Authledger: a novel Blockchain-based domain name authentication scheme

Zhi Guan, Abba Garba, Anran Li, Zhong Chen, Nesrine Kaaniche

AuthLedger: A Novel Blockchain-based Domain Name Authentication Scheme

Zhi Guan[1,3][a], Abba Garba[2,3][b], Anran Li[2,3][c], Zhong Chen [2,3][d], Nesrine Kaaniche [4][e]
[1]National Engineering Research Center for Software Engineering, Peking University, Beijing, China
[2]Institute of Software, EECS, Peking University, Beijing, China
[3]MoE Key Lab of High Confidence Software Technologies, Peking University, Beijing, China
[4]SAMOVAR, Telecom SudParis, CNRS, University of Paris-Saclay, France.
{guan, abbaggumel, lianran, zhongchen}@pku.edu.cn, nesrine.kaaniche@telecom-sudparis.eu

Keywords:
PKI, Blockchain, Authentication, Cryptography, Thin Nodes

Abstract:
Public Key Infrastructure (PKI) authentication mainly relies on Certificate Authorities (CAs) and have to be trusted by both domain operators and domain owners. In order to fairly distribute trust among entities involves in the certificate issuing process, it is necessary to balance the distribution rights among the entities and improve the control of certificate issuance for the certificate owners. Recently with the emergence of Blockchain, a public verifiable distributed ledger, several applications appeared taking advantage of this powerful technology. The unique features of Blockchain technology has capability to ensure trust, remove single point of failure attached to the current PKI systems and quick response to CAs misbehaviour. In this paper, we designed and implemented robust and scalable domain authentication based on blockchain technology for low constrained devices (thin client nodes). Our approach aim to achieve same level of security properties and authentication mechanism as DANE and CAA for low constrain devices but in a more trusted and decentralised structure. The proposed scheme is multi-fold. First, we proposed a domain authentication scheme to reduce the level of trust in CAs over certificate issuance process for using thin client nodes. Second, we implement our system using Ethereum smart contract. Third, we evaluate security and performance of the proposed system.

1 INTRODUCTION

Nowadays, Internet users are mainly relying on Public Key Infrastructures to authenticate end-user certificates and Domain Name Service (DNS) to translate human readable in to IP address to communicate on the web services. PKI enable secure mean of authenticating entities on the Internet. It includes policies and procedure required to issue, manage, distribute, validate, revoke digital certificate and manage public key encryption (Yakubov et al., 2018). The current approaches use to authenticate entities relies on third parties CAs to issue certificates that we usually trust. There is a disparity of rights between the different involved entities such as end-users and Certificate Authorities (CAs) (Scheitle et al., 2018). Indeed, users may initiate a certificate signing request, but are not allowed to judge whether the issuing authority has the right to issue certificate for the domain. (Matsumoto et al., 2017).

Several breaches of trust and certificates misissuance have been revealed recently (Kamat and Gautam, 2018). In September 2015, Google denounced Symantec's Extended Validation certificates due to Symantec issued a certificates without authorization for google domains (Rashid, 2015). Meanwhile, in 2015 Lenovo Superfish installed a local CA for its products to steal con-

[a] https://orcid.org/0000-1111-2222-3333
[b] https://orcid.org/1111-2222-3333-4444
[c] https://orcid.org/2222-3333-4444-5555
[d] https://orcid.org/3333-4444-5555-6666
[e] https://orcid.org/4444-5555-6666-7777

fidential data (Kamat and Gautam, 2018). Gervase in 2016 revealed an issue of certificate compromise associated with wosigns free certificate service on Mozilla.(Enisa, 2016).

These lethal phenomena results to disseminate the trust of the PKI to various authorities (CAs) (Khan et al., 2018). Nowadays, certificate mississuance is hard to detect due to lack of standard mechanisms to check which certificate authorities have the right to issue certificates for the domains (Kubilay et al., 2018).

Recently Domain Name System Security Extensions (DNSSEC) as a DNS resources records, uses a public key infrastructure (PKI) to offer additional security for authenticating data for domain name system (DNS). DNSSEC records allowed client via signed statements to specify which CAs are authorized to represent certificate for the domain (Gourley and Tewari, 2018). Certificate authority authorization (CAA) is another approach developed by Internet Engineering Task Force (IETF) as explained in [RFC 6844][1] to provides security guarantee against rogue CAs. In CAA, domain owners decides which CAs can issue a certificate for their domains via CAA resource records (Karaarslan and Adiguzel, 2018). Consequently, latter approaches are more prone to central point of failure due to their trust inherently attached to infrastructure like Internet Corporation for Assigned Names and Numbers (ICANN) [2](Berkowsky and Hayajneh, 2017).

Blockchain is a decentralised global ledger that contain a series of transactions in the form of blocks (Ali et al., 2016). Each block is secured by hash function to link to another Block in an orderly structured to form a Blockchain network (Yakubov et al., 2018). one of the key constrains of the current blockchain is scalability running full nodes in low constrain devices is not scalable. Ethereum is the second largest blockchain system in terms of value. The objective of such system is to store an arbitrary state in a distributed temper-proof manner (Matsumoto and Reischuk, 2016). Unlike Bitcoin, Ethereum used Turing-complete language and *EthereumVirtualMachine*(*EVM*) to represent language and computations.

In this paper, we extend a conference paper based on AuthLedger: A Novel Blockchain-based Domain Name Authentication Scheme. Aforementioned paper was proposed based on full nodes or global verification state. We design and implemented Blockchain based domain authentication that allow thin client to efficiently verify data in the block.

We present AuthLedger scheme extension using thin client and make several optimisations to make it more efficient and validate low resource devices such mobile. Since running full nodes from browser to fetch queries is computationally costly. Our optimisations use tools such as cryptographic accumulators, sparse merkle hash tree and bloom filters to allow efficient domain authentication for low resources devices.

The primary goal is limit the attack surface of the malicious CAs that issue certificate and any domain and equitably distribute the trust among the entities during certificate issuance process using blockchain while allow low resource devices to authenticate data efficiently.

