X509Cloud - Framework for a Ubiquitous PKI

Hitesh Tewari, Arthur Hughes, Stefan Weber, Tomas Barry
Trinity College Dublin, Ireland

***Abstract*—The SSL protocol has been widely used for verifying digital identities and to secure Internet traffic since the early days of the web. Although X.509 certificates have been in existence for more than two decades, individual user uptake has been low due to the high cost of issuance and maintenance of such certs. This has led to a situation whereby users are able to verify the identity of an organization or e-commerce retailer via their digital certificate, but organizations have to rely on weak username and password combinations to verify the identity of customers registered with their service. We propose the X509Cloud framework which enables organizations to issue certificates to their users at zero cost, and allows them to securely store and disseminate client certificates using the Bitcoin inspired blockchain protocol. This in turn will enable organizations and individuals to authenticate and to securely communicate with other users on the Internet.**

***Index Terms*—X.509, PKI, Blockchain, PoW, OTP, XMPP.**

I. Introduction

The secure socket layer (SSL) [1] protocol and its successor the transport layer security (TLS) [2] protocol have been the security protocols employed for verifying identities and securing traffic across open networks such as the Internet since the early days of the world wide web. One of the main building blocks of the SSL[1] protocol are X.509 certificates [3]. Although X.509 certificates have been in existence for more than two decades, individual user uptake has been low [4][5]. In particular, the majority of certificates issued today are for commercial organisations, as the cost associated with buying or renewing a certificate on an annual basis can be anywhere between $200 - $2000, depending on the type of identity verification that is required [6]. This cost has proven to be too prohibitive for individual users and thus the current low uptake rates of *client certificates*.

This in turn has led to the situation whereby end users (clients) are able to verify the identity of an organisation or e-commerce retailer (web server) via their X.509 certificate, whose *root certificate* is issued by a trusted certification authority (CA) and is pre-installed in the client's web browser certificate store [7]. On the other hand, an organization or merchant has to rely on username and password combinations to verify the identity of customers registered with their website or service. We know from the literature that there are numerous problems associated with using passwords for authentication - such as the use of weak passwords, reuse of the same password across multiple domains [8] etc.

In this paper we propose a mechanism to increase the uptake of X.509 certs by making use of a Bitcoin inspired blockchain called MutliChain [9] to *securely store and disseminate client certificates* to organisations and other users who wish to use them to authenticate, and securely communicate with others on the Internet. Our proposal *eliminates the need for users to have to purchase expensive certs from commercial organisations* [10], and *to have to go through cumbersome and time consuming identity verification procedures*. Instead we use the collective power of the Internet to verify and lock client certificates into a global blockchain - which we refer to as the X509Cloud service. In effect, our proposal empowers organisations to act as a *subordinate or intermediate* CA [11] for their employees and other users associated with the organisation. This will allow others to use a *cloud based certificate infrastructure to authenticate end users to websites, financial organisations, e-tailers etc.* This in turn could eliminate the need for passwords, and replace them with a truly ubiquitous public-key infrastructure (PKI) [12]. Our approach also *eliminates the need for certificate revocations lists (CRLs) to be updated and distributed in the event that a certificate is lost or stolen*.

The rest of this paper is organised as follows. We begin by directing the readers attention to some related blockchain initiatives. We then provide an overview of the X.509 PKI which is used to bind an organisation or individual's public key material to their chosen identity. We briefly describe some of the salient points of the Bitcoin protocol, namely the blockchain technology and the associated proof-of-work (PoW) algorithm, which is employed to lock-in Bitcoin transactions into the blockchain. We will also discuss the MultiChain protocol, and how it differs from the Bitcoin protocol (e.g. mining diversity). Next we concentrate our efforts on showing the reader how we can make use of the blockchain technology to quickly verify, store and disseminate client certificates using the X509Cloud framework. Finally, we present a well known use case which can easily leverage the X509Cloud framework.

