Using ZK Proofs to Fight Disinformation

By Trisha Datta and Dan Boneh

Dan Boneh

Follow

--

6

Listen

Share

Verifying where and when a digital image was taken has become increasingly difficult. Image provenance is especially concerning in the realm of news media. After Russia invaded Ukraine in February, several photographs and videos circulated online that falsely claimed to show the conflict. One illustrative example is this alleged side-by-side comparison of two photos; the first purportedly shows Syria after Russian air bombings, and the second supposedly shows Ukraine after the Russian invasion. Later analysis by USA Today showed that the first photo was actually taken in Iraq in 2017 and that the second photo, though indeed of Ukraine, was taken two weeks before the invasion. While these fact-checking services are important, enabling individuals to personally verify the provenance of photos would prevent them from having to rely on third-parties, and empower them to protect themselves against this kind of misinformation.

In an ideal world, photos would come with geographic locations and timestamps as well as proofs that this information is correct. Then, if a user sees a photo in an online news article, they could use the provided proof to verify when and where the photo was taken without having to trust the article's publisher or a third-party fact-checking website. This verification could be performed through something like a browser extension, which could automatically detect and verify these proofs. The rest of this blog post discusses our attempts to enable such a system.

The Coalition for Content Provenance and Authenticity (C2PA) proposed a standard to verify image provenance that relies on digital signatures. Cameras would "digitally sign" each photo taken along with a series of assertions about the photo (e.g., location, timestamp). However, photos are not published as is; they are often cropped, resized, and possibly converted to grayscale, before being posted in a news story. Once these edits are made, the public can no longer validate the signature on the original image, since only the edited image is published. In the C2PA standard, when a signed photo is edited by a C2PA-enabled editing application, the editing application will sign the edits that have taken place. The public can verify these signatures to validate the location and timestamp of the published photo.

A problem with this C2PA approach is that it relies on trusting the editing software to sign that the edits were made correctly. If an adversary could extract the signing key from the editing software, it could generate a valid signature for any image it wants, while still following the C2PA protocol. This will convince an honest verifier that the forged image is a valid photo, even though it is not. Similarly, a software vulnerability in the editing software may let an adversary sign any image of it choice, even without extracting the software's signing key.

We therefore need a method for editing a signed photo, so that a viewer who only has the edited photo can be assured that (i) the original unedited photo was properly signed, and was taken at the claimed time and location, and (ii) only permissible edits, such as cropping, resizing, and grayscale, were made to the signed photo. The security of the scheme should not require trusting the editing software.

To prove the provenance of edited photos, we use succinct zero-knowledge proofs. A zero-knowledge proof is a statement about a secret witness that can be verified by a user without the user learning anything about the witness, other than that the statement is true. These proofs are complete, meaning that verification will succeed on true statements with high probability, sound, meaning that verification will fail on false statements with high probability, and zero knowledge, meaning that nothing is revealed about the original image. These properties mean that the verifier need not trust the prover, which solves the trust problem posed by the C2PA protocol.

In the online news system described above, we would use these zero-knowledge proofs as follows. Every photo displayed in a news article would be accompanied by its metadata (e.g., location and timestamp), a description of the edits that were made to the original photo, and a succinct zero knowledge proof. The proof proves the following statement:

“The prover (i) knows an unedited photo that is properly signed by a C2PA camera, (ii) the metadata on the unedited signed photo is the same as the one attached to the public photo, and (iii) the public photo in the news article is the result of applying the claimed edits to the unedited photo.”

The viewer will only accept a photo if it is accompanied by a valid zero-knowledge proof, and the list of edits made to the original photo are “permissible.” By permissible we mean edits that would not fundamentally alter the content of the photo (this list from the Associated Press contains some examples).

We stress that the unedited photo and its signature are part of the secret witness, and not available to the public. The viewer only sees the edited photo, its metadata, and the zero knowledge proof. The “zero-knowledge” property ensures that the original photo is kept secret; this is desirable in cases where the original photo contained sensitive content that needs to be cropped out.

In 2016, Naveh and Tromer implemented zero-knowledge proofs for various photo edits, including cropping, transposing, flipping, rotating, and adjusting contrast/brightness. While this work demonstrated the feasibility of writing zero-knowledge proofs for image editing, the proving time of the implementation was too large to be practical.

Fortunately, advancements made in the past six years now allow us to generate zero-knowledge proofs for image editing in feasible amounts of time. To demonstrate this, we implemented programs to create zero-knowledge proofs for cropping, resizing, and grayscale conversion, which are all included in the list of allowable edits published by the Associated Press. Our implementations are written in circom, a language designed for generating zero-knowledge proofs. Each circom program generates a zero-knowledge proof that verifies that a set of constraints are satisfied for a set of (potentially secret) inputs. In our implementations, the inputs are the original photo, the new photo (e.g., the photo displayed in an online news article), and parameters associated with the edit, and the constraints assert that the new photo is obtained by performing the edit with the given parameters on the original photo. We discuss each program and timing results for proof generation below.

Our cropping proof-generation program creates constraints that assert that the new photo has the same RGB values as the original photo in the cropped range. Table 1 shows the timing results for this program.

Our resizing proof-generation program creates constraints that assert that the new photo can be obtained from the original photo through bilinear resizing, which calculates the RGB for every pixel in the resized image by taking a weighted linear combination of the RGB values of four pixels in the original image. Our circom program implements the resizing logic outlined in this blog post, which is consistent with tensorflow’s image resizing operation. Table 2 shows the timing results for this program.

Our grayscale proof-generation program creates constraints that assert that the RGB values of the new photo can be obtained by applying the standard grayscale calculation used by Photoshop to the RGB values of the original photo. This formula obtains the value for a pixel by taking a weighted linear combination of the RGB values: grayValue = 0.30R + 0.59G + 0.11B. To accommodate the floating-point numbers involved in this calculation, we scaled our values by a factor of 100 and passed in remainders for the rounding calculations as part of the secret witness. Table 3 shows the timing results for this program.

We ran the experiments on a Google Cloud e2-standard-8 VM instance (eight cores, 32 GB memory). The numbers reported are for “realistic” image sizes. In particular, the signature-producing Sony camera mentioned earlier is a 33 megapixel camera, and, depending on the client, photos on The New York Times are resized to 2048 x 1363 pixels.

Because all the proofs described here only need to be generated once by the publisher, the reported times are suitable for real-world use. Verification time is independent of image size and takes about a hundred milliseconds.