Efficient Privacy-Preserving Proofs for Image Transformations

Pierpaolo Della Monica
Sapienza University of Rome

Ivan Visconti
Sapienza University of Rome

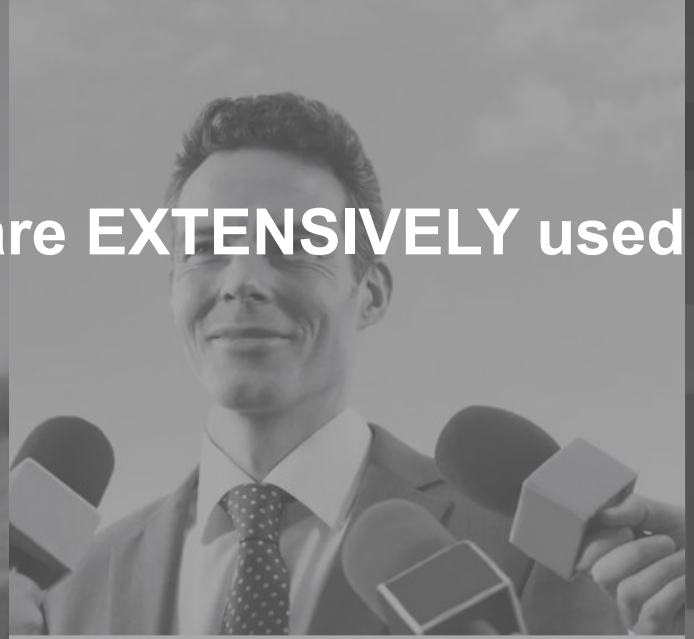
Andrea Vitaletti
Sapienza University of Rome

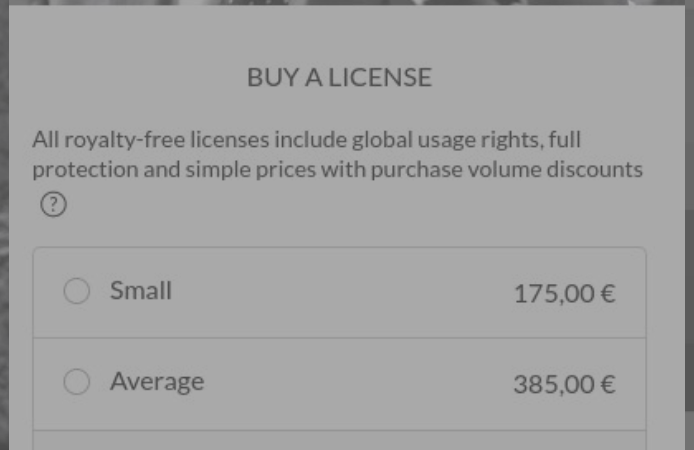
Marco Zecchini
Sapienza University of Rome

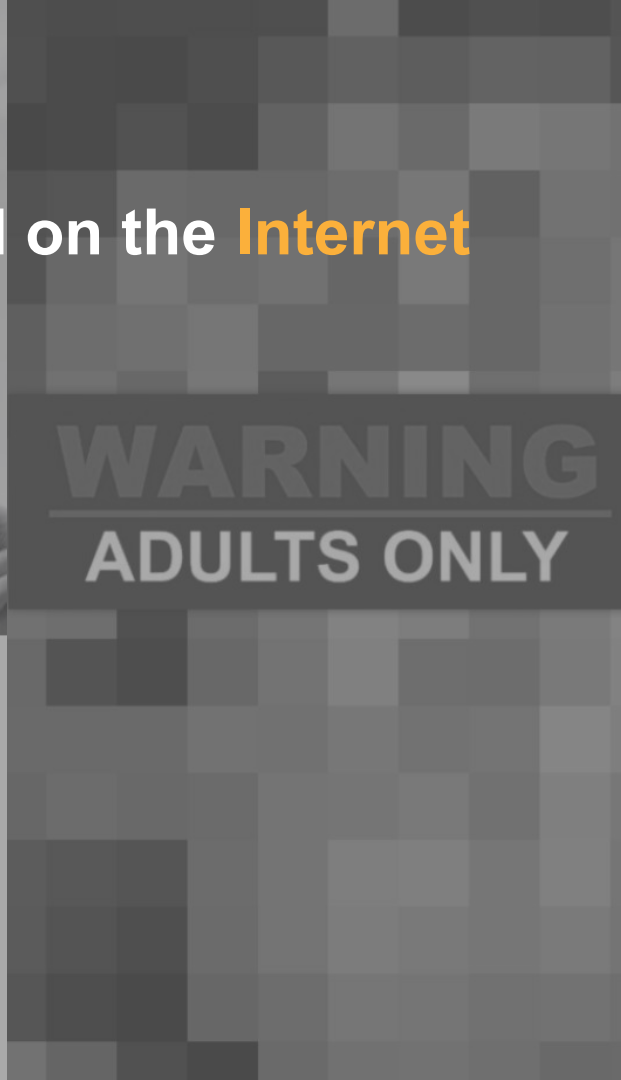
Accepted at: 46th IEEE Symposium on Security and Privacy (**IEEE S&P**)

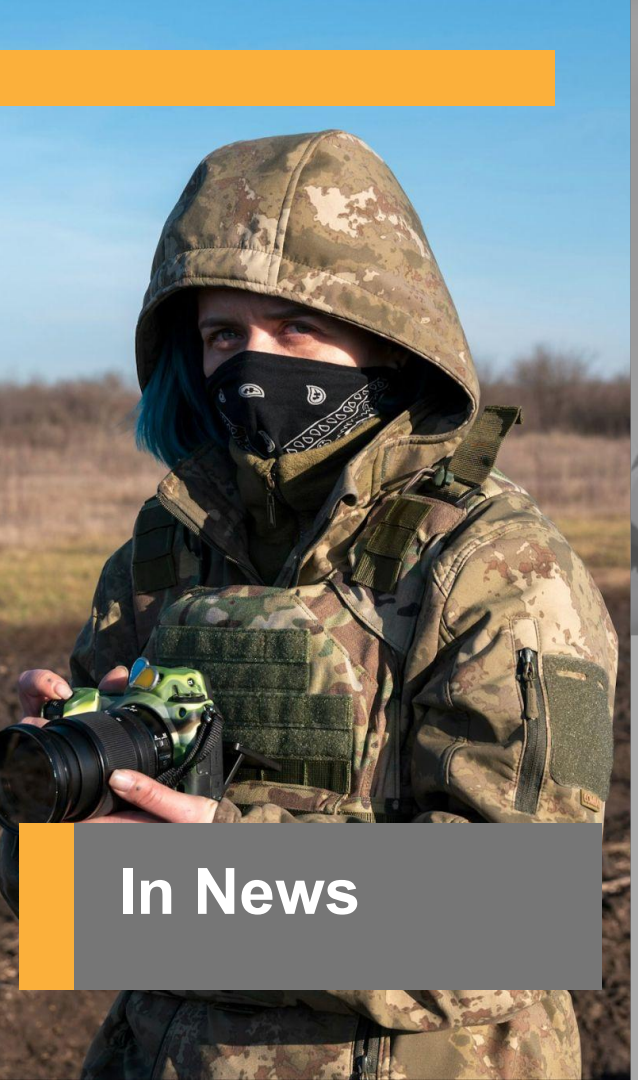
April 16, 2025

IMAGES are EXTENSIVELY used on the Internet









In News

BUY A LICENSE

All royalty-free licenses include global usage rights, full protection and simple prices with purchase volume discounts

Small	175,00 €
Average	385,00 €

WARNING

ADULTS ONLY

BUY A LICENSE

All royalty-free licenses include global usage rights, full protection and simple prices with purchase volume discounts

In Photo Agency

○ Average	385,00 €

WARNING

ADULTS ONLY

BUY A LICENSE

All royalty-free licenses include global usage rights, full protection and simple prices with purchase volume discounts

