Nicolas BERNARD

nico.bernard21@gmail.com

24th of April 2017

Contents

INTRODUCTION

A large amount of data are exchanged on the Internet everyday. These data might be confidential (credit card numbers for example) and their transfer must be secure.

So, the majority of the websites use SSL certificates to authenticate themselves to their clients and exchange encrypted data. We are going to analyse in this document how Sslchain allows us to manage certificates using a blockchain.

1 USUAL MANAGEMENT OF SSL CERTIFICATES

WHAT IS A SSL CERTIFICATE?

A SSL certificate allows us to associate a domain name, the identity of its owner and a public key.
Such a certificate have two main goals:

- Identify the server which send the web page to the client: is this really my bank webpage I am trying to access?
- Encrypted data exchanges between the client and the server thanks to the public key: my bank details aren't send on a clear channel.

HOW CAN I GET A SSL CERTIFICATE CURRENTLY?

Currently, the domain owner create his certificate giving his identity, the one of his website and generating a private key / public key couple. Then his certificate has to be signed by a certification authority (CA). This authority will check the informations given by the owner and sign the certificate with its private key. This operation costs money.

HOW TO VERIFY THE AUTHENTICITY OF A CERTIFICATE?

When a client asks for a web page, he receives the certificate from the server. Then, his web browser verify that the certification authority which signed the certificate is trustworthy. If it is, the client and the server can exchange encrypted data using the private / public key couple. Finally, they decide a common symmetric key and use the SSL protocol to secure their exchanges.

ISSUE

This system works well but it raise a main issue, it includes a middle man: the certification authority.
How could we avoid to require a trusted third party?

Sslchain answers this need proposing a new way to manage SSL certificates using a blockchain. The use of a blockchain allows a secure, decentralized, public way to store the certificates.

2 HOW DOES IT WORK?

TECHNOLOGIES USED

Sslchain is based on the Ethereum blockchain and also uses the web3 API to interact with Ethereum. Sslchain owns a smart-contract (*cf Sslchain.sol*) composed of a structure to store the certificates and a function to add new ones.

CREATION OF A CERTIFICATE

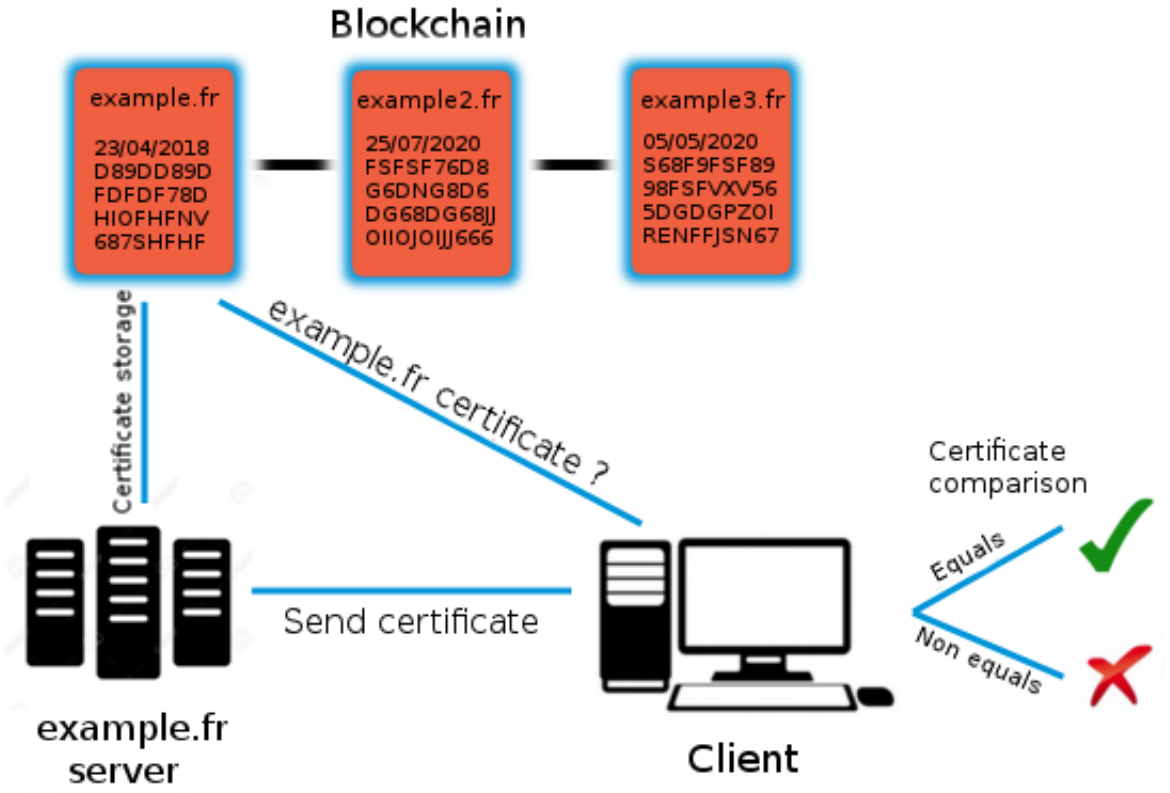
To create a SSL certificate, a webmaster needs to generate it using a tool such as *openssl*. He has to provide his personal details and then the program generate the certificate and the private / public key couple. Then he self-signs the certificate using his private key and go on Sslchain website (*cf* 4). He can add his certificate in the blockchain throw Sslchain website.