II. Related Work

The blockchain concept has been proposed as the basis for a number of approaches to online currencies and security mechanisms. Amongst these systems, a number of certificate mechanisms have been suggested that are closely related to the X509Cloud framework, but have significant differences.

Namecoin [13] defines a blockchain-based replacement for DNS that allows the distributed registration and retrieval of names and associated IP addresses. It provides functionality to register names as well as additional information such as certificates and physical addresses in a blockchain. This codebase has been used by projects such as CertCoin [14], NameID [15] and DNSChain [16] to provide blockchain-based

[1]For clarity we will make use of the term SSL throughout the rest of this paper when referring to SSL/TLS. Also we will interchangeably use the phrases certificate or cert to refer to a X.509 certificate.

certification of names. In contrast to the X509Cloud service, Namecoin relies on the generic definition of a blockchain that incorporates the same strong requirements for proof-of-work.

A number of projects have adopted the name BlockchainID. BlockchainID by Chris Ellis [17] proposes to create digital passports, strengthened through witnessing that can be documented with photography and video. The passport is then a combination of hashes for a personal image, a signature, signed with PGP signatures, hashed and timestamped. BlockchainID provides the description of the creation of a digital passport; however, in contrast to X509Cloud framework it does not include a digital certificate. This project is not to be confused with a project with the same name, Blockchain ID by Onename [18], which provides a replacement of Bitcoin wallet addresses through registration with Onename as the broker. BlockchainID by okTurtles is based on Namecoin and DNSChain and proposes a mechanism similar to the X509Cloud framework. It suggests the registration of encryption keys for the purpose of authentication at sites such as Facebook, Twitter, etc.

The various projects with the name BlockchainID are different from the X509Cloud framework in that they attempt to create an identity stored in a blockchain, whereas the X509Cloud framework focuses on the storage, retrieval and revocation of certificates.

Christopher Allens [19], [20] describes a proof-of-concept implementation that comes close to the idea of the X509Cloud framework in that it associates certificates with a blockchain, and uses the blockchain concept to provide a revocation mechanism. However, this approach focuses on self-signed certificates and does not involve third-party authorities for the creation and verification of certificates.

Guardtime [21] provides a proprietary service for key-less security infrastructure [22] based on the blockchain concept. The description mentions certificates and their replacement of keys; however, given that it is commercial solution, very little has been published about its realisation.

A number of researchers at Google have proposed Certificate Transparency [23]. This paper proposes a protocol that would enable anyone to audit a certificate authorities activity and stores issued certificates publicly in an append only database (called *logs* in the paper). The paper has a number of similarities to this one, however the targeted audience is different.

III. Background

A. X.509 Certificates and PKI

The primary task of a digital certificate is to confirm that the certificate's *public key* belongs to the certificate's *subject*. For example, a CA can digitally sign a special message (the certificate information) that contains the name of some user, say "Alice" and her public key. This must be done in such a way that anyone can verify the certificate was issued and signed by no one other than the CA. If the CA is trusted and it can be verified that Alice's certificate was issued by that CA, any receiver of Alice's certificate can trust Alice's public key contained within that certificate. The public key of the CA is widely known and is typically bundled with all major web browsers. Examples of some well known commercial CAs are Symantec, GoDaddy, Comodo etc.

A X.509 certificate consists of a number of certificate information ($CertInfo$) fields (see figure 1). The CA creates a hash of the *CertInfo* fields, and encrypts the result with its private key (which is kept under tight security) to create the *Certificate Signature* field ($Sig_{SK_{CA}}(CertInfo)$). Bob who has access to the CA's public key (PK_{CA}) and trusts it, can then use it to verify the CA signature on the cert and pass the *CertInfo* fields through the same hash algorithm (specified in the cert) to match the two hashes. If they are the same then Bob can trust Alice's public key and uses it to encrypt messages to her. The email address of the user can be part of the common name (CN) field of a certificate, and can be used to index the certificate in a global blockchain (see figure 1).

