DeCERT: A Decentralized Certificate Authority

http://decert.io

Leo Hentschker (leohentschker@college.harvard.edu)

May 29, 2018

Abstract

DeCert is a new certificate authority protocol seeking to combine advancements in decentralization and public key infrastructure. By building on the work of Lets Encrypt's open source certificate authority Boulder[1], DeCert is able to quickly and securely issue free TLS and SSL certificates. Furthermore, in recording all issued certificates on the Ethereum blockchain, DeCert takes a markedly different approach to the "trust problem" inherit in public key infrastructure. Rather than blindly trusting the certificates introduced by DeCert, users of the network engage in a token-based voting scheme to decide which certificates are trustworthy and which aren't. As any network participant can see all the certificates and votes on the network, DeCert is the first certificate authority to allow individual Internet users to make informed decisions on whom to trust. Furthermore, as the network's voting mechanism runs on the certificate level, DeCert presents the first scalable method of efficiently deprecating compromised certificates.

1 Introduction

There have been no fundamental changes to public key infrastructure (PKI) for decades. Since the introduction of the World Wide Web, Internet users and certificate authorities have been authenticating themselves to each other in essentially the same way. Trusted third parties called certificate authorities (CAs) issue encrypted certificates to website owners that allow them to prove their identities. Then, Internet users encrypt packets using these certificates, ensuring that only the designated website owners can read them [2]. While the ciphers being used to encrypt this communication become more advanced every year, this foundational requirement for website owners and Internet users to blindly trust CAs has not changed.

Unfortunately, this approach to PKI is seriously flawed. This is evidenced by the fact that in just the last few years

1. Symantec mis-issued more than 30,000 certificates [3]
2. Let's Encrypt issued over 15k certificates to phishing websites masquerading as PayPal [3]

3. An attack on Comodo allowed a hacker to gain access to a certificate for Google [3]

The core problem with current approaches to PKI is that in comparison to CAs, Internet users and website owners are forced to take a passive role. Without the ability to broadcast to the world when individual certificates have been compromised or check to see what certificates are being issued, Internet users and website owners are unable to prevent identity and data theft after breaches have occurred.

Fortunately, these problems can be solved by leveraging decentralized technologies. By placing all participants on one decentralized network, DeCert maintains a constantly updating record of valid TLS and SSL certificates. By merging this public record with a token-based voting scheme, DeCert introduces a new mechanism of deprecating certificates: consensus. Then, when faced with a new certificate, Internet users can query DeCert to determine whether or not the network thinks the certificate is valid.

DeCert is made up of:

1. **Smart contracts** - written in Solidity for the Ethereum network
2. **Custom CA** - custom certificate authority written in Golang
3. **Decentralized application** - web client to interact with DeCert on the blockchain
4. **Miscellaneous webservers** - to handle testing, deployment and maintenance of the system

All together, these make up a functioning, publicly available, alpha version of DeCert.

2 Certificate Authorities

TLS and SSL certificates are, literally, the secrets behind secure internet traffic. By encrypting packets with a private key that only the website knows, these certificates allow website owners to prove to the world that they do in fact own their domains. While incredibly helpful, the need for these certificates introduces a massive problem. How can we effectively distribute certificates at scale without either issuing certificates to the wrong people or letting the keys behind these certificates become compromised?

Currently, this problem is inadequately solved by private companies. Internet browsers like Google Chrome and FireFox trust organizations like Comodo, Qualys, Symantec, DigiCert and Let's Encrypt to only give out valid TLS and SSL certificates. Therefore, whenever these browsers encounter a certificate signed by one of these companies, they trust it completely. This approach has three fundamental problems.

1. CAs can only sign certificates with a very small number of root keys
2. There is no effective way to deprecate individual certificates
3. There is no way to know when new certificates have been issued

Because browsers have to store all trusted root keys, CAs can only sign certificates with a small number of signing identities. Therefore, if a CA's key becomes exposed, a huge proportion of the certificates they've issued become insecure. Furthermore, any compromise to a root key leaves browsers with two unsavory choices. First, they can choose not to load any website with a certificate related to that root key. If this key was issued by a major certificate authority – as the world saw when the CEO of Trustico revealed 23,000 private keys in an email [4] – this can simultaneously take down thousands of websites. Or, they can choose to continue browsing potentially unsafe websites and risk Internet users' data being stolen.

