Computer Science and Information Systems 21(1):21–36 https://doi.org/10.2298/CSIS230611068W

Applying SPIN Checker on 5G EAP-TLS Authentication Protocol Analysis

Qianli Wang

Software School,
East China JiaoTong University,
Nanchang, China
qianjustice2@163.com

Abstract. Currently, there is relatively little formal analysis and verification work on the 5G EAP-TLS authentication protocol. In this paper, we use the model checker SPIN to perform a formal analysis of the 5G EAP-TLS authentication protocol. Firstly, we analyze the process of the 5G EAP-TLS authentication protocol and abstract it to obtain a formal model of the protocol. Then, we describe the construction of the protocol model based on the Promela language. The unique feature of this paper is the replacement of the hash value of the 5G EAP-TLS authentication protocol with the message content field encrypted by an unknown subject public key. This is because the Promela language in SPIN has an eval function that can check the value of each field. This can replace the function of the hash function and make the Promela model construction more portable. The paper analyzes the attack paths of the protocol and reveals design flaws that undermine the expected identity authentication attributes and secret consistency of the protocol. The results not only provide a comprehensive understanding of the security properties of the 5G EAP-TLS authentication protocol but also offer valuable insights and guidance for the verification of the protocol's security properties, security design, and optimization of protocol implementation and interoperability.

Keywords: 5G EAP-TLS; SPIN; Formal Analysis; Model Checking.

1. Introduction

The introduction of 5G technology will undoubtedly change the way we use the Internet and communicate, but it also brings risks and challenges in terms of network security. Therefore, 5G users are not only concerned about the improvement of network speed, but also particularly concerned about ensuring network security [1]. 5G networks are constantly facing security issues such as eavesdropping and surveillance, device intrusion, privacy data leakage, honeypot attacks, and network fraud. In order to alleviate these risks and challenges, 5G networks need to adopt a series of security measures, such as encryption technology, authentication and access control of devices, encrypted storage and transmission of data, authentication and access control, vulnerability management, and security training.

In the 5G network, three identity authentication protocols are defined in the relevant 3GPP documents, including the 5G AKA (Authentication and Key Agreement) protocol

[2], the EAP-AKA protocol [2], and the 5G EAP-TLS protocol [2,3]. The first two protocols use symmetric key passwords and are protocols used for authentication and key negotiation. The 5G EAP-TLS protocol uses public key passwords and is a certificate-based authentication protocol. The 5G EAP-TLS protocol is a certificate-based authentication protocol that uses digital certificates to verify the user's identity. In this protocol, the authentication between the client and server is based on the certificates they hold, which are signed by a trusted third-party certificate authority (CA).The identity authentication and key negotiation methods provided by these three protocols are different, so their specific application scenarios are also different. So far, these protocols are still in the standardization process and are mainly developed and released in the form of RFCs. The main sources of serious security vulnerabilities in the implementation of informal protocols are protocol design issues and implementation problems, for example, as reported in literature [4,5].Formal analysis is an effective method to solve the ambiguity and verify the correctness of the protocol design. First, the protocol model is constructed using a formal language such as Promela, and then the formal protocol model is analyzed to see if it meets the required security properties. Symbolic model checking [6] is an important method for performing formal analysis of protocols. Since the first work [7] using this technique to analyze the design flaws of the Needham-Schroeder protocol, the symbolic model checking of security protocols has been a hot research field, it has been considered a powerful technique for formal analysis of security protocol [8-14] design, and has been applied in some real-world Protocols, such as TLS [15].

2. Related Work

Zhang Jingjing's work [1] built a formal model for the 5G EAP-TLS authentication protocol based on pi calculus and used the ProVerif model checker to formally verify the model, revealing several weaknesses and design flaws in the current protocol, thereby undermining the expected identity authentication attributes. Her work provides strong support for the formal verification of the 5G EAP-TLS authentication protocol. Chen Liping [16] studied the formal model of the EAP-TLS protocol and security attributes based on ProVerif, and verified the mutual authentication between user devices and the network, as well as the confidentiality of the security anchor key KSEAF and the user identity identifier SUPI in the protocol. Wang Yuedong [17] constructed a protocol interaction model based on the Diffie-Hellman pattern, then extended the Dolev-Yao attacker model, proposed two controlled participant scenarios and a mixed channel model, and finally used the verification tool SmartVerif to verify the confidentiality, authentication, and privacy of the protocol. This research provides valuable references for designing secure communication protocols. However, the formal analysis and verification of the 5G EAP-TLS authentication protocol, as a novel and important protocol, is relatively scarce, and it is urgently needed for scholars to analyze and verify it from different perspectives, different methods, and even different tools to better maintain the security of the network in the 5G environment.

SPIN [18], developed by Holzmann, is a general model checking tool used to verify logical consistency in concurrent systems. It is loved by scholars for its concise

modeling language, support for Linear Temporal Logic (LTL), and good algorithmic structure. The literature [19] uses SPIN to verify known vulnerabilities in the classic NSPK cryptographic protocol and proposes a static analysis message construction method to mitigate the state explosion problem, providing guidance for future scholars using SPIN for security protocol verification. Since then, a large number of researchers have used SPIN to formally verify security protocols and have achieved good results [20-23]. Therefore, this paper uses the model checker SPIN [24-25] to perform formal analysis and verification of the 5G EAP-TLS authentication protocol [26].

