Hindawi
Security and Communication Networks
Volume 2022, Article ID 8104884, 19 pages
https://doi.org/10.1155/2022/8104884

Research Article

Formal Security Analysis and Improvement Based on LonTalk Authentication Protocol

Tao Feng and **Yi Wu**

School of Computer and Communication, Lanzhou University of Technology, Lanzhou 730050, China

Correspondence should be addressed to Tao Feng; fengt@lut.cn

Received 17 April 2022; Accepted 24 June 2022; Published 12 July 2022

Academic Editor: Irshad Azeem

Security analysis of security protocol can be used to ensure communication security in the network. The process of security protocol analysis using the formal analysis method is simple and standardized, which is a research hotspot in the field of information security. In this study, a formal analysis method based on colored Petri net theory and Dolev-Yao attacker model is adopted to analyze LonTalk authentication protocol, and three types of attackable vulnerabilities including replay, tamper, and spoofing are found in LonTalk authentication protocol; thus, a secure LonTalk-SA authentication protocol is proposed. The LonTalk-SA authentication protocol was added with a trusted third-party server, which authenticates the identity of the sender and receiver and generates session keys through XOR operations on random numbers. The formal analysis of the new scheme shows that the new scheme can effectively resist three types of attacks, provide bidirectional authentication of communication nodes, and ensure the confidentiality, integrity, and authentication of messages during transmission, thus improving the security of protocols.

1. Introduction

Building automation system is a key part of smart buildings [1, 2], as it can highly achieve automatization and intelligent centralized management for all mechanical and electrical facilities and energy equipment in smart buildings. The combination of internet and traditional bus improves the efficiency of traditional bus; however, it also introduces the security problems existing on internet into building automation system [3–5], for example, attackers can easily tamper with, replay, eavesdrop, and other attacks on the data transmitted in an industrial control system.

Under the development of technology, there are increasingly articles pointing out that the LonTalk authentication protocol in building automation system has many vulnerabilities [6]. Literature [7–12] points out that the LonTalk authentication protocol has the following security vulnerabilities: (1) this authentication protocol only supports verifying the identity of the sender and cannot check the identity of the receiver. Only the sender can initiate the challenge-answer request; however, the receiver cannot, so the protocol can only carry out one-way authentication. (2) The key used for identity authentication between devices is only 48 bits, which cannot avoid brute-force cracking attacks. (3) Only part of the data segment is used for hash calculation. Address information and other header information cannot be protected. (4) The data transfer in clear text will lead to the leakage of information. (5) The sender must always authenticate with the receiver, so the communication session cannot be established. Literature [13, 14] points out that the LonTalk protocol is vulnerable to denial-of-service (DoS) attacks. So this will lead to huge performance consumption of nodes. Literature [15] proposes to use SHA-1 and AES encryption to encrypt data to ensure confidentiality and integrity. It will provide a key distribution mechanism when using the advanced Needham-Schroeder protocol. The sender device does not authenticate the third-party server and cannot guarantee the authenticity of the feedback message. Literature [16, 17] pointed out that the choice of the encryption algorithm is constrained by

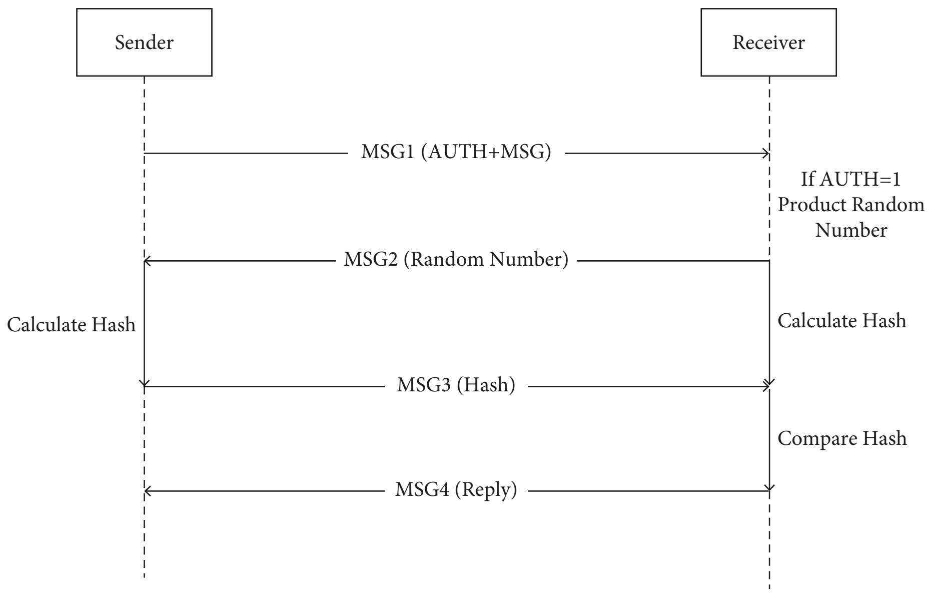

Figure 1: LonTalk authentication protocol.

embedded architecture, and the public key encryption scheme is limited by chip processing capability in low-end embedded system.

To sum up, the existing research work on LonTalk authentication protocol security mostly points out the lack of security of the protocol and puts forward some suggestions for protocol improvement, or focuses on the realization of its own security functions. At present, there is no research on formal analysis of LonTalk authentication protocol or introducing an attacker model to analyze the security of the protocol.

This study takes the LonTalk authentication protocol as the research object, takes the colored Petri net and Dolev-Yao attacker model as the basic theory, uses CPN Tools [18] to evaluate the security of the protocol, puts forward a new improved scheme, and verifies the security of the new scheme. The verification results show that the LonTalk-SA protocol has higher security.

2. Preliminary Knowledge

2.1. LonTalk Authentication Protocol. When the LonTalk authentication protocol is enabled, the 48-bit preshared key is used for identity authentication, and the sender and the receiver have the same key. The LonTalk authentication protocol model is shown in Figure 1. AUTH represents the identity authentication bit, MSG represents the message content, and Random Number represents the random number calculated by the sender. Hash indicates the hash value calculated by the sender based on the message and random number. Reply indicates that the receiver sends a reply to the sender about the authentication result.

The authentication mode process is as follows:

(1) The sender sends a message to the receiver that contains an authentication bit. If the bit is 1, the message requires identity authentications.

(2) The receiver responds with a random number and saves the hash value that combines the random number with the original message through a hash function.

(3) After receiving the random number, the sender calculates the hash value using the same method as the receiver and then sends the hash value to the receiver.

(4) After receiving the hash values from the sender, the receiver compares the two hash values. If the hash values are the same, the receiver successfully authenticates the sender.

2.2. Colored Petri Net Theory and CPN Modeling Tool. The CPN [19] is a graphical language that has strong advantages in modeling and verifying concurrent, distributed systems. The CPN Tools supports the hierarchical CPN models with and without time and uses good interpersonal interface technology to design the user graphical interface, which can not only edit, simulate, and analyze colored Petri nets but also support temporal CPN and hierarchical CPN. With the help of CPN Tools, users can easily model, simulate, and analyze parallel systems as well [20].

The CPN has certain advantages when compared with other popular automatic protocol security verification tools. The limited attack path set calculated by ProVerif [21] is far smaller than the attack path set extracted by the CPN-based methods. Scyther [22] tried to use the same method to provide state-space analysis. Although some attack paths could be found in this way, comprehensive security analysis still could not be achieved. Tamarin Prover [23] has high requirements for professional knowledge of modelers and is not so simple and intuitive compared with CPN. In addition, the highly free modeling process of CPN and the realization of different modeling and analysis methods for different

Table 1: Color set definition.

The key elements	Color set definition
AUTH	Colset AUTH = INT;
MSG	Colset MSG = STRING;
RN	Colset RN = INT;
PK	Colset PK = STRING;
RPDU	Colset RPDU = product AUTH * MSG;
CAPDU	Colset RAPD = product PK * CONTENT;
APDU	Colset APDU = REPLY;

protocols are important reasons for using CPN as the formal analysis of protocols.

2.3. Dolev-Yao Attacker Model. Dolev and Yao proposed a mathematical model for verifying public key cryptographic protocols, namely, the Dolev-Yao attacker model [24], which formally defined the behavior of attackers. Based on the assumption that the cryptographic system is "perfect," discussing the security properties of the protocol itself can help researchers focus on the intrinsic security properties of the protocol instead of discussing the security of the cryptographic algorithm.

The Dolev-Yao attacker model is introduced in the formal security analysis process of the protocol, which can eavesdrop, intercept, replay, and tamper with the messages exchanged between real entities during the operation of the protocol, and encrypt, decrypt, split, and combine the original messages and forge message content.

3. LonTalk Authentication Protocol HCPN Modeling

3.1. LonTalk Color Set Definition of Authentication Protocol Messages. The color set is established for the four messages exchanged between the receiver and the sender. First, AUTH, MSG, RN, REPLY, and PK are metainformation, and other information is constructed on the basis of meta information. AUTH represents the identity authentication bit, MSG represents the original data information, RN represents the random number generated by the receiver, and PK represents the key of the device to calculate the hash value. RPDU indicates authentication request packets consisting of AUTH and MSG, which are sent from the sender to the receiver. CAPDU is a random number sent by the receiver to the sender. RAPDU represents the hash value calculated by combining PK and MS. The APDU type indicates whether the receiver sends a message to the sender for identity authentication. The specific color set definition is shown in Table 1.

3.2. Formal Modeling of LonTalk Authentication Protocol. This study will use CPN Tools for formal modeling of the LonTalk authentication protocol. In the top-down sequence, the protocol top-level model is established first, and then, the protocol submodule is established. Ellipses represent places, rectangles represent transitions, and double-line transitions refer to substitution transitions, which include more detailed submodules below.

The top-level model of the LonTalk authentication protocol consists of 5 transitions and 10 places. The process of sending the first packet from the sender to the receiver is represented by the substitution transition Connection. The process in which the receiver receives the packet sent from the sender and replies to the sender and calculates the hash value that is represented by the substitution transition Production. The sender's process of receiving random numbers and calculating hash is known as the substitution transition Computation. Finally, the receiving end compares the two hash values, and the process of sending the authentication result back to the sender is represented by the substitution transition Comparation, as shown in Figure 2 for details.

Five detailed submodules are explained as follows. Figure 3 describes the internal model of the substitution transition Connection. The transition Combination first combines the authentication bit ID and message content MSG into an RPDU message, which is sent to the transition Send_MSG1, and finally sent to the receiver via the place send_RPDU. The place rec_CAPDU receives the information from the receiver, and then combines RPDU and RN into content information through the transition Combination1, which is sent to the model for calculating hash values.

Figure 4 shows the internal model of substitution transition Production. First, the place rec_RPDU will send the received information to the transition Division where will split the received messages. The transition Judge will judge whether the received ID is correct. If the ID is incorrect, it will send the received information to the place Discard. If the ID is correct, it is sent to the transition COMB2. At this time, the random number generated by the place will be sent to the place send_CAPDU through the transition Combination2 and sent to the sender through the place send_CAPDU. The transition Combination2 also combines the message RPDU and the random number RN into a content. Content is encrypted with key PK through the transition Combination4 and sent to the transition Combination5 after encryption. The transition Combination5 computes the hash value and finally sends the hash value to the place Compute.

Figure 5 shows the internal model of substitution transition Computation. The place Content receives the message and sends it to the transition Combination7. The transition Combination7 sends messages to the place Content1 and the place Content2. First, the transition Combiation6 will encrypt the value sent by the place Content with key PK and send it to the transition CC_RAPD. The transition CC_RAPD calculates the hash value of the RAPD packet, sends the hash value to the place C_RAPDU, and finally, sends it to the receiver through the place send_RAPDU. The place rec_APDI sends the received messages to the transition Store, which stores the received messages to the place Reply.

