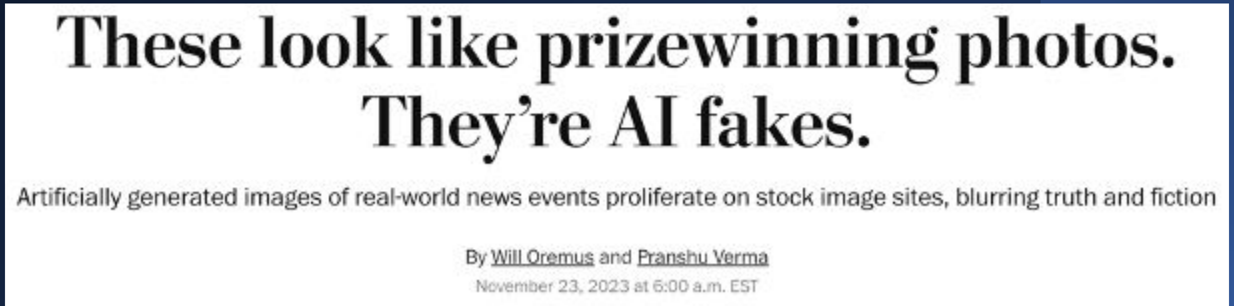
VerITAS: Verifying Image Transformations at Scale

Trisha Datta, Binyi Chen, Dan Boneh

Stanford University



These look like prizewinning photos. They're AI fakes.

Artificially generated images of real-world news events proliferate on stock image sites, blurring truth and fiction

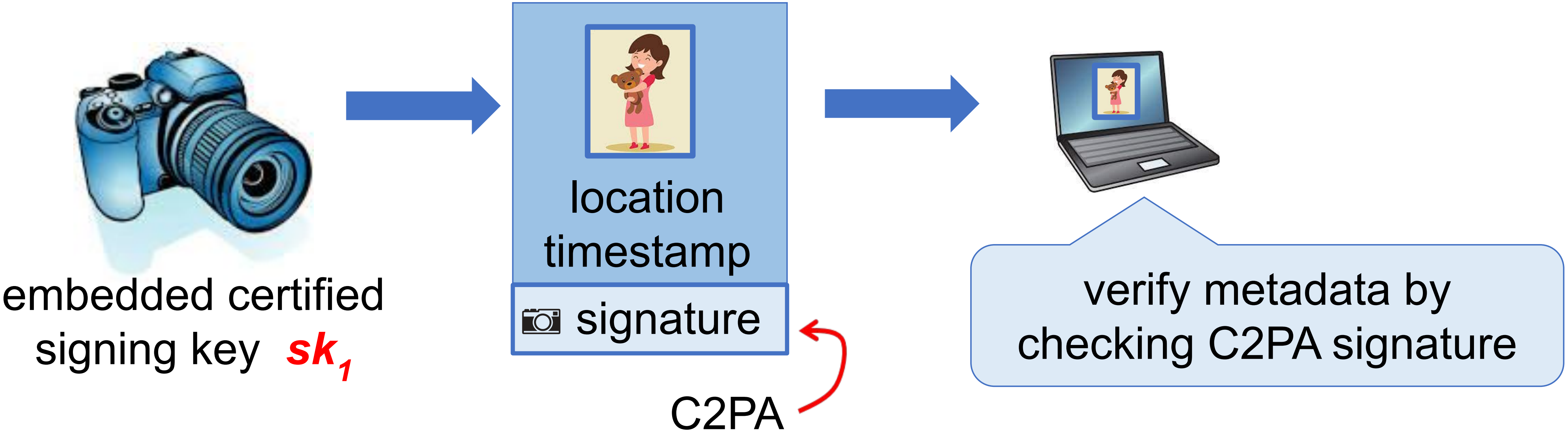
By Will Oremus and Pranshu Verma

November 23, 2023 at 6:00 a.m. EST

C2PA: A Content Provenance Standard

Nikon, Canon, Sony eye tamper-resistant digital signatures to combat deepfakes

Partnership for greater trust in digital photography: Leica and Content Authenticity Initiative