Secondly, without the infrastructure to effectively revoke individual certificates, website owners are powerless to stop a hacker from impersonating them after a certificate has been compromised. This means that until a certificate expires, malicious actors can use stolen certificates to masquerade as a website's owner with impunity. The potential consequences of this were highlighted when in 2011 it was discovered that Comodo issued fraudulent certificates for Google, Yahoo and Microsoft [5]. While these companies were able to work with browsers to quickly deprecate the dangerous certificates, small and even medium-sized Internet business simply do not have the resources to either recognize that one of their certificates has been compromised or do something about it.

Finally, the Internet has a serious phishing problem. One of the most common attack vectors on the web is a phishing attack, in which a malicious actor fakes a website in order to steal a user's online credentials. This has become increasingly common, with almost three quarters of ransomware being delivered through some form of phishing attack in 2017 [6]. One of the core components of these phishing attacks are fraudulent certificates. HTTPS connections with these fraudulent certificates trick users into thinking that their data is secure, making them more likely to fall victim. Without the ability to track what certificates are being issued, website owners don't know when their website is being impersonated. As a result, they have no way to combat these fraudulent certificates.

3 DeCert's Approach

DeCert's protocol obtains security through transparency and collective action. It engages all network participants – Internet users, website owners and CAs – to maintain a public record of valid certificates.

There are three critical actions that users can take on the network:

1. **Add a new certificate** - CA issues certificate to website owner and adds it to the blockchain

2. **Vote on a certificate** - website owner broadcasts on the blockchain that they think a certificate is valid or invalid

3. **Query for a certificate** - after seeing a certificate, an Internet user pulls its information from the blockchain to determine whether or not they trust it

3.1 Action 1 - Add a New Certificate

At its core, DeCert is a public certificate database. Therefore, perhaps the most critical step any network participant can take is to add a new certificate to that database. In DeCert, certificates are represented as Solidity structs with the following schema

```
pragma solidity ^0.4.18;

    ...
    struct Certificate {
        // who issued the certificate
        address issuer;

        // what are you requesting the certificate for
        string domain;

        // when does the certificate become valid
        uint256 validityStart;

        // when is the certificate no longer valid
        uint256 validityEnd;

        // unique identifier for the certificate for the CA
        uint256 serialID;

        // how many votes that the certificate is valid
        uint256 validVotes;

        // how many votes the certificate is invalid
        uint256 invalidVotes;

        // what is the ID of the Cert in the list?
        uint listID;

        // what is the certificate's signature?
        string signature;
    }

    ...

    /// @dev Add a certificate to the chain
    /// @param _duration how long is the cert valid for
    /// @param _serialID unique ID for the cert based on issuer
    function addCertificate(
        string _domain,
        uint256 _serialID,
        uint256 _duration
    )

    ...
```

At the moment, certificates reflect a paired down version of the X.509 standard. The fundamental idea is that when presented with a new certificate, Internet users should be able to look it up on the blockchain to determine whether or not it is valid. Therefore, certificates

on chain must contain all the information necessary to uniquely identify a given certificate.

3.2 Action 2 - Vote on a Certificate

After a certificate has been added, any member of the network can vote on whether or not they believe it is valid. The intention is that if a website owner or CA realizes that a certificate has been compromised, they can broadcast that knowledge to the rest of the network and prevent other actors from engaging with the fraudulent certificate.

```
pragma solidity ^0.4.18;

    ...

    struct Vote {
        // who voted
        address voter;

        // what did they vote on
        Certificate cert;

        // is it valid or not
        bool valid;

        // how many votes
        uint256 votes;
    }

    ...

    /// @dev Vote on whether or not a particular certificate is valid
    /// @param _issuer who issued the certificate
    /// @param _serialID what is the ID of the certificate
    /// @param _valid is the cert valid or not
    /// @param _votes how many votes to spend
    function voteOnCert(
        address _issuer,
        uint256 _serialID,
        bool _valid,
        uint256 _votes
    )

    ...
```

DeCert uses a token-based voting system built on an ERC-20 compliant token. Therefore, in order to vote, one must first purchase tokens from the network. For this initial beta test, DeCert is selling a total of 10,000 tokens.

