Abstract—Currently, the most commonly used scheme for identity authentication on the Internet is based on asymmetric cryptography and the use of a centralized model. The centralized model needs a Certificate Authority (CA) as a trusted third party and a trust chain of CA. However, CA-based PKI is weak in the single point of failure and certificate transparency. Our system, called SS-DPKI, propose a public and decentralized PKI system model. We describe a detailed scheme as well as application to use decentralized PKI based secure communication. Our proposal prevents storage overhead on the data size of transactions and provide reasonable certificate verification time.

I. INTRODUCTION
Since the public keys are in open areas, they are more likely to be exposed to external malicious attempts. Therefore, a kind of reliable infrastructure for public key management should be built and maintained. Public Key Infrastructure (PKI) supports the registration, revocation and verification of public keys used for public key cryptography, and enables linking of identities with public key certificates. PKI is the foundation on which technologies such as digital signatures and encryption can be used for large groups of users. PKIs deliver the elements fundamental for a secure and trusted environment for secure communication.

This paper starts with describing commonly used types of PKIs. Section 3 provides a system model for decentralized PKI. In section 4, we describe the detail design of Self-Signed certificate-based Decentralized Public Key Infrastructure (SS-DPKI). We introduce the TLS handshake protocol with our proposal in section 5. Section 6 analyzes the security and evaluation of the proposal. In section 7, we conclude the paper and provide an outlook on future work.

II. RELATED WORK
Various studies have been conducted to improve the security of the PKI. The existing approach can be divided into two approaches, mainly centralized and decentralized.

A. Centralized PKI
Centralized approaches rely on trusted third parties to ensure security for PKI systems. The CA-based centralized PKI system performs user registration, certificate issuance, revocation, certificate storage and maintenance. A certificate for a public key in X.509 standard [13] include the connection between the public key and the entity and the signature of the certificate issuer. Root CA, Intermediate CA and Issuing CA are connected to trust chain. A user must check the identity in the Registration Authority (RA) in order to issue the certificate. The Certificate Revocation List (CRL) [14] and Online Certificate Status Protocol (OCSP) [15] is used. Recently, there has been a problem that the certificate and the issuing CA are vulnerable to damage and manipulation, and the problem is solved by monitoring and auditing the certificate with Certificate Transparency (CT) [8], [9].

B. Decentralized PKI
The decentralized approaches are based on distributed architectures such as Blockchain and can be authenticated against IDs and public keys without a trusted CA.

Pretty Good Privacy (PGP) [10] has built a distributed authentication system, which provides ease of individual acquisition of the authentication. However, a malicious user can be associated with an imitation key. Furthermore, it is not consistent with PKI standards.

Certcoin [4] is a decentralized PKI scheme for Namecoin [3] based on identity retention. Certcoin is not considering the user identity squatting problem and does not provide a method to connect the same identity to multiple certificates. Also, it does not consider secure communication compatibility.

Authcoin [11] introduced a Challenge-Response protocol-based authentication phase to comply with the drawback of Certcoin. However, it does not consider credibility of whoever performs operations in validation and authentication process.

Cecoin [12] proposed a structure in which a certificate was recorded in Blockchain, a decentralized PKI was constructed to eliminate a single point of failure (SPOF), and an efficient search and verification of certificates. However, it is necessary to manage Certificate Libraries (CL) because certificates are registered to transactions.

C. Blockchain
Blockchain uses cryptographic algorithms (i.e., hash, digital signature), peer-to-peer networks, and distributed consensus protocols to provide tamper-proofing, decentralization, and consistency. The Blockchain is a list of block-hash points linked. The one-way property of hash ensures the integrity of the contents of the block to prevent arbitrary changes. Bitcoin's Proof-of-Work (PoW) [1] and Proof-of-Phone [22] technique allows the miner to compute hash puzzle problems and compete for compensation to validate transactions.

In this paper, we propose a system model and PoW scheme in which a public key is registered in a Blockchain and a block minor verifies the transaction.
III. DECENTRALIZED PKI SYSTEM MODEL

In this section, we briefly introduce system requirements and architecture model of SS-DPKI.

