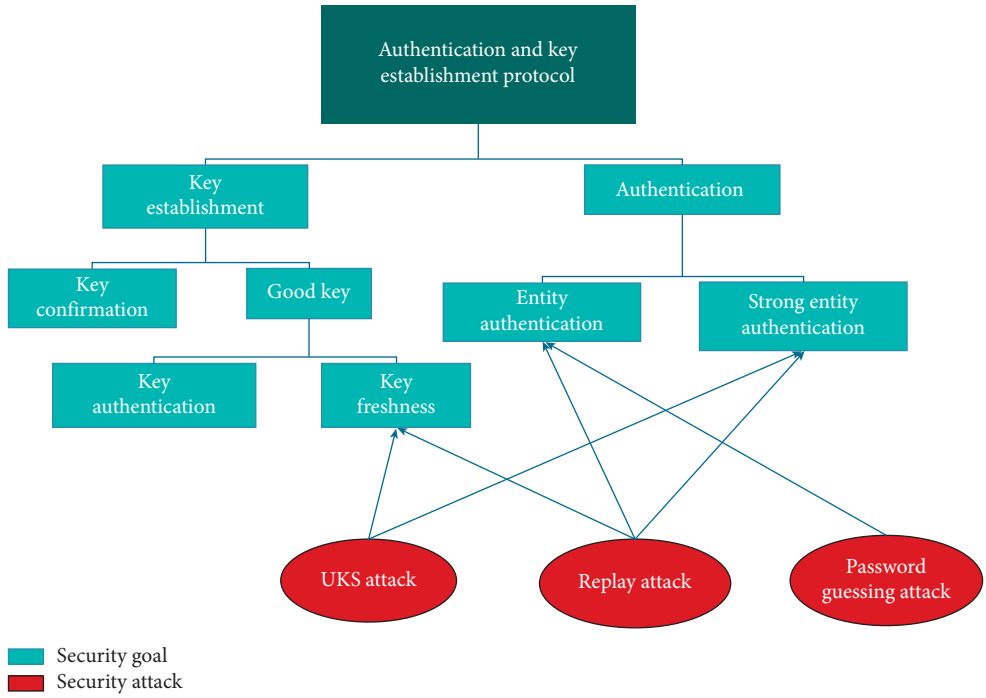

Figure 2: Security goals and attacks in cryptography protocols.

$1.A \longrightarrow B : N_A$
$2.B \longrightarrow A : Sig_B(N_A)$

(a)

$1.A \longrightarrow C_B : N_A$
$1'.C \longrightarrow B : N_A$
$2'.B \longrightarrow C : Sig_B(N_A)$
$2.C_B \longrightarrow A : Sig_B(N_A)$

(b)

$1.A \longrightarrow B : N_A$
$2.B \longrightarrow A : Sig_B(A, N_A)$

(c)

Figure 3: Strong authentication in security protocols. (a) A security protocol without strong authentication, (b) an attack against strong authentication and (c) improved protocol with strong authentication.

In this protocol, since B signs the nonce N_A, the entity A is assured that B has participated in the same protocol running as entity A. However, the entity B may suppose another entity like C, as his/her peer identity. Figure 3(b) shows an attack against this protocol. In this attack, the adversary C masquerades himself as entity B to A. At the same time, he begins a parallel session with entity B and forwards the response of B to A. As a result of this attack, entity A believes that he/she is contacting with entity B, while B assumes C as his peer entity.

2.1.2. Strong Entity Authentication. Strong entity authentication of A to B is provided if B has a fresh assurance that A has knowledge of B as his/her peer entity [1]. Based on this property, the adversary C has no way to convince B that he/she is in contact with C. Figure 3(c) shows an enhanced version of the protocol of Figure 3(a). In this protocol, the entity B signs his peer identity (ID_A) to make A sure that he recognizes A as his peer entity. As another example, consider protocol SPLICE/AS in Figure 4(a), designed by Yamaguchi et al. [28] to provide mutual authentication between client A and server B. However, Clark and Jacob [29] reported that this protocol cannot provide strong mutual authentication. As shown in Figure 4(b), the attacker C can replace the signature of A with C's signature. As a result, the entity A believes that the protocol has been held with entity B, while B assumes C as his peer entity. To prevent this attack, Clark and Jacob proposed to include the encrypted

$1.A \longrightarrow B : A, B, T_A, E_B(N_A), Sig_A(A, T_A, E_B(N_A))$
$2.B \longrightarrow A : B, A, E_A(B, N_A + 1)$

(a)

$1.A \longrightarrow C_B : A, B, T_A, E_B(N_A), Sig_A(A, T_A, E_B(N_A))$
$1'.C \longrightarrow B : C, B, T_A, E_B(N_A), Sig_C(C, T_A, E_B(N_A))$
$2'.B \longrightarrow C : B, C, E_C(B, N_A + 1)$
$2.C_B \longrightarrow A : B, A, E_A(B, N_A + 1)$

(b)

$1.A \longrightarrow B : A, B, T_A, E_B(ID_A, N_A), Sig_A(A, T_A, E_B(ID_A, N_A))$
$2.B \longrightarrow A : B, A, E_A(B, N_A + 1)$

(c)

Figure 4: Strong authentication in SPLICE/AS protocol. (a) SPLICE/AS protocol. (b) Clark and Jacob's attack against SPLICE/AS protocol. (c) Clark and Jacob's improved protocol.

identity of initiator (ID_A) in the message sent to responder (B). A modified version of this protocol is shown in Figure 4(c).

2.2. Key Establishment. Key establishment is the process whereby a shared secret (session key) becomes available to two or more parties, for subsequent cryptographic use [30]. In this regard, the following goals are assumed for cryptographic protocols.

2.2.1. Good Key. Usually, a session key is only useful if it is known to be fresh and shared to only authenticated and trusted parties. We call it a good key, if it achieves both requirements. More formally, the shared session key is a good key for A to use with B only if A is sure that the following requirements are both satisfied [1]:

(1) Key Freshness. Key freshness is achieved when the communicating parties are able to verify and make sure that the session key they agree with each other is fresh (new) and not replayed from an old session. This is usually achieved by a freshness value. There are two main freshness values used in cryptography protocol: timestamps and nonce [31].

(2) Timestamps. In this method, the current time of the sender is added to the key. As the receiver obtains the message, if there is an acceptable delay, the key is accepted. Otherwise, it aborts. The difficulty of using this method is clock synchronization requirements of sender and receiver. For example, consider the Denning and Sacco protocol [32] depicted in Figure 5. In this protocol, if the timestamp T_S is in a reasonable delay, the parties A and B make sure of the freshness of key K_{AB}.

(3) Nonce. In this method, before the sender sends the key, the recipient, for example, A, generates a nonce, N_A, and transfers it to party B. Then, the nonce, N_A, and the session key K_{AB} are both encrypted and sent to recipient, A. For example, consider the improved MSR protocol of Figure 6. In this protocol, the party B transfers K_{AB} to party A. In addition, it encrypts the nonce N_A with the session key K_{AB}. As the party A decrypts the message with the session key K_{AB} and obtains N_A, it makes sure of the freshness of the key.

(4) Key Authentication. Key authentication is defined as follows: the key should only be known to A and B and any

$1.A \longrightarrow S : ID_A, ID_B$
$2.S \longrightarrow A : E_{K_{AS}}(ID_A, K_{AB}, T_S)$
$3.S \longrightarrow B : E_{K_{BS}}(ID_A, K_{AB}, T_S)$

Figure 5: Using timestamp as freshness value in Denning and Sacco protocol.

mutually trusted parties (Gollmann [33] points out that this property can be regarded as the confidentiality of the key). For example, consider the Otway-Rees protocol of Figure 7(a). In this protocol, the server S distributes the session key K_{AB} to A and B. However, as pointed out by Boyd and Mao [34], the attacker can easily mount the attack of Figure 7(b). As a result of this attack, A believes that the key K_{AB} is shared with B, while it is shared with the adversary C. This is a violation of key authentication, as the adversary has access to session key K_{AB}. Abadi and Needham prevented this attack by proposing the protocol shown in Figure 7(c).

2.2.2. Key Confirmation. Key confirmation of A to B is provided if B has assurance that key K is a good key to communicate with A and that principal A has possession of K [1]. Key confirmation provides evidence for a party that his peer partner has received the same key. However, it does not imply entity authentication, as the key may be assumed to be shared with somebody else. In addition, this property cannot be provided for both parties, as one party should finish the protocol.

2.3. Security Attacks. There are many attacks that try to violate the security goals of cryptographic protocols. The most common attacks are described as follows. For more information about other types of attacks, refer to [1].

