Neural Computing and Applications (2020) 32:16819–16831
https://doi.org/10.1007/s00521-018-3929-8

SMART DATA AGGREGATION INSPIRED PARADIGM & APPROACHES IN IOT APPLNS

A machine learning-based scheme for the security analysis of authentication and key agreement protocols

Zhuo Ma[1] · Yang Liu[1] · Zhuzhu Wang[1] · Haoran Ge[1] · Meng Zhao[1]

Received: 9 October 2018 / Accepted: 30 November 2018 / Published online: 10 December 2018

Abstract
This paper proposes a novel machine learning-based scheme for the automatic analysis of authentication and key agreement protocols. Considering the traditional formal protocol analysis schemes, their analysis accuracies depend heavily on the prior knowledge possessed by the analyst and the subjective understanding of the protocol. The rapid development of artificial intelligence in security field shows that the ideal way to get rid of the dependency is to use machine learning. Hence, we elaborately compare more than 2000 protocol analysis results and select 500 most representative ones of them to build a protocol dataset. Combining the protocol representation method of traditional schemes, these selected protocols are expressed as weight matrixes based on security components. Furthermore, a machine learning-based security analysis model is proposed to automatically find the attacks of the protocol. For now, three types of attacks against authentication and key agreement protocols can be identified based on our model. And experiment results show that it can reach almost 72% upper-bound performance. From the derivative of the accuracy curves, it can be inferred that the performance of our scheme will definitely get better as the dataset expands.

Keywords Authentication protocols · Machine learning · Formal analysis of protocol security · Protocol dataset

1 Introduction

Over the past decades, the formal security analysis of protocols has always relied on the traditional model checking or theorem proving approaches [1–3]. Although these methods can find some vulnerabilities in known protocols or prove their security [4, 5], the accuracies of the analysis results depend heavily on the prior knowledge of the analyst and the subjective understanding of the protocol. This causes that two different analysts using the same method to analyze the identical protocol may have distinct results. For instance, Tingyuan et al. [6] proposed an ideal model of the Otway–Rees protocol. Based on BAN logic [7], he proved the security of the protocol. Liu et al. [8] also used BAN logic to analyze the same protocol, but pointed out that this protocol was vulnerable to man-in-the-middle attack and typing flaw attack. What is more, almost all of the relevant automatic analysis tools are based on these approaches [9–11]. This leads to that, for an apprentice or layman, the application of the automated analysis tools is difficult to get in the door.

To overcome the flaws mentioned above, we try to explore some novel ideas from other security areas. Having read many newly published papers, we find that the powerful data analysis capabilities of machine learning can help us to lower the difficulty of cyberspace security analysis. For instance, Shengyi et al. [12] developed a machine learning-based intrusion detection system (IDS) that allows a person, even without professional knowledge,

Zhuo Ma and Yang Liu have contributed equally to this work.

✉ Zhuo Ma
mazhuo@mail.xidian.edu.cn

Yang Liu
bcds2018@foxmail.com

Zhuzhu Wang
wangzhuzhu2018@163.com

Haoran Ge
1748491087@qq.com

Meng Zhao
zm_cici@163.com

[1] School of Cyber Engineering, Xidian University, Xi'an 710071, China

to be able to accurately classify disturbance, normal control operations and cyber-attacks. Matija et al. [13] used supervised machine learning to make it easier for security analyst to detect botnet with traffic flow. Inspired by these excellent research results, we try to propose our own machine learning-based model to make the machine to become a protocol analysis master which can help us analyze the security of the protocols automatically. In the model, we abstract the security analysis process of the protocol into a multi-classification model. The multi-classification is based on the attack types of the protocols under specific security goals. And by treating "no vulnerability" as a particular type of attack, we ensure that all protocols are able to be covered in the model. For machine learning-based models, there are always two key points that we should pay attention to. One is the features of the research target, and the other is machine learning models. In summary of the traditional formal protocol analysis methods, it can be discovered that BAN logic [7], strand space [14, 15] and almost every other method use symbolic security components to describe and analyze protocols. And countless research results have proved that such a way of protocol description is able to accurately reflect the protocol security characteristics. Hence, in our model, we also divide the messages of the protocol into security components as protocol features. But the difference is that, to facilitate machine learning, we further convert these components into weighted message vectors. As for the machine learning models, the first one that comes into our consideration is long short-term memory network (LSTM) because the protocol is essentially composed of messages with time sequence that can be well expressed by LSTM. Then, due to the complexity of the protocol and the difficulty of manually collecting protocols as dataset, we have to choose models to overcome the problems brought by the small size of samples with high dimension. Therefore, eXtreme Gradient Boosting (Xgboost) and support vector machine (SVM) are also appropriate for our model.

After determining the basic architecture of the above model, we do the following groundbreaking experiments. Firstly, more than 500 classic protocols are collected from more than 2000 papers. Secondly, with reference to [16], we define specific security goals and adversary model for our scheme. Then, we intuitively provide three dataset models to describe the security features of the protocol. In these models, all of the weighted message vectors are further processed in different forms. But they all make the vectors belonging to a specific protocol into a weight matrix or directly concatenate together. Thus, the 500 protocols can be just viewed as 500 computable numerical two-dimensional matrixes or 500 long vectors. And by a self-designed tool, the format conversion process of these protocols is completely automatic. Finally, we utilize LSTM, Xgboost and SVM to try to fit the mapping relationship between the protocol security features and the attacks.

It is worth noting that every security protocol contains local computational operations which are always related to very complex mathematical features. Directly adding these features into our model will more than doubled the difficulty of the feature extraction process. However, the fact is that the attacks of many protocols can be discovered without concern about the mathematical features, such as the analysis of dozens of protocols in [17]. Therefore, in this paper, we choose Dolev–Yao model [18] as our standard adversary model, who thinks that the cryptographic algorithms used in the protocols are ideal and unbreakable. And in future work, we will try to cover this defect.

The rest of the paper is organized as follows. Section II briefly introduces the formal description of security, including security goals and the adversary model in our analysis. In Section III, we provide three dataset models to describe protocols and determine four specific attack types. Section IV gives the experiment results to evaluate the performance of the proposed scheme. Section V concludes this paper.

2 Formal description of security

As far as we know, there is no one who has ever tried to do the similar research. For this reason, it is not realistic to directly analyze all types of protocols in the new presented scheme. In this section, four security goals and one adversary model are defined for only two-party or three-party entity authentication protocols and key agreement protocols. And the key agreement contains key agreement and key distribution. In addition, the security goals do not cover every possible goal, but they are enough for determining the attacks of the protocols that we focus on. Based on the definitions, the protocols in the dataset can be classified into four distinctive types.

