IJARSCT

ISSN (Online) 2581-9429

International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)

Impact Factor: 6.252

Volume 2, Issue 1, November 2022

Blockchain-based Certificate Verification

Harshada S. Nichit[1], Mitali J. Gadge[2], Pallavi S. Sonawale[3], Sneha S. Gadekar[4], Prof. Rote R.[3]
Students[1,2,3,4] and Guide[5]
Samarth Group of Institution College of Engineering Belhe, Maharashtra, India

Abstract: *Traditional public key infrastructures (PKIs) depend on trusted certification authorities (CAs) to issued certificates, used in SSL/TLS to verified web servers and establish secure channels. However, recent security incidents shows that CAs may issue fake certificates. In this paper, we suggest blockchain-based certificate transparency (CT) and revocation transparency (RT) to balance the complete authority of CAs. Our scheme is suitable to PKIs but significantly reinforces the security guarantees of a certificate. The CA-issues certificates and their cancellation status information of an SSL/TLS web server are published by the subject (i.e., the web server) as a transaction in the global certificate blockchain. The certificate blockchain acts as add only public logs to monitor CAs' certificate issues and revocation operations, and an SSL/TLS web server is permits with the cooperative control on its certificates. A browser compares the certificate received in SSL/TLS negotiations with the ones in the public certificate blockchain, and accepts it only if it is published and not cancelled. We apply the prototype system with Firefox and Nginx, and the trial results show that it establishes reasonable overheads.*

Keywords: Blockchain; Certificate transparency; Certificate revocation; Public key infrastructure; Trust management.

I. INTRODUCTION

In public key infrastructures (PKIs), a certification authority (CA) issues certificates to tie the public key of a server to its identity (e.g., a DNS name). These certificates are used in SSL/TLS to verified web servers. Trusting the CAs, browsers gain the servers' public keys from CA-signed certificates in SSL/TLS debate, to establish secure channels.

However, security incidents indicate that CAs are not so trustworthy as they are assumed to be. CAs may sign fake certificates due to reckless identity validations.

Government compulsions. Typical fake certificates tie a DNS name (e.g., www.facebook.com or www.gmail.com) to key pairs held by counterfeit web servers . Then, the dummy servers will successfully launch man-in-the-middle (Mitm) attacks, even if a browser follows the strictest steps of certificate validation to establish SSL/TLS sessions.

Certificate transparency (CT) was proposed to enhance the accountability of CA operations, by add only public logs . A CA-signed certificate is publicly recorded in the log servers; otherwise, a browser cancells it in SSL/TLS negotiations. So a fake certificate will be examined by interested parties, especially the owner of the DNS name (or the web server).

Independent auditors occasionally verify the honesty of the records to ensure that they are add- only, i.e., a (fake) certificate will never be deleted or updated after added. In this paper, we represents a blockchain-based storage to build public logs for certificates, which is basically add-only. A blockchain is a peer-to-peer (P2P) storage chain of blocks, each of which includes transactions . Each P2P node always considers the largest chain that it ever received as the valid version, and subsequent blocks are appended to the currently-valid chain and broadcast to other nodes. To add a block, a nonce needs to be found by brute force, which is called mining. The computation cost of block mining prevents attackers from forging another larger chain (i.e., modifying the certificate logs), after the valid version has been accepted by a majority of nodes.

II. THREAT MODEL AND DESIGN GOAL

Attackers attempt to copy an SSL/TLS web server using fake certificates, and a successful attack means that browsers accept the fake certificates in SSL/TLS debates. The attackers could adjust some CAs that are trusted by browsers, to sign fake certificates mandatory to the target web server's DNS name to any key pair. We assume that the attackers could directly adjust a number of web servers except the target server, while the maximum number of servers keep honest (i.e., follow their specifications of our scheme). We do not assume that the attackers could directly adjust the target web server. In such cases, the attackers copy the server directly using the adjusted key pair, and no fake certificate is needed. Such attacks cannot be seen by certificate management, and it is out of the scope of this paper.