A. Design goals

Our proposal is to resolve problems caused by centralized approaches and set design objectives to be provided by SS-DPKI as follows:

1. It shall provide essential functions of PKI such as registering, updating, revoking and verifying ownership of a public key.
2. It shall remove the trusted third party and be robust against SPOF attack.
3. It shall be compatible with standard secure communication protocol (e.g., SSL/TLS, HTTPS).
4. It should minimize the size of the data stored in the distributed ledger.
5. It should support multiple devices with a single user identity.
6. It should provide a mechanism to prevent identity squatting.

B. System model

The high-level system model for our proposal is illustrated in Fig. 1. Device, Relying Party and Block Miner are nodes in the P2P network.

We describe the role of basic entities in Fig. 1 as follows:

Self-Signed Certificate: A self-signed certificate is an identity certificate that is signed by the same entity whose identity it certifies. The certificate used in our proposal is a self-signed certificate with the device's private key. In CA based PKI, as CA performs trust anchor role. In SS-DPKI, each device acts as trust anchor and information to trust anchor is recorded in the block chain.

Blockchain Wallet (BCW): It is either a device, physical medium, software or a service that generates public and private key pairs, in which secure storage exists for securely protecting the private keys. It also generates digital signatures such as RSA and ECDSA.

Blockchain Network (BCN): BCN is a technology infrastructure that provides a ledger and smart contract service to Devices. Device forwards the Tx for public key and certificate registration to the network, and the Tx verified by a minor is distributed to all BCN's nodes and immutably recorded on their copy of the ledger.

Blockchain: Public key can be registered, stored, discarded, and extended with a Decentralized Identity (DID). Also, the DID and the public key stored in Distributed Ledger (DL) can be retrieved in authentication between Device and Relying Party, and the certificate is verified by the retrieved Tx data.

Device: Device generates the self-signed certificate as the owner of the DID and the public key. They create a proof of identity by recording the transaction such as registration, update, and revocation of the public key in the Blockchain.

Relying Party (RP): RP is a server or other device that wants to securely communicate with a Device. When they perform an authentication protocol for secure communication such as TLS [16] [17], they receive the other party's certificate and verify the DID and the public key of the Device based on the information of the distributed ledger.

Block Miner (BM): BM is the same as that of miners in the Bitcoin. The BM verifies Tx registration request of Device and adds the Tx to DL of Blockchain after verifying the identity, the public key and the certificate consistency and accuracy of the Device.

IV. OUR PROPOSAL

In this section, we describe a detailed scheme for building a decentralized PKI. The Device generates a public and private key pair and creates a self-signed certificate. Information for proof of ownership of a DID and a public key is registered to the Blockchain.

A. Essential functions of DPKI

In order to provide the essential functions, transaction information on registration, updating, and revoking of public keys is posted to the Blockchain. The DID is used as a keyword to search the public key.

As shown in Fig. 2., the transaction is the format of msg | sign, where msg is DID | CMD | PK | HC. The parameters of msg are briefly introduced in the figure. We will describe the functions of DPKI by taking Device A, Device B, and Device C as an example.

1) Registration

The Device creates a Tx for registration that contains a DID, a public key and a hash of the certificate. REG command to inform the registration request of the public key. When a device registers a public key, it is passed along with a certificate for verification of Tx. Once the certificate has been verified, a certificate of Device is not stored online (e.g., DL).
The first transaction in Fig. 2 is the public key registration request, and a transaction of Device A is

\[
DID_A | REG | PK_{A1} | HC_{A1} | \sigma_{A1},
\]

where \( \sigma_{A1} = \text{Sign} \{SK_{A1}, H(m_{A1})\} \)

BM checks whether their DID is registerable. If there is no registration record for DID, BM verifies the signature using the public key of the device that requested registration. The verification of the Tx is completed with verification of the digital signature.

\[
\text{Ver} \{PK_{A1}, H(m_{A1}) | \sigma_{A1}\}
\]

The BM adds the Tx to the end of the DL after it completes the Tx verification.