Thus, our contributions are summarised as follows:

- We extend authledger paper using thin client method based on the blockchain technology for identity authentication without a trusted third party.
- Provides an efficient and trustworthy certificate authentication process for low resources devices.
- We implement and conduct an experimental performances' analysis to validate the proposed system using Ethereum solidity smart contract environment.
- Analyse security implication of the proposed scheme by discussing different security threats and countermeasures.
- We evaluate security and performance of the proposed system including Browser plug-in extensions.

2 BACKGROUND AND RELATED STUDY

In this section, we first give a detailed technical background on certificate authorities (CAs) and its relevant issues in subsection 2.1. Then, we look at the Domain name system (DNS) including Domain name system security extensions (DNSSEC) and Domain name system based authentication (DANE) in subsection 2.2 Also in 2.3 we describe about Certificate authority and authorization (CAA). We finally present Blockchain PKI systems solutions in subsection 2.4.

[1]https://tools.ietf.org/html/rfc6844

[2]https://www.icann.org/

2.1 Certificate Authorities (CAs)

PKI involve several hardware and software entities to manage the digital certificates (Aishwarya et al., 2015). Certificate based authentication mainly use X.509 PKI standard to provide a strong level of clients authentication (Yakubov et al., 2018). Authentication occurs using a third party certification authorities (CA) (Kiayias et al., 2017). In the following, we present the entities that are involved in PKI system: A client provides information to a CA that verifies the user identity. Registration authority (RA) manages CA task such as authenticating users. Validation authority (VA) verifies and confirms whether digital certificate is used by adequate trustworthy CA.

Different entities involved in the PKI as shown in Figure 6:

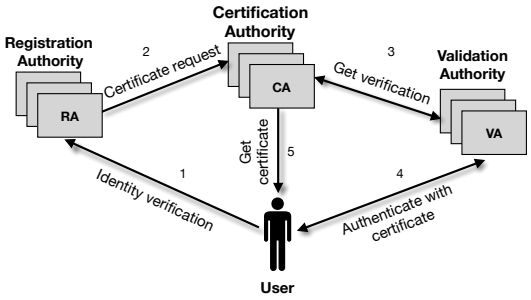

Figure 1: Entities involves in PKI

2.2 DNS Authentications

DNS is one of the most important internet protocol designed to connect web sites over the internet. A distributed database that provides a mapping service called Resource Records (RR). RR translates a domain name into IP address and an IP address to human readable domain names (Zhu et al., 2015). DNS is used as a distributed and hierarchical domain database. The hierarchical name space feature is very essential because it permits a relaying party to verify the identity of the requesting entity. However, DNS based on its original implementation does support detail security to protect attacks from occurring. This includes DNS spoofing/cache poisoning (Wei-hong et al., 2017), DNS hijacking and DNS rebinding (Lencse and Kadobayashi, 2018).

DNS security extensions (DNSSEC): DNSSEC was implemented to add up layer of trust on top of the DNS to ensure smooth authentication of information, on the other hand, maintain backward compatibility (Gourley and Tewari, 2018). DNSSEC using DNS based authentication of named entities (DANE) was designed based on Internet Engineering Task Force (IETF) [RFC6394-RFC6698] standard to protect cache poisoning attacks (Shulman and Waidner, 2017). The main purpose of DNSSEC is to address some security challenges of the existing PKI certificates authentication (Gourley and Tewari, 2018).

Essentially, in a DNSSEC deployed zone, the DNS response clients are digitally signed. Consequently, the client is able to determine whether the data contained in the response has been modified or not (Anon, 2019). However, the current DNSSEC does not protect DNS from a new form of Denial of Service (DoS) attacks (Zhu et al., 2015). Since DNS data verification poses additional overhead to network and servers. (Kubilay et al., 2018). Despite DNSSEC ensure authentication and data integrity. However, confidentiality of data cannot be guaranteed ().

DNSSEC system is a complex PKI authentication scheme to deploy which as a result, suffer from limited patronage with only %4 of the second level domains signed for DNSSEC (Gourley and Tewari, 2018) (Linkova, 2019).

Consequently, Security of DNSSEC inherently lies on PKI root at ICANN[3] which is vulnerable to central failure. Vulnerability is much easier to exploit with current DNSSEC system (Sehgal and Dixit, 2019). The Key corrective measure is to ensure domain holders and domain operators behave honestly (Matsumoto et al., 2017). Despite the current proposed solutions there is no standard method to incentivitize an entities that behave honestly, the compromised mis-issue certificate is reported manually and its time consuming (Matsumoto and Reischuk, 2016).

Furthermore, DNSSEC using DANE allows domain operators to make judgement about CA based on the statement related to each PKI certificate (Qin et al., 2017). It includes the following information:

- CA restrictions: the client should determine which CA should issue certificates.
- Service certificate restrictions: the client should accept only certain certificates.
- Trust anchor: the client should validate certificates for a particular domain based on the set of available domains provided chain of trust.

[3]https://www.icann.org/

DNS Authentication Name Entity (DANE) DANE use transport layer security protocol to allow X.509 digital certificate to bind domain names via DNS security extension (Dukhovni and Hardaker, 2015). DANE provides an alternatives authentication method using DNSSEC system to store certificates and sign keys that are used by transport layer security. DANE TLSA therefore equip third parties CAs allowing users to have a better control of the certificate issuance for their domains and protect against compromise certificate. DANE foreseen as a basis for binding public key data to domain names, since entities that verify binding public keys are the same entities that are responsible for managing suspicious public keys data and domain names (Zhu et al., 2015).

2.3 Certification Authority Authorization (CAA)

Certificate authority authorization provides additional measures to protect issuing accidental certificates (Scheitle et al., 2018). CAA resource record type was implemented to allow owner of the domain to identify those issuing certificate authorities that are allowed to issue certificates for the domain. During issuing certificates for the domains, certificates authorities evaluate the existence of the certificates thorough animate DNS resolving (Wei-hong et al., 2017). If records are missing or either accidentally authorized a given CA to issue a certificate for the domain. Consequently. Certificate authority authorization provides additional measures to disallow rogue or compromise certificates from issuance. Furthermore, DNS based authority does not prevent security threat associated with certificate authorities. That is presumably possible attacks may be launch to trivially bypass the CAA checking during issuing fraudulent certificate (Ruohonen, 2018).