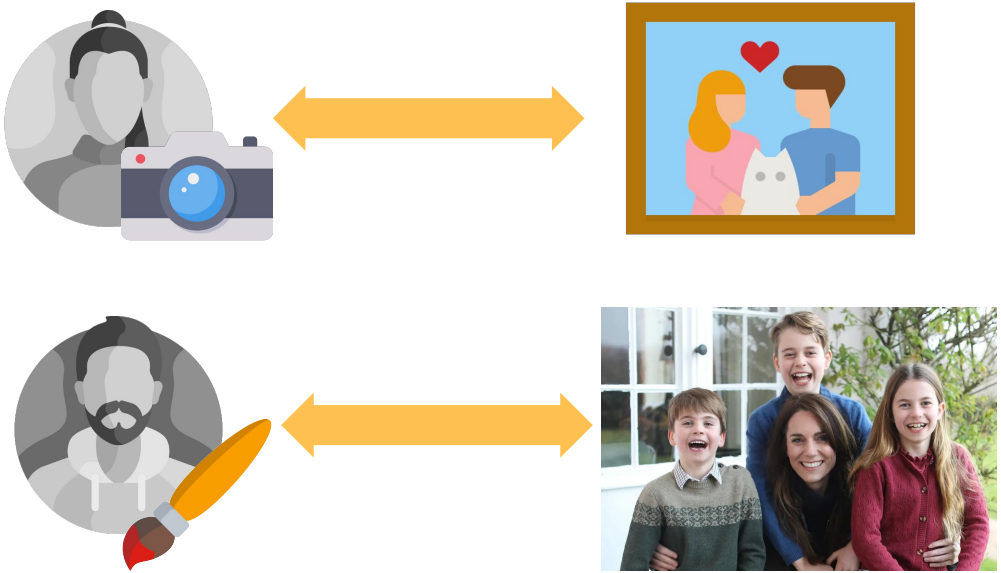
Small	175,00 €
Average	385,00 €

WARNING

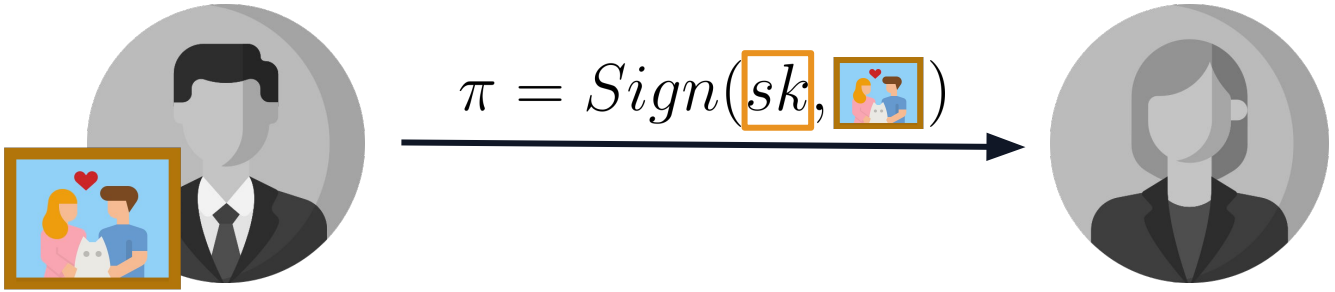
ADULTS ONLY

In Adult Sites

AUTHENTICITY
of **images** is important



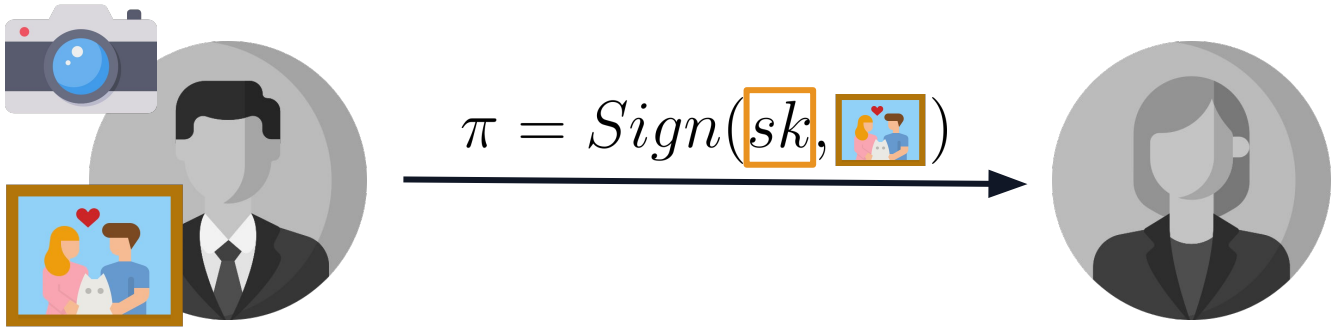
Assuming that a photo is supposed to be published as it is



AUTHENTICITY:

Signature schemes suffice

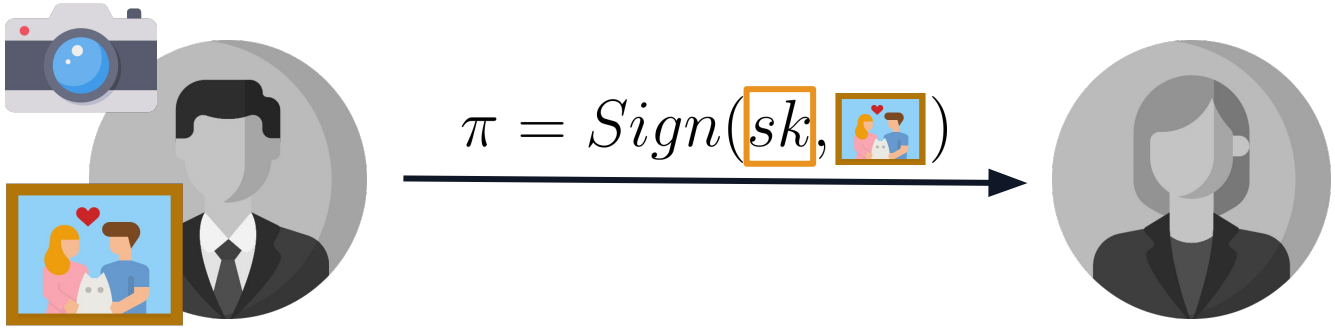
Assuming that a photo is supposed to be published as it is



AUTHENTICITY:

Signature schemes suffice

Assuming that a photo is supposed to be published as it is



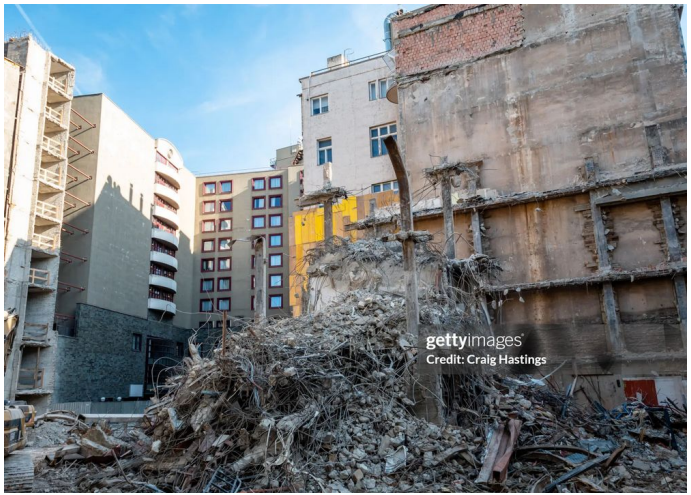
C2PA

AUTHENTICITY:
Signature schemes suffice

Adobe BBC Google intel Microsoft PUBLICIS GROUPE SONY truepic

But online **images** are edited…

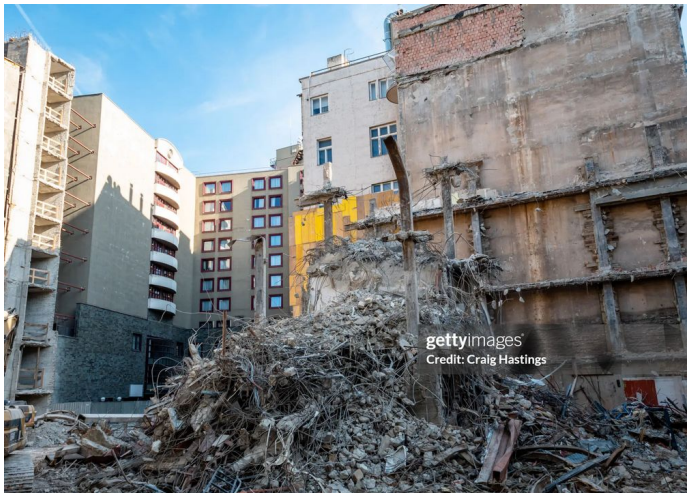
gettyimages®



Show preview of original images with watermarks and smaller dimension

But online **images** are edited…

gettyimages®



Show preview of original images with watermarks and smaller dimension

BBC NEWS

Censored images for privacy

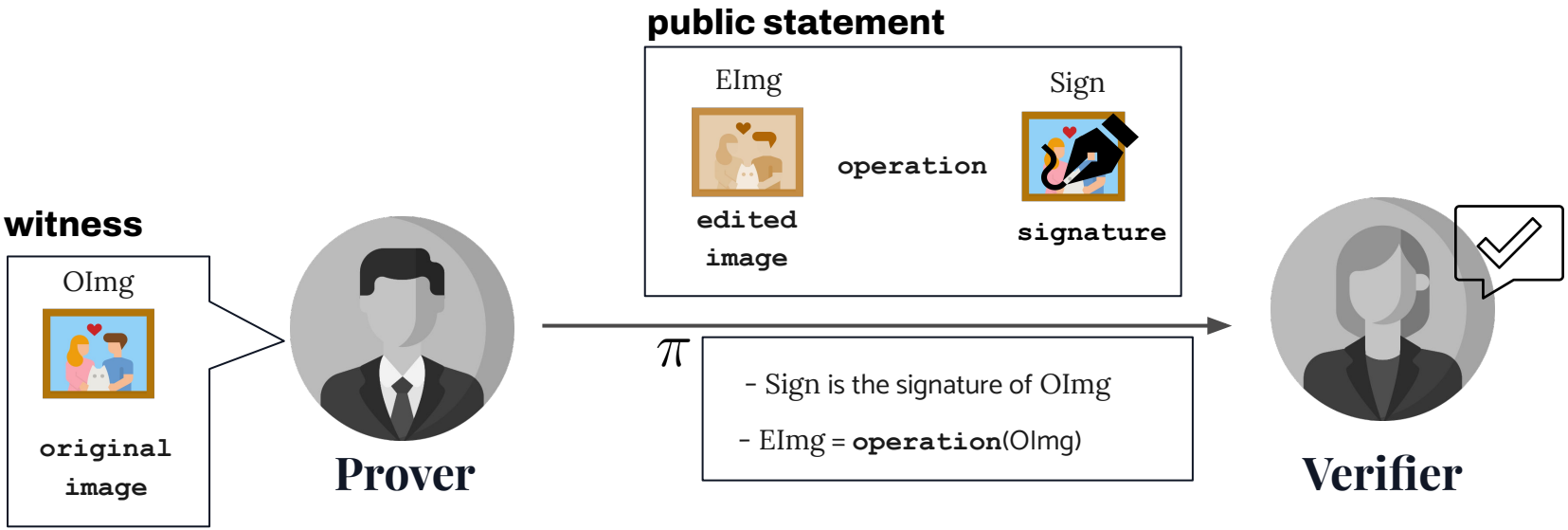
But online **images** are edited…

AUTHENTICITY

It is **not** possible to **verify** the C2PA **signature** of an edited **image** without accessing the original one

AUTHENTICITY:

a **ZK-SNARK** to **link** two images, an **original (and secret) image** and the corresponding **edited (and known) image**



AUTHENTICITY:

a **ZK-SNARK** to **link** two images, an **original (and secret) image** and the corresponding **edited (and known) image**.

