Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH

Open Access

A secure cross-domain authentication scheme based on threshold signature for MEC

Lei Chen[1,2], Chong Guo[3*], Bei Gong[3], Muhammad Waqas[4,5], Lihua Deng[6] and Haowen Qin[3]

Abstract

The widespread adoption of fifth-generation mobile networks has spurred the rapid advancement of mobile edge computing (MEC). By decentralizing computing and storage resources to the network edge, MEC significantly enhances real-time data access services and enables efficient processing of large-scale dynamic data on resource-limited devices. However, MEC faces considerable security challenges, particularly in cross-domain service environments, where every device poses a potential security threat. To address this issue, this paper proposes a secure cross-domain authentication scheme based on a threshold signature tailored to MEC's multi-subdomain nature. The proposed scheme employs a (t,n) threshold mechanism to bolster system resilience and security, catering to large-scale, dynamic, and decentralized MEC scenarios. Additionally, the proposed scheme features an efficient authorization update function that facilitates the revocation of malicious nodes. Security analysis confirmed that the proposed scheme satisfies unforgeability, collusion resistance, non-repudiation and forward security. Theoretical evaluation and experimental simulation verify the effectiveness and feasibility of the proposed scheme. Compared with existing schemes, the proposed scheme has higher computational performance while implementing secure authorization updates.

Keywords Authentication, Threshold signature, Authorization update, Mobile edge computing

Introduction

With the rapid development of wireless communication technology, mobile edge computing (MEC) has become an indispensable key paradigm in people's production and life [1]. Compared with traditional cloud computing, MEC reduces the reliance on central servers and focuses on deploying computational and storage resources near terminal devices [2]. As a result, MEC achieves a more decentralized and flexible computing model, improves the communication efficiency and responsiveness of application services, and satisfies the urgent requirements of terminal devices for real-time and efficient data processing. Benefiting from the widespread deployment of the Internet of Things (IoT), MEC has created a highly intelligent, real-time interactive eco-network by tightly connecting IoT terminals, edge devices and application systems [3–5]. From smart transportation and digital medical care to smart city and smart manufacturing, the comprehensive assistance of MEC has accelerated the transformation of various fields to digitalization, automation and convenience [6–8]. Moreover, MEC facilitates connectivity and convergence among different domains. However, the more rapid and widespread the adoption of MEC is, the more the ensuing security risks and pitfalls

*Correspondence:
Chong Guo
chongguo@emails.bjut.edu.cn
[1] College of Compute, National University of Defense Technology, No. 137 Yanwachi Street, Changsha 410073, Hunan, China
[2] Center for Strategic Studies, Chinese Academy of Engineering, Bingjiaokou Hutong, Beijing 100088, Beijing, China
[3] Faculty of Information Technology, Beijing University of Technology, 100 Pingleyuan, Beijing 100124, Beijing, China
[4] Faculty of Engineering and Science, School of Computing and Mathematical Sciences, University of Greenwich, Park Row, London SE10 9LS, England, UK
[5] School of Engineering, Edtih Cowan University, 270 Joondalup Drive, Joondalup 6027, WA, Australia
[6] China Telecom Research Institute, 1835 Pudong South Road, Shanghai 200122, Shanghai, China

cannot be ignored, especially in cross-domain information interactions.

In the complex MEC ecosystem, a large number of terminal devices, edge servers, and cloud service platforms are interconnected, breaking down the barriers of industries and domains, and forming a large and heterogeneous data network [9]. Terminal devices in the MEC system can easily become potential attack targets. Attackers may use terminal devices with forged identities to connect to edge servers and then invade the MEC system to tamper with data or control other devices remotely [10]. Therefore, identity authentication technology is particularly important. Identity authentication is not only a process of confirming the identity of a user or device but also a necessary step to ensure the security of the entire MEC ecosystem. An effective identity authentication mechanism can prevent unauthorized devices from accessing the MEC system, thereby reducing the risks associated with data leakage, forgery, and malicious tampering [11]. This contributes to the overall enhancement of the credibility of the MEC system. Therefore, it is highly important to research and design identity authentication technology solutions for MEC scenarios.

Password-based identity authentication and certificate-based identity authentication are currently the main identity authentication technologies and have been applied to the internet to provide reliable security guarantees [12]. Password-based identity authentication is the most basic and easy-to-implement scheme [13]. The system can authenticate the identities of devices one by one by verifying the user name and password group that is pre-configured by the device. However, passwords not only have the hidden danger of being leaked, intercepted, and guessed but also impose the burden of storage and management due to the large number of terminal devices. Compared with password-based authentication, certificate-based authentication has greater security [14]. However, compared with the traditional Internet, MEC is characterized by a large scale, multiple subdomains, dynamic networking and limited computing resources for terminal devices. This approach has led to many challenges in the direct application of traditional identity authentication technology in MEC systems, such as single points of bottleneck, certificate distribution, and unexpected offline and cross-domain authentication. Due to its large scale, the MEC system is usually divided into multiple subdomains and managed independently. Each subdomain also varies in size, computing resources, and security requirements. Coordinating and balancing the authentication cost of each subdomain while unifying the authentication scheme is the primary challenge in the cross-domain authentication process of an MEC system. Second, dynamic networking frequently changes the topology of MEC systems and participants, which greatly increases the complexity of cross-domain authentication schemes. Therefore, the cross-domain authentication scheme deployed in MEC systems must be flexible and robust enough to adapt to this dynamic network change. In addition, terminal devices in MEC system typically have limited computational and storage resources. Therefore, the cross-domain authentication scheme must fully consider the limitations of resources and minimize the impact on device performance.

To address the above challenges, this paper proposes a cross-domain authentication scheme based on a threshold signature for MEC. Different from traditional signature technology, the threshold signature allows a member of the group together to generate a signature, not by a single entity. In a threshold signature scheme, the signature key is split into multiple parts and assigned to different members of the group. A valid signature can be generated only when a sufficient number of members cooperate. This approach provides greater security and flexibility because it is not dependent on any single entity. In our scheme, the MEC system is divided into multiple subdomains, and each subdomain has one head node and multiple signature nodes. The head node is responsible for authorizing the signature node but is not directly involved in the generation of authentication credentials. The terminal devices can select a portion of signature nodes and send an authentication message to apply for signatures. If and only if the number of received signatures reaches the threshold can the terminal device synthesize its own authentication credential. The verification nodes are outside the subdomain and can use the public key of the subdomain where the terminal device is located for identity authentication. In this paper, the adoption of the threshold signature to construct an identity authentication scheme not only solves the single point of bottleneck caused by centralized authentication authority but also enhances the robustness and security of the system. In contrast to the traditional centralized authentication mechanism, our scheme uses the (t,n) threshold secret sharing mechanism to implement the threshold signature. Therefore, in our scheme, the generation of authentication certificates for terminal devices does not require the participation of all the signature nodes, but only the satisfaction of the predefined threshold. This approach provides feasible distributed authentication and greatly reduces the impact of a single signature node being offline. Therefore, our scheme is suitable for large-scale, dynamic and decentralized MEC scenarios. In addition, our scheme supports dynamic authorization to signature nodes and satisfies forward secrecy. By embedding the authorization secret value and the identity of the signature node into the authorization key, the head node

issues a unique authorization private key for the signature node. Only the signature node with the authorization private key can generate valid signatures. When the signature node is corrupted in the subdomain, the head node can revoke the malicious signature node by updating the authorization public key.