2.1 Security goals

Given that people have done a lot of work on the formal definition of security goals in the previous research [19–22], we only briefly describe the security goals needed in this paper. Readers may refer to [16] for details. As shown in Fig. 1, there are four goals to reach for mutual authentication and key agreement protocol. And every goal contains several security properties. Remarkably, the four security goals are not all expected by the both two types of the protocols. For the mutual entity authentication protocol, the first goal below is enough. But a secure key agreement protocol typically needs to satisfy them all. And according to the definition, the unilateral authentication protocols are

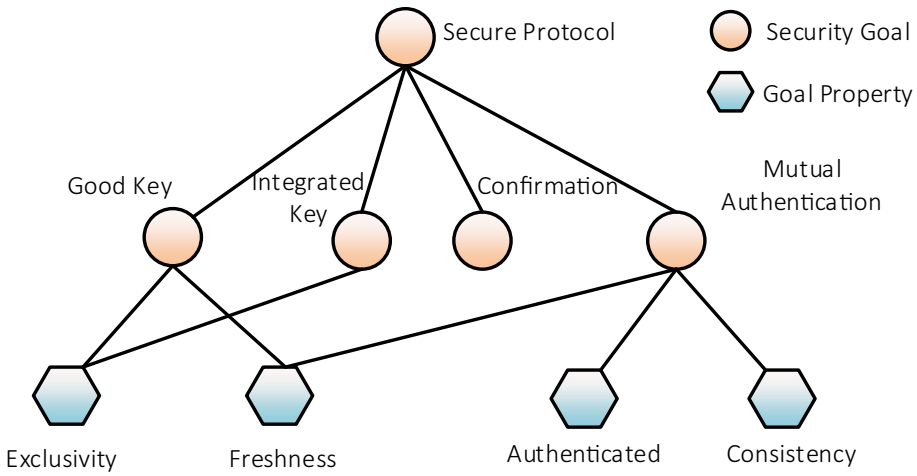


Fig. 1 Hierarchy of mutual authentication and key agreement goals

considered insecure in our model. The four goals can be briefly described as follows.

2.1.1 Mutual entity authentication [16]

- First party A is assured of the identity of a second party B involved in a protocol and that the second has actually participated, and vice versa. This process can also be done with a trusted third party S.
- The corroborative evidence is fresh to assure that B has knowledge of A as his peer entity and vice versa.
- When a principal A accepts the other party's identity, the other party's record of the partial or full run matches A's.

2.1.2 Good key [16]

- For the shared session key, a good key is fresh and known only to A and B and mutually trusted parties.
- For the public session key, a good key is fresh and its corresponding private key is known only to the owner.

2.1.3 Integrated key [16]

- For a key transport protocol, if the key is accepted by any principal, it must be the same key as chosen by the key originator.
- For a key agreement protocol, if the key is accepted by any principal, it must be a known function of only the inputs of the protocol principals.

2.1.4 Key confirmation [16]

- First party is assured that a second party actually has possession of a particular session key in the form of a nonce or a timestamp and vice versa.

2.2 Adversary model

We choose Dolev–Yao model [18] as our standard adversary model. However, because the research about machine learning-based protocol analysis is still at a blank stage, the adversary model is weakened compared to the native model. Firstly, to make the machine learning-based model focus on only one protocol at one time, we limit the ability of an adversary to launch an attack by using information from multiple protocols. Plenty of lightweight protocols use password as their encryption key, but it is a consensus that the pure password-based encryption is weak. And it can be discovered that attacks against password are often hard to analyze by message components alone, like dictionary attack. Thus, we limit the password guessing ability of the adversary. What is more, in the next section, the dataset construction shows that too much care about the order of message components makes it difficult to learn the protocol features. For this reason, we do not allow the adversary to launch an attack against message type. Through the limitation above, the number of attacks that the adversary can launch is limited into a small scale. But based on our previous experience of failure, it is still too hard to make the machine learning models to get ideal convergence result under current small size of dataset. Hence, we further pick only four out of the attacks as research objects. The specific definitions of our adversary model are as follows.

- The adversary controls only one type of the protocol communications at the same time between all principals, but can observe all messages sent, alter message, insert new messages, delay messages or delete messages.
- The adversary is able to obtain any old session key distributed in old key transport protocol runs, but unable to guessing password or other long-term keys used in protocol runs.

- If necessary, the adversary can be a legitimate protocol principal and initiate a session with an honest insider.
- The adversary has no ability to distinguish the order of protocol components.
- The cryptographic algorithms used in the protocol are unbreakable for the adversary.

From first three definitions above, we guarantee that the adversary is capable of modifying messages, initiating legitimate sessions and terminating a session. But he cannot have enough ability to break a key or a password, even by brute force. The penultimate definition allows us to not have to care about the order of the message components, which counts a lot when we design the dataset model. Through the last definition, we limit the cryptographic analysis ability of the adversary. Otherwise, our research would be completely untenable. In conclusion, the attacks that we concentrate on are only including replay attack, modification attack, reflection attack and man-in-the-middle attack. Other attacks like DDOS attack, typing attack password guessing attack or protocol interaction attack are ignored.

3 Dataset construction

There is no one who has ever built such a kind of dataset for the security analysis of the protocol. Consequently, in this paper, we try to provide three dataset models to describe protocols. All of the protocols selected for the models are analyzed based on symbolic protocol model [23]:

- Symbolic protocol model is also known as Dolev–Yao model. In the model, data are represented symbolically as terms of a free-term algebra, and cryptographic functions are represented as operations in the same algebra. The ideal properties of real cryptographic functions are captured by the algebraic properties of the symbolic operations.

Our planned work in this section consists of two steps: The first step is to construct models to identify the threats that are not related to local computational operations, and the second step is to expand the analysis area to any kinds of threat. Up to now, we have made good progress in the first step. But for the second step, the features of local computation are too complex to extract, because dozens of possible mathematic operations in protocols usually have no fixed pattern of composition. And directly adding every possible features of local computation will definitely double the difficulty of machine learning under such small size of dataset. Hence, in the future work, besides expanding our dataset, we intend to use principle component analysis (PCA) or linear discriminant analysis (LDA) to extract pivotal features. Thus, by reducing the data dimension without losing precision as much as possible, we may be able to overcome the problems mentioned above. Without the consideration of local computation security, the point that we pay great attention to in this paper is limited to only message transmission process. As a result, deliberately, the formal description of security in Section II and the security of the 500 protocols we select have nothing to do with protocol calculation process. Moreover, the representation of protocols in symbolic like BAN logic is unaccommodated to the neural network. Therefore, we try to convert all of the protocols into weight matrixes or vector.

3.1 Features of protocols

To accurately analyze the security of protocols, appropriate protocol features should be firstly extracted. After analyzing the machine learning-based network intrusion detection [24] and the traditional formal analysis of the protocol [1–3], it can be discovered that every protocol is able to be divided into message components. And previous research results consolidate the fact that these components can reflect the security features of the protocol very well. Hence, in our models, each protocol feature corresponds to a kind of message component. The commonly used features can be divided into two types, including dynamic type and static type. The former one mainly describes the running characteristics of the protocol, like port number, message frequency and message length. Intrusion detection system often utilizes them to judge whether there is malicious traffic. The other types, such as random number, timestamp and cryptographic operation, are always used to prove the security of the protocol or find its vulnerability through theoretical analysis. Almost all of the formal analysis is based on these static features. In comparison, the static one is more appropriate for our analysis. It is unadvisable to utilize every possible static feature, because it will introduce a lot of redundancy into the dataset models. Consequently, we pick 8 message parameters, 4 cryptographic operations and 4 types of keys out of them. They are given as follows.

- *Message parameters* (*MP*): participant identity, timestamp, random number, public key, cert, shared key, DH parameter and password.
- *Cryptographic operations* (*CO*): encryption, signature, hash and keyed hash.
- *Keys* (*K*): public key, private key, shared key and password.