The main contributions of this paper are twofold: First, it replaces the hash value of the 5G EAP-TLS authentication protocol with a message content field encrypted using an unknown subject public key. This is because the Promela language in SPIN has an "eval" function that can detect the value of each field, which can replace the functionality of the hash function. The advantage is that it makes the Promela model construction more portable. Second, based on the formal model in literature [1], this paper uses the model checker SPIN to perform formal analysis of the 5G EAP-TLS authentication protocol and verifies the confidentiality, authenticity, and secret consistency of the user identity SUPI, the pre-master key Rprekey, and the master key Kseaf.

The rest of this paper is organized as follows. Section 2 provides a detailed introduction to the abstract model of the 5G EAP-TLS protocol, and describes the process of constructing the protocol model based on the Promela language, and implements the security properties of the 5G EAP-TLS authentication protocol. In Section 3, the results of SPIN verification are presented, and the attack paths of the protocol are discussed. Section 4 discusses the results obtained. Finally, Section 5 summarizes the work of this study and outlines future directions.

3. Research Methods

3.1. Abstraction Of 5G EAP-TLS Authentication Protocol

Based on the protocol flow described above, this section will abstract the 5G EAP-TLS authentication protocol. The formalized model of the abstracted protocol is shown in Fig.1. Since the message transmission between the service network and the home network is usually wired and forwarded, it can be assumed that the message transmission between them is secure. Therefore, this article abstracts the 5G EAP-TLS authentication protocol as an interaction between two roles: user terminal A and network B, where network B represents a combined module of the service network and the home network.

First, the user terminal encrypts the identity SUPI and a newly generated random number Na using the public key of network B, and sends the message to network B. Upon receiving the message, network B sends an EAP-TLS protocol start signal Start to the terminal A, and the terminal generates a new random number Na1 and sends it back to the network. After receiving the response from the terminal, network B generates its own random number Nb and its certificate and sends them to the terminal. The

certificate is represented as {B}SKb in this article. The terminal verifies the certificate first, and then generates a pre-master key Kab and a master key Kseaf, where Kseaf is calculated from Na1, Nb, and Kab. Since Na1 and Nb are transmitted in plaintext, the secrecy of Kseaf will be guaranteed by Kab. The terminal calculates the hash value of the previous two message contents and sends its own certificate, the signed hash value, and the hash value encrypted with the master key Kseaf to network B. Note that since the hash function is a one-way encryption function and has anti-inverse property, a public key of an unknown subject can be used to simulate hash encryption in the formalization process. This is because anyone can encrypt a message using this public key, but the private key corresponding to this public key is secret, so no one can reverse the hashing process. To simplify the model, this article uses the Hash function to calculate the hash value used for signature, and uses the message encrypted with the unknown subject public key PKz to replace the hash value encrypted with the master key.

After receiving the message, the network side verifies the signature of the terminal A, calculates the master key Kseaf based on the second received random number Na1, its own random number Nb, and the received pre-master key Kab. The network side then decrypts the message using Kseaf, compares it with its own first two messages after performing a hash operation, and if the hash value matches the received hash value, the network side successfully verifies the identity of terminal A. The network B then encrypts its third and fourth messages using an unknown principal's public key PKz, and then encrypts them using Kseaf before sending them to terminal A. After receiving the message, terminal A decrypts it using Kseaf and compares the decrypted message to verify the identity of the network side.

Unlike the formal model in the literature [1], this article replaces one hash value in the fifth message and the hash value in the sixth message with the encrypted message content field using an unknown principal's public key PKz. This is because the SPIN Promela language has an eval function that can check the value of each field, which can replace the function of the hash function and make the construction of the Promela model more convenient.

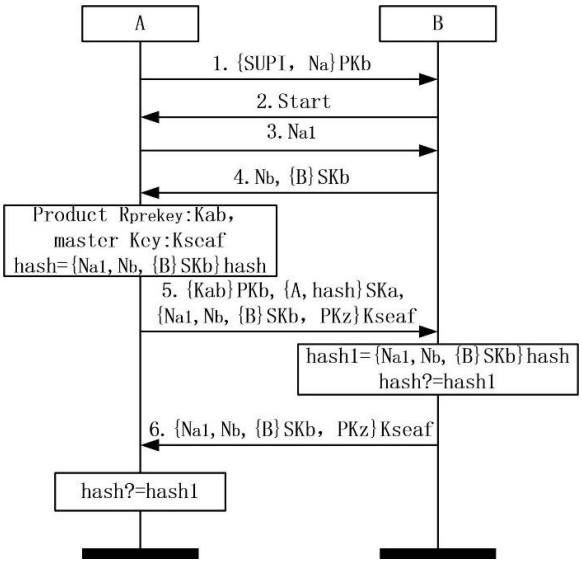

Fig. 1. Abstracted 5G EAP-TLS authentication protocol

3.2. Model Assumptions

The formal model of a security protocol is typically described as a collection of honest entities that interact with each other according to the protocol rules, and an intruder who is powerful enough to perform all possible attack behaviors. The communication network between the entities is abstracted as the protocol message channels.

To better reveal the security vulnerabilities in the protocol, rather than the weaknesses of the cryptographic system used by the protocol, we usually make perfectness assumptions about the underlying cryptographic algorithms of the protocol, as follows:

-Only the entity that knows the corresponding decryption key can decrypt a message.

It is impossible to derive the encryption key from the encrypted message itself.

-There is sufficient redundancy in the protocol messages to enable the decryption algorithm to detect whether the message is encrypted with the expected key.

-The encryption components cannot be tampered with, and it is not possible to forge a new large encryption component using multiple encryption components.