The rest of this paper is organized as follows. "Related work" section reviews related work on signatures and authentication. "Preliminaries" section introduces the essential mathematical knowledge involved in this paper. "Construction" section describes the construction of the proposed scheme. "Correctness and security" section analyses the correctness and security of the scheme. "Performance analysis" section describes the performance of the proposed scheme through theoretical evaluation and experimental simulation. "Conclusion" section gives the conclusion.

Related work

Public key cryptography was first proposed by Diffie and Hellman in 1976 [15]. Although the computational performance of public key cryptography is lower than that of symmetric cryptography, it effectively simplifies key management and successfully implements the secure distribution of keys. The advent of public key cryptography also introduced the concept of digital signatures, which provided a new method for identity authentication. In traditional digital signature algorithms, the signer constructs a public/private key pair, where the private key is usually selected randomly, while the public key is generated through specific mathematical operations or polynomial-time algorithms. The signer keeps the private key alone and can use the private key to generate a signature for the message. The public key is made public in the system and is used by other entities to verify the validity of the signature. The operations and algorithms for generating the public key are unidirectional to ensure that only the signer's public key cannot be used to guess the corresponding private key.

Zhong et al. [16] proposed a privacy-preserving identity authentication scheme based on a certificateless aggregate signature to protect the security of vehicular communication. Regrettably, this scheme cannot resist channel measurements. The conditional privacy-preserving authentication scheme proposed by Zhang et al. [17] solves the leakage problem during a side-channel attack. This scheme uses the Chinese remainder theorem to generate domain keys for the vehicles in the domain and uses elliptic curve cryptography (ECC) and modular division operations as the main operations to further reduce the computational complexity of vehicles. Subsequently, Jan and Khan [18] proposed a fast identity authentication scheme for the unmanned aerial vehicle (UAV) internet based on [17] using identity and aggregate signatures. For secure cooperation between UAVs and ground control stations on the UAV internet, Jan et al. [19] proposed a secure identity authentication scheme based on ECC. Yang et al. [20] proposed a decentralized authentication architecture for the internet of vehicle (IoV). This architecture uses a threshold signature to implement identity authentication between vehicles, and the edge node assists in computation to reduce authentication delay.

Cross-domain identity authentication To alleviate the security risk caused by complete trust in a single authority, Basin et al. [21] proposed a new public key infrastructure (PKI) architecture. This scheme builds a mandatory and public certificate information integrity verification log in the system to implement the auditing and accountability of authoritative behavior. In this scheme, all authorization behaviors of the authority are recorded in the log and are subject to the supervision of all entities in the system. This approach enhances the security of cross-domain authentication but also brings complexity and onerous certificate maintenance costs. Therefore, scheme [21] is difficult to deploy in the real world. To avoid the certificate management problem caused by the PKI-based authentication architecture, Yuan et al. [22] proposed a cross-heterogeneous domain authenticated key agreement scheme for an enterprise instant messaging system. This scheme achieves cross-domain identity authentication between the PKI domain and the identity-based cryptosystem (IBC) domain, but it requires substantial computational and communication costs and is therefore not suitable for resource-constrained IoT scenarios. To meet these lightweight requirements, Zhang et al. [22] proposed a multidomain secure authentication scheme by combining the bilinear pairing operation and the short signature algorithm. By using the inter-domain dual-signature algorithm, the scheme achieves certificateless cross-domain authentication; that is, it avoids the potential hidden key leakage of identity-based authentication and solves the certificate distribution and single-point bottleneck problems caused by certificate-based authentication. However, the authentication of this scheme depends on the trusted authority outside the domain and lacks an effective revocation mechanism. Jia et al. [23] proposed a fast response cross-domain authentication scheme for the IoT based on identity passwords and threshold signatures. To relieve the management burden of multidomain certificates and achieve decentralized identity authentication, this scheme uses identity symbols to replace digital certificates issued by an authority. Gan [24] proposed a secure threshold signature scheme based on the dual-pair vector space and proved the fully adaptive security of the scheme in the standard model using the dual-form signature [25]. This

scheme has high security and high performance and has the potential to be deployed in the IoT for device authentication. In a real IoT environment, in addition to avoiding a single bottleneck and alleviating the burden of certificate management, it is also necessary to implement control over malicious authorities. However, all existing schemes lack the functionality to securely and effectively revoke malicious authorizations, which is the problem addressed in this paper.

Preliminaries

Bilinear pairing

Given two multiplicative cyclic groups G and G_T of order p. g is the generator of G. is a pairing operation. If e satisfies:

1) Bilinear: For $\forall a, b \in Z_p$ and $\forall u, v \in G$, $e(u^a, v^b) = e(u, v)^{ab}$;
2) Non-degeneracy: $e(g, g) \neq 1_{G_T}$, where 1_{G_T} is the unit element of G_T;
3) Computability: For $\forall u, v \in G$, there is an efficient algorithm that can calculate $e(u, v)$.

Then, we call e a c.

Discrete logarithm problem

Suppose that G is a multiplicative cyclic group of order p. g is the generator of G. A known binary group (g, g^x) solving for $x \in Z_p$ is a discrete logarithm problem and is denoted as $DL_g(g, g^x) = x$. The discrete logarithm problem is currently unsolvable. Therefore, given g and g^x, no attacker can obtain x.

(t, n) threshold secret sharing

Suppose there are n participants sharing secrets. The set of participants is defined as $P = \{P_1, P_2, \cdots, P_n\}$. The secret value to be shared is defined as s. Each participant P_i holds a split value s_i of the secret value s, where $i \in [1, n]$. If a secret sharing mechanism satisfies:

1) Any t or more participants can recover the secret value s by using their secret split value;
2) Any less than t participants cannot obtain any information about the secret value s by the secret split value they hold.

