Lightweight identity authentication scheme for IoT devices based on blockchain

Wanli Ma¹², Qi Zhang¹, Junwei Ma¹, Honglin Xue¹, Xiaowei Hao¹, Haitao Li¹, Weidong Jia¹, Yaning Meng¹, Zhifang Wen¹ and Jianliang Zhang¹

¹ Information and Communication Branch, State Grid Shanxi Electric Power Company, No.3, Heyuan Road, Changfeng business district, Jinyuan District, Taiyuan, China
² This work was supported by Science and Technology Project of State Grid Shanxi Electric Power Company, Project No 52051C19006S.

E-mail: MaWanLi@sx.sgcc.com.cn

Abstract. Aiming at the resource limitation and security problems of Internet of things devices, a lightweight identity authentication scheme is proposed. In the scheme, the identity authentication model is designed, and the model is implemented by using the blockchain smart contract. Through the analysis, the scheme can effectively complete the identity authentication under the condition of limited equipment resources.

1. Introduction
With the wide application of Internet of things technology, more and more terminal devices are connected to the Internet of Things (IoT), and a series of new security challenges have emerged [1,2]. The limited resources of IoT devices aggravate these security challenges, which makes the traditional communication protocols and security schemes inefficient or even impossible to achieve for the IoT. In view of the widespread existence of IoT devices and their adoption in key applications, the security issues related to the IoT are becoming more and more worrying. In the case of serious threat to life safety, the outbreak of any security vulnerability will have a profound impact. At this time, security becomes the highest requirement, especially considering the possible damage caused by malicious devices without authentication in the Internet of things system, authentication is particularly important. Hsiu-Lien Yeh et al. [3] propose a novel user authentication protocol for wireless sensor networks (WSNs), using Elliptic Curves Cryptography (ECC) and smart cards. The authentication phase involves three entities: user, gateway node and login node. The user registers with the gateway node in advance. During authentication, the user sends a login request to the login node, and the login node applies to the gateway node for authentication of the user. The authentication result is returned to the user by the login node. Generally speaking, the authentication process is complicated. Moosavi S. R. et al. [4] propose an architecture that authentication and authorization of a remote end-user is done by distributed smart e-health gateways to unburden the medical sensors from performing trivial tasks, this proposed architecture relies on the certificate-based Datagram Transport Layer Security (DTLS) handshake protocol, therefore, the certification process is complex. Sheetal Kalra et al. [5] propose a secure ECC based mutual authentication protocol for secure communication of embedded devices and cloud servers using Hyper Text Transfer Protocol (HTTP) cookies. The proposed scheme achieves mutual
authentication and provides essential security requirements. Chen Wang et al. [6] present a novel system named emergency medical system without the assistance of doctors. The authors claim that the scheme can shorten the time of device authentication in an emergency to achieve fast authentication. However, a smart mobile device cannot specify who will authenticate it.

This paper proposes a lightweight identity authentication scheme based on identity and bilinear mapping. In the scheme, the intelligent device is required to solidify an identity in its hardware as its identity ID. Before the smart device is connected to the Internet of things, the trusted organization initializes the key for it, and stores its public information on the blockchain. The identity authentication of smart devices is realized by identity based signature scheme, and the authentication process information is also stored on the blockchain.

2. Related technologies

2.1. Blockchain
Blockchain technology was first introduced in Satoshi Nakamoto's paper "bitcoin: a peer-to-peer electronic cash system" [7]. Blockchain is a kind of chain data structure formed by connecting data blocks of chronological order, and ensures that it cannot be tampered with by cryptography mechanism. Its core concepts include three aspects: transaction, block and chain, as shown in Figure 1. Transaction leads to the change of the ledger book's state. The block records the transactions and states results in a period of time, which is a consensus on the current state, and the role of chain is to ensure the integrity of the ledger book, it is to connect the blocks of the sequence of occurrence, which is the log record of the whole state change. In a sense, blockchain is a state machine. Every transaction attempts to change a state. The block generated by consensus is the result of state change caused by all transactions in the blockchain. At the same time, the blockchain runs on a distributed P2P network, each node stores a complete ledger, which can verify the validity of the ledger data. The blockchain consensus mechanism ensures that the updating of the state of the ledger needs the approval for most nodes. Therefore, the block chain has the characteristics that cannot be tampered with.