-The intruder model in this paper is based on the well-known Dolev-Yao model, which provides an important principle: never underestimate the capabilities of the intruder. Therefore, we always assume that the intruder is very powerful, able to monitor the entire communication network for eavesdropping, interception, analysis, forgery, and other behaviors, and can also participate in the protocol interaction as a normal entity. In addition, the intruder can continually learn knowledge during the protocol execution and use it to attack the normal protocol process. To better represent the capabilities of the

intruder, this paper assumes that honest entities cannot communicate directly with each other, and that all messages will be intercepted by the intruder.

The construction of a security protocol model based on SPIN can be roughly divided into two steps: modeling the honest entities and modeling the intruder. The following sections will provide specific introductions.

3.3. Honest Entity Modeling

The first step in modeling the 5G EAP-TLS protocol is to define a finite set of names for the protocol model. The Promela definition is as follows：

```
mtype = {NULL,SUPI,Start,Na,Na1,Nb,A,B,I,
R, gD,PKa,PKb,PKi,PKz,SKa,SKb,SKi,Kab,
Kai,Kseaf,Kseaf1};
```

NULL represents a placeholder, SUPI is the unique identifier of the initiator, Na, Na1, and Nb are random numbers generated by honest parties, A and B are the identity tags of honest parties, I is the identity tag of the intruder, R is the service constant of the intruder, gD is the generic data of the intruder, PKa, PKb, and PKi represent the public keys of the parties, PKz represents the public key of an unknown party, SKa, SKb, and SKi represent the private keys of the parties, Kab represents the session key between honest parties A and B, and Kai represents the session key known by the intruder.

Next, I define the communication channels for protocol interaction. Considering the length of the message field in the protocol, I define three synchronous channels with lengths of 4, 12, and 7, respectively. The specific Promela implementation is as follows:

```
chan ca = [0] of {mtype, mtype, mtype, mtype};
chan cb = [0] of {mtype, mtype, mtype,mtype,
mtype, mtype, mtype,mtype, mtype, mtype,
mtype, mtype};
chan cc = [0] of {mtype, mtype, mtype,mtype,
 mtype, mtype, mtype}
```

The channel ca is used to transmit messages 1, 2, 3, and 4, channel cb is used to transmit message 5, and channel cc is used to transmit message 6. For honest parties, the first field of the message is filled with the sender's identity. Any remaining fields are filled with NULL.

After abstraction, the 5G EAP-TLS protocol involves two honest parties. Therefore, this article defines the initiator process PIni and the responder process PRes to respectively depict the behaviors of the two roles in the protocol, and instantiates the parameters. The initiator process has four process parameters, which are self, party, nonce1, and nonce2. Among them, the parameter "self" indicates the subject of the initiator process itself, "party" indicates the expected communication subject of the initiator process, "nonce1" indicates the first random number generated by the initiator, and "nonce2" indicates the second random number generated by the initiator random number. According to the abstracted protocol specification, there are 6 communication

statements in the initiator process. In order to reduce the number of statement interactions between processes and better reduce the state, the "atomic" keyword is used to combine the message receiving statement with the next message Send statements are combined into one atomic step. The process performs message type matching and simulates the function of the hash function by using the "eval(x)" function to determine whether the received value is equal to x. As mentioned above, in order to simplify the message field length, we use the inline function "Hash" in the process to hash the initiator's previous message content, and update the value of the global variable "hash", thereby replacing the complete message 5. The first hash function in the field.

In addition, some macros and inline functions used to describe the nature of the protocol are included in the initiator process, including: "IniRunning" and "IniCommit" are used to represent the initiation and commit phases of the protocol between the initiator and the responder; the inline function "Secrecy" is an assertion used to determine whether the confidentiality of the session key "Kab" has been violated; the inline function "IniSec" is used to update the value of the global variable "AiniSec", which is used to store the session secret obtained by the initiator key. The specific implementation will be described in detail in the section on security attributes.

The responder process is similar to the initiator process. The process parameters include "self" representing the responder process itself and "nonce1" representing the responder's random number. According to the protocol specification, the responder needs to compare the received hash value with the hash of the previous messages it participated in. If the hash values are equal, the responder process can continue to execute. The specific implementation of the responder process will not be described in detail.

Finally, an initialization process needs to be defined to instantiate the initiator and responder processes. The instantiation of the main processes is represented using process instance statements in this process.

The initiator process can choose to communicate with the subject B or with the intruder I, while the responder process only needs to respond to protocol session requests from other subjects and does not need to select communication partners. The PI process is the intruder process and will be introduced later.

3.4. Intruder Modeling

The intrusion message construction strategy in this article adopts the static analysis method in reference [19], which can reduce the number of protocol states and obtain a simpler model by manually screening out invalid messages forged by intruders.

First, I define a set of local variables x1 to x10 to receive the message fields intercepted by the intruder. Secondly, the intruder's knowledge representation is restricted by the static analysis method to define the intruder's initial knowledge set. This article defines the intruder's initial knowledge as subject identity, relevant public keys, the intruder's own key and generic data, including A, B, I, PKa, PKb, PKi, SKi, Kai, gD. When the intruder intercepts a message, it can decompose and learn from the message to increase its knowledge. If the message cannot be decrypted, the entire encryption component is learned. Tab.1 shows all the messages that the intruder may intercept and the knowledge items that can be learned.

The first column in Tab.1 shows all the messages that the intruder may intercept, and the second column shows the knowledge that the intruder can learn through intercepting the messages. Tab.2 lists the messages that the intruder can forge and the knowledge required to send the messages.