Consequently, the success of the Certification Authority Authorization (CAA) as a largest DNS security mechanism relies on smooth cooperation between CAs, DNS operators and domain owners. Generally, DNS based extension (Aishwarya et al., 2015); Certificate authority Authorization (Scheitle et al., 2018); are DNS resource record types built to assist certificate issuance and verification.

It clearly shows that reducing level of trust associated with centralized trusted authorities in traditional PKI system, would enormously help to achieve stronger security, and by doing so indeed a cornerstone for achieving robust, and secure PKI systems.

2.4 Blockchain Based PKI

Traditional PKI system is based on centralised trusted authorities to issue certificates and authenticate users. Authentication using blockchain is basically based on decentralised method, with no trusted authorities (Wang et al., 2018). Blockchain based solutions are resistance to most of the attacks associated with current PKI systems (Baldi et al., 2017) (Kalodner et al., 2015). Blockchain records are immutable, temper proof and trust (Ali et al., 2016). Trust in the traditional PKI systems rely on single authorities to authenticate clients(Yakubov et al., 2018). Unlike PKI system, Blockchain based systems trust is distributed among the nodes across the entire network of global ledger. Blockchain technology allow an execution of arbitrary logic known as programmable contract. Smart contract is a program that executes on the blockchain by network of mutually distrustful nodes without requiring a trusted authority (Wang et al., 2018). Several solutions are proposed based on the blockchain technology to address the current challenges of PKI systems namely:

NameCoin:

Namecoin (Kalodner et al., 2015) proposed a namespace blockchain based system that provides a novel solution to the current challenges of decentralized namespaces (Nam,). Namecoin was first Blockchain based PKI solutions implemented, designed to serve as a decentralized DNS for bit addresses. However, one of the major drawbacks of the namecoin solution is that, Blockchain in the namecoin is static that is coin is un-spendable (Kalodner et al., 2015). Meanwhile domain names are stored in a public ledger of every namecoin transaction ever executed, lack of integration with traditional DNS such as browser adds-on usability which has resulted namecoin from widely adoption and use (Xander Lammertink, 2019).

BlockStack:

BlockStack is an implementation of a new type of decentralised internet infrastructure focusing on decentralized application layer of equivalent to traditional PKI/CA system to traditional internet architecture (Ali et al., 2016). Blockstack use blockchain technology to build a domain name system and public key infrastructure alike in a more decentralized structure.

Using Blockstack clients can run a decentralized app using Blockstack browsers and gives the users ultimate right and ownership to control their data (Ali et al., 2016). From one hand, Blockstack implementation provides key revocation to execute in the Blockstack name service. In order to verify the authenticity of the operations in the Blockstack, there is need to download the entire blockchain and process the data using virtual chain. However, verification and authentication is not in the implementation of design of the Blockstack (Ali et al., 2016). Essentially, Blockstack is a project which aim to bring the trusted nature of the Blockchain to all of the Internet. Instead of relying on remote, third party servers for access to data and the Internet, Blockstack will remove those trust points and give them back to the users via blockchains (Dong et al.,).

IKP:

Instant Karmer PKI (IKP) is an implementation of ethereum blockchain design to enhance the certificate transparency, participants in the IKP are encouraged to find and report any rogue certificates and get rewards accordingly in a decentralised way (Matsumoto et al., 2017). The main aim of IKP is to rewards users for vigilance over CAs and automate process of managing CA misbehavior in a more decentralize manner. The implementation was based Ethereum's platform to run the smart contracts. In the IKP system domain owners and the CAs adhere to IKP contract as input using domain registration policies, which specify a list of CAs that allow to issue certificate for the domains. CAs can get punishment based on the contract reaction policy (Matsumoto and Reischuk, 2016). However, the software required to test IKP is not yet widely available (Fries, 2019). Moreover, implementation of IKP is not practical as a result of RSA signature verification required to update the keys which are not in the original implementation of Ethereum smart contracts (Fries, 2019).

2.5 Simplify payment Verification/ Light Client Nodes:

Transactions

Bloom filters

Spark Merkle Hash Tree

Challenges of Blockchain

Security

Network reliability and throughput

selfish mining

concensus breaking and

2.6 Towards a better domain authentication system using Thin client:

One of the biggest challenges with authledger for deployment is that every client store the entire blockchain that is client has to store the entire blockchain in order to validate transactions. In order to validate the transactions, digital currencies like ethereum. This comprises and downloading the entire chain of blocks which it takes sometimes and needs considerable amount of giga space of bandwidth and storage. Thus, less resource intensive devices like mobile phones unable to verify transactions freely without trusting the whole blockchain or full nodes. Simplify payment verification is an approach use in most of the populous blockchains such as bitcoin and ethereum that allow light client to verify transactions by having access to only block headers in the blockchain.

The Ethereum blockchain is currently at () and it appears to be growing linearly at the rate of () a month. A naive deployment of AthLedger requires any device or client to conduct verification to have a large storage capacity. Hence, it is not possible for low resources devices for instance, browser on a smart phones or thin client to have that much available storage or capacity to handle this task.

3 MODEL AND DESIGN GOAL

3.1 Threat Model

In this section, we describe different adversaries capabilities: From Blockchain, to malicious client entities that may launch a colluding attacks (between several compromised CAs).

First scenario we assume there are M full nodes in the blockchain network, in which the entire nodes with p proportions is controlled by malicious attackers, and the other nodes of $1-p$ are honest full nodes;the number of verification nodes is N, among which q accounts for the verification node is controlled by a malicious attacker, and the other $1q$ proportion of the verification nodes are honest nodes.

Additionally we assume malicious entity replicate public keys as the authenticating node to launch (Sybil attack).

Second scenario, from client server side we assume CAs and validating authorities are malicious which act arbitrarily such as binding fake certificate. Moreover, Domain name server (DNS) is assume to be corrupted. We assume that malicious entities cannot collude the hash function of the standard cryptographic protocols.

We also assume when Browser is performing checking relevant to a CA may filter erroneous certificates.