3.3 Action 3 - Query for a Certificate

Now that certificates have been uploaded and network participants have voted on which certificates they believe are valid, Internet users can use this information when browsing the

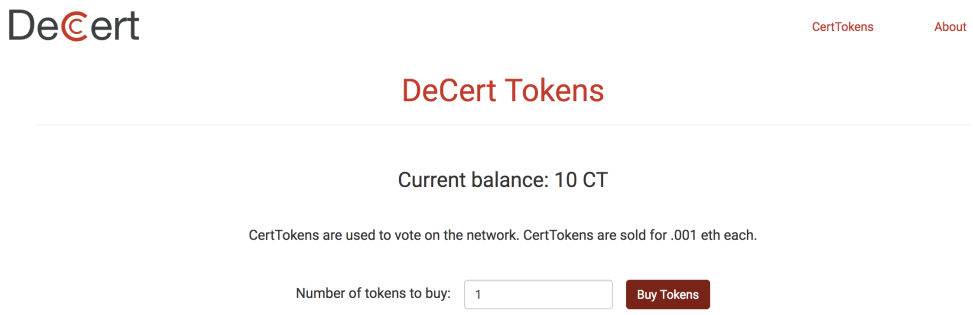

Figure 1: Purchasing tokens from the network

web. Whenever they are presented with a new certificate, they can read information from the blockchain and determine whether or not they should trust it.

```
pragma solidity ^0.4.18;

    ...
    /// @dev Read properties of a cert from the chain
    /// @param _issuer who issued the certificate
    /// @param _serialID unique ID for the cert based on issuer
    function getCertificate(
        address _issuer,
        uint256 _serialID
    )
        public
        view
        returns (
            address issuer,
            uint256 serialID,
            string domain,
            uint256 validityStart,
            uint256 validityEnd,
            uint256 validVotes,
            uint256 invalidVotes
        )
    ...
```

This method of certificate querying is innovative for two reasons. First, it lets users dynamically choose which certificates to trust based on the opinion of their peers. Secondly, it

allows the network to effectively deprecate individual certificates. Consider, for example, a website owner who realizes that their certificate has been compromised and broadcasts this information to the network by voting. Simply by looking at the number of valid and invalid votes on a given certificate, anyone visiting the website is instantly notified of the potential dangers surrounding that certificate.

4 DeCert Implementation

The literal implementation of the protocol can be broken up into 5 different components.

4.1 Piece 1 - Smart Contracts

The core of DeCert is made up of a series of Ethereum smart contracts. These contracts the three previously mentioned actions:

1. Allow CAs to publish certificates they issue to the network
2. Allow network participants to vote on whether or not they believe a certificate is valid
3. Allow anyone to read certificate data from the network

These contracts have been deployed to Ropsten, an Ethereum test network designed for piloting projects.

4.2 Piece 2 - Certificate Authority

In order to programmatically add certificates to the network, we implemented a certificate authority that interacts with the previously deployed smart contracts. This step was simplified by forking the code from Boulder, an open source CA maintained by Let's Encrypt. In order to integrate Let's Encrypt's code with the decentralized network, after issuing a certificate, we add an additional call to a local webserver that interacts with the previously deployed contracts and stores the issued certificate on the blockchain. Furthermore, because this CA is nearly identical to Let's Encrypt's, users wishing to request new certificates from it can use existing tools like certbot.

A live implementation of the forked CA can be accessed at `https://ca.decert.io`.

4.3 Piece 3 - Frontend to Simplify Network Interactions

One of the most important components of any Ethereum application is an intuitive frontend. Using the following tools, we implemented a decentralized web application to allow users to easily view contracts and vote on whether or not they believe them to be valid.