We call this secret mechanism a (t, n) threshold secret sharing mechanism, where t is called the threshold.

Lagrangian interpolation method

Suppose there is a polynomial function $f(x)$. $(x_0, f(x_0)), (x_1, f(x_1)), \cdots, (x_n, f(x_n))$ are the $n+1$ known points that $f(x)$ passes through, where $x_0, x_1, \cdots x_n \in Z_p$. The expression for $f(x)$ can be found using the Lagrangian interpolation method as shown in Eq. (1).

$$f(x) = \sum_{i=0}^{n} \left(f(x_i) \prod_{j=0, j \neq i}^{n} \frac{x - x_j}{x_i - x_j} \right) \tag{1}$$

Construction

The architecture of our scheme is shown in Fig. 1. The proposed scheme includes four types of entities: head node, signature node, terminal device and verification

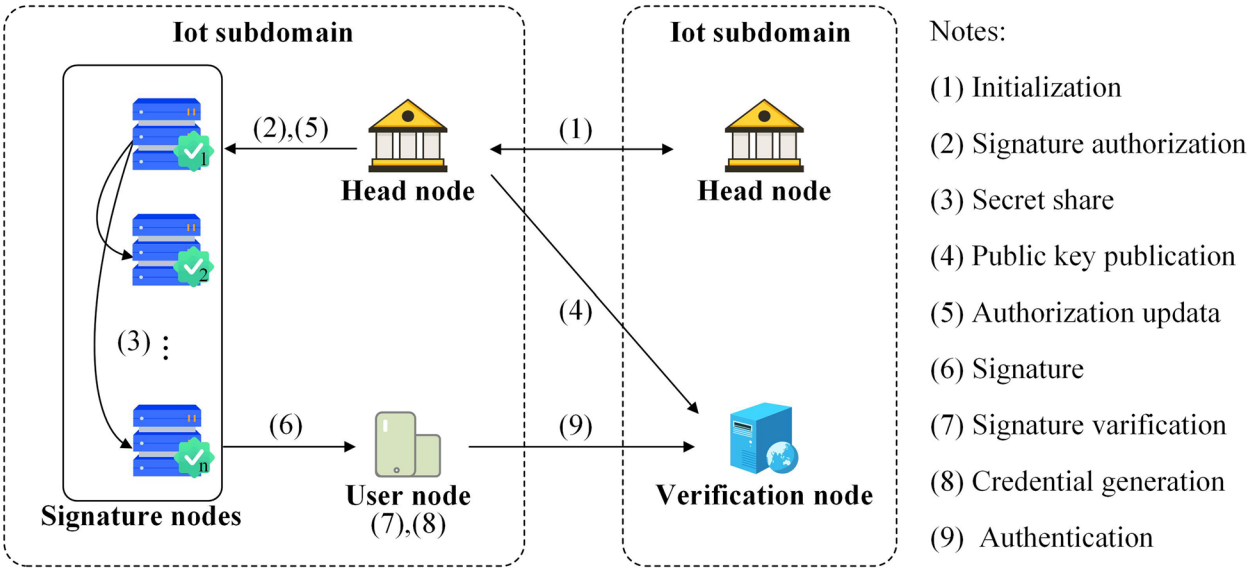

Fig. 1 Scheme architecture

node. Let there be one head node and n signature nodes in an MEC subdomain. $i \in [1, n]$ represents the identity of the signature node. t represents the signature threshold for the MEC subdomain. Our scheme included the following 9 steps. Additionally, the main symbols used in this section are defined in Table 1.

Initialization

The head nodes of all subdomains negotiate together to generate the public parameters of the MEC system. First, the head nodes negotiate to select multiplicative cyclic groups G and G_T of order p. The generator of G is g. Then, they choose a bilinear pairing $e : G \times G \rightarrow G_T$ and a one-way hash function $H : \{0,1\}^* \rightarrow Z_p$. The hash function is used to generate a digest value for the authentication message. Finally, the public parameter $PP = \{p, G, G_T, g, e, h\}$ is published for the entire MEC system.

Signature authorization

The head node randomly selects $\alpha, \beta, \mu \in Z_p$ as its secret value. Specifically, α and β are used to authorize the signature node. μ is used to implement the authorization update of the signature node. The head node set $PK_1 = g^{\beta}$ as the component of the public key for the subdomain. Subsequently, the head node selects $t-1$ random values $a_1, a_2, \cdots, a_{t-1} \in Z_p$ as the coefficients, and then constructs a polynomial $f(x) = \alpha + \beta + a_1 x + \ldots + a_{t-1} x^{t-1}$ of order $t-1$. Then, for each signature node in the subdomain, the head node calculates $d_i = f(i)$ as the secret split value and then generates the initial authorization private key $\delta_i = {PK_1}^{d_i + \mu}$, and the part of the signature node's public key $\Gamma_{i,1} = g^{d_i}$, where $i \in [1, n]$. Finally, the head node sends $(\delta_i, \Gamma_{i,1})$ to the corresponding signature node via the secret channel and keeps (α, β, μ) secret.

Secret sharing

Each signature node first performs the following steps:

1) Selects $\gamma_i, z_i \in Z_p$ as its secret values.
2) Calculates $\Gamma_{i,2} = g^{\gamma_i}$ as the part of its public key.
3) Constructs a polynomial $f_i(x) = z_i + a_{i,1} x + \ldots + a_{i,t-1} x^{t-1}$ to implement threshold secret sharing for z_i, where $a_{i,1}, a_{i,2}, \cdots, a_{i,t-1} \in Z_p$.
4) Calculates $f_i(j)$ as the secret split value to the signature node j via the secret channel, where $\forall j \in [1, n]$.
5) Calculates $Z_i = g^{z_i}$ and $\left\{A_{i,l} = g^{a_{i,l}} : l \in [1, t-1]\right\}$, and then publishes them in the MEC subdomain. Z_i and $A_{i,l}$ are used to check the correctness of the secret split value $f_i(j)$.