Note that the usage of the keys in *MP* is completely different from *K*. *MP* is used to describe the data carried in the message. *K* only participate in the cryptographic

operations. But once the key distribution or key agreement messages are accepted by both entities, the keys will be converted from *MP* to *K*. In addition, calculation features of the protocol are ignored in our scheme. The detail usage of these features is given in the following parts.

3.2 Categories of protocols

The category of a protocol in the dataset corresponds to the attack that the protocol is vulnerable to. In fact, even though the attacks are limited to only four types in Section II, it is still difficult to identify them at the same time. Consequently, we further classify the four attacks into three categories to label the dataset. Before classification, the man-in-the-middle attack is divided into two types, including weak man-in-the-middle attack and other man-in-the-middle attacks. The former type of attack is caused by the lack of key confirmation, which may lead to the violation of *Security Goal 4* in Section II. The reason why we deliberately pick it out is that this attack is very common and its features are easier to distinguish than other attacks. Then, the three categories are given as follows.

The first category only includes replay attack, which is mainly caused by the lack of freshness parameter, like random number and timestamp, or the wrong usage of freshness verification. This type of attack is very distinctive, because it only affects several features about freshness. Weak man-in-the-middle attack constitutes the second category alone. And all of the other types of attack make up the third category. It can be observed that the third category is the most complex one among the three categories to analyze. Therefore, to better learn its features, the number of protocols that belongs to it is the largest in our dataset. Moreover, we regard the secure protocols as the fourth category. In this way, we are capable of, respectively, identifying four protocol labels corresponding to the four categories that have no intersection with each other. Table 1 gives the number of protocols of the four categories.

3.3 Models of dataset construction

Before the description of dataset models, some definitions are given as follows.

Firstly, a protocol message parameter set *SP* and a parameter property set *PP* are defined as follows.

$$SP = \{sp_1, sp_2, \ldots, sp_n\} \tag{1}$$

$$PP = \{pp_1, pp_2, \ldots, pp_n\} \tag{2}$$

where sp_i can be participant identity, timestamp, random, public key, cert, shared key, password, DH parameter or other message parameters. And pp_i is one of the parameter attributes, including parameter index, encryption key, signature key and some attributes about hash. In addition, every $sp_i \in SP$, respectively, corresponds to a specific set PP_i. Because each protocol can always be represented by finite message parameters and their corresponding cryptographic operations, the lengths of *SP* and *PP* are fixed. In the following dataset models, $|SP|$ is set to a fixed integer N and $|PP|$ is M.

To reduce the dataset dimensions, a normalized function f_n is defined as follows:

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

According to Eq. (3), the dimension of the message vector can be reduced from $N \times M$ in making the results of machine learning to converge better. Based on the definitions above, a protocol $P = \{m_1, m_2, \ldots, m_k\}$, where m_i is one of the messages that constitutes the protocol, is able to be expressed as a matrix in the three models below.

3.3.1 Literal conversion model (LCM)

In LCM, the process of converting the designated protocol to a weight matrix is similar to the process algebra CSP [25]. The difference is that message components are literally interpreted into a fixed-length one-dimensional vector in our model, not logic statements. As shown in Fig. 2, a protocol message m_i is directly mapped into a vector $m_i = (sp_{i,1}, sp_{i,2}, \ldots, sp_{i,l})$. $|m_i|$ is a fixed value $L \times M$, where L is the maximum number of message parameters in a protocol message and $M = |PP|$ is the length of $sp_{i,j}$. Specially, each type of parameter has its own unique index which is predefined and always given to $p_{i,j,1}$. The values of other parameter properties are obtained according to the actual protocol. And their possible values ω are also predefined. If the number of message parameters is less than L, the extra $sp_{i,j}$ will be set to zero vector. Furthermore, m_i is normalized by function f_n as follows.

$$\begin{aligned} m_i' &= f_n(m_i) = (f_n(sp_{i,1}), f_n(sp_{i,2}), \ldots, f_n(sp_{i,l})) \\ &= (\lambda_1, \lambda_2, \ldots, \lambda_l) \end{aligned} \tag{4}$$

By Eq. (4), L becomes the reduced length of message vector. Figure 3 shows an example of the conversion

Table 1 Numbers of protocols of the four categories

Category	Freshness	Weak man in middle	Others	Secure
Number of protocols	52	120	113	213

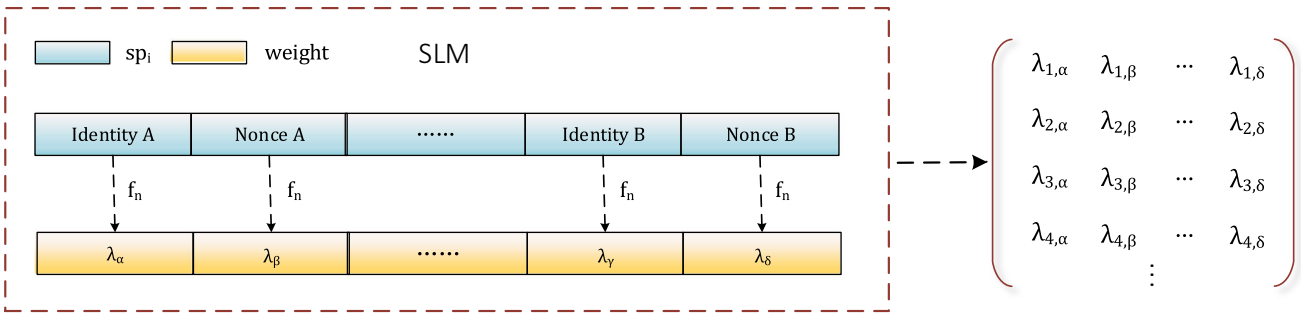


Fig. 2 Conversion process of SLM

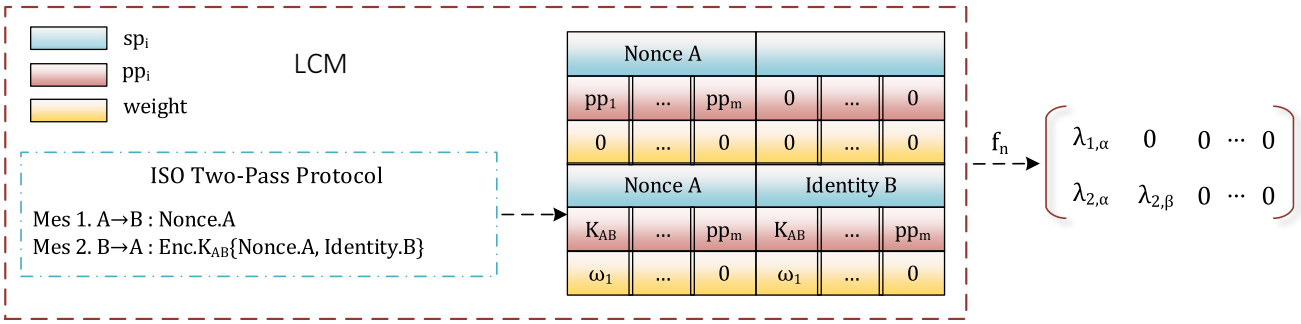


Fig. 3 Conversion process of LCM

process of the ISO symmetric key two-pass unilateral authentication protocol.

