A Permissioned Blockchain with Proof of Location for Digital Cinema

Eric Diehl
Sony Pictures Entertainment
Culver City, CA, USA
https://orcid.org/0000-0003-2682-3067

***Abstract*— This paper presents the use of a Hyperledger Fabric blockchain to manage the inventory of digital cinema projectors. It is necessary for the protected distribution of digital movies to digital theaters. The document discloses a commitment-based protocol that proves a device's location.**

Keywords— blockchain, smart contract, Hyperledger Fabric, digital cinema, proof of location

I. INTRODUCTION

For many years, the movie industry has transitioned from analog projection to digital projectors. Digital projection offers a better visual experience than its analog counterpart. The secure distribution of digital movies to digital theaters requires an accurate inventory of digital projectors. This paper discloses a method to generate proof of location for a device. The proof of location uses a user-friendly commit protocol based on QRCode.

II. RELATED WORK

Several schemes of proof of locations using blockchain. Nosouhi et al. disclose a scheme where a user proves his/her location proof having neighboring devices witness its presence using Bluetooth [1]. Amoretti et al. modify the Proof of Stake for adding a proof of location provided by witnesses' nodes in a peer-to-peer network [2]. Once more, the witnesses are in the neighbor range with short-range communications.

III. PROOF OF LOCATION

The Blockchain Trusted Device List (BC-TDL) handles the allocation of exhibitor's projectors to its auditoriums. It implements many features to ensure an accurate inventory. In addition, it can verify whether the projector is in the theater it claims to be. This paper describes the corresponding protocol. This protocol is a two-step process: a commitment phase and a challenge answer phase.

A. Commitment

During the first step, the exhibitor commits to prove that its projector *D* is in auditorium *A* of its theater *T*. This commitment invokes the transaction "*commit device.*"

The blockchain generates two nonces: *seed* and *challenge*. It calculates then *answer* such as

$$answer = SHA256(seed|challenge)$$

The blockchain changes the projector's state to "*committed*" and stores both *answer* and *seed* with the projector's information in the blockchain ledger. The blockchain turns *challenge* into a QRCode. Only the committed projector should be able to display this QRCode.

The secure distribution of movies to digital theaters encrypts the content into an encrypted container called Digital Cinema Packages (DCP). Each element of the DCP is encrypted with AES-128 in CBC mode. Each authorized projector receives the decryption key in a message called Key Delivery Message (KDM), uniquely encrypted for the projector. Thus, BC-TDL delivers the QRCode as a DCP with the corresponding KDM. Both DCP and KDM are sent to the exhibitor.

B. Answer commitment

During the second step, the exhibitor plays back the DCP with the committed projector *D*. Using a mobile application, the exhibitor captures the displayed image. The mobile application sends the extracted QRCode, i.e., *challenge*, together with the mobile device's GPS coordinates to BC-TDL as a transaction.

The blockchain verifies whether the captured QRCode carries the expected value, i.e., that the previous equation is valid and whether the captured position corresponds to the close vicinity of the theater's location provided when registering the theater in the blockchain. If successful, the transaction proved the location of the projector.

Providing such proof of location would not be systematic. It may be required when installing a projector in a new location or case of suspicious activity.

C. Security considerations

Can an attacker guess *challenge?* The information stored in the ledger is public. Thus, the attacker may know *answer* and *seed*. SHA256 is a secure cryptographic one-way hash function. The best attack is brute force exploration. For 256-bit, it is not feasible.

Can an attacker who has a rogue projector impersonate the position of a legitimate theater? The attacker could take a picture of the QRCode in its location and later send the QRCode from the impersonated theater. A projector can only be committed to an auditorium that the device's owner owns. Thus, to initiate the commitment, the attacker should either create a ghost theater shadowing a legitimate location or succeed in committing the rogue projector to the legitimate theater's auditorium.

The "create facility" transaction checks potential shadowing before approval, thus, preventing the first solution. BC-TDL rejects the creation of a new theater in the close vicinity of an existing one. The second solution requires the attacker to impersonate the legitimate exhibitor, i.e., having access to the legitimate exhibitor's Hyperledger Fabric private key. A usual security hypothesis is that an attacker has not access to the user's private key.

A security assumption is that the attacker cannot forge the GPS coordinates sent by the mobile application. Tamper resistance and jailbreaking detection for the mobile application are mandatory. Some commercial solutions provide reasonable protection.

IV. Proof of Concept

A. The blockchain

The first decision was the choice of blockchain technology: permission-less versus permissioned. The DCI ecosystem is a very controlled environment. Each actor must be unambiguously identified. Thus, selecting a permissioned blockchain was a natural choice. BCTDL uses Hyperledger Fabric 2.2.0 [3].

B. The architecture

The POC uses three peer nodes and three orderer nodes belonging to the same organization. Each peer uses a CouchDB. The nodes and the databases are in Docker containers running on EC2 instances on Amazon Web Services.