2.3.1. Unknown Key Share (UKS) Attack. As defined by Blake-Wilson and Menezes [35], an Unknown Key Share (UKS) attack is an attack whereby an entity A ends up believing that she shares a key with B and although this is in fact the case, B mistakenly believes the key is instead shared with entity E ≠ A. This attack targets strong authentication and key freshness of the protocol. For example, consider the Helsinki Protocol in Figure 8(a). A UKS attack on Helsinki's

$$1.A \longrightarrow B : ID_A, Cert_A, N_A$$
$$2.B \longrightarrow A : E_A(K_{AB}), E_{K_{AB}}(ID_A, Cert_B, N_A)$$

Figure 6: Using nonce as freshness value in MSR protocol.

protocol was published by Horng and Hsu [36]. As shown in Figure 8(b), B ends up believing she shares a session key $f(K_{BA}, K'_{AB})$ with A. However, A assumes C as his peer entity whom he shared the key $f(K_{AB}, K_{BA})$ with. Mitchell and Yeun [37] proposed to improve this protocol by adding B's identity to message 2 (Figure 8(c)).

2.3.2. Replay Attack. Replay attack occurs whenever the adversary interferes with the protocol run by inserting a message which has been captured in previous sessions of the protocol. Usually, this attack is used to mount other types of attacks, as well. A detailed taxonomy of replay attacks is described by Syverson [38].

2.3.3. Password Guessing Attack. Another common attack to compromise the authentication of legitimate users is through offline guessing of users' passwords. Passwords are generally used to encrypt messages or to authenticate one party to another. In this attack, the adversary needs to access some public parameters and messages, which are usually captured by eavesdropping. If the parameters that are coupled with the password are known to the adversary, he/she can guess the password (as they are usually of low entropy) and check the correctness of his/her guess. For example, if the password of user A is transmitted as $h(\text{Password}_A)$, the attacker can easily guess Password_A and check its correctness by taking a hash of it as $h(\text{Password}_A)$ and see if it is equal to the transmitted message ($h(\text{Password}'_A)? = h(\text{Password}_A)$). However, if the Password_A is coupled with unknown and high entropy parameters like random number N_A, the attacker has no way to check the correctness of his guess [39].

3. Application of Machine Learning in Security Analysis of Authentication and Key Agreement Protocols

The idea of using machine learning to analyze authentication and key agreement protocols was first presented by Ma et al. [18], who suggested training the network by designing a classification problem. Similar to any classification problem in machine learning, we need a set of datasets and their corresponding categories (labels) to train the network. Here, every protocol is an instance of the dataset and the attack that the protocol is vulnerable to is its label. After training the network with a set of protocols and the attacks (categories) that they are vulnerable to, we expect the network to analyze unseen and new protocols and find what kind of attack they are prone to. In this regard, we need a model to map every protocol to an instance of the dataset. In the following, we discuss the dataset model and the categories (labels) of the problem.

3.1. Dataset Construction Model. Dataset construction model is a mapping relation between protocol messages and instances of the dataset. Ma et al. suggested two approaches to convert every protocol to an instance of the dataset. Here, every protocol $P = m_1, m_2, \ldots, m_k$ corresponds to a matrix in the dataset and every message m_i of the protocol corresponds to a vector of the matrix. Before the description of Ma et al.'s dataset models, some definitions are given as follows. Firstly, a message parameter set SP and a parameter property set PP are defined for every message of the protocol:

$$\begin{aligned} \text{SP} &= \{sp_1, sp_2, \ldots, sp_n\}, \\ \text{PP} &= \{pp_1, pp_2, \ldots, pp_n\}. \end{aligned} \tag{1}$$

Here, sp_i denotes any message parameter such as timestamp, participant identity, and random number. Also, pp_i denotes message attributes such as index of parameters, encryption key, and signature key. For example, consider the protocol of Figure 1. This protocol consists of messages m_1 and m_2. For each message, the set of message parameters SP and parameter property sets PP are as follows:

$$\begin{aligned} \text{SP} &= \{N_A, ID_A\}, \\ \text{PP} &= \{K_{AB}\}. \end{aligned} \tag{2}$$

In addition, the lengths of SP and PP are assumed to be fixed (N and M, resp.). If the lengths of the SP and PP are less than N and M, zero values are added to the set. As a result, every message is described by an $N * M$ vector. To reduce the dimension of the message vector, a normalization function is defined as follows:

$$f_n(sp_i) = f_n(PP_i) = f_n(pp_1, pp_2, \ldots, pp_m) = \lambda_i. \tag{3}$$

In the following, after reviewing Ma et al.'s dataset model, we describe our proposed dataset model followed by its comparison with Ma et al.'s dataset model.

3.1.1. Review of Ma et al.'s Model. Ma et al. developed three models, namely, TLM (two-layer model), LCM (literal conversion model), and SLM (single-layer model), to convert every protocol message to a message vector. LCM and SLM models are almost the same. Further, their subtle differences are not clearly explained in [18]. In the following, we only describe TLM and SLM models:

(1) Two-Layer Model (TLM). In TLM, an empty message vector $m_i = sp_{i,1}, sp_{i,2}, \ldots, sp_{i,N}$ is predefined. Here, N is the maximum number of message parameters in the whole dataset. Here, $sp_{i,j}$ is a predefined zero vector of size M. Every dimension of this vector corresponds to a specific property such as plaintext index, encryption key index, and signature key index. For every message parameter sp_i, the property parameters are filled according to the actual protocol. As a result, every message is represented as an $N * M$ vector. Figure 9 shows an example of the TLM conversion model of ISO symmetric authentication protocol in Figure 1.

$1.A \longrightarrow B : A, B, \{N_A, A, B\}\ K_{AS}$
$2.B \longrightarrow S : A, B\ \{N_A, A, B\}\ K_{AS}, \{N_B, A, B\}\ K_{BS}$
$3.S \longrightarrow B : \{N_A, K_{AB}\}\ K_{AS}, \{N_B, K_{AB}\}\ K_{BS}$
$4.B \longrightarrow A : \{N_A, K_{AB}\}\ K_{AS}$

(a)

$1.A \longrightarrow C_B : A, B, \{N_A, A, B\}\ K_{AS}$
$2.C_B \longrightarrow S : A, C, \{N_A, A, B\}\ K_{AS}, \{N_C, A, B\}\ K_{BS}$
$3.S \longrightarrow C_B : \{N_A, K_{AB}\}\ K_{AS}, \{N_B, K_{AB}\}\ K_{BS}$
$4.C_B \longrightarrow A : \{N_A, K_{AB}\}\ K_{AS}$

(b)

$1.A \longrightarrow B : A, B, N_A$
$2.B \longrightarrow S : A, B, N_A, N_B$
$3.S \longrightarrow B : \{N_A, A, B, K_{AB}\}\ K_{AS}, \{N_B, A, B, K_{AB}\}\ K_{BS}$
$4.B \longrightarrow A : \{N_A, A, B, K_{AB}\}\ K_{AS}$

(c)

Figure 7: Key authentication in Otway-Rees protocol. (a) Otway-Rees protocol. (b) Boyd and Mao's attack against Otway-Rees protocol. (c) Abadi and Needham's improved protocol.

$1.A \longrightarrow B : E_B\,(A, K_{AB}, N_A)$
$2.B \longrightarrow A : E_B\,(K_{BA}, N_A, N_B)$
$3.A \longrightarrow B : N_B$

(a)

$1.A \longrightarrow C : E_C\,(A, K_{AB}, N_A)$
$1'.C_A \longrightarrow B : E_B\,(A, K_{AB}, N_A)$
$2'.B \longrightarrow C_A : E_B\,(K_{BA}, N_A, N_B)$
$2.C \longrightarrow A : E_B\,(K_{BA}, N_A, N_B)$
$3.A \longrightarrow C : N_B$
$3'.C_A \longrightarrow B : N_B$

(b)

$1.A \longrightarrow B : E_B\,(A, K_{AB}, N_A)$
$2.B \longrightarrow A : E_B\,(B, K_{BA}, N_A, N_B)$
$3.A \longrightarrow B : N_B$

(c)

Figure 8: UKS attack in Helsinki protocol. (a) Helsinki's protocol. (b) UKS attack against Helsinki protocol. (c) Mitchell and Yeun's improved protocol resistant against UKS.