3.3.2 Two-layer model (TLM)

Unlike LCM, an empty message vector $m_i = (sp_{i,1}, sp_{i,2}, \ldots, sp_{i,n})$, where $sp_{i,j} = m_i^{'}$ is set to zero vector, is predefined before the conversion in TLM. All of the $\mathrm{sp}_{i,j} = m_i^{'}$ are placed in order and have their own specific meanings. It means that each dimension of the vector $m_i^{'}$ corresponds to a fixed parameter property, like $ID_A.Encryption\_Key$, $Nonce_B.Signature\_Key$ or $Times-tamp_S.Index$. Then, when a message of the protocol comes, its parameters and corresponding cryptographic operations just have to be put into the right dimensions. Also, every possible value of the attributes is predefined. Because all of the possible message parameters in set SP are put into the message vector, $|m_i^{'}|$ in TLM is a fixed value $N \times M$. Figure 4 shows the details of the conversion process. Due to the fact that most of the protocol messages are made up of only several parameters, the density of the valid data in the final weight matrix is usually very low in TLM. But among the three models, TLM is the most intuitive one to describe protocol.

3.3.3 Single-layer model (SLM)

Until the message parameters and corresponding cryptographic operations are put into the vector m_i, the SLM conversion process is identical to the TLM. However, in this model, all the $sp_{i,j}$ are further processed by the normalized function f_n. We are able to obtain a normalized vector $m_i^{'}$ as follows.

$$\begin{aligned} m_i^{'} &= f_n(m_i) = (f_n(sp_{i,1}), f_n(sp_{i,2}), \ldots, f_n(sp_{i,n})) \\ &= (\lambda_1, \lambda_2, \ldots, \lambda_n) \end{aligned} \tag{5}$$

$|m_i^{'}| = N$ is the length of message parameter set SP. Thus, the two-layer construction of message vector in TLM is reduced to single one in SLM. And the number of data dimensions is also changed from $N \times M$ to N. It is worth mentioning that, since the current dataset that we use is still small, the normalized function may help us low the difficulty of classification for machine learning. Figure 4 shows the details of conversion process.

4 Comparison of dataset construction and experiments results

In this section, we compare the performance of the three kinds of dataset constructions in different machine learning models. From the experiment results, it can be seen that the classification accuracies of them rise with the increase in

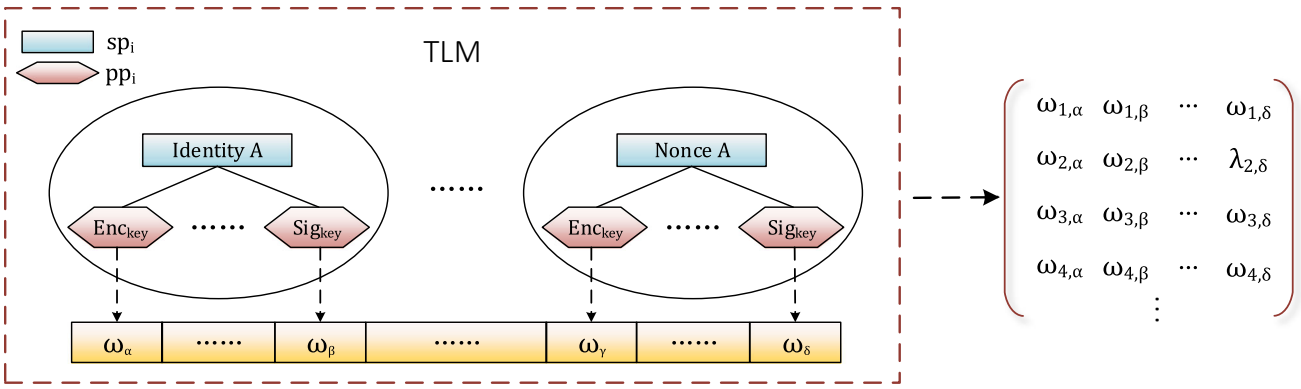


Fig. 4 Conversion process of TLM

the dataset size. Hence, we believe that the fusion of formal security analysis and the machine learning technique has considerable prospect. Since there is no one who has ever solved such a novel problem with machine learning, we try to experiment with three different models. Since there is no one who has ever analyzed protocol security with machine learning, we try to experiment with three different models. Taking into account the nature of the network protocols, timing sensitive comes into our sight firstly. In common, every protocol is made of several message sequences that are processed by the participants in order, just like the system log. Therefore, inspired by DeepLog [26], a system log analysis scheme, we choose LSTM as the most ideal machine learning model. Then, from the experiment results of LSTM, it can be discovered that the training results under the model has reached more than 68%, but there is still overfitting. And this is due to the fact that the dimensions of our three models are high and the amount of protocols samples we collect is not very large. To avoid this problem, we consider Xgboost and SVM. By fitting additive tree model and introducing regulation term into traditional boosting method, Xgboost enables automatic feature selection and captures high-level interactions without interruption. And this lets Xgboost better interpret our model to avoid overfitting. As for SVM, it has excellent nonlinear generalization ability to high dimension and small sample evaluation problem. What is more, limited by objective factors, we can only use 500 protocols found in more than 2000 papers at current stage. This number will be increased in future work. In order to avoid overfitting, the learning ability of LSTM is minimized as much as possible. For example, the number of neurons in hidden layer is set to 16, which is small, but completely sufficient to learning the features of the protocol. However, in Xgboost, the max depth of the tree is set to 10, which is a little larger than the default. Because this model itself has a strong mechanism to avoid overfitting. In addition, all of the models belong to typical classification models [27, 28]. Figures 8, 9 and 10 show the accuracies of the proposed dataset constructions with different machine learning models.

4.1 Comparison of dataset constructions

Table 1 concludes the main differences among the three models. Because every dimension corresponds to a security feature and L is usually less than N, LCM has the least number of features. This makes LCM to have the most concise representation and the fastest convergence speed. However, it also means that the attributes of the protocol messages are hidden the deepest, which may lead to a disturbing training result in the machine learning models. TLM overcomes the deficiencies of LCM by introducing more features. It can be observed that, since normalized function is not applied, TLM is the only model that can directly obtain the original protocol information from the weight matrix. Thus, it becomes the most intuitive model. Unfortunately, more features bring not only intuition, but also more chances to cause overfitting under the current small-scale dataset. Compared to the other two models, SLM performed most evenly on all the indicators. This makes it balance the advantages and disadvantages between the two models and most likely to achieve the best performance in the current research stage. What is more, the experimental performances in the next section will prove the comparison results above (Table 2).