Table 1. Knowledge that an intruder can ultimately obtain

Received message	Learning knowledge
{SUPI,Na}PKb; {SUPI,Na}PKi; Start; Na1; gD; Nb; Nb,{B}SKb; {Kab}PKb,{A,hash}SKa, {{Na1,Nb,{B}SKb},pkz} Kseaf; {Kab}PKb,{A,hash}SKa, {{Na1,Nb,{B}SKb}PKz} Kseaf; {Kab}PKb,{A,hash}SKa, {{Na1,gD,{B}SKb}PKz} Kseaf; {Kai}PKi,{A,hash}SKa, {{Na1,Nb,{B}SKb}PKz} Kseaf1; {Kai}PKi,{A,hash}SKa, {{Na1,Na,{B}SKb}PKz} Kseaf1; {Kai}PKi,{A,hash}SKa, {{Na1,gD,{B}SKb}PKz} Kseaf1; {{Na1,Nb,{B}SKb}PKz} Kseaf; {{Na,Nb,{B}SKb}PKz} Kseaf; {{gD,Nb,{B}SKb}PKz} Kseaf;	{SUPI,Na}PKb; SUPI; Na; Start; Na1; Nb; {B}SKb; {Kab}PKb; {A,Na1,Nb,{B}SKb}SKa; {{Na1,Nb,{B}SKb}PKz} Kseaf; {A,Na1,gD,{B}SKb}SKa; {{Na1,gD,{B}SKb},PKz} Kseaf; {{Na1,Na,{B}SKb}PKz} Kseaf; {A,Na1,Na,{B}SKb}SKa; {{Na,Nb,{B}SKb}PKz} Kseaf; {{gD,Nb,{B}SKb}PKz} Kseaf; {Na1,Nb,{B}SKb}PKz

The intersection of the knowledge that the intruder may obtain and the knowledge that the intruder needs is the truly useful knowledge for the intruder.

The intersection is shown below:

- Atomic message：Na; Start; Na1; Nb;SUPI;
- Composite message：

$\{SUPI,Na\}PKb;\{B\}SKb;\{Kab\}PKb;$

$\{A,Na1,Nb,\{B\}SKb\}SKa;$

$\{\{Na1,Nb,\{B\}SKb\}PKz\}Kseaf;$

$\{\{gD,Nb,\{B\}SKb\}PKz\}Kseaf;$

$\{\{Na,Nb,\{B\}SKb\}PKz\}Kseaf;$

$\{\{Na1,gD,\{B\}SKb\}PKz\}Kseaf;$

$\{Na1,Nb,\{B\}SKb\}PKz;$

Table 2. Knowledge that an intruder may need

Send Message	Required knowledge
{SUPI,Na}PKb;{I,Na}Pkb;	SUPI;
{SUPI,Nb}PKb;	Na;
{SUPI,gD}PKb;{I,gD}PKb;	Na1;
Start;Na1;gD;Na;	Nb;
Nb,{B}SKb;gD,{B}SKb;	{SUPI,Na}PKb;
Nb,{I}SKi;gD,{I}SKi;	{SUPI,Nb}PKb;
{Kab}PKb,{A,hash}SKa,	Start;
{{Na1,Nb, {B}SKb}PKz}Kseaf;	{B}SKb;
{Kai}PKb,{A,hash}SKa,{{Na1,Nb,	{Kab}PKb;
{B}SKb}PKz}Kseaf1;	Kab;
{Kab}PKb,{A,hash}SKa,{{Na,Nb,	Kseaf;
{B}SKb} PKz}Kseaf;	{A,Na1,Nb,{B}SKb}S
{Kai}PKb,{A,hash}SKa,{{Na,Nb,	Ka;
{B}SKb} PKz}Kseaf1;	{{Na1,Nb,{B}SKb}
{Kab}PKb,{A,hash}SKa,{{gD,Nb,	PKz}Kseaf;
{B}SKb} PKz}Kseaf;	{A,Na,Nb,{B}SKb}S
{Kai}PKb,{A,hash}SKa,{{gD,Nb,	Ka;
{B}SKb}PKz}Kseaf1;	{{Na,Nb,{B}SKb}
{Kab}PKb,{I,hash}SKi,{{Na1,Nb,{B}S	PKz}Kseaf;
Kb} PKz}Kseaf;	{A,gD,Nb,{B}SKb}S
{Kai}PKb,{I,hash}SKi,{{Na1,Nb,	Ka;
{B}SKb} PKz}Kseaf1;	{{gD,Nb,{B}SKb}
{Kab}PKb,{I,hash}SKi,{{Na,Nb,	PKz}Kseaf;
{B}SKb} PKz}Kseaf;	{{Na1,gD,{B}SKb}
{Kai}PKb,{I,hash}SKi,{{Na,Nb,	PKz}Kseaf;
{B}SKb} PKz}Kseaf1;	{{Na1,gD,{I}SKi}
{Kab}PKb,{I,hash}SKi,{{gD,Nb,	PKz}Kseaf;
{B}SKb} PKz}Kseaf;	{Na1,Nb,{B}SKb}
{Kai}PKb,{I,hash}SKi,{{gD,Nb,	PKz
{B}SKb} PKz}Kseaf1;	
{Na1,Nb,{B}SKb}PKz}Kseaf;	
{{Na1,gD,{B}SKb}PKz}Kseaf;	
{{Na1,Nb,{B}SKb} PKz}Kseaf1;	
{{Na1,gD,{B}SKb}PKz}Kseaf1;	
{I,Nb}PKb;	
{{Na1,Nb,{I}SKi}PKz}Kseaf;	
{{Na1,gD,{I}SKi}PKz}Kseaf;	
{{Na1,Nb,{I}SKi}PKz}Kseaf1;	
{{Na1,gD,{I}SKi}PKz}Kseaf1;	