embedded certified signing key sk_1

location
timestamp

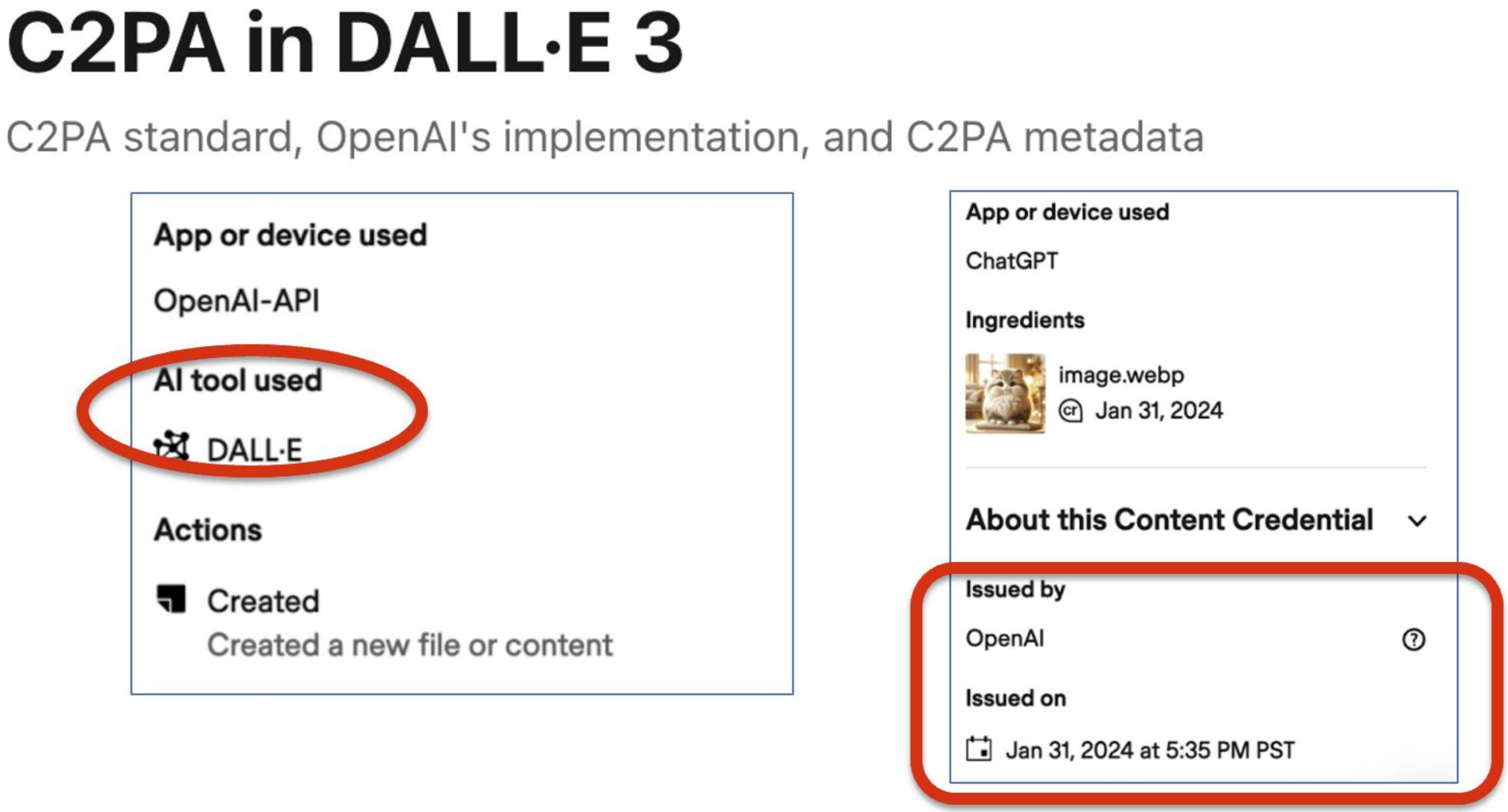
📷 signature

C2PA

verify metadata by checking C2PA signature

Adobe BBC intel Microsoft PUBLICIS GROUPE SONY Truepic

Not just news organizations…



C2PA in DALL·E 3

C2PA standard, OpenAI's implementation, and C2PA metadata

App or device used

OpenAI-API

AI tool used

DALL·E

Actions

Created

Created a new file or content

App or device used

ChatGPT

Ingredients

image.webp

Jan 31, 2024

About this Content Credential

Issued by

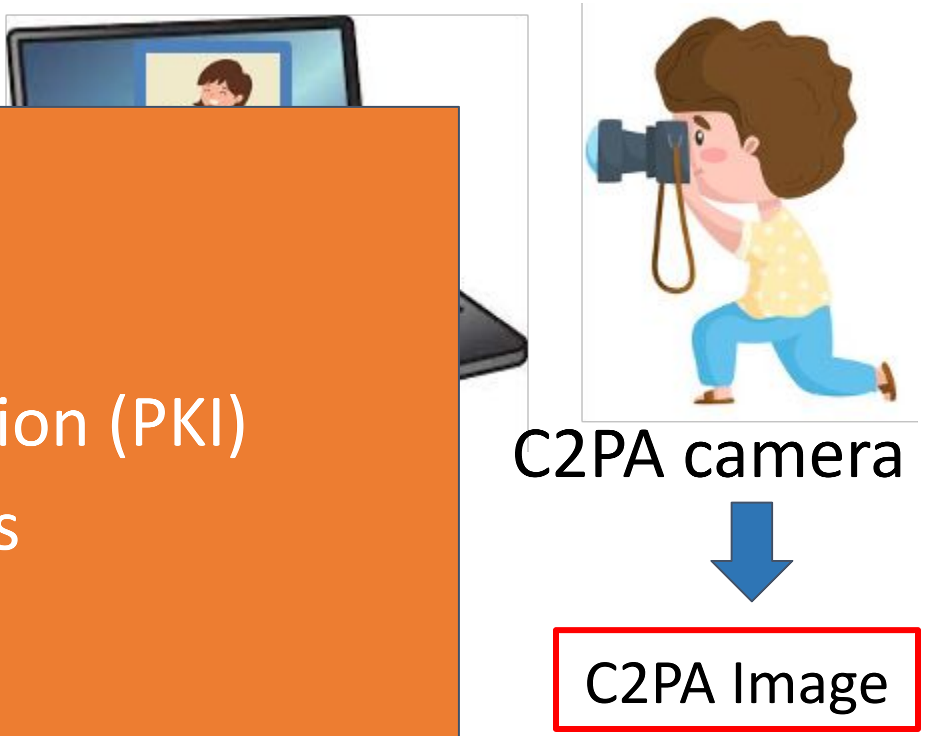
OpenAI