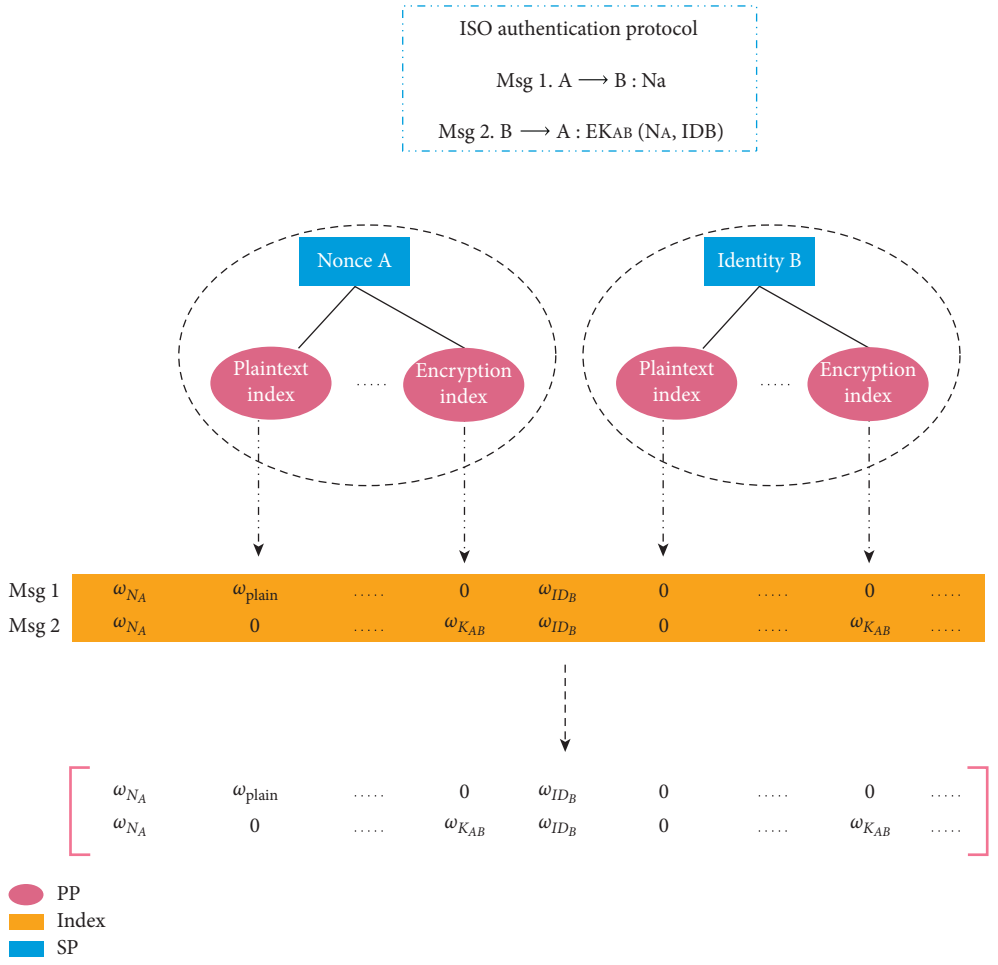

Figure 9: Conversion process in TLM.

(2) Single-Layer Model (SLM). Similar to TLM model, an empty message vector $m_i = \{sp_{i,1}, sp_{i,2}, \ldots, sp_{i,N}\}$ is predefined, where N is the maximum number of message parameters in the whole dataset. However, in this model, the normalized function f_n is applied to every message parameter $sp_{i,j}$. Thus, every message m_i is represented as follows:

$$m_i = f_n(sp_{i,1}), f_n(sp_{i,2}), \ldots, f_n(sp_{i,N}) = (\lambda_1, \lambda_2, \ldots, \lambda_n). \quad (4)$$

As a result of applying the normalization function, the number of data dimensions is reduced from $N * M$ to N. A schematic of this conversion model is shown in Figure 10.

3.1.2. Our Proposed Model. Despite Ma et al.'s models that consider each message component individually, our proposed model is closer to the real representation of protocols. In this model, every parameter property, pp_i, is represented by a corresponding index, that is, encryption index, signature index, and so on. Then, the indices of the set of message parameters, sp_i, that are in plaintext, encrypted, signed, or hashed together, are put after the plaintext index, encryption index, signature index, and hash index, respectively. The main advantage of this method is that parameters are not modeled individually but alongside other adjacent parameters. Therefore, every message vector would be as follows:

$$m_i = \{pp_{i,1}, sp_{i,1}, sp_{i,2}, \ldots, pp_{i,m}, sp_{i,1}, sp_{i,2}, \ldots\}. \quad (5)$$

As a result, the size of the message vector is reduced to $4 * L$, since the only parameter properties we considered are plaintext, encryption, signature, or hash. Here, L is the maximum number of message parameters that are in plaintext, encrypted, signed, or hashed together. A schematic of this model is depicted in Figure 11.

3.1.3. Comparison of Our Proposed Model with Previous Models. Although Ma et al.'s model could receive remarkable results for the first time, it models each message parameter separately, while in cryptographic protocols, each message parameter is bound to other message parameters. For example, consider the protocol message depicted in Figure 12. In this figure, as shown in red lines, all versions of Ma et al.'s model consider each message component separately. As a result, the machine cannot learn the fact that the set of message parameters are hashed together, while our model considers bound messages together. This is an important point in some attacks such as password guessing attacks, where the adversary exploits the fact that the password is bound to some low entropy parameters (Section 2.3.3). In addition, the dimensions of Ma et al.'s model are so high, which reduces the implementation speed of machine learning models. A comparison of dataset dimensions is shown in Table 1.

3.2. Category. Categories are the labels that we assign to datasets to distinguish the attack in which the protocol is vulnerable to. In Ma et al.'s scheme [18], protocols were labeled based on replay attack and key confirmation. In this paper, we develop more datasets and label them according to more security goals and attacks such as Unknown Key Share attack and strong entity authentication, key freshness, password guessing attack, and key authentication. In this section, after reviewing Ma et al.'s categories, we describe their deficiency and propose our categories to label the datasets.

3.2.1. Review of Ma et al.'s Categories. Ma et al. [18] designed a multiclassification problem to analyze authentication and key agreement protocols with machine learning. Ma et al. suggested that every protocol is either secure or prone to one attack limited to replay attack (Section 2.3.2), lack of key confirmation (Section 2.2.2), or other attacks. Accordingly, Ma et al. associated a category number with every protocol ranging from 1 to 4 (Figure 13). Then, they collected around 500 protocols and divided them according to the attack they are prone to.

3.2.2. Deficiency of Ma et al.'s Categories. Although Ma et al. received remarkable results for the first time, the results are only valid for limited number of protocols. Only around 100 protocols were collected for each category. Limited number of datasets reduces the generalization ability of the analysis tool. Furthermore, most of the protocols are vulnerable to multiple attacks. For example, consider the following protocol in Figure 14.

At the same time, it suffers lack of key confirmation, as neither S nor B is not sure if the other party has received the session key K_{AB}. As a result, the multiclassification problem is not an appropriate framework to analyze protocols with. In the next section, we propose a new framework with a larger number of datasets to analyze security protocols with machine learning.

3.2.3. Our Proposed Categories. Considering the deficiencies of Ma et al.'s categories, we provide more datasets for each category. Further, we design a binary classification problem in which the protocols are either prone to a specific attack or secure against that (Figure 15). In this regard, the following attacks/goals are considered for each problem.

(1) Strong Authentication and Unknown Key Share Attack. As explained in Section 2.1.2, strong entity authentication of A to B is provided if B has a fresh assurance that A has knowledge of B as his/her peer entity. The most common attack that targets this property is UKS attack (Section 2.3.1). For the purpose of analyzing security protocols against this property, we develop around 1000 protocols that are either secure or prone to an attack that violates this property. For example, consider an instance of this dataset in Figure 16. Figure 16(a) shows the SPLICE/AS protocol [28] which is labeled as category 1, since it is prone to the attack presented by Clark and Jacob [29] (Figure 4(b)) and does not achieve strong authentication. An improved version of this protocol