- JavaScript/React - used to build a reactive, single page application
- Truffle - JavaScript development framework for Ethereum
- MetaMask - hosts a remote Ethereum node

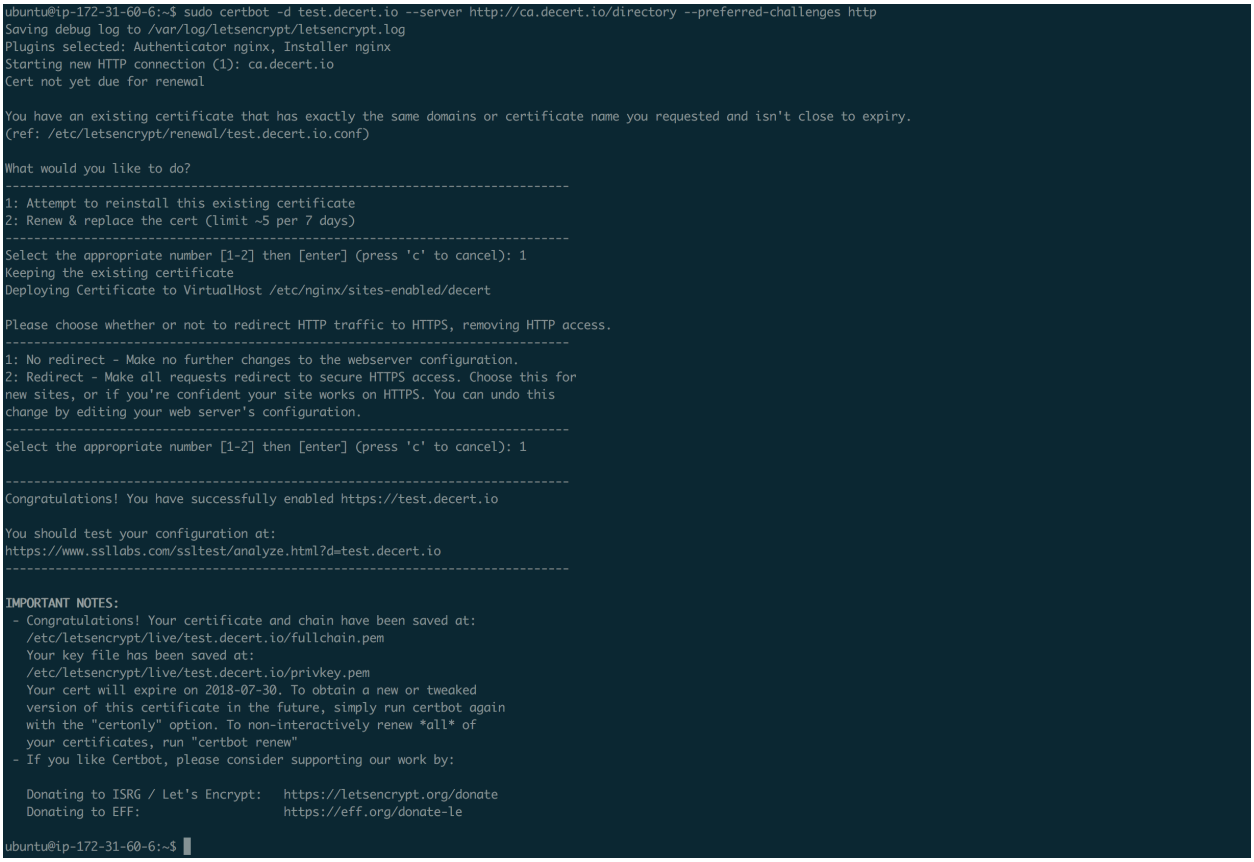

Figure 2: Renewing a certificate using certbot and our custom CA

A live version of the website is running at http://decert.io.

Note: you will need to have MetaMask installed and running on Ropsten to use the site properly.