The signature is computed on the cryptographic hash of a large image, and thus the statement refers to a huge (and tough) computation over a large input; this represents **by far** the most **demanding computation** for the Prover

public statement

EImg

edited image

operation

Sign

signature

witness

OImg

original image

Prover

π

- Sign is the signature of OImg
- EImg = operation(OImg)

Verifier

[NT S&P2016]
Image authenticity through cryptography.
Extremely computational intensive
(e.g., tests on 128x128 images)

[NT S&P2016] *A. Naveh and E. Tromer, "**PhotoProof: Cryptographic Image Authentication for Any Set of Permissible Transformations**" - S&P - 2016*

[NT S&P2016]
Image authenticity through cryptography.
Extremely computational intensive
(e.g., tests on 128x128 images)

[KHSS 2022]
Image authenticity adopting digital signatures from cameras but **deviating from C2PA (2021) standard**.

Tests on HD image either missing **confidentiality** (computing on AWS) or relying on **HPC**.

[NT S&P2016] *A. Naveh and E. Tromer, "**PhotoProof: Cryptographic Image Authentication for Any Set of Permissible Transformations**" - S&P - 2016*
[KHSS 2022] *D. Kang, T. Hashimoto, I. Stoica, and Y. Sun, "**ZK-IMG: Attested Images via Zero-Knowledge Proofs to Fight Disinformation.**" - arXiv.org - 2022*

[NT S&P2016]
Image authenticity through cryptography.
Extremely computational intensive
(e.g., tests on 128x128 images)

[DB RWC2023]
Image authenticity adopting Lattice Hash and Poseidon Hash for digital signatures.

Test on 30 MP image but **significant requirements on the computing platform.**

[KHSS 2022]
Image authenticity adopting digital signatures from cameras but **deviating from C2PA (2021) standard**.

Tests on HD image either missing **confidentiality** (computing on AWS) or relying on **HPC**.

[NT S&P2016] *A. Naveh and E. Tromer, "**PhotoProof: Cryptographic Image Authentication for Any Set of Permissible Transformations**" - S&P - 2016*
[KHSS 2022] *D. Kang, T. Hashimoto, I. Stoica, and Y. Sun, "**ZK-IMG: Attested Images via Zero-Knowledge Proofs to Fight Disinformation.**" - arXiv.org - 2022*
[DB RWC2023] *T. Datta and D. Boneh, "**Using zk-proofs to fight disinformation**" - RWC - 2023*

[NT S&P2016]
Image authenticity through cryptography.
Extremely computational intensive
(e.g., tests on 128x128 images)

[DB RWC2023]
Image authenticity adopting Lattice Hash and Poseidon Hash for digital signatures.

Test on 30 MP image but **significant requirements on the computing platform.**

[KHSS 2022]
Image authenticity adopting digital signatures from cameras but **deviating from C2PA (2021) standard**.

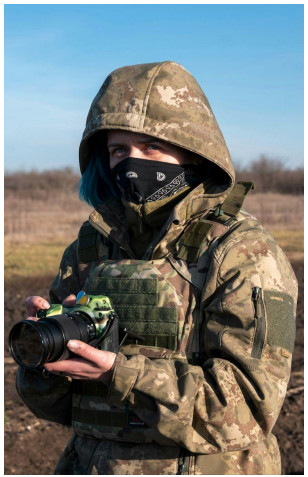
Tests on HD image either missing **confidentiality** (computing on AWS) or relying on **HPC**.

[LHCLCC MIPR2023]
Image authenticity proves correctness of a transformation considering only a small portion of an image.

Experimental results **similar** to ***[KHSS 2022]*** when the entire image is involved.

[NT S&P2016] *A. Naveh and E. Tromer, "**PhotoProof: Cryptographic Image Authentication for Any Set of Permissible Transformations**" - S&P - 2016*
[KHSS 2022] *D. Kang, T. Hashimoto, I. Stoica, and Y. Sun, "**ZK-IMG: Attested Images via Zero-Knowledge Proofs to Fight Disinformation.**" - arXiv.org - 2022*
[DB RWC2023] *T. Datta and D. Boneh, "**Using zk-proofs to fight disinformation**" - RWC - 2023*
[LHCLCC MIPR2023] *K. Li, C. Hsu, M. Chang, F. Liu, S. Chien, and W. Chen, "**Region-aware photo assurance system for image authentication**" - MIPR - 2023*

DO WE ALWAYS NEED **SUCCINCTNESS** ?



In news websites

DO WE ALWAYS NEED **SUCCINCTNESS** ?

In news websites

DO WE ALWAYS NEED **SUCCINCTNESS** ?

In news websites

BUY A LICENSE

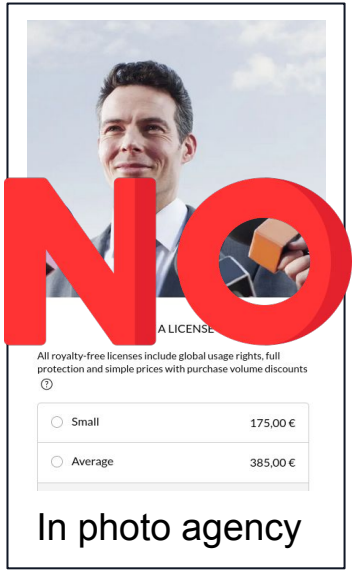
All royalty-free licenses include global usage rights, full protection and simple prices with purchase volume discounts

Small	175,00 €
Average	385,00 €

In photo agency

DO WE ALWAYS NEED **SUCCINCTNESS** ?

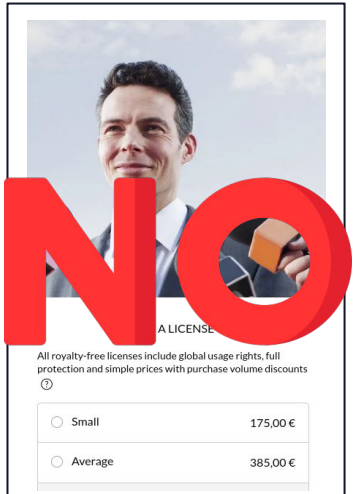
In news websites



In photo agency

DO WE ALWAYS NEED **SUCCINCTNESS** ?

In news websites

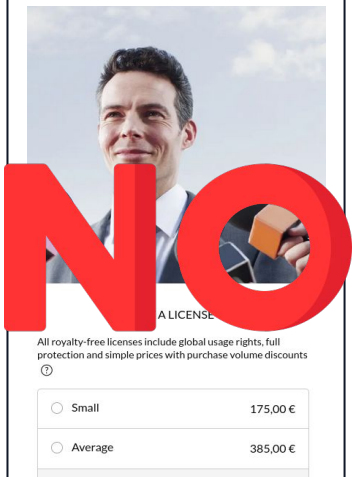


In photo agency

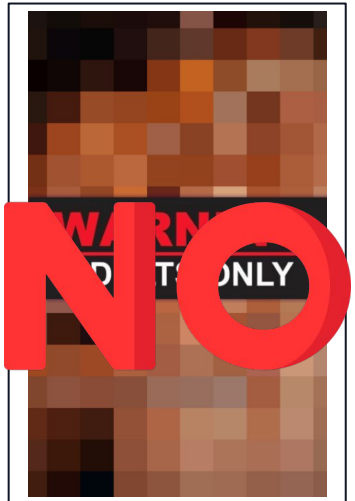
In adult sites

DO WE ALWAYS NEED **SUCCINCTNESS** ?

In news websites



In photo agency



In adult sites

SUCCINCTNESS OF THE PROOF IS OFTEN AN **OVERKILL** IN SEVERAL SCENARIOS AND A SUCCINCT FRAUD PROOF CAN BE GOOD ENOUGH

Our Results (IEEE S&P 2025)

We propose a system to prove **image authenticity guaranteeing**:

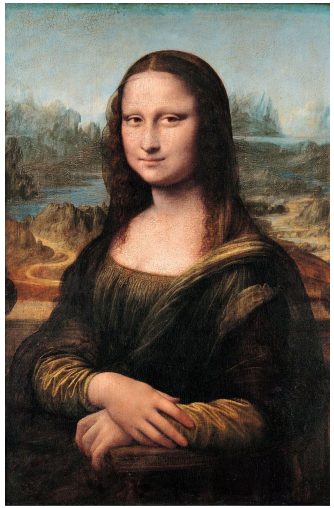
1. **Low memory consumption** for the **prover** (no HPC, your laptop is just fine)

2. **Succinct Fraud Proofs** fast verification for usability (e.g., browsers) and compactness for blockchains

3. **Confidentiality** of the original **image** (no cloud infr.) and **authenticity** of the transformed image defined and proved (starting with [NT S&P2016])

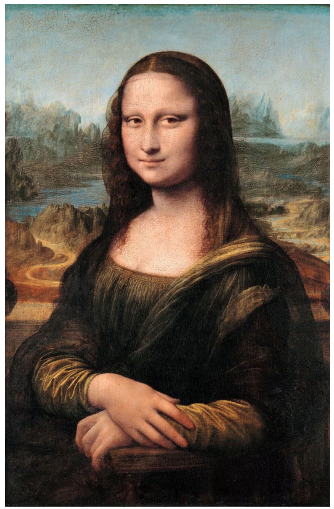
4. **It works with SHA256, used by C2PA** standard (at an additional, still affordable, cost for proof computation and size)

Image **Tiling**

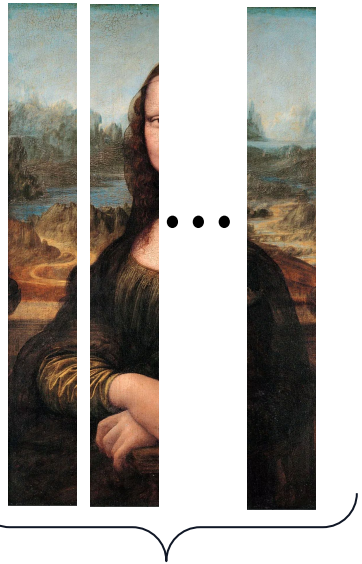


*For large images proving knowledge of a pre-image of the hash is the real **bottleneck***

