On-block certs: blockchain-based lightweight digital certificates

Nícolas F. R. A. Prado[1], **Marco A. A. Henriques**[1]

[1] Department of Computer Engineering and Industrial Automation (DCA)
School of Electrical and Computer Engineering (FEEC)
University of Campinas (Unicamp)
13.083-852 – Campinas, SP – Brazil

{n185142,marco}@dca.fee.unicamp.br

Abstract. *Identity management applications need some kind of tamper resistance, characteristic which we can borrow from a blockchain. This makes it possible to create, use and check certificates that are light and tightly coupled with a blockchain. To better understand this concept, we developed a proof-of-concept for a system that creates and retrieves the new lightweight digital certificates from the blockchain. The system can simplify both the certificate management for a Certification Authority and the checking of certificate revocation status by users. Using it, we can sign and verify digital signatures on documents, interfacing with the blockchain to retrieve a given certificate and check for its validity.*

1. Introduction

The everyday usage of the internet for activities that require security, such as e-commerce and online banking, made digital certificates very common. To make the communication private and tamper resistant, the messages are encrypted and signed, using the sender's private key. But in order to guarantee that this signing was made by the sender, certificates are used, binding a public key to its owner [Paar and Pelzl 2010].

Although this system works well, it can be cumbersome. Presently, certificates of most Certification Authorities (CAs) are already baked-in in the internet browsers, and at the moment of access of a website, its certificate is received and checked for its authenticity against the CA's certificate. Although that could be enough for most cases, the correct procedure involves visiting the CA's website to check if the certificate is in the Certificate Revocation List. An alternative to this method is to store the certificates and their status on a blockchain, which is the one adopted in this project.

A blockchain [Antonopoulos 2017] is a structure that provides tamper resistance to the data stored inside of it, by being distributed on a peer-to-peer network and utilizing consensus mechanisms.

With the usage of a blockchain, the link between public key and transaction information is a natural consequence of the blockchain's structure. Moreover, the blockhain would not only store the certificate and its status in the same structure but also make it lighter, while also providing great accessibility, since it's distributed.

Ethereum's blockchain [Wood 2014] was the one chosen for this system due to its great popularity, meaning it is more adopted and therefore more secure, and to its

capacity to store arbitrary data as a part of each transaction, which is required for creating the certificates.

The purpose of this project is to create a proof-of-concept of a system which can store and retrieve certificates using the blockchain as the underlying storage structure in order to obtain a tamper-proof system that integrates certificates and their revocation status.

Besides making it possible to create and use smaller certificates, the proposed system takes advantage of having the certificates available in thousands of blockchain sites. This feature improves both security and availability of digital certificates for CA's clients.

2. Funcionality

The basic building block of the system is the on-block certificate, which is a transaction from a CA's address to the user's address, with the user's informations in the data field as well as additional information. Due to this structure, both the signature and the public key, which are mandatory on certificates, can be ommited from the data field of the transaction, since the first is already present as the transaction's signature, and the second can be derived from the receiver's address (by using his or her signature). Due to this, the on-block cert can be lighter than common certificates.

Being a full proof-of-concept, this system provides an interface in which it is possible not only to create and get certificates using the blockchain, but also to perform the full process: From generating the keys all the way to verifying the signature of a message with a key that is guaranteed by an on-block certificate. Therefore, the possible use cases are the following:

1. Key pair generation
2. On-block cert generation
3. On-block cert query
4. Certificate revocation
5. Message signing
6. Message signature confirmation

It's also important to note that both on-block certificate generation as well as revocation, can be done only by the CA, since their signature is required for creating the transaction, while the others can be done by the user.

As an example of the usage of this system, consider a user named Bob, who wants to send a signed message to another user named Alice. The procedure to accomplish this is the following:

1. Bob uses module 1 to generate a new cryptographic key pair, which is used to derive Bob's Ethereum address from the public key.
2. Bob asks the CA to create him an on-block cert, gives it his information along with his Ethereum's address and sends a signed transaction to the CA's Ethereum address to prove that the given address is truly his.
3. The CA uses module 2 to create the on-block cert.
4. Bob signs a message using module 5, and sends the signed message to Alice.

5. Alice uses module 6 to check that the signature is valid and therefore the message is legitimate.

In case Bob's private key is compromised, he can inform this fact to the CA, who will use module 4 to update the status of the on-block cert, making it invalid for future checks by module 6.

3. Implementation

The proof-of-concept system was developed in the Python programming language. It has a command line interface and provides a command for each of the use cases presented. Following is a brief description of each command:

- `gen-keys` — generates a cryptographic key pair and saves the private key in an encrypted keystore file with a given password
- `issue` — creates a transaction with an on-block cert on the blockchain with the given user information
- `get-cert` — retrieves an on-block cert along with its status from the blockchain
- `sign` — generates a .sig file containing the signature of the given file
- `check-sig` — checks if a file's signature is valid and also if the on-block cert of the address that signed the file is valid
- `revoke-cert` — updates the status of an on-block cert by creating a new transaction with the revoke operation for the given address

Besides commands, the system also uses a configuration file to provide configuration data like the blockchain to use (testnet or otherwise), the keystore file location and the CA addresses to trust while determining the state of an on-block cert. It also provides information about the CA, which is written in each on-block cert emitted.