Figures 5, 6 and 7 reflect the data density of the three models in a visual way. In order to visualize the data

Table 2 Comparison of three models

	LCM	TLM	SLM
Number of features	L	$N \times M$	N
Construction layer	1	2	1
Data density	Low	High	Low

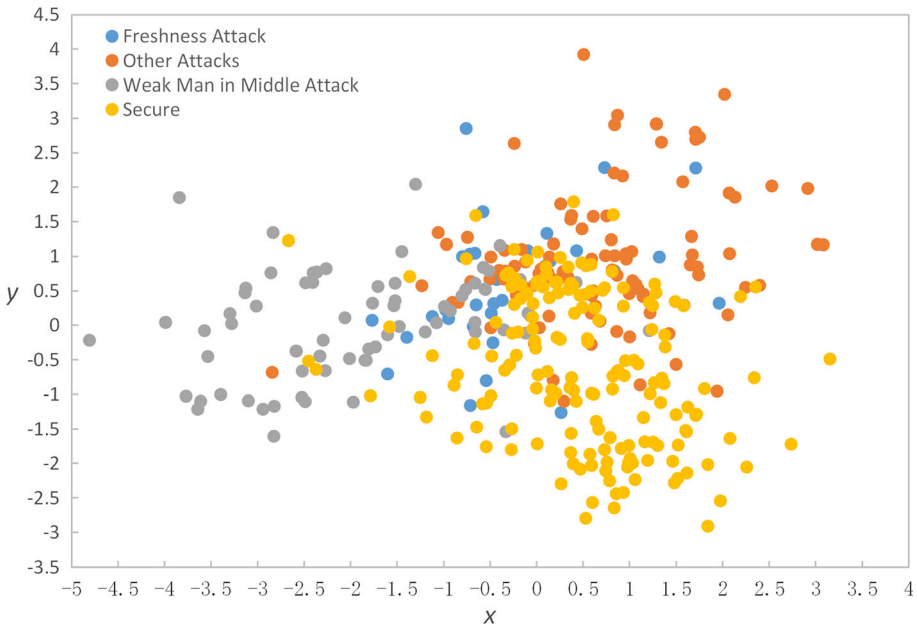


Fig. 5 Visualization of LCM data distribution

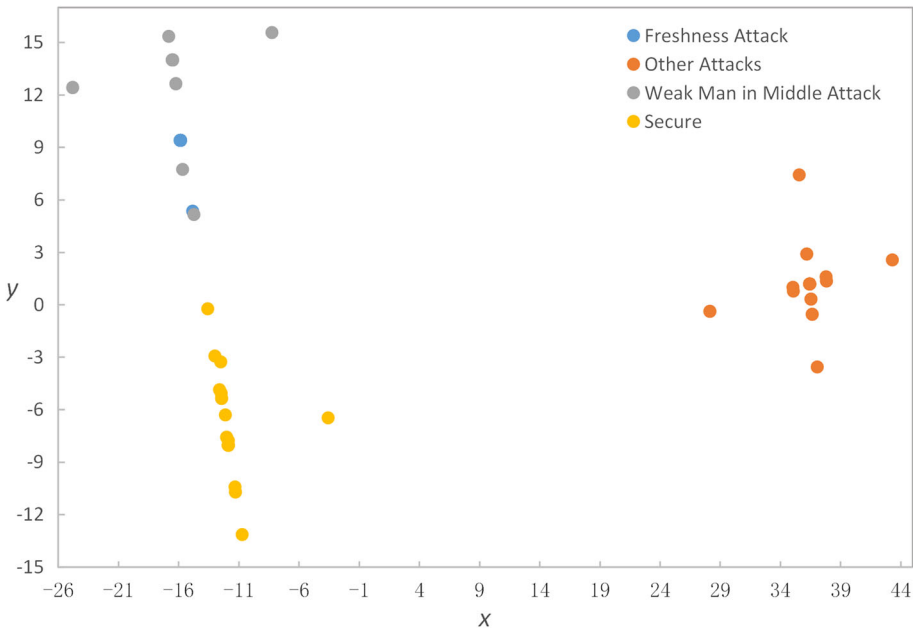


Fig. 6 Visualization of TLM data distribution

distribution, we use linear discriminant analysis (LDA) to process the dataset. LDA is one of the supervised learning models, through which the dimension of data can be reduced to two.

Figure 5 shows that most samples of *Weak Man-in-the-Middle Attack*, *Other Attack*, *Security* are well distributed, but some samples of *Freshness Attack* in LCM are mixed with the samples of the other three categories. In addition to indicate the fact that the features of *Freshness Attack* are weakened under LCM, this phenomenon may also improve the chances to misclassification. There are two reasons we find that can lead to the mixture. The first one is that, compared to the protocol numbers in the other three categories, the number of protocols belonging to *Freshness Attack* is the least at current stage. The other reason is that almost every freshness attack is only related to the misuse of several freshness parameters. The way of literal conversion is unable to give distinctive features of such small difference.

In Fig. 6, the number of protocols is same as Fig. 5 and the samples are also well distributed. The difference is that the data are concentrated in only several points around. This is because the dimension of TLM is too high, and the difference in the eigenvalues extracted by the LDA is too

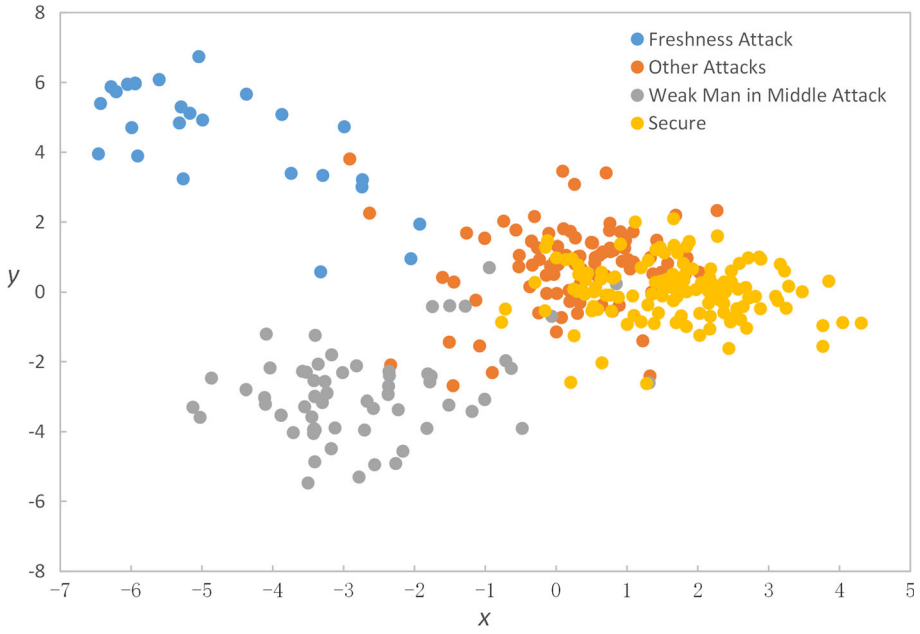
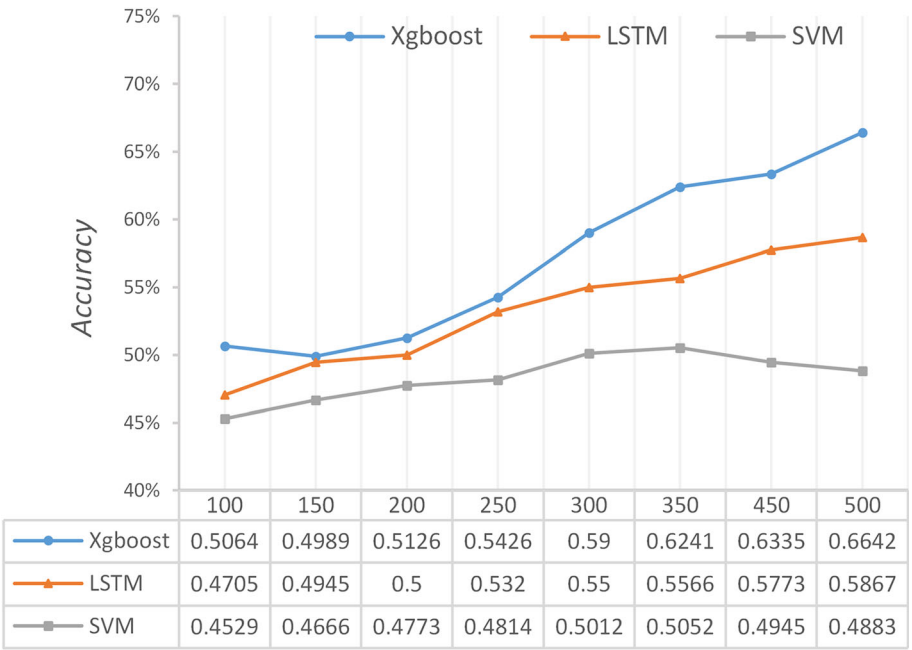


