A Blockchain-based E-petition system

Xiangman Li August 2021

# Introduction

Petition systems have always been a means of conveying public opinion to those in power. In the 21th century, petitions are one of the most common forms of political activity [1]. With the development of the Internet, e-petitions are becoming more and more popular. The most popular e-petition system was the UK parliament’s upgrade e-petitions systems in 2015. 14 million people signing at least one petition, and over 30,000 e-petitions are received [2], becoming the highest usage parliamentary e-petitions system worldwide. At the same time, there are many petition sites to help people start and run non-political petitions. The e-petition system is expected to solve some of problems with the paper-based petition system. First, the paper petition ballots are counted by the Counting Agent, which has possible to change the number of votes accidentally or intentionally. Second, since the counting process and results are managed and stored centrally, it the public is hard to verify the correctness of the paper petition result. At the same time, the result would be incorrect due to the paper petitions could get lost in the mail by accident.

However, the petition system still has plenty of critics because the public’s expectations are high, and little has been achieved. The public still cannot check the result on time and verify if the result is fair. In addition, personal information is not suﬃciently protected, and petitioners are concerned that their information will be leaked on the Internet and cause unnecessary trouble. In addition, the paper-based petition would have a hard time dividing and statistic the petitioners, and it is challenging to arrange diﬀerent groups casting diﬀerent numbers of voting. Hence, using Blockchain and Attribute-Base Signature (ABS) to build a petition system is considered.Blockchain is decentralized through the participation of members.No single point of failure. No single user and group can have control, and the data entered on Decentralized Blockchains are irreversible, which means anyone can participate and validate the data. These features of Blockchain promise that the process and result of the petition are fair and transparent. In an attribute-based signature (ABS), users sign messages with using their attributes from an attribute authority, which means a signature is a claim regarding the attributes that signer owns; it does not show the signer’s identity.[3] The primary ABS considers the single attribute authority, but El Kaafarani et al.[4] and Okamoto et al.[5] provide a scheme that can avoid central authority and involve multiple attribute authorities, which means diﬀerent signers would satisfy the diﬀerent policies and sign diﬀerent messages. This solution avoids having a designated tracing authority but also adds accountability to attribute-based signatures. In addition, it allows signers to link some of their signatures with the same veriﬁer directly and keep anonymity.[6]

**Contribution.** Consequently, in this report, our target is improving a blockchain-based and attribute base signature e-petition systems. The contributions are listed as follows.

* 1. We formalize the system model of e-petition systems based on blockchain and attribute base signature. 2)A construction of e-petition system based on blockchain and attribute base signature is described. It proves that it satisﬁes fairness, multiple attributes, keeps identity privacy,and unforgeability. In addition, we

modify the blockchain-based voting system scheme in [7] and the ABS in [6] to establish the basic e-petition system and add statistical steps to complete the self-tallying part in the e-petition system.

# Related Work

## E-voting system

E-voting system is similar to the e-petition system, which they have the similar voting process. Vot- ing systems have been a hot topic, and there are many blockchain-based voting systems out there. Most blockchain voting systems tend to use cryptocurrencies to vote on candidates, such as Bitcoin[9][10], and Privacy-enhancing altcoins[11] are traditional cryptocurrency-based voting systems. There are also voting systems built using Ethereum and smart contracts[12]. The Zerocoin voting system[11] helps voters maintain their privacy better. At the same time, the decentralized automatic statistical electronic voting protocol based on the blockchain[13] designed by McCurry et al. is better to constrain the voting rules.

## Attribute-based Signature(ABS)

So far, attribute-based encryption has spawned many new systems.ABS is mainly divided into three poli- cies: non-monotonic policy, monotonic policy, and threshold-based policy. At the same time, ABS is derived from initial multi-authority ABS[16], traceable ABS[5], linkability ABS[6], etc. Each system has its distinct characteristics, which give unlimited possibilities for encrypted signatures.

## E-petition system

Most of today’s e-petition systems are used by government and school departments to collect people’s wishes. Some petition systems[14] use statistics on addresses, IP addresses, or email addresses and use multiple layers of encryption to protect people’s privacy. The cost of this approach is proportional to the number of petitioners, and the system is prone to failure if the number of petitioners is large. At the same time, people worry that their information will be leaked due to a failure of the information base. At the same time, there are petitioning systems that combine blockchain and javascript to guarantee that each person can only vote once[15].

# Syntax of e-Petition system

## System model

As mentioned before, users who have diﬀerent attributes will sign the diﬀerent messages, representing dif-

ferent voting numbers. In addition, in Attribute-based Signature, a set AA = *{AAi}n* of attribute authorities

*i*=1

are established, and AA*i* is the space where attribute authority *AAi* manages attributes. Suppose that *α c* AA

is a set of attributes that satisfy a speciﬁc predicate, and *aeα* =*÷ a*A*i*, so attribute a belongs to *AAi*.

