Received: 8 February 2022 | Accepted: 4 April 2022

DOI: 10.1002/cpe.7198

RESEARCH ARTICLE

WILEY

Identity authentication for edge devices based on zero-trust architecture

Haiqing Liu[1] | Ming Ai[1] | Rong Huang[1] | Rixuan Qiu[2] | Yuancheng Li[1]

[1]School of Control and Computer Engineering, North China Electric Power University, Beijing, China

[2]State Grid Jiangxi Information Telecommunication Company, Nanchang, China

Correspondence
Yuancheng Li, School of Control and Computer Engineering, North China Electric Power University, 2 Beinong Road, Changping District, Beijing, China.
Email: ncepua@163.com

Funding information
The State Grid Jiangxi Information & Telecommunication Company Project, Grant/Award Number: 52183520007V

Summary

Device identity authentication is the first line of defense for edge computing security mechanisms. Many authentication schemes are often accompanied by high communication and computational overhead. In addition, due to the continuous enhancement of network virtualization and dynamics, the security requirements for logical boundaries of many enterprise information systems "cloudification," and the huge data security challenges faced by enterprise core assets, all make the original εone-time authentication, all the wayε trust model no longer reliable. Therefore, the paper proposes a local identity authentication and roaming identity authentication protocol based on a zero-trust architecture. First, we propose a revocable group signature scheme, the expiration time is bound to the key of each edge terminal device. According to this solution, since the identity authentication token generated by the expired key is invalid, it does not need to be included in the revocation list, which improves the efficiency of revocation checking. Compared with the current identity authentication protocol, this article not only builds a model based on the zero trust architecture, effectively solves the shortcomings of the network security protection architecture, but also considers the unforgeability of the expiration time, and realizes effective revocation and more efficient identity authentication.

KEYWORDS

group signature, identity authentication, revocation, roaming authentication, zero trust

1 | INTRODUCTION

With the in-depth development and wide application of the Internet of Things and big data technologies, a series of applications that benefit the people, such as smart transportation and smart medical care, have emerged driven by innovation. However, with the rapid increase in the number of devices connected to the network, the data transmitted in the network is also increasing geometrically, and traditional cloud computing centers can no longer meet the low-latency, dense network access and service requirements. Therefore, using scattered computing and storage resources at the edge of the network to perform distributed data processing tasks and relieve the load of the cloud computing center will become the main direction of the development of the Internet of Things. In this case, edge computing came into being.

Edge computing, as a new computing model centered on network edge devices, supports the migration of cloud-centric tasks the edge of the network and brings services closer to the edge and in a wider range. By deploying edge terminal devices, services can reside on edge devices, which can ensure efficient network operations and service delivery while processing massive amounts of data. However, the new features of edge computing make the data security and privacy protection of mobile terminal users face new challenges. On the one hand, when mobile terminal users unload tasks to mobile edge server through wireless channel, they may be eavesdropped by attackers, leading to user privacy leakage. On the other hand, mobile terminals have limited storage and computing capabilities and are based on traditional public key infrastructure. Because of the large

resource consumption and long response time, the identity authentication protocol cannot be deployed at the edge. Therefore, designing a new identity authentication protocol to realize the secure access of mobile terminal users is critical.

In addition, the edge computing data center is built in an open network environment, the internal users, network environment, data flow and application flow of the system are more complex, which are vulnerable to malicious attacks from different network locations, resulting in key privacy data leakage, abnormal service termination and other adverse consequences. The original trust model is no longer reliable, so it is necessary to assume that any device is untrustworthy. The zero-trust architecture is established in the context of the gradual failure of the traditional network boundary trust system, and advocates the concept of breaking a single network boundary, and provides comprehensive and dynamic access control to users, devices, and applications. Applying the zero-trust architecture to the edge-side identity authentication protocol can effectively deal with the security risks caused by the insecurity of the intranet.

1.1 | Related works

The concept of trust put forward by Eschenauer et al.[1] refers to a pair of relationships between entities, which must be established by evidence of the belief relationship and those known recipients can accept it. s story. Therefore, some scholars such as Blaze et al.[2] try to use trust management in distributed systems to improve the security of the authorization mechanism in distributed systems. There are also a small number of people such as Chu et al.[3] who study the application of trust management to web application management systems to achieve the purpose of monitoring trust decisions of all clients and servers. However, with the widespread application of power distribution Internet of Things, the new features of edge computing, such as geographical dispersion and unattended operation, make the original trust model no longer reliable, so establishing a zero trust model in the edge environment is important.

The zero trust model was first proposed by the analysis company Forrester Research[4] in 2010 to deal with new attack methods in information security. This model has a "never trust, always verify" principle. Therefore, no matter whether the generated data traffic is generated from an internal network or an external network, it is untrusted by default. Zero trust network security has been used and implemented by some companies. For example, in 2011, Google launched the beyond corp project based on this concept to practice.[5] In 2017, Gartner released a continuous adaptive risk and trust assessment model at the Security and Risk Management Summit, and proposed that zero trust is the basis for the realization of CARTA's ambition. Cisco's whitelist-based policy model supports a zero-trust security architecture. In 2020, the National Institute of Standards and Technology released a draft of the zero trust architecture.[6] The concept of zero trust has been widely recognized by the industry, but it is still in the stage of technological exploration and research.[7] To realize the concept of zero trust, it is critical to study key technologies related to identity authentication to provide support for rapid identification and authentication of identity under the zero trust model.

Identity authentication technology is an extension and expansion of physical entities in the real world to the virtual network world. In a computer network, when one of them requests a certain benefit-related business from the other party, it is necessary to complete identity authentication before relevant communication services can be carried out. Static password is the simplest identity authentication technology,[8] which is widely used in internet applications such as network management, e-mail, and remote login. However, the identity scheme based on static passwords has many disadvantages such as easy to be eavesdropped on, and poses great security risks. In response to these shortcomings, Lamport first proposed a dynamic password authentication scheme based on the Hash function.[9] This scheme was later improved to the SIKey scheme[10,11] and was widely used. Later, it was found that the scheme could not resist decimal attacks.[12] Sun et al.[13] proposed a two-factor authentication scheme based on hash algorithm on smart card, but this scheme has the disadvantage of not being able to resist password guessing attacks. Many scholars have tried to continuously improve the dynamic password scheme based on the Hash function,[14–17] but they are unable to resist all security threats. Hwang[18] et al. proposed a two-factor dynamic password scheme with timestamp; in order to adapt to the fast and flexible requirements of mobile terminals, Hallsteirfsen et al.[19] proposed a dynamic password scheme based on short message service.

In 2020, Cui et al.[20] proposed a blockchain-based multi-WSN authentication scheme for the Internet of Things. By building a blockchain network in different types of nodes, the identity authentication of each node and different communication scenarios was finally realized. In the same year, Zhang et al.[21] proposed a hybrid blockchain model based on a trust mechanism. Through internal authentication and cross-domain authentication, respectively, identity authentication under local and global blockchain networks was completed, effectively reducing unnecessary consumption. Shen et al.[22] proposed an identity management and authentication model based on identity signatures, and completed the identity authentication of devices in different IoT scenarios through an efficient and secure blockchain-assisted authentication mechanism, and passed off-chain storage and authentication. The key agreement mechanism solves the throughput bottleneck and anonymous identity authentication problems, respectively. In 2021, Mouade[23] proposed an enhanced identity authentication protocol based on the original identity authentication protocol in the Internet of Things, which effectively improved the data security of users and devices. Wang et al.[24] also proposed a handover identity authentication model in a multi-server edge computing environment in 2021, allowing users to authenticate to the nearest edge server according to their location during the handover process, greatly reducing communication overhead.

An important issue that should be resolved in the process of identity authentication is the withdrawal of members. Li et al.[25] proposed a group signature scheme with verifier local revocation based on the DH hypothesis, which solved the problem of non-exemption in some group signature

schemes. In 2018, Maharage et al.[26] added a member revocation mechanism on the basis of the dynamic group signature scheme. Meanwhile, combined with member registration, they proposed a fully dynamic group signature that supports revocation, but it cannot fight against quantum computers. In the same year, San et al.[27] proposed a lattice-based VLR group signature, which does not rely on public key encryption and can complete member revocation. However, if the group is too large, the lattice based zero knowledge proof's security is low. In 2020, Zhang et al.[28] separated the generation of revocation tokens from member keys based on an identity encoding technology, and proposed a new zero-knowledge proof protocol, thereby ensuring complete anonymity. Wang et al.[29] proposed a code-based VLR group signature in 2021. By embedding binary code, it can break away from the dependence on the encryption scheme and improve the efficiency of member revocation. In the same year, Olivier et al.[30] used a group signature with an implementation key and extended it, so that it can be used in high-granularity revocation scenarios, and it is also very effective in practical applications.

1.2 | Our contributions

a. The paper takes the revocable group signature scheme as a building block, embeds the expiration time into the edge terminal device's key, and proposes a local identity authentication and roaming identity authentication protocol based on a zero-trust architecture.
b. In the local identity authentication protocol, the paper defines a new model that takes into account the expiration time's unforgeability. In this article, a complete subtree algorithm is used to revoke expired devices. Compared with the previous revocation list, we have reduced the revocation list's size and ensured that even if the edge server is dishonest, the solution still has backward unlinkability and non-frameability.
c. In the roaming identity authentication protocol, the paper first uses a group signature scheme to realize the completely anonymous identity authentication of the device, and then uses a key exchange mechanism to complete the mutual authentication between the device and the foreign server. Compared with the previous anonymous roaming authentication protocol, we have not only significantly improved the revocation check's efficiency, but also achieved more efficient identity authentication.

2 | THE PROPOSED SCHEME

Referred to the overall framework of zero-trust architecture, as shown in Figure 1, this article builds a zero-trust network security architecture capability model with core security capabilities such as "dynamic identity authentication, continuous trust evaluation, and dynamic access control",

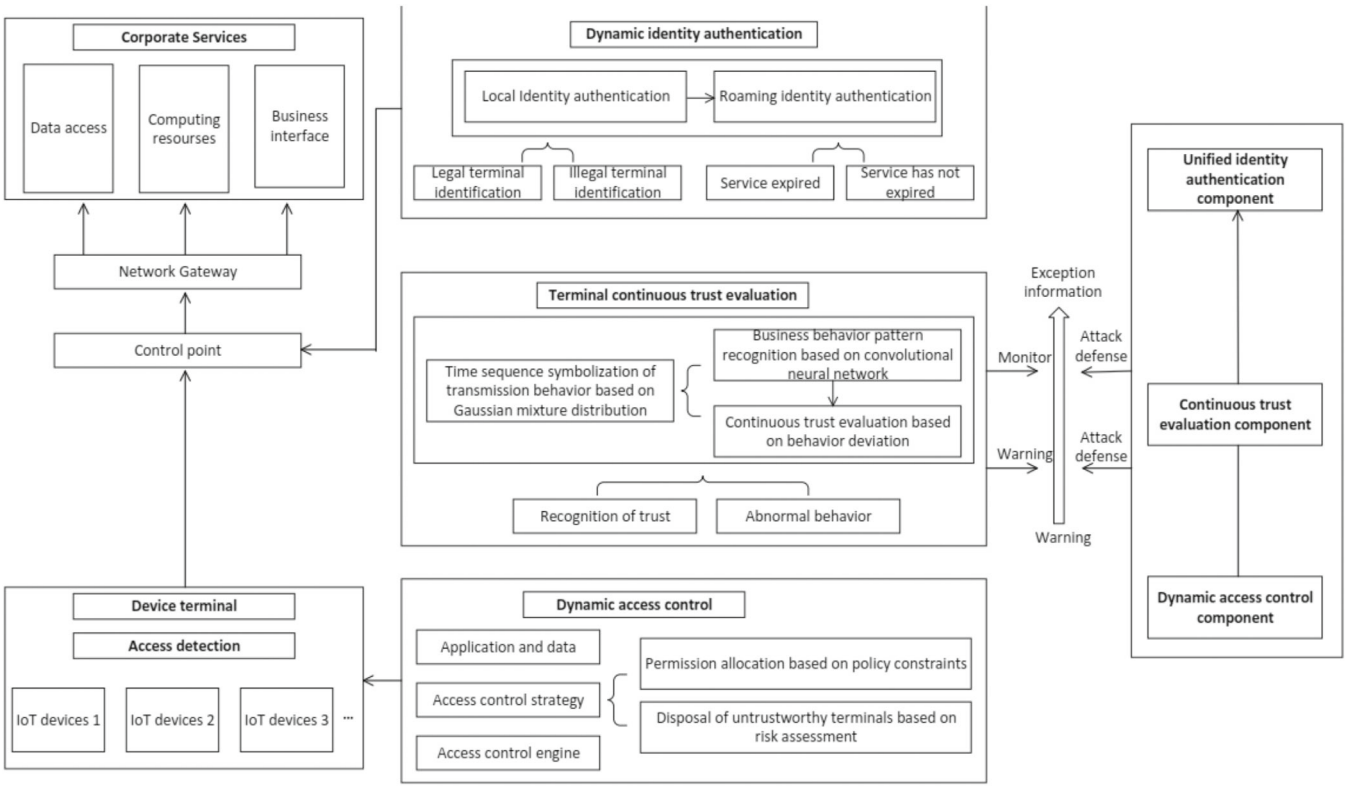

FIGURE 1 Zero-trust architecture framework diagram

which can effectively solve the deficiency of network security protection architecture. This article focuses on the dynamic authentication part that describes the logical components. Considering the change of terminal behavior location, this article studies the identity authentication protocol of local service and roaming service based on group signature. Considering whether the terminal equipment is in service time, this article studies the revocation problem of expired and unexpired terminals, to solve the identification of illegal terminals and expired terminals and the unique identity authentication of legal terminals.