![Figure 1. Block data structure.](image)

2.2. Smart contract
Smart contract is one of the core technologies of blockchain. This concept was first put forward by Nick Szabo, a well-known computer scientist and cryptologist, in 1994. He defined [8] smart contract as A set of promises, including protocols within which the parties perform on the other promises, the protocols are usually implemented with programs on a computer network, or in other forms of digital electronics, thus these contracts are "smarter" than their paper-based ancestors.
The smart contract endows the blockchain with programmable features [9,10]. After the smart contract is written according to the business logic, it is deployed to the blockchain network node. The external application calls the smart contract, executes the transaction on the chain and accesses the data on the blockchain according to the definition of the contract. Therefore, smart contract builds a bridge for the external application to access the block and status data.

2.3. Identity based encryption
Identity based encryption and signature was proposed by Adi Shamir in 1984 [11]. The scheme includes a trusted key generation center (KGC). Users choose attribute set that can represent themselves as the public key, and the corresponding private key is distributed by KGC.

In this paper, we propose an identity based signature scheme, which uses bilinear mapping to achieve identity authentication. It mainly includes seven random algorithms:

1) System parameter initialization. Let $G_1$ be a cyclic additive group generated by $p$, whose order is a prime $q$, and $G_2$ be a cyclic multiplicative group with the same order $q$. $P$ is the generator of $G_1$. Define a bilinear map $e: G_1 \times G_1 \rightarrow G_2$ and two hash functions $H_1: \{0,1\}^* \rightarrow G_1^*$, $H_2: \{0,1\}^* \rightarrow Z_q^*$. KGC randomly selects $s \in Z_q^*$ as the system master key, and calculates $P_{pub} = s \cdot P$, stores $s$, calculates $g = e(P, P)$, and publishes system parameters:

$$\text{Params} = \{G_1, G_2, e, q, P, P_{pub}, H_1, H_2\}$$

2) Partial private key generation. Input $\text{Params}$ and user's identity information $ID$, and KGC will perform the calculation and return Partial private key $D_{ID} = s \cdot Q_{ID}$, $Q_{ID} = H_1(ID)$. KGC sends $D_{ID}$ to smart devices through a secure channel. The smart device verifies the authenticity by equation $e(D_{ID}, P) = e(Q_{ID}, P_{pub})$.

3) Secret value generation. Input $\text{Params}$ and user's identity information $ID$, and output user's secret value $K_{ID} \in Z_q^*$.

4) User private key generation. Input $\text{Params}$ and user secret value $K_{ID}$ to calculate $S_{ID} = K_{ID} \cdot D_{ID}$.

5) User public key generation. Input $\text{Params}$ and user's secret value $K_{ID}$, and output user's public key $P_{ID} = K_{ID} \cdot P_{pub}$.

6) Signature. Input $\text{Params}$, the user's private key $S_{ID}$ and the message to be signed $M$ output the signature $\sigma$. When the message $M$ is signed, the user chooses randomly $a \in Z_q^*$, calculate $r = g^a$, $V = H_2(M \parallel ID \parallel P_{ID} \parallel r)$, $U = V - S_{ID} + a \cdot P$. Finally, the user's signature to message $M$ is $\sigma = (V, U)$.

7) Verification. Input $\text{Params}$, message $M$, the signature $\sigma$ to be verified, the user's identity information ID and public key $P_{ID}$, and output whether the output signature is valid. During verification, $R = e(U, P) \cdot e(Q_{ID}, -P_{pub})$ is calculated, then we test whether the equation $V \leftarrow H_2(M \parallel ID \parallel P_{ID} \parallel R)$ is true. If it is true, the signature is recognized as valid. If not, the signature is invalid.