Eclipse attacks MItM attacks Sybil attacks (Letz, 2019)

3.2 Entities and Architecture

Our proposed system consists of five entities defined as follows:

- Certificate Authority (CA): It represents each authorized entity to be able to issue digital certificates in this case, to join AuthLedger to register the information on the Blockchain.
- Domain Name Server (DNS): Maintains the directory of the certificate owner and identity binding.
- Browser Extension: complete domain name Transport layer security (TLS) set up.
- Validating Authority: Put vigilance during CA operations for suspicious certificates and report any misbehaviour for an entity.
- Blockchain: Which contain full nodes verify binding request and confirm request from validating authorities.
- Thin Client:

As depicted in Figure 2 [1] Domain initiates an identity binding request in the blockchain.[2] Validating authority via full nodes verify the requests. [3] A domain updated a trusted CAs in the Blockchain. [4-5] The domain name sends a CA list that is trusted by blockchain network to complete certificate issuance process. [6-7] Client initiates a secure connection through the browser-plug-in to obtain the certificate own by the domain name. [8-10] Browser then requests trusted CAs list from blockchain and compare the validity of the information obtained.

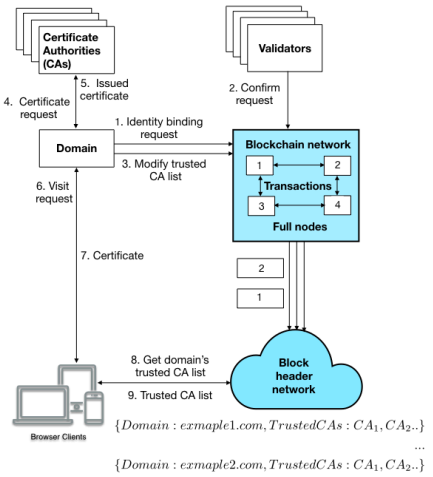

Figure 2: Entities and its Functionalities in AuthLedger

4 DOMAIN NAME AUTHENTICATION DESIGN FOR THIN NODES

4.1 Domain Authentication

In order to achieve the entity authentication in this paper we provide an authentication procedures based on time [T] and count [C], detail process describe below:

The identity of the binding process is broadly divided into 3 steps:

1. The domain name send an authentication request to the blockchain.
2. After transaction confirmation, the transaction and block are placed on the server side for verification.
3. Node is verify the transaction after the verification period or number of times takes to complete the identity binding.

4.1.1 Authentication Verification based on Time

In time-based authentication process when the verification reaches a specified time it owns the domain name server and completes the binding procedures.

As depicted in figure 3.

Time Based Domain Process: Hypothesis

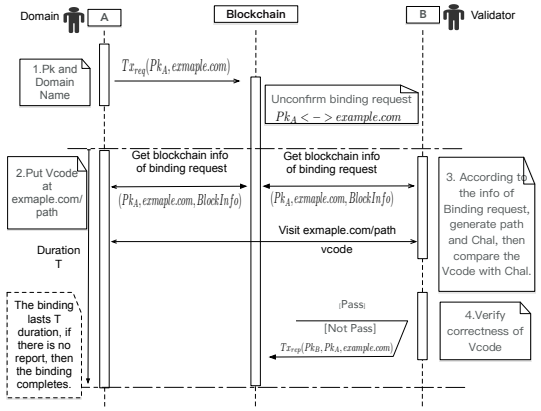

Figure 3: Time Based Domain Authentication

Suppose A owns key fairs and domain name e.g *example.com* such that: $(Pk_A \| Sk_A \| Dm_{example.com})$; verifier B with the following parameters of key fairs: $(Pk_B \| Sk_B)$

Transaction Type

In this scenario, the following three types of transactions are:

- Binding transaction request: Tx_{req} $(Pk_A \| Dm_{example.com})$ binder issues a binding request containing its own public key Pk_A and domain name *example.com*.
- Report transaction: Tx_{rep} while processing the binding, the verification node finds that the binder's authentication information is incorrect. Transaction can be reported and rejected.
- Verify transaction: Tx_{vrf} reporting transaction can be verified by the validator.

Binding Procedures: The specific process of binding is shown in Figure 3, which includes the following steps:

- A Publishes the binding transaction Tx_{req} $(Pk_A, example.com)$ to the blockchain.
- Validator B accesses the domain name to verify that A operates correctly. If not, sends a report transaction Tx_{rep} $(Pk_B, Pk_A, example.com)$ to the blockchain.
- After A maintains the verification time T, the verification service can be stopped and the binding is completed.

4.1.2 Authentication Verification Based on Count.

In count based authentication system verification process reaches the specified number of counts to complete the identity binding as shown in figure 4.

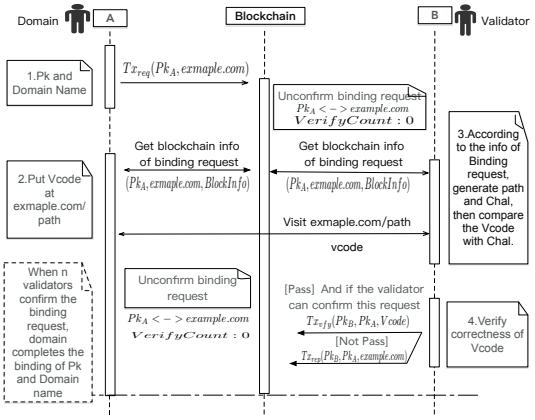

Figure 4: Count Based Domain Authentication Framework.

Count-Based Domain Process: Hypothesis Suppose A owns key fairs and domain name example.com such that: $(Pk_A \| Sk_A \| Dm_{example.com})$; verifier B owns key fairs: $(Pk_B \| Sk_B)$.

Transaction Type: In count based the transaction verification process includes the followings:

- Binding transaction request: $(Pk_A \| Dm_{example.com})$ binder issues a binding request containing its own public key and domain name example.com.
- Reporting transaction: Tx_{rep} if the verification node finds that the binding information is not accurately placed then identity binding can be rejected.
- Verification transaction: Tx_{vfy} validator completes the comparison of the verification information, the validator sends the transaction that has passed the verification process.