Image **Tiling**



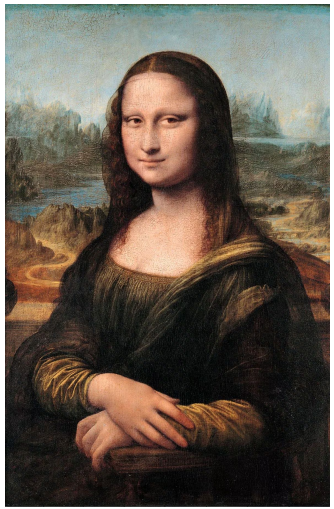
*For large images proving knowledge of a pre-image of the hash is the real **bottleneck***



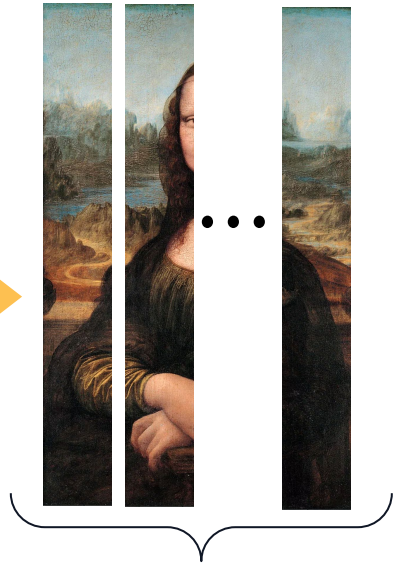
Each tile has a reduced dimension and it is possible to split the computational effort

This methodology consists of **splitting** the image into several smaller **tiles.**
For each tile, a **ZKP** can be defined, enabling **hashing** for a **shorter witness** and producing multiple hashes that represent different subimages.

Image **Tiling**



*For large images proving knowledge of a pre-image of the hash is the real **bottleneck***

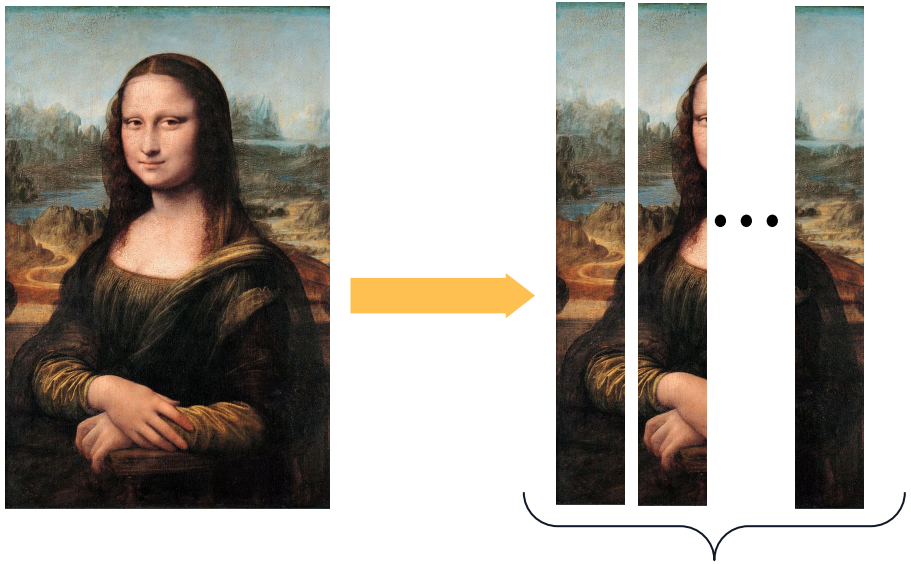


Each tile has a reduced dimension and it is possible to split the computational effort

This methodology consists of **splitting** the image into several smaller **tiles.**
For each tile, a **ZKP** can be defined, enabling **hashing** for a **shorter witness** and producing multiple hashes that represent different subimages.

It is important that the transformation of the full image can be computed working locally tile by tile. Many natural transformations follow this approach.

Image **Tiling**



*For large images proving knowledge of a pre-image of the hash is the real **bottleneck***

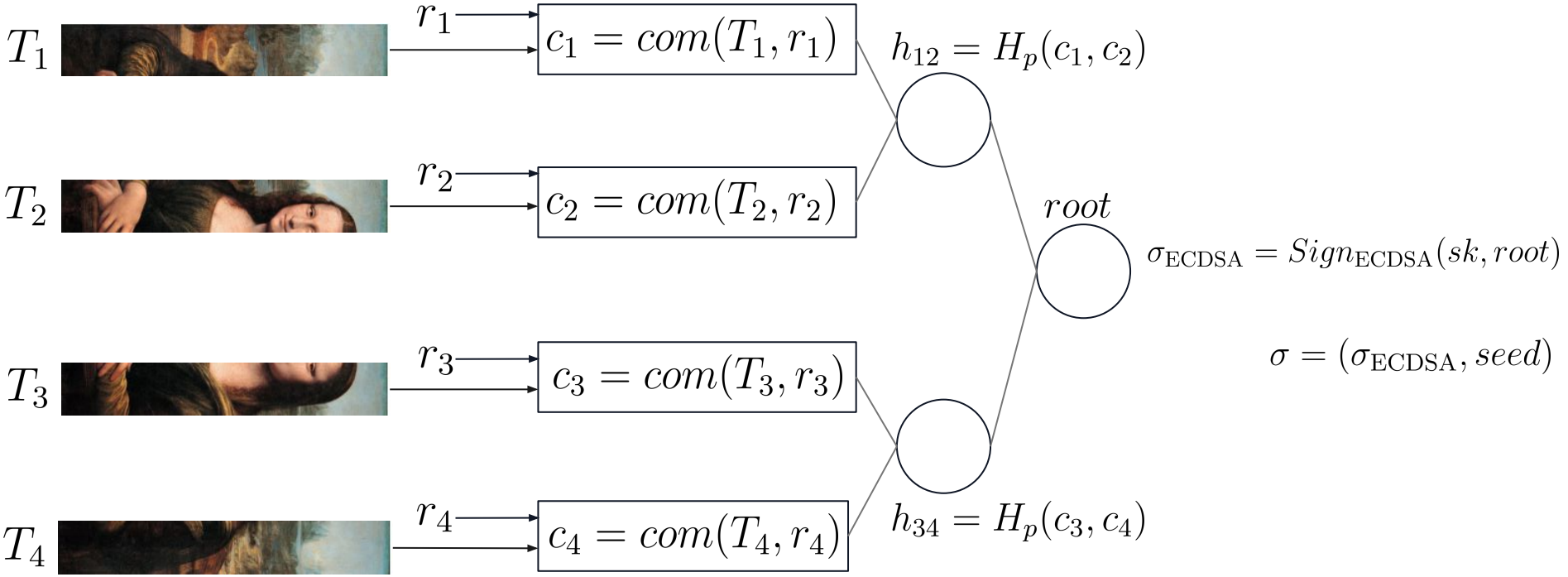
Each tile has a reduced dimension and it is possible to split the computational effort

This methodology consists of **splitting** the image into several smaller **tiles.**
For each tile, a **ZKP** can be defined, enabling **hashing** for a **shorter witness** and producing multiple hashes that represent different subimages.

The camera signs the entire picture only.

The **Signature** Scheme

$r_i = PRF(seed, i)_{i \in \{1,2,3,4\}}$



The **Signature** Scheme

$r_i = PRF(seed, i)_{i \in \{1,2,3,4\}}$

r_1

$c_1 = com(T_1, r_1)$

r_2

$c_2 = com(T_2, r_2)$

r_3

$c_3 = com(T_3, r_3)$

r_4

$c_4 = com(T_4, r_4)$

$h_{12} = H_p(c_1, c_2)$

$h_{34} = H_p(c_3, c_4)$

$root$

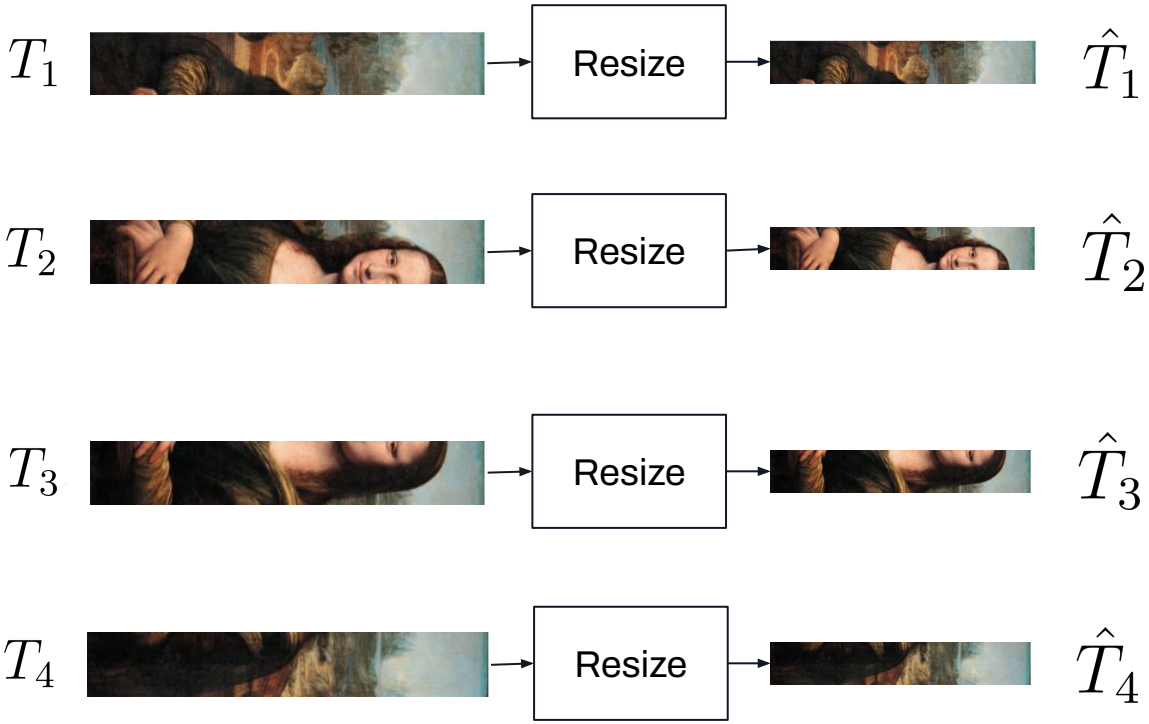
$\sigma_{\text{ECDSA}} = Sign_{\text{ECDSA}}(sk, root)$