Subsequently, for $\forall l \in [1, n]$, the signature node i checks the correctness of $f_l(i)$ with Eq. (2). After receiving n correct secret split values $\left\{f_l(i) : l \in [1, n]\right\}$, the signature node i can calculate $v_i = \sum_{l=1}^{n} f_l(i)$ as its secret shared value and $\Gamma_{i,3} = g^{v_i}$ as the last part of its public key. Finally, the signature node i generates and publishes its public key $\Gamma_i = \left\{\Gamma_{i,1}, \Gamma_{i,2}, \Gamma_{i,3}\right\}$ in the MEC subdomain.

$$g^{f_i(j)} \stackrel{?}{=} Z_i \prod_{l=1}^{t-1} \left(A_{i,l}\right)^{j^l} \tag{2}$$

Public key publication

The head node first collects $\{Z_i : i \in [1, n]\}$ published by all signature nodes. Subsequently, the head node calculates the components of the subdomain public key PK_2 and PK_3 with Eq. (3). Finally, the head node publishes

Table 1 Definitions of symbols in "Construction" section

Symbol	Definition
n	Number of signature nodes
t	Size of signature threshold
(α, β, μ)	Secret values of the head node
d_i	Secret split value generated by the header node for the signature node i
δ_i	Initial authorization private key of the signature node i
(γ_i, z_i)	Secret values of the signature node i
$f_i(j)$	Secret split value generated by the signature node i for the signature node j
v_i	Secret shared value of the signature node i
Γ_i	Public key of the signature node i
PK	Public key of the subdomain
m	Authentication message
(η_i, σ_i)	Signature
(η, σ, Γ)	Authentication credential

the public key $PK = \{PK_1, PK_2, PK_3\}$ in the MEC system. Specifically, PK_1 is used to ensure that the signature node receives the initial authorization from the header node. PK_2 is used to prevent a signature node that has been removed from generating a valid signature. PK_3 is used to check that the number of valid signatures generated for the authentication certificate reaches the threshold.

$$\begin{aligned} PK_2 &= g^{\mu} \\ PK_3 &= g^{\alpha+\beta} \prod_{i=1}^{n} Z_i = g^{\alpha+\beta+\sum_{i=1}^{n} z_i} \end{aligned} \tag{3}$$

Authorization update

The head node randomly selects $\mu' \in Z_p$ as the secret value for the authorization update. Then, the head node calculates $PK'_2 = g^{\mu'}$ as the new authorization public key and publishes it in the MEC system. Subsequently, for each signature node, the head node calculates a new authorization private key $\delta_i{}' = \delta_i {PK_1}^{\mu'-\mu} = {PK_1}^{d_i+\mu'}$ and sends it to the corresponding signature node via the secret channel. By checking the Eq. (4), the signature node can verify whether the authorization component generated by the head node is correct. If the equation holds, $\delta_i{}'$ is correct, and the signature node saves $\delta_i{}'$. Otherwise, the signature node reobtains the authentication component from the head node.

$$e(\delta_i{}', g) \stackrel{?}{=} e(\Gamma_{i,1} PK'_2, PK_1) \tag{4}$$

Signature

The terminal device constructs an authentication message $m = m_{sd}||m_{ts}(||m_{id}||m_{other})$, where m_{sd} represents the subdomain where the node is located, m_{ts} represents the timestamp when the authentication message was generated, m_{id} represents the identity of the node, and m_{other} represents other content that needs to be provided. m_{sd} and m_{ts} are the necessary information for authentication. They can indicate the MEC subdomain to which the terminal device belongs at a certain time. m_{id} and m_{other} are the optional information. If the terminal device is unwilling to disclose its identity to the verification node, m_{id} will not be provided. Then, the terminal device sends the authentication message m to the online signature nodes in the subdomain to obtain signatures.

After receiving the authentication message, the signature node first checks whether the timestamp m_{ts} is within the validity period. If m_{ts} has expired, the signature node refuses to sign and returns false to the terminal device. Otherwise, the signature node generates the signature (η_i, σ_i) with Eq. (5) and returns it to the terminal device.

$$\begin{aligned} \eta_i &= \prod_{j=1, j\neq i}^{t} (j/(j-1)) \\ \sigma_i &= \delta_i^{\eta_i} {PK_1}^{\gamma_i H(m) + v_i \eta_i} \end{aligned} \tag{5}$$

Signature verification

The terminal device verifies the correctness of the signature (η_i, δ_i) by the Eq. (6). If the equation holds, the signature is correct. The terminal device will save the signature and use it to generate an authentication credential. Otherwise, a new signature request will be sent to the signature node.

$$e(\sigma_i, g) \stackrel{?}{=} e\left(\Gamma_{i,2}^{H(m)} \cdot \left(\Gamma_{i,1} \cdot \Gamma_{i,3} \cdot PK_2\right)^{\eta_i}, PK_1\right) \tag{6}$$

Credential generation

After receiving t correct signatures, the terminal device calculates $\eta = \sum_{i=1}^{t} \eta_i$, $\sigma = \prod_{i=1}^{t} \sigma_i$ and $\Gamma = \prod_{i=1}^{t} \Gamma_{i,2}$. Finally, the terminal device generates the authentication credential (η, σ, Γ).

Authentication

After receiving the authentication message and authentication credential from the terminal device, the verification node first splits the authentication message by $\{m_{id}, m_{sd}, m_{ts}, m_{other}\} = m$. Then, the verification node checks whether the timestamp m_{ts} is within the validity period. If the timestamp is valid, the verification node further verifies the authenticity of the authentication credential (η, σ, Γ). Otherwise, the verification node returns false to the terminal device to indicate that the authentication has not passed. Before authentication, the authentication node is informed about the MEC subdomain to which the terminal device belongs according to m_{sd}. Subsequently, the authentication node obtains the public key $PK = \{PK_1, PK_2, PK_3\}$ of the subdomain. Finally, the verification node verifies the authentication credential by the Eq. (7). If the equation holds, the identity of the terminal device is true.

$$e(\sigma, g) \stackrel{?}{=} e\left(\Gamma^{H(m)} \cdot PK_3 \cdot {PK_2}^{\eta}, PK_1\right) \tag{7}$$

Correctness and security

Correctness

The correctness of the proposed scheme includes the correctness of the signature and the correctness of the authentication credential. The establishment of the equation $e(\delta_i{}', g) = e(\Gamma_{i,1} PK'_2, PK_1)$ proves that the signature is correct. This can be proven by the following derivation formula.