Fig. 7 Visualization of SLM data distribution



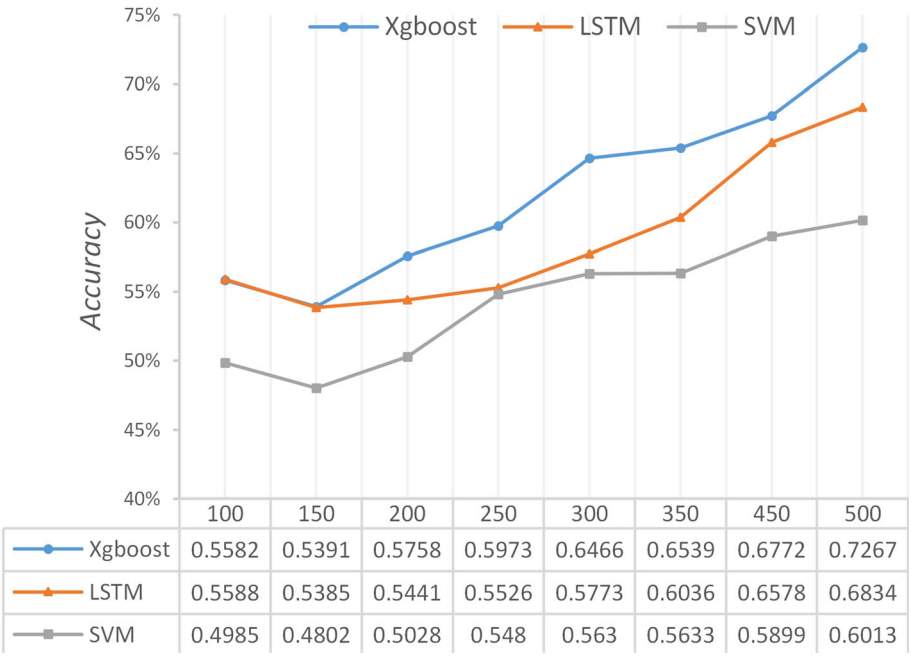
	100	150	200	250	300	350	450	500
Xgboost	0.5064	0.4989	0.5126	0.5426	0.59	0.6241	0.6335	0.6642
LSTM	0.4705	0.4945	0.5	0.532	0.55	0.5566	0.5773	0.5867
SVM	0.4529	0.4666	0.4773	0.4814	0.5012	0.5052	0.4945	0.4883

Fig. 8 Accuracies of Xgboost, LSTM and SVM in LCM with different numbers of protocols

small. Figure 7 reflects the distribution status of the four attacks in SLM. It can be seen that the protocols vulnerable to *Freshness Attack* and *Weak Man-in-the-Middle Attack* are distinguishable. Part of *Other Attack* samples and Secure samples are mixed together and hard to classify. This is due to that some protocols belonging to *Other Attack* have more similar features to the secure ones. For instance, consider the authentication protocol example which can be attacked by reflection:

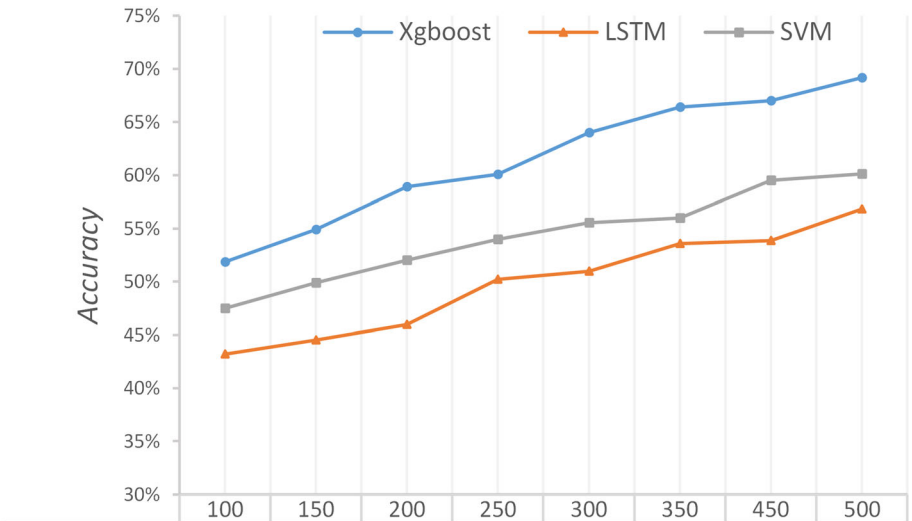
$$A \Rightarrow B : N_A$$
$$B \Rightarrow A : N_B, HMAC_{K_{AB}}\{ID_B, ID_A, N_A\}$$
$$A \Rightarrow B : HMAC_{K_{AB}}\{ID_A, ID_B, N_B\}$$

Compared to a secure protocol, ISO Three-Pass Mutual Authentication with CCFs in [17], this protocol has almost same protocol components. And they do try to authenticate the participants. However, on the whole, there are flaws in its authentication process. It is the existence of protocols like the above ones that leads to the confusion between the



	100	150	200	250	300	350	450	500
Xgboost	0.5582	0.5391	0.5758	0.5973	0.6466	0.6539	0.6772	0.7267
LSTM	0.5588	0.5385	0.5441	0.5526	0.5773	0.6036	0.6578	0.6834
SVM	0.4985	0.4802	0.5028	0.548	0.563	0.5633	0.5899	0.6013

Fig. 9 Accuracies of Xgboost, LSTM and SVM in TLM with different numbers of protocols



	100	150	200	250	300	350	450	500
Xgboost	0.5185	0.549	0.5893	0.6008	0.64	0.6642	0.6703	0.6917
LSTM	0.4317	0.445	0.4594	0.502	0.5099	0.5358	0.5387	0.5683
SVM	0.475	0.499	0.52	0.54	0.5554	0.56	0.5952	0.6013

Fig. 10 Accuracies of Xgboost, LSTM and SVM in SLM with different numbers of protocols

two categories and the less than ideal classification accuracy shown in Figs. 8, 9 and 10.