3) Revoking

Device creates a Tx for revoking the public key (PK_{C1}) registered in the Blockchain. REG command to inform the revocation request of the public key.

In the Fig. 2, the fifth Tx is a public key revocation request, and the Tx of Device is

\[
DID_C | REV | PK_{C1} | HC_{C1} | \sigma_{C1},
\]

where \( \sigma_{C1} = \text{Sign} \{SK_{C1}, H(m_{C1})\} \)

BM verifies the signature using the public key (PK_{C1}) of the Device.

\[
\text{Ver} \{PK_{C1}, H(m_{C1}) | \sigma_{C1}\}
\]

The BM adds Tx to the end of the DL after the verification.

4) Searching

Device or RP belonging to BCN can retrieve the active public key that matches a DID. The operation to find the Tx matching the DID is checked from the information registered recently. If Device or RP finds the Tx, it performs the signature verification process.

B. Multiple Certificate with one identity

A user can create and use unique public/private key pairs for each device in a DID if a Device has secure storage like a Secure Element (SE).

As shown in Fig. 3, when a User A has a mobile phone, a laptop and a tablet, if an authentication service using a DID can be provided through only one device, the user becomes inconvenient. In order to solve this problem, a new public key is extended and registered based on the primary public key that is initially registered with respect to the DID.

![Blockchain Structure for SS-DPKI](image)

Fig. 2. Blockchain Structure for SS-DPKI

2) Updating

The Device creates a Tx for updating the public key (PK_{A1}) registered in the Blockchain with the new public key (PK_{A2}). The UPD command is entered to inform the update of the public key associated with DID (DID_{A1}). When the device updates a public key, it is passed along with a certificate for verification of Tx and certificates.

In Fig. 2, the fourth Tx is the public key update request, and Tx of the Device is

\[
DID_A | UPD | \{PK_{A1}, PK_{A2}\} | HC_{A2} | \sigma_{A2},
\]

where \( \sigma_{A2} = \text{Sign} \{SK_{A2}, H(m_{A2})\} \)

\[
\sigma_1 = \text{Sign} \{SK_{A1}, H(m_{A1}) | HC_{A1}\},
\]

\[
\sigma_2 = \text{Sign} \{SK_{A2}, H(m_{A2})\}
\]

BM verifies the signature using the old public key (PK_{A1}) of Device. After the signature verification is completed, the verification is performed with the new public key (PK_{A2}).

\[
\text{Ver} \{PK_{A1}, H(m_{A1}) | HC_{A1} | \sigma_{A1}\},
\]

\[
\text{Ver} \{PK_{A2}, H(m_{A2}) | \sigma_{A2}\}
\]

The BM adds Tx to the end of the DL after the verification.

![Assigning multiple certificates to one ID](image)

Fig. 3. Assigning multiple certificates to one ID
User A is capable of DID-based authentication in Device A1. At this time, if Device A2 wants to be authenticated using the same DID, the device connected to DID A can be extended to Device A2. To extend the public key associated with the DID, Device must configure a transaction as follows:

\[ \text{DID}_A \mid \text{EXT} \mid \{ \text{PK}_{A1}, \text{PK}_{A2} \} \mid \text{HC}_{A2} \mid \sigma_{A2} \]

Where \( \sigma_{A2} = \sigma_1 \mid \sigma_2 \),
\[ \sigma_1 = \text{Sign}(\text{SK}_{A1}, H(\text{m}_{A1} \mid \text{HC}_{A2})) \]
\[ \sigma_2 = \text{Sign}(\text{SK}_{A2}, H(m_{A2})) \]

Ownership of the primary public key (PK_{A1}) is authenticated by verifying the digital signature on the Tx. Since creation of the valid signature for the Tx can only be generated by the Device A1 that owns the primary private key (SK_{A1}). The authentication of the owner of the new public key (PK_{A2}) is verified by the digital signature calculated by the new private key (SK_{A2}). It can prevent the use of DID on malicious devices because only the devices (Device A1/A2) registered in the BCN are authorized the use the DID_A.