$$
\begin{aligned}
& e(\sigma_i, g) \\
&= e\left(\delta_i^{\eta_i} {PK_1}^{\gamma_i H(m)+\nu_i\eta_i}, g\right) \\
&= e\left({PK_1}^{\eta_i(d_i+\mu)} {PK_1}^{\gamma_i H(m)+\nu_i\eta_i}, g\right) \\
&= e\left(g^{\gamma_i H(m)} \cdot g^{\eta_i(d_i+\nu_i+\mu)}, PK_1\right) \\
&= e\left(\left(g^{\gamma_i}\right)^{H(m)} \cdot \left(g^{d_i} \cdot g^{\nu_i} \cdot g^{\mu}\right)^{\eta_i}, PK_1\right) \\
&= e\left(\Gamma_{i,2}^{H(m)} \cdot \left(\Gamma_{i,1} \cdot \Gamma_{i,3} \cdot PK_2\right)^{\eta_i}, PK_1\right)
\end{aligned}
\tag{8}
$$

The correctness of the authentication credential can be given by the equation $e(\sigma, g) = e(\sigma, g) = e\left(\Gamma^{H(m)} \cdot PK_3 \cdot {PK_2}^{\eta}, PK_1\right)$. The specific formula is derived as follows.

$$
\begin{aligned}
& e(\sigma, g) \\
&= e\left(\prod_{i=1}^{t} \sigma_i, g\right) \\
&= e\left(\prod_{i=1}^{t} \delta_i^{\eta_i} {PK_1}^{\gamma_i H(m)+\nu_i\eta_i}, g\right) \\
&= e\left(\prod_{i=1}^{t} {PK_1}^{\gamma_i H(m)+\eta_i(\nu_i+d_i+\mu)}, g\right) \\
&= e\left(\prod_{i=1}^{t} g^{\gamma_i}, PK_1\right)^{H(m)} e\left(g^{\sum_{i=1}^{t}\eta_i\mu}, PK_1\right) e\left(g^{\sum_{i=1}^{t}\eta_i d_i+\eta_i\nu_i}, PK_1\right) \\
&= e\left(\prod_{i=1}^{t} \Gamma_{i,2}, PK_1\right)^{H(m)} e\left(g^{\mu\sum_{i=1}^{t}\eta_i}, PK_1\right) e\left(g^{f(0)+\sum_{i=1}^{n} f_i(0)}, PK_1\right) \\
&= e(\Gamma, PK_1)^{H(m)} e\left({PK_2}^{\eta}, PK_1\right) e\left(g^{\alpha+\beta+\sum_{i=1}^{n} z_i}, PK_1\right) \\
&= e\left(\Gamma^{H(m)} \cdot PK_3 \cdot {PK_2}^{\eta}, PK_1\right)
\end{aligned}
\tag{9}
$$

Unforgeability

Unforgeability is the most critical security attribute of our scheme. The unforgeability includes the unforgeability of the signature and the unforgeability of the authentication credential. In terms of the unforgeability of the signature, each valid signature embeds the authorization private key $\delta_i = {PK_1}^{d_i+\mu}$ assigned by the head node, the private key γ_i chosen by itself, and the secret shared value $\nu_i = \sum_{l=1}^{n} f_l(i)$ among the signature nodes. These secrets are contained in the public key $\Gamma_i = \{\Gamma_{i,1}, \Gamma_{i,2}, \Gamma_{i,3}\}$ of the signature node. However, there is no probabilistic polynomial time (PPT) attacker obtaining $(\delta_i, \gamma_i, \nu_i)$ from the public key Γ_i, because the discrete logarithm problem is unsolvable. Additionally, the head node knows δ_i. However, γ_i and ν_i are only saved by the signature node. Therefore, the head node cannot forge any valid signatures. In terms of the unforgeability of the authentication credential, an attacker who cannot generate a valid signature is even less likely to forge a valid authentication credential, since the authentication credential is aggregated from t signatures.

Resistance to collusion attack

In our scheme, the authentication credential is based on the signatures generated by t signature nodes. Resistance to collusion attacks can ensure that valid authentication certificates cannot be forged through collusion when the number of signature nodes is less than t. A valid signature embeds the secret value α, β of the head node and the secret value z_i of each signature node. Our scheme uses a polynomial of order $t-1$ to implement secret splitting. The variable term in the polynomial cannot be eliminated to recover the secret value as a constant term when fewer than t signature nodes collude together. Therefore, the forged authentication credential cannot satisfy $\sum_{i=1}^{t} \eta_i d_i + \eta_i \nu_i = f(0) + \sum_{i=1}^{n} f_i(0) = \alpha + \beta + \sum_{i=1}^{n} z_i$, which will result in $e(\sigma, g) \neq e\left(\Gamma^{H(m)} \cdot PK_3 \cdot {PK_2}^{\eta}, PK_1\right)$ during authentication.

Non-repudiation

Non-repudiation ensures that the signature node cannot deny the signature generated by itself or the authentication credentials in which it participated. When malicious signature nodes appear in the MEC subdomain, non-repudiation can cause signatures and authentication credentials to be used as evidence for tracking and auditing them. In our scheme, the signature node must publish its public key $\Gamma_i = \{\Gamma_{i,1}, \Gamma_{i,2}, \Gamma_{i,3}\}$ in the subdomain, where $\Gamma_{i,1}$ is generated by the head node based on the secret split value $d_i = f(i)$, $\Gamma_{i,2}$ is generated by the signature node based on its own private key, and $\Gamma_{i,3}$ is generated based on the secret shared value $\nu_i = \sum_{l=1}^{n} f_l(i)$ among the signature nodes. Therefore, the public key of each signature node is unique and bound by its own identity. In addition, unforgeability prevents other nodes from forging signatures that match the public key. Therefore, the equation $e(\sigma_i, g) = e\left(\Gamma_{i,2}^{H(m)} \cdot \left(\Gamma_{i,1} \cdot \Gamma_{i,3} \cdot PK_2\right)^{\eta_i}, PK_1\right)$ can not only verify the correctness of signatures but also be used to trace the signature node with the signature. The signature node cannot deny the tracking result.