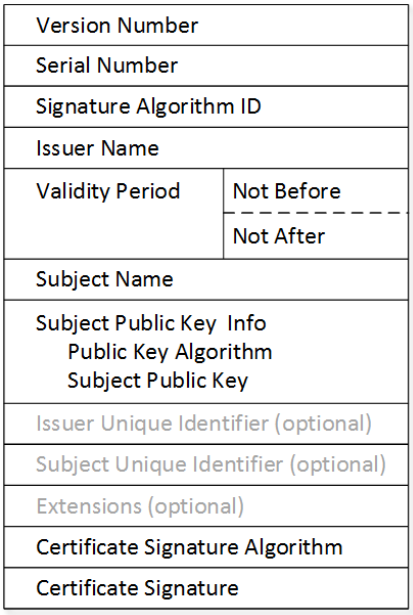

Fig. 1. X.509 Certificate Fields

B. Bitcoin and Blockchains

Bitcoin is a *decentralised*, *pseudo-anonymous* electronic cash scheme [24]. The Bitcoin protocol is decentralised in the sense that the participants collectively verify all of the transactions in the network. The security of Bitcoin is based around the assumptions that a majority of the nodes in the network are honest, and that the proof-of-work algorithm employed will deter any sybil attacks [25], as the amount of computational resources required will be greater than 50% of the network resources.

All Bitcoin transactions are stored in a public ledger known as the *blockchain*. The blockchain is a timestamped public ledger of all transactions that have ever been conducted on the Bitcoin network. A block in the blockchain consists of a block header and a number of associated individual transactions (which are readable by all parties within the Bitcoin network).

The first block in the chain is known as the *genesis block*, followed by blocks that have been created by *miners*. Miners in the Bitcoin network are nodes that keep a complete and up to date version of the blockchain, and compete to try to be the first to add the next valid block into the blockchain, so that they can earn some bitcoins and or transaction fees. Valid transactions are irreversibly locked into the blockchain using the *proof-of-work* algorithm by the miners, who work for a reward for solving the next PoW problem.

Each new block contains one or more new transactions that have been received by the miner within a specified time interval (e.g. every ten minutes). These are repeatedly hashed in pairs to form a *Merkle tree* [26]. The root of the Merkle tree along with the hash of the previous block is stored in the block header thereby chaining all the blocks together. This ensures that a transaction cannot be changed without modifying the block that records it and all following blocks. This property of the blockchain makes double spending of bitcoins difficult.

The Bitcoin protocol is very interesting from the point-of-view of a decentralised cryptocurrency, and there have been numerous Bitcoin inspired cryptocurrency proposals over the last few years [30]. It is however the Bitcoin inspired blockchain technology that has generated great excitement within the cryptographic community in recent times. A number of diverse protocols such as e-voting and e-contracts which make use of blockchain technology have been proposed [31][13][32].

C. MultiChain

The MultiChain protocol differs from the Bitcoin protocol in terms of the type of data that is intended to be stored in the blockchain. The Bitcoin blockchain is intended to store data related to transactions between two independent entities. While it is possible to store arbitrary data in the Bitcoin blockchain [33], it is highly discouraged and limited to a maximum of 80 bytes (therefore, storing larger amounts of data would require multiple *op_return* sections chained together). The MultiChain blockchain, however, was designed specifically to store arbitrary data that could be shared between non-trusting parties. The capacity of the raw data stored in a MultiChain transaction can be up to several megabytes.

MultiChain is a private blockchain, this means that it is not entirely decentralized, but a CA can allow other parties varying levels of access to the blockchain. These access levels include read access, mining ability, and access propagation, for example. The use of a private blockchain protocol enables X509Cloud implementations to limit exposure of an individuals certificate to services that want to integrate the X509Cloud infrastructure while still allowing services to verify the contents of the blockchain.

