# Nexus blockchain PKI

## Statement of work and architecture document

Version: 0.3.1
Release Date: 2017-05-17

## Introduction

This paper describes uses cases, requirements and architecture and for a proof of concept for a Nexus blockchain CA. The intention is to showcase blockchain PKI and allow a potential customer to register to a blockchain CA to download a blockchain based certificate. The customer should then be able to validate this certificate on his or her own by using our GitHub hosted open source software that connects to local Ethereum node.
The validation part is transparent to the user and will be accompanied by a white paper that explains the concept and its merits. The issued certificate will be what we call a hybrid certificate. I.e. it can be validated through traditional PKI as well as thru blockchain record and still works with all existing systems that work with X.509 certificates.
The user has normal asymmetric keypair and the accompanying certificate is signed with PKI signature as expected and blockchain holds the indelible record of the certificate which serves as blockchain signature. Usage of the keypair for authentication/signing/encryption will be as always but the validation of the certificate can be done through the blockchain.

The goal behind the demonstration is not only to showcase blockchain PKI but also investigate security, performance, pros/cons etc. from an academic perspective.
The expected outcome from the research initiative is:

- A working demonstration as described in this paper
- A white paper describing the technology behind the blockchain PKI and highlighting aspects such as security, intellectual property, performance, scalability etc.

## Background

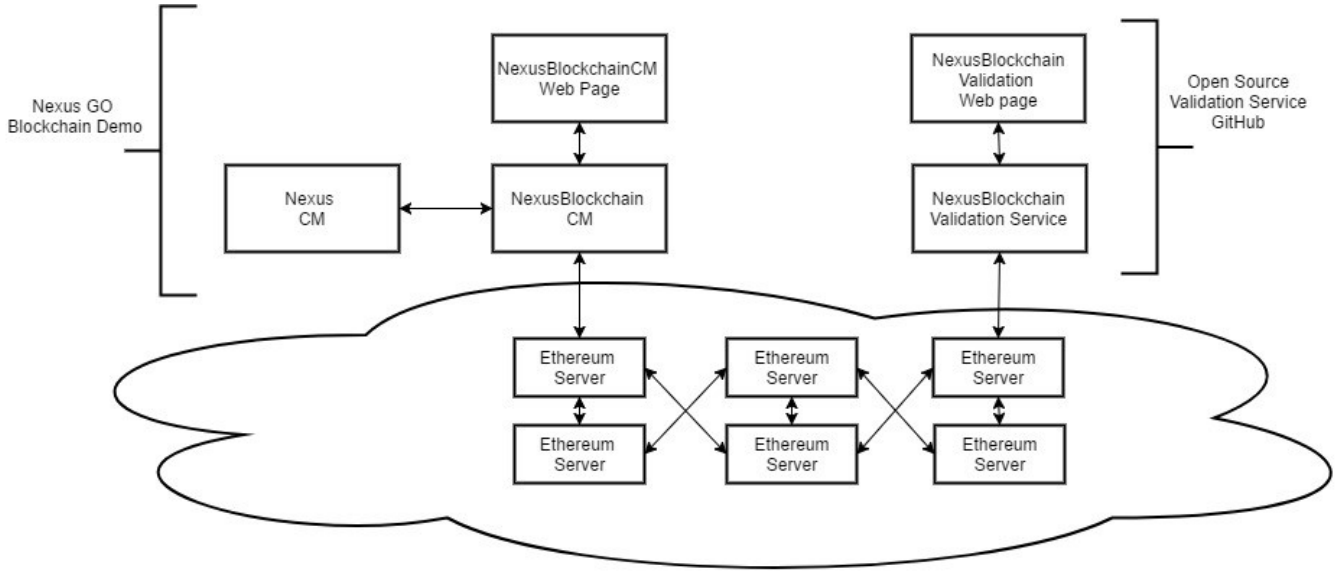
Blockchain technology is an emerging technology that has attracted some really large investments and is expected to radically improve banking, identity management, supply-chain and other transaction networks, giving them new opportunities for innovation and growth while reducing cost and risk. Records on the distributed ledger can store and show virtually anything of value: your identity, a will, a deed, a title, a license, intellectual property and almost any type of financial instrument, everything in a public way but easily also encrypted, so that only appropriate parties can decrypt and verify the content.
Nexus research will initially focus on Identity management, data signatures and time stamping. Nexus has over the years been successful in delivering traditional PKI based solutions to these problems. However dependency on a central authority presents serious limitations in large-scale scenarios. A hierarchical nature with a single root of trust and the risk of root keys compromise also poses a limitation for large rollouts. Blockchain on the other hand is a distributed P2P network and does not suffer from these limitations.
There are still many challenges to be resolved before blockchain becomes a mature technology. There are several competing blockchain based systems where Bitcoin is the original one and by far the most widespread. It is entirely focused on financial transactions and it is at the moment suffering from some very deep architectural and organizational flaws, that could not be foreseen, but which hinder its further development.
Another universal challenge is the privacy aspect and the trade-off regarding public vs private blockchain. For example to make public blockchain transaction without revealing any details of it to

