上海交通大学
SHANGHAI JIAO TONG UNIVERSITY

A Survey of Distributed PKI

Log Creative

May 19th, 2024

Contents

1 Main challenges

Public Key Infrastructure (PKI) is "a set of policies, processes, software, and workstations used for the purpose of administering certificates and public-private key pairs, including the ability to issue, maintain, and revoke public key certificates[1]."

1.1 Traditional PKI (PKIX)

The deployment of PKI on the Internet has been dominated by the ITU-T X.509 standard [2], known as PKIX. A traditional PKIX is composed of five main components: Certificate Authority (CA), Server, Registration Authority (RA), Client, and Repository. However, PKI is not a silver bullet for security and there are several challenges [3].

CAs are the corruptible central points of failure. This is the major challenge of traditional PKIs. There have been CA failures across those years [4], like key leakage, email spoofings, null prefix attacks, internal system compromise, cryptographic flaws, etc. Once the CA is down, then the authentication will not be performed. Virtually all Internet software now relies on these centralized authorities.

Cyber-attacks. The traditional PKIX architecture may suffer from various cyber attacks [4]: CA Compromises; Compelled Certificate Attack; BGP Route Hijacking; Dangling DNS Record Attack, Deprovisioned cloud instance attack, Expired domain attack, Discontinued service attack; DNS Record Spoofing Attack; Split-World Attack; DNS Cache Poisoning Attack; Vantage Point Downgrade Attack; Truststore Attack; Malicious CA Insertion Attack; Revocation Information Blocking Attack; Zombie Certificate Attack;Efail Attack.

Centralized PKI lacks scalability and flexibility. Earlier traditional public key infrastructure was a challenge to be deployed [5]. The centralized system is very vulnerable facing various attacks in the context of scalability and flexibility. Some companies spend significant resources fighting security breaches caused by misbehaving CAs mentioned earlier [6].

CA Authority is an issue. There are questions about the authority of CAs to grant specific authorizations in the certificates they issue. And the CA itself may not be trustworthy as well.

PKI users may not ensure the private key. The PKI user in all likelihood do not claim a safe figuring framework with physical access controls, protecting, air divider, organize security, and different insurances; client stores the private key on an ordinary PC which is of great possibility a hot environment compared to a physically isolated code environment. Therefore there is a risk that the private key may be leaked by accident.

1.2 DPKI

Distributed Public Key Infrastructure (or Decentralized Public Key Infrastructure, DPKI) makes use of use blockchains and other forms of distributed consensus to provide the same functionality of centralized systems like X.509 while also providing improved scalability and removing **singe points of failure** for a more secure system [7]. With the advent of DPKI, **scalability and flexibility** issues have been addressed. However, some old implementations of DPKI could still have flaws and face several challenges [8]:

SPKI Simple Public Key Infrastructure (SPKI) [9] to bind privileges or rights (access control) to the public key. However, this approach is **never finalized** and later merged with the simple distributed security infrastructure (SDSI) [10] that deals with groups and group-membership certificates. SDSI is the first step towards a decentralized public key infrastructure, allowing entities to make local decisions about access control based on their local policies and knowledge, rather than relying on a central authority for all decisions. However, there could still be **a problem of trust** (country in this case).

Log-based PKI. The log-based PKI provides transparency and accountability by creating publicly available logs for the generated certificates. This approach greatly suffers from **deployment challenges** and **certificate revocation issues**. The PKISN (PKI Safety Net) [11] also follows the same footsteps as in log-based PKI, but proposes an approach to solve the certificate revocation problem when a trusted CA's private key is compromised. However, this approach as well suffers from the same **deployment issues**.

2 SOTA of DPKI

The current state-of-the-art (SOTA) of DPKI are smart contract DPKI, QChain, and DKS-PKI.

2.1 Smart Contract DPKI

Description. The first provably secure DPKI based on smart contracts [12] was proposed in 2017, but it is RSA-based and impractical to implement for large-scale projects. A Merkle-tree based smart contract DPKI [13] was proposed later to reduce the computational cost. This allows for the size of the smart contract state, which is the most expensive resource to access on the blockchain, to be minimized as well as make sure its security.

The core idea is **decoupling the storage** of (identity,public-key) pairs **from the verification** of their validity. The storage of information relevant to the protocol, e.g., (identity,public-keys) pairs, is offloaded to an external database component, which is modeled as unreliable database (*UDB*). Based on Hash-tree accumulator, clients can perform, locally, additions and deletions of elements and supply the smart contract with appropriate witnesses which prove that the operations were performed honestly. To generate all involved witnesses, clients query *UDB* for the history of operations. The accumulator's underlying data structure (Merkle tree) is a balanced, binary tree, with the accumulator's value represented as the hash of the root node and witnesses are hash paths starting from a node leading up to the root node. This leads to constant size accumulator values, but witnesses have $O(\lambda \log(n))$ bit size where n is the number of accumulated elements and λ is the security parameter. Figure 1 shows the Constructor, Register, Revoke, and RetrieveState operations.