User revocation is important for both local and roaming protocols. Because if the user's subscription period has expired, the local or foreign server needs to find out whether the user has been revoked, so as to prevent any revoked user from entering the network again. Therefore, it is one of the most challenging problems to achieve practical and efficient user revocation. Based on this, this article designs a local and roaming identity authentication scheme of edge terminal devices based on zero trust structure, and carries out anonymous identity authentication and terminal equipment revocation based on revocable group signature scheme, to provide secure, efficient and flexible identity authentication services for edge environment.

3 | PROTOCOL DESIGN

3.1 | Local identity authentication

Identity authentication is one of the indispensable security measures in the mobile edge computing architecture and the first line of defense to ensure the secure communication between the edge terminal and the edge server. Local identity authentication means that a device is in a local location, and only needs to connect to the local network to get services without connecting to the external network. As shown in Figure 2, the local authentication model of edge computing includes three types of entities: cloud server, edge server and edge terminal equipment.

The top layer is the cloud server layer, it consists of a large number of distributed cloud servers, with powerful computing power and a large amount of storage space. The cloud server layer usually receives data from multiple edge servers, and processes and analyzes all data globally to realize data statistics, decision-making and other services. The middle layer is the edge server layer, which consists of several edge nodes with certain computing power, mainly including mobile base stations, routers and so on. The edge server layer is usually deployed in different geographical locations, and each terminal can select the edge server closest to its geographical location for access and communication, so as to reduce the network delay. The main function of the edge server is to extend cloud computing to the edge of the network, so as to reduce the data processing and computing load of the terminal and help the terminal to complete the computing tasks with high real-time requirements. At the same time, it is also responsible for providing limited storage services to terminals and transferring data between the edge terminal layer and the cloud server layer. Not only that, several adjacent or close edge servers can also cooperate with each other to balance the computing, network and storage loads. The bottom layer is the edge terminal layer, which consists of a large number of edge terminal equipment, mainly including mobile devices and Internet

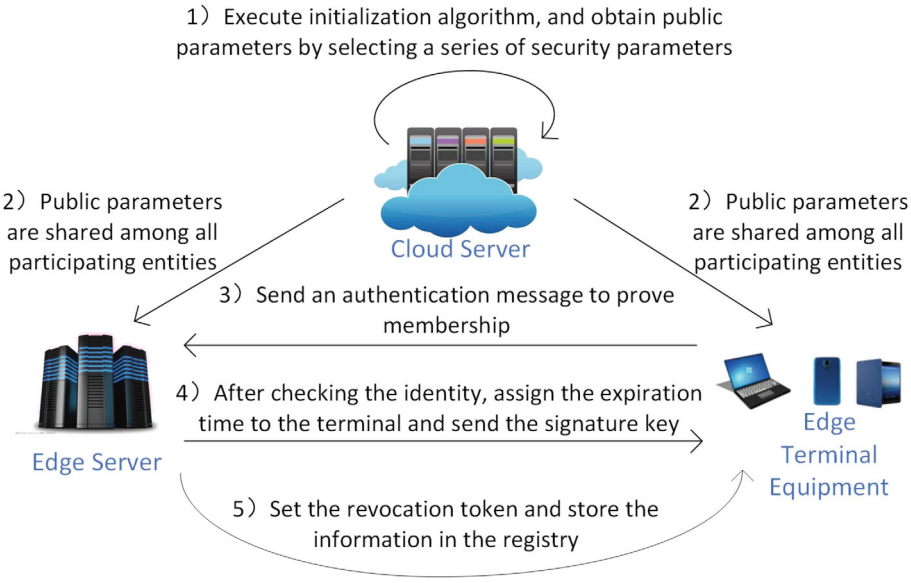

FIGURE 2 Local authentication model diagram of edge computing

of Things devices. These devices are mainly responsible for sensing, collecting and generating raw data, and sending the data to the upper layer for further data processing and storage.

Before the terminal equipment connects to the local network, it needs identity authentication to protect other equipment which needs to connect to the network and network service providers. Therefore, this article proposes a local identity authentication protocol. In the local identity authentication protocol, the local access authentication scheme in an edge computing scenario mainly includes system initialization, key generation algorithm, joining algorithm, revoking algorithm, signing algorithm for group members, signature verification algorithm, opening algorithm, and tracking algorithm. The interaction process of the eight algorithms in the protocol is introduced below.

System initialization. The initialization algorithm is executed by the cloud platform, and public parameters are obtained by selecting a series of security parameters. The public parameters are shared among all participating entities in the scheme, and the registry *reg* is initialized to ø. The specific steps are as follows:

1. Choose $(G_1, G_2, G_T, e, g_1, g_2)$, where g_1 and g_2 are generators of G_1 and G_2, respectively.
2. Choose $f, g, \tilde{g}, h_0, h_1, h_2 \overset{\$}{\leftarrow} G_1, \gamma_A, \gamma_B, \gamma_O \overset{\$}{\leftarrow} Z_p$ and $H : \{0,1\}^* \rightarrow Z_p, H_0 : \{0,1\}^* \rightarrow Z_p$.

Key generation algorithm. After the initialization phase, the group signature key is generated. Let $vk_A = g_2^{\gamma_A}, vk_B = g_2^{\gamma_B}, \hat{g} = f^{\gamma_O}$, output the master private key $msk = (\gamma_A, \gamma_B, \gamma_O)$ and the master public key $gpk = \left(\left(G_1, G_2, G_T, e, g, \hat{g}\right), f, \tilde{g}, g_1, g_2, h_0, h_1, h_2, vk_A, vk_B, H, H_0\right)$.

Join/issue algorithm. Edge computing terminal device and group manager edge server is responsible for running this algorithm. The joining algorithm inputs the master public key gpk, while the issuing algorithm takes the master private key msk, the registry reg, and the expiration time τ_i as input. Finally, the algorithm outputs the signature key usk_i, expiration time τ_i and terminal device key usk_i, and the publishing algorithm outputs the registry reg[i], which stores expiration time τ_i and a revocation token grt_i.

In case of natural cancelation, that is, expiration time $\tau < t$, the time information is managed by a binary tree BT. The number of leaf nodes of binary tree is the same as the maximum length T of time. Set $Path(\eta) := (u_1, u_2, \ldots, u_l)$, where u_1 is the root node, $u_l = \eta, l = logT$. The expiration time τ and the current time t are allocated to the leaf node η, and the edge server generates the node signatures contained in the $Path(\eta)$, and then sends these signatures to the edge terminal device with the expiration time τ. If a leaf node is associated with time *t*, all the leaf nodes on its left are revoked, and the edge server generates a node signature, and publishes the signature as expired information at time t. For example, if $T = 8$, the binary tree has eight leaf nodes, as shown in Figure 3. When τ is associated with node K, the signer with expiration time $\tau < t$ has the signatures of nodes A, B, E, and K, while nodes H and I are revoked, and then node C and node E become the new root nodes. Therefore, it contains the signatures of nodes C and E. Then the signer can prove that they have node E's signature without revealing the node itself in this way. In the lower part of Figure 3, it only contains the signature of C. Since $\tau < t$ has passed, the signatures of A, B, E, and K are not included in ei_t.

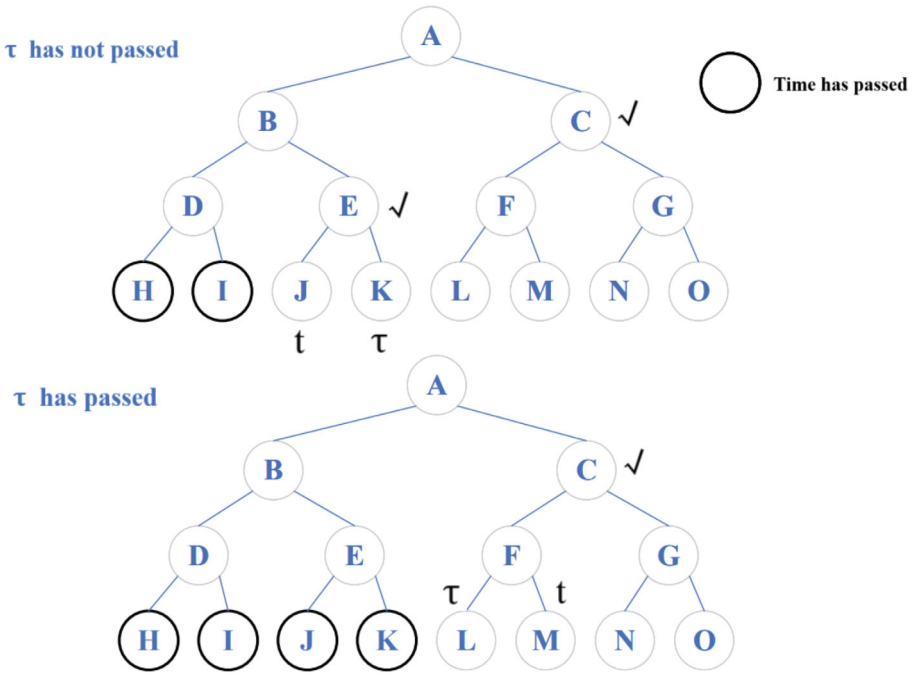

FIGURE 3 An example graph where the leaf node is 8

A signer with an expiration time τ has a certificate $\{(A_j, \xi_j, \zeta_j)\}_{j\in[1,l]}$ as the signature key gsk, where $A_j = \left(gh_0^{\zeta_j} h_1^{u_j} X_i\right)^{1/\xi_j+\gamma_A}$, where $X = h_2^{usk}$ is only contains the signature of C. ζ_j and ξ_j are random values, and g, h_0, and h_1 are all common values. Each A_j is two messages' BBS + signature. The edge server runs the CS-TBK(BT, t) algorithm and outputs $Y := (v_1, v_2, \ldots, v_{num})$ at time t. Expiration information ei_t contains $\{(B_{i,t}, \xi_i', \zeta_i')\}_{i\in[1,num]}$, each $B_{i,t}$ is two messages $v_i \in Y$ and the BBS + signature of the current time t. Since the CS-TBK(BT, t) algorithm is run, if $\tau < t$, there is a node $u \in Path(\eta) \cap Y$. Therefore, a signer who is not revoked can use the zero-knowledge proof to prove two signatures' existence of the same node in its signature keys gsk and ei_t, respectively. At this time, if a revoked signer still attempts to calculate a valid signature, it must prepare the BBS + signature B. However, due to that the BBS + signature has the characteristics of unforgeability, it ensures that the expired signer cannot calculate a valid signature, so as to ensure that the key expiration time cannot be forged.