For a settlement server, the attackers could use all its long-term secrets (e.g., key pairs) to create any messages. In particular, the attackers might collaboration with spiteful servers to certify a fake publishing key. That is, we consider the attack scenario where the publishing key pair is held by attackers and fake certificates are issued by adjusted CAs at the same time.

The attackers do not hold computation resources to randomly change the blockchain , and all cryptographic primitives are secure. The attackers cannot block the network for a long time to take attack actions; that is, honest entities communicate with each other in a loosely coexisted manner.

We aim to protect browsers against the MitM and copy attacks using fake certificates. In the attack case that CAs are compromised to sign fake certificates, if a browser follows our scheme to validate server certificates in SSL/TLS debate, the certified peer is ensured to be the legal server of the visited DNS name. Even in the extreme case that a fake certificate has been published in the blockchain by a compromised publishing key pair, the browser cancels this fake certificate and it will be recovered by corrective transactions in the certificate blockchain.

III. BLOCKCHAIN-BASED CT AND RT

here are two types of certificate transactions. A Type-I transaction is issued by a web server using its publishing key pair, to ocassionally publish certificates. When a certificate comes to an end and is renovated, the new one will be included in the next transaction. If a certificate is cancelled, it will be removed from the next transaction, and the communicating CRL file or OCSP response will be included instead. Type- II transactions are used to loaded or reset publishing key pairs. When a DNS name (or web server) is initially introduced into this community, its publishing key is signed by a number of web servers (called certifiers) using their publishing key pairs. The publishing key may be reset by another Type-II transaction, if it is compromised or lost.

Certificate transactions labeled with a same DNS name, either Type-I or Type-II, are chained chronologically, as shown in Figure 1. For every web server, its continuous history of certificates and publishing keys is archived publicly in the certificate blockchain. We also design a series of rules, enabling honest web servers to take counter measures to recover their certificates or publishing keys, after a fake certificate or publishing key is observed.

3.1 Subject-controlled Certificate Publication

1. A certificate is published by its subject, in Type-I certificate transactions. Each Type-I transaction includes:
2. DNS_Name, the DNS name of the web server.
3. Prev_TX_Hash, the hash value of the previous transaction with the same DNS name, either Type-Ior Type-II.
4. Type, marked as Type I.
5. Validity, the validity period of this certificatepublication.
6. List_of_Cert_Chain, a list of published certifi-cate chains. The web server may hold multiple certificates.
7. Next_Publishing_Key, the publishing publickey. The corresponding private key is used to signthe *next* Type-I transaction with the DNS name.
8. Sig, the signature of this transaction using the current publishing key pair (i.e., the one bound in the most recent transaction with this DNS name, either Type-I or Type-II).

3.2 CT and RT

A web server ocassionly publishes its certificates, and all certificates are openly visible in the inherently attached only blockchain to achieve CT. If a new certificate is issued for the web server, it will be included in the next Type-I transaction

For Type-I transactions, there is an upper limit for the validity period, and it is generally smaller than that of certificates. Thus, a certificate is published for several times during its lifecycle, and then its upgraded cancellation status is also reflected in certificate transactions. When a certificate is cancelled and the web server eliminated it from the next transaction before it expires, the corresponding cancellation information is included instead. So CAs' cancellation operations are also recorded publicly in the blockchain. For example- a certificate Cert2 is published in the transaction TX1 and revoked later, so it is excluded from the next transaction TX2 and the CRL file is in TX2 instead.
Because the validity period of Type-I transactions shall be larger than the update period of OCSP responses/CRL files, the transparent revocation status may be not so fresh. Anyway, a web server can urgently publish a new Type- I transaction once a certificate is cancelled, instead of waiting for the next period.