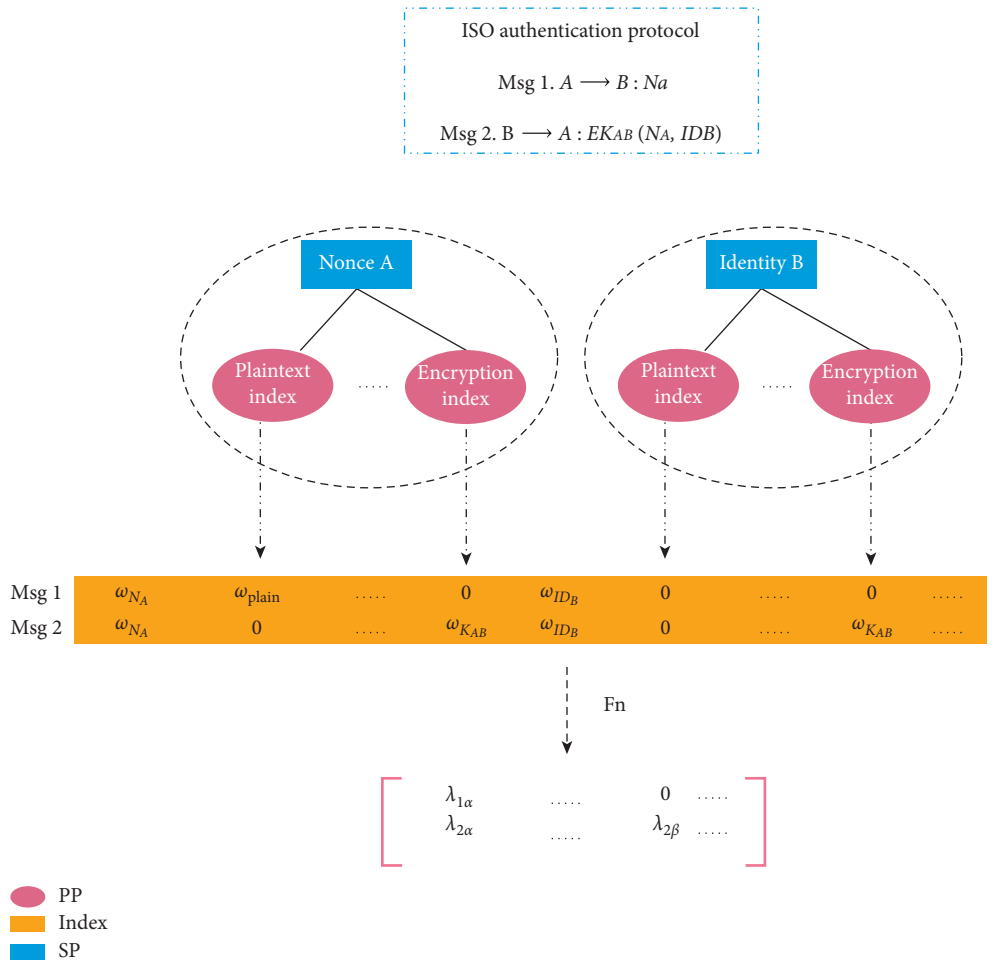

Figure 10: Conversion process in SLM.

is shown in Figure 16(b). As it is secure against this attack, it is labeled as category 0.

As another example, consider the Helsinki protocol [36] in Figure 8(a). As this protocol is vulnerable to UKS attack (Figure 8(b)), it is labeled as category 1 (Figure 17(a)). As suggested by Mitchell et al. [37], a secure version of this protocol is labeled as category 0 (Figure 17(b)).

(2) Key Freshness. As said in Section 2.2.1.1, key freshness is achieved in security protocols if the parties can verify and make sure that the session key they agree with each other is fresh and not replayed from an old session. To analyze security protocols against this property, more than 1500 datasets were developed which are either secure or prone to lack of key freshness. An instance of this dataset is shown in Figure 18. In Figure 18(a), a secure scheme (improved MSR scheme in Figure 6) is shown which is labeled as category 0, while the scheme in Figure 18(b) lacks key freshness, as no freshness value is used to transfer session key K_{AB}.

As another example, consider the key agreement protocol of Figure 19. Here, the session key is a^{xy}. The protocol shown in Figure 19(a) is vulnerable to replay attack which violates the key freshness of the scheme. As the adversary can replay message 2 and convince A to agree on a different session key than party B, thus, it is labeled as category 1. However, the scheme of Figure 19(b) is secure against this attack, as the adversary can no longer replay message 2, since he/she fails to forge the signature of B which includes the freshness parameter a^x.

(3) Key Authentication. According to the definition of Section 2.2.1.1, the key should only be known to A and B and any mutually trusted parties. To analyze security protocols against this property, around 1200 protocols were provided as dataset. Each instance of the dataset is either prone to key authentication or secure against this property. For example, consider Otway-Rees protocol as an instance of dataset in Figure 20(a). As explained in Section 2.2.1.1, this protocol cannot provide key authentication. As a result, it is labeled as category 1. Abadi and Needham's protocol is labeled as category 0 as it achieves key authentication (Figure 20(a)).

(4) Password Guessing Attack. According to the definition of Section 2.3.3, the attacker is able to guess the secret password if it is hashed together with other public parameters. To analyze security protocols against this property, around 1500 protocols were provided as dataset. Each instance of the

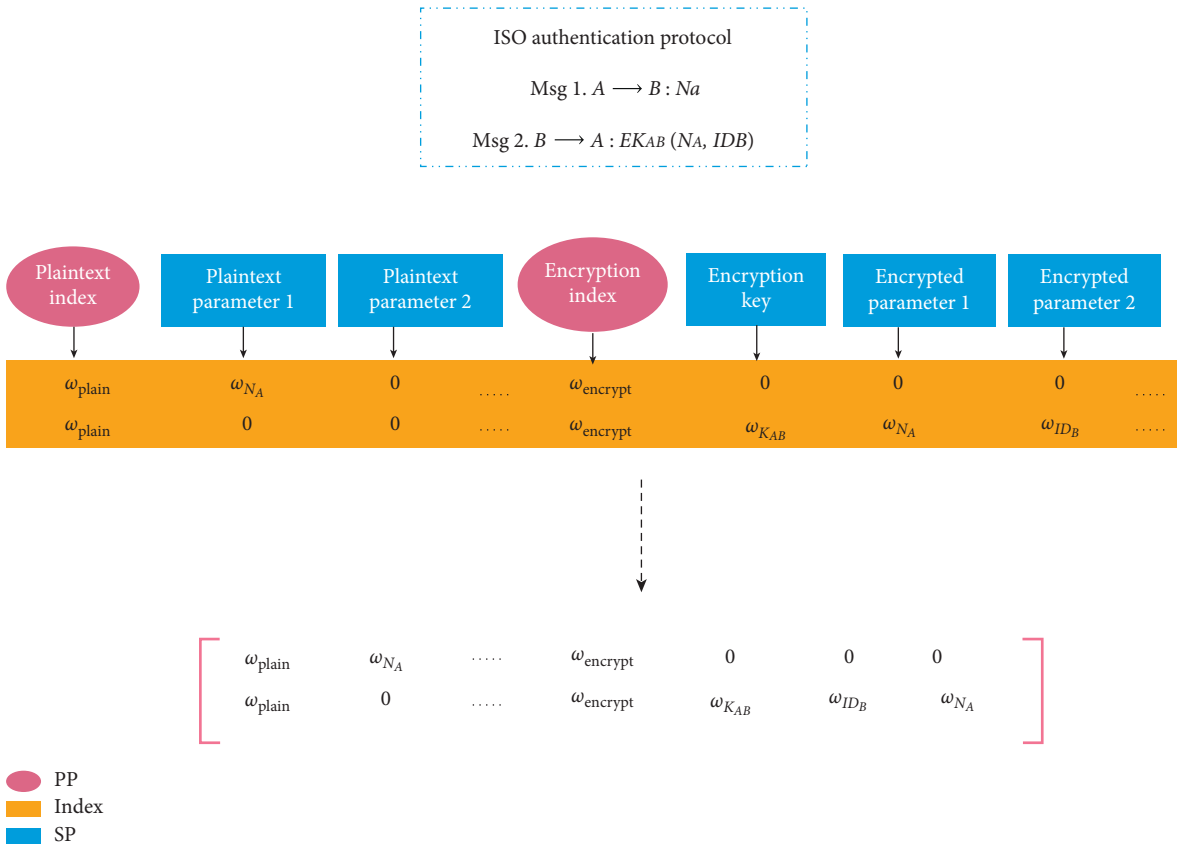

Figure 11: Our proposed model.