public, both parties identities and content of the transaction should be opaque to everyone else, yet accessible to both parties and this should be a natural part of the system, without unnecessary overhead or extra key management. This requires sophisticated identity management, which is the main focus of Nexus research and development efforts.
The eIDAS legislations will drive large-scale identity solution rollouts where blockchain will be a good fit. The good news is that eIDAS regulations are technology neutral. *…This Regulation should be technology neutral. The legal effects it grants should be achievable by any technical means provided that the requirements of this Regulation are met.* In other words, eIDAS will drive the blockchain migration forward.

## Architectural overview

The proposed PoC setup consists of 2 web pages and 2 services;

- NexusBlockchainCM Web page.
  A simple web page where the user can enter personal info and request a blockchain based certificate. Certificates can be revoked from this page as well.
- NexusBlockchainCM service.
  A rest based service to issue and revoke blockchain based certificates. The service returns a PKCS#12 (.p12) file
- NexusBlockchainValidation web page
  A web page where the user can paste or upload a certificate and check its validity and and revocation status.
- NexusBlockchainValidation service.
  A REST based service for validation of blockchain based certificates.



# Use cases

## Basic use cases

**There are four basic use cases defined:**

1. Issue blockchain based CA certificate
2. Issue a blockchain based (user) certificate
3. Validate blockchain based certificate
4. Revoke blockchain based certificate

**Actors:**

- Certificate requestor a.k.a. user.
- CA Admin, this is a variant of user, one who is asking for CA certificate to be issued for him.
- Issuer, this is the owner and admin of CA which is issuing the certificate for user or CA Admin.
- Validator. E.g. the person or system that wants to validate a certificate.

## Use case realization

### Use case #1 Issue blockchain based CA certificate

1. The CA admin generates public/private keypair used to control the smart contact in Ethereum.
2. The CA admin creates a CA contract with a specified unique contract id in the blockchain and populates owner id.
3. The CA admin generates the CA certificate in Nexus CM including the blockchain extension with contractId from (2)., contractId of the issuer CA (or zero for selfsigned) and HashAlgoId in the certificate.
4. CA admin puts the CA certificate into his contract in binary form into the CACertificate field.
5. The CA admin submits the CA certificate to the issuer.
6. Issuer puts the hash of the issued CA hash into his issuedCertificateHashes field in the block chain contract.

**API**

N/A This will be a manual configuration in the PoC while setting up the genera certificate chain structure of the service.

### Use case #2 Issue a blockchain based certificate

1. User enters certificate user info such as subject name, organization etc. on the portal web page including password for the generated p12 file.
2. The portal sends a certificate generation request to NexusBlockchainCM service with the above parameters.
3. The NexusBlockchainCM service prepares a .p12 certificate request to Nexus CM with help of the Nexus CM SDK. The required blockchain attributes in the cert are defined per issuing CA in Nexus CM and is injected by Nexus CM.
4. Nexus CM generates asymmetric keypair, generates the certificate and wraps everything in a .p12 file with the password set.
5. The .p12 is returned to the NexusBlockchainCM
6. NexusBlockchainCM extracts the certificate from the .p12 file and calculates the hash of the certificate.
7. NexusBlockchainCM updates the smart contract and add the hash of the certificate into issuedCertificateHashes array.
8. The .p12 file is returned to the user.