If it is revoked in advance, at this stage, the edge server stores a revocation token $grt_i = \tilde{X}_i$, where $\tilde{X}_i = \tilde{g}^{usk_i} \in \mathbb{G}_1$. Then the edge server select $y_t \xleftarrow{\$} Z_p$, sets $\tilde{h}_t = g^{y_t}, \hat{h}_t := g_2^{y_t} \in \mathbb{G}_2$ at time t, then the equation $e(\tilde{h}_t, g_2) = e(\tilde{g}^{y_t}, g_2) = e(g, \hat{g}^{y_t}) = e\left(g, \hat{h}_t\right)$ holds. A group signature σ contains $\tilde{h}_t^{\beta}, g^{d(usk_i+\beta)}, g^d, g_2^d$, where β and d are the signer's random choices. If an edge terminal device EU_i is revoked in advance, then the edge server calculates $grt_{i,t} := grt_i^{y_t} = \tilde{h}_t^{usk_i}$, and stores $grt_{i,t}$ to RL_t. $grt_{i,t}$ is the revocation token in the current time period t, which satisfies $e\left(grt_{i,t}\tilde{h}_t^{\beta}, g_2^d\right) = e\left(\tilde{h}_t^{usk_i+\beta}, g_2^d\right) = e\left(g^{y_t(usk_i+\beta)}, g_2^d\right) = e\left(g^{d(usk_i+\beta)}, g_2^{y_t}\right) = e\left(\tilde{g}^{d(usk_i+\beta)}, \hat{h}_t\right)$. By checking whether the equation holds at $grt_{i,t}$, the verifier can obtain the signer who was revoked in advance. In general, the specific steps to join/issue the algorithm are as follows:

(1) The signer selects $EU_i \xleftarrow{\$} Z_p$, calculates $X_i = h_2^{usk_i}$ and $\tilde{X}_i = g^{usk_i}$, and then adds $(X_i, \tilde{X}_i)_i$ sent to the edge server. The edge terminal device EU_i proves its key usk_i to the edge server as follows.

First, EU_i chooses $r_x \xleftarrow{\$} Z_p$, calculates $R = h_2^{r_x}, \tilde{R} = g^{r_x}, c_x \leftarrow H'(X_i, \tilde{X}_i, R, \tilde{R}), s_x = r_x + c_x usk_i$, and send (s_x, c_x) to the edge server. After receiving it, the edge server checks the $c_x = H'\left(X_i, \tilde{X}_i, h_2^{s_x}/X_i^{c_x}, g^{s_x}/X_i^{c_x}\right)$.

(2) The edge server assigns the leaf node η to the expiration time τ_i. For all $u_j \in Path(\eta) := (u_1, u_2, \ldots, u_l)$, the edge server calculates BBS + signature $\{(A_j, \xi_j, \zeta_j)\}_{j\in[1,l]}$, where $A_j = \left(g_1 h_0^{\zeta_j} h_1^{u_j} X_i\right)^{1/\xi_j+\gamma_A}$, and sends $gsk_i = \left((A_j, \xi_j, \zeta_j), u_i\right)$ for $j \in [1, l]$ and τ_i to the terminal equipment EU_i.

(3) The edge server sets $grt_i = \tilde{X}_i$ and stores $(\{A_j\}_{j\in[1,l]}, \tau_i, grt_i)$ in the registry $reg[i]$.

Revocation algorithm. It is mainly used to generate expiration information and revocation list. To check whether the signed edge terminal device is revoked, an expiration time constraint is set for each signer. Therefore, in addition to proving the membership to the edge server, the edge terminal device also should prove that the expiration time τ_i has not passed. If $t < \tau_i$, calculate the revocation token $grt_{i,t}$ at time t, and store $grt_{i,t}$ in the revocation list RL_t. At last, output (ei_t, RL_t). Specific steps are as follows:

(1) Choose $y_t \xleftarrow{\$} Z_p$, compute $\tilde{h_t} = g^{y_t}$ and $\hat{h_t} = g_2^{y_t}$.

(2) Generate expiration information: Get $Y := (v_1, \ldots, v_{num}) \leftarrow CS - TBK(BT, t)$. Calculate BBS+ signature $\{(B_{i,t}, \xi_i', \zeta_i')\}_{i\in[1,num]}$, set $ei_t = \left(\left(v_i (B_{i,t}, \xi_i', \zeta_i')\right)_{i\in[1num]}, \tilde{h_t}\right)$, where $B_{i,t} = \left(gh_0^{\zeta_i'} h_1^{v_i} h_2^t\right)^{1/\xi'+\gamma_B}$.

(3) Generate revocation list: Calculate $grt_{i,t} = grt_i^{y_t}$ for all $i \in RU_t$, and set $RL_t = \left(\tilde{h_t}, \hat{h_t}, \{grt_{i,t}\}_{i\in RU_t}\right)$.

(4) Output (ei_t, RL_t).

Signing algorithm. The algorithm takes the master public key gpk, the signature key gsk_i, the terminal device's key usk_i, the message to be signed m, ei_t and time t as input, and the output signature σ. Specific steps are as follows:

Assuming $t < \tau_i$, there is a node u such that $((A, \varepsilon, \zeta), u)$ is included in gsk_i, where $A = \left(gh_0^{\zeta} h_1^{u} X_i\right)^{1/\varepsilon+\gamma_A}$; ei_t also contains $((B_t, \varepsilon', \zeta'), u)$, where $B_t = \left(g_1 h_0^{\zeta'} h_1^{u} h_2^{t}\right)^{1/\xi'+\gamma_B}$. Choose $\alpha \xleftarrow{\$} Z_p$ to calculate the ciphertext $\varphi_1 = f^{\alpha}, \varphi_2 = A\tilde{g}^{\alpha}, \varphi_3 = B_t\hat{g}^{\alpha}$; choose $\beta, d \xleftarrow{\$} Z_p$ to calculate the ciphertext for premature revocation $\varphi_4 = \tilde{h}_t^{\beta}, \varphi_5 = g^{(x_i+\beta)d}, \varphi_6 = g^d, \varphi_7 = g_2^d$.

Choose $r_\alpha, r_\beta, r_\zeta, r_\xi, r_{\zeta'}, r_{\xi'}, r_u, r_x, r_\delta, r_{\delta'} \xleftarrow{\$} Z_p$, compute $R_1 = e(h_0, g_2)^{r_\zeta} e(h_1, g_2)^{r_u} e(h_2, g_2)^{r_x} e(g_1, g_2)^{r_\delta} e(\varphi_2, g_2)^{-r_\varepsilon} e\left(vk_A, \tilde{g}\right)^{r_\alpha}$, $R_2 = e(h_0, g_2)^{r_{\xi'}} e(h_1, g_2)^{r_u} e\left(\hat{g}, vk_B\right)^{r_\alpha} e\left(\hat{g}, g_2\right)^{r_{\delta'}} e(g_2, \varphi_3)^{-r_{\xi'}}, R_3 = \varphi_1^{r_\xi} f^{-r_\delta}, R_4 = \varphi_1^{r_{\xi'}} f^{-r_{\delta'}}, R_5 = \tilde{h}_t^{r_\beta}, R_6 = \varphi_6^{r_x+r_\beta}$, compute $c \leftarrow H(\varphi_1, \ldots, \varphi_7, R_1, \ldots, R_6, m)$. Compute $s_\alpha = r_\alpha + c\alpha, s_\beta = r_\beta + c\beta, s_\zeta = r_\zeta + c\zeta, s_\varepsilon = r_\xi + c\varepsilon, s_{\zeta'} = r_{\zeta'} + c\zeta', s_{\xi'} = r_{\xi'} + c\xi', s_u = r_u + cu, s_x = r_x + cx_i, s_\delta = r_\delta + c\delta, s_{\delta'} = r_{\delta'} + c\delta'$, output signature $\sigma = \left(\varphi_1, \ldots, \varphi_7, s_\alpha, s_\beta, s_\varepsilon, s_u, s_x, s_\delta, s_{\zeta'}, s_{\varepsilon'}, s_{\delta'}, c\right)$.

Verification Algorithm. The verification algorithm is divided into verification check and revocation check. When the edge terminal device wants to join the group, a revocation token grt_i is generated. The revocation tokens are different at different times, so we represent the specific revocation token of the equipment as $grt_{i,t}$. Meanwhile, if the equipment EU_i is revoked at t, $grt_{i,t}$ is included in the revocation list RL_t and used for premature revocation. The edge server can implicitly use the revocation token to track the signer, that is, it can calculate $grt_{i,t}$ for all grt_i. and checks the revocation check equation.

In the case of natural expiration, the signer revoked after expiration cannot calculate the signature that has passed the verification check, only the verification check is needed; However, if the signer is revoked before the expiration time due to damaged credentials, the verifier needs to run the revocation check process. The larger the size of the undo list, the greater the calculation cost. In general, the verification algorithm takes gpk, t, σ, m and RL_t as input, and the output is valid or invalid. Specific steps are as follows:

(1) Verification check: If $e(\psi_6, g_2) = e(g, \psi_7)$, then calculate $R_1' = e(h_0, g_2)^{s_\zeta} e(h_1, g_2)^{s_u} e(h_2, g_2)^{s_x} e\left(\tilde{g}, vk_A\right)^{s_\alpha} e(\tilde{g}, g_2)^{s_\delta} e(\varphi_2, g_2)^{-s_\varepsilon} \left(e(\varphi_2, vk_A)/e(g, g_2)\right)^{-c}$, $R_2' = e(h_0, g_2)^{s_{\xi'}} e(h_1, g_2)^{s_u} e\left(\hat{g}, vk_B\right)^{s_\alpha} \left(e(\varphi_3, vk_B)/e(g_1, g_2) e(h_2, g_2)^t\right)^{-c} e\left(\hat{g}, g_2\right)^{s_{\delta'}} e(\varphi_3, g_2)^{-s_{\xi'}}, R_3' = \varphi_1^{s_\xi} f^{-s_\delta}, R_4' = \varphi_1^{s_{\xi'}} f^{-s_{\delta'}}, R_5' = \tilde{h}_t^{s_\beta} \cdot \varphi_4^{-c}, R_6' = \varphi_6^{s_x+s_\beta} \varphi_5^{-c}$; otherwise the output is invalid; if $c \neq H(\psi_1, \ldots, \psi_7, R_1', \ldots, R_6', m)$, the output is also invalid.

(2) Revocation check: If there is $grt_{i,t}$ such that $e\left(grt_{i,t}\psi_4, \psi_7\right) = e\left(\psi_5, \widehat{h}_t\right)$, the output is invalid.

Otherwise, the output is invalid.

Opening algorithm. Take $gpk, msk, t, reg, \sigma, m, RL_t$ as input, and output the sign of signer i or RL_t. If $invalid \leftarrow Verify\left(gpk, t, \sigma, m, RL_t\right)$, then output $\perp$. Otherwise, analyze $\sigma = \left(\psi_1, \ldots, \psi_7, c, s_\alpha, s_\beta, s_\zeta, s_\xi, s_{\xi'}, s_{\xi''}, s_u, s_x, s_\delta, s_{\delta'}\right)$, $msk = (\gamma_A, \gamma_B, \gamma_O)$. Calculate $A = \psi_2/\psi_1^{\gamma_O}$, search for i, make $reg[i]$ contain A, and output i. If there is no such entry, output $\perp$.

Tracking algorithm. The goal of the tracking algorithm is to find those terminal devices that illegally leak the keys and revoke them. As a mechanism combined with the revocation mechanism, it can achieve seamless integration between revocation and tracking, and there is no need to undo any changes to the algorithm. What we mainly use is "black box" tracking, that is, instead of disassembling the decoder, we provide it with encrypted messages and observe its output (decrypted messages), trying to find out who has leaked the key.