Figure 6 shows the internal mode for the substitution transitions Comparation. The place rec_RAPDU represents receiving a hash value from the sender, and then, the transition Compare1 compares the received hash value with the hash value calculated by the receiver, and if the hash

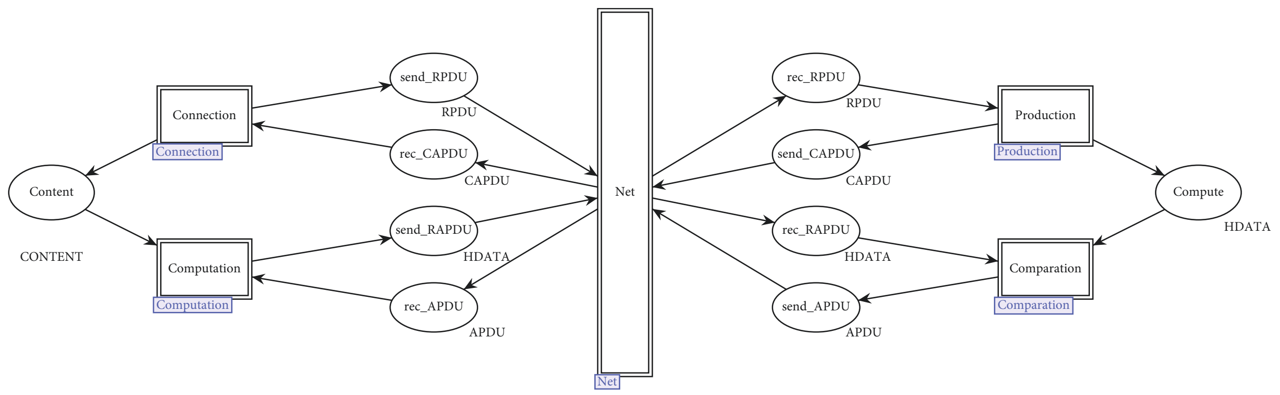

Figure 2: CPN top-level model of LonTalk authentication protocol.

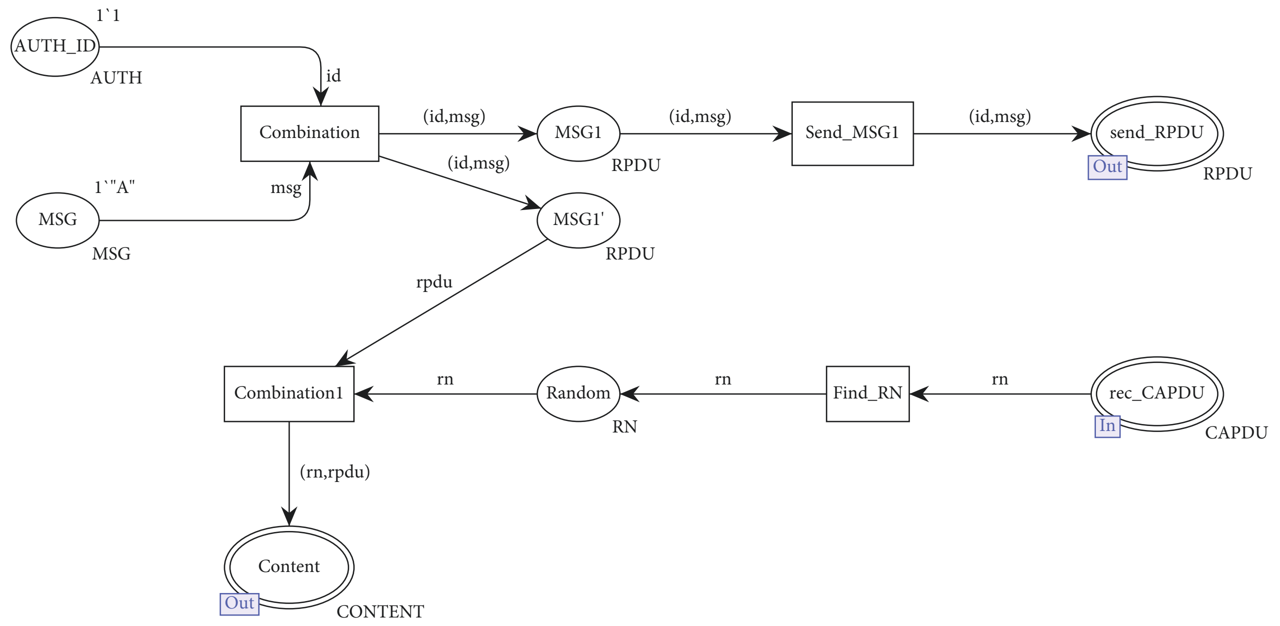

Figure 3: Substitution transition Connection internal model.

value is different, it is sent to the place Discard. If the hash values are the same, a success message will be sent to the sender via the place send_APDU, indicating that the identity authentication of the sender is successful.

Figure 7 shows the internal model of the substitution transition Net. The transition Transmit_RPDU indicates that the sender sends an identity message to the receiver. The transition Transmit_CAPDU indicates that the receiver sends a random number to the sender. The transition Transmit_RAPDU indicates that the sender sends the calculated hash value to the receiver. The transition Transmit_APDU indicates that the receiver sends the result of hash value comparison to the sender.

3.3. *LonTalk Model Consistency Analysis.*

The CPN model of the LonTalk authentication protocol is verified by using the state-space analysis tool. By analyzing the results of state space in Table 2, it can be found that the number of nodes and directed arcs in state space is the same as that of strongly connected nodes and strongly connected arcs, indicating that the original model established by us does not have the condition that leads to state cycles. All state nodes are reachable; the dead node count is 1, indicating that all requests are executed by the slave. There are two dead transitions DiscardID and Error_REPLY. The transition DiscardID is used to indicate that the identity authorization bit cannot activate the authentication service. The transition Error_REPLY indicates that the hash value on the receiving end is incorrect. These two transitions are dead transitions, indicating that the model does not have the above two situations, consistent with the expected, indicating that the protocol can run normally.

3.4. *Security Evaluation of LonTalk Authentication Protocol Based on Attacker Model.*

Replay, spoofing, and tampering attacks are introduced to the network transmission level of the original model. The places and transitions marked in blue in Figure 8 simulate replay attacks. The transition TA

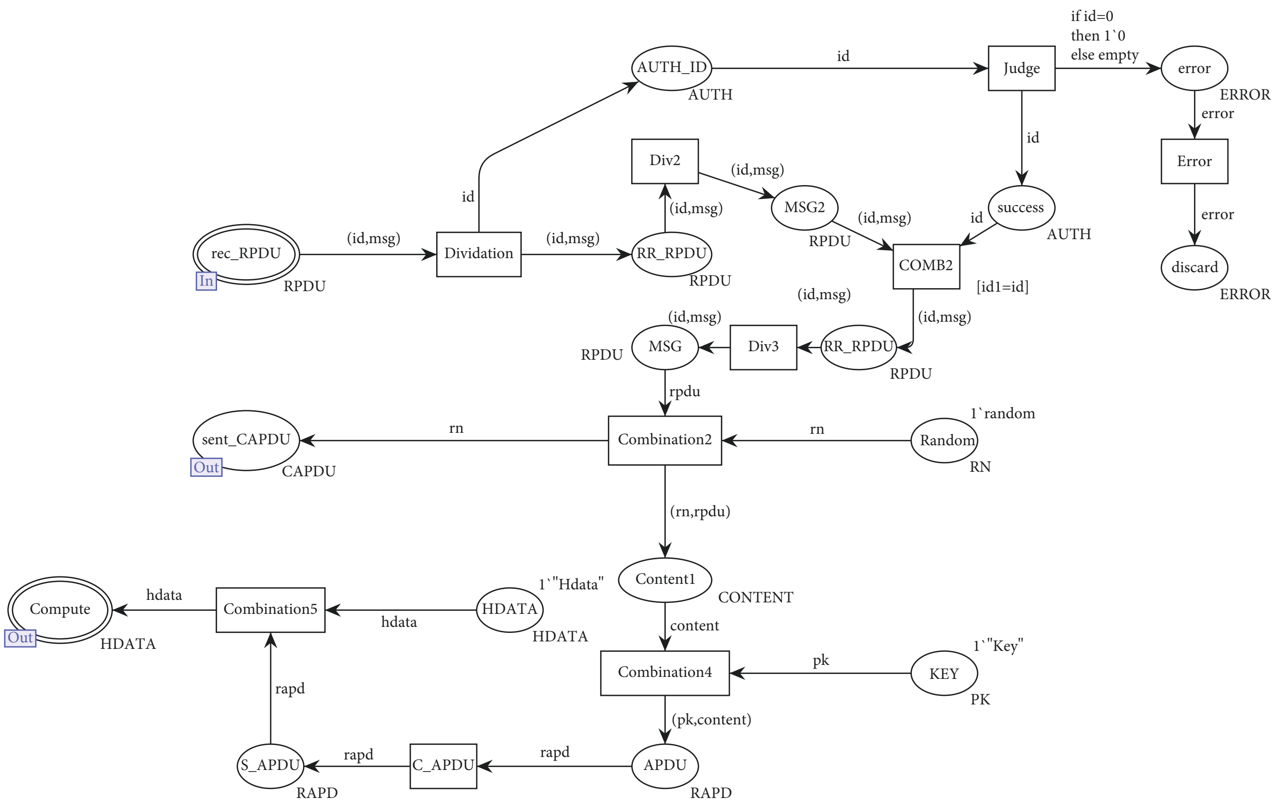

FIGURE 4: Substitution transition Production internal model.

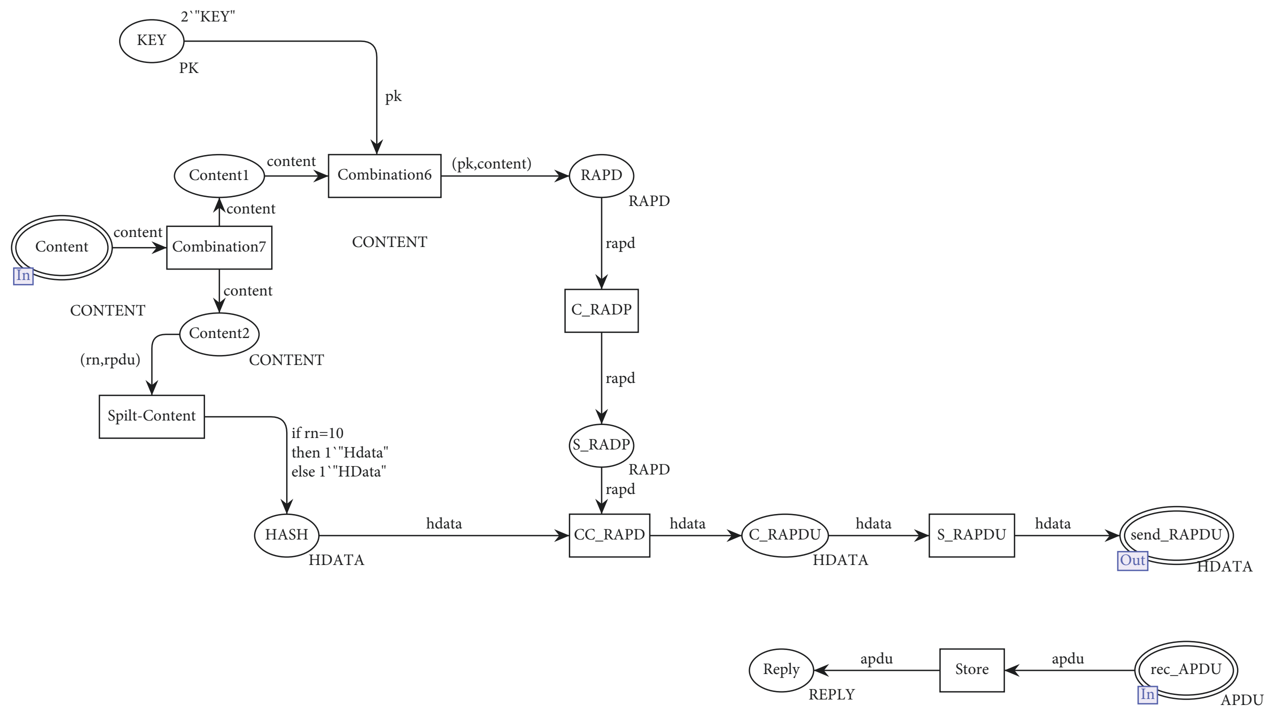

FIGURE 5: Substitution transition Computation internal model.

