Trusted Sharing of Data Under Cloud-Edge-End Collaboration and Its Formal Verification

1st Xuejian Li
School of Computer Science and Technology
Anhui University
Hefei Anhui , China
lxj@ahu.edu.cn

2nd Mingguang Wang
School of Computer Science and Technology
Anhui University
Hefei Anhui , China
e21301223@stu.ahu.edu.cn

***Abstract*—With the development of cloud computing and edge computing, data sharing and collaboration have become increasing between cloud edge and end. Under the assistance of edge cloud, end users can access the data stored in the cloud by data owners. However, in an unprotected cloud-edge-end network environment, data sharing is vulnerable to security threats from malicious users, and data confidentiality cannot be guaranteed. Most of the existing data sharing approaches use the identity authentication mechanism to resist unauthorized accessed by illegal end users, but the mechanism cannot guarantee the credibility of the end user's network environment. Therefore, this article proposes an approach for trusted sharing of data under cloud-edge-end collaboration (TSDCEE), in which we verify the trustworthiness of the data requester's network environment based on the mechanism of attribute remote attestation. Finally, this article uses model checking Spin method to formally analyze TSDCEE, and verifies the security properties of TSDCEE.**

***Index Terms*—Data sharing, Cloud-edge-end collaboration, Remote attestation, Model checking**

I. Introduction

With the rapid development of the Internet, data security has become an important issue of widespread concern. On the one hand, enterprises or organizations outsource a large amount of data to the cloud to pursue the convenience, economy, efficiency, reliability, and scalability advantages brought by cloud computing, expecting the cloud to guarantee data security. On the other hand, in the cloud-edge-end network environment, terminal devices access the Internet to request data stored on the cloud by the data owner. However, with the addition of terminal devices, data security risks also increase. In the network environment, terminal devices carrying malicious codes or viruses may successfully connect to the cloud, send malicious attacks to cloud storage, steal or destroy data, resulting in damage to user data. Therefore, how to ensure the security and trusted sharing of data is a problem worth studying[1].

In an open network environment, communications between computers may be intercepted or tampered with by attackers, resulting in data leakage and security breaches. Therefore, to protect the security of data in the cloud storage environment, on the one hand, the data needs to be encrypted and stored in the cloud to ensure the confidentiality of the data; on the other hand, a secure communication protocol is required to ensure the integrity of the data during the data sharing process. Ruj *et al.*[2] designed a private access control protocol for cloud-stored data to ensure the security of data storage in the cloud and user privacy. The protocol ensures the anonymity of user identity and increases access control features, allowing only legitimate users to access data stored in the cloud. Nevertheless, when the protocol stores data, users need to perform complex encryption operations on the data before storing it in the cloud. Lu *et al.*[3] proposed a inter-cloud secure data sharing approach, In this paper, the advanced Petri net is used to model the method, and the Z3[4] constraint solver is used to verify that the method satisfies security properties such as confidentiality, authenticity, and anti-collusion. However, this method is not suitable for secure data sharing in the cloud-edge environment.

Currently, the authentication mechanism[5] in most cloud data sharing schemes verifies whether the user knows a specific password or possesses some kind of authentication credentials. This approach works well for securing access to resources, but is powerless for detecting malware or unauthorized modifications, whereas remote attestation mechanisms verify that a computer or device is in the expected state. It can be used to detect malware or unauthorized modification, and verify whether the device has the required software and hardware configuration. The detailed remote attestation mechanism can be referred to [6].

Therefore, based on attribute remote attestation, this paper proposes a trusted sharing of data under cloud-edge-end collaboration (TSDCEE), which can ensure that end users under a trusted network have the right to access resources in the cloud, and can resist malicious access from untrusted terminals. The specific structure of the scheme is given in the next section.