First, an important procedure in the tracking mechanism is to give a partition $P = P_{i_1}, P_{i_2}, \ldots, P_{i_m}$, and there are two possibilities for an illegal box output: either (i) when encrypting partition P, the box cannot be decrypted with a probability greater than the threshold, or (ii) a subset P_{i_j} is found so that P_{i_j} contains the leaker. Such a process is called subset tracking. The subset tracking program first tests whether the box decodes messages with partitions $P = P_{i_1}, P_{i_2}, \ldots, P_{i_m}$ with a sufficient probability greater than the threshold (for example, >0.5), and if not, draw a conclusion and output that the box cannot be decrypted with P. Otherwise, it needs to find the subset P_{i_j} that contains the leaked key. In other words, if P_{i_j} contains only one possible candidate, it must be the device that leaked the key, and we will permanently revoke the device without harming legitimate terminal devices. Otherwise, we divide P_{i_j} into two roughly equal subsets and continue with a new partition. Specific steps are as follows:

Each stage initially uses the current partition $P = P_{i_1}, P_{i_2}, \ldots, P_{i_m}$ for subset tracking. If the output box of the tracking result cannot be decrypted with P, it means that a method has been found to disable the box without harming any legitimate edge terminal devices. Otherwise, let P_{i_j} be the set output by the subset tracking program, that is, P_{i_j} contains the device that leaked the key, and split P_{i_j} into $P_{i_{j1}}$ and $P_{i_{j2}}$. Let $F \subset P$ denote the boundary of P. Since $P_{i_{j1}}$ and $P_{i_{j2}}$ are the new boundaries at this time, let $F' = F \cup P_{i_{j1}} \cup P_{i_{j2}}$, if P_{i_j} is in the boundary of F and P_{i_k} is a subset of its partners in F, then $F' = F'/P_{i_k}$ (remove P_{i_k} from the boundary). Next, the coverage C is calculated for all receivers not covered by F', and the new partition P' is defined as the union of C and F'.

3.2 | Roaming identity authentication

Roaming identity authentication means that the mobile terminal can get rid of the limitation of the coverage of the home network. When the terminal roams to a network outside the home network, no matter where it is located, it can stay connected in the global Internet by connecting with external networks. When connecting to an external network, the privacy of the device or server is usually exposed, so the identity of the mobile terminal must be authenticated before connecting.

Due to the particularity of the global Internet, roaming authentication protocol should not only verify the validity of the equipment identity, but also ensure that the revoked terminal equipment re-enters the network due to the expiration of the service. Therefore, the verification stage of roaming identity authentication includes validity check and revocation check. Validity check means verifying whether the authentication token is generated by a legitimate and valid terminal equipment; Revocation check refers to verifying whether the user has expired. If some terminals expire naturally due to the expiration of the subscription period or are revoked in advance due to the leakage of credentials and so forth these terminals will be added to the revocation list.

Total cost of revocation check = number of revocation terminals × calculation cost specific to RCheck function, which shows that to reduce the total cost of revocation check, we can reduce the number of revocation terminals, or reduce the calculation cost by designing Boolean functions more effectively. However, the number of revoked terminals will definitely increase with the passage of time. Therefore, this article proposes a roaming identity authentication protocol, which can reduce the revocation cost when the number of revoked terminals is large. Figure 4 is the roaming identity authentication model proposed in this article. When a mobile terminal roams to a foreign country and needs to connect to an external network, it needs to perform identity authentication and key exchange with the foreign server, while the home server needs to send authentication data and revocation list to the foreign server. Next, we first describe the revocable group signature scheme in roaming identity authentication, and then propose a complete roaming protocol.

1. Revocable group signature scheme

The revocable group signature scheme consists of a master key generation algorithm, a prenegotiation algorithm, a user joining algorithm, a signing algorithm, a verification algorithm, and a tracking algorithm.

Master key generation algorithm. The remote domain authentication server as the group manager selects a bilinear group pair $\mathbb{G}_1$ and $\mathbb{G}_2$, and randomly selects $h, \tilde{h}, \tilde{g} \in \mathbb{G}_1, g, Q \in \mathbb{G}_2$, and $\gamma_1, \gamma_2, z \in Z_p^*$, and then sets $g_1 \leftarrow \psi(g_2), z_1 \leftarrow g_2^{\gamma_1}, z_2 \leftarrow g_2^{\gamma_2}$. Then the master public key is $gpk \leftarrow \left(g_1, g_2, h, \tilde{h}, \tilde{g}, z_1, z_2, H, l, Q, Q^z, Q^{z^2}, \ldots, Q^{z^l}\right)$, where H is a hash function in the range of $\mathbb{Z}_p$, and l is the maximum length of time; the master key is $msk \leftarrow (\gamma_1, \gamma_2)$. At the same time, the edge server also maintains a revocation list RL initialized as an empty set.

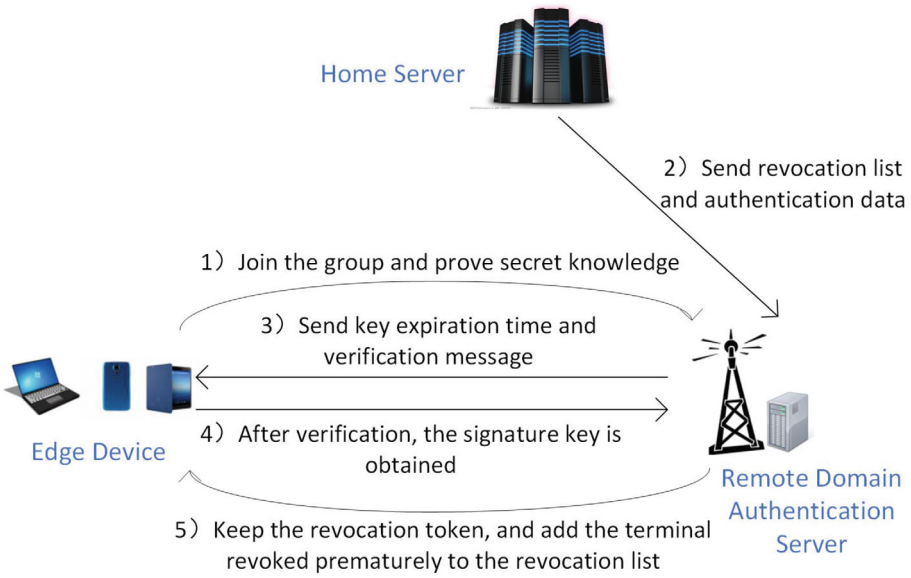

FIGURE 4 Roaming authentication model

Prenegotiation algorithm. In this stage, each edge terminal sends a prenegotiation message M to the roaming server, which contains a parameter $g^{r_{EU}}$ for session key agreement in the authentication phase, where r_{EU} is a random number selected by the edge terminal device. In addition, the edge device uses its private signature key to sign the prenegotiation message by applying the signing algorithm, and then sends the signed message to the roaming server. After receiving this message, the roaming server first checks whether the timestamp is within the allowable range compared with the current time, and then verifies the signature through the verification algorithm. If both verifications are passed, the roaming server caches M. In other words, this stage can not only be used to improve the authentication speed, but also can be executed periodically to update the negotiation parameters to further reduce the possibility of session key leakage.

User joining algorithm. The new edge terminal device EU_i as a user can request to join the group. Specific steps are as follows:

(1) EU_i randomly selects a secret $f_i \in \mathbb{Z}_p$.

(2) EU_i choose some random numbers $\hat{f} \in \mathbb{Z}_p$, calculate $\hat{F} \leftarrow h^{\hat{f}}$, then calculate $\hat{r} \leftarrow \hat{f} + H\left(F_i, \hat{F}\right) f_i \ mod\ p$, send $F_i, \hat{F}, \hat{r}$ to the remote domain authentication server.

(3) After receiving $F_i, \hat{F}, \hat{r}$, the server checks whether the equation $h^{\hat{r}} = \hat{F} F_i^{H\left(F_i, \hat{F}\right)}$ is established. If so, the terminal device proves the knowledge of f_i to the edge manager.

(4) The remote domain authentication server calculates $\left\{\tau_{ij}\right\}_{j\in[1,l]} \leftarrow 1 - ENC(\tau_i)$, where τ_{ij} is the key expiration time of the terminal device, and $1 - ENC(\tau_i)$ is a set. When this set has no elements of length j, we introduce the dummy string into τ_{ij}. For all $j \in \{1, \ldots, l\}$, the remote domain authentication server calculates $A_{ij} \leftarrow (g_1 F_i)^{1/(\tau_{ij}\gamma_1 + \gamma_2 + x_{ij})}$, where $x_{ij} \in Z_p^*$, and then $\left(\tau_i, \left\{x_{ij}, A_{ij}\right\}_{j\in[1,l]}\right)$ are sent to the edge terminal device EU_i.

(5) EU_i verifies $e\left(A_{ij}, z_1^{\tau_{ij}} z_2 g_2^{x_{ij}}\right) = e(g_1 F_i, g_2)$ for all j∈{1, … ,l}, and finally gets the key $gsk_i \leftarrow \left(\left\{x_{ij}, A_{ij}\right\}_{j\in[1,l]}, \tau_i, f_i\right)$.

(6) The remote domain authentication server retains the revocation token $grt_i \leftarrow \left(\tau_i, \left\{x_{ij}\right\}_{j\in[1,l]}\right)$ and stores $\left(EU_i, grt_i\right)$ in the user database . If the terminal device EU_i is revoked in advance, the group manager will add grt_i to the revocation list RL.

Signing algorithm. A terminal device EU_i signs a message m at time t by the following steps:

(1) Calculate $\left\{\tau_{ij}\right\}_{j\in[1,l]} \leftarrow 1 - ENC(\tau_i)$ and $\left\{t_j\right\}_{j\in[1,l]} \leftarrow 0 - ENC(t)$; find an index $1 \le k \le l$ makes $\tau_{ik} = t_k$. In order to ensure the anonymity of devices, choose a random number $B \in \mathbb{G}_1$, and calculate $J \leftarrow B^{f_i}$ and $K \leftarrow B^{x_{ik}}$;

(2) Then, for the random number $\alpha_1, \alpha_2, \alpha_3, \alpha_4 \in \mathbb{Z}_p$, $T \leftarrow A_{ik}\tilde{h}^{\alpha_1}, W \leftarrow w\tilde{h}^{\alpha_2}, V_1 \leftarrow \tilde{g}^{x_{ik}}\tilde{h}^{\alpha_3}, V_2 \leftarrow \tilde{g}^{t_k}\tilde{h}^{\alpha_4}$ are calculated separately, and the elements T, W, V_1, V_2 are treated as promises of A_{ik}, w, x_{ik}, and t_k, respectively, and they bind the signature to these values in a hidden way.

(3) Next, in order to prove to the verifier that the signature is a good form of knowledge proof system, some auxiliary values need to be calculated. First randomly select $r_x, r_f, r_t, r_{\alpha_1}, r_{\alpha_2}, r_{\alpha_3}, r_{\alpha_4} \in \mathbb{Z}_p$, let $\theta_1 \leftarrow x_{ik}\alpha_1, \theta_2 \leftarrow \alpha_1\alpha_3, \theta_3 \leftarrow t_k\alpha_1, \theta_4 \leftarrow \alpha_1\alpha_4, \theta_5 \leftarrow t_k\alpha_2, \theta_6 \leftarrow \alpha_2\alpha_4$, and then randomly select $r_{\theta_1}, r_{\theta_2}, r_{\theta_3}, r_{\theta_4}, r_{\theta_5}, r_{\theta_6} \in \mathbb{Z}_p$ and calculate the auxiliary value $R_1 \leftarrow B^{r_f}, R_2 \leftarrow B^{r_x}, R_3 \leftarrow K^{r_{\alpha_1}} B^{-r_{\theta_1}}$, R_1, R_2, R_3 demonstrate a good form of secret components known only to the user, providing exculpability and traceability. Calculate $R_4 \leftarrow e(h, g_2)^{-r_f} e(T, g_2)^{r_x} e\left(\tilde{h}, z_1\right)^{-r_{\theta_3}} e\left(\tilde{h}, g_2\right)^{-r_{\theta_1}} e(T, z_1)^{r_t} e\left(\tilde{h}, z_2\right)^{-r_{\alpha_1}}, R_5 \leftarrow \tilde{g}^{r_x}\tilde{h}^{\alpha_3}, R_6 \leftarrow \tilde{g}^{r_{\theta_1}}\tilde{h}^{r_{\theta_2}} V_1^{-r_{\alpha_1}}, R_7 \leftarrow \tilde{g}^{r_{\theta_3}}\tilde{h}^{r_{\theta_4}} \cdot V_2^{-r_{\alpha_1}}$, R_4, R_5, R_6, R_7 is about the validity of the key issued to the user. R_8, R_9, R_{10} are used to prove whether the expiration time is later than the current time, $R_8 \leftarrow \tilde{g}^{r_{tk}}\tilde{h}^{r_{\alpha_4}}, R_9 \leftarrow e(W, Q)^{r_t} e(\tilde{h}, Q)^{-r_{\theta_5}} e\left(\tilde{h}, Q^z\right)^{-r_{\alpha_2}}$, $R_{10} \leftarrow \tilde{g}^{r_{\theta_5}} V_2^{-r_{\alpha_2}} \cdot \tilde{h}^{r_{\theta_6}}$.