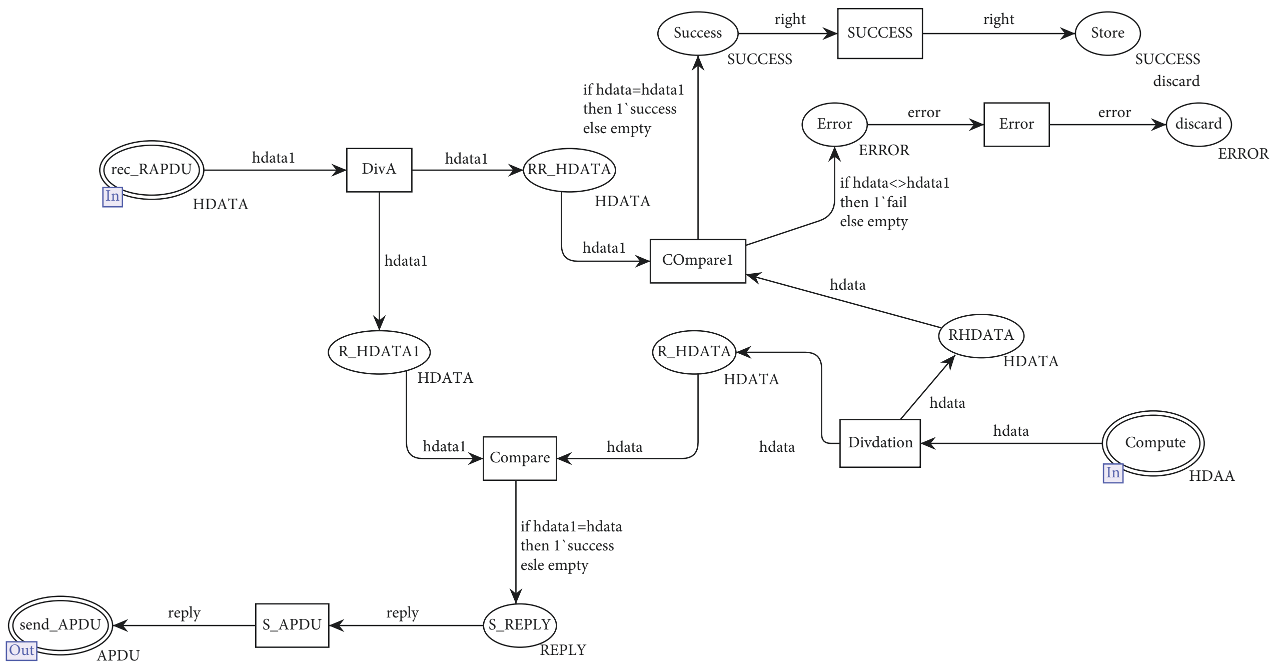

Figure 6: Substitution transition Comparation internal model.

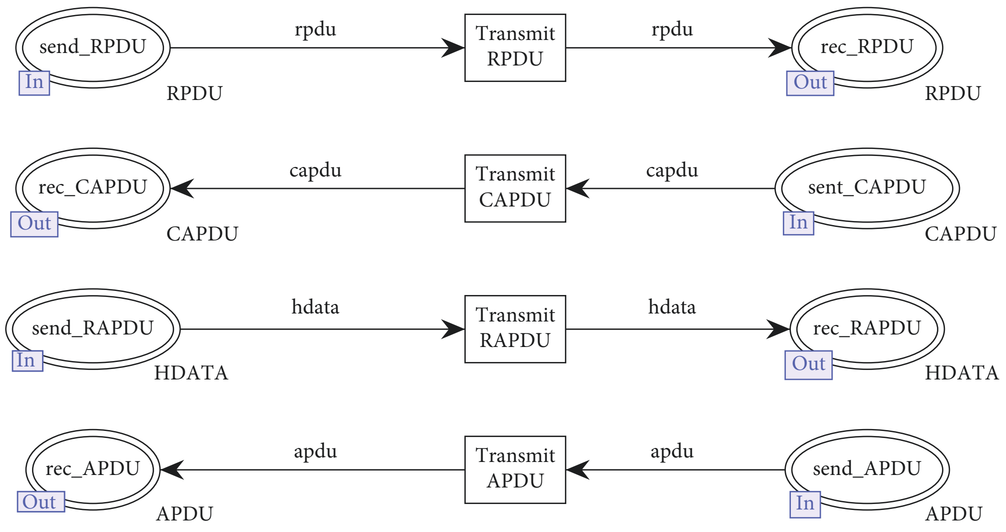

Figure 7: Substitution transition Net internal model.

Table 2: State-space analysis of original LonTalk protocol model.

Type	Number	Name
State-space node	73	
State-space arc	116	
SCC graph node	73	
SCC graph arc	116	
Live transition	0	
Dead marking	1	
Dead transition	2	Error_REPLY/Discard

intercepts the information during the first transmission of the protocol. The place Distri can store decomposed and undecomposed information, and the transition TC indicates that an attacker after decomposition rules will form the atomic information saved to the place by the place P3. The transition TH saves the undecipherable information in the place P4, and the transition TD means that the attacker synthesizes atomic messages, saves the synthesized messages in the place P5, and uses concurrency control the place SP to limit the synthesis rules to the transition TD. The transition TF synthesizes the attacker's message and sends it to the channel port place. The expression on the red marked arc in Figure 8 simulates a tamper attack on the transition place. TAttack is introduced into the expression, and attacks are launched through the place Hash_Attack. The pink part in Figure 8 simulates spoofing attack, including all transitions in the network transmissions process Transmit RPDU, Transmit CAPDU, Transmit RAPDU, and Transmit APDU.

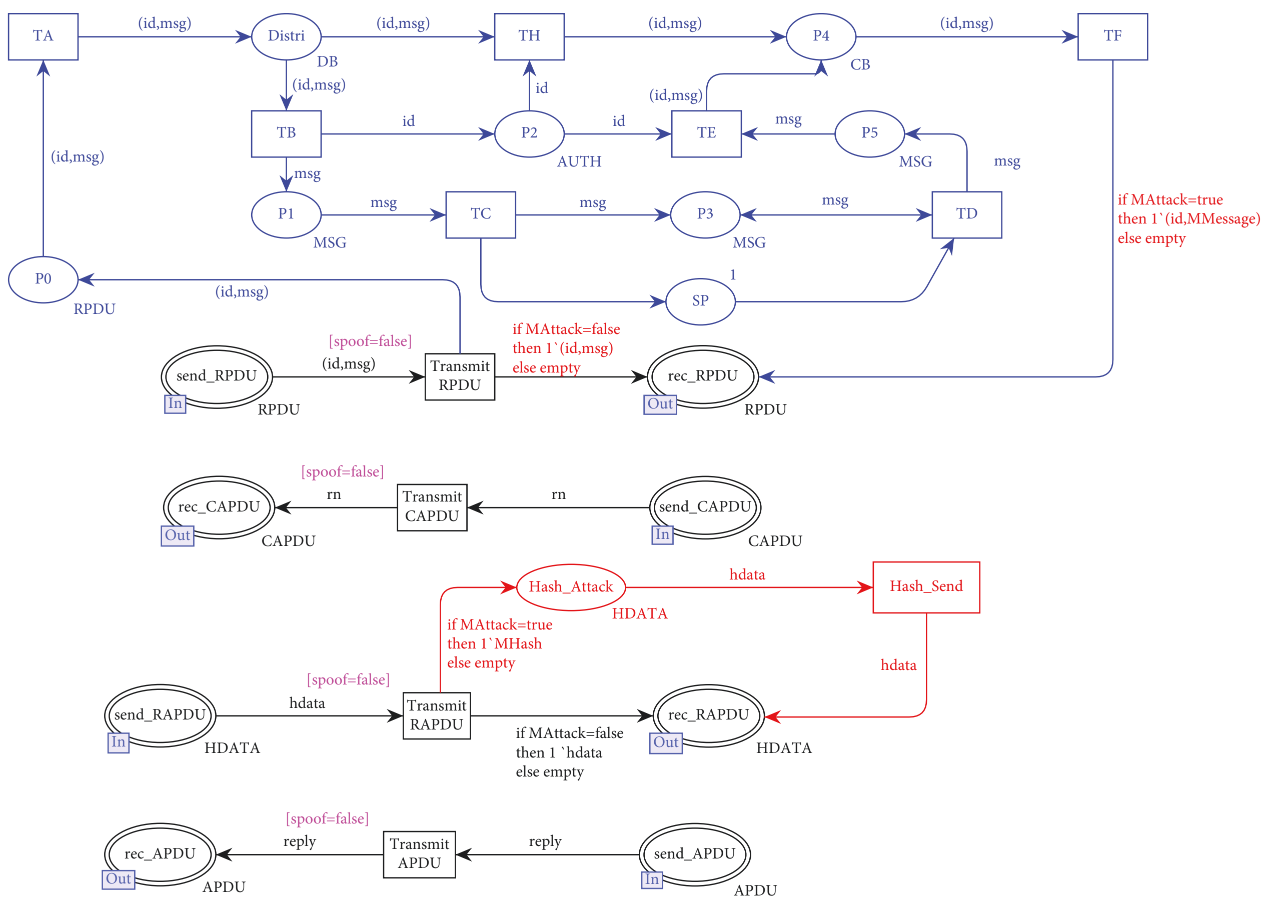

Figure 8: Attacker model of LonTalk protocol authentication.

3.5. Analysis of LonTalk Authentication Protocol Security Attributes. From the state-space report of the attacker model shown in Table 3, the number of state-space nodes, directed arcs, and strongly connected nodes and strongly connected arcs is the same, indicating that all state nodes in the attacker model of this protocol are reachable. When the attacker model was introduced, the number of nodes and arcs in its state space increased less than the original model, indicating that the state space was not too large or exploded after the attacker model was introduced.

By comparing the original model with the state space after adding the attack model, it is found that the number of dead nodes and dead transitions does not change. After capturing the first message sent by the sender, the attacker modifies the MSG in the message because the message is transmitted in plaintext. The modified message is sent to the receiver. After receiving the message, the receiver returns a random number to the sender, and the attacker eavesdrops on this random number. When the sender receives the random number, it is combined with the initial message to calculate the hash value and sends the calculated hash value to the receiver. The attacker intercepts the message and sends its calculated hash value to the receiver. After receiving the message, the receiver compares the hash values and finds that the result is the same. The identity authentication succeeds on the receiver, and the receiver sends the message.

Table 3: Comparison of model state space.

Type	Original model	Attack model
Status space node	73	203
Status space arc	116	449
SCC graph node	73	203
SCC graph arc	116	449
Dead transitions	2	2
Dead markings	1	1

After receiving the successful authentication message, the receiver confirms that it is successfully authenticated. The subsequent messages can be eavesdropped by the attacker. Through the comparison of the state space, it can be found that the attacker can effectively launch an effective attack on the LonTalk authentication protocol, which reflects the existence of replay, tampering, and spoofing vulnerabilities in the protocol, and the confidentiality, integrity, and validity of data in the process of message transmission cannot be guaranteed.

4. New LonTalk Authentication Protocol Scheme

4.1. LonTalk-SA Authentication Protocol Scheme. Aiming at tampering, spoofing, and replay vulnerabilities in the LonTalk authentication protocol, the LonTalk-SA

Table 4: Notation table.