Binding Process:

Authentication process based on the number of validations require more interaction between the validators and the blockchain than time based.

- A Publishes binding transaction T_{xreq} $(Pk_A, \| example.com)$ on the blockchain.
- A gets the block information $Info_{block}$ of the location where T_{xreq} is located, and calculates its value $(Path, Chal) = F(Info_{rcv}Block)$ to put it under domain control.
- B obtains the verification content from example.com/Path, submits the transaction Tx_{vfy} $(Pk_B, Pk_A, Vcode)$ to complete the certificate

verification. Once the K has been verified, the binding is complete. The entire process same as time-based requiring the binder to publish her public key Pk_A, domain name example.com onto the blockchain.

4.2 Reporting Misbehaviour

Two different identity binding schemes given above are used in order to report any suspicious transactions or fake binding to complete the process. When an honest verification node C discovers a suspicious binding, it initiates a report transaction to the blockchain which is basically the same as the content submitted by the authentication transaction, but replaces the last $Vcode$ with the domain name address that needs to be bound, so the report transaction contains its own public key PkC and binding the public key PkA and the domain name example.com.

However, for a malicious verification node, it may also initiate a report transaction against a true and valid binding. In order to confirm the validity of a report transaction, it is necessary to judge the transaction by the verification node on the blockchain. At this point, the reported transaction and the domain name submitted by the binding transaction are single entities in the P2P network to verify the request. When the report transaction is submitted to the network and confirmed by the blockchain just like the binding request randomly select n validators to verify the request and determine the validity of the reporter.

4.3 Selection Verifier or Look Up

In spite the advantages provides by Auth-Ledger to weaken the CAs to issue certificates to any domain, clients also have to be prevented from the use of corrupted/ erroneous certificate. In Time-based scheme, a process for selecting a verifier is necessary to prevent the node from colluding the verification transactions of other nodes for its own benefit. Moreover, if the verification node is not filtered by certain rules other node can verify the binding identity. In this scenario, the malicious binder can use different public keys as the authentication node and then confirm the malicious binding to launch an attack ().

Meanwhile, in order to get more rewards, the legitimate verifier will also use as many accounts as possible to verify the transaction in order to get more incentives. In order to avoid the above situation, this scheme designs a method of randomly selecting verification nodes according to the binding information to ensure that the verification can be carried out effectively. When the binding information is publish on the blockchain, it is converted to the reference value $hashc_{mp}$ according to the binding information PkA, $example.com$ and the block information $Infor_{cvBlock}$ of the transaction location:

hashcmp =SHA512 (PkA example.com InforcBlock) (3.3) After the reference value is obtained, the verification node will be judged by signing the block in- formation incorporated into the block, the method of judgment is shown below:

distham =SHA512 (Sigprc(InfocurBlock)), hashcmp)<=d+ t(3.4)

Where distham(a,b) represent two numbers a and b; Prs represents the private key of the authentication node.d ; t;just; $Info_{curblock}$; sig is a unique signature algorithm. The above judgment method enables a verifier to make an attempt every time a block is generated, and judge whether or not it would become a verification node. also, as time passes, the difficulty of becoming a verification node is also decreasing, ensuring that a verifier will be selected. table

add

In the Time-based scheme, a protocol for selecting a verifier is needed to prevent the node from incorporating or lagging the verification transactions of other nodes for its own benefit; more importantly, if the verification node is not filtered by certain rules rather, each node can verify the binding identity. In this scenario, the malicious binder can use different public keys as the authentication node and then confirm the malicious binding that is launch an attack().

Meanwhile, in order to get more rewards, the verifier will also apply for as many accounts as possible to verify the transaction.

When the binding information is publish on the blockchain, it is converted to the reference value $hash_{cmp}$ according to the binding information Pk_A, example.com and the block

information $Info_{rcvBlock}$ of the transaction location:

$$hash_{cmp} = \text{sha}_{512}(\text{Pk}_A \| example.com \| Info_{rcBlock}) \quad (3.3)$$

After obtaining the based value the verification will judge by signing the block information in the blockchain. Whether or not you have the right to verify this request, the method of judgment is as shown in 3.4:

$$dist_{ham} = \text{sha}_{512}(\text{Sig}_{prc}(Info_{curBlock})), \text{hash}_{cmp}) <= d + \Delta_t \backslash t_{adjust} \quad (3.4)$$

4.4 Verify the accompanying information for the transaction:

The verification node monitors the identity binding request submitted on the blockchain at any time together with the corresponding report information. When the two messages are posted on the blockchain, it means that the verification node needs to participate in the confirmation of the event, and each verification node there will be a chance to be the node to verify the event. For the identity binding request and the confirmation of the prosecution information, both the public key *PkC* and the binders public key *PkA* must be submitted. The difference is that one needs to provide the verification content *Vcode*, and the other needs to provide the domain name *example.com*. On this basis, the verifier is also required to provide a content based on the capacity proof, to ensure that the identity is not generated at random, but at a certain cost. Suppose the proof of capacity to be provided is pr f , which needs to meet the following conditions: Verify(Pkc,Prf)= SHA512 (PkA Vcode|example.com InfocurBlock)(3.5)
Where Veri f y is the verification function of the capacity proof.

4.5 Financial Incentives

In order to encourage enough verification nodes to join the network and ensure the identity binding can be completed, some rewards should be given to the verification nodes. During the verification process, if there is a domain name that is not reasonably placed with the verification information, the report transaction can be initiated to the blockchain, and after the transaction is confirmed, it can be rewarded accordingly; In order to ensure the reward balance in the system, the initiator of the binder has to pay a certain fee, and the verifier will get the corresponding reward when discovers the error. Because these entities exist in a network, all identities are represented by public keys (or addresses converted from public keys); as described in the previous scenario, assuming the domain name has *PkA*, the verifier has *PkB* , and the reporter has *PkC* , each has its own account balance on the blockchain. The penalty and reward mechanism for related op- erations are shown in table 3:

As shown in table 1:

First rule In table 1, fee is deposited by the initiator, instead of unlimited arbitrary identity binding operations; thus avoiding malicious adversary to initiate unlimited invalid identity binding. Second rule prevent the verification node from randomly initiating a reporter transaction to disrupt the completeness of the normal binding; The third rule node that incentivises the verification operation to attract more nodes, but needs to deposit a fee to prevent some nodes from misbehaving Fourth rule same as third rule attract more nodes to join the system.