II. TSDCEE Approach

A. The overview of TSDCEE

The TSDCEE approach proposed in this article is shown in Fig. 1. ES serves as a verification server to determine the security status of the DSR platform and sends the determination result to DPR. When DSR is trusted, DPR sends the encrypted key for the requested data to ES, which is then sent along with the data to DSR. The relevant entities are described as follows:

- DSR: Data requester is a user who apply for access to cloud resources. There is a TPM in the DSR end to generate identity keys.
- DPR: The data possesser is the uploader of cloud data. It formulates access control policies to determine the key distribution of ciphertext data.
- ES: Edge cloud server. On the one hand, it acts as the entrusted computing service of DSR, deciding whether to share data with DSR. on the other hand, it can verify the security status of DSR end platform, and send the verification result to DSR.
- DS: Cloud server, database of sensitive resources. The sensitive data stored in the cloud discussed in this article is encrypted ciphertext data.
- TTP: Trusted third parties, servers that are trusted by other entities. In order to initialize TSDCEE, TTP needs to convert the configuration information of the DSR platform into attribute details and send it to DSR. In addition, in order to ensure communication security, TTP calculates keys for DSR, DPR and ES respectively.

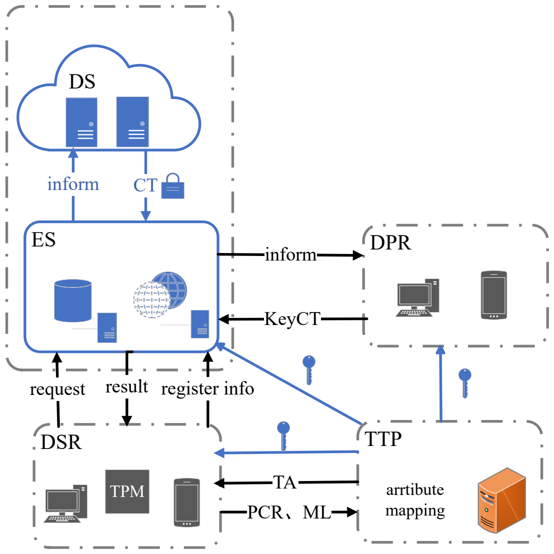

Fig. 1. The overview of TSDCEE.

In order to facilitate the description, the symbols used in the process of sharing data are given in Table I. The overall process of sharing data includes the following six steps:

Step1: As shown in (1), DSR uses the identity key AIK generated by its TPM platform to encrypt the message Req and sends it to ES. Among them,$Req = (ID_a, ID_b, N_1, DataId, EK, P)$.

$$\text{DSR} \rightarrow \text{ES} : m_1 = \{Req\}AIK_p. \tag{1}$$

Step2: The ES contains the EK information of all host platform TPMs in the security domain under its jurisdiction. After receiving M1, ES searches and matches in the database according to the identity information EK sent by DSR. If ES has a legitimate identity record for DSR, the identity authentication of DSR by ES is considered successful, and the next operation is performed. otherwise, the request from DSR fails and ES immediately terminates the next operation. After a successful authentication of DSR by ES, ES matches the data information in the request message with the data directory information in ES, searches the data records in the data directory to determine the purpose cloud to which the requested data belongs. Once the requested data is successfully searched and the DS that stores the requested data is identified, ES notifies DS to provide the correct encrypted data. The above process is shown in (2).

$$\begin{aligned} &\text{ES} \rightarrow \text{DS} : m_2 = (N_2, DataId) \\ &\text{DS} \rightarrow \text{ES} : m_3 = (N_3, CT). \end{aligned} \tag{2}$$

Step3: As shown in (3), The ES verifies TA to determine whether DSR is secure. If the security of DSR is satisfied, ES notifies DPR and sends a portion of the request Req' information to DPR. DPR authenticates the ES server by determining whether the ID in the message sent by ES matches the ES identity ID. If they are consistent, DPR generates the corresponding shared policy SP and sends it to ES. ES authenticates DPR through the identity ID and password PW sent by DPR. If authentication fails, ES immediately terminates the next step. As follows, $Req' = (ID_a, ID_b, N_1, DataId)$

$$\begin{aligned} &\text{ES} \rightarrow \text{DPR} : m_4 = (ID_b, N_4, \{Req'\}Key_{ES\|}) \\ &\text{DPR} \rightarrow \text{ES} : m_5 = \{ID_d, PW, SP\}Key_{ES\|DPR}. \end{aligned} \tag{3}$$

