Decentralized Image Authentication Mechanism for Android Camera

Arav Raval, Shivam Kak, Huseyin Sahin, Jason Ding

ECE ‘26 | COS ’26 | COS ‘26 | COS ’25 | Princeton University | COS / ECE 473 | Professor Pramod Viswanath | 17 April 2024

Introduction

In an era where trust and authenticity are paramount in the digital realm, our project emerges as a groundbreaking solution at the forefront of blockchain-based image authentication. With a focus on preserving user privacy and ensuring the veracity of visual content, we introduce an innovative NFT collection-based image authentication service tailored for Android devices. Every day a total of 24 million AI generated images are created, amassing a total of 15 billion images.

The core principle of our project revolves around the creation of a unique NFT for each authenticated image. Upon detecting new photos within the user's Android phone, our system automatically initiates the authentication process. It waits for an event emitted directly from the Android device, signaling that a photo has been taken. This event-driven approach ensures that the authentication process is triggered only when a genuine photo-capturing event occurs, mitigating the risk of tampering or manipulation by malicious actors.

Objectives

Short Term:

- Authenticate and sign images captured by a camera (i.e. Android/Apple) and integrate with existing camera software
- Ensure signature cannot be replicated or forged by making use of immutable metadata unique to individual users
- Mint NFTs and maintain an aggregated database for all signed, authenticated images
- Query images against this repository to discover deepfake, edited, or doctored images

Long Term:

- Create a worldwide ecosystem of camera-verified images and NFTs with corresponding image hashes
- Hardware-level automations: working with chip manufacturers to further secure signature mechanism upon photo capture event
- Restore consumer trust in digital media

Existing Work

1. Truepic: Truepic employs blockchain to create a secure and tamper-proof chain of custody for images. They use cryptographic hashing to generate a unique fingerprint for each image, which is then recorded on a blockchain ledger. This allows users to verify the authenticity of images and detect any alterations.
2. Serelay: Serelay utilizes blockchain and advanced computer vision algorithms to verify the authenticity of images. They enable users to capture and certify images directly from their mobile devices, with the blockchain ensuring that the images are unaltered and trustworthy.
3. Blockpact: Blockpact focuses on leveraging blockchain to combat deepfakes and image manipulation. Their platform integrates blockchain technology to establish a secure and transparent record of image provenance, enabling users to verify the authenticity of images.

Disadvantages

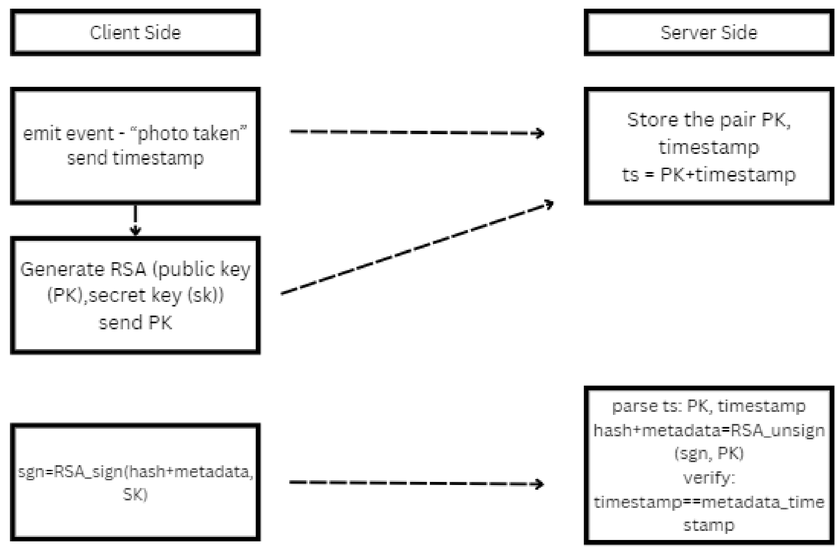
- Pictures must be taken on the company’s app allowing them to see the image as it is taken.
- MITM attack mitigation depends on the metadata attached in the app.