Symbol	Definition
A, B, S	Sender A, receiver B, and server S
ID_I	The unique identity of device I
X, N_A	Random number generated by the sender A
Y, N_B	Random number generated by the receiver B
K_{AS}	Secret key between sender and server
K_{BS}	Secret key between receiver and server
K	Session key between sender and receiver
TS_1, TS_2	The current timestamp
E(key, M)	The message M is encrypted using the key
H(.)	One-way hash function
$\|\|$	Concatenation operation

authentication protocol is proposed in this study. Neuron chip is the core of the LonTalk authentication protocol [25]. Each Neuron chip contains three 8-bit embedded CPUs, onboarded memory, and 11 general I/O pins. On the premise that the chip and memory performance are not high, a trusted third-party server is introduced for authentication. Before the device performs authentication, the server can send the master key, which is used for communication between the server and the device. Because hash functions and XOR operations do not require much computational performance [26], they are suitable for chips like Neuron that have limited storage, processing, and transmission capabilities. Using the hash function to calculate the hash value of messages can ensure the integrity of transmitted messages and reduce chip computing resource consumption. Through key negotiation between devices, two random numbers are used to calculate the session key.

4.2. LonTalk-SA Authentication Protocol Communication Process. The improved protocol communication process of the message flow diagram is as follows, and the specific symbols are shown in Table 4 as follows:

(1) When sender A and receiver B perform identity authentication, sender A generates random numbers X and N_A, A encrypts two random numbers, ID_A and ID_B, with the master key K_{AS}. ID_A is sent to the server S along with the encrypted packets.

(2) When the server receives the message from A, it uses the primary key K_{AS} to decrypt it. After decryption, the server uses K_{BS} to encrypt the two random numbers which are sent by A and adds the timestamp TS_1 and N_A the encrypted packet which will send sent to B. Finally, the server sends an encrypted message to A.

(3) After receiving the message from the server, A uses the primary key K_{AS} to decrypt, and no change was found in N_A, after decryption with the master key K_{AS}. Then, A sends the packets, which were sent from the server to B together with ID_A and ID_B to B.

(4) After receiving the message, B decrypts it with the master key K_{BS}, and obtains two random numbers and time stamps generated by the server. Then, the timestamps are used to check whether the message is under replay attack. In this case, B also generates two random numbers Y and N_B. B combines ID_A, ID_B, Y, N_B, and N_A, encrypts them with the master key K_{BS}, and finally sends the ID_B and encrypted packets to the server.

(5) The server receives the message from B and decrypts it with the key K_{BS}. The server encrypts Y, N_B, and N_A messages with the key K_{AS} and adds the timestamp TS_2. Finally, K_{BS} is used to encrypt N_B and the encrypted message sent to A and sends it to B.

(6) After receiving the message from the server, B uses K_{BS} to decrypt the message to check whether the N_B is tampered with. Then, B sends the rest of the data to A.

(7) After receiving the message from B, A decrypts it with the master key K_{AS}, obtains A random number Y and the timestamp, and checks whether the timestamp exceeds the preset time range. A evaluates the hash from N_A and N_B. Then, N_A, N_B, and hash values are combined and sent to B using the session key K for encryption.

(8) B also performs XOR operations on X and Y to generate session key K. B uses session key K to decrypt the message, recalculates the hash values of N_A and N_B, and compares them with the hash values of the message sent by A. If the hash values are the same, B successfully authenticates A. In addition, B calculates the hash values of N_B and N_A, encrypts the hash values with the session key K, and sends the hash values to A.

(9) After receiving the message, A uses session key K to decrypt the message, calculates the hash values of N_B and N_A, and compares the calculated hash values with those sent by user B. If the hash values are the same, A successfully authenticates B. The specific process is shown in Figure 9.

5. LonTalk-SA Authentication Protocol Formal Analysis

5.1. LonTalk-SA Authentication Protocol HCPN Model. The CPN modeling is carried out for the LonTalk-SA authentication protocol. The top-level CPN model of the LonTalk-SA authentication protocol is shown in Figure 10. The top-level model simulates the entire session process of the protocol, including the protocol communicator, communication network, and packet transmission. The substitution transitions A and B represent two communication parties, the substitution transition Server represents the trusted third-party server, and the substitution transition Net represents the communication network.

The mid-level model of the LonTalk-SA authentication protocol consists of 8 substitution transitions and 19 places. The process by which A sends an authentication request to Server is represented by the substitution transition A_To_Server. The process by which the server responds to A request sent by A is represented by the substitution transition RequestA. The process of B sending an authentication

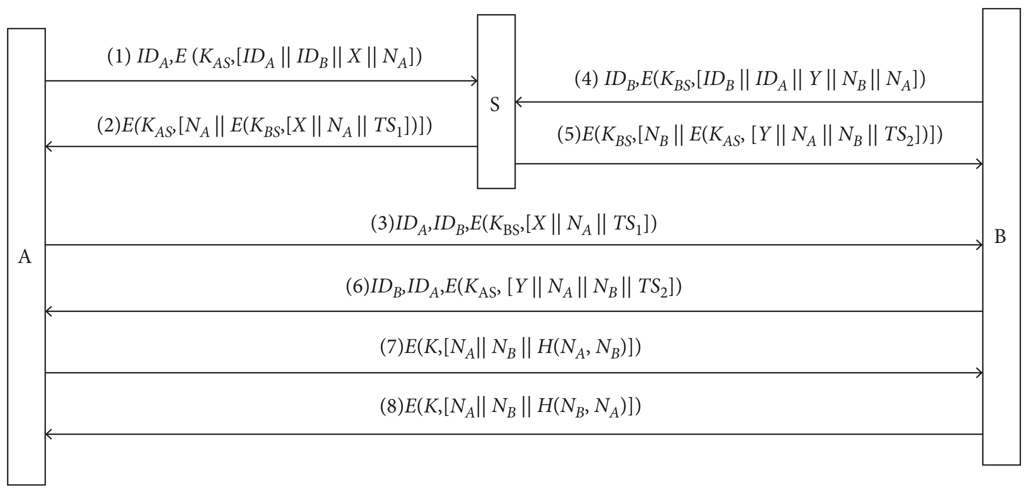
(1) $ID_A, E(K_{AS}, [ID_A \| ID_B \| X \| N_A])$
(2)$E(K_{AS}, [N_A \| E(K_{BS}, [X \| N_A \| TS_1])])$
(4) $ID_B, E(K_{BS}, [ID_B \| ID_A \| Y \| N_B \| N_A])$
(5)$E(K_{BS}, [N_B \| E(K_{AS}, [Y \| N_A \| N_B \| TS_2])])$
S
(3)$ID_A, ID_B, E(K_{BS}, [X \| N_A \| TS_1])$
(6)$ID_B, ID_A, E(K_{AS}, [Y \| N_A \| N_B \| TS_2])$
(7)$E(K, [N_A \| N_B \| H(N_A, N_B)])$
(8)$E(K, [N_A \| N_B \| H(N_B, N_A)])$
A
B

Figure 9: LonTalk-SA authentication message flow model.

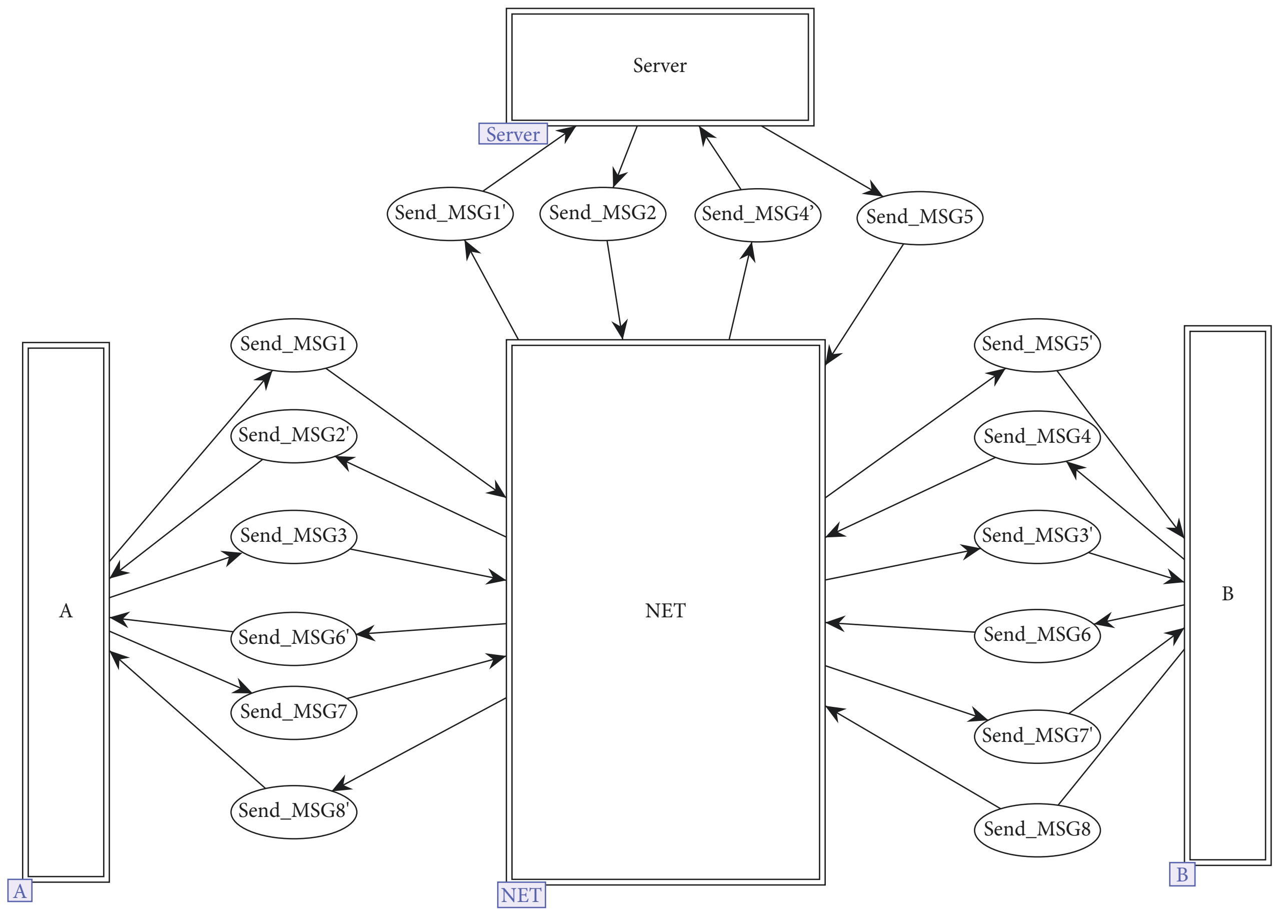

Figure 10: LonTalk-SA top-level model.

request to the server is represented by the substitution transition B_To_Server, and the process of Server replying to the request sent by B is represented by the substitution transition RequestB. The process of A sending encrypted packets to B and obtaining random numbers is constituted by the substitution transition A_To_B. The process of B sending encrypted packets is represented by the substitution transition B_To_A. The process by which A calculates the session key and hash is represented by the substitution transition A_HashTo_B. The process by which B calculates

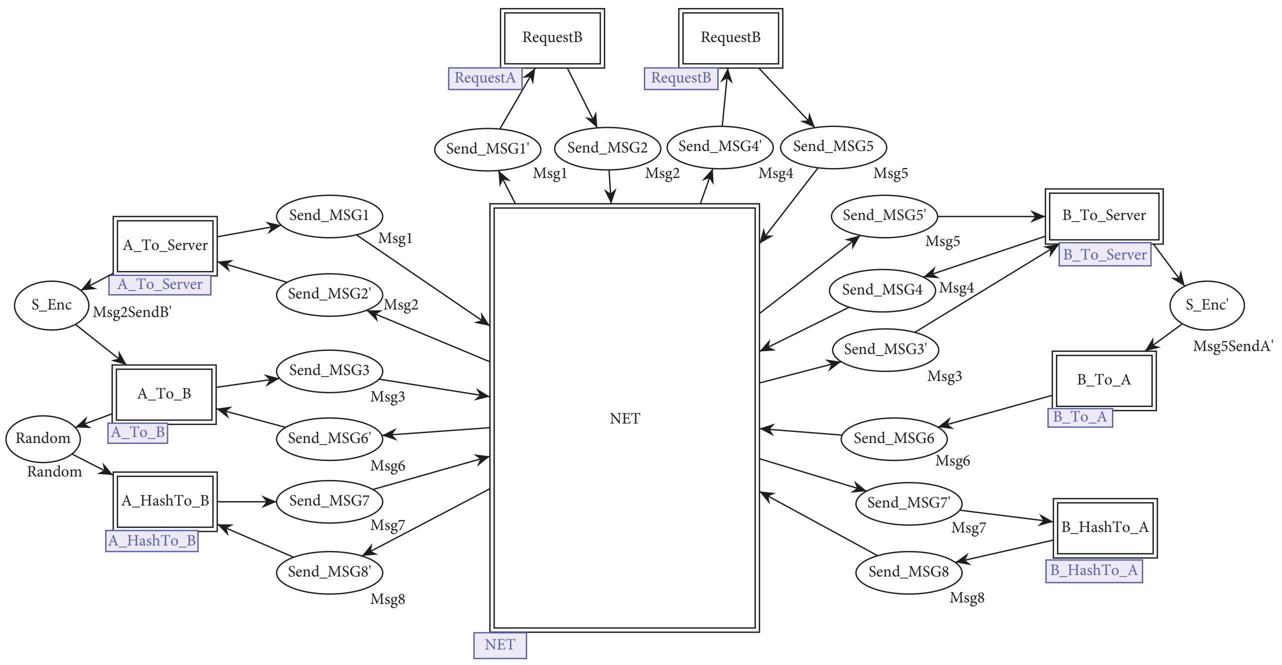

Figure 11: LonTalk-SA authentication protocol mid-level model.