Smart Contract State: $c_1, c_2, \lambda_1, \lambda_2 \in \mathbb{Z}$

1) $\mathsf{Constructor}(\lambda_1, \lambda_2)$:
   Store input values λ_1, λ_2 to the corresponding state variables
   $c_1 \leftarrow \mathsf{InitAcc}(\lambda_1)$
   $c_2 \leftarrow \mathsf{InitAcc}(\lambda_2)$
2) $\mathsf{Register}(id, pk, W_2, c_{add_1}, W_{add_1}, c_{add_2}, W_{add_2})$:
   if $\mathsf{sizeof}(id) \neq \lambda_2 \vee \mathsf{CheckUpdate}(c_2, c_{add_2}, W_{add_2}, id) = 0 \vee \mathsf{sizeof}(id, pk) \neq \lambda_1 \vee \mathsf{CheckUpdate}(c_1, c_{add_1}, W_{add_1}, (id, pk)) = 0 \vee \mathsf{VerifyNonMem}(c_2, W_2, id) = 0$
      return fail
   endif
   $c_1 \leftarrow c_{add_1}$
   $c_2 \leftarrow c_{add_2}$
3) $\mathsf{Revoke}(id, pk, W_1, \sigma_{sk}(pk), c_{del_1}, W_{del_1}, c_{del_2}, W_{del_2})$:
   if $\mathsf{sizeof}(id) \neq \lambda_2 \vee \mathsf{sizeof}(id, pk) \neq \lambda_1 \vee \mathsf{VerifyMem}(c_1, W_1, (id, pk)) = 0 \vee \mathsf{VerifySig}(\sigma_{sk}(pk), pk) = 0 \vee \mathsf{CheckUpdate}(c_1, c_{del_1}, W_{del_1}, (id, pk)) = 0 \vee \mathsf{CheckUpdate}(c_2, c_{del_2}, W_{del_2}, id) = 0$
      return fail
   endif
   $c_1 \leftarrow c_{del_1}$
   $c_2 \leftarrow c_{del_2}$
4) $\mathsf{RetrieveState}()$:
   return $(c_1, c_2, \lambda_1, \lambda_2)$

Figure 1 Pseudocode of the smart contract DPKI in the hash-based construction

Advantage. It is found that this implementation could easily and securely be used on Ethereum's mainchain, particularly if developers of the project were willing to make changes that would allow for a DPKI to be faster and more efficient.

Disadvantage. However, one challenge that remains is witnesses being dependent on the number of id-key pairs registered ($O(\lambda \log(n))$ mentioned earlier), which can become verbose over time unless steps are taken to "reset" the accumulators from time-to-time.

2.2 QChain

Description. QChain (Quantum-resistant Blockchain-based DPKI) [14] is its own blockchain system where each block stores the id-pk pairs and uses their hashes for the Merkle hash tree structure, while it is secure against quantum computing attacks by modified GLP signature with NTT (Number Theoretic Transformation) [15] operations.

Figure 2 illustrates a typical use case of QChain to setup secure communication. The first operator initiates the genesis block as well as setting up the parameters for the overall chain. Then, QChain creates another block and users run QChain.User.Setup() and QChain.User.Add() algorithms in order to register a public key. These keys can be verified using QChain.User.Verify() which allows users to determine if a key they are provided is authenticated or not.

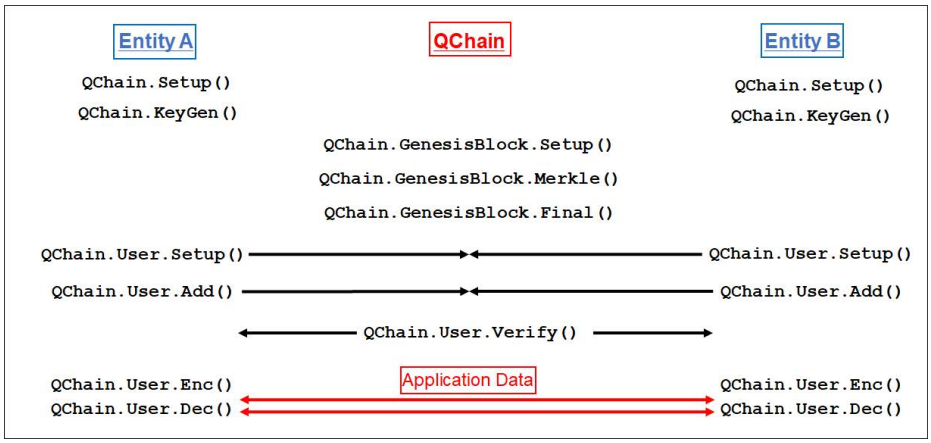

Figure 2 A typical use case of QChain to setup secure communication.