Methodology

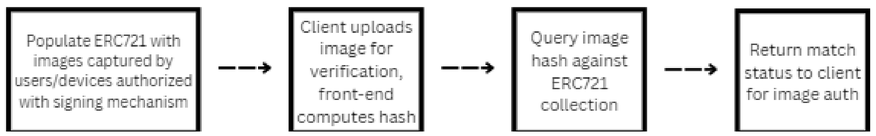
- **NFT Minting**: Whenever a photo or video is captured, we automatically mint an NFT by storing its unique hash in our database. This NFT serves as a digital fingerprint, providing a verifiable record of the media's authenticity.
- **Verification Process:** Users can easily verify the authenticity of an image or video by uploading it to our app. The app hashes the media file and checks if the resulting hash matches any NFTs in our database. If a match is found, the media is confirmed as genuine and unaltered.
- **MITM Attack Prevention:** To mitigate the risk of Man-in-the-Middle (MITM) attacks, we implement a robust authentication mechanism. Upon capturing an image or video, our app dynamically generates a secure secret key and corresponding public key pair for the user. This key pair is then used to digitally sign the image's hash and accompanying metadata, including a timestamp of the capture event.
- When the signed data is transmitted to our server, it undergoes verification. The server decrypts the signature using the user's public key and validates the timestamp against the actual capture event recorded by the device's camera API. This ensures the integrity of the image and guards against unauthorized modifications during transmission.
- By combining NFTs, hashing, and cryptographic signatures, our system provides a reliable method for verifying the authenticity of media files while safeguarding against potential tampering or manipulation. This approach not only enhances trust in the integrity of digital content but also strengthens security measures against malicious interventions.

Proof of Concept Diagrams

Signature transmission



App Pipeline



Tech Stack

The core components of this app are satisfied with the following implementations:

- Colllection of verified NFTs: ERC721 Standard Token (class of assets)
- Difference Hashing on images: OpenCV with python backend
- Deployment of ERC721 locally with Anvil chain or ganache, infura for mainnet deploy
- Android Camera API via Kotlin
- Full Stack App (Kotlin, Javascript, HTML, CSS)

It is important to note the modularity of the three main components involved in this app. The frontend defines a user interface in which a user is allowed to upload an image they would like to verify. The hash of the image is computed. The front end queries the ERC721 NFT collection for a hash matching the input image. If one is found, then the front-end replies with a authentication approval. Separately, a Kotlin app responds to events emitted when changes are made in DCIM camera output folder on an Android device. In response to such an event, it hashes the new photo written to DCIM, receives a unique signature from the ERC721 contract, and gains authorization to mint an NFT to this contract with the newly captured image’s hash

Future Work & Concerns

We hope to prevent key generation compromise: If an attacker can compromise the key generation process or obtain users' secret keys, they could impersonate legitimate users and sign fraudulent data. This could potentially lead to the creation of false NFTs or the manipulation of image authenticity verification.

We can ZK-proof the hashes and user metadata so as to increase privacy (prevent activity tracking of each user). The timestamp transmission security must be imrpoved so that we know our method to mitigate MITM works.

Additionally, we currently only have proof of concept for Android devices due to ability to track writes to the DCIM folder and hence photo capture events. This is not as explicitly available with Apple PhotoKit, so we hope to experiment with other alternatives with which we can track camera capture events for iOS devices.

Acknowledgements

Professor Viswanath, Electrical and Computer Engineering, Princeton University
Benjamin Finch, COS 473 TA
Viraj Nadkarni, COS 473 TA

References

1. EveryPixel. (n.d.). AI in Images: Statistics, Patterns, Trends. EveryPixel Journal.
2. Google Patents. (n.d.). Method and system for image forensics and authentication using blockchain technology.
3. Kiran, R., Krishnaveni, V., & Rao, G. A. (2019). Deep learning based image forgery detection techniques: A survey. Signal Processing: Image Communication, 73, 142-157. https://doi.org/10.1016/j.image.2018.11.004
4. Serelay. (n.d.). Serelay: Image Authentication for Trustworthy Content.
5. Nitulescu, A., & Baum, C. (2018). Survey on zk-SNARKs.
6. Imperva. (n.d.). Man-in-the-Middle Attack (MITM).
7. Nodle. (n.d.). Combating AI Generated Images with Photo NFTs. Medium.
8. Stack Overflow. (n.d.). Android - Listen to photo taken event.