[Proposal] Implementation and Analysis of Apple's CSAM Detection System

Alessandro Baccarini
anbaccar@buffalo.edu
University at Buffalo
October 14, 2021

Earlier this August, Apple unveiled its novel Child Sexual Abuse Material (CSAM) detection system[1] to inhibit the spread of CSAM and aid law enforcement in pursuing criminals. The framework is designed to automatically detect known CSAM images stored in iCloud Photos, and allow Apple to report the offending users to the National Center for Missing and Exploited Children (NCMEC). The system would be present on all US-based iPhone and iPad devices running iOS 15 and iPadOS 15, respectively, with the goal of adding support to macOS and watchOS in the future. The announcement received significant backlash from media and the tech community, stating that this system is effectively a backdoor an authoritarian government can leverage to conduct censorship on any material deemed "inappropriate." However, on September 8, 2021, Apple ultimately decided to delay the rollout of this feature citing feedback from "customers, advocacy groups, and researchers." No projected release date has been provided at the time of writing.

The goal of this project is implement and verify Apple's protocol using open-source software (such as [1, 8, 3] for the machine learning component) and standard cryptographic libraries (such as Python's `cryptography` package) on a small-scale dataset. The Private Set Intersection (PSI) system proposed in [2] is constructed as follows: First, the database CSAM Hashes is stored on the client's device. PSI is performed locally, which is used to determine if there exists a match without revealing the result. The device generates "safety voucher" that encodes the match result and user's decryption key, which is subsequently uploaded to iCloud for PSI matching. Threshold secret sharing is leveraged to ensure the user's decryption key cannot be recovered without meeting some predefined "threshold." If this threshold is crossed, Apple can retrieve the decryption key for the flagged user's photos, manually verify that they in fact CSAM images, and report the user to NCMEC. A basic overview of the system is shown in Figure 1, and a technical summary is given in [5]. The following cryptographic tools are used in this system:

NeuralHash: NeuralHash is a perceptual hashing function for images based on neural networks and produces a "fingerprint" of the input. The algorithm passes the input image into a convolutional neural network to produce an N-dimensional floating point descriptor, which are hashed using Hyperplane LSH (Locality Sensitivity Hashing) to produce an M-bit value as the NeuralHash of the image. Unlike a standard cryptographic hash function (such as MD5 and the SHA family), NeuralHash is insensitive to small perturbations of the input image, such as cropping and pixel inversion. Note, the publicly available reverse-engineered version of the NeuralHash algorithm has been shown to not be collision resistant [8]. Apple claims a proprietary server-side algorithm is run to verify the results [4].

Private Set Intersection (PSI): Private Set Intersection (PSI) is a multi-party protocol that allows each party to compute the intersection of their inputs in a privacy-preserving manner, such that

[1] https://www.apple.com/child-safety/

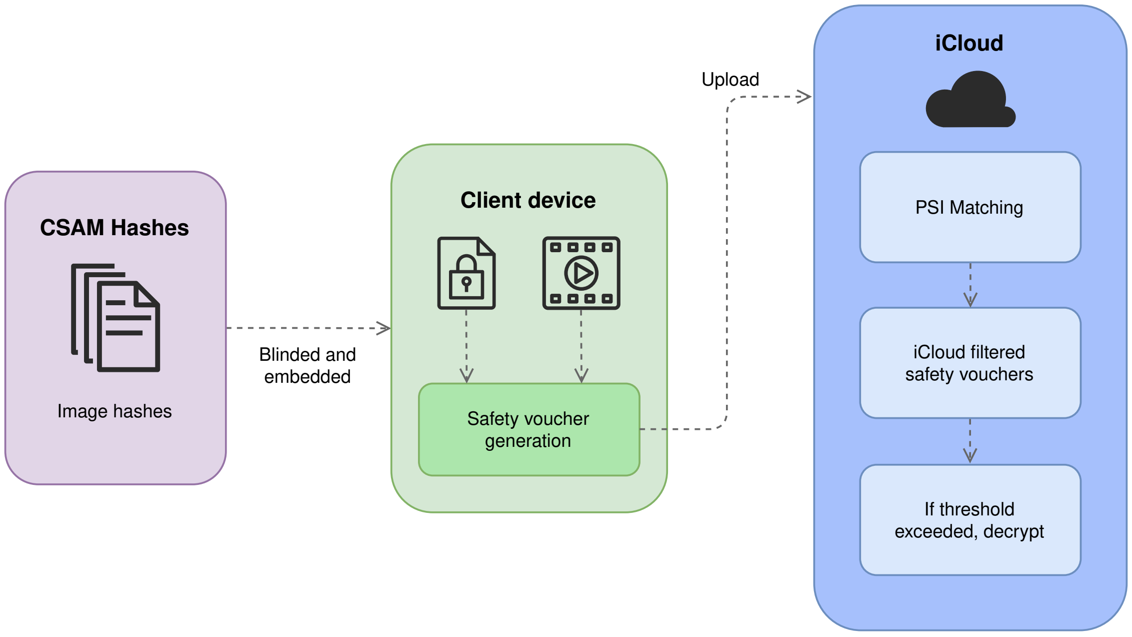
CSAM Hashes
Image hashes
Blinded and embedded
Client device
Safety voucher generation
Upload
iCloud
PSI Matching
iCloud filtered safety vouchers
If threshold exceeded, decrypt

Figure 1: Basic protocol functionality.

only shared inputs are revealed [7]. The security model for PSI requires privacy for the server, privacy for the client, the absence of any false positives, and the protocol does not require a correct output for a malicious client. PSI is based on a variety of cryptographic primitives, including symmetric encryption, Elliptic curve with Decision Diffie-Hellman, hash functions, HMAC for secure key derivation, Shamir Secret Sharing, and pseudorandom functions.