4.4 Piece 4 - Backend to Allow Searching Over Deployed certificates

One fundamental problem with decentralized networks is that they don't allow for indexed queries. Therefore, in order to enable searching at scale, I implemented a backend server that duplicates the data being stored on chain in a remote database. In order to create this server we made use of:

- Django - for easily writing python webservers
- PostgreSQL - for data storage and searching
- Nginx - reverse proxy server
- Infura - hosts a remote Ethereum node, used to pull data from the blockchain without running our own instance

A live version of the API the backend is running at https://api.decert.io.

4.5 Piece 5 - Test Website

In order to test our CA implementation, we needed a toy webserver to encrypt traffic to. Again, we created a basic implementation using Django and Nginx. This webserver is alive running at https://test.decert.io.

Note: because your browser doesn't trust the root key of the DeCert CA, you will be warned that you are visiting an insecure website if you go to either https://api.decert.io or https://test.decert.io.

5 Next Steps

In its current state, DeCert has two major problems:

1. No browser integrations
2. Overly simplistic voting

5.1 Lacking Browser Integrations

Without browser integrations, DeCert is just a proof of concept. At the moment, it cannot be used to either accept or reject a certificate from an actual website as it does not hook into any browser's SSL or TLS handling. Furthermore, without existing infrastructure around dynamically accepting new certificates, overriding this functionality fell beyond the scope of this project. Therefore, a necessary next step for this project would be to dive into how browsers interact with certificates.

5.2 Overly Simplistic Voting

The current voting scheme forces honest participants to continuously outspend malicious participants on all certificates. As users pay tokens to vote, if a bad actor concentrates their resources and outspends good actors with respect to a single certificate, they can convince the network that their certificate is valid when it isn't. Therefore, bad actors have an incentive to spend money and claim that a particular certificate is valid when it isn't.

One potential fix is to use quadratic voting, in which users must pay for the square of the number of votes they exercise [7]. This would make it significantly harder for bad actors to outspend the rest of the network. However, malicious network participants are still able to decrease trust in DeCert by claiming that all certificates are fraudulent. Therefore, this approach also requires all network participants to be vigilant in determining whether any particular certificate added to the network is valid.

Therefore, a better approach is to provide a fixed number of non-fungible tokens to trusted network participants. Under this system, these participants can effectively monitor certificates on the network and deal with security problems as they arise. Furthermore, as

long as no single entity controls the majority of the tokens on the network, no individual interest will be able to misrepresent the validity of a certificate.

6 Conclusion

Over the last few years, the brilliant work by Let's Encrypt has shown the world that all web traffic can and should be encrypted. Within the confines of modern public key infrastructure, more Internet users are accessing Internet content in a secure environment than ever before. However, we as a community cannot stop there. Without the ability to effectively deprecate individual certificates, theory and history show that current PKI systems aren't nearly as robust as they should be. While decentralized networks are still an immature technology, the consensus-based voting schemes they enable can solve many of the problems plaguing current PKI systems. Therefore, as the cryptographic community continues to improve how the world encrypts their web traffic, we recommend keeping blockchains in mind. For, while DeCert is just a proof of concept, it shows that decentralized networks can solve some of the most fundamental problems facing PKI today.

References

[1] "Boulder." https://github.com/letsencrypt/boulder, 2018.

[2] "Public key infrastructure." https://msdn.microsoft.com/en-us/library/windows/desktop/bb427432(v=vs.85).aspx, 2018.

[3] Jayaraman, Bargav, Li, Hannah, Evans, and David, "Decentralized certificate authorities," Oct 2017.

[4] Z. Whittaker, "Trustico compromises own customers' https private keys in spat with partner," Mar 2018.

[5] P. Roberts, "Phony ssl certificates issued for google, yahoo, skype, others," Mar 2011.

[6] "2017 global threat intelligence report." https://www.dimensiondata.com/Global/Downloadable Documents/2017 Global Threat Intelligence Report as released by NTT Security.pdf, 2017.

[7] "Quadratic voting + smart contracts = powerful governance model." https://medium.com/eximchain/quadratic-voting-smart-contracts-powerful-governance-model-b8efa4ddeef1, 2017.