Advantage. It can keep **offline states** (except for the initiation of the genesis block) while X.509 v3 must keep online states (particularly by a trusted third party like CA) in order for users to verify public keys. The construction uses extended X.509 certificate, which makes it **easy to integrate** with current X.509 standards. It does **prevent single points of failure**, which is an inherent problem with X.509.

On the other hand, it is **quantum-resistant** by using NTT operations.

Disadvantage. There is no security proof for the protocol provided by the authors and therefore there is a possibility for non-quantum based adversaries to trick the system into accepting malicious registrations or for an adversary to deceive users by using a non-current association of a public key [7]. It does not seem clear how the immutable property of QChain allows for users to revoke associations.

2.3 DKS-PKI

Description. DKS-PKI (Distributed Key Server for PKI) [16] utilizes a consensus-based distributed network of authority nodes, permissioned storage nodes, and publicly accessible key-server nodes to ensure that the certificate registration/issuance, storage, distribution, and revocation mechanisms are secure, transparent, and fault-tolerant. All the certificates (registered public keys) in DKS-PKI are accessible using their associated unique identifiers (UIDs) through the key-server nodes.

In the architecture shown in Figure 3, the authority nodes (ANs), the transaction repository nodes (TxRNs), and the key-server nodes (KSNs) communicate to each other through the broadcasting nodes (BNs) only. However, the same type of nodes may have direct communication channels (e.g., AN-AN, TxRN-TxRN, BN-BN) between them for data synchronization.

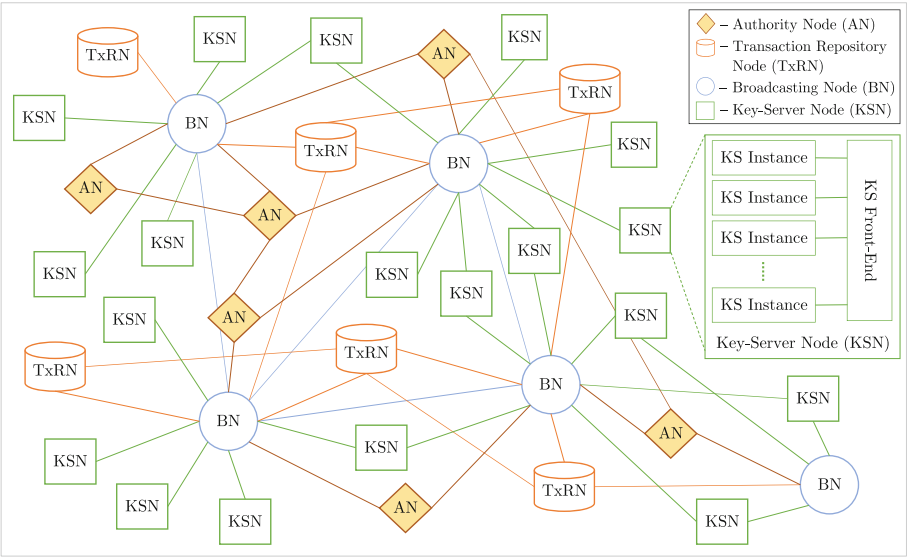

Figure 3 Architecture of DKS-PKI

When the BNs receive an incoming message from a subscribed node, they publish this message to the other subscribed nodes only. Each AN validates certificate registration and revocation requests submitted to it. For each registration request, it generates a public key certificate and its associated 256-bit unique identifier (UID). Then, the AN generates a temporary transaction enclosing these information.

After validating registration/revocation requests, the ANs broadcast temporary transactions enclosing entity information and their signed public key certificates to the other ANs and the TxRNs. After successful consensus between the ANs, the TxRNs generate the sealed transactions from these temporary transactions.

Then, the TxRNs store these sensitive entity information, issued or revoked certificates, and their associated UIDs permanently. The TxRNs also transmit the issued certificates and their associated UIDs to the KSNs. These KSNs need to be publicly available at all times and are responsible to manage certificate requests from anywhere in the world.

Advantage. There are two major challenges for CA-based PKIs: (a) increased intermediate CAs gives rise to the possibility that the mis-issuance of certificates or the disclosure of any such trusted CAs' private keys can cause serious problems to the security of internet communications; (b) these CAs have ample powers to maliciously or erroneously issue duplicate certificates or revoke the existing certificates without the owner's consent.