Threshold Secret Sharing: Threshold Secret Sharing is a cryptographic technique where some secret x and is distributed as shares x_i to a set of n participating parties $P_i, \ldots P_n$. Any t users can reconstruct the secret, but any amount $t-1$ or fewer cannot recover any information about x. Specifically, this is referred to as (n,t)-*threshold scheme* (with $t \leq n$) [6]. In the CSAM detection system, the user's decryption key for their photos is the secret, and is used to generate a share (based on Shamir Secret Sharing) once a potential match is found. A secondary encryption based on PSI is performed on the secret share and encrypted image. The scheme ensures information about images stored in iCloud remains private if the threshold is not met. If a sufficient number of matches are found, the PSI protocol guarantees only the matched images can be recovered. It is necessary to obfuscate the number of matched images until the threshold is exceeded, otherwise Apple could learn this information without properly verifying the images' contents. Therefore, "synthetic vouchers" are periodically sent to the server with garbage data replacing the shares and underlying image information, along with a header that always produces a match on the server.

Apple claims the system ensures the following security guarantees:

- Apple cannot recover the user's photos without meeting the threshold number of shares of the key.
- The likelihood of a false positive is statistically insignificant (1 in 1 trillion claimed by Apple).
- No information is learned about non-matched images.

- The user cannot retrieve any information from the CSAM database.
- The user cannot identify which images were flagged as CSAM by the system.

We plan to implement the system as a terminal-based program in Python, where we have two consoles functioning as the client (C) and server (S) as follows:

- Client C:
  1. C has access to a database of NeuralHashes functioning as the known CSAM hash list.
  2. The user inputs a photo's filename into the console to emulate the process of taking a photo on their phone.
  3. In the background, C's device computes the NeuralHash of the input image and matches it against the database.
  4. If a match is found, C computes a cryptographic header derived from the NeuralHash and encrypts a payload consisting of the image's information and a share of the user's private key, thus constituting the safety voucher.
  5. C sends the safety voucher to S.
- Server S:
  6. S receives the voucher from C.
  7. Using the cryptographic header, S attempts to decrypt the payload.
  8. If steps 6 and 7 are successful a sufficient number of times to surpasses the threshold, S can recover the user's encryption key and retrieve the matched images' information.

All of the necessary cryptographic tools (symmetric encryption, Elliptic curve with Decision Diffie-Hellman, hash functions, HMAC for secure key derivation, Shamir Secret Sharing, and pseudorandom functions) needed for the system are publicly available as part of the `cryptography`[2] package and other related open-source software. Basic networking is achievable through the `socket`[3] package. Sample images will be taken from the ImageNet[4] dataset.

References

[1] Anish Athalye. NeuralHash Collider. https://github.com/anishathalye/neural-hash-collider, September 2021. original-date: 2021-08-19T00:06:20Z.

[2] Abhishek Bhowmick, Dan Boneh, Steve Myers, and Kunal Talwar Karl Tarbe. The Apple PSI System. https://www.apple.com/child-safety/pdf/Apple_PSI_System_Security_Protocol_and_Analysis.pdf.

[3] Nikhil Cheerla. NeuralHash: An Adversarial Steganographic Method For Robust, Imperceptible Watermarking. https://github.com/nikcheerla/neuralhash, September 2021. original-date: 2018-02-17T03:55:00Z.

[4] Joseph Cox, Lorenzo Franceshi-Bicchierai, and Samantha Cole. Apple Defends Its Anti-Child Abuse Imagery Tech After Claims of 'Hash Collisions'. https://www.vice.com/en/article/wx5yzq/apple-defends-its-anti-child-abuse-imagery-tech-after-claims-of-hash-collisions.

[5] Apple Inc. CSAM Detection: Technical summary. https://www.apple.com/child-safety/pdf/CSAM_Detection_Technical_Summary.pdf.

[6] Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography.* Chapman and Hall/CRC, 2014.

[7] Ágnes Kiss, Jian Liu, Thomas Schneider, N Asokan, and Benny Pinkas. Private set intersection for unequal set sizes with mobile applications. *Proc. Priv. Enhancing Technol.*, 2017(4):177–197, 2017.

[2] https://github.com/pyca/cryptography
[3] https://docs.python.org/3/library/socket.html
[4] https://www.image-net.org/

[8] Asuhariet Ygvar. AppleNeuralHash2ONNX. https://github.com/AsuharietYgvar/AppleNeuralHash2ONNX, September 2021. original-date: 2021-08-15T19:52:47Z.