(4) Compute $s_{\alpha_1} \leftarrow r_{\alpha_1} + c\alpha_1, s_{\alpha_2} \leftarrow r_{\alpha_2} + c\alpha_2, s_{\alpha_3} \leftarrow c\alpha_3 + r_{\alpha_3}, s_{\alpha_4} \leftarrow r_{\alpha_4} + c\alpha_4; s_{\theta_1} \leftarrow r_{\theta_1} + c\theta_1, s_{\theta_2} \leftarrow r_{\theta_2} + c\theta_2, s_{\theta_3} \leftarrow r_{\theta_3} + c\theta_3, s_{\theta_4} \leftarrow r_{\theta_4} + c\theta_4, s_{\theta_5} \leftarrow r_{\theta_5} + c\theta_5, s_{\theta_6} \leftarrow r_{\alpha_3} + c\theta_6; s_x \leftarrow r_x + cx_{ik}, s_f \leftarrow r_f + cf_i, s_t \leftarrow r_t + ct_k$, where c ← H (gpk, t, m, B, J, K, T, W, $V_1, V_2, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}$).

(5) After the calculation is completed, the edge terminal device outputs the signature of message m and time t: $\sigma \leftarrow \left(B, J, K, T, W, V_1, V_2, c, s_{\alpha_1}, s_{\alpha_2}, s_{\alpha_3}, s_{\alpha_4}, s_{\theta_1}, s_{\theta_2}, s_{\theta_3}, s_{\theta_4}, s_{\theta_5}, s_{\theta_6}, s_x, s_f, s_t\right)$.

Verification Algorithm. The verification algorithm is divided into verification check and revocation check. When receiving the signature σ of the message m and the signature time t, the verifier first needs to perform a verification check, that is, to verify the validity of the signature to ensure that the signature is not generated by the device being revoked. Next, a revocation check is required, and the output is valid only if the verification check is passed and the user is not revoked, otherwise the output is invalid.

(1) Verification check. Compute $\{t_j\}_{j\in[1,l]} \leftarrow 0 - ENC(t), Y \leftarrow Q^{\prod_{j=1}^{l}(z+t_j)}$; then compute $\tilde{R}_1 \leftarrow B^{s_f} J^{-c}, \tilde{R}_2 B^{s_x} K^{-c}, \tilde{R}_3 \leftarrow K^{s_{\alpha_1}} B^{-s_{\theta_1}}, \tilde{R}_4 \leftarrow e(h, g_2)^{-s_f} (e(T, z_2)/e(g_1, g_2))^c e(T, z_1)^{s_t} e(T, g_2)^{s_x} e(\tilde{h}, z_1)^{-s_{\theta_3}} e(\tilde{h}, g_2)^{-s_{\theta_1}} e(\tilde{h}, z_2)^{-s_{\alpha_1}}, \tilde{R}_5 \leftarrow \tilde{g}^{s_x} \tilde{h}^{s_{\alpha_3}} * V_1^{-c}, \tilde{R}_6 \leftarrow \tilde{g}^{s_{\theta_1}} \tilde{h}^{s_{\theta_3}} V_1^{-s_{\alpha_1}}, \tilde{R}_7 \leftarrow \tilde{g}^{s_{\theta_3}} \tilde{h}^{s_{\theta_4}} V_2^{-s_{\alpha_1}}, \tilde{R}_8 \leftarrow \tilde{g}^{s_t} \tilde{h}^{s_{\alpha_4}} V_2^{-c}, \tilde{R}_9 \leftarrow \cdot (e(W, Q^z)/e(Y, Q))^c e(\tilde{h}, Q^z)^{-s_{\alpha_2}} e(W, Q)^{s_t} e(\tilde{h}, Q)^{-s_{\theta_5}}, \tilde{R}_{10} \leftarrow \tilde{g}^{s_{\theta_5}} \tilde{h}^{s_{\theta_6}} V_2^{-s_{\alpha_2}}$, check if the equation $c = H(gpk, t, m, B, J, K, T, W, V_1, V_2, \tilde{R}_1, \tilde{R}_2, \tilde{R}_3, \tilde{R}_4, \tilde{R}_5, \tilde{R}_6, \tilde{R}_7, \tilde{R}_8, \tilde{R}_9, \tilde{R}_{10})$ holds. If the equation is true, the revocation check is performed, otherwise the output is invalid.

(2) Revocation check. For each grt_i in the list RL, first parse $\{\tau_i, \{x_{ij}\}\} \leftarrow grt_i$, then calculate $\{\tau_{ij}\}_{j\in[1,l]} \leftarrow 1 - ENC(\tau_i)$ and $\{t_j\}_{j\in[1,l]} \leftarrow 0 - ENC(t)$, find an index 1 ≤ k ≤ l such that $\tau_{ik} = t_k$; if there is k such that the equation of $K = B^{x_{ik}}$ holds, It proves that the user has been revoked and the output is invalid.

Tracking algorithm. The remote domain authentication server can track the signer by opening a signature σ. First parse $\{\tau_i, \{x_{ij}\}\} \leftarrow grt_i$, and then calculate $\{\tau_{ij}\}_{j\in[1,l]} \leftarrow 1 - ENC(\tau_i)$ and $\{t_j\}_{j\in[1,l]} \leftarrow 0 - ENC(t)$, find an index 1 ≤ k ≤ l such that $\tau_{ik} = t_k$; if there is k such that the equation of $K = B^{x_{ik}}$ holds, then it is proved that the signer of σ is the edge terminal device EU_i, thus completing the tracking of the signer.

2. Complete anonymous roaming authentication protocol.

A complete roaming protocol consists of a revocable group signature scheme and a key exchange mechanism. Many edge terminal devices can independently connect to the edge server at the same time. Next, we describe how a single device roams to an external edge server for authentication and establishment of a session key. The interactive protocol is shown in Figure 5.

(1) The edge device EU_i selects a random number $r_{EU} \in Z_p$ and a temporary pseudonym TP, sends $(HS, TP, g^{r_{EU}})$ to the server FS, where HS is the local edge server of the edge device.

(2) The server FS selects a random number $r_{FS} \in Z_p$ and calculates $\sigma_{FS} \leftarrow Sig_{sk_{FS}}(m_{FS})$, where Sig is the traditional digital signature scheme of each server, and sk_{FS} is the signing key of the remote domain authentication server, $m_{FS} = FS \parallel HS \parallel TP \parallel g^{r_{FS}} \parallel g^{r_{EU}}$; FS sends $(FS, g^{r_{FS}}, \sigma_{FS})$ to EU_i, delete r_{FS} from memory after calculating $\kappa \leftarrow (g^{r_{EU}})^{r_{FS}}$.

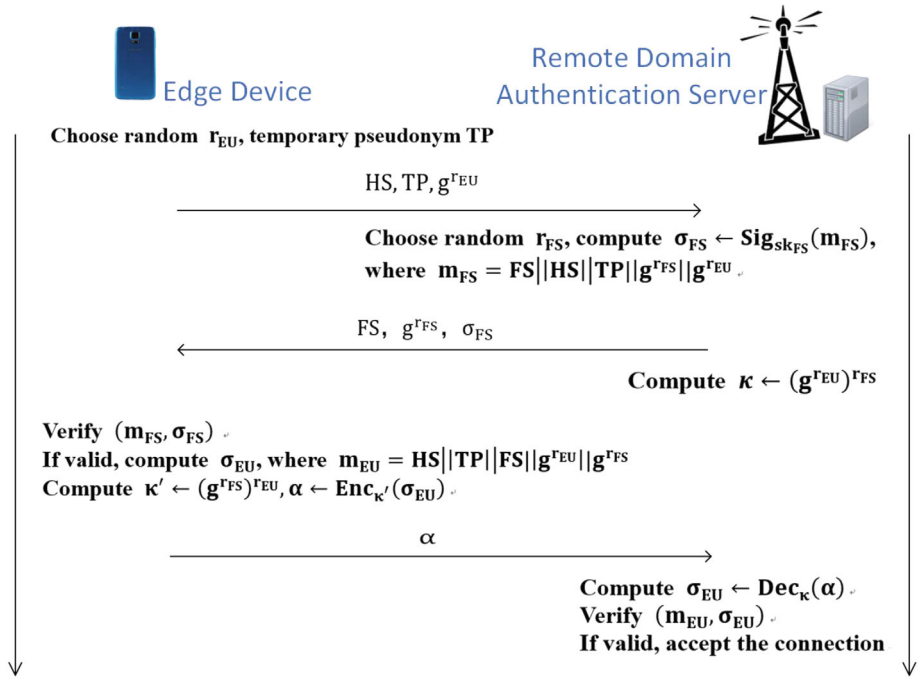

FIGURE 5 Connection diagram for establishing session key

(3) At this time, the edge device EU_i receives the $(FS, g^{r_{FS}}, \sigma_{FS})$ from the remote domain authentication server, and runs the verification algorithm to verify σ_{FS}. If the output is valid, then runs the signature algorithm to generate EU_i group signature σ_{EU}, calculate $\kappa' \leftarrow (g^{r_{FS}})^{r_{EU}}$ and then use κ' as the session key, encrypt σ_{EU} into $\gamma \leftarrow Enc_{\kappa'}(\sigma_{EU})$, and send γ to FS; if the output is invalid, the connection is rejected.

(4) The remote domain authentication server FS receives the γ sent by EU_i and uses κ to decrypt it to get $\sigma_{EU} \leftarrow Enc_{\kappa}(\gamma)$, and then runs the verification algorithm to verify σ_{EU}. If the output is valid, κ is used as the session key and accept the connection; if the output is invalid, reject the connection.

4 | ANALYSIS AND COMPARISON

4.1 | Security analysis of local authentication protocol

We proved that the proposed scheme is backward unlinkability, traceability, and non-frameability.

1. Backward unlinkability.

Backward unlinkability is used to ensure that even if the corresponding signer has been revoked, the signer's identity will not be revealed from the signature. For any adversary A and safety parameter $\lambda \in N$, we define the experiment $Exp_A^{BU}(\lambda)$ as follows:

(1) Select $b \xleftarrow{\$} \{0, 1\}$, and use the key generation algorithm to generate the master public key gpk, master private key msk and the registration table reg;

(2) Let $HU := \emptyset; CU := \emptyset; RU := \emptyset$, calculate $b' \leftarrow A^{Add,WReg,Key,Revoke,Sign,Ch}(gpk)$. Among them, *Add* is an oracle to add users. It allows attacker A to add an honest edge terminal device to the group. After entering the identity EU_i and the expiration time, the oracle is calculated by running the join/issue algorithm (gsk_i, τ_i), and add the identity to the HU; *WReg* a write-registration-table, it updates the registry $reg[i]$ to M after entering the identities EU_i and M; *Key* is the user key predictor that displays (gsk_i, usk_i) and adds the identity EU_i to the CU; *Revoke* is the revocation oracle, allowing A to revoke the honest user, so that the oracle is in called at time $t-1$. When the identity EU_i is input, the oracle calculates $RL_t \leftarrow Revoke(gpk, msk, t, reg, RU_t)$, outputs (ei_t, RL_t), and add RU_t to RU; *Sign* is a signing oracle machine, which is called at time t to calculate the signature $\sigma \leftarrow Sign(gpk, gsk_i, usk_i, m, t, ei_t)$ and return to σ; Finally, Ch is the challenge oracle machine, which selects EU_0, EU_1 in the HU, and calculates $\sigma^* \leftarrow Sign(gpk, gsk_{i'}, usk_{i'}, m^*, t^*, ei_{t^*})$ and return σ^*.