* + - **Setup** (1*λ*): This algorithm is executed to set up the whole system. It returns a public parameters pp

when input a unary string, which *λ* is the security parameter.

*x* Account Establishment: This protocol is executed to establish the user’s account. It has three algorism AASetup, UkeyGen, AttKeyGen.

*x* AASetup(aid,**pp**): Generating public/private key pair(pk*AA, skAA*) by attribute authority AA*aid*. The authority keep sk*AA* privity and release pk*AA* to the public.

*x* UkeyGen (id,pp): Generating user’s private key *skid* by user id.

*x* AttKeyGen (*f* (*skid*),id,a,*skAA*): Is run by AA that manages attribute a. AttKeyGen generate the private

key skid of user id, and bound id and injective one-way function *f* (*skid*).

After establishing the account, users will get a privacy account and are eligible for signing in the next step.

* + - **Sign** (p, Ω, *skid,α*,*skid*): This algorithm is executed to produce a signature if the user’s attributes satisfy the predicate Ω. Users can use corresponding secret keys *skid,α* =*{skid,αi }αieα* to generate a signature *σ* = *σABS* on the petition p.
    - **Verify** (*σ, {*pk*AA}i*, Ω, p): Take the signature *σ* on the petition p and verify the signature. The attribute authorities’ veriﬁcation keys *{pkAAi }i* manage attributes that are involved in Ω. It will return 1 if the signature is valid.
    - **Tally** (verify(), c*AAi,*(t*i*)(*ien*),MP): This algorithm statistic valid signature, and diﬀerent attribute au- thority will has diﬀerent count number. After the petition number is equal to the required number, it will

show the percentage of each petition received and output the petition Result.

## Security Model.

The security requirement of an e-petition system is shown as follows.

**Anonymity.** This require that the signature secret the user’s identity and the attributes. We consider the adversary as follows,

*x*Adversary’s Power: The adversary control all attribute. It can take the signer’s private key who is chosen.

At the same time, it can get any attributes’ secret key and it owns a signing oracle which can question messages

on behalf of honest users.

**Unforgeability.** Users cannot refer to signature policies that their attribute set is not satisﬁed to output signatures on messages, even if they collect attributes together. This policy guaranteed clustering resistance.

*x*Adversary Power: It owns a signing oracle, and it can ask for user’s personal secret key. At the same time,

it can corrupt attribute authority and ask for attribute’s secret key.

**Correctness.** Signatures generated by honest users must be properly veriﬁed.

## Block model.

**Block model.** Bilinear Group. A bilinear group is a tuple P =(G1,G2,G*T* ,*p*,*g*1,*g*2,*e*), where G1, G2, and G*T* are groups of a prime order *p*, *g*1 and *g*2 generate G1 and G2. [6] We would like use Type-3 in [8] to build our bilinear map G1 *o* G2 *-* G*T* , which G1 */*= G2, G1 and G2 do not have computable isomorphisms.

**Digital Signature.** We would like to use Digital Signature (DS) scheme which called one-time signature. It is correct and unforgeability, which means the adversary can just sign one-times only if it generates a new public/private key pair. We will use a diﬀerent type of full Boneh-Boyen signature scheme according to [6],

namely, BB *十* scheme. The schemes are described as follows,

*y*

*x* KeyGen(*P* ): Let *x, y -* Z*←*, set (*X,Y*) = (*gx, g* ), which (*X,Y*) is the public key and (*x, y*) is the private

2 2

key.

*x* BB.Sign(sk,*m*): Take the signature *σ* = *g*1*/*(*x*+*ry*+*m*) on massage *m e* Z*←*. In *σ*, *r -* Z*←* and *x*+*ry* +*m /*= 0.

In addition, the signature *σ* = (*g*1 *x hz*)(1*/x*+*ry*+*m*) in BB│.

1

*x* Verify: In BB│ scheme, if *e*(*σ, X x Y r x gm*) = *e*(*g*1 *x hz, g*2) output 1, otherwise 0.

2

1

**Non-interactive Zero Knowledge Proofs.** In this system, each proof needs its own new public refer- ence string. Common reference strings are usually not random strings. According to [6], a Non- interactive Zero-Knowledge Proof system is a tuple of algorithms for relation *R*. *R* is an NP relation on pairs (*x, y*) with a

language *C/*= *{yl ax,* (*x, y*) *e R}*. The NIZK system is composed by (Setup, Prove, Verify, Extract, SimSetup,

SimProve). NIZK is required Completeness, soundness and zero-knowledge. Completeness required that the

honest veriﬁer will be trust this statement if the statement is true. Soundness required that only those who can prove that they know the intended secret will get the information. Zero-knowledge required that no witness information is shown.