This architecture solves the two major problems with the existing CA-based PKI: (a) This proposed architecture ensures transparency and accountability of the certificate issuers (authority nodes). Any authority node (AN) cannot issue certificates in the network without reaching consensus with the other ANs. This approach **solves the certificate mis-issuance problem**. (b) Next, certificate revocation is effective as soon as the authority nodes reach a consensus on the revocation transaction. After that, the revoked certificates are separated from the rest of the issued certificates in the TxRNs and the KSNs. This **solves the certificate revocation issue** as well.

The **performance** of the certificate distribution mechanism shows reasonable results with the increasing number of certificate requests.

The distributed architecture **prevents DoS/DDoS attacks** on any node and improves the availability of the PKI network for authentication.

Disadvantage. It is relatively different from the PKIX architecture, which may introduce the difficulty of making it widely adopted.

References

[1] P. A. Grassi, M. E. Garcia, and J. L. Fenton, "Digital identity guidelines: Revision 3," National Institute of Standards and Technology, Gaithersburg, MD, Tech. Rep., Mar. 2020. DOI: 10.6028/nist.sp.800-63-3.

[2] D. Solo, R. Housley, W. Ford, and T. Polk, "Internet X. 509 public key infrastructure certificate and crl profile," *Internet Engineering Task Force, Networking Group*, 1999.

[3] C. Ellison and B. Schneier, "Ten risks of pki: What you're not being told about public key infrastructure," *Comput Secur J*, vol. 16, no. 1, pp. 1–7, 2000.

[4] S. Khan, F. Luo, Z. Zhang, *et al.*, "A survey on x.509 public-key infrastructure, certificate revocation, and their modern implementation on blockchain and ledger technologies," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 4, pp. 2529–2568, 2023. DOI: 10.1109/COMST.2023.3323640.

[5] M. Gupta, S. Tanwar, T. K. Bhatia, S. Badotra, and Y.-C. Hu, "A comparative study on blockchain-based distributed public key infrastructure for IoT applications," *Multimedia Tools and Applications*, vol. 83, no. 12, pp. 35 471–35 496, Sep. 2023, ISSN: 1573-7721. DOI: 10.1007/s11042-023-16970-x.

[6] weboftrust.info, *Decentralized public key infrastructure*, 2015. [Online]. Available: https://www.weboftrust.info/papers/#pki.

[7] N. Kartha, *An overview of distributed public key infrastructures (DPKIs)*, 2022. [Online]. Available: https://www.cs.utexas.edu/~dwu4/courses/sp22/static/projects/Kartha.pdf.

[8] A.-T. Dumitrescu and J. Pouwelse, *Failures of public key infrastructure: 53 year survey*, 2024. arXiv: 2401.05239 [cs.DC].

[9] C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen, *SPKI Certificate Theory*. Sep. 1999. DOI: 10.17487/rfc2693.

[10] R. Rivest and B. Lampson, *SDSI - a simple distributed security infrastructure*, 1996. [Online]. Available: https://people.csail.mit.edu/rivest/sdsi10.html.

[11] P. Szalachowski, L. Chuat, and A. Perrig, "PKI Safety Net (PKISN): Addressing the too-big-to-be-revoked problem of the TLS ecosystem," in *2016 IEEE European Symposium on Security and Privacy (EuroS P)*, IEEE, Mar. 2016. DOI: 10.1109/eurosp.2016.38.

[12] C. Patsonakis, K. Samari, M. Roussopoulos, and A. Kiayias, "Towards a smart contract-based, decentralized, public-key infrastructure," in *Cryptology and Network Security: 16th International Conference, CANS 2017, Hong Kong, China, November 30—December 2, 2017, Revised Selected Papers 16*, Springer, 2018, pp. 299–321.

[13] C. Patsonakis, K. Samari, A. Kiayiasy, and M. Roussopoulos, "On the practicality of a smart contract PKI," in *2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON)*, IEEE, Apr. 2019. DOI: 10.1109/dappcon.2019.00022.

[14] H. An and K. Kim, "QChain: Quantum-resistant and decentralized PKI using blockchain," in *2018 Symposium on Cryptography and Information Security (SCIS 2018)*, IEICE Technical Committee on Information Security, 2018.

[15] T. Güneysu, T. Oder, T. Pöppelmann, and P. Schwabe, "Software speed records for lattice-based signatures," in *Post-Quantum Cryptography*, P. Gaborit, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 67–82, ISBN: 978-3-642-38616-9.

[16] A. Faisal and M. Zulkernine, "Dks-pki: A distributed key server architecture for public key infrastructure," in *Lecture Notes in Computer Science*. Springer Nature Switzerland, 2022, pp. 23–43, ISBN: 9783031236907. DOI: 10.1007/978-3-031-23690-7_2.