# [Progress Report] Implementation and Analysis of Apple's CSAM Detection System

Alessandro Baccarini
anbaccar@buffalo.edu
University at Buffalo
November 5, 2021

Since the initial proposal, we have progressed towards completion of this project. We outline the progress we hve made and address the difficulties encountered thus far in this document.

The first We have decided to implement the threshold-based PSI protocol (tPSI-AD), since this is substantially easier to understand and relies solely on classical cryptographic constructions, whereas the fuzzy protocol (ftPSI-AD) relies on the novel idea of Detectable Hash Functions, an area which is not well explored.

Furthermore, we are replacing some of the more obscure cryptographic tools (such as Cuckoo hashing and hashing to elliptic curves) with standard building blocks, which provide similar if not the same security guarantees while sacrificing performance advantages. Specifically, we aim to replace Cuckoo hashing with hash tables, replace the elliptic curve $E(\mathbb{F}_p)$ with a Diffie-Hellman group of prime order $q$ (where a point on the curve is mapped to a key) and the discrete log problem is hard, and the Hash to Elliptic Curves construction with a simple hash function such as SHA-3.

Regarding additional documentation, Pinkas's presentation [5] outlines fundamental PSI constructions that are useful for understanding the tPSI-AD protocol, including:

- A naive approach – $A$ and $B$ agree on a hash function $H$, $B$ sends $H(y_1), \ldots, H(y_n)$ to $A$, $A$ compares it to $H(x_1), \ldots, H(x_n)$ and finds the intersection.
- A PSI protocol based on Diffie-Hellman – similar as naive approach, except $B$ sends $(H(y_1))^\beta$, $\ldots, (H(y_n))^\beta$ to $A$ and $A$ simultaneously sends $(H(x_1))^\alpha, \ldots, (H(x_n))^\alpha$ to $B$. Then $B$ sends $((H(x_1))^\alpha)^\beta, \ldots, ((H(x_n))^\alpha)^\beta$ to $A$, and $A$ compares the results.
- A PSI protocol based on Blind RSA – $B$ chooses an RSA key pair $((N, e), d)$, and $A$ chooses random $r_1, \ldots, r_n$, computes $x_1 (r_1)^e, \ldots, x_n (r_n)^e$ and sends to $B$. $B$ computes and sends $(H(y_1))^d, \ldots, (H(y_n))^d$ and $(x_1 (r_1)^e)^d, \ldots, (x_n (r_n)^e)^d$, which equals $(x_1)^d r_1, \ldots (x_n)^d r_n$. $A$ divides each input by $r_i$, applies $H$ and compares the lists.

These simple constructions may prove useful for detailed analysis of tPSI-AD later on. Zhang et. al [7] designed an efficient threshold PSI protocol based on garbled Bloom filters and threshold secret sharing. Their paper analyzes existing solutions and the fundamental aspects of PSI. Other useful online resources include [6, 4].

The accompanying document for the project has been updated to summarize the work of [2] and describe the replacements we make for the sake of simplicity. Regarding the implementation, we have tested and confirmed the functionality of the NeuralHash tool from [3], as well as identified the relevant documentation and examples within [1].

The project is approximately 50% completed, with the bulk of the progress coming from determining what actually needs to be implemented/what can be replaced with a simpler version,

and reading background materials on PSI. The following milestones remaining: finalize the modified protocol description using standard cryptographic constructions, conduct through analysis to verify the security guarantees are met from the original tPSI-ad protocol, and finish the actual implementation.

## References


[1] Python Cryptographic Authority. pyca/cryptography documentation. https://cryptography.io/en/latest/, 2021.

[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] Sarah Jamie Lewis. A closer look at fuzzy threshold PSI (ftPSI-AD). https://pseudorandom.resistant.tech/a_closer_look_at_fuzzy_threshold_psi.html, 2020.

[5] Benny Pinkas. Private set intersection. https://cyber.biu.ac.il/wp-content/uploads/2017/01/15.pdf, 2015.

[6] Avishay Yanai. Private set intersection. https://decentralizedthoughts.github.io/2020-03-29-private-set-intersection-a-soft-introduction/, 2020.

[7] En Zhang, Jian Chang, and Yu Li. Efficient threshold private set intersection. *IEEE Access*, 9:6560–6570, 2021.