3.3 Initialization and Reset of Publishing Keys

Type-II transactions are used to a) initialize the publishing key of a web server, and b) reset it if compromised or lost. Note that, even when the publishing key pair is not compromised, the web server may update it in Type- I transactions. Each Type-II transaction is signed by at least a certain number (denoted as G) of certifier web servers, and also by the certified server itself using a key pair bound in one of its certificates. That is, we require at least G web servers to certify it directly and one CA to do indirectly.

A Type-II transaction includes the following fields:

1. DNS Name, the DNS name of the web server.
2. Previous TX Hash, the hash value of the previous transaction with the same DNS name, either Type-I or Type-II.
3. Type, marked as Type II.
4. Publishing Key, the public key of the certified publishing key pair.
5. Certifier Group, a list of certifiers' DNS names. The corresponding web servers are authorized to cooperatively control the publishing key of this DNS name.
6. List of Cert Chain, a list of web server's certificate information, optional. It is used to issue certificates in Type-I transactions.
7. Sig by Owner, a signature by the certified web server. It is confirmable using a CA-signed certificate binding the DNS name, which is also included in this field.

List of Sig, a list of signatures signed by certifiers using their current publishing key pairs. The sponsors DNS names are also in this field

B. Certificate Validation by Browsers

First of all, a browser convey with the P2P storage network to incrementally download the up-to-date block headers. Browsers download and store only block headers but no transactions, to reduce the overheads of convey and storage. The browser validates whether the downloaded headers are chained truely and each header contains a valid PoW nonce, and updates its local copy if a longer chain is received. This harmonization may be performed when there is no SSL/TLS negotiation.
The certificate transactions to verify a certificate are sent by the visited web server during the SSL/TLS negotiation. A browser sends its certificate deals request as an SSL/TLS extension. The server, if it supports the pro- posed Type-II transactions are used to a) initialize the publish- ing key of a web server, and b) reset it if compromised or lost. Note that, even when the publishing key pair is not compromised, the web server may update it in Type- I transactions. Each Type-II transaction is signed by at least a certain number (denoted as G) of certifier web servers, and also by the certified server itself using a key pair bound in one of its certificates. That is, we require at least G web servers to certify it directly and one CA to do indirectly.

C. Limited Storage of Blocks

Expired Type-I transactions are pointless in certificate proof, so a browser only needs to reserve the recent block headers within the upper limit of the validity period of Type-I transactions (denote as T_I). T_I is set great enough

(e.g., 10 days), so the longest branch of the block chain is always mined on one of the stored headers.
A collier needs to store all blocks, including title and deal, to *a*) verify the connection of certificates and publishing keys of a web server among its Type-I and Type-II transactions, *b*) find the certifiers' publishing keys to verify Type-II transactions, and *c*) check whether a assistance is intelligent as a certifier or not.
The following designs reduce the cache requirement of miners. Each web server publishes its Type-II action repeatedly, even when it does not reset the issue keypair or modify its certifier group. The period is denotedas T_{II} and $T_{II} > T_I$. In these "shadow" Type-II transactions, both Publishing Key and Certifier Group must be identical with those in the previous transaction, and List of Cert Chain must be absent. A shade transaction is signed only by the web server itself, and no signature by certifiers is needed. Finally, if a web server is involved in any event, a flag is set with its DNS name in the block header.
So a miner only stores *a*) the recent block headers withinthe certain period of time (denoted as T_G, and $T_G > T_{II}$), to check whether a server is qualified as a certifier, and *b*) the latest transactions of both types of each DNS name within T_{II}. If a web server does not issue shadow Type-II action, it is still able to issue repeatedly Type-I transactions, but cannot reset its publishing key or modify the certifier group any more.