Step4: As shown in (4), ES then evaluates whether the security attribute SA of DSR satisfies the sharing policy SP. If satisfied, ES sends the verification result of DSR, denoted as Result, to DPR. Upon receiving Result, DPR sends the corresponding decryption key to ES.

$$\begin{aligned} &\text{ES} \rightarrow \text{DPR} : m_6 = \{Result\}Key_{ES\|DPR}. \\ &\text{DPR} \rightarrow \text{ES} : m_7 = \{KeyCT\}Key_{ES\|DPR}. \end{aligned} \tag{4}$$

Step5: As shown in (5), After receiving the key, ES sends the encrypted data CT and the key KeyCT to the data consumer DSR.

$$\text{ES} \rightarrow \text{DSR} : m_8 = (CT, \{KeyCT\}Key_{ES\|DSR}). \tag{5}$$

Step6: Upon receipt of both the ciphertext and key, the DSR is capable of decrypting and subsequently inspecting the information.

III. Security Verification of TSDCEE Approach Based on SPIN

A. Security requirements and threat model

The security requirements of the TSDCEE approach, as shown in Fig. 2, are as follows:

- Confidentiality: Unauthorized users cannot access plaintext shared data.
- Authenticity: In TSDCEE, ES first checks the identity information in the request message. If the identity authentication of DSR is successful, ES sends the requested information to DS and DPR. DPR authenticates ES based

TABLE I
SYMBOLS AND DESCRIPTIONS

Symbols	Descriptions
IDa	Entity identity of DSR
IDb	The entity identity of ES
IDd	The entity identity of DPR
$N_i(i \in 1,2,3,4)$	Random numbers
P	Attribute collection, including TA and SA
PW	The DPR's password
$DataId$	The index of data
CT	The ciphertext of data
$KeyCT$	The decryption key
$Key_{ES\|DPR}$	Shared secret for ES and DPR
$Key_{ES\|DSR}$	Shared secret for ES and DSR
TA	The trustworthy attributes of DSR
SA	The security attribute of DSR
SP	Sharing policies formulated by DPR

on the identity information in the request message. After the identity authentication of ES is successful, ES sends the key of the data to DPR. ES authenticates DPR based on the password in the message. After the successful identity authentication of DPR by ES, ES sends the data and key to DSR.

- Collusion-resistance: In the TSDCEE approach, DSR has two attributes, TA and SA. When both properties are satisfied, the ES will send the ciphertext CT and the key $KeyCT$ to the DSR. Malicious users intend to obtain data by colluding with ES, but such malicious users do not have the correct attribute SA of the sharing policy formulated by DSR. We assume ES is honest and do not consider any active attacks from it.

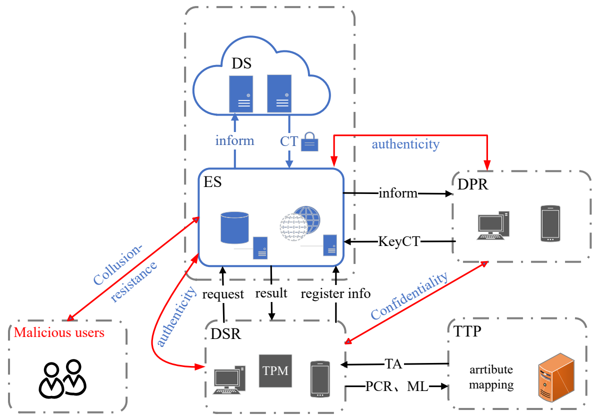

Fig. 2. Security requirements of the TSDCEE approach.

B. Formal Analysis and Verification

1) Message Passing Modeling: The TSDCEE approach involves a total of four honest subjects, and the length of the communication messages between each entity is different. In order to model the approach more clearly, this section declares the message type, channel and variables as follows:

$$\begin{aligned}
&mtype = \{ok, err, Msg1, Msg2, Msg3, Msg4,\\
&\quad Msg5, Msg6, Msg7, Msg8, Key_{SP},\\
&\quad Key_{SS}, Key_{EK}, Key_{CT}, AIK_S,\\
&\quad AIK_P, DataID, CT, SA, TA,\\
&\quad agentDSR, agentES, agentDS,\\
&\quad agentDPR, agentI, NULL\}.\\
&chan\ network_1 = [0]of\{mtype, mtype, mtype,\\
&\quad mtype, mtype, mtype, mtype,\\
&\quad mtype, mtype, mtype\}.\\
&chan\ network_2 = [0]of\{mtype, mtype, mtype,\\
&\quad mtype, mtype\}.\\
&chan\ network_3 = [0]of\{mtype, mtype, mtype,\\
&\quad mtype, mtype, mtype, mtype\}.\\
&bool\ ACK\_CT, ACK\_Key_{CT}, Resist\_Collusion.
\end{aligned} \tag{6}$$