Issued on

Jan 31, 2024 at 5:35 PM PST

Is this a cat-and-mouse game?

The picture-of-picture attack:



Many other challenges

(1) Key extraction and revocation (PKI)

(2) Privacy ⟶ group signatures

(3) GPS spoofing

Now every v

- Is this
- Can attacker defeat the filter?

[see Chimera, UCLA, to appear at Usenix Sec 2025]

[, Rivadeneira, “Harms Modeling in the C2PA,” 2022]

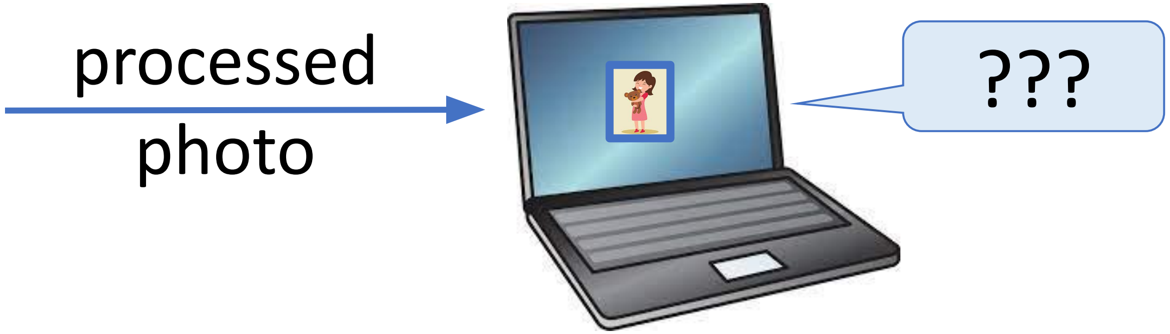
A Problem: Post-Processing

Newspapers often process photos before publication

- At minimum, need to resize (90 MB ⟶ 8 MB)
- Allowable operations from the *Associated Press:*