BM verifies whether a public key included in the certificate for which a public key extension is desired in the DID_A is the same as the public key contained in the Tx, verifies whether the signature value for the entire certificate is the same as the hash value included in the Tx. After completing certificate verification, BM performs signature verification by a primary public key included in Tx and performs digital signature verification on Tx of the command for the extended public key. After the Tx verification is completed, Device A2 stores the certificate and completes the registration process.

C. Certificate Verification

The procedure for verifying the certificate in the secure communication process is shown in Fig. 4.

When a Device receives a DPKI-based certificate during encrypted communication such as SSL/TLS, it looks up the same Tx as DID of the certificate in the DL of Blockchain. It verifies whether the certificate of the partner device to be communicated is the correct certificate registered in the Blockchain.

Verification procedure is as follows:
1. Check that a DID included in the certificate is same as a DID value of a Tx. Verify that a DID of the certificate is the same as a DID contained in the issuer X.500 Name and subject X.500 Name. The corresponding DID must be the same as the DID of the Tx.
2. Compare the public key contained in the certificate with the key contained in the TX.
3. Verify the CA Digital Signature with a public key value on the hash corresponding to the signature algorithm ID and the signature algorithm for the tbsCertificate [14] excluding the CA digital signature in the certificate.
4. Calculate the SHA-256 hash value for the whole X.509 Certificate and check whether it is equal to HC (Hash of Certificate) value of transaction.

V. TLS HANDSHAKE WITH SS-DPKI

Our proposal can be used for secure communication, such as SSL/TLS, just like a centralized PKI. It is defined as a structure compatible with TLS 1.2 Handshake protocol [16] as follows and is designed to be compatible with other secure communication protocols such as TLS 1.3 [17]. The TLS 1.2 protocol with SS-DPKI, which is currently the most widely used, is described as follows.

1. Client Hello

Device A (as a client) sends ClientHello message, including client_version, random, session Id, cipher_suites, and
compression_methods, to Device B (as a server).

2. Server Hello
Device B sends ServerHello including the server_version, random, session_id and cipher_suite to Device A.

3. Server Certificate
Device B sends ServerCertificate to the client. SS-DPKI carries a self-signed server certificate.

When Device A receives Device B's certificate, it finds Tx for a public key corresponding to DID of Device B in the DL. The process of verifying the certificate based on the retrieved Tx is described in section 4.C.

4. Server Key Exchange
Device B sends ServerKeyExchange including parameters for key exchange to Device A.

5. Certificate Request
If Device B needs to authenticate Device A, it sends CertificateRequest to request Device B's certificate.

6. ServerHello Done
Device B sends a ServerHelloDone to Device A to indicate that Server Hello command has been terminated.

7. Client Certificate
Device A sends ClientCertificate to Device B. Self-signed Device A’s certificate is delivered in SS-DPKI.

8. Client Key Exchange
Device A sends ClientKeyExchange including parameters for key exchange to Device B.

9. Certificate Verify
Device A sends a CertificateVerify to Device B to provide explicit verification of the client certificate.

10. Finished
Device A sends Finish (client-side) to Device B to confirm that the handshake protocol has been terminated normally.

11. Finished
Device B sends Finish (server-side) to Device A to confirm that the handshake protocol has been terminated normally.

VI. SECURITY ANALYSIS AND EVALUATION

The most important thing to consider when configuring a PKI is security for the system. The system needs guaranteed and secure management of public keys. In a decentralized PKI model using Blockchain technology, the size of a data stored in the DL should be minimized. In addition, compatibility with well-known standard protocols such as TLS and HTTPS should be considered for use in secure communications.