As shown in (6), the status, identity, communication key of entities, and communication message type in TSDCEE are declared in $mtype$. Message channels $network_1$, $network_2$, $network_3$ are respectively used for message passing between different entities in TSDCEE. The variable ACK_CT indicates whether the malicious user has intercepted the ciphertext data. ACK_Key_{CT} indicates whether the attacker knows the key to decrypt the data and $Resist\_Collusion$ indicates whether the malicious user can obtain the data by colluding with ES.

2) Communication Agent Modeling: First, this section uses the modeling language Promela to model each communication entity in TSDCEE to simulate the system operation process. The identities of DSR, ES, DS, DPR and the intruder are represented by $agentDSR$, $agentES$, $agentDS$, $agentDPR$ and $agentI$. $Msg1$, $Msg2$, $Msg3$,$\cdots$, $Msg8$ represent the messages in the communication process of the TSDCEE. ok and err are the communication status values of the entity, representing success and failure. The communication status of each entity is represented by $statusDSR$, $statusES$, $statusDS$ and $statusDPR$. First, the DSR initiates a data request. After the TSDCEE approach is successfully run, the status of each entity participating in the communication is set to ok. As shown in Fig. 3, the running process of the system can be correctly simulated without adding an attacker.

The two most important security attributes of a security protocol are confidentiality and authentication. Confidentiality ensures that important messages during communication cannot be intercepted and identified by illegal third parties. Authentication determines the identity security of communication entities. This section uses the ltl formula to express the security properties satisfied by TSDCEE as follows:

- Confidentiality: During the implementation of the TSDCEE approach, ensure that data is not stolen by malicious users This property is expressed using the ltl formula

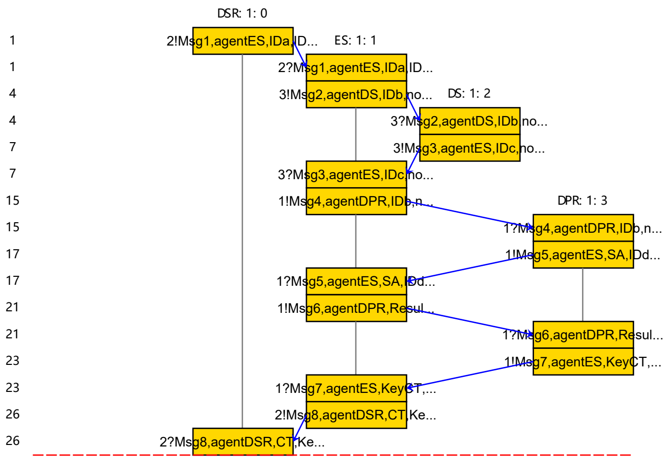
DSR: 1: 0
ES: 1: 1
DS: 1: 2
DPR: 1: 3
1 2!Msg1,agentES,IDa,ID...
1 2?Msg1,agentES,IDa,ID...
4 3!Msg2,agentDS,IDb,no...
4 3?Msg2,agentDS,IDb,no...
7 3!Msg3,agentES,IDc,no...
7 3?Msg3,agentES,IDc,no...
15 1!Msg4,agentDPR,IDb,n...
15 1?Msg4,agentDPR,IDb,n...
17 1!Msg5,agentES,SA,IDd...
17 1?Msg5,agentES,SA,IDd...
21 1!Msg6,agentDPR,Resul...
21 1?Msg6,agentDPR,Resul...
23 1!Msg7,agentES,KeyCT,...
23 1?Msg7,agentES,KeyCT,...
26 2!Msg8,agentDSR,CT,Ke...
26 2?Msg8,agentDSR,CT,Ke...