$\sigma = (\sigma_{\text{ECDSA}}, seed)$

$B_3 := [c_4, h_{12}]$

Represents the Merkle Branch to verify c_3

Local **Transformation**



Proof generation

For $i \in 1, ..., 4$ **then**

$$x_i := (\text{Resize}(\cdot), \hat{T}_i, c_i)$$

$$\pi_i$$

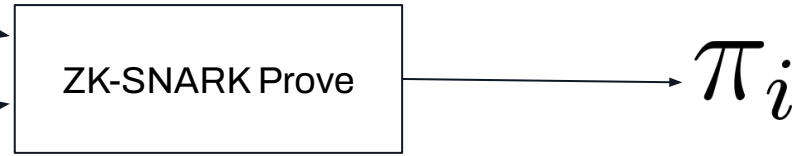
$$w_i := (T_i, r_i)$$

$$\Big(c_i = com(T_i, r_i) \wedge \hat{T}_i = \text{Resize}(T_i)\Big)$$

Proof generation

For $i \in 1, ..., 4$ **then**

$$x_i := (\text{Resize}(\cdot), \hat{T}_i, c_i)$$



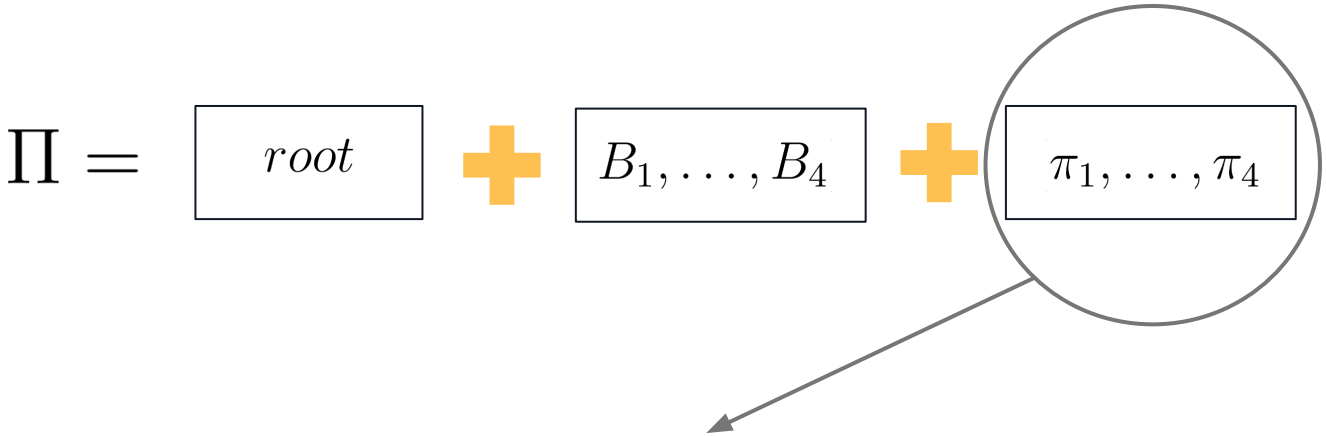
$$\pi_i$$

$$w_i := (T_i, r_i)$$

$$\Big(c_i = com(T_i, r_i) \wedge \hat{T}_i = \text{Resize}(T_i)\Big)$$

$$\prod = \boxed{root} \quad \boxed{B_1, \ldots, B_4} \quad \boxed{\pi_1, \ldots, \pi_4}$$