Forward security

Forward secrecy can ensure that each time the head node executes an authorization update, the previously distributed authorization private key can be invalidated, thus realizing the dynamic management of the signature nodes. In our scheme, for each authorization update, the head node randomly selects μ as the secret value, and publishes the authorization public key $PK_2 = g^{\mu}$. At the same time, the head node also needs to update the authorization private key for each signature node to ensure that it matches the authorization public key. The authorization private key $\delta_i = {PK_1}^{d_i+\mu}$ not only embeds the user's secret split value d_i but also blinds the authorization secret value μ. If the authorization private key is

Table 2 Definitions of symbols in "Performance analysis" section

Symbol	Definition
E	Exponential operation in $\mathbb{G}$
P	Bilinear pairing operation
n	Number of signature nodes
t	Size of threshold
$\|\mathbb{G}\|$	Bit length of the element in $\mathbb{G}$
$\|\mathbb{Z}_p\|$	Bit length of the element in $\mathbb{Z}_p$
$\|m\|$	Bit length of the authentication message m

Table 3 Theoretical performance of the proposed schemes

Phase	Computation	Communication
Initialization	-	$\|G\| + \|Z_p\|$
Signature authorization	$2nE$	$2n\|G\|$
Secret sharing	$(t+2)E$	$(t+3)\|G\|$
Public key publication	$3E$	$3\|G\|$
Authorization update	$2E$	$(n+1)\|G\|$
Signature	$2E$	$\|G\| + \|Z_p\|$
Signature verification	$2P + 2E$	-
Credential generation	-	$2\|G\| + \|Z_p\| + \|m\|$
Authentication	$2P + 2E$	-

not updated, the signature for the terminal device will fail to pass the verification with the equation $e(\sigma_i, g) \stackrel{?}{=} e\left(\Gamma_{i,2}^{H(m)} \cdot \left(\Gamma_{i,1} \cdot \Gamma_{i,3} \cdot PK_2\right)^{\eta_i}, PK_1\right)$. Moreover, because the discrete logarithm problem is unsolvable, malicious signature nodes cannot infer the value of μ from the authorization public key, much less to forge a valid authorization private key.

Performance analysis

In this section, we analyze the performance of the proposed scheme from two perspectives: theoretical evaluation and experimental simulation. In addition, Table 2 gives definitions of the symbols used in this section.

Theoretical evaluation

The theoretical evaluation analyzes the computational performance and communication performance of our scheme. In terms of computational performance, the efficiency of the algorithms is demonstrated by counting the main mathematical operations needed. In cryptographic schemes based on bilinear pairing, the computational complexity of bilinear pairing is usually the highest, followed by that of exponential pairing. Compared with the above two operations, other mathematical operations, such as constant operation, hash operation and multiplication operation, can be ignored. Therefore, we count the number of bilinear pairing operations and exponential operations required in the nine algorithms in Table 3. In terms of communication performance, we evaluated the data sending cost, that is, how much bit-length data need to be sent for each algorithm.

Table 3 shows that the performances of the signature authorization and authorization update algorithms run by the head node are the lowest. The signature authorization algorithm needs to execute $2n$ exponential operations and send $2n|G|$ bit data. This is because the head node needs to generate a public key $\Gamma_{i,1} = g^{d_i}$ and an authorization private key $\delta_i = PK_1{}^{d_i+\mu}$ for each signature node. Similarly, during authorization update, the head node needs to perform only two exponential operations to generate a new authorization public key $PK'_2 = g^{\mu'}$ and the component $PK_1{}^{\mu'-\mu}$ is used to update the authorization private key. Then, the head node executes the fast multiplication operation to generate a new authorization private key $\delta_i{}' = \delta_i PK_1{}^{\mu'-\mu}$ for the signature node. The new authorization public key needs to be published in the MEC system, and the new authorization private key needs to be sent to each signature node secretly. Therefore, the head node needs to send $(n+1)|G|$ bit data to complete the authorization update of the subdomain. Fortunately, the signature authorization algorithm only needs to be run once the subdomain is created, and the authorization update algorithm only needs to be run in stages. Therefore, even the computational performance, which is linearly related to the number of signature nodes, is acceptable. In addition, the secret sharing algorithm run by signature nodes requires the same linear exponential operations and data communication. This is because, to achieve secret sharing among the signature nodes, the polynomial function $f_i(x)$ needs to be composed of $t-1$ coefficients to generate validation items $A_{i,l} = g^{a_{i,l}}$ and publish them in the subdomain. However, the secret sharing only needs to be performed once. Therefore, the signature node does not impose a heavy computational burden. Except for the above three algorithms, the computational performances of the other algorithms all reach a constant level, which is very efficient.

Experimental simulation

The experimental environment was a Lenovo desktop computer with an 11th generation Intel processor and 16 GB of memory. Moreover, Ubuntu 16.04 was used as the operating system. The experimental programs were written in Python 3.5 based on the charm-crypto architecture. In the experimental simulations, we set the elliptic curve to Type A, that is, $E(F_q) : y^2 = x^3 + x$, where the length of q is 512 bits. The groups G and G_T involved in

bilinear pairing are the p-order subgroup of $E(F_q)$. Therefore, in terms of communication cost, $|Z_p| = 160bits$ and $|G| = |G_T| = 1024bits$. In addition, the one-way hash function $H : \{0,1\}^* \rightarrow Z_p$ used in the proposed scheme is designed based on the hash function in the charm-crypto architecture.

To show the performance of the proposed scheme more intuitively, we compared it with the IRBA scheme [23] and the FASTS scheme [24]. Figure 2 illustrates the signature time comparison of the three schemes. From Fig. 2, we can see that the signature times of the FASTS scheme and our scheme remain almost constant, while the signature time of the IRBA scheme is linearly correlated with the number of thresholds. The reason is that the signature algorithm of the IRBA scheme needs to be executed $3t$ times exponential operation in G_T. However, the exponential operation in G_T is much faster than the exponential operation in G, approximately one-tenth of the computational complexity. Therefore, the signature time of the FASTS scheme increases only slowly with increasing threshold size. In addition, our scheme requires less signature time than the FASTS scheme because our scheme only needs to perform two exponential operations to generate the signature (η_i, δ_i), while the FASTS scheme needs to perform three exponential operations.

Figure 3 shows the verification times of the three schemes. Obviously, the verification times of the three schemes are independent of the threshold. The verification time of our scheme is shorter than that of the IRBA scheme but longer than that of the FASTS scheme. In our scheme, the head node can revoke the signature node by stopping the update of the authorization private key. In addition, our scheme satisfies forward security. Therefore, our scheme needs to perform additional operations when verifying signatures. For the enhanced security of the scheme, this computational cost is important.

Figure 4 compares the credential generation times of the three schemes. From Fig. 4, we can see that the generation times of the three schemes increase with the size of the threshold. The generation time of IRBA scheme is the highest, and the generation times of the FASTS scheme and our scheme are almost the same. The main operation for generating authentication credentials is the fast multiplication operation. The computation time of the credential generation algorithm is far lower than that of the signature, signature verification and authentication algorithms and does not exceed 150 us when the threshold size is less than or equal to 10. This indicates that terminal devices need to bear a very small computational burden, and can be applied to terminal devices with resource constraints.