The X509Cloud infrastructure makes use of MultiChain data for storing certificates [34]. MultiChain data streams create an append only database which can be used by the X509Cloud infrastructure to store key value pairs for each entry where keys are the JID (Jabber Identifier), and the value is the X509 certificate for an individual in hexadecimal. MultiChain exposes a publish command which can be used to write the entry which will be seen by all nodes that have access to the MultiChain blockchain.

D. Mining Diversity

It is possible (however unlikely) for a single party to maliciously influence the Bitcoin network by taking over half of the nodes in the network. This is a key concern in private blockchains which typically have a much smaller number of active nodes. MultiChain attempts to alleviate this concern by implementing a mining diversity construct.

The mining diversity scheme determines how often white listed miners should attempt to mine blocks. The mining diversity is a value between 0 and 1 which dictates what proportion of the miners should have mined before a miner can add another block to the blockchain (if the mining diversity is 0.5, then a miner should not attempt to add to the blockchain before half of the existing miners have added to the blockchain since its last addition). Higher values for the mining diversity result in a blockchain that is more difficult for an individual actor to control. Higher values for the mining diversity also result in a slower confirmation time of blocks.

Mining is done in a round robin manner in blockchain networks that implement the mining diversity scheme (such as MultiChain). The mining diversity scheme has a degree of leniency to allow for the presence of malfunctioning nodes. The higher the mining diversity value, the more important this leniency can be.

IV. System Overview

The X509Cloud framework is motivated by the need to promote greater use of public key cryptography to enable individuals to send digitally signed and encrypted messages to each other. It further aims to enable users and organisations to mutually authenticate each other using public key certificates, thereby eliminating the need for usernames and passwords. As we highlighted in section III-A *the main cost and effort involved in issuing X.509 certificates is verifying the identity of users by the CA*. This usually requires a form of photographic identification such as passport or driving license, along with other identifying information to be presented in a secure manner to the CA. Finally, there is the issue of lost certificates and the subsequent dissemination of certificate revocation lists (CRLs).

We propose to address the above issues in the X509Cloud framework by using blockchain technology such that it becomes very easy for user identities to be verified by the issuing CA, so that they can issue certificates with very little administrative overhead. Our approach eliminates the need for CRLs to be updated and distributed in the event that a certificate is lost or stolen. Users that wish to communicate with others using public key cryptographic techniques can query the global X509Cloud service and obtain the user's public key certificate. Below we detail the steps in issuing and obtaining certificates using the X509Cloud system.

A. Certificate Generation and Issuance

Today most citizens in the developed world have electronic identities issued to them via their employers or by government agencies such as the Department of Motor Vehicles (DMV), Social Security Administration (SSA) in the United States etc. Take for example our own organisation - Trinity College Dublin (TCD) which is a university in Ireland. All the staff and students in the university have an account on the college servers that allows them access to various college services. The college databases store personal and academic details about each person associated with the university. A similar situation can be found in most other organisations worldwide. Also, most citizens today have a social security number issued to them by their country of residence, and also have authentication credentials to log into the associated servers to be able to update their details and request services e.g. Revenue Commissioners in Ireland.