(3) If $b' = b$, it returns 1, otherwise it returns 0.

(4) For any adversary A, the advantage of success $Adv_A^{BU}(\lambda) := |\ Pr\left[Exp_A^{BU}(\lambda) = 1\right] - 1/2\ |$ can be ignored, the scheme is not backward linkable.

Let us use a series of games to simulate event E_1.

Game G_0: The same as the definition of backward unlinkability.

Game G_1: It is basically the same as game G_0 in event E_1, except that the challenge signature is programmed and calculated by the random oracle machine H.

Game G_2: Basically the same as game G_0 in event E_1, the difference is: challenge signature $\sigma^* = (\psi_1^*, \psi_2^*, \psi_3^*, \psi_4^*, \psi_5^*, \psi_6^*, \psi_7^*, c^*, s_\alpha^*, s_\beta^*, s_\zeta^*, s_\xi^*, s_{\zeta'}^*, s_{\xi'}^*, s_u^*, s_x^*, s_\delta^*, s_{\delta'}^*), \psi_2^*, \psi_3^* \xleftarrow{\$} \mathbb{G}_1$.

Game G_3: Basically the same as game G_1 in event E_1, the difference is: challenge signature $\sigma^* = (\psi_1^*, \psi_2^*, \psi_3^*, \psi_4^*, \psi_5^*, \psi_6^*, \psi_7^*, c^*, s_\alpha^*, s_\beta^*, s_\zeta^*, s_\xi^*, s_{\zeta'}^*, s_{\xi'}^*, s_u^*, s_x^*, s_\delta^*, s_{\delta'}^*), \psi_5^* \xleftarrow{\$} \mathbb{G}_1$.

Suppose that the successful event of A in game G_i is P_i.

First, let $\left(\left(G_1, G_2, G_T, e, f, \widehat{f}\right), f^a, f^b, Z\right)$ be an instance of DDH1. We construct an algorithm B to distinguish whether $Z = f^{ab}$ is true. B first set $Z := a, \gamma_O := b$ and $\widehat{g} := f^r$, and select $r \xleftarrow{\$} \mathbb{Z}_p$. Since B has all secret values, B can respond to all queries sent by A. In the challenge phase, B chooses all values except $f, \tilde{g}, \widehat{g}$, where $\tilde{g} = f^{\gamma_O} = f^b$, and then select (A, B_{t^*}) according to the Scheme. B then set $\psi_1^* := f^a, \psi_2^* := AZ, \psi_3^* := B_{t^*} Z^r$, and calculate all components except σ^*. If $Z = f^{ab}$, it means that B simulates the game G_1 correctly, otherwise it means that B simulates the game G_2 correctly, thus proving that $|\ Pr\left[P_1\right] - Pr\left[P_2\right]\ | \leq Adv_{DDH1}(\lambda)$.

Then suppose $\left((G_1, G_2, G_T, e, g_1, g_2), g_1', g_2', h, \widehat{h}, g^a, {g_1'}^b, Z\right)$ is an instance of DLIN.

Algorithm B is used to distinguish whether the equation $Z = h^{a+b}$ holds. B guesses the time when EU_{i^*} will join the group and the challenge time t^* with probability of $1/q_A$ and $1/q_R$, respectively. Suppose B guesses correctly. B first implicitly set $x_{i^*} := a, \gamma_O := b$ and $y_{t^*} := y_{t^*}' c$, and select $y_{t^*}' \xleftarrow{\$} \mathbb{Z}_p$. For some $c \in \mathbb{Z}_p, g_1' := g_1^c, g_2' := g_2^c$. B chooses $y_t \xleftarrow{\$} \mathbb{Z}_p$. Since B has the secret keys of all signers except EU_{i^*}, if these have nothing to do with EU_{i^*}, B can respond to all queries sent by A.

For revocation query, when $t = t^*$, EU_{i^*} has not been revoked; when $t \neq t^*$, B can be calculated by $grt_{i^*,t} = (g^a)^{y_t}$ to revoke EU_{i^*}, which proves that the scheme has backward unlinkability.

For signing queries at $t = t^*$, B randomly selects $\psi_1, \psi_2, \psi_3 \xleftarrow{\$} \mathbb{G}_1, \beta, d \xleftarrow{\$} \mathbb{Z}_p$, and calculates $\psi_4 = \tilde{h}_{t^*}^{\beta}$, then the equation $\psi_4^{1/y_{t^*}} = \left(\left(g^{y'_{t^*}c}\right)^{\beta}\right)^{1/y'_{t^*}c} = g^{\beta}$ holds. B calculate $\psi_5 = \left(g^a g^{\beta}\right)^d, \psi_6 = g^d, \psi_7 = g_2^d$, then the revocation check relation equation $e\left(\psi_5, \hat{h}_{t^*}\right) = e\left(\left(g^a g^{\beta}\right)^d, g_2^{y'_{t^*}c}\right) = e\left(\left(g^a \psi_4^{1/y'_{t^*}}\right)^d, g_2^{y'_{t^*}c}\right) = e\left(\left(g^{y'_{t^*}c}\right)^{x_{i^*}} \psi_4, g_2^d\right) = e\left(grt_{i^*,t^*}\psi_4, \psi_7\right)$ holds.

For signing queries at $t \neq t^*$, B selects $\psi_1, \psi_2, \psi_3, \psi_4 \xleftarrow{\$} \mathbb{G}_1, d \xleftarrow{\$} \mathbb{Z}_p$, computes $\psi_5 := \left(g^a \psi_4^{1/y_t}\right)^d, \psi_6 = g^d, \psi_7 = g_2^d$. Then revocation check relation equation $e\left(\psi_5, \hat{h}_t\right) = e\left(\left(g^a \psi_4^{1/y_t}\right)^d, g_2^{y_t}\right) = e\left((g^a)^{y_t}\psi_4, g_2^d\right) = e\left(\psi_7, grt_{i^*,t}\psi_4\right)$ holds.

To calculate the challenge signature, B selects $\psi_1^*, \psi_2^*, \psi_3^* \xleftarrow{\$} \mathbb{G}_1$, calculates $\psi_4^* = \left(g_1'^{\,b}\right)^{y'_{t^*}} = \left(g^{cb}\right)^{y'_{t^*}} = \left(g^{y'_{t^*}c}\right)^b = \tilde{h}_{t^*}^{\beta^*}$, sets $\psi_5^* = Z, \psi_6^* = h, \psi_7^* = \hat{h}$. If $Z = h^{a+b}$, it means that B simulates the game G_2 correctly, otherwise it means that B simulates the game G_3 correctly. Since the challenge signature does not depend on the challenge bit now, $Pr\left[P_3\right] = 1/2$, which proves that $|\ Pr\left[P_2\right] - Pr\left[P_3\right]\ | \leq Adv_{DLIN}\left(1/\left(q_A q_R\right) - q_s q_h/p\right)$.

2. Traceability.

Traceability, just as its name implies, is to ensure that valid signatures are tracked. It not only ensures that an adversary who does not have a signature key cannot calculate a valid signature, but also ensures that after the signing key reaches the expiration time, no adversary can use the key to generate a valid signature . You can also find those terminal devices that have illegally leaked the keys and revoke them. For any adversary A and safety parameter $\lambda \in N$, we define the experiment $Exp_A^{Trace}(\lambda)$ as follows:

(1) First, generate the master public key gpk, the master private key msk and the registration table reg, set $CU := \emptyset; Set := \emptyset$;

(2) Calculate $(\sigma^*, m^*, t^*) \leftarrow A^{STI,RReg,Revoke,Sign}(gpk)$. Among them, STI is sent to the issue oracle, which allows A to represent the damaged edge terminal equipment EU_i to participate in the protocol, assigns gsk_i, τ_i, usk_i to A, and adds EU_i to CU; RReg is the write-registration-table oracle, which is used to display the contents of the registration table $reg[i]$; Revoke is the oracle to revoke, allowing A to revoke the honest user. When the time is $t-1$, the oracle is called. After inputting RU_t, it runs $RL_t \leftarrow Revoke\left(gpk, msk, t, reg, RU_t\right)$, and finally outputs $\left(ei_t, RL_t\right)$; Sign is the signing oracle. When the time is t, the oracle is called, and after inputting i and m, it runs $\sigma \leftarrow Sign\left(gpk, gsk_i, usk_i, m, t, ei_t\right)$. Finally Sign returns σ, and adds (m, t, σ) to Set;

(3) Next, if it is satisfied that the verification algorithm returns valid, $(m^*, t^*, \sigma^*) \notin Set$, and condition a: $i \notin CU \backslash RU_{t^*}$ or $i = \perp$ obtained from the output of the opening algorithm; and condition b: the output of the opening algorithm is $i \in CU \backslash RU_{t^*}$ and $\tau_t < t^*$ returns 1 if one of them is satisfied, otherwise it returns 0;

(4) If the advantage of success for any adversary A $Adv_A^{Trace}(\lambda) :=|\ Pr\left[Exp_A^{Trace}(\lambda) = 1\right] - 1/2\ |$ is negligible, so the scheme is traceable.

Since in the joining algorithm, an edge terminal device sends $X_i = h_2^{usk_i}$, the edge server uses the signature key of the BBS + signature scheme to sign usk_i, so that $A_j = \left(g_1 h_0^{\zeta_j} h_1^{u_j} X_i\right)^{1/\xi_j + \gamma_A}$ and the BBS + signature scheme is that the edge server can sign usk_i without knowing usk_i, so we need to introduce the KOSK assumption to require the adversary to disclose the keys of honest users. In the end, our solution satisfies the traceability of the random oracle model under the q-SDH assumption and the KOSK assumption.

3. Non-frameability.

Non-frameability is used to ensure that no adversary can produce a valid signature that can be traced to an honest signer. Here, an honest signer refers to a signer who cannot let the adversary know usk_{i^*}. For any adversary A and safety parameter $\lambda \in N$, we define the experiment $Exp_A^{Nf}(\lambda)$ as follows:

(1) First, generate the master public key gpk, the master private key msk and the registry reg, set $HU := \emptyset; CU := \emptyset; Set := \emptyset$;

(2) Calculate $(\sigma^*, m^*, t^*, i^*) \leftarrow A^{STU,RReg,Usk,Sign}(gpk, msk)$, where STU is sent to the user oracle, which allows A represents the damaged edge server to participate in EU_i's joining protocol, and then adds EU_i to the HU; RReg is the read-registration-table oracle, which is used to display the contents of $reg[i]$; Usk is the user-secret-keys oracle, it is used to display $\left(gsk_i, usk_i\right)$ on the input EU_i and add EU_i to the CU; Sign is the signing oracle. When the time is t, this oracle is called, and after inputting i and m, it runs $\sigma \leftarrow Sign\left(gpk, gsk_i, usk_i, m, t, ei_t\right)$. Finally it returns to σ, and adds (m, t, σ) to Set;

(3) If it is satisfied that the verification algorithm returns valid and the i^* obtained by the opening algorithm output satisfies $i^* \in HU \bigwedge i^* \notin CU \bigwedge (m^*, t^*, \sigma^*) \notin Set$, then it returns 1. Otherwise return 0;

(4) If for any adversary A, the advantage of success $Adv_A^{Nf}(\lambda) :=|\ Pr\left[Exp_A^{Nf}(\lambda) = 1\right]\ |$ is negligible, then the program is Non-frameability.