Proof verification and Fraud proof

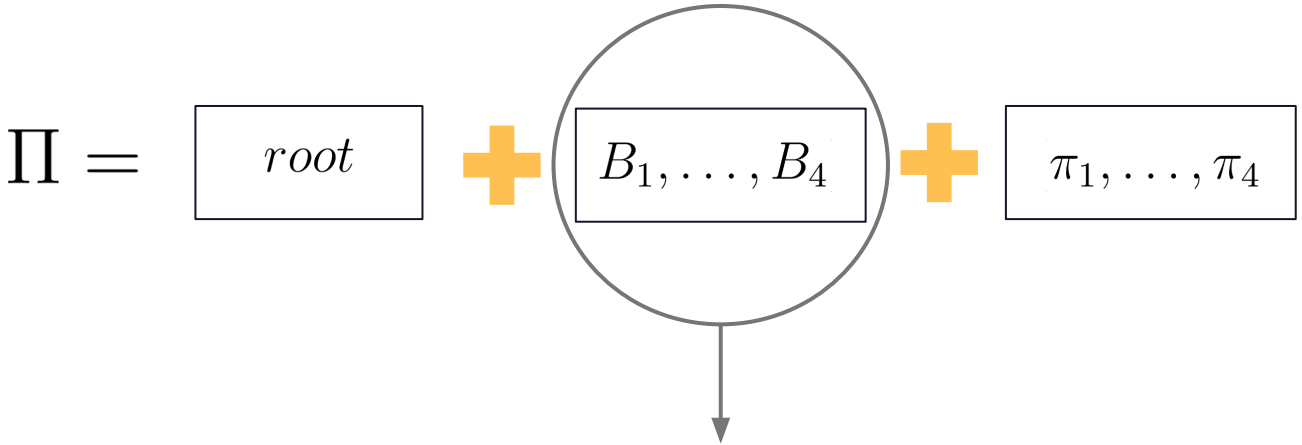


For $i \in 1, ..., 4$ **then**

$$VerifyProof(vk_i, x_i, \pi_i)$$

If not correct, provide π_i as a **FRAUD PROOF**

Proof verification and Fraud proof

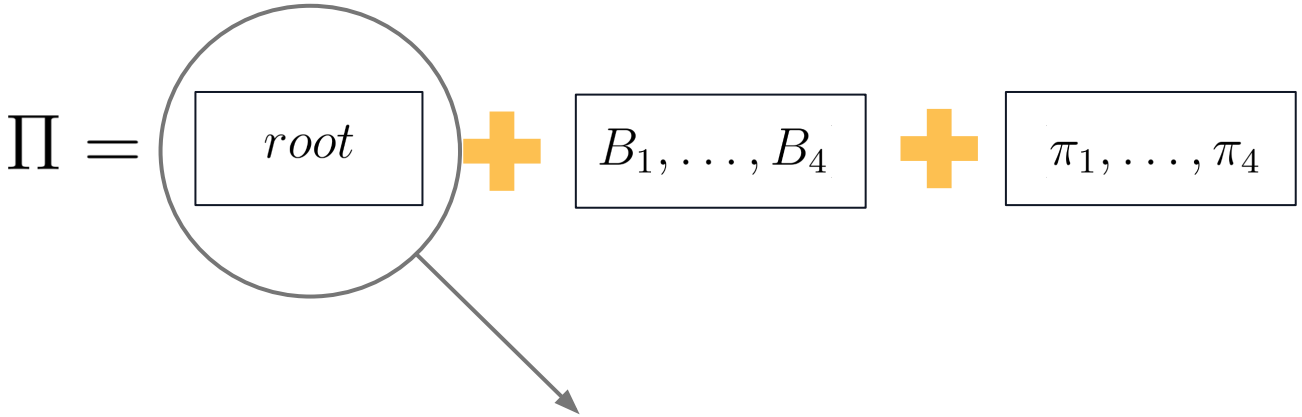


For $i \in 1, ..., 4$ **then**

$$VerifyLeaf(c_i, root, B_i)$$

If not correct, provide B_i as a **FRAUD PROOF**

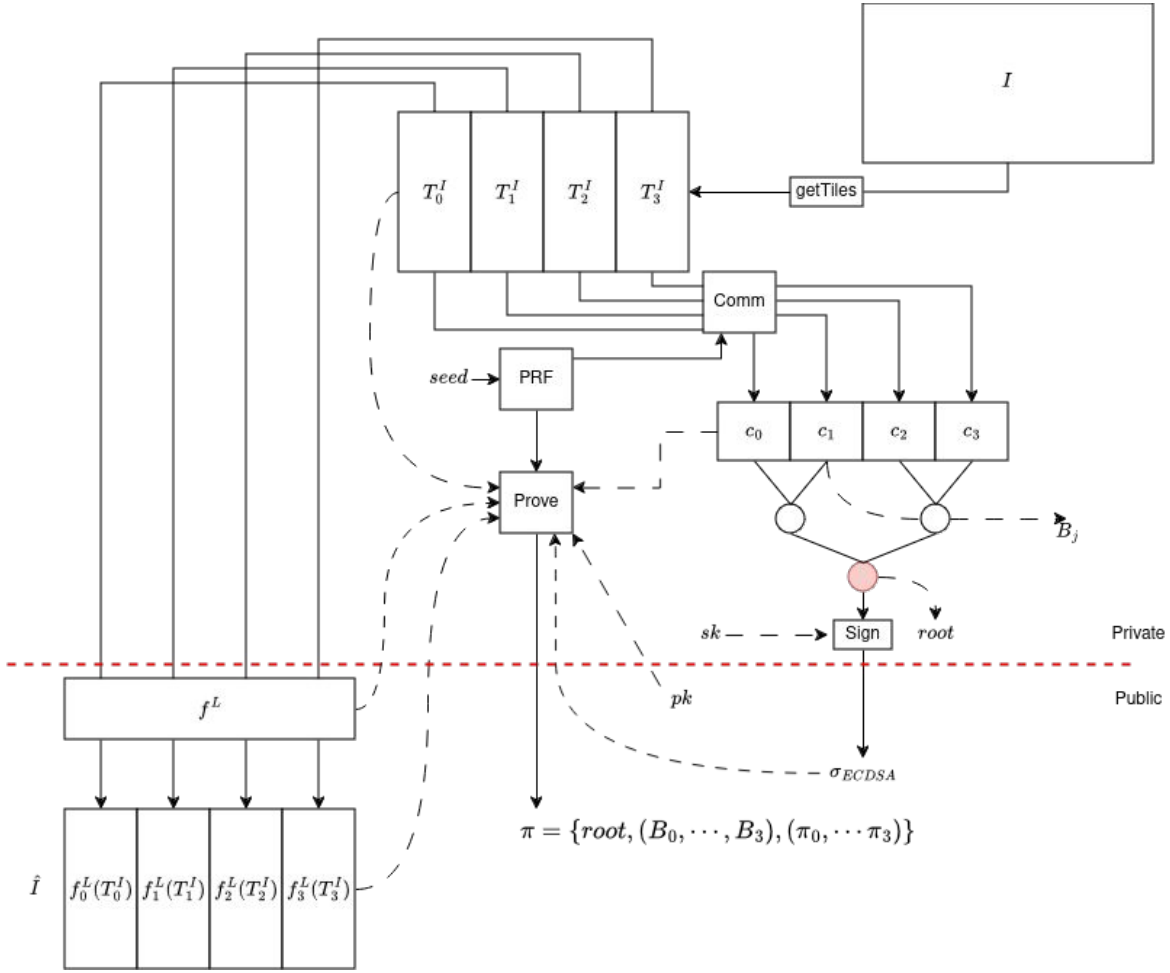
Proof verification and Fraud proof



If not correct provide $\sigma_{ECDSA}, root$ as a **FRAUD PROOF**

Architecture overview for an **image** divided into **4 tiles**

Note that some transformations (e.g., crop) do not require ZK-SNARKs for all tiles.



Our Adaptive PoK Definition

Proof of Knowledge (PoK): Φ *has the* Proof of Knowledge *property for an auxiliary input distribution* $\mathcal{Z}$, *if for every PPT* $\mathcal{A}$ *there exists a PPT* extractor Ext *and a negligible function* negl *such that the following probability is at most* $\mathsf{negl}(\lambda)$

$$\Pr\left[\begin{array}{c} \mathsf{IHVerify}(\mathsf{crs}, x, \pi) = 1 \bigwedge \\ \hat{I} \neq f^L(I_j), 1 \leq j \leq m \bigwedge \\ (\mathsf{VerifySign}(\mathsf{pk}^*, I, \sigma) = 0 \vee f^L(I) \neq \hat{I}) \end{array} \middle| \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{IHSetup}(1^\lambda) \; ; (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda) \\ aux_Z \leftarrow \mathcal{Z}(\mathsf{crs}) \\ (x = (\hat{I}, \mathsf{pk}^*, f^L), \pi) \\ \leftarrowtail \mathcal{A}^{O_{Sign}(\mathsf{sk}, \cdot)}(\mathsf{crs}, aux_Z, \mathsf{pk}) \\ (I, \sigma) \leftarrow \mathsf{Ext}(\mathsf{crs}, aux_Z, \mathsf{pk}, \mathsf{qt}) \end{array}\right]$$

where $\mathsf{qt} = \{I_j, \sigma_j\}$, *with* $|\mathsf{qt}| = m$, *is the transcript of all queries to the signature oracle* O_{Sign} *and its answers, specifically* I_j *is the* j*-th query and* σ_j *(i.e., the signature of* I_j *using* sk*) is the* j*-th answer.*

Our Adaptive Hiding Experiment

$$ExpImageIndistinguishability_{\mathcal{A},R}^{O_{Sign},O_T}(\lambda)$$

$\mathsf{crs} \leftarrow \mathsf{IHSetup}(1^\lambda)$; $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$

$(I_0, I_1) \leftarrow \mathcal{A}^{O_{Sign}(\mathsf{sk},\cdot)}(\mathsf{pk}, \mathsf{crs})$; $b \leftarrow\$ \{0,1\}$

$b' \leftarrow \mathcal{A}^{[O_{Sign}(\mathsf{sk},\cdot);\ O_T(\mathsf{crs},\mathsf{sk},I_b,\cdot)]}(\mathsf{pk}, \mathsf{crs})$

If f *passed as input to* O_T *is such that* $f(I_0) \neq f(I_1)$

$\vee\ I_0, I_1 \notin \mathcal{I}_{N,M}$ ***then:*** **return** 0

return $(b == b')$

Experiments

Our approach is **generic** and can be instantiated with different ZK proofs.

The following experiments were conducted using **Groth16** as ZK-SNARK instantiation, facilitating a **comparison** with the contemporary **state-of-the-art performance** and outcomes.

Experiments - Technical choices

We used **circom** and **snarkjs** to compile and setup the circuit on Groth16

- 3 transformations (bilinear resize, grayscale and crop)

Rapidsnark to parallelize the proof generation on AMD/Intel CPU (no Apple CPUs)

- Among the best performer to generate SNARKs according to Celer.network analysis

circom
CIRCUIT COMPILER

Experiments - Further optimizations

We used optimizations proposed by Khovratovich to optimize Poseidon circuits with large input. In particular:

For bytes:

1. Pad the byte string with byte `0x7`, then with zero bytes up to the multiple of 28.
2. Split the string into 28-byte chunks.
3. Assign each chunk to the scalar.

Optimal tile size?

Time and memory consumptions in the proof generation computing a Poseidon hash and a SHA256 compression are **linear** in the size of the input (as long as no **swap**).

Experiments

FEASIBILITY ON 30MP IMAGE

We run the test on Intel i7@1.8 GHz, 8 cores and 16 GB of RAM

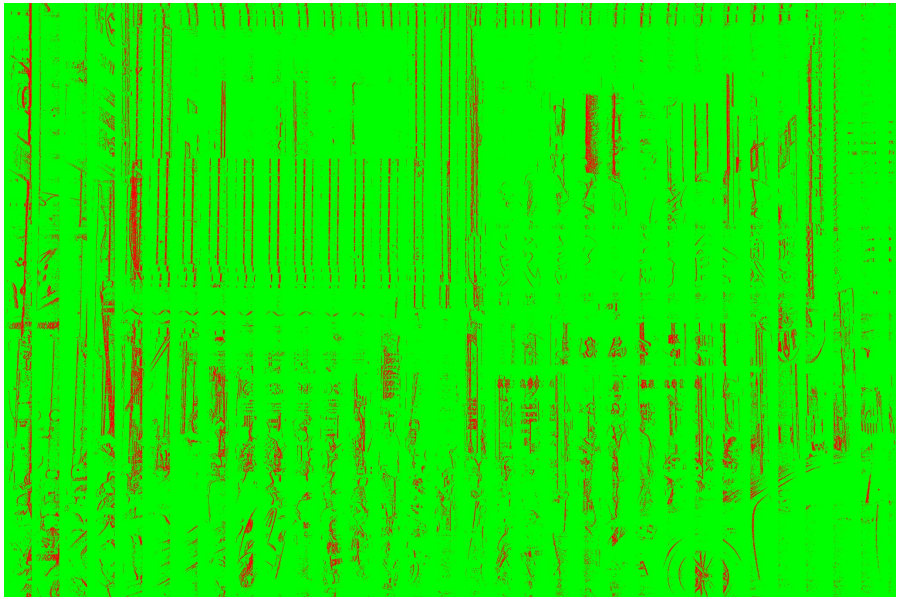
We divided the image in **131 tiles** of **513x361 pixels**

- Tile Proof generation:
  17.25 sec and 4.2 GB of RAM.
- Image Proof generation:
  2260 sec (~**38 min**) and **4.2 GB** of RAM.
- Verification time:
  65 sec (**0.5 sec per Tile**) and **<150MB** of RAM
- Proof size:
  800 bytes per tile (104.8 KB in total)

Setup operations must be performed **only once** for each fixed **dimension** and required ~90 min
Fraud Proof requires a maximum of 0.5 sec and has a maximum size of 800 bytes.

Experiments

ON THE **QUALITY** OF **LOCAL** RESIZING



A filter that **highlights pixels** with a **variance** of at least **5** in any of the **RGB** channels.