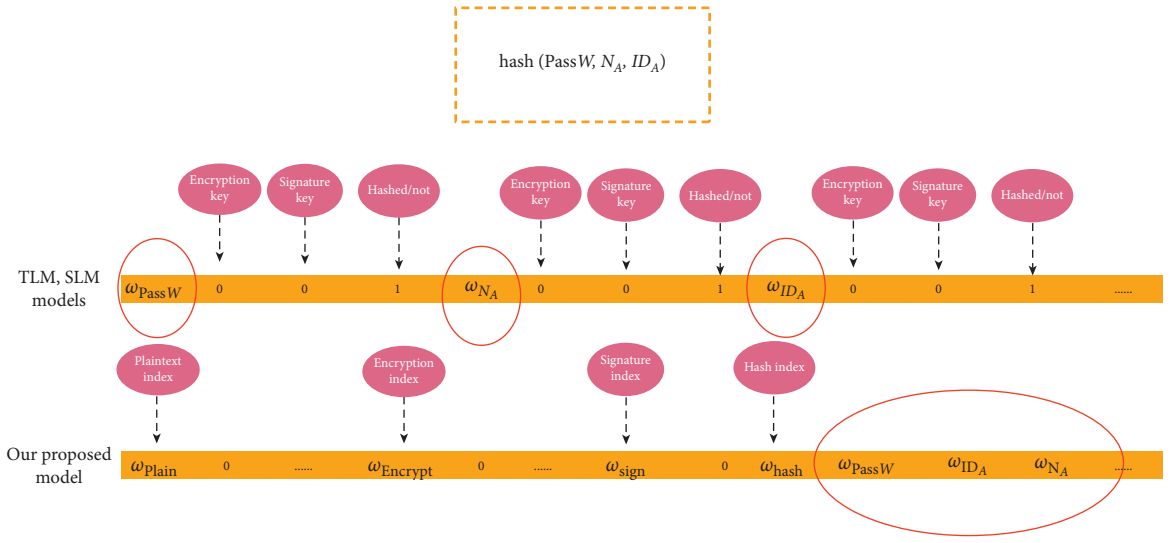

Figure 12: Deficiency of TLM and SLM model in considering message parameters separately.

Table 1: Comparison of dataset dimensions and density.

	TLM	SLM	Our proposed model
Number of features	$N * M$	N	$4 * L$
Data density	High	Low	Low

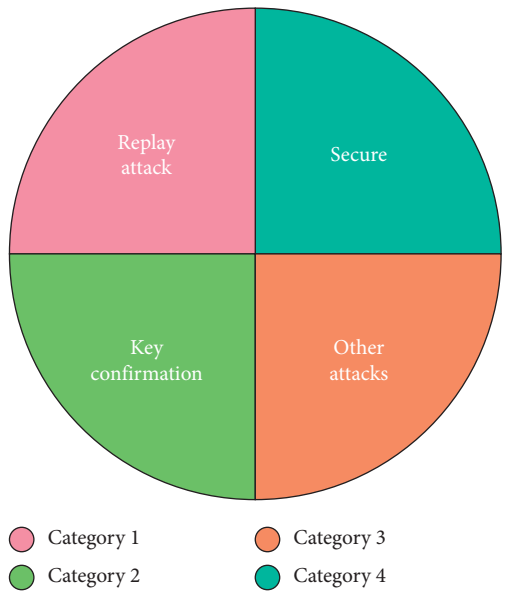

Figure 13: Protocol analysis as a multiclassification problem.

$$1.A \longrightarrow B : N_A, ID_A$$
$$2.B \longrightarrow S : N_A, ID_A, N_B, ID_B$$
$$3.S \longrightarrow B : E_{K_{AS}}(K_{AB}, ID_B), E_{K_{BS}}(K_{AB}, ID_A)$$
$$4.B \longrightarrow A : E_{K_{AS}}(K_{AB}, ID_B)$$

(a)

$$1.A \longrightarrow B : N_A, ID_A$$
$$1'.A \longrightarrow B : N'_A, ID_A$$
$$2.B \longrightarrow S : N_A, ID_A, N_B, ID_B$$
$$2'.B \longrightarrow I_S : N'_A, ID_A, N'_B, ID_B$$
$$3.S \longrightarrow B : E_{K_{AS}}(K_{AB}, ID_B), E_{K_{BS}}(K_{AB}, ID_A)$$
$$3'.I_S \longrightarrow B : E_{K_{AS}}(K_{AB}, ID_B), E_{K_{BS}}(K_{AB}, ID_A)$$
$$4'.B \longrightarrow A : E_{K_{AS}}(K_{AB}, ID_B)$$

(b)

Figure 14: An example protocol that suffers both replay attack and lack of key confirmation. (a) A vulnerable security protocol. (b) Replay attack against the protocol.

dataset is either prone to password guessing attack or secure against this attack. For example, consider the Lee-Sohn-Yang-Won password-based protocol as an instance of dataset in Figure 21(a). This protocol is prone to password guessing attack, as the parameters that are hashed together with the password, that is, A and B, are all public and accessible by the adversary. Accordingly, it is labeled as category 1. However, the protocol depicted in Figure 21(b) is secure against password guessing attack thanks to the secret parameter K_{AB}, since the attacker has no way to guess the password and verify its correctness.

4. Experimental Results

In this section, we apply our proposed model along with previous models, namely, TLM and SLM models, to analyze different security properties of authentication and key

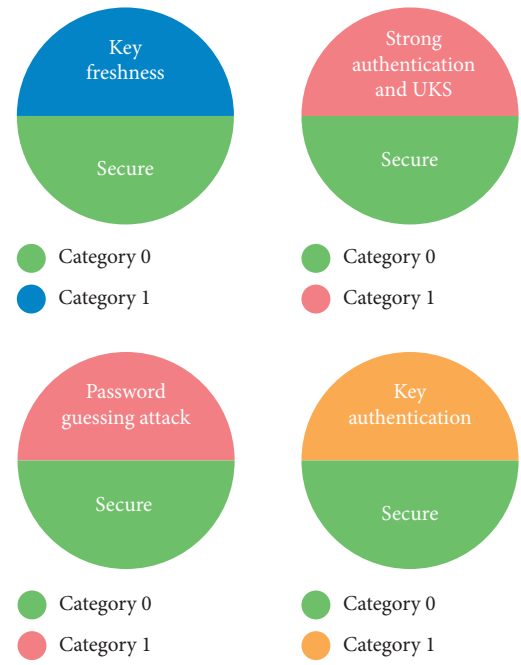

Figure 15: Protocol analysis as a binary classification problem.

agreement protocols such as resistance to Unknown Key Share (UKS) attack, key freshness, key authentication, and resistance to password guessing attack. Then, we compare the performance of our proposed model against previous models, namely, TLM and SLM models. The results indicate that our proposed model outperforms the previous models by at least 10–20 percent in all of the machine learning models. In addition, for more complex security properties and attacks such as UKS attack and key authentication, the increase of the dataset size has almost no effect on the classification accuracy, which indicates that these dataset constructions are unable to train the machine. Inspired by the experimental results of Ma et al., we apply XGBoost approach to our classification problem. To improve the accuracy, we modified the default value of the number of gradient boosted trees and maximum depth value of the trees in XGBoost model such that the best results are received with either (10, 3), (10, 15), (20, 15), or (30, 30) where the first component represents the number of gradient boosted trees and the second component represents the maximum depth value of the trees. In addition, to gauge whether deep learning-based approaches are appropriate in this framework, multilayer perceptron (MLP) model is also employed. The results indicate a promising prospect for the integration of deep learning with protocol analysis. The hidden layer size of the MLP model is set to either (15, 15), (20, 20), or (30, 30). In the following, we discuss the experimental results for each analysis, namely, resistance to Unknown Key Share (UKS) attack, key freshness, key authentication, and resistance to password guessing attack.

4.1. Experimental Results of Analyzing UKS Attack. As shown in Figures 22 and 23, in analysis of UKS attack, the classification accuracy of our proposed model rises with the

$1.A \longrightarrow B : A, B, T_A, E_B(N_A), Sig_A(T_A, E_B(N_A))$
$2.B \longrightarrow A : B, A, E_A(B, N_A + 1)$
Category: 1

(a)

$1.A \longrightarrow B : A, B, T_A, E_B(A, N_A), Sig_A(T_A, E_B(A, N_A))$
$2.B \longrightarrow A : B, A, E_A(B, N_A + 1)$
Category: 0

(b)

Figure 16: An instance of dataset in analysis of strong authentication and UKS. (a) SPLICE/AS protocol denoted by category 1. (b) SPLICE/AS improved protocol denoted by category 0.

$1.A \longrightarrow B : E_B(A, K_{AB}, N_A)$
$2.B \longrightarrow A : E_B(K_{BA}, N_A, N_B)$
$3.A \longrightarrow B : N_B$
Category: 1

(a)

$1.A \longrightarrow B : E_B(A, K_{AB}, N_A)$
$2.B \longrightarrow A : E_B(B, K_{BA}, N_A, N_B)$
$3.A \longrightarrow B : N_B$
Category: 0

(b)

Figure 17: Another instance of dataset in analysis of strong authentication and UKS. (a) Helsinki protocol denoted by category 1. (b) Mitchell et al.'s protocol denoted by category 0.