**API**

byte[] issueBlockchainCertificate(List<Cert details>, pwd)
returns binary .p12 file

**Use case #3 Validate a Blockchain issued certificate**

1. The certificate to be validated is uploaded to or pasted in the NexusBlockchainValidation web page
2. The web page converts the cert to binary(p12) format and sends a validation request to the NexusBlockchainValidation service
3. The NexusBlockchainValidation service parses the X.509 (DER/ASN.1) encoded certificate and retrieves the blockchain contractId & HashAlgoId encoded inside the certificate structure See section certificate encoding for more details
4. The NexusBlockchainValidation service calculates the hash of the certificate
5. The service checks that the hash of the certificate is in the list of issued certificates in the blockchain contract.
6. The service checks that the hash of the certificate is not in the list of revoked certificates in the blockchain contract.
7. The service retrieves the CA certificate from the blockchain contract.
8. This process is repeated for each CA in the trusted chain. Repeat from step 3 with the CA certificate until the rootCA is found or the trusted certificate is met.
9. If certificates starting from the immediate one are all validated up to the one, which is trusted, then this is a valid certificate.

**API**

validateBlockChainCertificate(byte[] cert, Boolean verifyChain, hash[] trustedCertificates)
Returns
Validated, // True if all certs in the chain are validated
List<boolean validated, byte[] certificate, int ConfidenceLevel> // For each certificate in the chain
ConfidenceLevel is how much mining power has gone into the verification, this is essentially equivalent to the age of the certificate.

**Use case #4 Revoke a blockchain based certificate**

1. The certificate to be validated is uploaded to or pasted in the NexusBlockchainCM web page
2. The web page converts the cert to byte format and sends a revocation request to the NexusBlockchainCM service
3. The NexusBlockchainCM service hashes the X.509 certificate and stores the hash in the list of issued certificates in the blockchain contract.
4. The NexusBlockchainCM service revokes the certificate for traditional PKI in Nexus CM
5. The service returns if the operation was successful or not

**API**

Boolean revokeBlockChainCertificate(byte[] certificate)

## Block chain contract & data model

The blockchain contract includes the following data fields
+owner
+CAcertificate
+issuedCertificateHashes[]
+revokedCertificateHashes[]

End user certificates are only stored as hashes due to privacy reasons but also to save space. The CA certificates are stored in the blockchain so a full validation of the chain of trust can be done without having the immediate certificates beforehand.

Todo: Describe contract in more detail witth owner data, contract ID and validation of contract code.

## Certificate encoding

We propose that blockchain based certificates should be hybrid certificates and both be verifiable using traditional PKI as well as verifiable using blockchain technology.
In blockchain based PKI the hash of the certificate is stored in the issuing CA smart contract. The blockchain address of this contract and the algorithm used to calculate the hash are encoded into the certificate as certificate extension parameters.
For now we propose that these extension OIDs are stored somewhere under the Nexus Oid tree {iso(1) member-body(2) se(752) 115} under a leaf called 'blockchain-PKI'
E.g. {iso(1) member-body(2) se(752) Nexus(115) ? ? ? blockchain-PKI(?) HashAlgo(0)} is the OID of the Hash algo.

| Index | Type | Name | Description |
|---|---|---|---|
| 00 | OID | HashAlgo | Value defined by nist see {joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3) nistAlgorithm(4) hashAlgs(2)} |
| 01 | OCTET STRING | IssuerCaContractIdentifier[20] | Contract address in Ethereum for issuing CA |
| 02 | OCTET STRING | CaContractIdentifier[20] | Contract address in Ethereum if the cert is a CA cert. [Optional] |
| 03 | PrintableString | BlockchainName | E.g. 'Ethereum-Public', 'Ethereum-Test' |

## Blockchain white paper

The accompanying white paper should describe the technology behind the proposal and analyze relevant aspects of blockchain PKI such as security, IP, performance, scalability, choice of actual variant etc.
Questions to be answered are for an example (but not limited to)

- How much ether does it cost to register a standard certificate? Bitcoin? Euro?
- How is the confidence level defined for a registered certificate? How many blocks needs to be mined before it can be valid to be secure.
- How quick is the data replicated to another node? In the same country? Different countries in Europe? To USA? How quick is it before it can be securely vetted?
- How scalable is the solution. E.g. what is the validation performance for 1000 certs, 100 000, 1 000 000, 10 000 000 certificates
- What requirements is needed to set up a validation service including an Ethereum replica?
- How many Gigabytes will be stored in a local copy of Ethereum?
- Compare private and public Ethereum networks
- Compare and explain why Ethereum was chosen
- Discuss privacy aspects of blockchain PKI
- What IP is in the field? List applicable patents in the field?
- Discuss security threats. Different attacks on Ethereum and generic blockchain
- Etc…….