7% of pixels in total

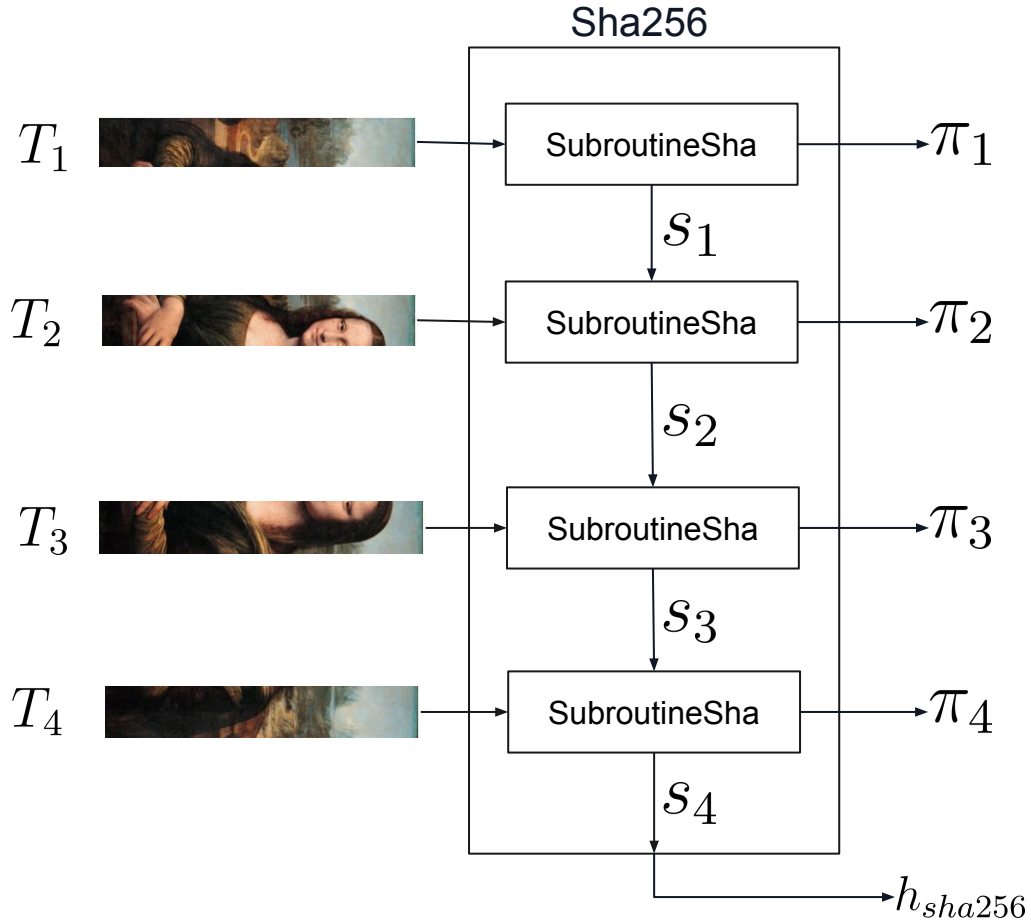
Experiments

ON THE **QUALITY** OF **LOCAL** RESIZING

Resize on the **Full Image**

Resize and merge on the **Tiles**

Compliance with C2PA



$$\Big\{(\text{Resize}(\cdot), \hat{T}_2, c_2, z_2, z_1), (T_2, s_2, s_1, r_1, r_2, r_3) :$$
$$s_2 = \text{subroutineSha256}(T_2, s_1) \wedge$$
$$c_2 = \text{Comm}(T_2, r_3) \wedge$$
$$z_1 = \text{Comm}(s_1, r_1) \wedge$$
$$z_2 = \text{Comm}(s_2, r_2) \wedge$$
$$\hat{T}_2 = \text{Resize}(T_2)\Big\}$$

For an HD image using only 4 GB, the **proof generation time** is **3088 sec** (51 min), with a proof **size** of **280 KB**.

The **verification time** is 178.5 sec, the **fraud proof** verification time is **0.5 sec,** with a fraud proof **size** of **800 B**.

More Experiments

	Tile Dimension	Setup		Prove		Verify	
	Pixels	Memory (GB)	Time (sec)	Memory (GB)	Time (sec)	Memory (GB)	Time (sec)
Crop	184756	14.1	5319	3.4	20.8	0.15	0.6
Resize	184756	14.1	5232	3.4	18.9	0.15	0.6
Grayscale	80000	14.7	5544	4.5	25.7	0.15	0.6

Table 1: Performance of a ZK-snark using TilesProof-MT (see Section 4.1.4).

	Tile Dimension	Setup		Prove		Verify	
	Pixels	Memory (GB)	Time (sec)	Memory (GB)	Time (sec)	Memory (GB)	Time (sec)
Crop	2666	14.7	3129	4.2	18.5	0.15	0.6
Resize	2666	14.7	3107	4.2	18.1	0.15	0.6
Grayscale	2666	14.7	3244	4.3	18.7	0.15	0.6

Table 2: Performance of a ZK-snark using TilesProof-C2PA (see Section 4.2).

Extended COMPARISON with *[KHSS 2022]*

	Prov	Ver (FPVer)	Proof Size (FP Size)	Resources	
ZK-IMG (Resize) [KHSS22]	328.2s	6.9 ms (N.A.)	3.04 KB (N.A)	70.7 GB Intel Xeon 8375C, 64 vCPU	☹
This paper (Resize) [TilesProof-MT]	94.5 s	3 s (0.6 s)	4 KB (800 bytes)	3.4 GB, Intel Corei7-8565U, 16 vCPU	☺
This paper (Resize) [TilesProof-C2PA]	6262 s	207.6 s (0.6 s)	276.8 KB (800 bytes)	4.2 GB, Intel Core i7-8565U, 16 vCPU	☺
ZK-IMG (Crop) [KHSS22]	328.2s	5.3 ms (N.A.)	3.04 KB (N.A)	70.7 GB, Intel Xeon 8375C, 64 vCPU	☹
This paper (Crop) [TilesProof-MT]	104 s	3 s (0.6 s)	4 KB (800 bytes)	3.4 GB, Intel Core i7-8565U, 16 vCPU	☺
This paper (Crop) [TilesProof-C2PA]	6401 s	207.6 s (0.6 s)	276.8 KB (800 bytes)	4.2 GB, Intel Core i7-8565U, 16 vCPU	☺

Table 3: Performance comparison between our work and [KHSS22] from HD to SD. We use 5 tiles of size 184756 pixels for TilesProof-MT tests. We use 346 tiles of size 2666 pixels for TilesProof-C2PA tests. Prov = time for the prover, Ver = time for the verifier, FPVer = time for the fraud proof verifier.

Leveraging the work of [GMN FC22]

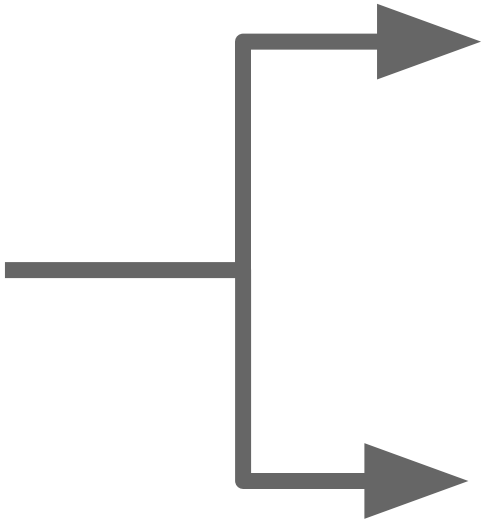
[GMN FC22] *N. Gailly, M. Maller and A. Nitulescu, "**SnarkPack: Practical SNARK Aggregation**" - Financial Cryptography - 2022*

According to the benchmarking conducted in **[GMN FC22]**, through an **aggregation** technique called SNARKPACK it is possible to **verify** 8192 Groth16 proofs in ~ 33 ms, while 16384 Groth16 proofs can be verified in ~ 58 ms.

Since in the worst case considered in our experiments, TilesProof-C2PA needs 9003 tiles for a 30 MP image, namely 9003 Groth16 proofs, applying this technique in our context is expected to provide a significant speed up in proof size and verification time.
This speed up is however strictly dependent on the use of the Groth16 ZK-SNARK.

* **[GMN FC22]** aggregates 8192 proofs in 8.7s where tests were executed on **32 cores / 64 threads** machine with AMD Raizen Threadripper CPUs exploiting GeForce RTX 3090 GPU. Technique highly based on **parallelization**.
Aggregate 8192 proofs on my i7@1.8 GHz, **8 cores** takes around 4h. Note that there is **no requirement for privacy** in this step.

Concurrent work



[DCB S&P25]
With respect to the ***[DB RWC23]***, they add a signature scheme for signers that do not have hardware constraints that is very snark-friendly, **cutting off C2PA compatible cameras**.

[DEH PETS25]
Recursive proof system on tiles to achieve succinctness.

Similar experimental results but **constraints on type of transformation,** even **simple ones** (e.g., **rectangular crop**).

[DCB S&P25] *T. Datta, B. Chen and D. Boneh, "**VerITAS: verifying image transformations at scale**" - S&P - 2025*
[DEH PETS25] *S Dziembowski, S Ebrahimi and P Hassanizadeh "**VIMz: Verifiable image manipulation using folding-based zkSNARKs**" - PETS - 2025*

Are there other application contexts?

Application Context

NFTs for Confidential Assets

Limitation of NFTs

There has been **criticism** on what being **owner** of a **digital artwork** means; this is due to the fact that **everyone** can **download digital data**, therefore enjoying it, and everyone can **create copies becoming** their **author**

Are such **problems** inherent?

Are **NFTs** for artworks really **useful**?

Application Context

NFTs for Confidential Assets

Limitation of NFTs

There has been **criticism** on what being **owner** of a **digital artwork** means; this is due to the fact that **everyone** can **download digital data**, therefore enjoying it, and everyone can **create copies becoming** their **author**

Are such **problems** inherent? Are **NFTs** for artworks really **useful**?

Naive Solution

Encrypt the **artwork** only to the **owner**

This can **limit** the **transfer** (and thus the value) of the owned **asset**.

ZK-SNARKs

NFTs for Confidential Assets

claim

EImg

edited image

operation

C_OImg

Hash_OImg

Comm_Key

secret witness

Key

encryption key

OImg

original image

Prover (seller)