$1.A \longrightarrow B : ID_A, Cert_A, N_A$
$2.B \longrightarrow A : E_A(K_{AB}), E_{K_{AB}}(ID_A, Cert_B)$
Category: 1

(a)

$1.A \longrightarrow B : ID_A, Cert_A, N_A$
$2.B \longrightarrow A : E_A(K_{AB}), E_{K_{AB}}(ID_A, N_A, Cert_B)$
Category: 0

(b)

Figure 18: An instance of dataset in analysis of key freshness. (a) MSR scheme without key freshness denoted by category 1. (b) Improved MSR scheme with key freshness denoted by category 0.

$1.A \longrightarrow B : a^x, Sig_A(ID_A, ID_B, a^x)$
$2.B \longrightarrow A : a^y, Sig_B(ID_A, ID_B, a^x, a^y)$
Category: 0

(a)

$1.A \longrightarrow B : a^x, Sig_A(ID_A, ID_B, a^x)$
$2.B \longrightarrow A : a^y, Sig_B(ID_A, ID_B, a^y)$
Category: 1

(b)

Figure 19: Another instance of dataset in analysis of key freshness. (a) A secure key agreement protocol with key freshness denoted by category 0. (b) A vulnerable key agreement protocol without key freshness denoted by category 1.

$1.A \longrightarrow B : A, B, \{N_A, A, B\} K_{AS}$
$2.B \longrightarrow S : A, B, \{N_A, A, B\} K_{AS}, \{N_B, A, B\} K_{BS}$
$2.S \longrightarrow B \{N_A, K_{AB}\} K_{AS}, \{N_B, K_{AB}\} K_{BS}$
$4.B \longrightarrow A : \{N_A, K_{AB}\} K_{AS}$
Category: 1

(a)

$1.A \longrightarrow B : A, B, N_A$
$2.B \longrightarrow S : A, B, N_A, N_B$
$3.S \longrightarrow B : \{N_A, A, B, K_{AB}\} K_{AS}, \{N_B, A, B, K_{AB}\} K_{BS}$
$4.B \longrightarrow A : \{N_A, A, B, K_{AB}\} K_{AS}$
Category: 0

(b)

Figure 20: An instance of dataset in analysis of key authentication. (a) Otway-Rees's protocol without key authentication denoted by category 1. (b) Abadi and Needham's protocol with key authentication denoted by category 0.

$1.A \longrightarrow B : A, a^x, \text{hash}(A, B, \text{Password}_A)$
$2.B \longrightarrow A : B, a^y, \text{hash}(K_{AB}, a^y, A, B, \text{Password}_A)$
$3.B \longrightarrow A : \text{hash}(K_{AB}, a^y)$
Category: 1

(a)

$1.A \longrightarrow B : A, a^x, \text{hash}(A, B, K_{AB}, \text{Password}_A)$
$2.B \longrightarrow A : B, a^y, \text{hash}(K_{AB}, a^y, A, B, \text{Password}_A)$
$3.B \longrightarrow A : \text{hash}(K_{AB}, a^y)$
Category: 0

(b)

Figure 21: An instance of dataset in analysis of password guessing attack. (a) Lee-Sohn-Yang-Won password-based protocol prone to password guessing attack denoted by category 1. (b) Improved Lee-Sohn-Yang-Won password-based protocol resistant to password guessing attack denoted by category 0.

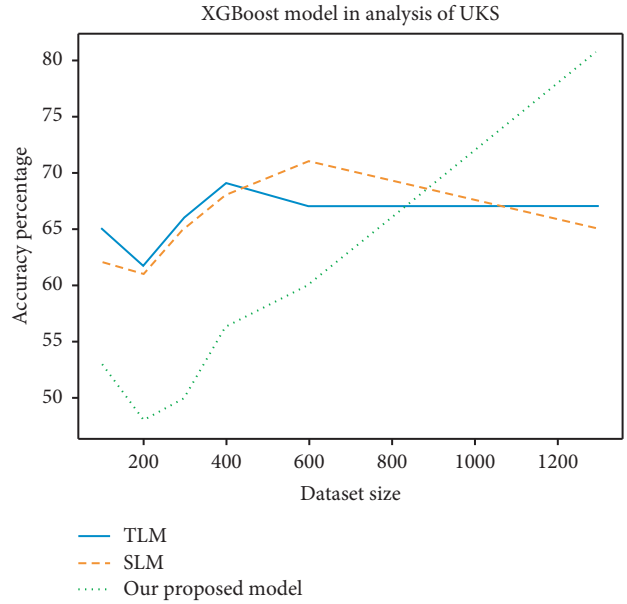

Figure 22: Analysis of UKS with XGBoost model.

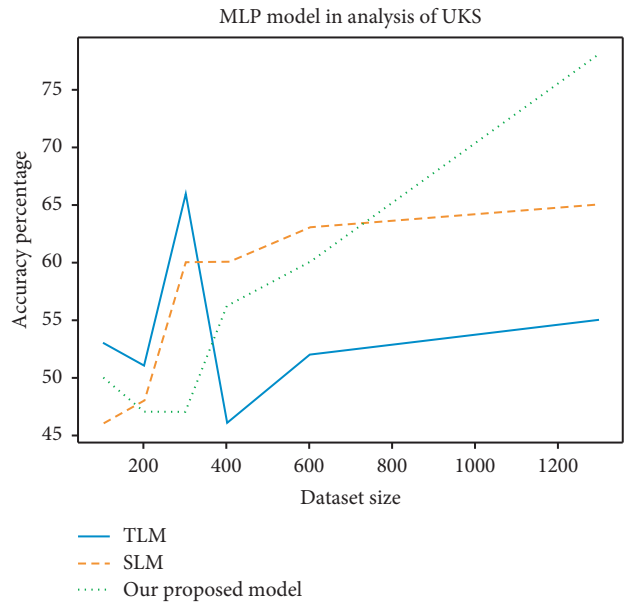

Figure 23: Analysis of UKS with the MLP model.

increase of the dataset size, as opposed to the other two models, namely, TLM and SLM models, in which the dataset size has almost no effect on the classification accuracy. For a large number of datasets, that is, the number of protocols is 1300, the classification accuracy reaches over 80% which is 20% higher than the other two models. The higher classification accuracy of TLM and SLM models for a low number of protocols, that is, 100–600, may be tempting to conclude that our model is unable to train the machine. However, with the increase of the number of datasets, the accuracy of the TLM and SLM models either decreases or remains constant. Further, the performance of TLM model is extremely fluctuating in MLP algorithm.

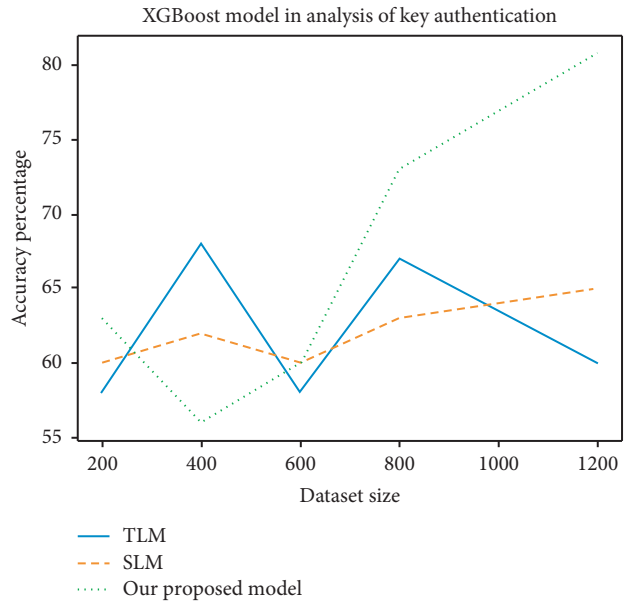

Figure 24: Analysis of key authentication with the XGBoost model.

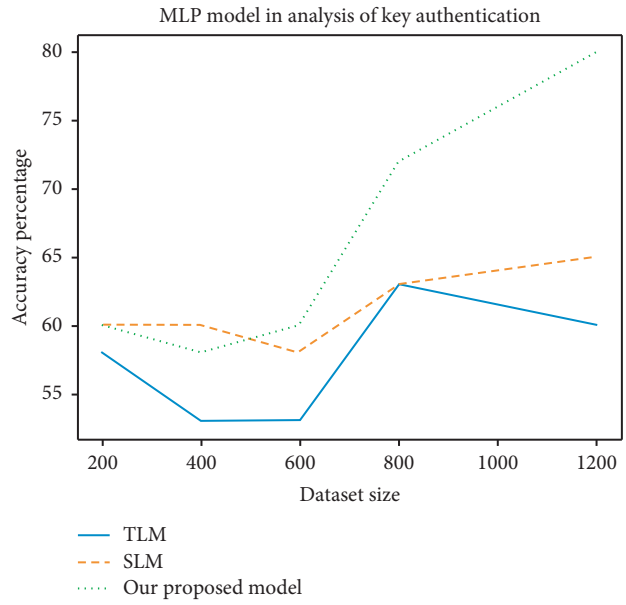

Figure 25: Analysis of key authentication with the MLP model.