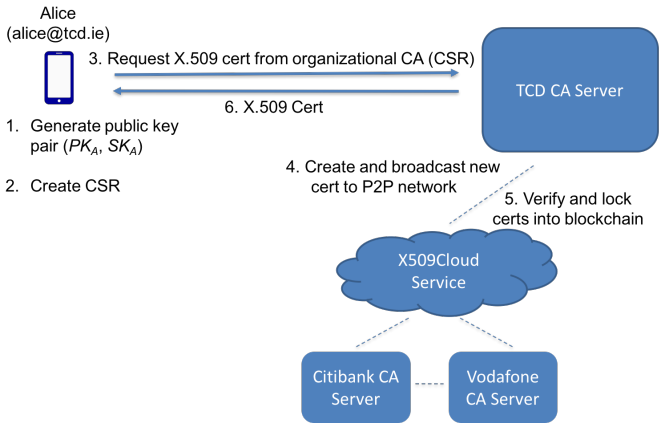

Fig. 2. Certificate Generation

We propose to use such organisations and other commercial entities as the issuing CA for client certificates. Users that wish to obtain a X.509 cert will use the organisation's mobile app to request a certificate. The app will securely generate the user's public key pair on their mobile device, log them into the CAs systems using their *existing organisational credentials*, and request a certificate. Going back to the example of TCD, users will login via the app using their college credentials. The app will prompt them for some personal details such as their name, email address etc., generate a certificate signing request (CSR) [35] and send it to the organisational CA.

The CSR contains information identifying the applicant (such as a distinguished name in the case of an X.509 certificate) which must be signed using the applicant's private key. The CSR also contains the public key chosen by the applicant. The signature by the requester prevents an entity from requesting a bogus certificate of someone else's public key. Since the TCD servers already have the required information about the user they will be able to verify that the fields in the CSR has been populated with the correct information. The organisational CA will then create a X.509 certificate using the required fields. It will digitally sign the CSR with its private key, and return a copy of the newly created cert to the user once its has been locked into the blockchain (see figure 2).

B. Locking Certificates into the Blockchain

Once a CA generates a certificate it will broadcast the cert to the global CA network. Each participating CA in the system will be able to verify the authenticity of the cert by downloading the public key certificate of the issuing CA, and using the public key material contained within to verify the digital signature on the user certificate in question. CAs will group certs received within a certain time frame (e.g. a ten minute window) into a transaction block, and will try to solve the PoW problem to lock-in the transactions in the block.

The first CA to solve the PoW problem will broadcast the result to the entire network. All other CAs will be easily able to verify the result and move onto solving the next transaction block. Solving the PoW problem is a *not-for-profit* activity which will be carried out by all participating CAs in the system. Their motivation for participating in the system will be the fact that certificates issued by them will be locked into a global blockchain which will become the de-facto *trust anchor* for locating X.509 client certificates.

C. Certificate Revocation

The Achilles' heel of the current X.509 based PKI is the fact that it is very hard to revoke existing certificates [3]. Once a certificate has been revoked the updated status of the certificate has to be communicated widely within the system. However experience has shown that distributing CRLs has proven to be a cumbersome and error prone task.

This problem is easily solved using the X509Cloud framework. If a user looses or accidentally exposes their private key they can use the X509Cloud app (see figure 2) to request a new cert from their CA. The X509Cloud app would make use of two-factor authentication when requesting or revoking a certificate. For example, if a user has their phone stolen, it is vital that the thief not be able to request a new certificate on behalf of the victim, locking them out of all of their profiles. By enforcing two factor authentication for requesting and revoking certificates a user can be kept safe from the above scenario with minimal extra effort required by a user.

The CA will go through the procedure outlined in section IV-A to generate a new cert. The new certificate will be broadcast to the network, and once verified will eventually be locked into the global blockchain. When a user queries the X509Cloud service for a user certificate they will get back the *latest cert* associated with user identifier *thereby invalidating all previous certs for a given user-id.*

If a user leaves an organisation or is fired then the organisational CA administrator can immediately generate a new cert on behalf on the employee and lock it into the X509Cloud service. This invalidates their old key pair and the user will no longer be able to access the organisational servers or send digitally signed messages on behalf of the organisation. It should be noted that this is a departure from current practice whereby commercial CAs will only revoke a certificate if it is reported to be lost etc.

D. Storage Costs and Scalability

In order for the X509Cloud framework to get widespread adoption we will need to ensure that the system scales, and has fast response times for queries made to the cloud service.