  cropping, grayscale, blurring…

Problem: browser cannot verify the C2PA signature of a processed photo



processed photo

???

The solution proposed by C2PA is not ideal … can we do better?

A Cryptographic Solution: zk-SNARKs!

Editor has:

Orig

📷 signature

$\underbrace{\text{Orig} \quad \text{signature}}_{\text{witness}}$

Laptop has:

Ops

(blur, resize, …)

Edited

$\underbrace{\text{Ops} \quad \text{Edited}}_{\text{statement}}$

Editor creates a proof π that:

> I know (***Orig, sig***) such that:
>
> 1. ***sig*** is a valid signature on ***Orig***
> 2. ***Edited*** is the result of applying ***Ops*** to ***Orig***
> 3. metadata(***Edited***) = metadata(***Orig***)

Do we need ZK ZK?

⇒ Laptop verifies π and shows metadata to user

How to prove?

π

I know (***Orig***, ***sig***) such that:

1. ***sig*** is a valid signature on ***Orig***
2. ***Edited*** is the result of applying ***Ops*** to ***Orig*** **?**
3. metadata(***Edited***) = metadata(***Orig***)

Verifying Edits in a SNARK Prover

- PhotoProof (Naveh and Tromer, 2016): a few minutes to generate photo editing proofs for 128 x 128 pixel image

- New tools enable faster development and bigger statements!
  - Plonky2 library (“Plonk PIOP” + FRI PCS)
    - Write arithmetic circuit C_{edit} s.t. $C_{\text{edit}}($***Orig***$)$ = ***Edited***

Our work: proof for a 6000 x 4000 image using Plonky2

- resize, crop, grayscale, blur ⟶ proof gen. time ≤ 4 minutes
- Proof size: ≈100 KB (≪image size), verification time: 0.7 sec

(can shrink proof with recursion)

Verifying Signatures in a SNARK Prover

π

I know (***Orig***, ***sig***) such that:

1. ***sig*** is a valid signature on ***Orig*** ?
2. ***Edited*** is the result of applying ***Ops*** to ***Orig*** ✓
3. metadata(***Edited***) = metadata(***Orig***)

Problem: the SNARK proof must check that a pair (***Orig***, **h**) satisfies **h** = hash(***Orig***)...but ***Orig*** is 90MB!

Verifying Signatures in a SNARK Prover

Attempt 1

Verify Hash

I know (***Orig***, ***hash***) such that:

hash = SHA256(***Orig***)

Too slow for 90 MB!

Attempt 2

Verify Hash

I know (***Orig***, ***hash***) such that:

hash = Poseidon(***Orig***)

SNARK-friendly hash
…but still too slow for 90MB!

We propose two methods:

(1) Verify Lattice Hash

I know (***Orig***, ***hash***) such that:

hash = Poseidon(LatticeHash(***Orig***))

good for camera and prover

(2) Verify a PCS

I know (***Orig***, ***hash***) such that:

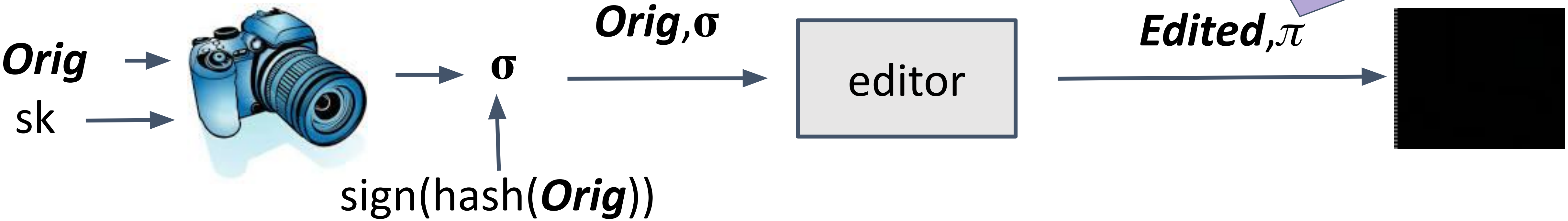
hash = **PCS**(***Orig***)

great for prover!