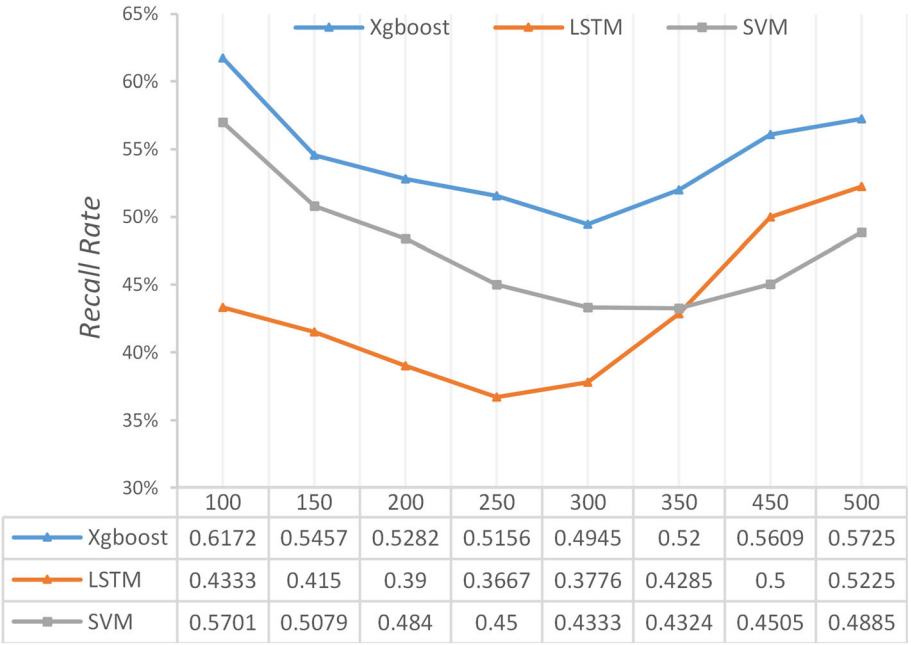
4.2 Experiment results

Figures 8 and 10 show that the accuracy of SVM is significantly lower than the other two models. As the number of samples varies from 100 to 500, its accuracy always fluctuates between 45 and 50%. This means that SVM is completely incapable of learning the security features of the protocols in the two models. But as shown in Fig. 9, its performance is better than LSTM. Considering the details of the dataset constructions in Section III, it can be discovered that this phenomenon is mainly due to the use of f_n. The normalized function we use in LCM and SLM turns multiple dimensions of parameter properties into single dimension. It leads to that some feature information is hidden. And the common core functions of SVM are not powerful enough to mine it. However, SVM has excellent

nonlinear generalization ability for small samples with high dimensions in TLM.

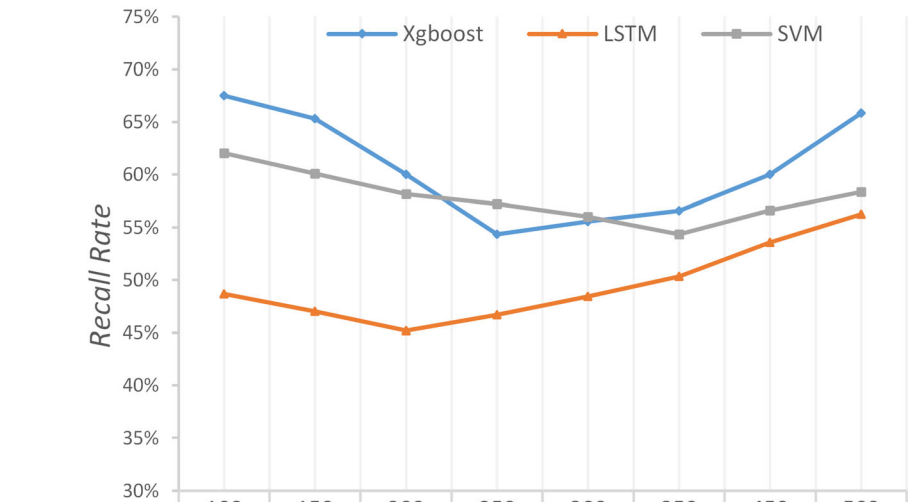
In Figs. 8, 9 and 10, the performance of LSTM and Xgboost both increases as the sample size increases. This fits well with the fact that the higher performance of a machine learning model always requires more training samples, if the model is appropriate to the problem. It is worth mentioning that, in TLM, the accuracy of LSTM is the lowest because LSTM is prone to overfitting for small samples with high dimensions. And thanks to the powerful mechanism and the high flexibility of Xgboost to avoid overfitting, it has the best performance in all of the three dataset constructions. In addition, from the trends of all the accuracy curves, it can be seen that, for all of the models, there is still a great deal of research prospect.

Can the proposed scheme accurately identify secure protocols? We, respectively, calculate the recall rates of Xgboost, LSTM and SVM in the three proposed dataset models to answer the question. The experiment results are given in Figs. 11, 12 and 13. And the recall rate is



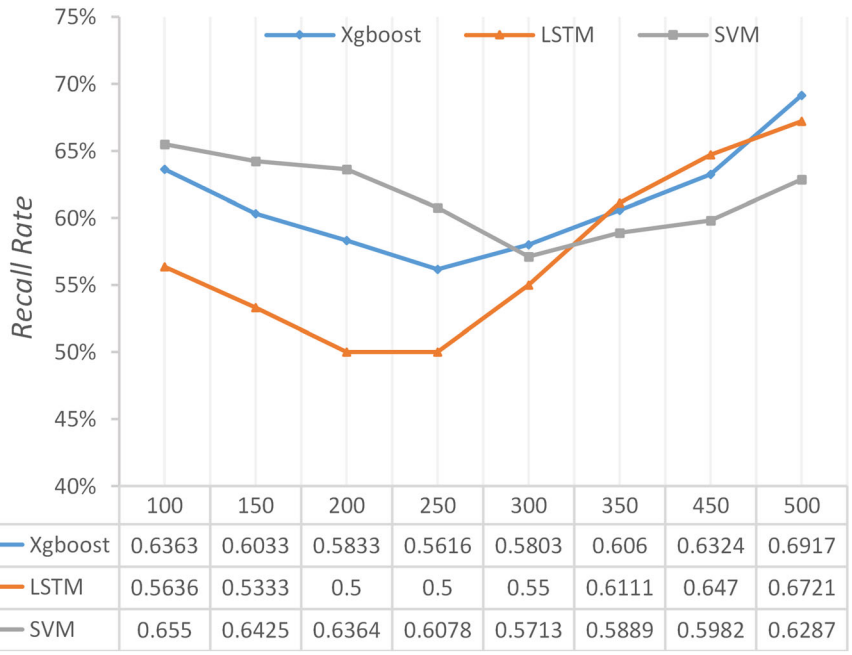
	100	150	200	250	300	350	450	500
Xgboost	0.6172	0.5457	0.5282	0.5156	0.4945	0.52	0.5609	0.5725
LSTM	0.4333	0.415	0.39	0.3667	0.3776	0.4285	0.5	0.5225
SVM	0.5701	0.5079	0.484	0.45	0.4333	0.4324	0.4505	0.4885