The usage of one of the commands (`check-sig`) of the program is shown in Listing 1, and an on-block cert is shown in Figure 1 through a blockchain explorer's site. The program relies heavily on pyethereum [pye], the reference implementation of Ethereum in Python, to interact with Ethereum's blockchain. The code for the project can be found on the following site `https://github.com/regras/cert-on-block`.

Listing 1. Command and output for checking a signature on the developed program (check-sig command)

```
>>./main.py check-sig --file test/sample\_file.doc --sig-file test/
   sample\_file.sig --config config.conf
2018-07-31 16:50:04,096 - INFO - Starting new HTTP connection (1):
   rinkeby.etherscan.io
2018-07-31 16:50:04,512 - INFO - The signature is valid.
2018-07-31 16:50:04,512 - INFO - Certificate status: valid
2018-07-31 16:50:04,512 - INFO - Certificate data: {'Issuer': {'C': '
   Brazil', 'ST': 'Sao Paulo', 'O': 'Unicamp', 'OU': 'FEEC', 'CN': '
   ReGrAS CA'}, 'Validity': {'Not Before': '2018-07-31 16:23:06', 'Not
    After': '2020-10-10 20:00:00'}, 'Subject': {'C': 'Brazil', 'ST': '
   Sao Paulo', 'L': 'Campinas', 'O': 'Unicamp', 'OU': 'FEEC', 'CN': '
   Nicolas F R A Prado', 'emailAddress': 'n185142@dac.unicamp.br'}}
```

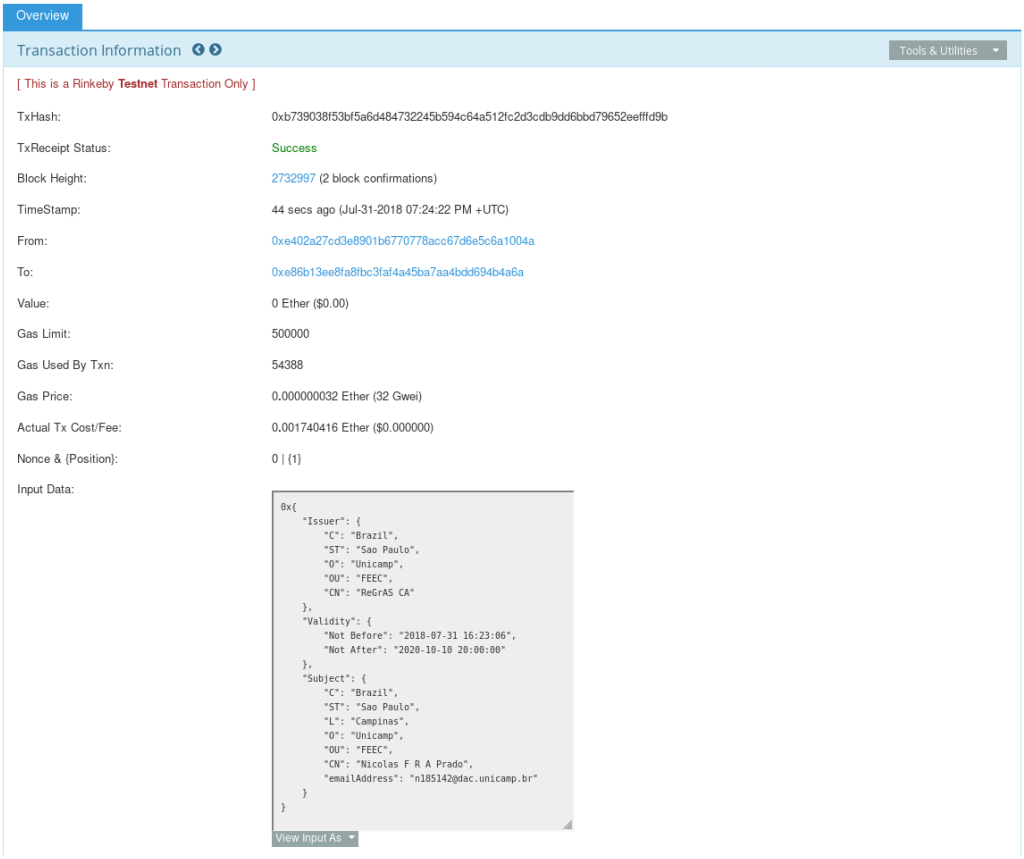

Figure 1. Ethereum transaction containing a sample on-block cert

4. Conclusion

The developed system leverages blockchain's characteristics to make it possible to use digital certificates that are lighter and tightly tied to transactions on the blockchain. Due to the distributed nature of the blockchain, this system makes it more reliable to validate a certificate, since it doesn't rely on a single site from the CA.

5. Future work

As future work, a plugin could be created to integrate the developed system with an e-mail client or web browser, making it easier to use and more attractive to people not acquainted to the command line interface.

References

Pyethereum. `https://github.com/ethereum/pyethereum`. (accessed on 06/09/2018).

Antonopoulos, A. M. (2017). *Mastering Bitcoin: unlocking digital cryptocurrencies*. "O'Reilly Media, Inc", 2nd edition.

Paar, C. and Pelzl, J. (2010). *Understanding Cryptography: A Textbook for Students and Practitioners*. Springer.

Wood, G. (2014). Ethereum: A secure decentralised generalized transaction ledger. `http://gavwood.com/Paper.pdf`. (accessed on 06/09/2018).