These informations are stored in the blockchain using the smart-contract deployed on Ethereum. The smart-contract first check that there are no other certificate for the same domain name or that they expired. So, for one domain name, there is only one valid certificate in the blockchain. The webmaster can then configure his server to send the certificate to the clients.

VERIFICATION OF A CERTIFICATE

In order to verify that a certificate sent by a server is valid, Sslchain has a *NodeJs* script (in the future this script will be automatically executed by the web browser). This program compares the certificate received from the server and the one present in the blockchain for the same domain name. If they match, the certificate is validated and the encrypted exchanges can go on.

WORKING DIAGRAM



3 REQUESTED TOOLS

NB: All the following commands have been tested on Ubuntu 16.04 using Google Chrome only.

ETHEREUM CONNEXION

In order to install all the components required to connect Ethereum blockchain, please execute the following commands.

```
# sudo apt-get install software-properties-common
# sudo add-apt-repository -y ppa:ethereum/ethereum
# sudo apt-get update
# sudo apt-get install ethereum
```

You can now connect Ethereum test blockchain called *test-net* using the command:

```
# geth --testnet --fast --rpc --rpcapi db,eth,net,web3,personal --cache=1024
--rpcport 8545 --rpcaddr 127.0.0.1 --rpccorsdomain "*" console
```

NB: The synchronisation with Ethereum might take a long time and need to be done.

NODEJS MODULE

To run the website and create certificates, go to *sslchain_site* repertory and install the following modules:

```
# sudo npm install web3
# sudo npm install truffle
# sudo npm install webpack
# sudo npm install copy-webpack-plugin
```

To use the verification script, go into *sslchain_script* and install these modules:

```
# sudo npm install web3
# sudo npm install openssl-cert-tools
```

4 TRY SSLCHAIN

First start the connexion to the blockchain:

```
# geth --testnet --fast --rpc --rpcapi db,eth,net,web3,personal --cache=1024
--rpcport 8545 --rpcaddr 127.0.0.1 --rpccorsdomain "*" console
```

CREATE A CERTIFICATE

Unlock your Ethereum account:

```
web3.personal.unlockAccount(<address>, <password>, 15000)
```

Then in the repository *sslchain_site*, execute the command *npm run dev* and go to `http://localhost:8080` to access Sslchain website. In the menu *Create a certificate* you can fulfill the form to save your certificate in the blockchain. Once your certificate will be mined, it will appear in *Overview* section.

NB: Adding a certificate to the blockchain might take few minutes.

VERIFY A CERTIFICATE

Run the command:

```
# node sslchain.js my_domain_name.com
```

If you previously add your certificate for your domain name, the program will get its certificate and check if the one you entered in the blockchain is equal.

CONCLUSION

This first version of Sslchain allows us to store SSL certificates publicly and safely using the blockchain technology. We succeeded to avoid a trusted third party such as a certification authority.

The next step of this project will be to deploy the smart contract on the real Ethereum blockchain and not only on the *test-net*. Then, get Sslchain website online with an interface to pay with classic money or Ethers directly. The last thing will be to integrate the verification program in the web browsers so that the clients get the same experience as now.