4.2. Experimental Results of Analyzing Key Authentication. In the analysis of key authentication, TLM and SLM dataset constructions fail to train the machine. According to Figures 24 and 25, increase of the dataset size not only has no effect on the classification accuracy but also decreases the accuracy in case of TLM model. Meanwhile, the classification accuracy of our proposed model rises with the increase of the dataset size and reaches over 80% for 1200 protocols which is 15–20% higher than the other two models.

4.3. Experimental Results of Analyzing Key Freshness. Key freshness is a simpler security property compared to UKS attack and key authentication, as it affects only a few

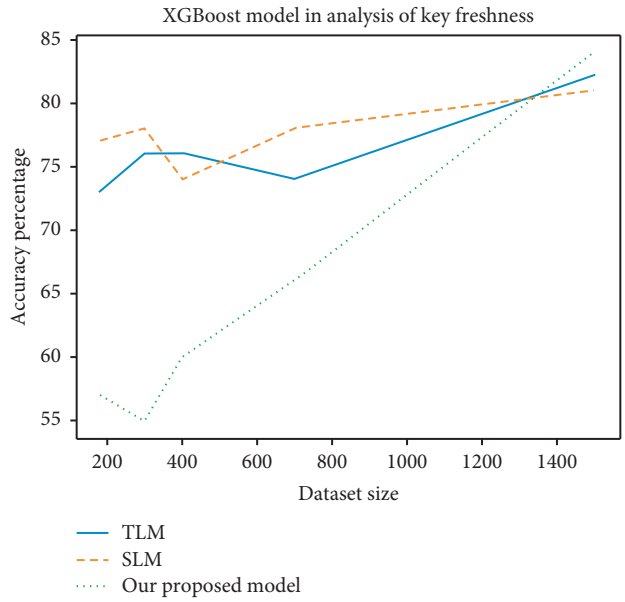

Figure 26: Analysis of key freshness with the XGBoost model.

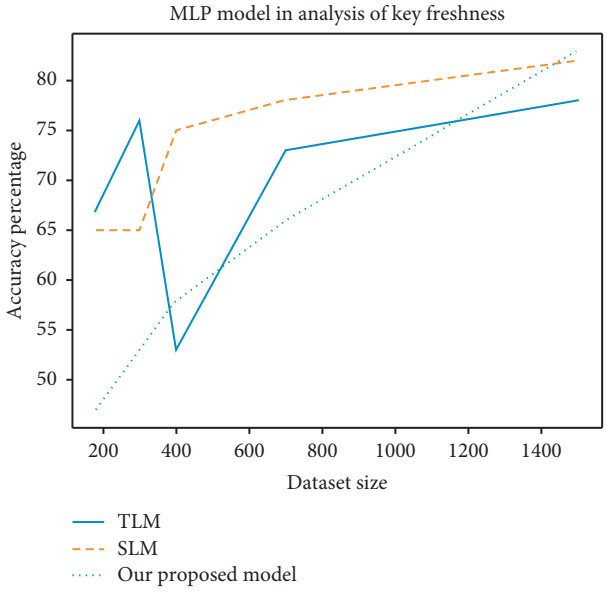

Figure 27: Analysis of key freshness with the MLP model.

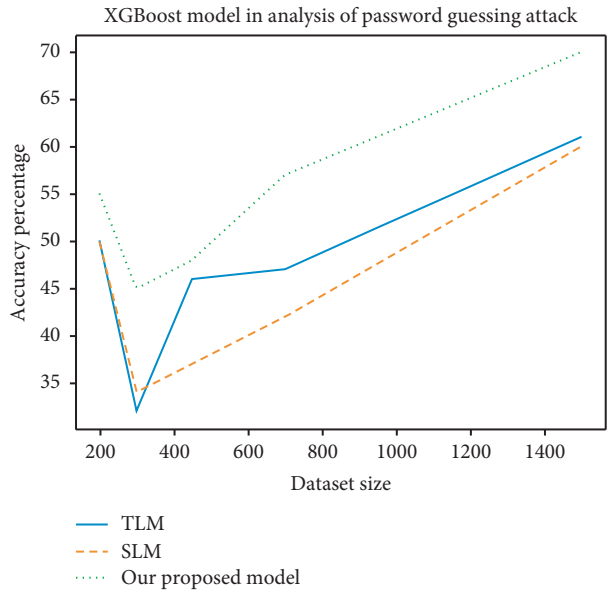

Figure 28: Analysis of password guessing attack with XGBoost model.

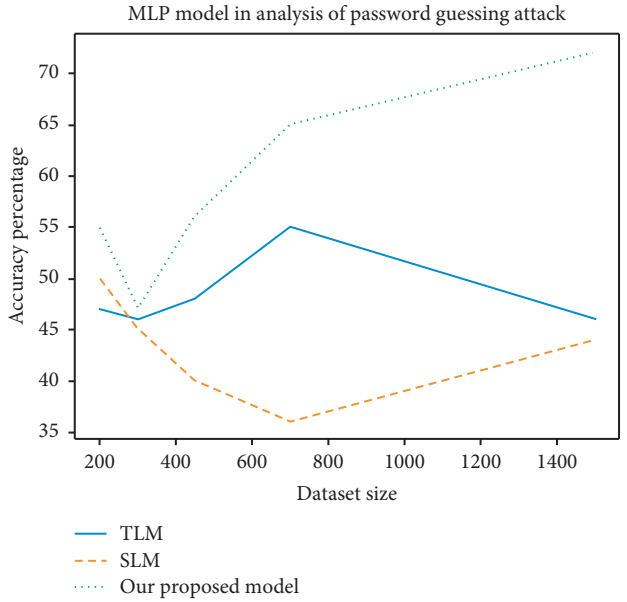

Figure 29: Analysis of password guessing attack with MLP model.

parameters such as timestamp and nonce. As a result, the classification accuracy of TLM and SLM models still improves with the increase of the dataset size. However, as shown in Figures 26 and 27, the rate of the increase of classification accuracy of our proposed model is almost three times more than the TLM and SLM models. Similar to UKS attack, the performance of TLM model is so fluctuating in MLP algorithm. For a large number of datasets, that is, the number of protocols is 1500, the classification accuracy reaches over 80% which is 10% higher than the TLM and SLM models.

4.4. Experimental Results of Analyzing Password Guessing Attack. As depicted in Figures 28 and 29, in MLP approach, the classification accuracy of TLM and SLM dataset constructions extremely decreases with the increase of the dataset size. Although XGBoost solver is able to train the machine using the TLM and SLM dataset constructions, its classification accuracy is still much lower than our proposed dataset construction. For a large number of datasets, that is, the number of protocols is 1200, the classification accuracy reaches 60% which is still 10% lower than our proposed dataset construction.

5. Conclusion, Limitation, and Future Work

Considering the difficulties of formal protocol analysis approaches, researchers have begun to apply machine learning in this area. In this paper, we investigated Ma et al.'s framework as the first attempt in applying machine learning to protocol security analysis. The main limitation of Ma et al.'s framework is that it only considers replay attack and key confirmation. Further, it exploits multiclassification as a security framework for such analysis in which every protocol or dataset is either secure or prone to a security attack such as replay attack, key confirmation, or other attacks. However, we show that multiclassification problem is not an appropriate framework. As a result, we propose binary classification in which every protocol is either prone to a specific attack or secure against that. In addition, more security properties and attacks are considered to analyze protocols against, such as strong authentication and Unknown Key Share (UKS) attack, key freshness, key authentication, and password guessing attack. Despite previous dataset construction models suggested by Ma et al., in our proposed dataset construction model, the classification accuracy increases with the increase of the dataset size, which represents the fact that our proposed dataset construction model is capable of training the machine to analyze security attacks and properties. The most evident limitation of our work is the fact that the accuracy of our scheme is only 80%. However, for a practical analysis scheme, we need an ideal analysis scheme with an accuracy of 100%. As a future work, more datasets can be provided to reach an ideal analysis scheme. In addition, more complex security properties can be analyzed using machine learning techniques such as pretraining and few-shot learning.