One of the first things that we need to quantify is the disk storage costs for each X.509 certificate that we store in the cloud. For a 4096-bit RSA key the size of the certificate is 990 bytes [36] - smaller if we use a 3072-bit NIST recommended RSA key, or if we make use of ECC keys [37]. On average we require approximately 1KB of storage for each cert. In order to store 100 million certs one would require 100 gigabytes of storage, 1 billion certs would require 1 TB of disk storage etc. Given current disk storage costs this does not seem like an onerous requirement for the X509Cloud service.

We also need to quantify the PoW computational costs required to lock a particular set of certificates into the blockchain. The maximum (easiest) target used by SHA256 mining devices is 0x00000000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF i.e. 32-bits which requires 2^{32} or ≈4.3 billion steps. A 40-bit target size would require ≈1 trillion steps. Modern day GPUs can do ≈1 billion hash calculations per second. Therefore it would require ≈17 minutes to solve a 40-bit PoW puzzle.

Lookup of certs needs to be fast, and by using the Multi-Chain protocol we can achieve an $O(n \log n)$ lookup time.

E. Cloud Service Registration

We make use of the extensible messaging and presence protocol (XMPP) [38] to push messages in real-time towards a user's registered mobile device in the X509Cloud framework. Specifically we push a one-time password (OTP) towards the user which is encrypted with the user's public key that is obtained from a X.509 cert stored in the blockchain and associated with the user's email address.

Fig. 3. XMPP JID Registration

Prior to any messages being exchanged between the X509Cloud XMPP server and client (on the user's mobile device), the user registers a JabberID (JID) with the XMPP server. In the case of the X509Cloud framework the user registers their email address where the "@" symbol before the domain name is replaced by literal "-at-". For example a TCD user with an email address alice@tcd.ie will register a JID of the form alice-at-tcd.ie@X509Cloud.net.

F. Use Case

Once we have a global PKI in place whereby each user has a public key certificate issued to it by a trusted CA and stored in the global X509Cloud, we have the ability to change the way we can authenticate users to websites and network services. With the necessary changes in place we can do away with traditional username and password combinations and make use of more secure mechanisms such as a challenge-response authentication protocol.

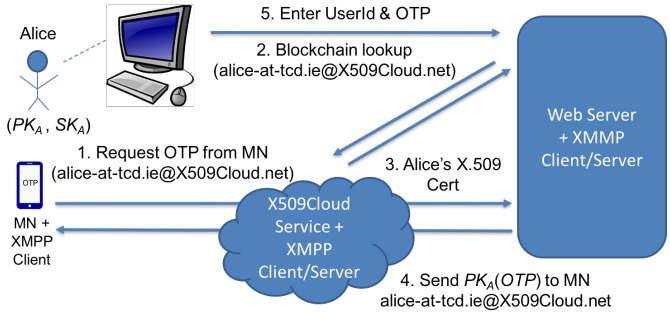

Fig. 4. Secure OTP Login

As shown in figure 4, a user wishing to authenticate themselves to a web service will request an OTP for the server (e.g. Google) they wish to authenticate to by communicating with their XMPP client on their mobile node (MN) via the X509 Cloud service. The web service's XMPP client in turn will use the supplied email address in the JID to locate the user's X.509 certificate in the global blockchain stored on the X509Cloud service. The web server will then create and locally store an OTP for the user. The web service's XMPP server will encrypt the OTP with the public key of the user (e.g. $PK_{Alice}(OTP)$) and forward it to the X509Cloud XMPP server, which in turn will push it in real-time to the user. The XMPP client on the user's mobile device will automatically decrypt the encrypted OTP (assuming the correct corresponding private key is stored on the mobile device), and display the result on the screen. The user will then input their user-id and OTP into the password field on the service provider's webpage as normal, and will be logged into the system.