5 SECURITY ANALYSIS

we analyze the security of the proposed design based on two perspectives: Blockchain and Domain client server. We also compare AuthLedger with the current approaches of the PKI authentication schemes from security and privacy point of view in table 4. We analyzed the security of our proposed system from the Blockchain perspective according to the following properties:

Property 1: Full node in the blockchain net- work with p proportion is controlled by malicious attackers to launch %51 Attack obviously, when p $> 1/2$, the attacker can complete the control of the blockchain network regardless of whether the under- lying blockchain uses the PoW consensus mechanism.

Property 2: Malicious Binding Analysis. When an authentication request is initiated, it needs to be verified from the verification node selected in the network. According to the selection process of the previous validator, it is known that the process of selecting the validator each time is a random process. In this case, if the random al-

Table 1: Table describe incentive parameters.

Txt. Name	Initiator	PK	Txt. Def.	Txt. Fee	Incentives	Balance
Tx_{req}	Domain Name	Pk_A	Binding Identity	-P_1	0	Balance (Pk_A)−P_1
Tx_{rep}	Reporter	Pk_C	Reporting	-P_2	$R_1 = P_2 + \frac{P_1}{2}$	Balance (Pk_C)+$\frac{P_1}{2}$
Tx_{vfy1}	Validator	Pk_B	Validation	-P_3	$R_2 = P_3 + \frac{P_1}{K}$	Balance (Pk_B)+$\frac{P_1}{K}$
Tx_{vfy2}	Validator	Pk_B	Confirm	-P_3	$R_3 = P_3 + \frac{P_1}{2n}$	Balance (Pk_B)+$\frac{P_1}{2n}$

gorithm we designed is completely random, then the probability of selecting entire K nodes as malicious nodes is:

$$Pr_1 = \prod_{i=0}^{K-1} \frac{qN-i}{N-i} \tag{1}$$

Following our design requirements, when the report transaction is generated, the validator will be selected according to the content of the report transaction, and half of the votes will be confirmed. The probability of passing the confirmation is:

$$Pr_2 = \prod_{i=0}^{n/2} \frac{qN-i}{N-i} \tag{2}$$

Therefore, when an attacker has a verification node that accounts for the entire network q, the prob- ability of completing a malicious binding is as shown in Equation 3.

$$\begin{aligned} Pr_{attack} &= Pr_1 + (1-Pr_1) * Pr_2 \\ &= Pr_1 + Pr_2 - Pr_1 * Pr_2 \\ &= \prod_{i=0}^{K-1} \frac{qN-i}{N-i} + \prod_{i=0}^{n/2} \frac{qN-i}{N-i} - \prod_{i=0}^{K-1} \frac{qN-i}{N-i} * \prod_{i=0}^{n/2} \frac{qN-i}{N-i} \end{aligned} \tag{3}$$

Property 3:Denial binding analysis. we assume when binding authentication request is initiated, identity binding is completed after waiting for the validator to complete the verification. However, in case malicious attacker may disrupt the binding. The probability of successfully disrupting this request is shown below:

$$Pr_{denial} = \prod_{i=0}^{n/2} \frac{qN-i}{N-i} \tag{4}$$

Property 4:Sybil Attack. the biggest security flaws of decentralised systems are malicious nodes (Sybil nodes) Blockchain is built based on open P2P network any node can generate number of identities to increase the probability as verification node. Therefore, additional information is required including a verifiable workload certificate. (Counter measures)

Property 4:Hard Fork.

(Solat, 2017) fork arises when there are too different block discover in the network at the same time the fork inadvertently poses a security vulnerability wherein attackers can replicate transactions into the other network; this continues to this day, and we quantify this behavior by using.....

Domain server perspective includes the following properties:

Property 1: We assume most of the Certificate authorities and validating authorities are malicious which act arbitrarily such as binding fake certificate or certificate can be issued from invalid CA. We assume that Domain name server (DNS) are corrupted.

We assume most of the Certificate authorities and validators are malicious which act arbitrarily such as fake binding certificate or certificate can be issued either from CA which is invalid or corrupted due to insufficient security policies. We assume that Domain name server (DNS) are assume to be corrupted. In addition to that, when a malicious attacker gain control over the domain name, in this case, may likely to have a control over the trusted CA to launch an attack. In order to reduce the impact of this situation, in this this paper we suggests that the do- main name should place a trust list change monitoring program on different servers, and promptly give reminders when the trust list changes, for the domain name owner to confirm. Meanwhile to ensure the authentication binder has a control over the domain name server, in the above scenario, the submitted Tx r e q is used to generate the authentication content Path and Chal, which ensures the correlation between the verification content and the submission request as in formula (1).

The submitted content is converted to the verification content by using sha256 as a hash function, which is an reversible processes. In the placement of the verification content, Vc o d e is used to bind

the public key corresponding to the private key signature of the Chal, to ensure the ownership of the private key binder.

Property 2: Browser client. Whether Browser is performing checking relevant to a CA may filter erroneous or corrupted certificates. As mentioned in this paper the main purpose is to weaken the CA to issue a certificate of any domain using blockchain technology. In order to make sure any certificate issued by adversary is detected, additional checks need to be conducted to the client side. Therefore browser need to interact with Blockchain to query whether the received certificate is granted by a trusted CA. Additional checks need to be conducted from the client side to ensure any certificate issued by adversary is detected.

Advantages/Benefit of Blockchain Based PKI

Security Comparison of the PKI Authentication system as described in table 2.

6 EVALUATION

6.1 Implementation Prototype.

Entities involve in the implementation process include the followings: 1. User: domain sends identity binding request and update trusted CA list to interact interface in the Blockchain. 2. Relying Party: relying party obtains a certificate through a browser and establishes secure communication, it needs to check the validity of the certificate through Blockchain by query trusted CA list. 3. Verification Nodes: verify any domain name initiated authentication requests on the blockchain. Also verification node needs to monitor and report any misbehaviour during verification process. 4. Full node: Monitor all transactions in the Blockchain network. Each time a block is created, the initiator will get the reward corresponding to the proportion of the computational power use. Figure 5 describe the system architecture of the proposed system.