Data Availability

Supplementary codes and datasets are available at https://github.com/zahednejad/protocol-analysis-with-machinelearning.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by Guangzhou University and the Guangdong Provincial Natural Science Foundation, under Grant no.2018A030310071.

References

[1] C. Boyd, A. Mathuria, and D. Stebila, *Protocols for Authentication and Key Establishment*, Vol. 1, Springer, Heidelberg, Germany, 2003.

[2] T. Dierks and E. Rescorla, *The Transport Layer Security (TLS) Protocol*, 2008, https://www.hjp.at/doc/rfc/rfc5246.html.

[3] B. Blanchet, "Modeling and verifying security protocols with the applied pi calculus and ProVerif," *Foundations and Trends® in Privacy and Security*, vol. 1, no. 1-2, pp. 1–135, 2016.

[4] C. J. Cremers, "The Scyther Tool: verification, falsification, and analysis of security protocols," in *Proceedings of the International Conference on Computer Aided Verification*, Springer, Berlin, Germany, pp. 414–418, July 2008.

[5] A. Armando, D. Basin, Y. Boichut et al., "The AVISPA tool for the automated validation of internet security protocols and applications," in *Proceedings of the International Conference on Computer Aided Verification*, Springer, Berlin, Germany, pp. 281–285, 2005, July.

[6] B. Blanchet, "CryptoVerif: computationally sound mechanized prover for cryptographic protocols," *Formal Protocol Verification*, vol. 117, p. 156, 2007.

[7] A. Bleeker and L. Meertens, "A semantics for BAN logic," in *Proceedings of the DIMACS Workshop on Design and Formal Verification of Security Protocols*, New Brunswick, NJ, USA, September 1997.

[8] I. Cervesato, "The Dolev-Yao intruder is the most powerful attacker," in *Proceedings of the 16th Annual Symposium on Logic in Computer Science—LICS*, vol. 1, Boston, MA, USA, June 2001.

[9] F. J. T. Fábrega, J. C. Herzog, and J. D. Guttman, "Strand spaces: why is a security protocol correct?" in *Proceedings of the 1998 IEEE Symposium on Security and Privacy*, IEEE, Oakland, CA, USA, pp. 160–171, 1998 May.

[10] H. Maghrebi, T. Portigliatti, and E. Prouff, "Breaking cryptographic implementations using deep learning techniques," in *Proceedings of the International Conference on Security, Privacy, and Applied Cryptography Engineering*, pp. 3–26, Kharagpur, India, 2016.

[11] S. Picek, I. Petros, and S. Jaehun, "On the performance of convolutional neural networks for side-channel analysis," in *Proceedings of the International Conference on Security, Privacy, and Applied Cryptography Engineering*, Springer, Cham, Switzerland, pp. 157–176, 2018.

[12] A. Gohr, "Improving attacks on round-reduced speck32/64 using deep learning," in *Proceedings of the Annual International Cryptology Conference*, Springer, Cham, Switzerland, 2019.

[13] J. So, "Deep learning-based cryptanalysis of lightweight block ciphers," *Security and Communication Networks*, vol. 2020, Article ID 3701067, 11 pages, 2020.

[14] T. Li, X. Liu, Z. Qin, and X. Zhang, "Formal analysis for security of Otway–Rees protocol with ban logic," in *Proceedings of the First International Workshop on Database Technology and Applications*, pp. 590–593, Hubei, China, 2009.

[15] K. Liu, J. Ye, and Y. Wang, "The security analysis on Otway–Rees protocol based on ban logic," in *Proceedings of the Fourth International Conference on Computational and Information Sciences*, pp. 341–344, Chongqing, China, 2012.

[16] S. Pan, T. Morris, and U. Adhikari, "Developing a hybrid intrusion detection system using data mining for power systems," *IEEE Transactions on Smart Grid*, vol. 6, no. 6, pp. 3104–3113, 2015.

[17] M. Stevanovic and J. M. Pedersen, "An efficient flow-based botnet detection using supervised machine learning," in *Proceedings of the International Conference on Computing, Networking and Communications*, pp. 797–801, Honolulu, Hawaii, 2014.

[18] Z. Ma, Y. Liu, Z. Wang, H. Ge, and M. Zhao, "A machine learning-based scheme for the security analysis of

authentication and key agreement protocols," *Neural Computing and Applications*, pp. 1–13, 2018.
[19] T. Chen, T. He, M. Benesty, V. Khotilovich, and Y. Tang, "Xgboost: extreme gradient boosting," *R Package Version*, vol. 4-2, pp. 1–4, 2015.
[20] J. Daemen and V. Rijmen, *The Design of Rijndael: AES-The Advanced Encryption Standard*, Springer Science Business Media, Berlin, Germany, 2013.
[21] D. Coppersmith, "The data encryption standard (DES) and its strength against attacks," *IBM Journal of Research and Development*, vol. 38, no. 3, pp. 243–250, 1994.
[22] ISO Information, "Technology-security techniques-entity authentication-part 2: mechanisms using symmetric encipherment algorithms," *International Standard*, 1999.
[23] B. Zahednejad, M. Azizi, and M. Pournaghi, "A novel and efficient privacy preserving TETRA authentication protocol,"in Proceedings of the 2017 14th International ISC (IranianSociety of Cryptology) Conference on Information Security andCryptology (ISCISC), IEEE, pp. 125–132, Shiraz, Iran, 2017, September.
[24] S. M. Pournaghi, B. Zahednejad, M. Bayat, and Y. Farjami, "NECPPA: a novel and efficient conditional privacy-preserving authentication scheme for VANET," *Computer Networks*, vol. 134, pp. 78–92, 2018.
[25] A. Akbarzadeh, M. Bayat, B. Zahednejad, A. Payandeh, and M. R. Aref, "A lightweight hierarchical authentication scheme for internet of things," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 7, pp. 2607–2619, 2019.
[26] ISO Open, "Systems interconnection-basic reference model-part 2: security architecture," ISO Open, Geneva, Switzerland, 1989.
[27] T. Matsumoto, Y. Takashima, and H. Imai, "On seeking smart public-key-distribution systems," *Transactions of the IECE of Japan*, vol. E69, no. 2, pp. 99–106, 1986.
[28] S. Yamaguchi, K. Okayama, and H. Miyahara, "Design and implementation of an (authentication system in WIDE internet environment," in *Proceedings of the IEEE Region 10 Conference on Computer and Communications Systems*, pp. 653–657, Hong Kong, 1990.
[29] J. Clark and J. Jacob, "On the security of recent protocols," *Information Processing Letters*, vol. 56, no. 3, pp. 151–155, 1995.
[30] J. Alfred, C. Paul, and A. Scott, *Handbook of Applied Cryptography*, CRC Press, Boca Raton, FL, USA, 1997.
[31] L. Gong, "Variations on the themes of message freshness and replay," in *Proceedings of the 6th IEEE Computer Security Foundations Workshop*, IEEE Computer Society Press, Franconia, NH, USA, pp. 131–136, 1993.
[32] D. E. Denning and G. M. Sacco, "Timestamps in key distribution protocols," *Communications of the ACM*, vol. 24, no. 8, pp. 533–536, 1981.
[33] D. Gollmann, "Authentication by correspondence," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 1, pp. 88–95, 2003.
[34] C. Boyd and W. Mao, "On a limitation of BAN logic," in *Advances in Cryptology-Eurocrypt '93*, T. Helleseth, Ed., vol. 765, pp. 240–247, Springer-Verlag, Berlin, Germany, 1994.
[35] S. Blake-Wilson and A. Menezes, "Entity authentication and authenticated key transport protocols employing asymmetric techniques," in *Security Protocols 5th International Workshop*, B. Christianson, Ed., vol. 1361, p. 137, Springer-Verlag, Berlin, Germany, 1998.
[36] G. Horng and C.-K. Hsu, "Weakness in the Helsinki protocol," *Electronics Letters*, vol. 34, no. 4, pp. 354-355, 1998.
[37] J. Chris and C. Y. Yeob, "Fixing a problem in the Helsinki protocol," *ACM Operating Systems Review*, vol. 32, no. 4, pp. 21–24, 1998.
[38] P. Syverson, "A taxonomy of replay attacks," in *Proceedings of the 7th IEEE Computer Security Foundations Workshop*, IEEE Computer Society Press, Franconia, NH, USA, pp. 187–191, 1994.
[39] R. Graham, "How hackers will crack your password," 2009, http://www.darkreading.com/hacked-off/how-hackers-will-crack-your-password/227700892.