# Construction of e-petition system

## Overview of construction

In this section, the construction of e-petition system is described. In Setup algorithm, the common reference

*crs* is generated for NIZK system. Then a pseudo-key pair (**pkpsdo***,* **skprdo**) is generated for Digital Signature scheme **D2**. At the same time, a public parameter **pp** = (**pkpsdo***,* **crs***,* A*, H*), where *H* is the collision-resistant hash function.

In account establish algorithm, a public/private key pair(**pkaid***,* **skaid**) is generated for Digital Signature Scheme **D1** after a new attribute authority is created. If the user **id** needs to generate a signing key that the

attribute is *a e* A,the managing attribute authority the managing attribute authority need to sign **id**, a and one-way function on user’s private key **skid** together. This signature is the private key for user’s attribute.

Then the user will use this signature to sign the messages that the attributes satisﬁed. The signature will generate a NIZK proof that the modiﬁed predicate Ω*/* will still has proper user’s attribute be satisﬁed.

According to the [6], the languages for the NIZK proofs are deﬁned as follows. We deﬁne a monotone Boolean function Φ : *{*0*,* 1*}n -* 0*,* 1, and a matrix in a ﬁeld F has the size *l o w*. Then a labelling function

make a transition from *l* to *n*. If every (*x*1*, . . . , xn*) *e {*0*,* 1*}n*, M is a monotone span program for Φ *over* F*, and v*

is the secret vector in it, which means,

[Φ(*x*1*, ...xn*) = 1] *兮* [*av e* F1*×t* : *v x M* = [1*,* 0*, ...,* 0] (*Vi* : *xa*(*i*) = 0 *÷ vi* = 0)]

Base on the deﬁnition of the span program, the satisﬁability of Ω*/* can be proved. User need to prove the property of the *v e* Z*\_*Ω*卜\_* with *v x M* = [1*,* 0*, ...,* 0]. The zero element in *v x M* is the attribute that user does not need, and user need to prove the property of the attribute and pseudo-attribute if the element is non-zero.

*p*

In Tally algorithm, *cAAi* need to count diﬀerent times base on the petition rule. If Verify (*σ, {pkAA}i,* Ω*, p*) = 1, then *cAAi* will be counted until *cAAi* get the maximum number of petitions MP. Then, the algorithm will

calculate the ratio of *cAAi* to the MP, and output the result.

* **Setup(**1*λ***)**:

*x* (*crs, xk*) *- NIZK.Setup*(1*λ*)

*x* (**pkpsdo***,* **skprdo**) *- DS.KeyGen*(1*λ*)

*x* Compute a Collision-resistant hash function *H* : *{*0*,* 1*}\* - MDS x* **pp** = (**pkpsdo***,* **crs***,* A*, H*)

*x* Return **pp**

* **Account Establish**:

*x* (**pkaid***,* **skaid**) *- DS.KeyGen*(1*λ*)

*x* Return (**pkaid***,* **skaid**)

*x skid -* UKeyGen(*id, pp*)

* **Sign** (*p,* Ω*, skid,α, skid*) :

*x* Return *l* if Ω(*A*) = 0

*x* Ω*/* := Ω *apsdo,* **M** *e* Z*α×θ* is the Ω*/* span program.

*x* Compute *α* := *{ai}\_*Ω*卜\_* to be the attribute in Ω*/*

*i*=1

*x π - NIZK.Prove*(*crs, x, y*) if R(*x, y*) = 1. ***『***

*x* ((**pk** = *{pki}\_*Ω*卜\_,* **a** = *{ai}\_*Ω*卜\_*), (sk*id,* id*,*v,***#* = *ī#á }|*Ω *|***)) :

*i*=1

*\_*Ω*卜\_\_*1

D1*.*Verify(pk*i*,id,sk*id,*a*i, σai* )=1)

*i*=1

***￥ ≥*=1**

(***谷***M=[1,0,...,0])

(*vi* = 0

*i* =1

(v*\_*Ω*卜\_*=0 D2*.*Verify(*pkpsdo, apsdo,σapsdo* )=1)

*x* Return *σ*:=*{σa }n*

*i*

*i*=1

* **Verify** (*σ, {*pk*AA}i*, Ω, p)

*x* Return NIZK.Verify(crs,*π*)

* **Tally**(verify(), c*AAi,*(t*AAi* )(*ien*), MP)

*x cAAi* = *cAAi* + (*tAAi* )(*ien*) if Verify(*σ, {*pk*AA}i*, Ω, p) = 1

*x* Return *cAAi* if *cAAi* = MP

*x* Return ( *cAAi* )*n*

*MP*

*i*=1

## Detail of construction

User’s task in e-petition system is make a zero-knowledge proof of *π*.