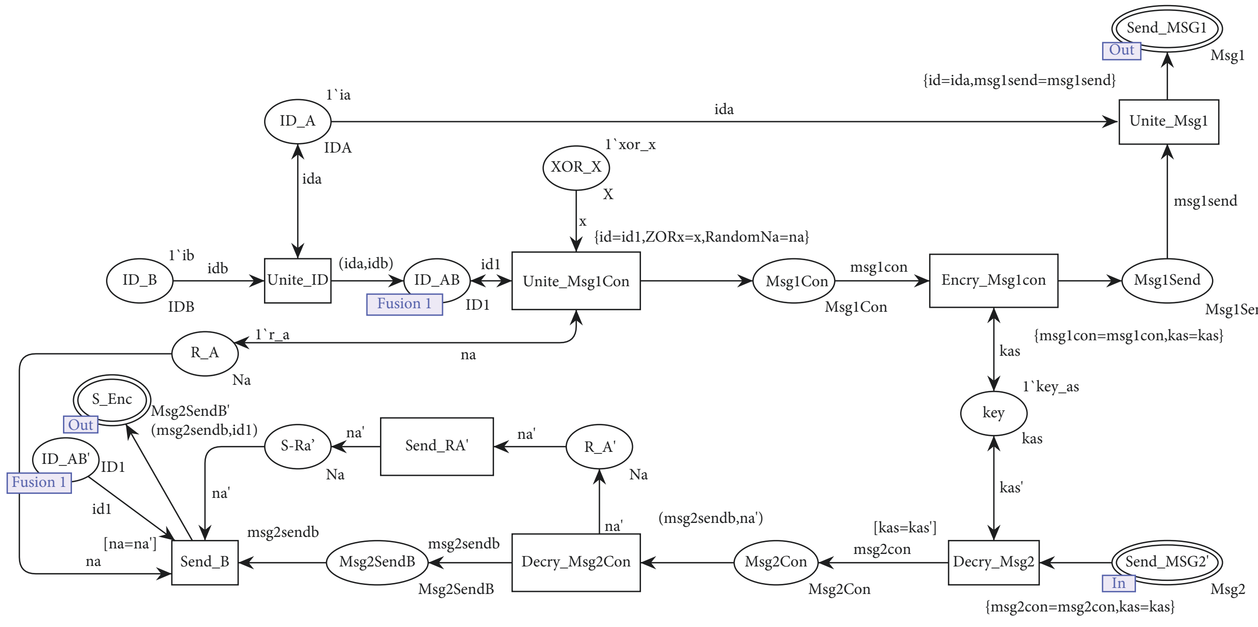

Figure 12: Substitution transition A_to_Server internal model.

the session key and hash is represented by the substitution transition B_HashTo_A. The LonTalk-SA middle-layer model is shown in Figure 11.

Figure 12 details the internal model of the substitution transition A_To_Server. Sender A combines ID_A and ID_B with the transition Unite_ID and sends it to the place ID_AB. The transition Unite_Msg1Con combines ID_A and ID_B and the two random numbers generated by A and sends it to the transition Encry_Msg1Con. The transition Encry_Msg1Con encrypts the information using the master key between A and *Server* and sends it to the transition Unite_Msg1. The transition Unite_Msg1 finally combines A identity information with the message sent by the place S_MSG1 to the place Send_MSG1. When the place Send_MSG2 receives the message, it sends it to the transition Decry_Msg2 and decrypts it with the key K_{AS}. The transition Decry_Msg2Con sends a random number in the received message to the place R_A'. The transition Send_B uses a guard function to decide

```
(1) Compute ida, idb, $x$ and na
(2) Compute id1 = Combine (ida, idb),
    Compute msg1con = Combine (x, id1)
(3) Compute key = kas, msg1send = Encrypt (msg1con, kas)
(4) Send msg1 = {ida, msg1send} to Server
```

Algorithm 1: Send_Msg1().

```
(1) Receive Msg2, where Msg2 = Encrypt (msg2con, kas)
(2) Decrypt Msg2 using kas'.
    if (kas = = kas') then
       msg2con = Decrypt (Msg2, kas')
    else return Fail
(3) na, msg2sendb = Split (msg2con)
(4) if (na = = na') then
    send msg2sendb' = {id1,msg2sendb}
    else return Fail
```

Algorithm 2: Receive_Msg2().

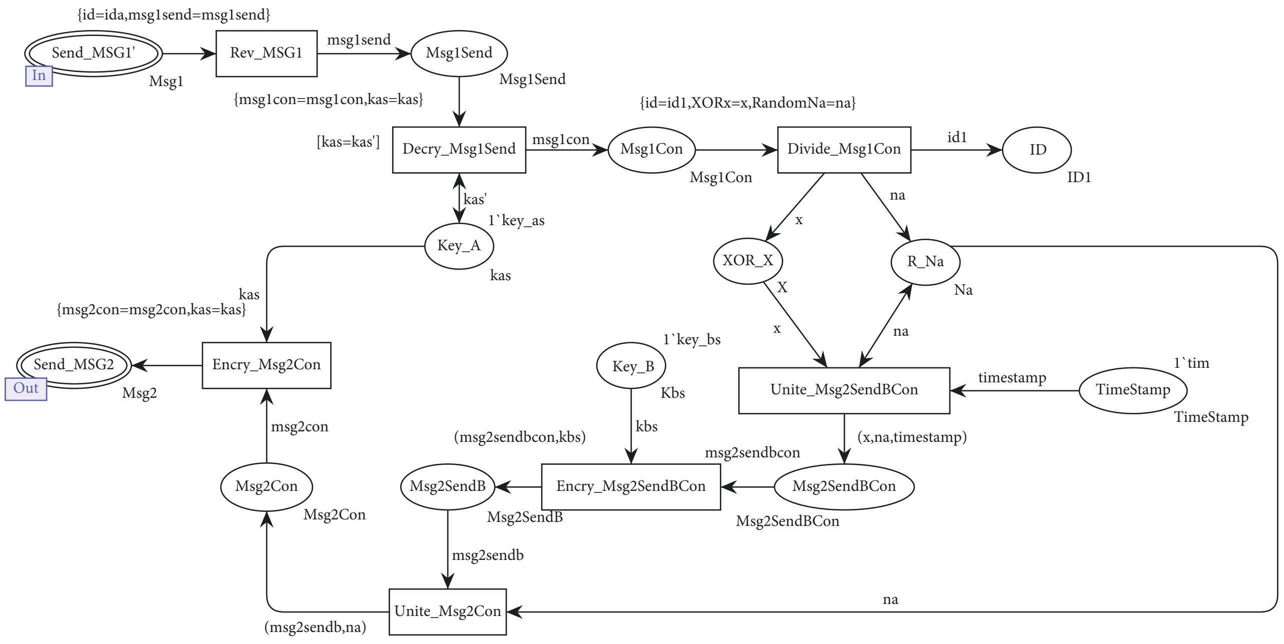

Figure 13: Substitution transition RequestA internal model.

whether to send messages from the place Msg2SendB to the place S_Enc. The pseudocode of the substitution transition A_To_Server is shown in Algorithms 1 and 2.

Figure 13 details the internal model of the substitution transition RequestA. The transition Decry_Msg1Send sends the decrypted results to the transition Divide_Msg1Con with the key K_{AS}. The transition Divide_Msg1Con splits the received message and sends the decomposed X and N_A to transition Unite_Msg2SendBCon. The transition Encry_Msg 2SendBCon encrypts the message with the key K_{BS} and sends the encrypted message to the transition Unite_Msg2Con. Finally, the transition Encry_Msg2Con encrypts the received message with the key K_{AS} and sends it to the place Send_MSG2.

Figure 14 details the internal model of the substitution transition A_To_B. The place Send_MSG6' receives the message and sends it to the transition Divide_Msg6. The transition Divide_Msg6 splits the message into id and msg5senda, id is sent to the place ID_BA, and msg5senda is sent to the place MSg5senda. The transition Divide_ID2 saves id to the place ID_A and the place ID_B, respectively. After the transition, Decry_Msg5SendACon receives the message from the place Msg5SendA, decrypts the message using the key K_{AS}, and sends the decrypted data to the

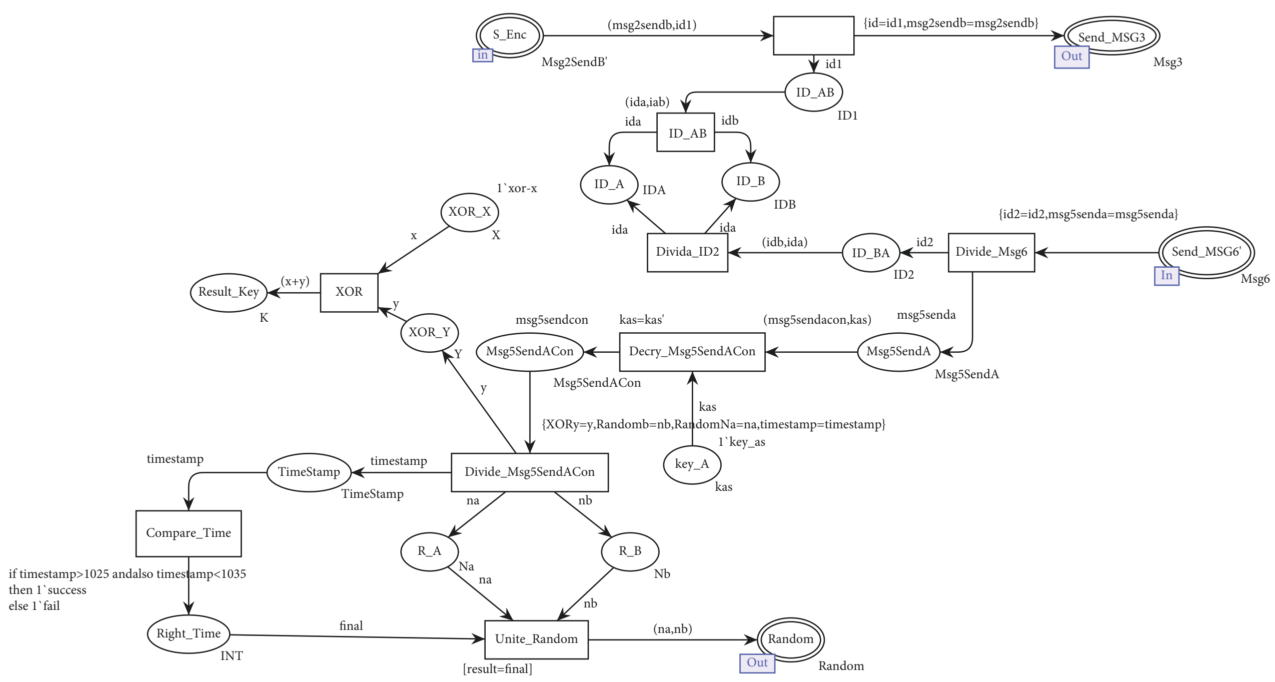

FIGURE 14: Substitution transition A_to_B internal model.