6.2 Smart Contract Solidity

All entities in the proposed system need to be connected through a blockchain. The functions in the smart contract include the followings:

The domain name authentication function: When the domain name calls this function, a series of

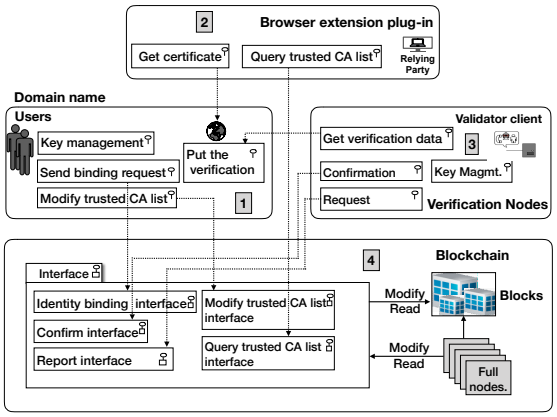

Figure 5: System Architecture

verification nodes are selected to complete the verification through the verification node.

Trusted list storage: Update the data stored in the smart contract and complete the control of its own trusted CA.

Trust list query: complete the query of the trusted CA list of a specific domain name.

Implementation of these modules relies on Ethereum smart contract to perform the functions of the above interfaces using Solidity programming language. Moreover, Solidity is used to complete the mapping between the Ethereum contract address and the domain name. Sample code for interactions describe below.

```
struct Domain {
        string name;
        uint count;
        string trustCAs;
        uint stBlock;
        address[] validator;
        address addr;
        bool isEntity;
    }
    mapping(address => Domain)reg_domain;
    mapping(string => Domain)auth_domain;
    uint constant auth_times = 10;
    uint constant limit_blocks = 100;
```

Meanwhile based on the the underlying blockchain platform, transactions that cause any changes in the Ethereum blockchain need to consume so-called Gas. In order to call transaction on ethereum smart contract, the transaction needs to be sent to the blockchain first. The required cost is called transaction costs, which is calculated according to the large size of the data. While execution cost, it needs to be based on the calculation of the transaction to perform cost assessment. The current Gas

Table 2: Table describe security comparison of the proposed system.

	DNSSEC-DANE	CAA	AuthLedger
Level of Trust	Weak	Weak	Strong
50% attack protection	N/A	N/A	Strong
Sybil attack protection	N/A	N/A	Strong
Fake binding detection	Weak	Weak	Strong
Domain compromise	Strong	Strong	weak
Misbehaviour Incentives	Weak	Weak	Strong
Browser validation	Weak	Weak	Strong

station estimate is : 1 gas = 3 Gwei[4]. The cost of the binding and trust list modification for example.com describe below:

6.3 Browser Plug-in Validation

Browser selection: The main operation of plug-in is to obtain the trust CA list of the domain name in the blockchain and and compares with the certificate issuer. Browser validate of the trust CA list that has been recorded on the blockchain. it can be done based on the peer to peer network.

6.4 Experiment

In order to test the performance of the proposed system, we used Ali Cloud server configuration is as follows: CPU: 1 core, Operating System: Ubuntu 16.04 (64 bit), Memory: 2GB, Disk: 40GB, Golang: 1.8. and Ethereum: Titanium (v1.8.7). Meanwhile machine configuration we used, CPU: 8 cores, Operating System: macOs High Sierra (version 10.13.3) and Chrome: Version 66.0.3359.139 (64-bit).

Blockchain network contain five Ali cloud servers: full node; to complete the blockchain maintenance network; the other five servers as the authentication nodes, each server starts 10 authentication clients, a total of 50 authentication nodes; one Ali cloud deploys web services as a domain name server and runs the domain name guest. The local PC acts as a relying party, simulates the access to the domain name and completes the acquisition of the trusted CA list.

[4]https://ethgasstation.info/

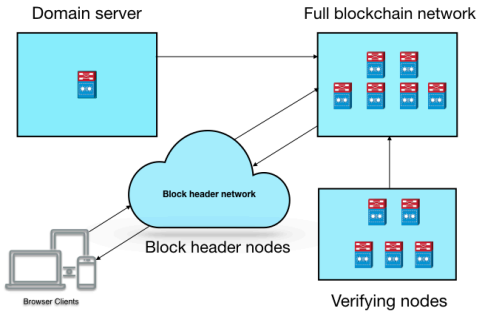

Figure 6: System configuration

7 CONCLUSION AND FUTURE WORK

In this paper, we proposed AuthLedger a novel Blockchain-based domain name authentication scheme using Ethereum smart contract to allows a client to trust which CA can issue a certificate for the domain using Blockchain technology. The paper demonstrated an efficient and trustworthy algorithm for certificate authentication process. we also analyze security implication of the proposed scheme by discussing different security threats and countermeasures. Moreover, since this is a short paper, future work will concentrate on detail implementation of the AuthLedger to get more experimental results. Will conduct a detailed performance and usability analysis of the proposed system.

Table 3: Trade execution cost.

Name of Trade	Send Data	Tx. Fee (Gas)	Impl. Cost (Gas)	TTL Cost (Gas)	Price ($)
Binding request	Pk_A, *example.com*	189451	165555	355006	1.977
Verification operation	Pk_A, *example.com*, V_{code}	208475	197342	405 817	2.26
Modify trust list operation	"CA List"	36906	14674	51580	0.287

Acknowledgement This work is partially supported by the National Key Research and Development Program of China NO.2018YFB0803601. and NSFC No.61672060.

REFERENCES

Namecoin. https://namecoin.org/. Accessed: 2019-01-24.

Aishwarya, C., Raghuram, M., Hosmani, S., Sannidhan, M., Rajendran, B., Chandrasekaran, K., and Bindhumadhava, B. (2015). Dane: An inbuilt security extension. In Green Computing and Internet of Things (ICGCIoT), 2015 International Conference on, pages 1571–1576. IEEE.