IV. SECURITY ANALYSIS

If CAs are trustworthy and web servers are honest, a web server periodically publishes its certificates using the publishing key pair, which is certified by at least *G* servers. If a certificate becomes finish or cancle, it will be excluded. Meanwhile, a web server constant renew its publishing key, after it is introduced into the group (i.e., its publishing key is at first certified). If the publishing key is departed, the web server will contact its certifiers to altered it.
Next, the suggest scheme is analyzed under various strike scenarios as pursue. When some entities except the target web server were compromised, we present the corresponding cure and evaluate the attack impacts. In this attack analysis on certificate transactions, we assume that the certificate block chain is include only, so that a fully confirmed transaction never will never be remove or change. Then, the network attacks on the certificate block chain are also examined. Note that, an attack is successful, if and only if a false certificate is published in the certificate block chain and accepted by some browsers.

Security with Compromised Key Pairs:

In order to impersonate a web server, an attacker might compromise a CA to sign fraudulent certificates, or compromise some web servers' publishing key pairs. We donot consider the situation that, the key pair bound in the target server's certificate is compromised by attackers. In this situation, the attackers could directly impersonate the web server and launch M it M attacks, without fraudulent certificates. Such attacks should be prevented or mitigated by the approaches other than certificate management, which are out of the scope of this paper.
Set of all possible next attack actions when different key pairs are compromised. if only publishing key pairs are compromised, the attackers can set List of Cert Chain or modify the publishing key pair in fraudulent Type-II transactions. When the attackers compromise CAs and some servers' publishing key pairs, the possible subsequent attack actions include: publish fraudulent certificates, modify the publishing key pair of the target server, and compromise the publishing key pairs of *G* certifiers. Then, if the publishingkey pairs of *G* certifiers are compromised, the possible attack actions include: set List of Cert Chain, publish fraudulent certificates, reset the publishing key pair, and modify the certifier group.

V. CONCLUSION

We propose to record certificates and revocation status information in the global block chain, which is inherently append-only, to achieve CT and coarse-grained RT. In the certificate block chain, a certificate is periodically published as transactions by its subject, to balance the absolute authority of CAs. These transactions provide a continuous history of certificates for each SSL/TLS web server. The publishing key pairs used to sign certificate transactions, are also con- trolled cooperatively by CAs and the community of web servers, and recorded transparently in the block chain. A series of regulations are designed to facilitate honest servers to sign countermeasure transactions and recover their certificates or publishing keys, under various attack scenarios. The scheme integrates CT, RT and subject-controlled certificate publication into block chain based append only logs. It is compatible with X.509 PKIs but significantly reinforces the security guarantees of a certificate.

REFERENCES

[1]. M. Abadi, A. Birrell, I. Mironov, T. Wobber, and Y. Xie, "Global authentication in an untrustworthy world," in 14th USENIX Conference on Hot Topics in Operating Systems (HotOS), 2013.
[2]. M. Alicherry and A. Keromytis, "Doublecheck: Multi-path verification against man-in-the-middle attacks," in 14th IEEE Symposium on Computers and Communications (ISCC), 2009, pp. 557–563.
[3]. R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "IETF RFC 4033 - DNS security introduction and requirements," 2005.
[4]. C. Arthur. (2011) Rogue web certificate could have been used to attack Iran dissidents. [Online]. Available: https://iranian.com/main/news/2011/08/30/
[5]. rogue-web-certificate-could-have-been-used-attack-iran-dissidents. html
[6]. G. Ateniese and S. Mangard, "A new approach to DNS security (DNSSEC)," in 8th ACM Conference on Computer and Communications Security (CCS), 2001, pp. 86–95.
[7]. J. Braun, F. Volk, J. Classen, J. Buchmann, and M. Mu¨ hlha¨user, "CA trust management for the Web PKI," Journal of Computer Security, vol. 22, no. 6, pp. 913–959, 2014.
[8]. M. Castro and B. Liskov, "Practical Byzantine fault tolerance," in 3rd Symposium on Operating Systems Design and Implementation (OSDI), 1999, pp. 173–186.
[9]. Censys. (2016) Censys public reports. [Online]. Available: https://censys.io/
[10]. M. Chase and S. Meiklejohn, "Transparency overlays and applications," in 13th ACM Conference on Computer and Communications Security (CCS), 2016, pp. 168–179.