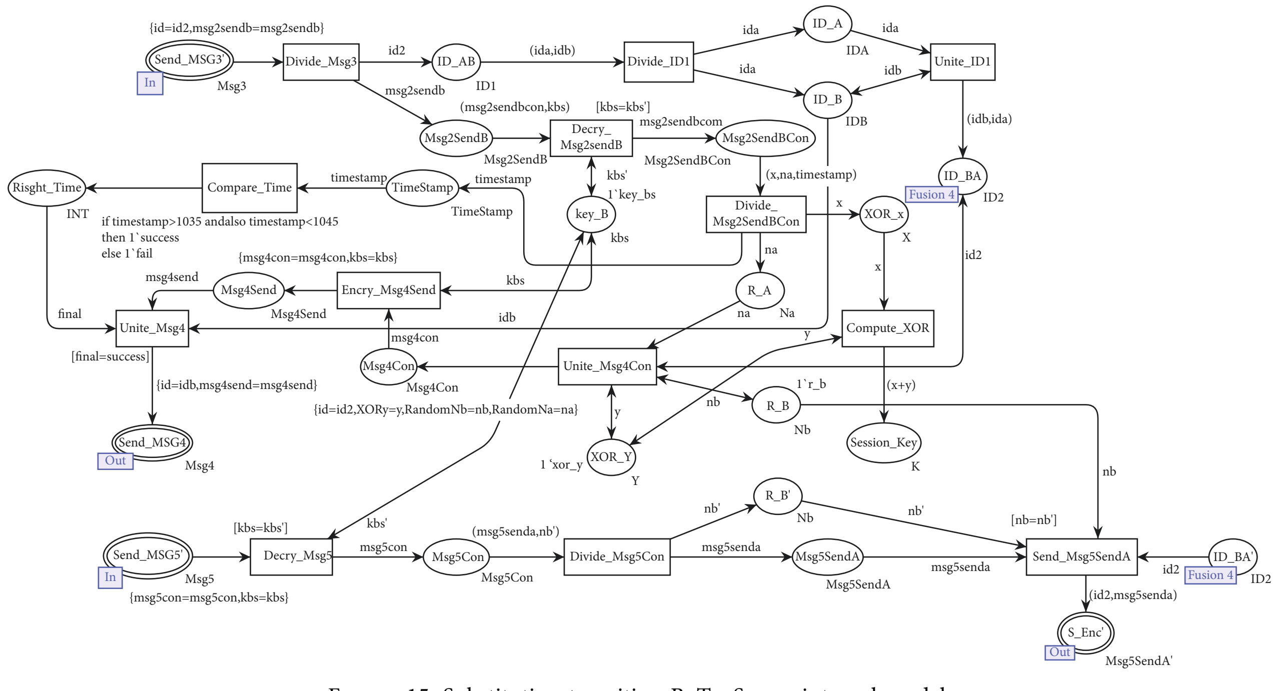

FIGURE 15: Substitution transition B_To_Server internal model.

transition Divide_Msg5SendACon. The place XOR_X and the place XOR_Y send the received messages to the transition XOR, which calculates the session key. The transition Unite_Random combines the received random numbers and timestamp together and sends it to the place Random.

Figure 15 details the internal model of the substitution transition B_To_Server. The place Send_MSG3' sends the message to the transition Divide_Msg3. The transition Divide_Msg3 splits the message into the place ID_AB and the place Msg2SendB. The place ID_AB sends the message to the transition Divide_ID1, which sends the split message to the place ID_A and the place ID_B. The place Msg2SendB sends the message to the transition Decry_Msg2SendB, and the transition Decry_Msg2SendB decrypts with key K_{BS} and sends the decrypted message to the transition Divide_Msg2SendBCon. The transition Divide_Msg2SendBCon separates the received message and sends it to the place XOR_X, the place R_A, and the place Timestamp,

```
(1) id2, msg2sendb = Split (msg3), where msg3 is sent from Server
(2) ida, idb = Split (ida, idb), and new id2 = Combine (idb, ida)
(3) msg2sendb = {msg2sendbcon, kbs}
    if kbs = = kbs' then
       msg2sendbcon = Decrypt (msg2sendb, kbs')
    else return Fail
(4) x, na, timestamp, msg3con = Split (msg2sendbcon)
    if timestamp >= CurrentTime ()-5 AND timestamp <= CurrentTime () + 5 then
       msg4con = Combine (id2, nb, na, y)
       msg4send = Encrypt (msg4con, kbs)
       Send (msg4), where msg4 = {idb, msg4send}
    else return Fail
```

ALGORITHM 3: Send_MSG4.

```
(1) receive msg5, where msg5 = {msg5con, kbs}
(2) if kbs = = kbs' then
       msg5con = Decrypt (msg5, kbs')
    else return Fail
(3) msg5senda, nb' = Split (msg5con)
    if nb = = nb' then
       Send (msg5senda'), where msg5senda = {id2, msg5senda}
    else return Fail
```

ALGORITHM 4: Compute_Msg5SendA'.

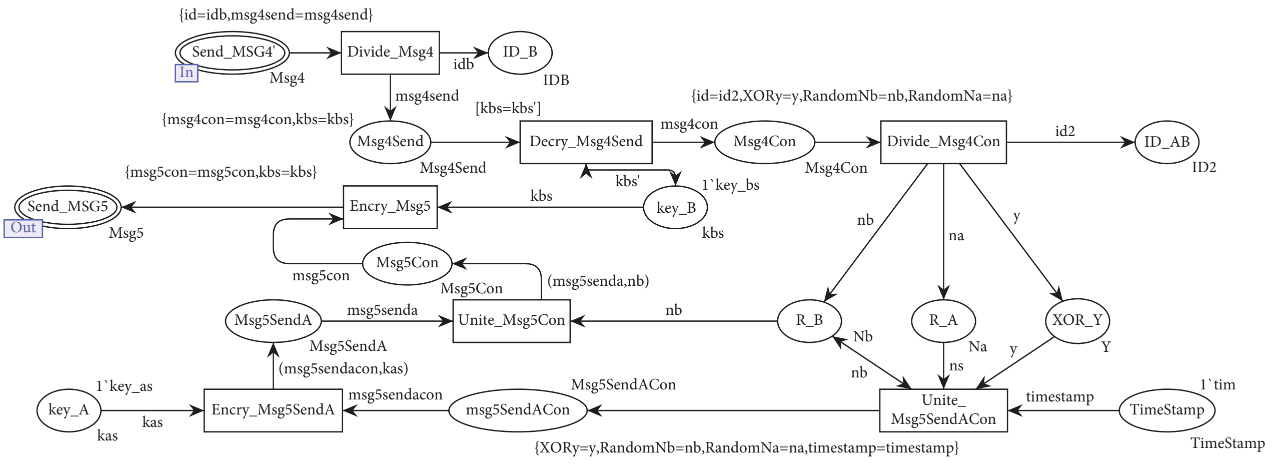

FIGURE 16: Substitution transition RequestB internal model.

respectively. The transition Encry_Msg4Send encrypts the received message with the master key K_{BS} to the place Msg4Send. The place Timestamp determines the stored timestamp and sends the result to the transition Unite_Msg4. The place Send_MSG5' sends the received message to the transition Decry_Msg5, the transition Decry_Msg5 decrypts the message with the key and sends the decrypted message to the transition Divide_Msg5Con, and the transition Divide_Msg5Con splits the received messages to the place R_B' and the place Msg5SendA. The place XOR_X and the place XOR_Y send the received information to the transition XOR, which calculates the session key. The transition Compare_RB compares the messages sent by the place R_B with those sent by the place R_B' and sends the comparison to transition Send_Msg5SendA. Place Msg5SendA sends the result to the transition Send_Msg5SendA, which ultimately sends the result to the place S_Enc'. The pseudocode of the substitution transition B_To_Server is shown in Algorithms 3 and 4.

Figure 16 details the internal model, that is, the substitution transition RequestB. The place Send_MSG4' sends the message to the transition Divide_Msg4, and the transition Divide_Msg4 splits the message to the place ID_B and the place Msg4Send. The transition Unite_Msg5SendACon combines N_A, N_B, Y, and timestamp and sends them to the transition Encry_Msg5SendA, and the transition Encry_Msg5SendA

```
(1) idb, msg4send = Split (msg4), where msg4 is sent from sender
(2) msg4send = {msg4con, kbs}
    if kbs = kbs’ then
       msg4con = Decrypt (msg4send, kbs’)
    then return Fail
(3) id2, $y$, na, nb = Split (msg4con)
    compute timestamp = CurrentTime ()
    msg5sendAcon = combine (nb, na, $y$, timestamp)
(4) msg5senda = Encrypt (msg5sendacon, kas)
    msg5con = Combine (nb, msg5senda)
    msg5 = Encrypt (msg5con, kbs)
    Send (msg5)
```

Algorithm 5: Send_MSG5 ().

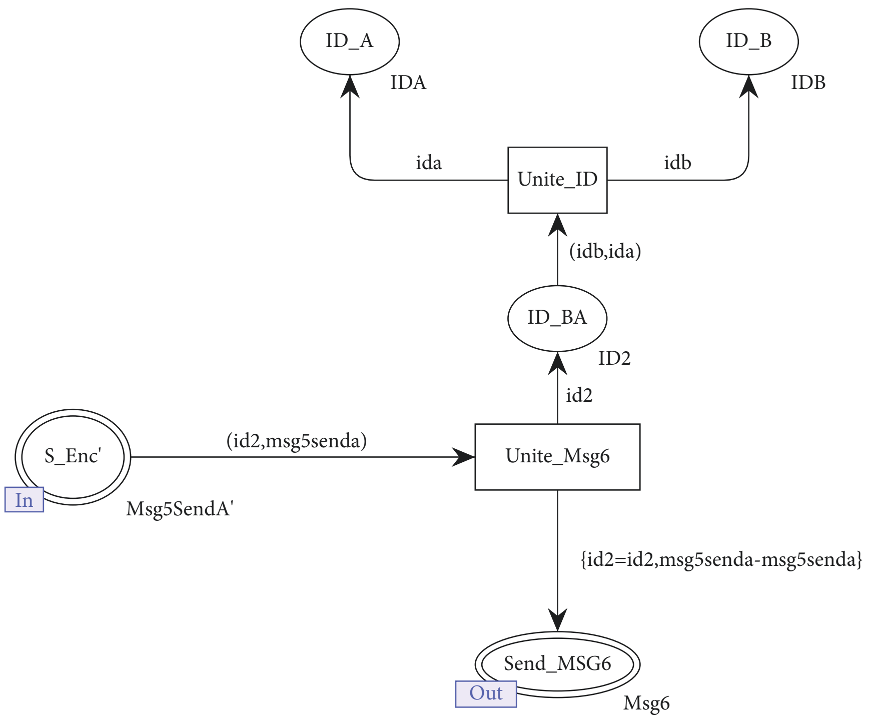

Figure 17: Substitution transition B_To_A internal model.

encrypts received messages using the master key K_{AS}, sending the encrypted message to the transition Unite_Msg5Con. The transition Unite_Msg5Con combines the received message with N_B and sends it to the transition Encry_Msg5. Finally, the transition Encry_Msg5 encrypts the received message using the master key K_{BS} and sends the encrypted message to the place Send_MSG5. The pseudocode of the substitution transition RequestB is shown in Algorithm 5.

Figure 17 details the internal model, that is, the substitution transition B_To_A. The place S_Enc’ receives the message and sends it to the transition Unite_Msg6, which sends id from the received message to the transition Unite_ID. The place Send_MSG6 sends the message to the receiver.