Figure 5 shows the efficiency of authentication. Consistent with the results of the theoretical analysis, the authentication times of the three schemes are not affected by the size of the threshold. The authentication times for the FASTS scheme and our scheme are significantly lower than that of the IRBA scheme. The authentication time for our scheme is approximately 4300 us, which is slightly greater than the 3100 us required for the FASTS scheme.

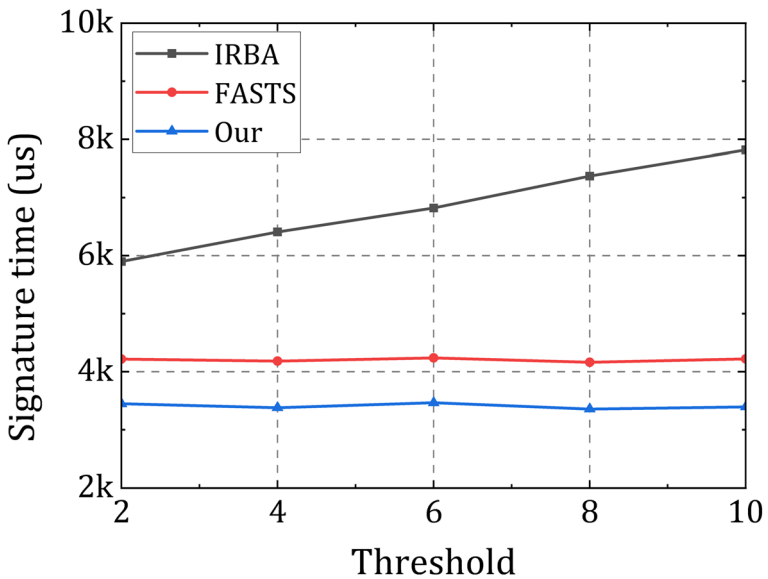

Fig. 2 Comparison of the signature time

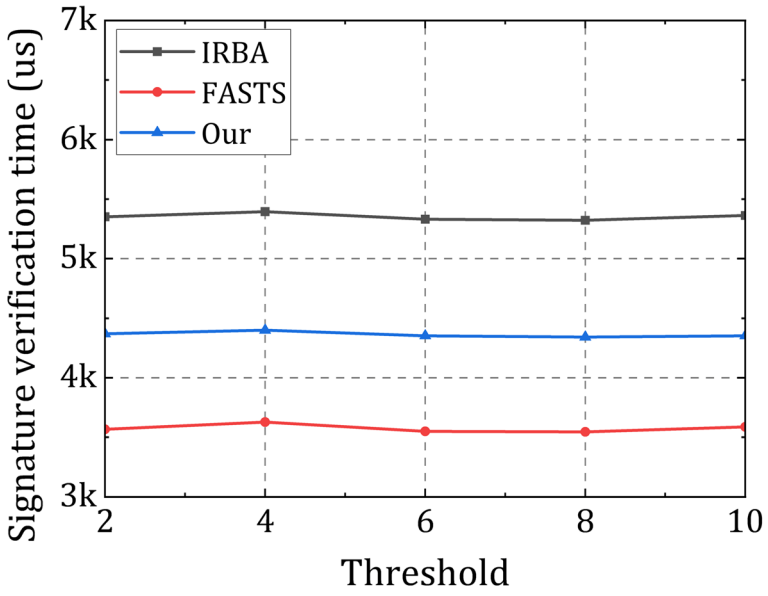

Fig. 3 Comparison of the signature verification time

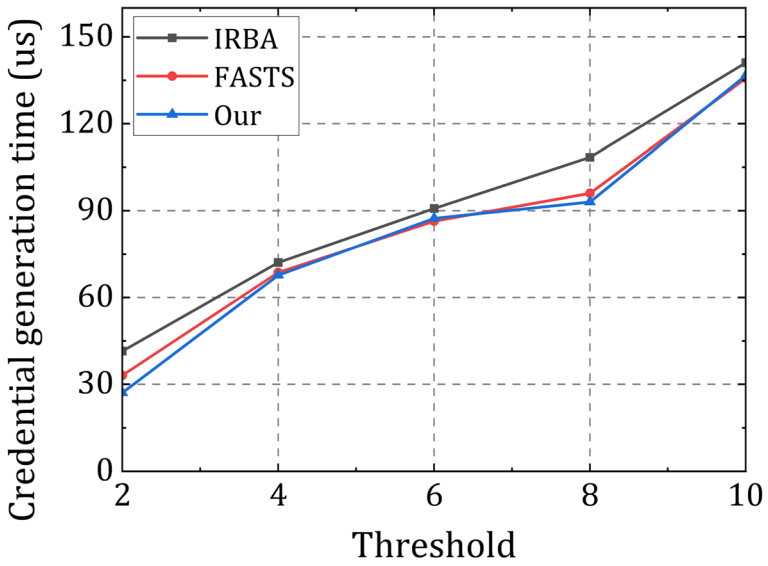

Fig. 4 Comparison of the credential generation time

Conclusion

In this paper, we propose a cross-domain authentication scheme based on threshold signatures for MEC. The proposed scheme uses the (t,n) threshold secret sharing mechanism to achieve decentralized signatures and avoid single-point bottlenecks. The authorization update function enables the head node to remove malicious signature nodes in subdomains securely and efficiently. The security analysis proves that the proposed scheme satisfies correctness, unforgeability, resistance to collusion attacks, non-repudiation and forward security. We also evaluate the performance of the proposed scheme and compare it with other schemes via simulation.

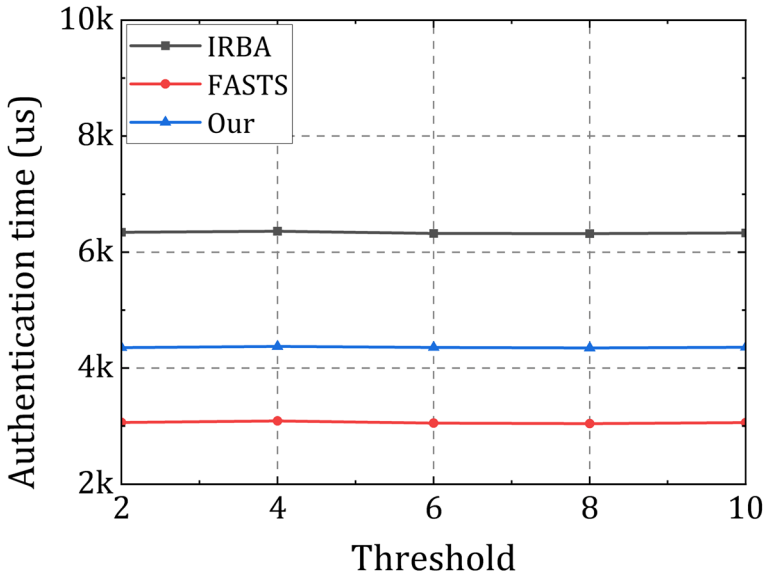

Fig. 5 Comparison of the authentication time

Acknowledgements
This work was supported by the National Natural Science Foundation of China under Grant 61971014.