The ﬁrst step is user need to prove a secret vector *v* is spanned the public matrix **M** which we talked about in span program earlier. The purpose of this step is shown how to hide which subset of user’s attributes has used to satisfy the modiﬁed predicate Ω*/* [6]. The Proof process is shown as follows,

***谷* M**=[1,0,...,0],which also can be written the following form,

．

*x* Commitment of *v*

*\_*Ω*卜\_*

*i*=1

*vi***M***ij*=

,1 *j* = 1

．0 2 *< j < θ*

*βvi , βwi , wi -* Z*p, i* = 1*...α*

*vA* = *gvi x kβwi*

*i* 1 3

*βvi βwi*

*x* Proof of statement

*vi* = *g*1 *x k*3

*α α*

3

3

*Vj e* [1*,θ*]: Λ*j*=

*kwi.Mij* , *λj* = (*kwi.Mij* )*βti*

*i*=1

*i*=1

The second step is proof whether the signature can be veriﬁed base on the veriﬁcation key.

In Digital Signature, a veriﬁcation function can be used.

*e*(*σvi , X, Y r x gai\_\_id*) = *e*(*g*1*, g*2) *x e*(*hsk, g*2)

*ai*

2

1

According to the BB and BB│ scheme, the signature can be described as follows,

,

*σai*

1

．(*g*1*/xi*+*yiri*+*apsdo pseudo . attribute*

1

．(*g*1 *x hsk*)1*/xi*+*yiri*+*ai\_\_id attribute*

The public key of *ai* is *Xi* = *gxi* and *Yi* = *gyi* . We need to replace the zero-element *vi* in secret vector *v* to

2

2

the random number. The purpose is hide subset of the attributes which satisﬁed the predicate Ω.

# References

1. Hansard Society. (2010) ‘Audit of Political Engagement 7’.
2. Caygill, Thomas, and Anne-Marie Griﬃths. ”Parliament and petitions.” Exploring parliament (2018): 322-332.
3. Li, Jin, et al. ”Attribute-based signature and its applications.” Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security. 2010.
4. Okamoto T, Takashima K. Decentralized attribute-based signatures. InInternational Workshop on Public Key Cryptography 2013 Feb 26 (pp. 125-142). Springer, Berlin, Heidelberg.
5. El Kaafarani A, Ghadaﬁ E, Khader D. Decentralized traceable attribute-based signatures. InCryptogra- phers’ Track at the RSA Conference 2014 Feb 25 (pp. 327-348). Springer, Cham.
6. El Kaafarani, Ali, et al. ”Attribute-based signatures with user-controlled linkability.” International Con- ference on Cryptology and Network Security. Springer, Cham, 2014.
7. Li, Yannan, et al. ”A blockchain-based self-tallying voting protocol in decentralized IoT.” IEEE Transac- tions on Dependable and Secure Computing (2020).
8. Galbraith, Steven D., Kenneth G. Paterson, and Nigel P. Smart. ”Pairings for cryptographers.” Discrete Applied Mathematics 156.16 (2008): 3113-3121.
9. Zhao, Zhichao, and T-H. Hubert Chan. ”How to vote privately using bitcoin.” International Conference on Information and Communications Security. Springer, Cham, 2015.
10. Bistarelli, Stefano, et al. ”An end-to-end voting-system based on bitcoin.” Proceedings of the Symposium on Applied Computing. 2017.
11. Li, Yannan, et al. ”Traceable monero: Anonymous cryptocurrency with enhanced accountability.” IEEE Transactions on Dependable and Secure Computing (2019).
12. Takabatake, Yu, Daisuke Kotani, and Yasuo Okabe. ”An anonymous distributed electronic voting system using Zerocoin.” IEICE Technical Report 116.282 (2016): 127-131.
13. McCorry, Patrick, Siamak F. Shahandashti, and Feng Hao. ”A smart contract for boardroom voting with maximum voter privacy.” International Conference on Financial Cryptography and Data Security. Springer, Cham, 2017.
14. A. goverment, ”ELECTRONIC PETITIONING TO THE HOUSE OF REPRESENTATIVES Chapter 4 (page 41-50),” Australia, 2009.
15. Wahab, Jad. ”Coconut E-Petition Implementation.” arXiv preprint arXiv:1809.10956 (2018).
16. Okamoto, Tatsuaki, and Katsuyuki Takashima. ”Decentralized attribute-based signatures.” International Workshop on Public Key Cryptography. Springer, Berlin, Heidelberg, 2013.