V. Adversary Model

There are various attacks that must be considered when designing such a system. One of the most important is the malicious modification of the blockchain storing certificates. In many blockchain implementations, including Bitcoin, there exists the potential for a "51 percent attack". In short, if a single actor or party controls 51% of the network they would be able to manipulate network. In the X509Cloud system this attack is dealt with in two different ways. The mining diversity factor of MultiChain allows us to change the required control to perform such an attack. The second method to deal with this attack comes from the permissioned aspect of MultiChain. The certificate issuer can issue read permissions to everyone but only authorize writes from known parties.

A second attack that can be performed is the guessing of one-time passwords. If a malicious actor knows that a user has requested a one-time password and has not used it they could attempt to brute force the password. This attack can be dealt with by enforcing guess limits or time limits on the one-time

passwords, or using specific hash functions for verifying the password. We do not attempt to deal with this issue as it is better dealt with by the service being logged in to.

VI. Password Managers

Password managers have become a common way for users to have complex passwords without having to remember the passwords. Password managers such as LastPass, KeePass and others are available as desktop applications, browser extensions and mobile apps. People who use these products only need to remember one master password to access all of their other passwords.

X509Cloud can be considered a more user friendly experience to password managers. For users with limited technical knowledge password managers can be seen as an unnecessary burden on the log in process and add complexity to a simple procedure. For these users X509Cloud could be ideal. Services can now benefit from better identity verification without users having to greatly modify the log in procedure.

VII. Future Work

There is interesting work being done by Blockstack [40] in creating layers on top of existing blockchains to leverage the large number of nodes in a network. Blockstack have performed some experimentation as to the pros and cons of deploying new blockchains and leveraging existing ones. Their paper indicates that a public blockchain without a similar number of nodes to Bitcoin is susceptible to attack.

VIII. Conclusion

In this paper we have outlined a mechanism to easily issue and verify client certificates issued by organisations to their employees. The X509Cloud framework proposes a cloud based blockchain of public key certificates which can be queried to obtain a certified public key associated with an individual. The service allows users and organisations to mutually authenticate each other thereby eliminating the need to weak password authentication, and also allows for encrypted and digitally signed messages to be exchanged between them.

References

[1] A. O. Freier, P. Karlton, and P. C. Kocher, "The Secure Sockets Layer (SSL) Protocol Version 3.0," https://rfc-editor.org/rfc/rfc6101.txt, Oct. 2015.

[2] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2," http://www.ietf.org/rfc/rfc5246.txt, Internet Engineering Task Force, August 2008.

[3] D. Cooper, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," https://rfc-editor.org/rfc/rfc5280.txt, Oct. 2015.

[4] P. Eckersley and J. Burns, "Is the SSLiverse a Safe Place?" https://www.eff.org/files/ccc2010.pdf, 2010.

[5] "SSL Survey," http://www.netcraft.com/internet-data-mining/ssl-survey.

[6] R. Lin, "Types of SSL certificates - choose the right one," http://www.symantec.com/connect/blogs/types-ssl-certificates-choose-right-one, 2014.

[7] "What is SSL Secure Sockets Layer and What Are SSL Certificates?" https://www.digicert.com/ssl.htm.

[8] M. Jost, "The Password Problem: A Call for Stronger Authentication," http://www.symantec.com/connect/blogs/password-problem-call-stronger-authentication.

[9] D. G. Greenspan, "MultiChain Private Blockchain, White Paper," http://www.multichain.com/download/MultiChain-White-Paper.pdf.

[10] "Certificate Authority," https://en.wikipedia.org/wiki/Certificate_authority.

[11] "Survival guides - TLS/SSL and SSL (X.509) Certificates," http://www.zytrax.com/tech/survival/ssl.html.

[12] "Public Key Infrastructure," https://msdn.microsoft.com/en-us/library/windows/desktop/bb427432(v=vs.85).aspx.

[13] "Namecoin," https://namecoin.info.

[14] D. V. Conner Fromknecht and S. Yakoubov, "Certcoin: A namecoin based decentralized authentication system," MIT Class Project 6.857, May 2014.

