Actively Secure Private Set Intersection in the Client-Server Setting

Yunqing Sun, Jonathan Katz, Mariana Raykova, Phillipp Schoppmann, and Xiao Wang WPEC 2024: NIST Workshop on Privacy-Enhancing Cryptography 2024 09/24/2024









The password check PSI properties:

- 1) Multi-client and multi-execution
- 2) Unbalanced and large-scale dataset
- 3) Fully malicious secure
- 4) Reusable server encoding

Protocol Features

Most prior active PSI:



Protocol Features

Ours:



Challenge: malicious security

Prior Work: OPRF-based Semi-honest PSI



Concern: *EY* may not be computed correctly

• *k*'s consistency in different executions

(for example: different subset of EY use different k to manipulate the intersection)

Our Solution: VOPRF-based Actively Secure PSI



EY is computed correctly!

- Any client could verify the consistency of *sk* in different executions
- The *sk* in *EY* has to match *pk*. Otherwise, it won't match any client's encoding.

Verifiable Functions and its Oblivious Evaluation

- VOPRF and its variant
 - Input not extractable [ECS+15, TCR+22]
 - Scheme not practical [BKW20, Bas24]
- VRF (Verifiable Random Function F') is overkill and VUF (Verifiable Unpredictable Function F) is sufficient ($F'_{sk}(x) = H(F_{sk}(x))$)
- A new **notion OVUF** (Oblivious Verifiable Unpredictable Function)

➢More efficient

[ECS+15]Adam Everspaugh, Rahul Chatterjee, Samuel Scott, Ari Juels, and Thomas Risten part. The pythia PRF service. In USENIX Security 2015, August 12–14, 2015. [TCR+22] Nirvan Tyagi, Sof Ia Celi, Thomas Ristenpart, Nick Sullivan, Stefano Tessaro, and Christopher A. Wood. A fast and simple partially oblivious PRF, with applications. In EUROCRYPT 2022, Part II, volume 13276 of LNCS [BKW20] Dan Boneh, Dmitry Kogan, and Katharine Woo. Oblivious pseudorandom functions from isogenies. In ASIACRYPT 2020, Part II, volume 12492 of LNCS, December 7–11, 2020. [Bas24] Andrea Basso. A post-quantum round-optimal oblivious PRF from isogenies. In SAC 2023, LNCS, August 2024.

OVUF Functionality (of a concrete VUF)



Locally check output validity

Check: (*sk*, *pk*) validity

OVUF Construction (Based on VUF by [DY05])

•
$$GEN(1^{\kappa})$$
: $sk \leftarrow Z_q, pk = g^{sk}$

•
$$F_{sk}(x) = g^{\frac{1}{x+sk}} (F_{sk}(-sk) = 1)$$

• $Vrfy(pk, x, y) \rightarrow 1$ iff $e(g^x \cdot pk, y) = e(g, g)$ (Or $pk = g^{-x}$ and y = 1)



[DY05] Yevgeniy Dodis and Aleksandr Yampolskiy. A verifiable random function with short proofs and keys. In PKC 2005, volume 3386 of LNCS, January 23–26, 2005.

Compute Secret Share of (sk + x)(a + b)[DG18]

Assume sk and a are held by party A, x and b are held by party B

$$(sk + x)(a + b) = sk \cdot a + sk \cdot b + x \cdot a + x \cdot b$$

Multiplicative to Additive share conversion (MtA)



Party A's secret share: $sk \cdot a + c_1 + c_2$

```
Party B's secret share: x \cdot b + d_1 + d_2
```

[DG18] Rosario Gennaro and Steven Goldfeder. Fast Multiparty Threshold ECDSA with Fast Trustless Setup. In CCS 2018, October 15-19, 2018.

OVUF Construction



OVUF Construction



OVUF Construction





Be caught with overwhelming probability

History of MtA

➢ Paillier encryption

➤Castagnos-Laguillaumie encryption

➢OT-based

- Semi-honest [Gil99]:
 - generate additive secret share of $a \cdot b_i$; $a \cdot b$ is a linear combination of $a \cdot b_i$.
- Malicious-secure [DKLs19]:
 - To prevent selective-failure attack: Encode input b to $Encode(b, g^R) \in \{0, 1\}^{n+l}$
 - Correctness: each OT selects two sets of values, where the second set used for

checking correctness of the output

[Gil99] Niv Gilboa. Two party RSA key generation. In CRYPTO'99, volume 1666 of LNCS, August 15–19, 1999.

[DKLs19] Jack Doerner, Yashvanth Kondi, Eysa Lee, and abhi shelat. Threshold ECDSA from ECDSA assumptions: The multiparty case. In 2019 IEEE Symposium on Security and Privacy. IEEE Computer Society Press, May 19–23, 2019.

Weak MtA: Our optimization for OVUF

- Keep the encoding $Encode(b, g^R) \in \{0, 1\}^{n+l}$
- Remove the correctness checking
- Result in an imperfectness in MtA correlation
- We prove that the imperfectness *e* is uniformly distributed





Verify the result via pk and catch the errors

Performance

- 10^8 server elements, 10^3 client elements
- 60MB of communication, run in under 5s in WAN network (120 Mbps), and 0.5s in LAN



Performance

• Compare with SOTA PSI [RR22]



Cost 0.017 USD per client 5x saving in network-caching infrastructure

[RR22] Srinivasan Raghuraman and Peter Rindal. Blazing fast PSI from improved OKVS and subfield VOLE. In ACM CCS 2022, November 7–11, 2022.

Other ways to distribute server encoding

So far, we use network caching (CDN) ! There are still other ways:

- Verifiable private information retrieval
- > Other solutions (eg. TOR network, bucketization)

Thanks

https://eprint.iacr.org/2024/570.pdf Will appear in CCS'24

Code will open source soon at <u>https://github.com/YunqingSun0-0/ovuf-psi</u>