3. Scheme design and implementation
The scheme is composed of four entities: Key Management Center (KGC), smart device, verifier and blockchain, as shown in Figure 2.
Key management centre (KGC). Trusted key service provider, mainly provide key generation services and key management services for users.

Smart Devices. The party to be authenticated, the smart device solidifies an identification inside the hardware as its identity ID.

Verifier. The party that authenticates the smart device.

Blockchain. Data on blockchain cannot be tampered with and the characteristics of distributed consensus can provide excellent protection for public information storage and management.

For the convenience of description, four operations are defined: (1) $\text{Sign}(Y, M)$, sign the message $M$ with user’s private key $Y$. (2) $\text{Verify}(Y, S)$, verify the signature $S$ with user’s public key $Y$. (3) $\text{Encryption}(Y, M)$, encrypt message $M$ with user’s public key $Y$. (4) $\text{Decryption}(Y, E)$, decrypt ciphertext $E$ with user’s private key $Y$.

In this paper, smart contract is used to realize the data interaction among KGC, smart device, verifier and blockchain, which mainly includes four interactive operations: system parameter registration, smart device registration, smart device initialization and authentication, as shown in Figure 3.
3.1. System parameter registration

KGC selects a secure bilinear mapping and hash function to generate system master key $s$, then generates system parameter $\text{Params}$, and adds timestamp $t$, then registers it in system parameter directory $D_{\text{sys}}$.

Algorithm 1: System Parameter Registration

| Input:          | $\text{Params}$                  |
|----------------|----------------------------------|
| Output:        | $(\text{Status} < \text{Success}/\text{Failure}>, \text{ID})$ |
| BEGIN          |                                    |
| IF not KGC's Account | return: (Failure, NULL)          |
| END IF         |                                    |
| $t \leftarrow \text{get timestamp}$ |                                 |
| $S \leftarrow Sn(KGC, \text{Params} \parallel t)$ |                                 |
| PID $\leftarrow \text{Generate unique ID}$ |                                 |
| Create record | $\text{Item} = (\text{PID}, \text{Params}, t, S)$ |
| Put Item into $D_{\text{sys}}$ |                                 |
| return:        | $(\text{Success}, \text{PID})$    |
| END            |                                    |

3.2. Smart device registration

Smart device submits its own identification ID and randomly generated $z \in Z_q^*$ to KGC for registration. KGC uses this information to generate partial private key for the smart device, and registers $Q_w = H(ID)$ in the smart device information directory $D_{\text{dev}}$. KGC transmits the partial private key to the user through the secure channel.

Algorithm 2: smart device registration

| Input:          | ID, $z$, PID                      |
|----------------|----------------------------------|
| Output:        | $(\text{Status} < \text{Success}/\text{Failure})$ |
| BEGIN          |                                    |
| IF not KGC’s Account | return: (Failure)             |
| END IF         |                                    |
| Obtain system parameters $\text{Params}$ according to PID |                                    |
| Calculate $Q_{ID} = H_1(ID)$ |                                    |
| Calculate $H_z = H_2(z)$ |                                    |
| $t \leftarrow \text{get timestamp}$ |                                    |
| Create record $R = (ID, Q_{ID}, H_z, PID, t)$ |                                    |
| IF R not found in $D_{\text{dev}}$ |                                    |
| delete R from $D_{\text{dev}}$ |                                    |
| END IF         |                                    |
| Put R into $D_{\text{dev}}$ |                                    |
| return:        | (Success)                         |
| END            |                                    |
3.3. Smart device initialization
Performed by smart devices. Smart Devices select their own secret $K_{ID}$, generate private key $S_{ID}$ and public key $P_{ID}$, then update $(Q_{ID}, P_{ID})$ to $D_{dev}$ on the blockchain.