Authors' contributions
The authors' contributions are as follows: Conceptualization, Software, Validation, Formal analysis, Writing - original draft: Lei Chen; Software, Formal analysis, Investigation, Writing - original draft, Writing - review editing: Chong Guo, Bei Gong; Writing - review editing, Visualization, Supervision, Project administration: Muhammad Waqas, Lihua Deng, and Haowen Qin.

Funding
This work was supported by the National Natural Science Foundation of China under Grant 61971014.

Availability of data and materials
No datasets were generated or analysed during the current study.

Declarations

Ethics approval and consent to participate
This declaration is not applicable.

Competing interests
The authors declare no competing interests.

Received: 19 January 2024 Accepted: 12 March 2024
Published online: 22 March 2024

References

1. Zhou W, Fan L, Zhou F, Li F, Lei X, Xu W, Nallanathan A (2023) Priorityaware resource scheduling for uav-mounted mobile edge computing networks. IEEE Trans Veh Technol 72(7):9682–9687 (2023). https://doi.org/10.1109/TVT.2023.3247431
2. Luo R, Jin H, He Q, Wu S, Xia X (2023) Enabling balanced data deduplication in mobile edge computing. IEEE Trans Parallel Distrib Syst 34(5):1420–1431
3. Qi L, Liu Y, Zhang Y, Xu X, Bilal M, Song H (2022) Privacy-aware point-of-interest category recommendation in internet of things. IEEE Internet Things J 9(21):21398–21408. https://doi.org/10.1109/JIOT.2022.3181136
4. Xie T (2023) Campus iot system and students' employment education innovation basedon mobile edge computing. Soft Comput 27(14):10263–10272. https://doi.org/10.1007/s00500-023-08288-5
5. Liu Y, Zhou X, Kou H, Zhao Y, Xu X, Zhang X, Qi L (2023) Privacy-preserving point-of-interest recommendation based on simplified graph convolutional network for geological traveling. ACM Trans Intell Syst Technol. https://doi.org/10.1145/3620677
6. Wang L, Deng X, Gui J, Chen X, Wan S (2023) Microservice-oriented service placement for mobile edge computing in sustainable internet of vehicles. IEEE Trans Intell Transp Syst 24(9):10012–10026. https://doi.org/10.1109/TITS.2023.3274307
7. Ma Z, Ma J, Miao Y, Liu X, Choo KKR, Yang R, Wang X (2020) Lightweight privacy-preserving medical diagnosis in edge computing. IEEE Trans Serv Comput 15(3):1606–1618
8. Khanh QV, Nguyen VH, Minh QN, Van AD, Le Anh N, Chehri A (2023) An efficient edge computing management mechanism for sustainable smart cities. Sustain Comput Inf Syst 38(100):867
9. Guo K, Yang M, Zhang Y, Cao J (2019) Joint computation offloading and bandwidth assignment in cloud-assisted edge computing. IEEE Trans Cloud Comput 10(1):451–460
10. Zheng X, Li M, Shah SBH, Do DT, Chen Y, Mavromoustakis CX, Mastorakis G, Pallis E (2022) Enhancing security-problem-based deep learning in mobile edge computing. ACM Trans Internet Technol 22(2):1–15
11. Mahmood K, Ayub MF, Hassan SZ, Ghaffar Z, Lv Z, Chaudhry SA (2022) A seamless anonymous authentication protocol for mobile edge computing infrastructure. Comput Commun 186:12–21
12. Saqib M, Moon AH (2023) A systematic security assessment and review of internet of things in the context of authentication. Comput Secur 125:103053
13. Al Kabir MA, Elmedany W (2022) An overview of the present and future of user authentication. In: 2022 4th IEEE Middle East and North Africa COMMunications Conference (MENACOMM), IEEE, p 10–17
14. Astorga J, Barcelo M, Urbieta A, Jacob E (2022) Revisiting the feasibility of public key cryptography in light of iiot communications. Sensors 22(7):2561
15. Hellman M (1976) New directions in cryptography. IEEE Trans Inf Theory 22(6):644–654
16. Zhong H, Han S, Cui J, Zhang J, Xu Y (2019) Privacy-preserving authentication scheme with full aggregation in vanet. Inf Sci 476:211–221

17. Zhang J, Cui J, Zhong H, Chen Z, Liu L (2019) Pa-crt: Chinese remainder theorem based conditional privacy-preserving authentication scheme in vehicular ad-hoc networks. IEEE Trans Dependable Secure Comput 18(2):722–735
18. Jan SU, Khan HU (2021) Identity and aggregate signature-based authentication protocol for iod deployment military drone. IEEE Access 9:130247–130263
19. Jan SU, Abbasi IA, Algarni F, Khan AS (2022) A verifiably secure ecc based authentication scheme for securing iod using fanet. IEEE Access 10:95321–95343
20. Yang A, Weng J, Yang K, Huang C, Shen X (2020) Delegating authentication to edge: a decentralized authentication architecture for vehicular networks. IEEE Trans Intell Transp Syst 23(2):1284–1298
21. Basin D, Cremers C, Kim THJ, Perrig A, Sasse R, Szalachowski P (2016) Design, analysis, and implementation of arpki: an attack-resilient public-key infrastructure. IEEE Trans Dependable Secure Comput 15(3):393–408
22. Yuan C, Zhang W, Wang X (2017) Eimakp: Heterogeneous cross-domain authenticated key agreement protocols in the eim system. Arab J Sci Eng 42:3275–3287
23. Jia X, Hu N, Su S, Yin S, Zhao Y, Cheng X, Zhang C (2020) Irba: An identity-based cross-domain authentication scheme for the internet of things. Electronics 9(4):634
24. Gan Y (2021) A fully adaptively secure threshold signature scheme based on dual-form signatures technology. Secur Commun Netw 2021:1–11
25. Gerbush M, Lewko A, O'Neill A, Waters B (2012) Dual form signatures: an approach for proving security from static assumptions. In: Advances in Cryptology–ASIACRYPT 2012: 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings 18. Springer, p 25–42

Publisher's Note