Let $\left(G_1, G_2, G_T, e, g, \tilde{g}, \hat{g}, \tilde{g}^x\right)$ be a DL instance, an algorithm B is used to solve the DL problem. Assuming that q_A is the number of queries sent to the user oracle, B guesses the terminal device $i^* \in \left[1, q_A\right]$ output by A in the final stage. Assuming that the guess is correct and the probability is $1/q_A$, B chooses $\theta_2 \xleftarrow{\$} \mathbb{Z}_p$, set $h_2 = \tilde{g}^{\theta_2}, usk_{i^*} := x, X_{i^*} := \left(\tilde{g}^x\right)^{\theta_2} = h_2^x, \tilde{X}_{i^*} := \tilde{g}^x$, then B selects all other values. When adding i^* to the group through STU query, B selects $s_x, c_x \xleftarrow{\$} \mathbb{Z}_p$, sets $c_x := H'\left(X_i, \tilde{X}_{i^*}, h_2^{s_x}/X_{i^*}^{c_x}, \tilde{g}^{s_x}/\tilde{X}_{i^*}^{c_x}\right)$, and sends (s_x, c_x) to A. A outputs the signature after receiving it, and B extracts x^* from the signature output by A, the extracted x^* satisfies $X_{i^*} = h_2^{x^*}$. Therefore, if the extraction effect is good, B outputs x^*. It can be concluded that our scheme satisfies the non-frameability under the DL assumption.

To sum up, in the local authentication protocol proposed in this article, the complete subtree algorithm is adopted to revoke expired devices. It ensures that even if the edge server is dishonest, the solution still has backward unlinkability and non-frameability.

4.2 | Security analysis of roaming identity authentication

1. Anonymity.

Anonymity means that the signer of the signature can know whether the signature was signed by him/her, but the adversary cannot determine which of the two members generated the signature. Next, we first define the game between adversary A and challenger C.

(1) C first obtains (gpk, msk), and A obtains gpk;

(2) Next, A sends a query to C. A first requests the creation of a new edge terminal device with a specified key expiration time, and C executes the user joining phase locally and generates (gsk_i, grt_i). C then returns the key of the edge terminal device gsk_i to A and returns the signature of the message m signed by gsk_i at time t and the revocation token grt_i of the edge terminal device;

(3) A outputs the message m and the index i_0, i_1 that will never be sent as a damaged or revoked query. After C randomly selects a bit b uniformly, it uses gsk_{i_b} to calculate the signature σ^* on m and sends σ^* to A;

(4) Similar to the step of (2), A continues to send queries to C except for the inability to issue the revocation query except the index i_0, i_1 in (3);

(5) Finally, A outputs its guess b' on b. If $b' = b$, we say that A wins the game, and the advantage of A to win the game is $| Pr[b' = b] - 1/2 |$.

Let $\left(g, g^{\tilde{a}}, g^{\tilde{b}}, g^{\tilde{c}}\right)$ be an instance of DDH, we construct an algorithm B to solve the DDH problem, that is, B tries to answer whether the equation $\tilde{c} = \tilde{a}\tilde{b}$ holds. B first honestly generates the master public key $gpk = \left(g_1, g_2, h, \tilde{g}, \tilde{h}, z_1, z_2, H, l, Q, Q^z, Q^{z^2}, \ldots, Q^{z^l}\right)$, the master private key msk is provided to A.

A can issue any query to B, because B has the master key, and therefore can answer all these queries. Assuming that A performs q join queries at this stage, we denote the index of this edge device as i^*. If $i \neq i^{**}$, then B will answer all queries honestly. If $i = i^*$ is added to the query, B calculates $\{\tau_{i^*j}\}_{j\in[1,l]} \leftarrow 1 - ENC(\tau_{i^*})$, where τ_{i^*} is the expiration time specified by A of the user EU_{i^*}; C returns $\perp$ when $t > \tau_{i^*}$. If $i = i^*$ and $t < \tau_{i^*}$ in the signing query, B selects a random value $\gamma \in \mathbb{Z}_p$, and calculates $= g^{\gamma}, J = \left(g^{\tilde{a}}\right)^{\gamma}, K = \left(g^{\tilde{a}}\right)^{\gamma y_j}$, then B selects random values $T, W, V_1, V_2 \in \mathbb{G}_1$, $c, s_{\alpha_1}, s_{\alpha_2}, s_{\alpha_3}, s_{\alpha_4}, s_{\theta_1}, s_{\theta_2}, s_{\theta_3}, s_{\theta_4}, s_{\theta_5}, s_{\theta_6}, s_x, s_f, s_t \in \mathbb{Z}_p$, calculates $\tilde{R}_1, \tilde{R}_2, \tilde{R}_3, \tilde{R}_4, \tilde{R}_5, \tilde{R}_6, \tilde{R}_7, \tilde{R}_8, \tilde{R}_9, \tilde{R}_{10}$ these values, $Y \leftarrow Q^{\prod_{j=1}^{l}(z+t_j)}$ and $\{t_j\}_{j\in[1,l]} \leftarrow 0 - ENC(t)$. Finally, B complements the random oracle H and sets $c = H\left(gpk, t, m, B, , J, K, T, W, V_1, V_2, \tilde{R}_1, \tilde{R}_2, \tilde{R}_3, \tilde{R}_4, \tilde{R}_5, \tilde{R}_6, \tilde{R}_7, \tilde{R}_8, \tilde{R}_9, \tilde{R}_{10}\right)$. If $i = i^*$ in the revocation query, B will stop.

In the challenge phase, A outputs indexes i_0, i_1 and t, m. If $i^* \in \{i_0, i_1\}$, S is like a signing query with input (i^*, t, m) to generate a signature, except that B sets $B = g^{\tilde{b}}, J = g^{\tilde{c}}, K = \left(g^{\tilde{c}}\right)^{y_j}$; if $i^* \notin \{i_0, i_1\}$, B stops. In the second query stage, because A cannot send a damaged or revoked query to EU_{i^*}, B will not stop, and the rest of the query will be the same as the previous query stage. Finally, A outputs the guess bit b''. If $i_{b'} = i^*$, B guesses $\tilde{c} = \tilde{a}\tilde{b}$, otherwise, B guesses $\tilde{c} \neq \tilde{a}\tilde{b}$.

If B does not stop and $\tilde{c} = \tilde{a}\tilde{b}$, the challenge signature is the same as the signature generated by EU_{i^*}, the probability of A being correct $\varepsilon > 1/2$, and if A is correct, B is also correct. If B does not stop, but $\tilde{c} \neq \tilde{a}\tilde{b}$, then the challenge signature has nothing to do with i_0 and i_1, the probability of A guessing right is $\varepsilon = 1/2$, and the probability of B being correct is also 1/2. Therefore, the overall probability of success for B is $\varepsilon/2 + 1/2 \times 1/2$. Since $\varepsilon > 1/2$, the success probability of B is also greater than 1/2, which proves that under the DDH assumption, the group signature scheme in this article is anonymous in the random oracle model.

2. Traceability

If the adversary cannot forge a valid signature that can be opened improperly, then we can say that the scheme satisfies traceability. Supposing that the adversary destroys the traceability of the group signature scheme with a probability of at least ε, an algorithm B needs to be constructed to solve the q-SDH problem in $\mathbb{G}_1$.

First assume that B has an (nl + 1)-SDH instance, where n is the maximum number of joining queries made by A. In the initial stage, given a series of elements of B $\left(g_1', g_2', g_2'^{\gamma_2}, \ldots, g_2'^{\gamma_2^2}, \ldots, g_2'^{\gamma_2^{nl}}, g_2'^{\gamma_2^{nl+1}}\right)$, calculate $\left(c, g_1'^{1/\gamma_2+c}\right)$, select $\left\{x_{ij}'\right\}$, where $i \in (1, n), j \in (1, l)$, calculate $g_2 = g_2'^{\prod_{i=1,j=1}^{i=n,j=l}\left(\gamma_2+x_{ij}'\right)}$, the final master public key is $gpk = \left(g_1, g_2, h, \tilde{h}, \tilde{g}, z_1, H, l, Q, Q^z, Q^{z^2}, \ldots, Q^{z^l}\right)$.

A first sends a joining query to B. According to the type of joining query, B selects a random $\hat{f}$ or a $\hat{f}$ that satisfies $\hat{F} = h^{\hat{f}}$ extracted from A, and next B needs to calculate the correct key for the edge terminal device. For $j \in (1, l)$, B calculates A_j, x_j as follows: $e\left(A_j, z_1^{r_j} z_2 g_2^{x_j}\right) = e(g_1 F, g_2)$, and B then returns the terminal device's key gsk_i and the signature calculated by the signing algorithm to A and adds EU_i to U. A finally outputs a message M^*, the expiration time t^*, a revocation list RL^* and a signature σ^*. A set of values (A^*, t^*, x^*, f^*) can be extracted from the forged signature so that $e\left(A^*, z_1^{t^*} z_2 g_2^{x^*}\right) = e\left(g_1 h^{f^*}, g_2\right)$, because the signature σ^* is actually the signature knowledge proof of this relationship; and due to that the adversary can forge the signature corresponding to the honest user, x^* is equivalent to x_{ij} so that the terminal device can be processed in the next request

without being destroyed. According to the above relationship, A^* satisfies: $A^* = \left(g_1^{1+af^*}\right)^{1/\gamma_2+\gamma_1 t^*+x^*}$, $(A^*)^{1/1+a\hat{f}} = (g_1)^{1/\gamma_2+\gamma_1 t^*+x^*} = \psi\left(g_2^{1/\gamma_2+\gamma_1 t^*+x^*}\right)$. Use x' to represent the value $\gamma_1 t^* + x^*$. Unless the edge terminal device is destroyed, $x_{i,j}$ will never be known by A. So basing on the discrete logarithm assumption, the probability of $x' = x'_{i,j}$ for any i, j can be ignored. Therefore, in the case of $x' \neq x'_{i,j}$, let $F(\gamma) = (\gamma + x')\, G(\gamma) + R$, where $G(\gamma)$ is a polynomial of degree nl-1 and R is a constant, so $(A^*)^{1/1+a\hat{f}} = \psi\left(g_2^{(R/x'+\gamma_2)+G(\gamma)}\right)$. Therefore, B can submit $[(A^*)^{1+a\hat{f}}\psi\left(g_2'^{-G(\gamma)}\right)]^{1/R}$ and x' as solutions to the $nl + 1 - SDH$ problem, which proves that our scheme satisfies traceability in the random oracle model under the q-SDH assumption.

3. Forgivability

If an adversary can forge an honest member's signature so that the member cannot raise objections, the scheme is unforgivable.

First, the algorithm B uses the nl-SDH problem instance to generate nl-1 problem keys, and then randomly selects a $x_{i',j'}$ from m the remaining keys and puts it into $grt_{i'}$. If the signature query is related to $x_{i',j'}$, B responds to the query; if it is irrelevant, then B has the signature key of the device and generates the signature normally. Finally, A submitted a forged document such that $x^* = x_{i',j'}$, where the forged signature is related to (A^*, t^*, x^*, f^*), the forged probability is 1/nl. Since this corresponds to a new SDH tuple, B can obtain the solution to the nl-SDH problem.

Then, since the server's identity verification is implemented by query-response pairs $(HS, ID, g^{r_{EU}})$ and σ_{FS}, and the signature scheme is unforgeable, so no device can generate a valid signature other than the foreign server that has the signature key sk_{FS}. Meanwhile, the server must generate a valid signature on the signature currently selected by the device, so the response attachment cannot work either.

Subscription verification is achieved through a message signature pair (m_U, σ_U). Since the scheme has been proved to be unforgeable, only the home server's legal device can contain the $g^{r_{FS}}$ related to the foreign server. Man-in-the-middle attacks are also invalid because $g^{r_{EU}}$ and $g^{r_{FS}}$ are sent in an authenticated manner, just the edge terminal device and the foreign server can export the session key. The anonymity of the user is achieved by the group signature scheme's anonymity, because only the main server contains the revocation token, so the foreign server not only has no revocable token, but also cannot find the signer's identity. The underlying group signature can also realize the non-traceability of the user. Because the group signature is selfless-anonymous, it also leads to the unlinkability of the signature.

4.3 | Efficiency analysis of roaming identity authentication protocol

This article compares the performance of our scheme with He[31] and Yang[32] and Kuo[33] et al.'s two roaming protocols that support user revocation.