Verifying Signatures in a SNARK Prover

Method 1: lattice hash

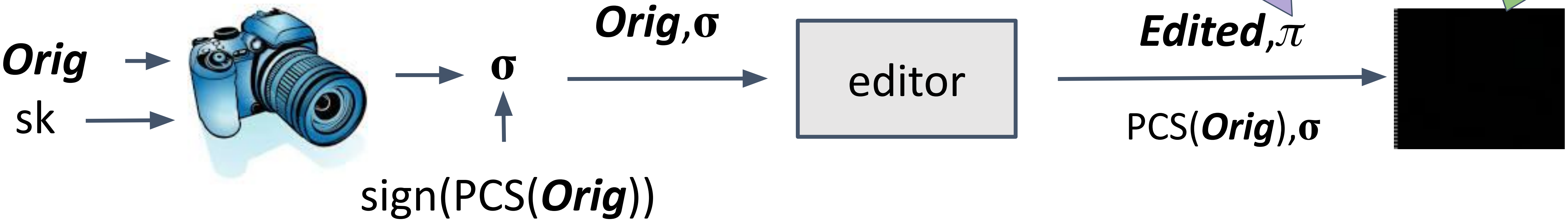
π proves edits and valid sig



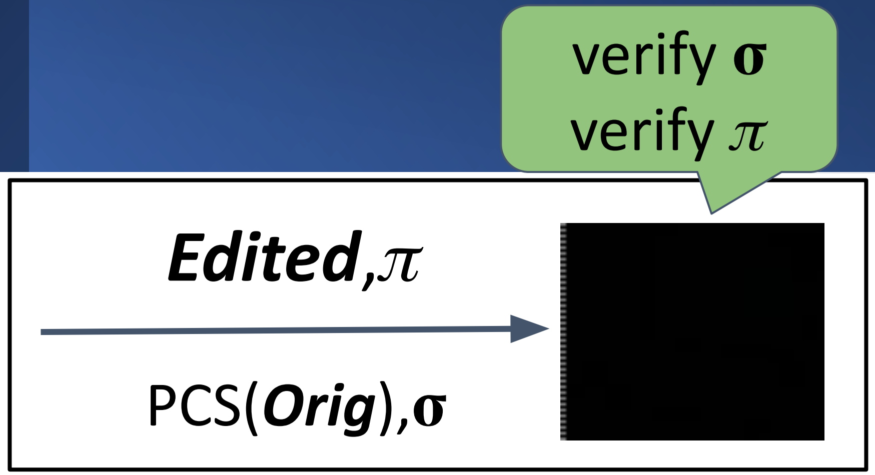
Method 2: use polynomial commitment scheme (PCS)

π proves edits only

verify **σ**
verify π



How is Method 2 secure?


verify $\boldsymbol{\sigma}$
verify π
Edited,π
PCS(***Orig***),$\boldsymbol{\sigma}$

