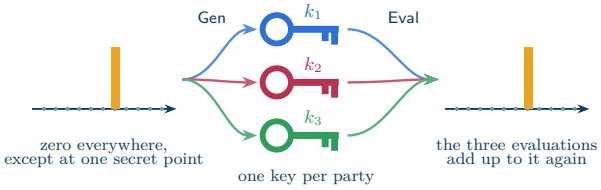
Implementing Multi-Party DPFs for SPDZ Triple Generation



a three-party DPF — and, beyond two parties, those three keys are the expensive part

Multi-party computation (MPC) protocols of the SPDZ family achieve active security against a dishonest majority by splitting the work in two: an input-independent preprocessing phase produces authenticated multiplication triples; a fast online phase then consumes these triples for evaluation. Generating and storing the huge number of triples needed in practice is the main bottleneck. Pseudorandom correlation generators (PCGs) [1] address it: the parties receive short correlated seeds and expand them locally into millions of triples.

For plain multiplication triples, the multi-party case reduces to pairwise two-party correlations. The authenticated version that SPDZ needs, however, contains degree-three terms, which the PCG of Abram and Scholl [2] handles with a three-party *distributed point function* (DPF) — the primitive drawn above, which splits a function that is zero everywhere except at one secret point into one key per party, each key alone hiding that point. Its keys grow with the square root of the domain size, and this single component caps the compression of the whole generator. New multi-party DPFs have appeared since, including an algebraic construction with keys polylogarithmic in the domain size [3]. A smaller key size is only half of the story, though: a PCG evaluates its DPFs on the entire domain, and the algebraic construction replaces cheap AES calls by matrix products and group exponentiations. Which DPF is the best choice for triple generation, and for which parameters?

In this thesis you will (i) study DPFs and their role in [2]; (ii) implement two three-party DPFs: the square-root construction used in [2] and the algebraic construction of [3]; and (iii) benchmark key size, key generation, and full-domain evaluation to locate the crossover points. Optionally, you can plug the winner into a simplified (trusted-dealer) version of the triple generator of [2] and measure the end-to-end compression. The topic suits a student who likes fast code (C++, Rust or Python). More information can be found by contacting Feiyang Qiu (email below).

[1] E. Boyle, G. Couteau, N. Gilboa, Y. Ishai, L. Kohl, and P. Scholl, "Efficient pseudorandom correlation generators from ring-LPN," in *Advances in Cryptology – CRYPTO 2020*, Springer, 2020, pp. 387–416.

[2] D. Abram and P. Scholl, "Low-communication multiparty triple generation for SPDZ from ring-LPN," in *Public-Key Cryptography – PKC 2022*, Springer, 2022, pp. 221–251.

[3] T. Krips and P. Pullonen-Raudvere, "Multi-party distributed point functions with polylogarithmic key size from invariants of matrices," Cryptology ePrint Archive, Paper 2025/978, 2025.

Practicalities

Promotor:	Bart Preneel
Daily supervisor:	Feiyang Qiu feiyang.qiu@esat.kuleuven.be office B01.34
Nature of the work:	20% literature, 20% theoretical work, 60% software
Number of students:	1

 © 2026 – 2027 COSIC