Algorithm 3: smart device initialization

| Input: | ID, $P_{ID}$, $z$, PID |
| Output: | (Status <Success/Failure>) |
BEGIN
Obtain system parameters $\text{Params}$ according to $PID$
Calculate $Q_{ID}^* = H_1(ID)$
Calculate $H_z^* = H_z(z)$
R $\leftarrow$ search record in $D_{dev}$ matching $Q_{ID}^* = Q_{ID}$ \( \overline{H_z^*} = H_z \)
IF R not found return: (Failure)
END IF
$\tau$ $\leftarrow$ get timestamp
Calculate $B = Sn(U, ID \parallel \tau)$
set $R = (ID, Q_{ID}^*, P_{ID}, PID, \tau, B)$
Update R in $D_{dev}$
return: (Success)
END

3.4. Authentication
Initiated by verifier. The verifier randomly selects $x \in Z_q^*$, adds the timestamp $\tau$ and the ID of smart device $u$ to form a message $M = (x, ID, \tau, Sn(V, ID \parallel \tau \parallel x))$, and initiates an authentication transaction to the blockchain. After discovering the authentication transaction, the smart device verifies the validity of $Vf(V, Sn(V, ID \parallel \tau \parallel x))$, and then immediately executes the signature algorithm to get the signature $\sigma$ and register $(\sigma, ID, \tau^*, Sn(U, ID \parallel \tau^* \parallel x))$ on the blockchain. After receiving the signature information, the verifier verifies the validity of $Vf(U, Sn(U, ID \parallel \tau \parallel x))$, executes the signature verification algorithm to determine whether the signature $\sigma$ is valid, thus verifying the identity of the smart device.

The whole authentication process can be divided into three stages: 1) the verifier submits the authentication request. 2) Smart devices provide evidence of identity. 3) The verifier verifies the validity of the identity evidence. The data interaction in the verification process is realized by smart contract.

3.4.1. Submit validation request. Initiated by the data verifier (the following V represents the verifier). The verifier selects randomly $x \in Z_q^*$ and submits the verification request to the smart contract. The request will be placed in the request list $L_{VR}$.

Algorithm 4.1: Request Identification

| Input: | ID, $x$ |
| Output: | (Status <Success/Failure>) |
BEGIN
t \leftarrow \text{get timestamp}
\text{Calculate } S = Sn(V, \ ID \parallel t \parallel x)
\text{Create record } M = (x, ID, t, S)
\text{Put } M \text{ into } L_{VR}
\text{return: (Success)}

3.4.2. \textit{Create identity evidence}. When the smart device U finds an unprocessed authentication request for itself in the request list \( L_{VR} \), it first verifies the validity of the data. If the verification is passed, it generates the identity evidence \( \sigma \) off the blockchain and encrypts it with the verifier's public key, and then registers the output \( \sigma_{enc} \) in the evidence list \( L_{Evi} \) on the blockchain.

\begin{algorithm}
\textbf{Algorithm 4.2: Create Identity Evidence}
\textbf{Input:} \( \sigma_{enc}, ID, x, PID \)
\textbf{Output:} \( \langle \text{Status <Success/Failure>} \rangle \)
\textbf{BEGIN}
\text{Obtain system parameters } \text{Params} \text{ according to } PID
\text{\( M' \leftarrow \text{search in } L_{VR}, \text{find } M = (x, ID, t, S) \text{ that matching ID, } x \)}
\text{IF } \text{\( M' \text{ not found} \)}
\text{return: (Failure)}
\text{END IF}
\text{Update status of } \text{\( M' \)} \text{ to Modified}
\text{\( t' \leftarrow \text{get timestamp} \)}
\text{Create signature } S' = Sn(U, \ ID \parallel t' \parallel x)
\text{Create identity evidence } E = (x, ID, t, S', \ \sigma_{enc})
\text{Put } E \text{ into } L_{Evi}
\text{return: (Success)}
\text{END}
\end{algorithm}

3.4.3. \textit{Verification of identity evidence}. The verifier takes the identity evidence \( E \) related to random number \( x \) from the evidence list \( L_{Evi} \), executes \( \text{Dec}(V, \ \sigma_{enc}) \) offline, decrypts and executes signature verification algorithm to determine the validity of the identity evidence.