First, it is necessary to initialize the knowledge items of the intruder using a bit-type variable based on the intersection obtained. When the intruder learns a certain knowledge item, update the value of the corresponding variable to 1. The intruder process behaves as an endless process, constantly receiving or sending messages from to the channel, which can be achieved by a do loop. Then construct the intruder's sending message statements based on Tab.2 and the content of the intersection, and construct the intruder's receiving statements using the intersection. The update of the intruder's knowledge is implemented through macros k1, k2, k3, and k5. The receive statements and the corresponding knowledge learning are both placed in d_step statements, and at the end of the statements, the variables x1, etc. need to be reset to zero, which helps to reduce the number of system states.

3.5. Security Attributes

This paper verifies the authentication, confidentiality and secret consistency of the 5G EAP_TLS protocol. Authentication requires mutual authentication of identities between the terminal and the network after the execution of the protocol is completed. This paper is based on the method of reference [19] to characterize weak consistency through LTL linear temporal logic. The specific implementation is as follows:

ltl e1 {([] (([]!r) || (!r U s))&&(([]!P) || (!p U q)))}

The specific meaning is that when role A initiates a protocol session with role B, role B must have participated in a session with role A before this. The reverse implication is also true. Macros such as IniRunningAB are used to update the values of the corresponding global variables.

Confidentiality of the 5G EAP_TLS protocol requires that the user terminal's unique identity SUPI, newly generated pre-master key Kab, and master key Kseaf will not be learned by intruders before the protocol execution between the user terminal and the network ends. This article uses the Secrecy macro to implement the protocol confidentiality assertion.

On the premise that the intruder does not participate in protocol interaction as an honest entity, the intruder cannot obtain Kab, Kseaf and SUPI by attacking the protocol.

The secret consistency of the 5G EAP_TLS protocol requires that at the end of the protocol execution between the user terminal UE and the network AUSF, the two parties should agree on the pre-master key Rprekey. In the abstract model of this article, this means that the main keys of roles A and B reach an agreement, which is implemented through LTL. The Promela implementation is as follows:

```
ltl e2 {[] ((AiniSec != NULL && BresSec !=
NULL) -> (AiniSec == BresSec))}
```

4. Experiment and Result

The experimental environment of this article is as follows: Intel i5 CPU, 64-bit Linux, 2G RAM, and Spin V6.5.1. In depth-first search mode, Spin is used to verify the authentication, confidentiality, and secret consistency of the 5G EAP-TLS protocol, with a maximum search depth of the default value 10000.

First, the authentication of the 5G EAP-TLS protocol is verified, and a loophole was found.

The attack path that violates the authentication is shown in Fig.2.

The specific attack steps are shown below:

Step 1: The initiator A sends SUPI and random number Na to the responder B, and encrypts them using the responder's public key PKb.

```
ca ! A, SUPI, Na,PKb;
```

Step 2: The intruder I intercepts the message and masquerades as responder B to send an EAP-TLS start message "Start" to the initiator A.

```
ca ! A, Start, NULL,NULL;
```

Step 3: The initiator A receives the message and sends a new random number Na1 to responder B.

ca ! A, Na1, NULL, NULL;

Step 4: The intruder intercepts the message and masquerades as responder B to send its own generic data gD and the certificate {B}SKb obtained from the previous session to initiator A.

ca ! A, gD, B, SKb;

Step 5: The initiator A receives the message, verifies the certificate, and then sends the pre-master secret Kab encrypted with B's public key, its own certificate, the hash value of the first two message contents signed, and the first two message contents encrypted with the master key using PKz.

cb ! A, Kab, PKb, A, hash, Ska, Na1, gD, B,
SKb, PKz, Kseaf;

Step 6: The intruder intercepts the message and captures the content of the message encrypted with the master key, then masquerades as responder B and sends it to initiator A. After receiving the message, A verifies the message fields and completes the authentication of B's identity.

cc ! Na1, gD, B, SKb, PKz, Kseaf;

As can be seen from the above attack path, the initiator A believes that it has completed mutual authentication with the responder B, but in reality, the responder B did not participate in the protocol execution with A, which violates the authenticity.

Next, use SPIN to verify the confidentiality of SUPI, pre-master key Kab, and master key Kseaf in the 5G EAP-TLS protocol. The verification results show that the confidentiality of SUPI, Kab, and Kseaf in the 5G EAP-TLS protocol can be satisfied, and no loopholes have been found.

Finally, SPIN is used to verify the secrecy consistency of the 5G EAP-TLS protocol, and a vulnerability was found.