First, consider the authentication on the device side. For the signature scheme, we use ECDSA[31] which takes 1 $\mathbb{G}_1$ exponentiation operation for signing, and 1 for verification. Then according to the benchmark of jPBC, the time required for various calculations in the system realization is estimated, and the efficiency is analyzed. We only calculate the time required for exponentiation and pairing. For exponentiation, while optimizing the base of exponentiation, we use the prenegotiation algorithm to process the data in advance, which improves the calculation speed. For pairing, we also optimized those elements where one of the paired elements is a constant. The specific analysis results are shown in Table 1.

Next, consider the performance of the remote domain authentication server. The main analysis here is the performance of the server's revocation check. This article first analyzes and verifies the time of a device. For each time the remote domain authentication server checks whether the current roaming user is consistent with the user in the revocation list, Huang and others need 2 pairing operations to complete the revocation check; Sammy needs 1 pairing operation; and the solution in this article requires 1 $\mathbb{G}_1$ to be exponentiated. Finally, this article analyzes the running time, as shown in Table 2.

The remote domain authentication server needs to check whether the terminal is in the revocation list, so we also check the time it takes for the roaming server to check whether the terminal is in the revocation list as the number of revoked terminals increases with time. Because in revocation, there are not only natural revocation caused by the expiration of the terminal contract, but also premature revocation caused by key leakage and so forth. Therefore, this article also considers the inspection time spent when the terminal is naturally revoked and revoked in advance in different proportions.

TABLE 1 Performance for authentication at device side

Operation	He	Yang	Kuo	Ours
G_1/G_2 exp (no precompute)	4	3	3	9
G_1/G_2 exp (precompute)	3	10	8	24
Pairing (no constant)	1	0	1	1
Pairing (1 constant)	2	5	4	8
Running time (s)	2.634	2.543	2.568	6.411

TABLE 2 Performance for authentication at remote domain authentication server

Operation	He	Yang	Kuo	Ours
G_1/G_2 exp (no precompute)	4	4	6	10
G_1/G_2 exp (precompute)	2	7	10	21
Pairing (no constant)	0	0	0	0
Pairing (1 constant)	4	3	4	6
Running time (s)	0.1228	0.1351	0.1325	0.3241

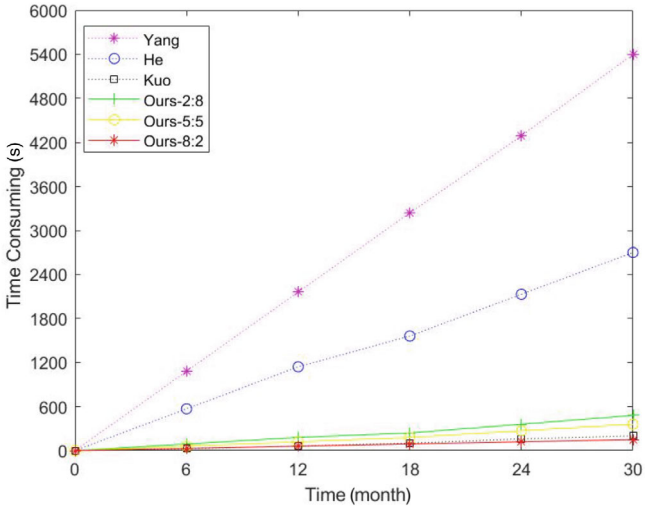

FIGURE 6 Time to check whether a device is in the revocation list

As can be seen from Figure 6, in the scheme proposed by Yang, He and Kuo, with the increase of time, the number of revoked devices is increasing, and the time consumed by revocation inspection of each device is gradually increasing, and it will not change according to the change of the ratio of natural revocation to premature revocation. In the protocol proposed in this article, among all revoked devices, the more the proportion of natural revocation is, the less time it takes to check.

To sum up, the protocol proposed in this article is generally acceptable, although it takes a lot of time in device-side authentication. At the same time, in practical work, the remote domain authentication server needs to reduce the revocation check time as much as possible, so the work of this article effectively improves the efficiency of revocation check, thus effectively improving the performance of roaming identity authentication protocol.

5 | CONCLUSION

This article proposes an anonymous local identity authentication and roaming identity authentication protocol based on a zero-trust architecture. Both protocols rely on group signature schemes, and both support effective revocation of certificates after their natural expiration. In this scheme, the expiration time can be bound to the key of each edge terminal device, so that the revocation list does not contain expired keys, which greatly improves the efficiency of revocation check and keeps the calculation cost of the signature algorithm constant. Security analysis shows that the scheme of this article has anonymity, traceability and other secure communication requirements. But in the local identity authentication protocol, we use a secret key knowledge (KOSK) assumption to send a signing query to the group signature scheme without knowing the message, but the KOSK assumption cannot be used in the existing public key infrastructure. Therefore, it is best to avoid using the KOSK assumption as much as possible. It is a future work of this article to delete this assumption.

ACKNOWLEDGMENTS

This work was supported in part by the State Grid Jiangxi Information Telecommunication Company Project "Research on de-boundary security protection technology based on zero trust framework" under Grant 52183520007V.

CONFLICT OF INTEREST

The authors declare that they have no conflicts of interest.

DATA AVAILABILITY STATEMENT

Data sharing not applicable to this article as no data set were generated or analyzed during the current study.

ORCID

Yuancheng Li https://orcid.org/0000-0001-8074-7259

REFERENCES

1. Eschenauer L, Gligor VD, Baras J. On trust establishment in mobile ad-hoc networks. In: Christianson B, Crispo B, Malcolm JA, Roe M, eds. *Security Protocols*. Lecture Notes in Computer Science. Vol 2845. Springer; 2002:47-66.
2. Blaze M, Feigenbaum J, Ioannidis J, Keromytis AD. The role of trust management in distributed systems security. In: Vitek J, Jensen CD, eds. *Secure Internet Programming*. Lecture Notes in Computer Science. Vol 1603. Springer; 1999:185-210.
3. Chu YH, Feigenbaum J, LaMacchia B, Resnick P, Strauss M. REFEREE: trust management for web applications. *Comput netw ISDN syst*. 1997;29(8–13):953-964.
4. Kindervag J, Balaouras S, Coit L. No more chewy centers: introducing the zero trust model of information security. *For Res*. 2010;3.
5. Ward R, Beyer B. BeyondCorp: a new approach to enterprise security, 2014.
6. Rose SW, Borchert O, Mitchell S, Connelly S. *Zero Trust Architecture*. 2020. Special Publication (NIST SP), National Institute of Standards and Technology, Gaithersburg, MD, [online], https://doi.org/10.6028/NIST.SP.800-207, https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=930420
7. Tian Y. Network security protection based on zero trust architecture. Information Technology and Informatization; 2020:154-157.
8. Chen T, Huang SH. Tree parity machine-based one-time password authentication schemes. Proceedings of the 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence); 2008:257-261; IEEE.
9. Lamport L. Password authentication with insecure communication. *Commun ACM*. 1981;24(11):770-772.
10. Haller NM. The s/key (tm) one-time password system. Proceedings of the Internet Society Symposium on Network and Distributed Systems; 1994:151-157.
11. Haller N, Metz C, Nesser P, Straw M. A one-time password system. Network Working Group Request for Comments. vol. 2289; 1998.
12. Chefranov AG. One-time password authentication with infinite hash chains. In: Sobh T, Elleithy K, Mahmood A, Karim MA, eds. *Novel Algorithms and Techniques In Telecommunications, Automation and Industrial Electronics*. Springer; 2008:283-286.
13. Sun HM. An efficient remote use authentication scheme using smart cards. *IEEE Trans Consum Electron*. 2000;46(4):958-961.
14. Yeh TC, Shen HY, Hwang JJ. A secure one-time password authentication scheme using smart cards. *IEICE Trans Commun*. 2002;85(11):2515-2518.
15. Shimizu A. A dynamic password authentication method using a one-way function. *Syst Comput Japan*. 1991;22(7):32-40.
16. Chien HY, Jan JK, Tseng YM. A modified remote login authentication scheme based on geometric approach. *J Syst Softw*. 2001;55(3):287-290.
17. Goyal V, Abraham A, Sanyal S, Han SY. The N/R one time password system. Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05); 2005:733-738; IEEE.
18. Li CT, Hwang MS. An efficient biometrics-based remote user authentication scheme using smart cards. *J Netw Comput Appl*. 2010;33(1):1-5.
19. Hallsteinsen S, Jorstad I, Van Thanh D. Using the mobile phone as a security token for unified authentication. Proceedings of the 2007 Second International Conference on Systems and Networks Communications (ICSNC 2007); 2007:68; IEEE.
20. Cui Z, Fei X, Zhang S, et al. A hybrid blockchain-based identity authentication scheme for multi-WSN. *IEEE Trans Serv Comput*. 2020;13(2):241-251.
21. Zhang S, Cao Y, Ning Z, Xue F, Cao D, Yang Y. A heterogeneous IOT node authentication scheme based on hybrid blockchain and trust value. *KSII Trans Internet Inf Syst*. 2020;14(9):3615-3638.
22. Shen M, Liu H, Zhu L, et al. Blockchain-assisted secure device authentication for cross-domain industrial IoT. *IEEE J Sel Areas Commun*. 2020;38(5):942-954.
23. Azrour M, Mabrouki J, Guezzaz A, Farhaoui Y. New enhanced authentication protocol for internet of things. *J Big Data Min Anal*. 2021;4(1):1-9.
24. Wang W, Huang H, Xue L, Li Q, Malekian R, Zhang Y. Blockchain-assisted handover authentication for intelligent telehealth in multi-server edge computing environment. *J Syst Archit*. 2021;115:102024.
25. Li JG, Gang S, Zhang YC. Practical group signature scheme with verifier-local revocation. *J Commun*. 2011;32(10):67.
26. Perera MNS, Koshiba T. Fully dynamic group signature scheme with member registration and verifier-local revocation. In: Ghosh D, Giri D, Mohapatra R, Sakurai K, Savas E, Som T, eds. *Mathematics and Computing*. ICMC 2018. Springer Proceedings in Mathematics & Statistics. Vol 253. Springer; 2018:399-415.
27. Ling S, Nguyen K, Roux-Langlois A, Wang H. A lattice-based group signature scheme with verifier-local revocation. *Theor Comput Sci*. 2018;730:1-20.
28. Zhang Y, Liu X, Yin Y, Zhang Q, Jia H. On new zero-knowledge proofs for fully anonymous lattice-based group signature scheme with verifier-local revocation. In: Zhou J, Conti M, Ahmed CM, Au MH, Batina L, Li Z, Lin J, Losiouk E, Luo B, Majumdar S, Meng W, Ochoa M, Picek S, Portokalidis G, Wang C, Zhang K. eds. *Applied Cryptography and Network Security Workshops. ACNS 2020*. Lecture Notes in Computer Science. Springer; 2020:381-399.
29. Wang L, Zhang K, Qian H, Chen J. Group signature with verifier-local revocation based on coding theory. *Secur Commun Netw*. 2021;2021:1-12.
30. Sanders O. Improving revocation for group signature with Redactable signature. In: Garay JA, ed. *Public-Key Cryptography – PKC 2021*. Lecture Notes in Computer Science. Vol 12710. Springer; 2021:301-330.
31. He D, Bu J, Chan S, Chen C, Yin M. Privacy-preserving universal authentication protocol for wireless communications. *IEEE Trans Wirel Commun*. 2010;10(2):431-436.

32. Yang G, Huang Q, Wong DS, Deng X. Universal authentication protocols for anonymous wireless communications. *IEEE Trans Wirel Commun*. 2010;9(1):168-174.
33. Kuo WC, Wei HJ, Cheng JC. An efficient and secure anonymous mobility network authentication scheme. *J Inf Secur Appl*. 2014;19(1):18-24.

How to cite this article: Liu H, Ai M, Huang R, Qiu R, Li Y. Identity authentication for edge devices based on zero-trust architecture. *Concurrency Computat Pract Exper*. 2022;34(23):e7198. doi: 10.1002/cpe.7198