\begin{algorithm}
\textbf{Algorithm 4.3: get Identity Evidence}
\textbf{Input:} \( ID, x \)
\textbf{Output:} \( \langle \text{Status <Success/Failure>}, \text{ Identity Evidence } E \rangle \)
\textbf{BEGIN}
\text{\( E' \leftarrow \text{search in } L_{Evi}, \text{find } E = (x, ID, t, S', \ \sigma_{enc}) \text{ that matching ID and } x \)}
\text{IF } \text{\( E' \text{ not found} \)}
\text{Return: (Failure, NULL)}
\text{END IF}
\text{Return: (Success, } E' \rangle)
\text{END}
\end{algorithm}
4. Experiments and analysis
We use ESP8266EX from Expressif analog smart devices and set up DAPP on Ethereum to test this scheme. ESP8266EX integrates a Tensilica L106 32-bit RISC processor, which achieves extra low power consumption and reaches a maximum clock speed of 160 MHZ. It uses external SPI flash to store user programs, and supports up to 16 MB memory capacity theoretically.

We use the elliptic curve \( y^2 = x^3 + 7 \) defined by the secp256k1 standard, the key length is 256 bits. We repeatedly run 100 times and get the results averaged. Finally, the signature and validation performance are shown in Table 1.

| Table 1. Signature & validation performance. |
|--------------------------------------------|
| **Signature Avg. Time (ms)** | **Verification Avg. Time (ms)** |
| 348                           | 672 |

5. Conclusion
In view of the limited resources and security problems of Internet of things devices, this paper proposes an identity authentication scheme based on identity and blockchain. In the scheme, the identity based signature scheme is used to construct the signature algorithm for identity authentication based on the characteristics of bilinear mapping. Smart contract technology is used to support smart device registration, initialization and authentication. Using the distributed characteristics of blockchain, the public data and verification processes involved in the scheme is stored on the blockchain, which can bring more convenience to the IoT, which is also a distributed structure. In our scheme, after the smart device is registered, it will store the key corresponding to its own device ID, which is a weakness. In the follow-up work, we can consider the mechanism that the smart device obtains the key dynamically.

Reference
[1] Leloglu E 2017 A Review of Security Concerns in Internet of Things Journal of Computational Chemistry 05(1) pp 121-36
[2] Sicari S, Rizzardi A, Greco L A and Coenporisini A 2015 Security, privacy and trust in Internet of Things Computer Networks 76 pp 146-64
[3] Yeh H, Chen T, Liu P, Kim T and Wei H 2011 A Secured Authentication Protocol for Wireless Sensor Networks Using Elliptic Curves Cryptography Sensors 11(5) pp 4767-79
[4] Moosavi S R, Gia T N, Rahmani A, Nigussie E, Virtanen S, Isoaho J and Tenhunen H 2015 SEA: A Secure and Efficient Authentication and Authorization Architecture for IoT-Based Healthcare Using Smart Gateways Procedia Computer Science 52 pp 452-9
[5] Kalra S and Sood S K 2015 Secure authentication scheme for IoT and cloud servers (Amsterdam: Elsevier)
[6] Chen W, Wenying Z, Sai J, Qi L and Anxi W 2018 Identity-Based Fast Authentication Scheme for Smart Mobile Devices in Body Area Networks Wireless Communications & Mobile Computing 2018 pp 1-7
[7] Nakamoto S 2008 Bitcoin: A peer-to-peer electronic cash system
[8] Szabo N 2018 Smart Contracts : Building Blocks for Digital Markets
[9] Christidis K and Devetsikiotis M 2016 Blockchains and Smart Contracts for the Internet of Things IEEE Access pp 2292-303
[10] Sadiku M N O and Eze G 2018 Smart Contracts : A Primer
[11] Shamir A 1985 Identity-Based Cryptosystems and Signature Schemes, Berlin, Heidelberg