The QRCode generator is an off-the-chain microservice that creates the DCP and KDM requested by the blockchain and delivers them to the exhibitor wallet.

The mobile application captures the QRCode and sends it with GPS coordinates to the blockchain. The mobile application runs on Android. It is developed in Java and uses the gateway programming model [4].

C. Generating the visual challenge

The proof of location requires a visual challenge under the form of a QRCode. The QRCode must be displayable by a projector. Thus, the QRCode must be turned into an encrypted DCP. Furthermore, the system must generate a corresponding KDM for the targeted projector. It is not possible for the blockchain to perform these operations. An off-chain server, called the generator, takes care of these operations. The generator needs the *challenge* as well as the certificate of the projector to generate the files.

It would have been possible to design a solution such as the smart contract calls the server. Such a design would add a strong coupling between the chaincode and the generator. We preferred a solution with loose coupling.

When receiving a valid transaction "*commit device,"* the chaincode generates the data described in Section III.A. The chaincode packages *challenge* and the device's certificate into a signed token. It returns the signed token to the exhibitor who passes it to the generator.

The generator validates the token. It turns the *challenge* into a QRCode. The QRCode is inserted inside a larger background image. ffmpeg transforms the image into a 10-second, fixed image, high definition, video sequence. The open-source software DCP-o-matic v2.12 [5] creates an encrypted DCP from this video sequence. It generates the corresponding KDM using the projector's X509 certificate in the received signed token. They are compressed into a less than 1 MB file. The generator creates a token and returns it to the exhibitor. The exhibitor retrieves the compressed file at its convenience by asking for the file identified by the provided token. Fig. 1 illustrates the timeline of this protocol.

The signed token is a Java Web Token (JWT) [6] with the payload *challenge* and the committed projector's certificate. JWT uses asymmetric authentication with 256-bit ECDSA. The chaincode handles a public key and not a private key. It is easier to protect a secret in an off-chain server than in a chaincode.

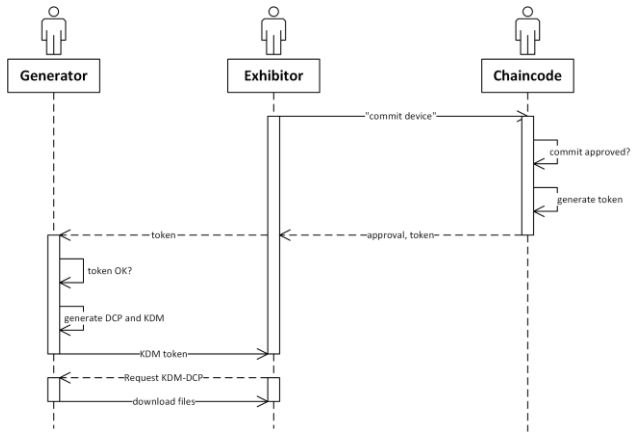

Fig. 1. Generating the visual challenge

The token identifying the compressed file does not require security. The KDM and DCP are useless if the attacker has not the corresponding projector.

D. Real Experiment

BC-TDL has been tested with the theaters in Sony Pictures Entertainment (SPE) lot at Culver City. SPE has four theaters used for screening on the lot. Each theater has from one to six auditoriums. The entire lot holds in a radius of less than 300m. We tested the proof of location protocol on the four theaters using the installed digital projectors. We first tested the projectors by committing them to their actual auditorium. The challenges succeeded. We then tested the projectors by committing them to an auditorium of the nearest neighboring theater. As expected, the challenges failed as the GPS coordinates did not fit.

Conclusions

The application of blockchain in the media world usually focuses on two domains: management of digital rights [7] and non-fungible tokens. BC-TDL is one of the first usages of blockchain in the media industry outside the two previous domains. The proof of location adds a security feature that does not exist in the current Digital Cinema ecosystem. This proof of location concept may be suitable to other contexts.

References

[1] M. R. Nosouhi, S. Yu, W. Zhou, M. Grobler, and H. Keshtiar, "Blockchain for secure location verification," Journal of Parallel and Distributed Computing, vol. 136, pp. 40–51, Feb. 2020, doi: 10.1016/j.jpdc.2019.10.007.

[2] M. Amoretti, G. Brambilla, F. Medioli, and F. Zanichelli, "Blockchain-Based Proof of Location," in 2018 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C), Jul. 2018, pp. 146–153, doi: 10.1109/QRS-C.2018.00038.

[3] "Welcome to Hyperledger Fabric — hyperledger-fabricdocs master documentation." https://hyperledger-fabric.readthedocs.io/en/latest.

[4] IBM, "Fabric SDK: New Programming Model," 2019, [Online]. Available: https://jira.hyperledger.org/secure/attachment/18031/fabric-programming-model-go.pdf.

[5] "DCP-o-matic." https://dcpomatic.com/.

[6] J. Bradley, N. Sakimura, and M. Jones, "JSON Web Token (JWT)." https://tools.ietf.org/html/rfc7519.

[7] "Eluvio," GitHub. https://github.com/eluv-io.