Ali, M., Nelson, J. C., Shea, R., and Freedman, M. J. (2016). Blockstack: A global naming and storage system secured by blockchains. In USENIX Annual Technical Conference, pages 181–194.

Anon (2015 (accessed May 15, 2019)). DNSSEC is Unnecessary. https://sockpuppet.org/blog/2015/01/15/against-dnssec/.

Baldi, M., Chiaraluce, F., Frontoni, E., Gottardi, G., Sciarroni, D., and Spalazzi, L. (2017). Certificate validation through public ledgers and blockchains. In ITASEC, pages 156–165.

Berkowsky, J. A. and Hayajneh, T. (2017). Security issues with certificate authorities. In Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON), 2017 IEEE 8th Annual, pages 449–455. IEEE.

Dong, Y., Kim, W., and Boutaba, R. Conifer: centrally-managed pki with blockchain-rooted trust.

Dukhovni, V. and Hardaker, W. (2015). Smtp security via opportunistic dns-based authentication of named entities (dane) transport layer security (tls). Technical report.

Enisa (2016). The WoSign Incident and Considerations.

Fries, J. (2017 (accessed May 20, 2019)). Using the blockchain to add automated financial incentives to the Public Key Infrastructure,. https://www.net.in.tum.de/fileadmin/TUM/NET/NET-2017-09-1/NET-2017-09-1_05.pdf.

Gourley, S. and Tewari, H. (2018). Blockchain backed dnssec. In 1st Workshop on Blockchain and Smart Contract Technologies - 21st International Conference on Business Information Systems, Fraunhofer FOKUS, Berlin, 18-20 July, 2018, pages 357–388. Fraunhofer FOKUS, Berlin.

Kalodner, H. A., Carlsten, M., Ellenbogen, P., Bonneau, J., and Narayanan, A. (2015). An empirical study of namecoin and lessons for decentralized namespace design. In WEIS. Citeseer.

Kamat, P. and Gautam, A. S. (2018). Recent trends in the era of cybercrime and the measures to control them. In Handbook of e-Business Security, pages 243–258. Auerbach Publications.

Karaarslan, E. and Adiguzel, E. (2018). Blockchain based dns and pki solutions. IEEE Communications Standards Magazine, 2(3):52–57.

Khan, S., Zhang, Z., Zhu, L., Li, M., Safi, K., Gul, Q., and Chen, X. (2018). Accountable and transparent tls certificate management: An alternate public-key infrastructure with verifiable trusted parties. Security and Communication Networks, 2018.

Kiayias, A., Russell, A., David, B., and Oliynykov, R. (2017). Ouroboros: A provably secure proof-of-stake blockchain protocol. In Annual International Cryptology Conference, pages 357–388. Springer.

Kubilay, M. Y., Kiraz, M. S., and Mantar, H. A. (2018). Certledger: A new pki model with certificate transparency based on blockchain. arXiv preprint arXiv:1806.03914.

Lencse, G. and Kadobayashi, Y. (2018). Methodology for dns cache poisoning vulnerability analysis of dns64 implementations. INFOCOMMUNICATIONS JOURNAL, 10(2):13–25.

Letz, D. (2019). Blockquick: Super-light client protocol for blockchain validation on constrained devices. IACR Cryptology ePrint Archive, 2019:579.

Linkova, J. (2016 (accessed May 15, 2019)). Let's

talk about IPv6 DNS64 DNSSEC,. https://blog.apnic.net/2016/06/09/lets-talk-ipv6-dns64-dnssec/.

Matsumoto, S. and Reischuk, R. M. (2016). Ikp: Turning a pki around with blockchains. IACR Cryptology ePrint Archive, 2016:1018.

Matsumoto, S., Reischuk, R. M., Szalachowski, P., Kim, T. H.-J., and Perrig, A. (2017). Authentication challenges in a global environment. ACM Transactions on Privacy and Security (TOPS), 20(1):1.

Qin, B., Huang, J., Wang, Q., Luo, X., Liang, B., and Shi, W. (2017). Cecoin: A decentralized pki mitigating mitm attacks. Future Generation Computer Systems.

Rashid, F. Y. (2015). Google threatens action against Symantec-issued certificates following botched investigation.

Ruohonen, J. (2018). An empirical survey on the early adoption of dns certification authority authorization. arXiv preprint arXiv:1804.07604.

Scheitle, Q., Chung, T., Hiller, J., Gasser, O., Naab, J., van Rijswijk-Deij, R., Hohlfeld, O., Holz, R., Choffnes, D., Mislove, A., et al. (2018). A first look at certification authority authorization (caa). ACM SIGCOMM Computer Communication Review, 48(2):10–23.

Sehgal, A. and Dixit, A. (2019). Securing web access—dns threats and remedies. In Emerging Trends in Expert Applications and Security, pages 337–345. Springer.

Shulman, H. and Waidner, M. (2017). One key to sign them all considered vulnerable: Evaluation of dnssec in the internet. In NSDI, pages 131–144.

Wang, Z., Lin, J., Cai, Q., Wang, Q., Jing, J., and Zha, D. (2018). Blockchain-based certificate transparency and revocation transparency . Financial Cryptography and Data Security. Springer International Publishing.

Wei-hong, H., Meng, A., Lin, S., Jia-gui, X., and Yang, L. (2017). Review of blockchain-based dns alternatives. , 3(3):71–77.

Xander Lammertink, M. D. (2015 (accessed May 15, 2019)). Namecoin as alternative to the domain name system,. https://www.sidnlabs.nl/downloads/theses/report.

Yakubov, A., Shbair, W., Wallbom, A., Sanda, D., et al. (2018). A blockchain-based pki management framework. In The First IEEE/IFIP International Workshop on Managing and Managed by Blockchain (Man2Block) colocated with IEEE/IFIP NOMS 2018, Tapei, Tawain 23-27 April 2018.

Zhu, L., Wessels, D., Mankin, A., and Heidemann, J. (2015). Measuring dane tlsa deployment. In International Workshop on Traffic Monitoring and Analysis, pages 219–232. Springer.