The attack sequence that violates the secrecy consistency is shown in Fig.3, and the specific attack steps are shown below:

Step 1: The initiator A sends SUPI and a random number Na to the responder B, and encrypts them using the responder's public key PKb.

ca ! A, SUPI, Na, PKb;

Step 2: The intruder I intercepts the message and forwards it to the responder B.

ca ! B, SUPI, Na, PKb;

Step 3: The responder B receives the message and verifies SUPI, then sends a protocol start message "Start" to the initiator A.

ca ! B, Start, NULL,NULL;

Step 4: The intruder intercepts the message and forwards it to the initiator A.

ca ! A, Start, NULL,NULL;

Step 5: The initiator A receives the message and sends a new random number Na1 to the responder B.

ca ! A, Na1, NULL, NULL;

Step 6: The intruder intercepts the message and forwards it to the responder B.

ca ! B, Na1, NULL, NULL;

Step 7: The responder B receives the message and sends its own random number Nb and certificate to the initiator A.

ca ! B, Nb, B, SKb;

Step8: The intruder intercepts the message and forwards it to the initiator A.

ca ! A, Nb, B, SKb;

Step 9: The initiator A receives the message, verifies the received certificate, and if verification succeeds, sends the pre-shared key Kab encrypted with B's public key, its own certificate, the hash value of the first two message contents signed, and the first two message contents encrypted with PKz that has been used with the master key.

cb ! A, Kab, PKb, A, hash, Ska, Na1, gD, B,

SKb,PKz, Kseaf;

Step 10: The intruder intercepts the message, modifies the pre-shared key Kab to Kai, forges a new master key Kseaf1 using the modified pre-shared key Kab, Na1, and Nb, and sends it to the responder B in the message format.

cb ! B, Kai, PKb, A, hash, Ska, Na1, Nb, B,

SKb, PKz, Kseaf1;

Step 11: The responder B receives the message, verifies the hash value, treats Kai as the pre-shared key and Kseaf1 as the master key, encrypts the first two message contents that have been encrypted with PKz using Kseaf1, and sends them to the initiator A.

cc ! B, Na1, Nb, B, SKb,PKz, Kseaf1;

Step 12: The intruder intercepts the message, uses fragments of previously intercepted messages to forge a message, and sends it to the initiator A. The initiator A receives the message, verifies it, and upon successful verification, considers itself to have achieved agreement with the responder B on the pre-shared key and the master key.

cc ! A, Na1, Nb, B, SKb, PKz, Kseaf;

As can be seen from the above attack steps, the initiator A generates a pre-master key Kab and a master key Kseaf, while the responder receives a pre-master key Kai and a master key Kseaf1. However, both parties believe that they have reached a consensus on the values of the pre-master and master keys. Therefore, the 5G EAP-TLS protocol violates the secret consistency

5. Discussion

The results of this paper provide valuable insights into the security of the 5G EAP-TLS authentication protocol from both theoretical and practical perspectives. The formal model of the protocol constructed based on the Promela language can be used to verify whether the protocol satisfies properties such as authentication, secrecy, and secret consistency, and to discover possible vulnerabilities in the protocol. The use of the model checker SPIN to perform formal analysis and verification of the protocol provides a reliable and effective method for ensuring the security of the 5G network environment.

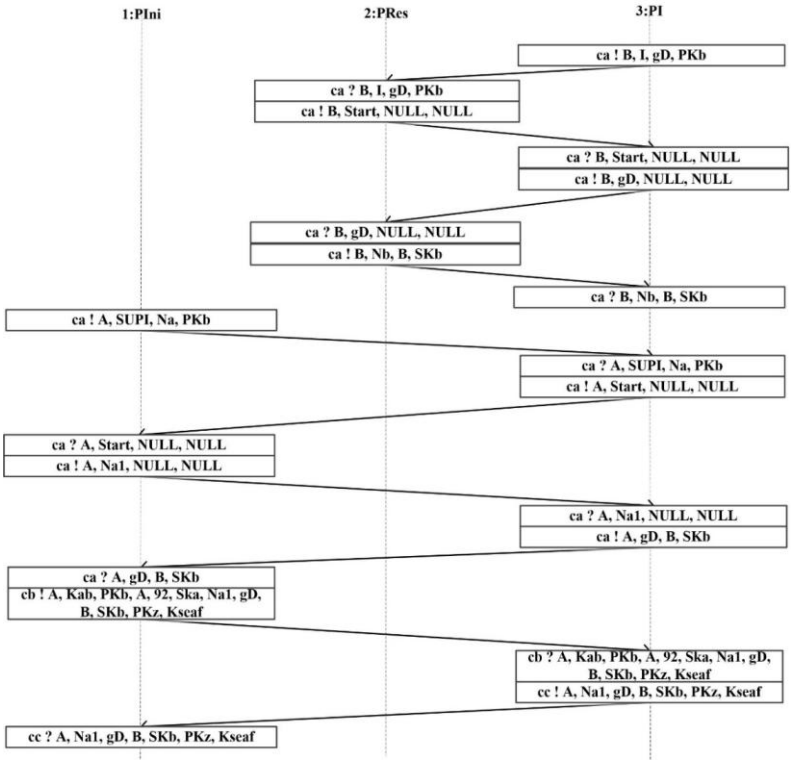

Fig. 2. Attack path for authentication violation of 5G EAP-TLS protocol

The unique feature of this paper is the replacement of the hash value of the 5G EAP-TLS authentication protocol with the message content field encrypted by an unknown subject public key, which not only makes the construction of the Promela model more convenient but also provides more comprehensive coverage of possible attack paths. In addition, the paper analyzes the attack paths of the protocol and reveals design flaws that undermine the expected identity authentication attributes and secret consistency of the protocol. This could serve as a reference for the future design and implementation of secure communication protocols in 5G networks.