Figure 18 details the internal model of the substitution transition A_HashTo_B. The place Random sends the received message to the transition Divide_Random, which sends the received message to the transition Unite_Msg7Con and the transition Compute_Hash. The transition Encry_Msg7 encrypts the received message with the session key K and sends it to the place Send_MSG7. The place Send_MSG8’ sends the received message to the transition Decry_Msg8, the transition Decry_Msg8 decrypts the received message using the session key K, and the decrypted message is sent to the transition Divide_Msg8Con. The place Hash’ and the place Hash send the stored information to the transition Compare_Hash for judgment. If two hash values are the same, the authentication is successful and the message is saved to the place Store. Otherwise, the message is saved to the place Discard.

Figure 19 illustrates the internal model of the substitution transition B_HashTo_A in detail. The transition

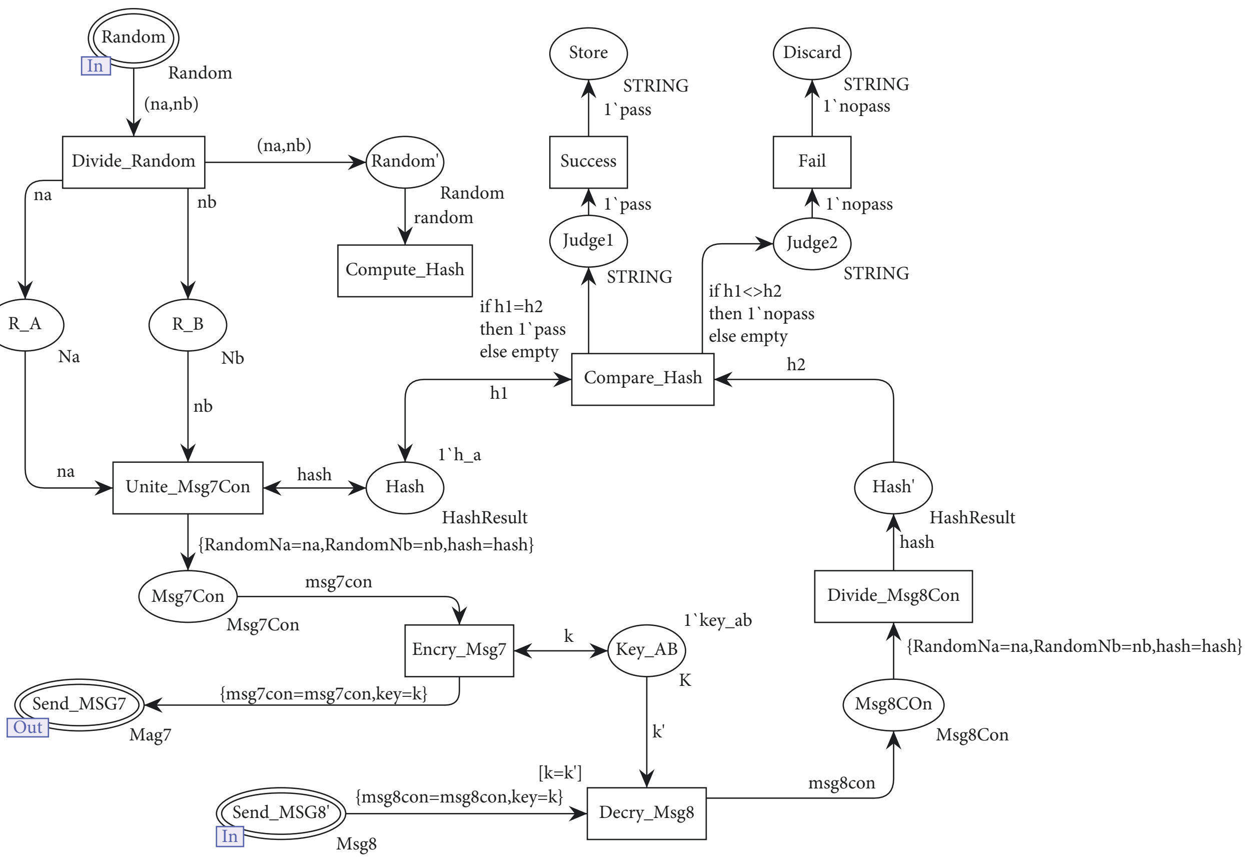

Figure 18: Substitution transition A_HashTo_B internal model.

Compute_Hash combines N_A, N_B, and hash and sends it to the transition Encry_Msg8. The transition Encry_Msg8 encrypts the received message using the session key K and sends the decrypted message to the place Send_MSG8. The place Send_MSG7' sends the received message to the transition Decry_Msg7, which decrypts the received message using the session key K and sends the decrypted message to the transition Send_Msg7Con. The transition Send_Msg7Con extracts the message hash from the received message and sends it to the place Hash'. The place Hash and the place Hash' send the two hashes to the transition Compare_Hash, which compares the two hashes. If the hash values are the same, the authentication succeeds and the message is saved to the store of the place. Otherwise, the message is saved to the place Discard.

5.2. Security Assessment of LonTalk-SA Authentication Protocol. The Dolev-Yao attacker model is introduced to carry out man-in-the-middle attack on the network level of the new scheme model, including tampering, spoofing, and replay attacks. The blue part simulates replay attack, the red part simulates tamper attack, and the purple part simulates spoofing attack. The details are shown in Figure 20.

5.3. LonTalk-SA Authentication Protocol Security Evaluation. The state-space report of the LonTalk-SA authentication protocol is compared with the state-space report of the LonTalk authentication protocol after adding the attack. The specific content is shown in Table 5. There are two dead transitions found in the state-space report of LonTalk-SA. These two dead transitions are since no attacker was introduced in the protocol and no security attack occurred, so the authentication of the two nodes was successful. The attacker model is introduced into the LonTalk-SA authentication protocol, and the number of nodes and places is reduced. The tampering attack T_Att is introduced into the protocol. Due to the wrong judgment of the Hash value, the protocol authentication fails, and the 9 dead transitions are all different in the Hash value, resulting in the failure of identity authentication. A replay attack R_Att is introduced into the protocol. The attacker resends the intercepted message to the receiver. After receiving the message, the receiver finds that the value of the timestamp in the message has exceeded the time range, so it is determined that a replay attack occurs, and 43 dead transitions occur. All are caused by wrong timestamp judgment. The spoofing attack S_Att is introduced into the protocol, the transition Send_MSG7' cannot be fired, and seven dead transitions are all caused by spoofing attacks. Since the transmission of the message is encrypted by the key, the attacker cannot decrypt the obtained information and cannot know the specific content in the message, so the attacker cannot launch tampering, replaying, and spoofing attacks on the LonTalk-SA authentication protocol. The new protocol guarantees the security of the message transmission process.

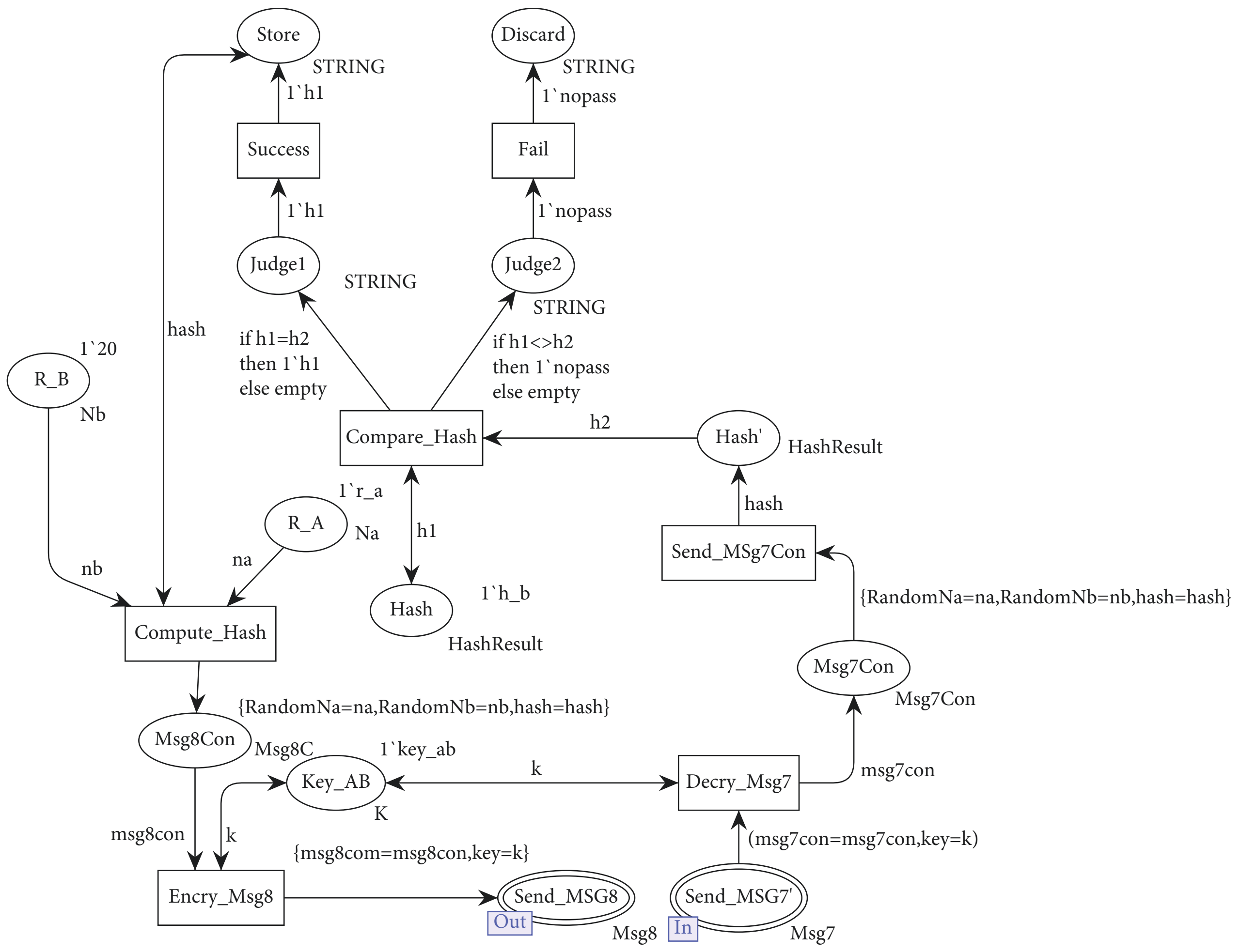

Figure 19: Substitution transition B_HashTo_A internal model.

5.4. Security Analysis of the New Scheme

(1) Antimalicious instruction: this kind of attack means that the attacker sends malicious packets to the node in order to destroy the system with malicious instructions. However, in the LonTalk-SA authentication protocol, the attacker cannot obtain the master key between the node and the server or the session key between nodes, so the packets sent by the attacker cannot be verified and the system cannot be damaged.

(2) Antieavesdropping attack: the attacker adopts the passive attack method to eavesdrop on the data transmitted in the network, analyze the data, and launch an attack on the node. Since all messages transmitted in the LonTalk-SA authentication protocol are encrypted with a key, the attacker cannot eavesdrop on the transmitted data.

(3) Antireplay attack: the attacker eavesdrops on the transmitted data and resends the intercepted data to the receiver in the next round of communication between nodes to deceive the receiver. The timestamp is added to the LonTalk-SA authentication protocol. If the timestamp in a packet exceeds the time range, the receiver directly discards the packet.

(4) Bidirectional authentication: in LonTalk-SA, the sender and receiver use the challenge-response mechanism to authenticate the identities of the communication parties.

(5) Perfect forward security: each communication party generates a random number for the calculation of the session key. Each authentication operation generates a new random number for the calculation of the session key, ensuring that historical communication messages will not be affected if the current session key is leaked.

Table 6 analyzes and compares the security of LonTalk-SA and LonTalk authentication protocols for security problems such as eavesdropping, replay, and bidirectional authentication. The data in the table fully show that LonTalk-SA provides better security.