Fig. 3. Communication Simulation of the TSDCEE approach.

shown in $p1$.

$$ltl\ p1\{[]\,((statusDS == ok\&\&statusDPR == ok) \rightarrow <>\,(partnerDS == agentES \&\&partnerDPR == agentES)) \rightarrow (ACK\_CT)\&\&(ACK\_KeyCT)\}$$

- Authenticity: In TSDCEE, the various entities of the authenticity representation system are able to identify the identity of the communicating entities with themselves. After the protocol is executed, the communication objects of DSR are ES, DS, and DPR. This property is expressed using the ltl formula shown in $p2$.

$$ltl\ p2\{\,[]\,((statusDSR == ok\&\&statusES == ok \&\&statusDS == ok\&\&statusDPR == ok) \rightarrow <>\,(partnerDS == agentES \&\&partnerDPR == agentES)) \rightarrow (ACK\_CT)\&\&(ACK\_KeyCT)\}$$

- Collusion-resistance: Malicious users cannot collude with ES to obtain plaintext data.This property is expressed using the ltl formula shown in $p3$.

$$ltl\quad p3\{[]\,((statusDS == ok\&\&statusDPR == ok) \rightarrow (!((ACK\_CT)\&\&(ACK\_KeyCT))) \&\&!(Resist\_Collusion))\}$$

We will provide explanations for the variables appearing in the ltl formula. Here, "ACK_CT" denotes whether the attacker intercepts the encrypted data. "ACK_KeyCT" indicates whether the attacker knows the decryption key of the data. "$Resist\_Collusion$" represents whether the attacker can pass the security verification of ES.

We sequentially pass the ltl formulas to the Spin model checking tool for verification, and the verification results are shown in Fig. 4. From the figure, it can be seen that "never claim + (p3)" indicates that the ltl formula constraints were

```
(Spin Version 6.5.0 -- 1 July 2019)
      + Partial Order Reduction
Full statespace search for:
      never claim           + (p3)
      assertion violations  + (if within scope of claim)
      acceptance cycles     + (fairness disabled)
      invalid end states    - (disabled by never claim)
State-vector 152 byte, depth reached 68,   errors: 0
      205 states, stored
      92 states, matched
      297 transitions (= stored+matched)
      70 atomic steps
hash conflicts:            0 (resolved)
```

Fig. 4. Verification results of the safety of the TSDCEE approach.

used in this run. "errors: 0" means that no errors occurred during the verification process, indicating that TSDCEE satisfies the security properties described by the ltl formula.

IV. CONCLUSION

This article proposes a data sharing approach, namely TSDCEE, which can achieve confidentiality, authenticity, and collusion-resistance. In order to formally verify the correctness and security of the TSDCEE approach, this article first uses ACP to formalize the correctness of TSDCEE, and then uses the Spin to verify the security properties satisfied by TSDCEE. Our formal analysis results show that TSDCEE achieves the expected goals, including confidentiality, authenticity, and collusion-resistance.

Our future research will focus on user attribute revocation and key management.

REFERENCES

[1] L. Kacha and A. Zitouni, "An overview on data security in cloud computing," *Cybernetics Approaches in Intelligent Systems: Computational Methods in Systems and Software 2017, vol. 1*, pp. 250–261, 2018.

[2] S. Ruj, M. Stojmenovic, and A. Nayak, "Privacy preserving access control with authentication for securing data in clouds," in *2012 12th IEEE/ACM International symposium on cluster, cloud and grid computing (ccgrid 2012)*. IEEE, 2012, pp. 556–563.

[3] J. Lu, Y. Zhang, L. Yang, and S. Jin, "Inter-cloud secure data sharing and its formal verification," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 1, p. e4380, 2022.

[4] L. De Moura and N. Bjørner, "Z3: An efficient smt solver," in *International conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2008, pp. 337–340.

[5] Z. Zhao and X. Xu, "Research on the application of computer data encryption technology in cloud security," *Cryptography*, vol. 14, no. 4, p. 14, 2018.

[6] N. Sumrall and M. Novoa, "Trusted computing group (tcg) and the tpm 1.2 specification," in *Intel Developer Forum*, vol. 32. Intel, 2003.