In summary, The SPIN model can abstract the EAP-TLS protocol into a state machine model, perform formal verification on the protocol process, and discover potential security vulnerabilities. This is beneficial for theoretically optimizing and repairing the EAP-TLS protocol. The SPIN model can automate testing of the EAP-TLS protocol, detect abnormal behavior during actual deployment and operation, and discover unknown security attack points. Once a security vulnerability is detected, the SPIN model can provide replication steps and repair suggestions to help developers fix the vulnerability in a timely manner and improve the security of 5G networks. The SPIN model detection results can serve as a test report for the 5G device certification function, providing reference for device certification and improving product quality. Formal validation of the EAP-TLS protocol can provide stricter specifications, which can help unify and optimize the differences between EAP-TLS implementations from different

vendors, and improve interoperability, the results of this paper not only provide a comprehensive understanding of the security properties of the 5G EAP-TLS authentication protocol but also offer valuable insights and guidance for the design and verification of secure communication protocols in 5G networks.

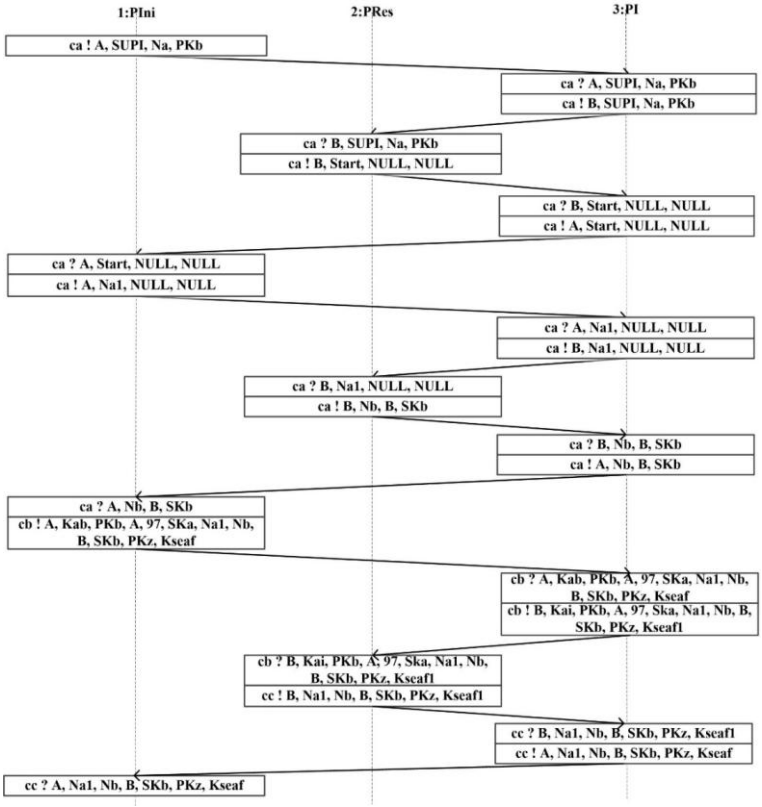

Fig. 3. Attack path for secret consistency violation of 5G EAP-TLS protocol

6. Conclusion

In this paper, a model detector SPIN is used to formally analyze the security properties of the 5G EAP-TLS authentication protocol. The special thing is that this article replaces the hash value of the 5G EAP-TLS authentication protocol with the message content field encrypted with the public key of the unknown subject. This is because there is an eval function in the Promela language of SPIN, which can detect the value of each field. This can replace the function of the hash function, and the advantage is that it can make the construction of the Promela model more convenient. This paper describes the 5G EAP-TLS authentication protocol process, abstracts the authentication protocol, and obtains the formal model of the 5G EAP-TLS authentication protocol. Then, the

protocol model construction based on Promela language is described in detail, the honest subject modeling and intruder modeling are constructed, and the weak consistency and secret consistency of the protocol are described by LTL linear temporal logic. Finally, the authentication and secret consistency of the protocol are verified. The results of verification and analysis reveal several design flaws, indicating that there are indeed loopholes in the identity authentication attributes and secret consistency of the 5G EAP-TLS authentication protocol, and the attack path has been run out.

In view of the defects of 5G EAP-TLS, the protocol will be improved next, and the improved protocol will be verified. At the same time, considering that the new protocol is becoming more and more complex, our next step is to improve the Promela modeling method to alleviate the state explosion problem.

Acknowledgment. The research was supported by Innovation and Entrepreneurship Training Program for College Students in Jiangxi Province.