In view of the security vulnerabilities in the LonTalk authentication protocol, and according to the insufficient computing performance of Neuron chips mentioned in the relevant literature, a trusted third-party server is introduced

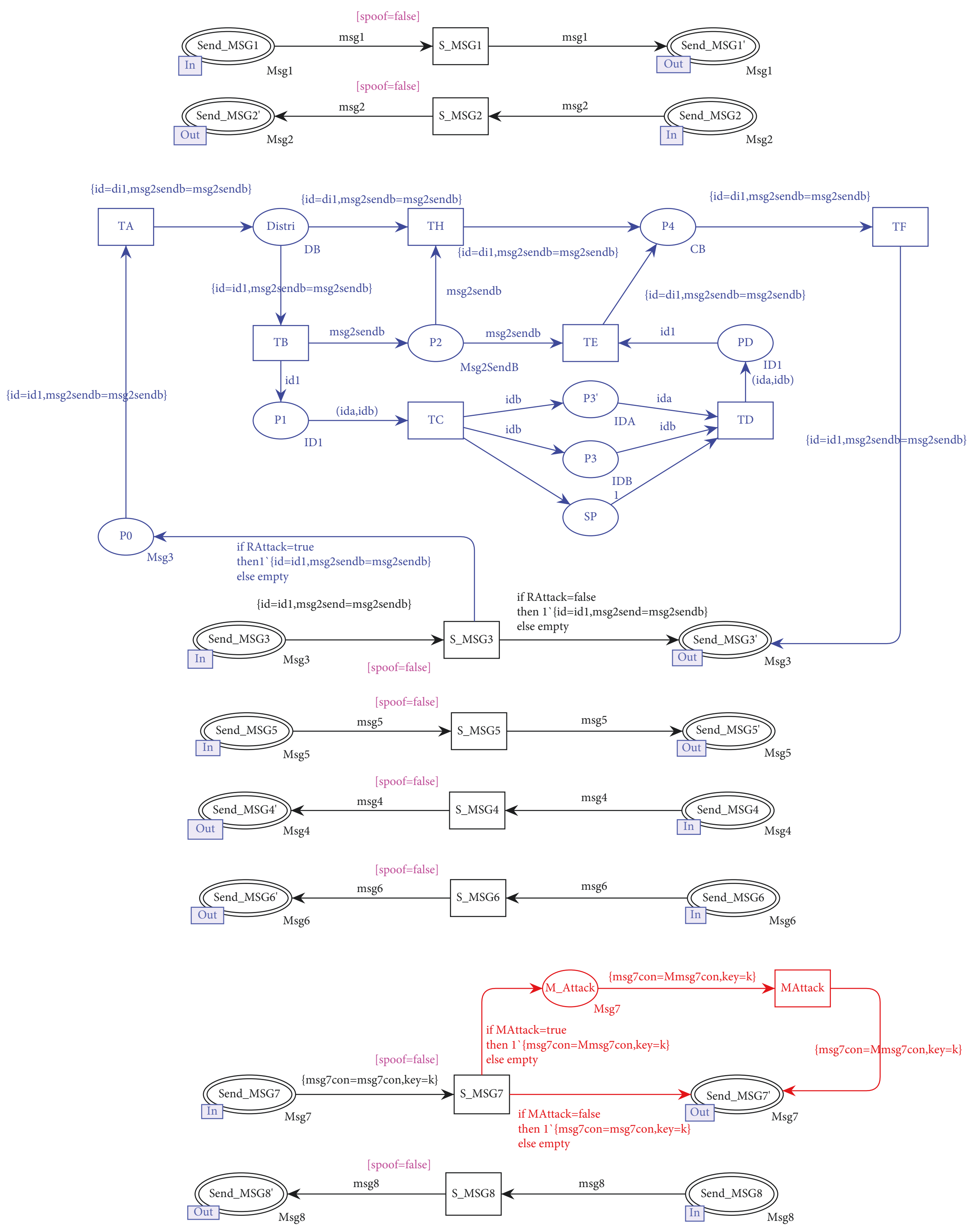

Figure 20: Security evaluation model of LonTalk-SA authentication service.

into the new protocol, and the third-party server completes the identity authentication of the interactive nodes. Third-party servers can add timestamps to messages to prevent replay attacks by attackers. The traditional protocol security improvement scheme usually adopts the third-party server to generate the session key and then transmit it to the communication node. This method saves the performance of the node to a certain extent; however, when an attacker launches

Table 5: Comparison of LonTalk-SA state space.

Type	No_Att	T_Att	R_Att	S_Att
State-space node	1230	846	86	910
State-space arc	4498	2962	160	3218
SCC graph node	1230	846	86	910
SCC graph arc	4498	2962	160	3218
Dead transition	2	9	43	7
Dead marking	1	1	1	1

Table 6: Security comparison between the original protocol and the improved protocol.

Agreement	Malicious instruction	Tamper attack	Replay attacks	Mutual authentication	Forward safety
LonTalk	×	×	×	×	×
LonTalk-SA	√	√	√	√	√

an attack on a third-party server, the session key may be leaked. Therefore, the LonTalk-SA authentication protocol adopts the key negotiation method. Each authentication node generates a random number and securely transmits the random number to the communicating party. The communicating parties perform the XOR operation on the two random numbers to calculate the session key. When the two communicating parties perform identity authentication again, a new session key can be calculated. The key agreement method also ensures the freshness of the session key.

6. Summary and Outlook

The LonTalk protocol is a standard protocol widely used in smart buildings, and its inherent security needs to be further studied. This study takes the LonTalk authentication protocol as the object. First, aiming at the security problems existing in the LonTalk authentication protocol mentioned in the relevant literature, the CPN Tools is used to model the protocol, and the Dolev-Yao attacker model is introduced to evaluate the security of the protocol. There are three types of attack vulnerabilities: tampering, replay, and spoofing. In view of the above security issues, and considering the low performance of Neuron chips in the LonTalk authentication protocol, in the new LonTalk-SA authentication protocol, a third-party server is introduced, and key negotiation is used to ensure the confidentiality, integrity, and authentication of the data in transmission process. The formalized security analysis of the LonTalk-SA authentication protocol shows that the new scheme can effectively defend against the above three attack methods. Timestamps are added to the LonTalk-SA authentication protocol to prevent replay attacks. Therefore, the synchronization of device clocks must be ensured when using time stamps. In the future work, we will consider whether the protocol meets the real-time requirements under the premise of ensuring security.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors would like to thank the National Natural Science Foundation of China (Grant No. 61762060) and the Foundation for the Key Research and Development Program of Gansu Province, China (Grant No. 20YF3GA016).

References

[1] J. Anca, T. Niculcea, P. Ranaweera, and N. Le-Khac, "Security considerations for internet of things: a survey," *SN Computer Science*, vol. 1, no. 4, p. 193, 2020.

[2] V. Anurag, S. Prakash, V. Srivastava, A. Kumar, and S. C. Mukhopadhyay, "Sensing, controlling, and IoT infrastructure in smart building: a review," *IEEE Sensors Journal*, vol. 19, no. 20, p. 9036, 2019.

[3] V. Graveto, Simöes, and P. Simöes, "Security of building automation and control systems: survey and future research directions," *Computers & Security*, vol. 112, Article ID 102527, 2022.

[4] M. Albany, E. Alsahafi, Elkhediri, and S. Elkhediri, "A review: secure internet of thing system for smart houses," *Procedia Computer Science*, vol. 201, pp. 437–444, 2022.

[5] R. Khatoun and J. Nebhen, "Survey on smart homes: vulnerabilities, risks, and countermeasures," *Computers & Security*, vol. 117, Article ID 102677, 2022.

[6] T. Feng, Y. Lu, and F. J. Fang, "Research on vulnerability and security technology of industrial Ethernet protocol," *Journal on Communications*, vol. 38, no. S2, pp. 185–96, 2017.

[7] J. Burke, P. Gasti, N. Nathan, and G. Tsudik, "Securing instrumented environments over content-centric networking: the case of lighting control and NDN," in *Proceedings of the 2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 394–398, Turin, Italy, April 2013.

[8] P. Ganesan, R. Venugopalan, P. Peddabachagari, A. Dean, and M. Sichitiu, "Analyzing and modelling encryption overhead for sensor network nodes," in *Proceedings of the WSNA*, pp. 151–159, San Diego CA USA, September 2003.

[9] Kastner and W. Kastner, "Security analysis of open building automation systems," *Lecture Notes in Computer Science*, vol. 6351, pp. 303–316, 2010.

[10] P. Jovanovic and S. Neves, "Dumb crypto in smart grids: practical cryptanalysis of the open smart grid protocol," *Crystals*, vol. 2015, p. 428, 2015.

[11] P. Kamal, A. Abuhussein, and S. Shiva, "Identifying and scoring vulnerability in scada environments," *Future Technologies Conference (FTC)*, vol. 2017, pp. 845–857, 2017.

[12] J. Ng, S. L. Keoh, Z. Tang, and H. Ko, "SEABASS: Symmetric-Keychain Encryption and Authentication for Building Automation Systems," in *Proceedings of the 2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*, pp. 219–224, Singapore, February 2018.

[13] N. Okabe, S. Sakane, K. Miyazawa, K. Kamada, A. Inoue, and M. Ishiyama, "Security Architecture for Control Networks Using IPsec and KINK," in *Proceedings of the The 2005 Symposium on Applications and the Internet*, pp. 414–420, Trento, Italy, February 2005.

[14] C. Schwaiger and A. Treytl, "Smart card based security for fieldbus systems," in *Proceedings of the EFTA 2003. 2003 IEEE conference on emerging technologies and factory automation*, vol. 1, pp. 398–406, Lisbon, Portugal, September 2003.

[15] X. Yan and W. Bo, "A security extension to LonWorks/LonTalk protocol," *International Journal of Digital Content Technology and its Applications*, vol. 7, no. 6, pp. 790–780, 2013.

[16] P. Ganesan, S. A. A. Hakeem, S. M. A. El-Kader, and H. Kim, "A key management protocol based on the hash chain key generation for securing LoRaWAN networks," *Sensors*, vol. 21, no. 17, 2021.

[17] P. Satyanarayana, "Detection and blocking of reply, false command, and false access injection commands in SCADA systems with modbus protocol," *Security and Communication Networks*, vol. 2021, Article ID 8887666, 15 pages, 2021.

[18] K. Jensen, L. M. Kristensen, and L. Wells, "Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems," *International Journal on Software Tools for Technology Transfer*, vol. 9, no. 3, pp. 213–254, 2007.

[19] A. V. Ratzer, L. Wells, H. M. Lassen et al., "CPN tools for editing, simulating, and analysing coloured Petri nets," in *Proceedings of the International Conference on Application and Theory of Petri Nets*, pp. 450–462, Berlin, Heidelberg, May 2003.

[20] X. Gong, T. Feng, and J. Z. Du, "Formal modeling and security analysis method for security protocols based on CPN," *Journal of Communications*, vol. 42, no. 09, pp. 240–53, 2021.

[21] J. Zhang, L. Yang, W. Cao, and Q. Wang, "Formal analysis of 5G EAP-TLS authentication protocol using proverif," *IEEE Access*, vol. 8, pp. 23674–23688, 2020.

[22] Q. Li, Y. Q. Ma, Y. L. Ma, and Q. F. Cheng, "Cryptanalysis and Enhancement of an Authenticated Key Agreement Protocol for Dew-Assisted IoT Systems," *Security and Communication Networks*, vol. 2022, Article ID 7125491, 11 pages, 2022.

[23] S. Meier, B. Schmidt, C. Cremers, and D. A. Basin, "The TAMARIN prover for the symbolic analysis of security protocols," *Computer Aided Verification*, vol. 8044, pp. 696–701, 2013.

[24] Yao and A. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983.

[25] J. W. Song and J. X. Cui, "Key technology and open test method of rail train field bus," *Journal of Physics: Conference Series*, vol. 1828, no. 1, Article ID 012066, 2021.

[26] S. M. Zhang, Y. C. Zhang, and B. Y. Wang, "A lightweight remote mutual authentication scheme for smart home environments," *Computer, Signals and Systems*, vol. 6, pp. 40–45, 2022.