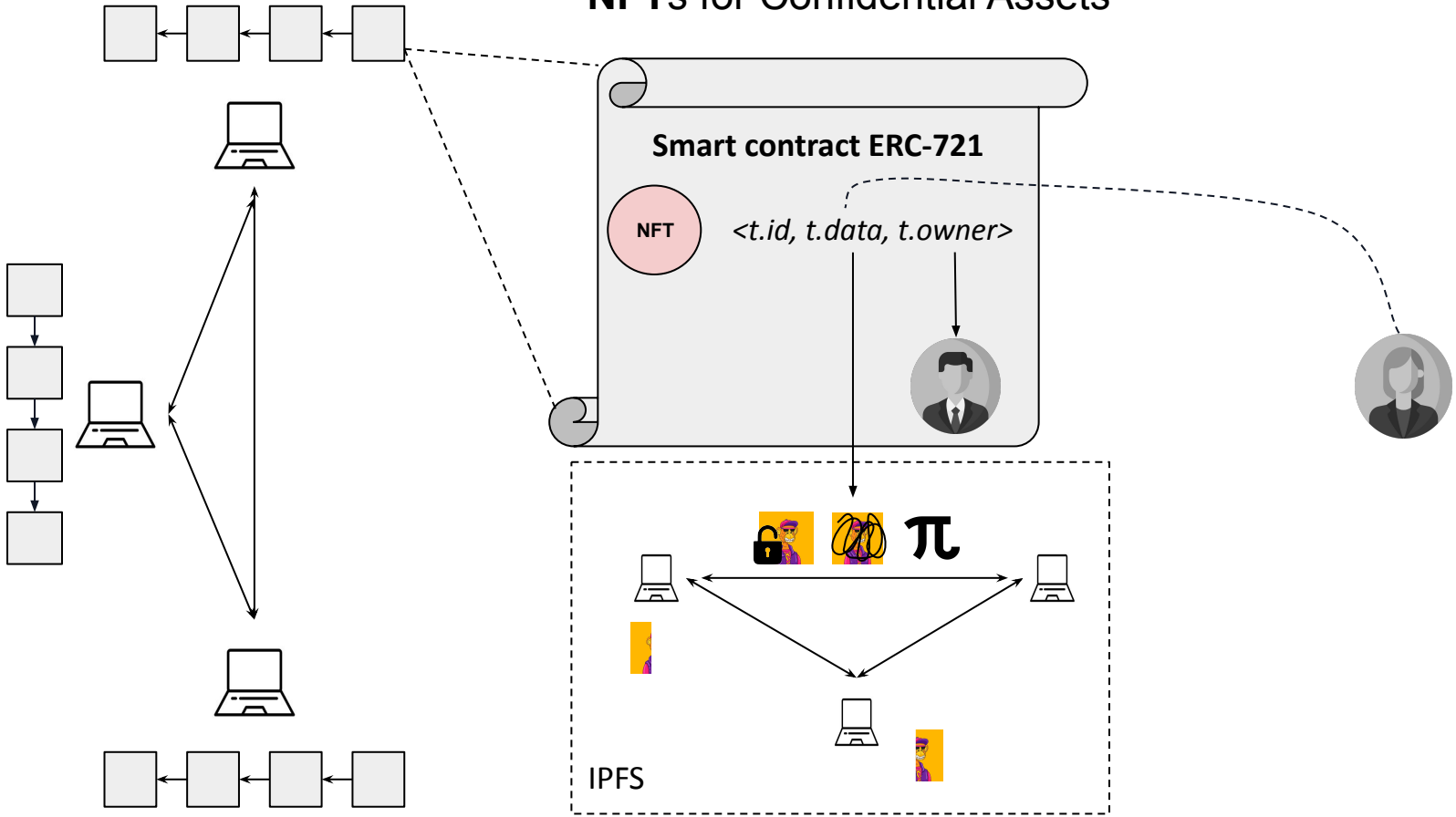
π **proof**

- Hash_OImg is the hash of OImg
- EImg = **operation**(OImg)
- C_OImg is the Enc of the OImg with Key
- Comm_key is the comm. of the key used in the Enc

Verifier (buyer)

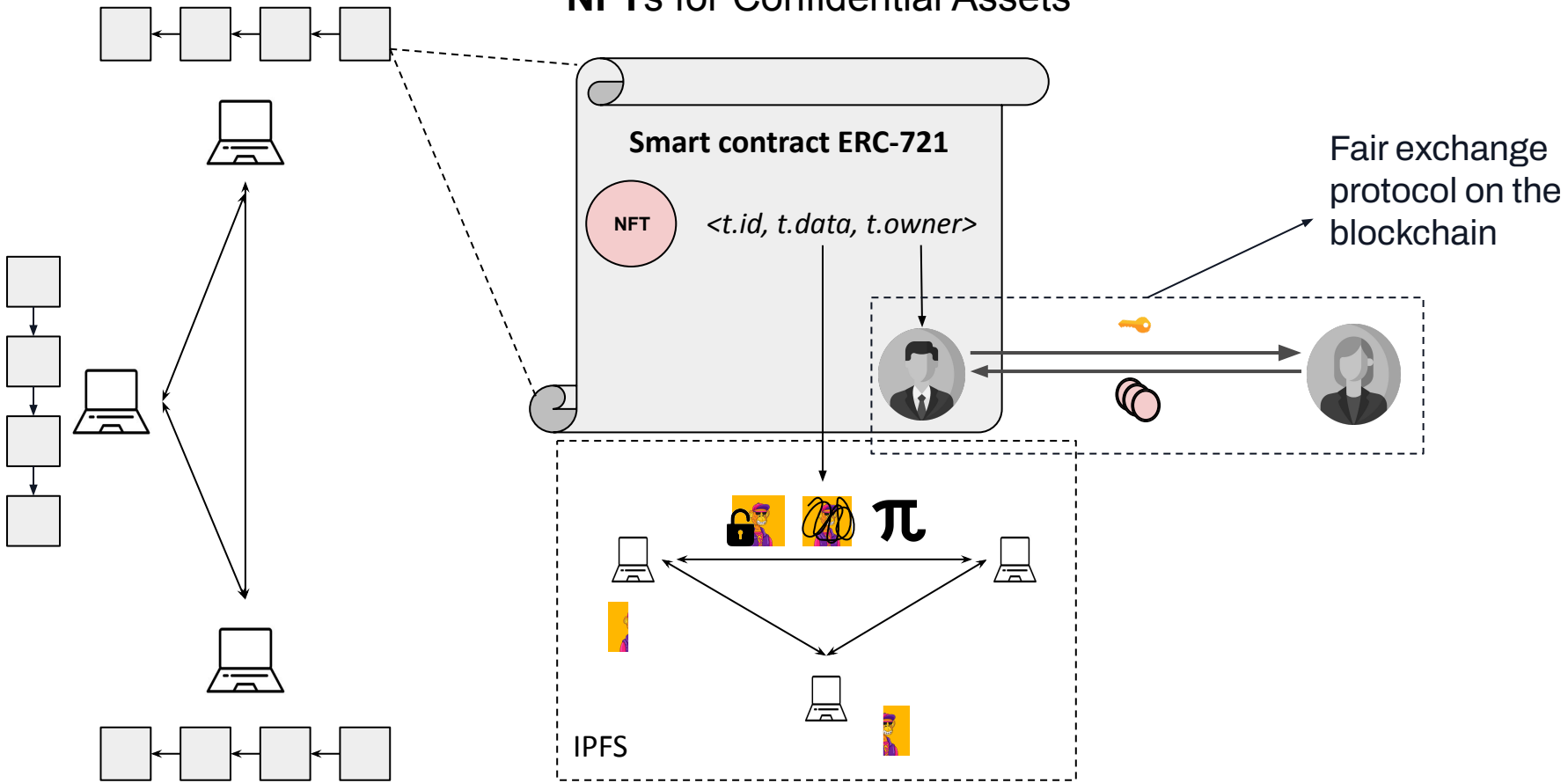
Application Context

NFTs for Confidential Assets



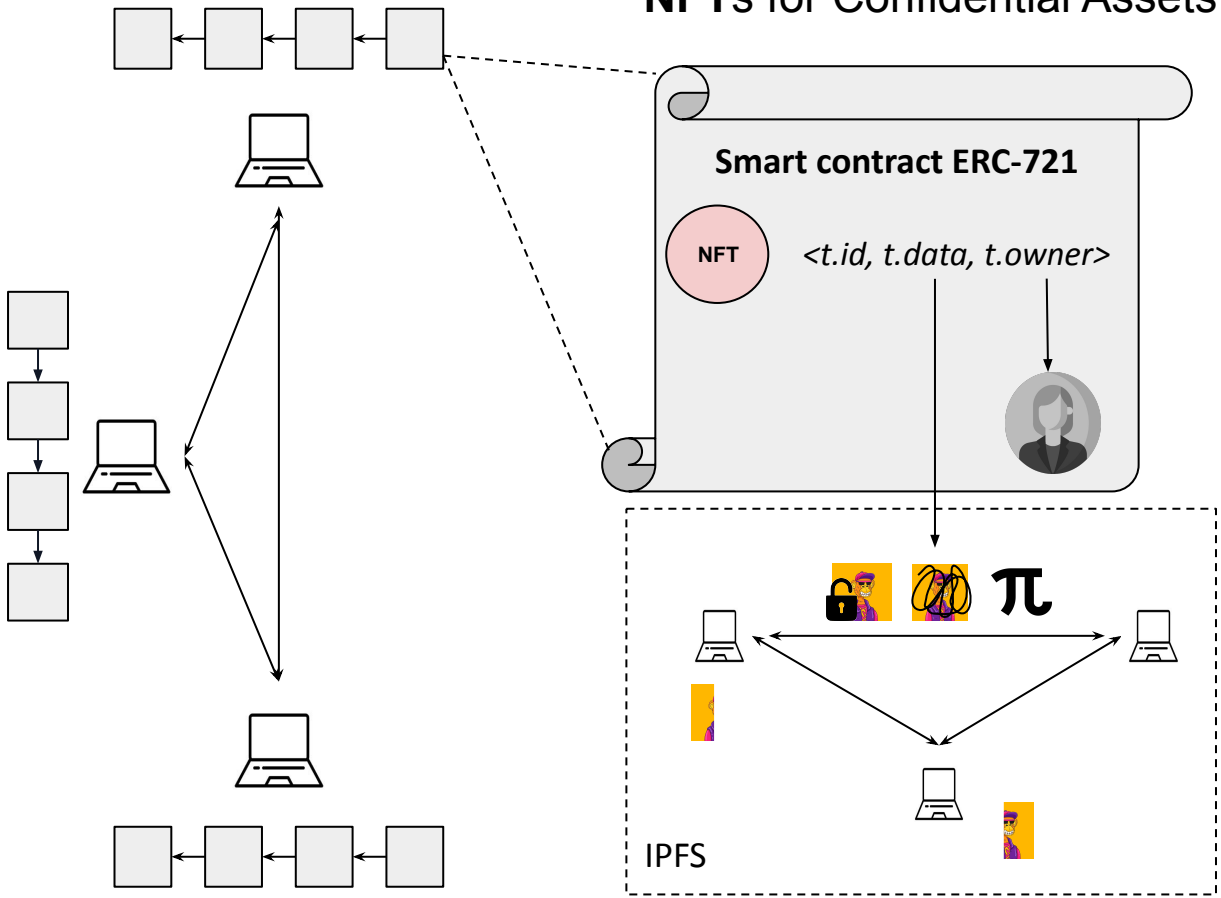
Application Context

NFTs for Confidential Assets



Application Context

NFTs for Confidential Assets



THANKS!

This presentation includes icons from Flaticon