References

1. Zhang, J., Yang, L., Cao, W., & Wang, Q. (2020). Formal analysis of 5G EAP-TLS authentication protocol using proverif. IEEE access, 8, 23674-23688. https://doi.org/10.1109/ACCESS.2020.2969474
2. 3GPP. (2020). Security architecture and procedures for 5G system. Technical Specification (TS) 3GPP TS 33.501 V17. 0.0 (2020–2012).
3. Dierks, T., & Rescorla, E. (2008). The transport layer security (TLS) protocol version 1.2 (No. rfc5246). https://doi.org/10.17487/RFC5246
4. Basin, D., Cremers, C., Miyazaki, K., Radomirovic, S., & Watanabe, D. (2014). Improving the security of cryptographic protocol standards. IEEE Security & Privacy, 13(3), 24-31. https://doi.org/10.1109/MSP.2013.162
5. Hirschi, L., Sasse, R., & Dreier, J. (2019). Security issues in the 5G standard and how formal methods come to the rescue. ERCIM News. https://hal.science/hal-02268822
6. Basin, D., Cremers, C., & Meadows, C. (2018). Model checking security protocols. Handbook of Model Checking, 727-762. https://doi.org/10.1007/978-3-319-10575-8_22
7. Lowe, G. (1995). An attack on the Needham− Schroeder public− key authentication protocol. Information processing letters, 56(3). https://doi.org/10.1016/0020-0190(95)00144-2
8. Mitchell, J. C., Mitchell, M., & Stern, U. (1997, May). Automated analysis of cryptographic protocols using mur/spl phi. In Proceedings. 1997 IEEE Symposium on Security and Privacy (Cat. No. 97CB36097) (pp. 141-151). IEEE. https://doi.org/10.1109/SECPRI.1997.601329
9. Song, D. X. (1999, June). Athena: a new efficient automatic checker for security protocol analysis. In Proceedings of the 12th IEEE Computer Security Foundations Workshop (pp. 192-202). IEEE. https://doi.org/10.1109/CSFW.1999.779773
10. Clarke, E. M., Jha, S., & Marrero, W. (2000). Verifying security protocols with Brutus. ACM Transactions on Software Engineering and Methodology (TOSEM), 9(4), 443-487. https://doi.org/10.1145/363516.363528
11. Armando, A., & Compagna, L. (2004). SATMC: a SAT-based model checker for security protocols. In Logics in Artificial Intelligence: 9th European Conference, JELIA 2004, Lisbon, Portugal, September 27-30, 2004. Proceedings 9 (pp. 730-733). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-30227-8_68
12. Basin, D., Mödersheim, S., & Vigano, L. (2005). OFMC: A symbolic model checker for security protocols. International Journal of Information Security, 4, 181-208. https://doi.org/10.1007/s10207-004-0055-7

13. Cortier, V., Delaune, S., & Lafourcade, P. (2006). A survey of algebraic properties used in cryptographic protocols. Journal of Computer Security, 14(1), 1-43. https://doi.org/10.3233/JCS-2006-14101
14. Blanchet, B. (2016). Modeling and verifying security protocols with the applied pi calculus and ProVerif. Foundations and Trends® in Privacy and Security, 1(1-2), 1-135. http://dx.doi.org/10.1561/3300000004
15. Cremers, C., Horvat, M., Hoyland, J., Scott, S., & van der Merwe, T. (2017, October). A comprehensive symbolic analysis of TLS 1.3. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (pp. 1773-1788). https://doi.org/10.1145/3133956.3134063
16. Chen Liping, Xu Peng, Wang Danchen & Xu Yang. (2022).Formal Verification Research of EAP-TLS Protocol. Computer Science (S2), pp. 685-689. https://doi.org/10.11896/jsjkx.211100111
17. Wang Yuedong, Xiong Yan, Huang Wenchao & Wu Jianshuang.(2021). A Formal Analysis Scheme for 5G Private Network Authentication Protocol. Information Network Security(09),1-7. http://netinfo-security.org/CN/10.3969/j.issn.1671-1122.2021.09.001
18. Holzmann, G. J. (1997). The model checker SPIN. IEEE Transactions on software engineering, 23(5), 279-295. https://doi.org/10.1109/32.588521
19. Maggi, P., & Sisto, R. (2002). Using SPIN to verify security properties of cryptographic protocols. In Model Checking Software: 9th International SPIN Workshop Grenoble, France, April 11–13, 2002 Proceedings 9 (pp. 187-204). Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-46017-9_14
20. Chen, S., Fu, H., & Miao, H. (2016, June). Formal verification of security protocols using Spin. In 2016 IEEE/ACIS 15th international conference on computer and information science (ICIS) (pp. 1-6). IEEE. https://doi.org/10.1109/ICIS.2016.7550830
21. Xiao, M., Song, W., Yang, K., OuYang, R., & Zhao, H. (2022). Formal Analysis of the Security Protocol with Timestamp Using SPIN. Computational Intelligence and Neuroscience, 2022. https://doi.org/10.1155/2022/2420590
22. Edelkamp, S., Leue, S., & Lluch-Lafuente, A. (2004). Directed explicit-state model checking in the validation of communication protocols. International journal on software tools for technology transfer, 5, 247-267. https://doi.org/10.1007/s10009-002-0104-3
23. Ninet, T., Legay, A., Maillard, R., Traonouez, L. M., & Zendra, O. (2019, August). Model checking the IKEv2 protocol using Spin. In 2019 17th International Conference on Privacy, Security and Trust (PST) (pp. 1-7). IEEE. https://doi.org/10.1109/PST47121.2019.8949057
24. Galaudage, S., Talbot, C., Nagar, T., Jain, D., Thrane, E., & Mandel, I. (2021). Building better spin models for merging binary black holes: evidence for nonspinning and rapidly spinning nearly aligned subpopulations. The Astrophysical Journal Letters, 921(1), L15. https://doi.org/10.3847/2041-8213/ac2f3c
25. Liu, M. Z., Wu, T. W., Sánchez, M. S., Valderrama, M. P., Geng, L. S., & Xie, J. J. (2021). Spin-parities of the P c (4440) and P c (4457) in the one-boson-exchange model. Physical Review D, 103(5), 054004. https://doi.org/10.1103/PhysRevD.103.054004
26. Nyangaresi, V. O., Abduljabbar, Z. A., & Abduljabbar, Z. A. (2021, December). Authentication and Key Agreement Protocol for Secure Traffic Signaling in 5G Networks. In 2021 IEEE 2nd International Conference on Signal, Control and Communication (SCC) (pp. 188-193). IEEE. https://doi.org/10.1109/SCC53769.2021.9768338

Qianli Wang is born on July 22, 2002, studies at Software School, East China JiaoTong University, Nanchang, China, seriously conduct scientific research and has a strict plan for the future.

Received: June 11, 2023; Accepted: September 10, 2023.