<table>
<thead>
<tr>
<th>Security Features</th>
<th>CA</th>
<th>CT</th>
<th>POP</th>
<th>Certcoin</th>
<th>Authcoin</th>
<th>Cecoin</th>
<th>SS-DPKI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single point of failure</td>
<td>X</td>
<td>X</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>Certificate Transparency</td>
<td>X</td>
<td>X</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>ID Squatting</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>Registration</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>Revocation</td>
<td>O</td>
<td>O</td>
<td>X</td>
<td>O</td>
<td>X</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>Verification</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>Multi-certificate</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>O</td>
</tr>
<tr>
<td>Prevent Storage Overhead</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>O</td>
</tr>
<tr>
<td>Compatible with SSL/TLS</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>O</td>
</tr>
</tbody>
</table>

A. Security Analysis

As compared to Table 1, SS-DPKI is resistant to security vulnerabilities against SPOF attacks and CT problems of CA-based PKI and has stronger compatibility with security protocols than other blockchain-based PKI models.

The centralized approach is vulnerable to SPOF attacks and the decentralized PKI does not affect the entire system if one device fails.

CA-based PKI does not provide real-time monitoring and status validation of certificates. CT [8] and Blockchain-based PKI models are free from this problem.

For ID squatting, the decentralized PKI model may be rather vulnerable. As with CA-based PKI, there is no certainty as to the process of visiting the RA to identify it offline.

SS-DPKI and Cecoin provide a mechanism for updating a public key or a certificate.

Users of Device will be able to authenticate more securely if they use different public keys for each device using the same DID. SS-DPKI provides the scheme described in section 4.B.

In terms of storage efficiency for storing public keys and certificates, CA-based PKI keeps certificates on individual devices. Of the decentralized schemes, SS-DPKI can also save on-line storage by storing certificates in individual devices while reducing the data size registered in Tx.

SS-DPKI is highly utilized because it is designed to be compatible with SSL/TLS protocol like CA-based PKI.

B. Evaluation

We plotted the experimental results on the transaction data size and the certificate validation time in Fig. 6 and Fig. 7.

1) Data size in transactions

In this section, we have measured a data size of transactions for registering a public key or certificate as compared with Cecoin.

For example, considering the registration process for the ECDSA 256 bits’ public key, the data size of the transaction is 171 bytes for SS-DPKI and 636 bytes for Cecoin.

- Cecoin (636 bytes per Tx): Value (4) | Cert (568) | PK (64)
- SS-DPKI (171 bytes per Tx): DID (40) | REG (3) | PKAI (64) | HC_{AI} (32) | σ_{AI} (32)

Data Size in Transactions

![Fig. 6. Data Size in Transactions](image-url)
2) Certificate verification

We implement the certificate verification simulator using a PC with the Intel Core i7-7700 CPU @ 3.6GHz and 16GB RAM. We use the OpenSSL library [6] (version 1.1.0) in order to implement certificate verification with the distributed ledger.

We also refer to the results of the certificate verification time in the CA-based PKI and the certificate verification time in the CA-based PKI with OCSP [18].

![Certificate Verification Time](image)

Fig. 7. Certificate Verification Time

CA-based PKI performs the best with 4.59 msec, but adding the OCSP online check overhead increases the verification time to 269.25 msec [18].

In CA-based PKI, certificate verification can be done quickly because only trust chain verification of certificate is performed. The software is also optimized for certificate validation.

In SS-DPKI, the size of the DL has a large effect on the time required for the certificate verification. The certificate validation time for a DL with 100 Tx is 16.69 msec, and for 10,000 Tx, the average validation time is 42.39 msec.

By verifying the status of the certificate in real-time, it is possible to validate the certificate securely, so it was possible to verify the certificate about 6 times faster than the result of CA with the OCSP.

VII. CONCLUSION AND FUTURE WORK

We propose a decentralized public key infrastructure based on Blockchain. It improves the problems of centralized PKI and provides flexibility for public key management. A user can use a public key of several devices in a single DID, and receive encrypted communication services such as SSL/TLS and HTTPS.

Our proposal is that the data size stored in the spreadsheet is 3.7 times smaller than the existing scheme (i.e., Cecoin) and can save 930k bytes based on 20k transactions. It can also provide certificate verification efficiency up to 21.2 times to 9.4 times faster than existing CA-based PKI that works with the OCSP server.

We plan to work on the construction of a decentralized PKI system that meets the requirements of self-sovereign identity [19], [20], and identity management system on the Blockchain [21].

REFERENCES