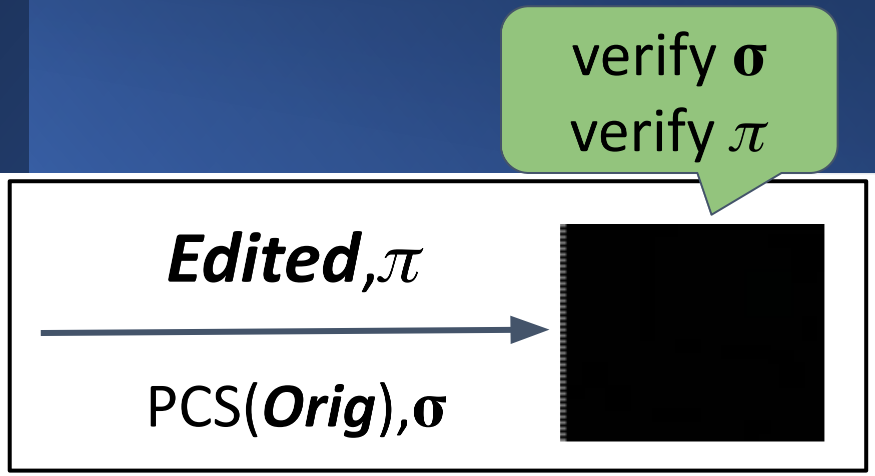
PLONK proof π proves that C_{edit}(***Orig’***) = ***Edited*** where ***Orig’*** is provided as witness data

Problem: what if ***Orig*** ≠ ***Orig’*** ??

→ Then edited image is unrelated to camera’s image

Solution: non-black-box use of PLONK!

How is Method 2 secure?



PLONK proof π proves that C_{edit}(***Orig'***) = ***Edited***
where ***Orig'*** is provided as witness data

Partial explanation of how to produce PLONK proof π:

1. Encode C_{edit} execution tableaux as a polynomial

T(x)

↓

Some ***T***(x) evaluations encode the witness ***Orig'***

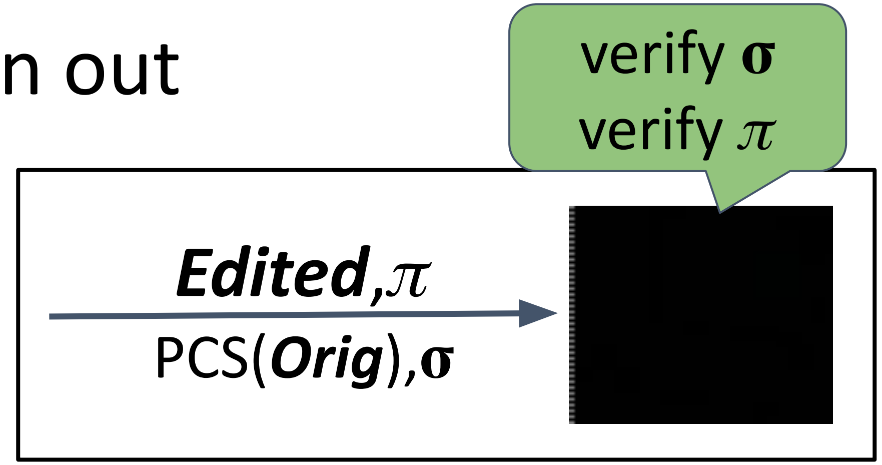
2. Generate proof:

a. Compute ***com***$_{\boldsymbol{T}}$ = PCS(***T***(x))
b. Prove gates evals in tableaux are correct
c. Prove circuit wiring in tableaux is correct

This works even if tableaux is committed via multiple polynomials!

Tradeoffs of Signing PCS vs. Lattice Hash

Signing PCS(***Orig***) takes signature verification out of the SNARK circuit


verify $\boldsymbol{\sigma}$
verify π
Edited,π
PCS(***Orig***),$\boldsymbol{\sigma}$

… but computing a PCS commitment is not feasible on a commercial camera

- Suitable for a cloud AI image generator
- Can be offloaded to an untrusted server

Verifying Signatures in a SNARK Prover

π

I know (***Orig***, ***sig***) such that:

1. ***sig*** is a valid signature on ***Orig*** ✓
2. ***Edited*** is the result of applying ***Ops*** to ***Orig*** ✓
3. metadata(***Edited***) = metadata(***Orig***)

Conclusions

Succinct proofs have become practical and easy to use

- An amazing success of theory of CS
- Development driven by blockchain

  but **many** non-blockchain applications

C2PA: a playground for many cryptographic techniques

- Many challenges to explore…

IEEE S&P'25: https://eprint.iacr.org/2024/1066.pdf