Fig. 11 Recall rates of Xgboost, LSTM and SVM in LCM with different numbers of protocols



	100	150	200	250	300	350	450	500
Xgboost	0.675	0.6533	0.6	0.5433	0.5554	0.5657	0.6	0.6585
LSTM	0.4867	0.47	0.4517	0.4667	0.4842	0.5034	0.5357	0.5625
SVM	0.6206	0.6011	0.5817	0.5724	0.56	0.5435	0.5663	0.5836

Fig. 12 Recall rates of Xgboost, LSTM and SVM in TLM with different numbers of protocols



	100	150	200	250	300	350	450	500
Xgboost	0.6363	0.6033	0.5833	0.5616	0.5803	0.606	0.6324	0.6917
LSTM	0.5636	0.5333	0.5	0.5	0.55	0.6111	0.647	0.6721
SVM	0.655	0.6425	0.6364	0.6078	0.5713	0.5889	0.5982	0.6287

Fig. 13 Recall rates of Xgboost, LSTM and SVM in SLM with different numbers of protocols

expressed as TP/(TP + FN), where TP is the number of correctly classified secure protocols and FN is the number of misclassified ones. As shown in the three figures, all the machine learning models have quite analogous pattern of recall rate in the three dataset models. When the protocol number is more than 300, the values and trends of the nine recall rate curves are similar to the multi-classification result in Figs. 8, 9 and 10. However, as the samples are no more than 200, the values of recall rate that looks good actually have no sense. Because under this condition, the number of samples is to small and the features of secure protocols are always overlearned. As for the best performance, almost 70% secure protocols can be correctly verified in Xgboost as the size of dataset reaches maximum.

5 Conclusion

This paper proposes a machine learning-based automated analysis scheme to assist in reducing the difficulty in the future protocol analysis. Through experiments, it is proved that the features we extract and the dataset we build can indeed help us identify the attacks in the protocol. This means that the fusion of machine learning and formal analysis of the protocol is exactly a significant and feasible attempt. However, although many attack types are ignored, there are no enough samples to allow classify each attack as a category at the early stage of our research. Thus, in the future work, more protocols will be gathered to expand our dataset. And we are planning to apply the hybrid model scheme into our research.

Acknowledgements The authors would like to thank the editor and the anonymous referees for their constructive comments.

Funding This work was supported by the National Natural Science Foundation of China (Grant No. U1764263, 61872283) and the Natural Science Basic Research Plan in the Shaanxi Province of China (Grant No. 2016JM6074).

References

1. Alabdulatif A, Ma X, Nolle L (2012) A framework for cryptographic protocol analysis using linear temporal logic. In: International conference on information society, pp 525–530
2. Jurcut A, Coffey T, Dojen R (2017) A novel security protocol attack detection logic with unique fault discovery capability for freshness attacks and interleaving session attacks. IEEE Trans Dependable Secure Comput 99:1
3. Hess AV, Modersheim S (2017) Formalizing and proving a typing result for security protocols in isabelle/hol. In: Computer security foundations symposium, pp 451–463
4. He D, Chen C, Chan S, Bu J, Yang LT (2013) Security analysis and improvement of a secure and distributed reprogramming protocol for wireless sensor networks. IEEE Trans Ind Electron 60(11):5348–5354
5. Cohn-Gordon K, Cremers C, Dowling B, Garratt L, Stebila D (2017) A formal security analysis of the signal messaging protocol. In: IEEE European symposium on security and privacy (EuroS&P). IEEE, pp 451–466
6. Li T, Liu X, Qin Z, Zhang X (2009) Formal analysis for security of Otway–Rees protocol with ban logic. In: First international workshop on database technology and applications, pp 590–593
7. Burrows M, Abad M, Needham RM (1989) R.m.: a logic of authentication. Proc R Soc A Math Phys Eng Sci 426(1871):1–13

8. Liu K, Ye J, Wang Y (2012) The security analysis on Otway–Rees protocol based on ban logic. In: Fourth international conference on computational and information sciences, pp 341–344
9. Bhargavan K, Corin R, Zalinescu E (2008) Cryptographically verified implementations for tls. In: ACM conference on computer and communications security, pp 459–468
10. Chaki S, Datta A (2009) Aspier: an automated framework for verifying security protocol implementations. In: IEEE computer security foundations symposium, pp 172–185
11. Li L, Sun J, Liu Y, Sun M, Dong JS (2017) A formal specification and verification framework for timed security protocols. IEEE Trans Softw Eng 99:1
12. Pan S, Morris T, Adhikari U (2015) Developing a hybrid intrusion detection system using data mining for power systems. IEEE Trans Smart Grid 6(6):3104–3113
13. Stevanovic M, Pedersen JM (2014) An efficient flow-based botnet detection using supervised machine learning. In: International conference on computing, networking and communications, pp 797–801
14. Petrakos N, Kotzanikolaou P, Douligeris C (2012) Using strand space model to verify the privacy properties of a fair anonymous authentication scheme. In: Panhellenic conference on informatics, pp 105–110
15. Gibson-Robinson T, Kamil A, Lowe G (2015) Verifying layered security protocols. J Comput Secur 23(3):259–307
16. Boyd C, Mathuria A (2003) Protocols for authentication and key establishment. Inf Secur Cryptogr 364(1849):3215–3230
17. Clark J (1997) A survey of authentication protocol literature : version 1.0
18. Dolev D, Yao AC (1983) On the security of public key protocols. Inf Theory IEEE Trans 29(2):198–208
19. Menezes AJ, Vanstone SA, Oorschot PCV (1997) Handbook of applied cryptography. CRC Press, Boca Raton
20. Haley C, Laney R, Moffett J, Nuseibeh B (2008) Security requirements engineering: a framework for representation and analysis. IEEE Trans Softw Eng 34(1):133–153
21. Basin D, Cremers C, Miyazaki K, Radomirovic S, Watanabe D (2015) Improving the security of cryptographic protocol standards. IEEE Secur Priv 13(3):24–31
22. Zhang W, Lin Y, Xiao S, Wu J, Zhou S (2016) Privacy preserving ranked multi-keyword search for multiple data owners in cloud computing. IEEE Trans Comput 65(5):1566–1577
23. Avalle M, Pironti A, Sisto R (2014) Formal verification of security protocol implementations: a survey. Formal Asp Comput 26(1):99–123
24. Dong B, Wang X (2016) Comparison deep learning method to traditional methods using for network intrusion detection. In: IEEE international conference on communication software and networks, pp 581–585
25. Lowe G, Roscoe B (1996) Using CSP to detect errors in the TMN protocol. Softw Eng IEEE Trans 23(10):659–669
26. Du M, Li F, Zheng G, Srikumar V (2017) DeepLog: anomaly detection and diagnosis from system logs through deep learning. In: ACM Sigsac conference on computer and communications security, pp 1285–1298
27. Xiao B, Wang K, Bi X, Li W, Han J (2018) 2D-LBP: an enhanced local binary feature for texture image classification. IEEE Trans Circuits Syst Video Technol. https://doi.org/10.1109/tcsvt.2018.2869841
28. Ma Z, Wang X, Ma R, Wang Z, Ma J (2018) Integrating gaze tracking and head-motion prediction for mobile device authentication: a proof of concept. Sensors 18(9):2894