[15] D. Kraft, "Nameid," https://nameid.org/, accessed 24 Oct 2016.

[16] G. Slepak, "Dnschain," https://okturtles.com/other/dnschain_okturtles_overview.pdf, accessed 24 Oct 2016.

[17] C. Ellis, "Blockchainid," https://github.com/MrChrisJ/World-Citizenship, accessed 21 April 2016.

[18] R. Shea, J. Light, and G. Lepage, "Blockchainid," https://github.com/blockstack/blockchain-id/wiki, Oct. 2015, accessed 01 May 2016.

[19] C. Allen, "Revocable, self-signed tls certificates," https://github.com/ChristopherA/revocable-self-signed-tls-certificates-hack, accessed 21 April 2016.

[20] T. Dierks and C. Allen, "Rfc 2246: The tls protocol version 1.0," https://rfc-editor.org/rfc/rfc2246.txt, Internet Engineering Task Force, Jan. 1999.

[21] Guardtime, "Keyless security infrastructure," https://guardtime.com/ksi-technology, accessed 21 April 2016.

[22] R. L. Ahto Buldas and A. Truu, "Efficient quantum-immune keyless signatures with identity," Jan. 2014, cryptology ePrint Archive: Report 2014/321, 20140902:073252.

[23] B. Laurie, A. Langley, and E. Kasper, "Certificate transparency," Tech. Rep., 2013.

[24] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," http://www.bitcoin.org, 2008.

[25] J. R. Douceur, "The sybil attack," in *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, ser. IPTPS '01. London, UK, UK: Springer-Verlag, 2002, pp. 251–260.

[26] R. C. Merkle, "Protocols for public key cryptosystems." in *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 1980, pp. 122–134.

[27] "Hashcash Proof-of-Work System," http://www.hashcash.org, 1997.

[28] Q. H. Dang, "Recommendation for Applications Using Approved Hash Algorithms," Gaithersburg, MD, United States, Tech. Rep., 2009.

[29] M. Nielsen, "How the Bitcoin Protocol Actually Works," http://www.michaelnielsen.org/ddi/how-the-bitcoin-protocol-actually-works, Dec. 2013.

[30] "List of cryptocurrencies," https://en.wikipedia.org/wiki/List_of_cryptocurrencies.

[31] J. Evans, "Enter The Blockchain: How Bitcoin Can Turn The Cloud Inside Out," http://techcrunch.com/2014/03/22/enter-the-blockchain-how-bitcoin-can-turn-the-cloud-inside-out, Mar. 2014.

[32] "Ethereum," https://www.ethereum.org.

[33] "Embedding data in the blockchain with OP_RETURN," https://21.co/learn/embedding-data-blockchain-op-return/#embedding-data-in-the-blockchain-with-op_return.

[34] "MultiChain data streams," http://www.multichain.com/developers/data-streams/.

[35] M. Nystrom and B. Kaliski, "PKCS #10: Certification Request Syntax Specification Version 1.7," http://www.ietf.org/rfc/rfc2986.txt, Internet Engineering Task Force, Nov 2000.

[36] "X509 certificate examples for testing and verification," http://fm4dd.com/openssl/certexamples.htm.

[37] E. B. Barker, "Sp 800-57. recommendation for key management, part 1: General (revision 4)," http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r4.pdf, Tech. Rep., Jan 2016.

[38] P. Saint-Andre, "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence," https://rfc-editor.org/rfc/rfc6121.txt, Oct. 2015.

[39] R. Housley, "Cryptographic Message Syntax (CMS)," http://www.ietf.org/rfc/rfc5652.txt, Internet Engineering Task Force, Sep 2009.

[40] M. Ali, J. Nelson, R. Shea, and M. J. Freedman, "Blockstack: A global naming and storage system secured by blockchains," in *2016 USENIX Annual Technical Conference (USENIX ATC 16)*. USENIX Association, 2016, pp. 181–194.