# Off-Key: Inconsistent MAC Keys Cause Selective-Failure Attacks in SPDZ-Family MPC

Marcel Keller Sela Navot Nirvan Tyagi

Secure multi-party computation (MPC) lets several parties jointly evaluate a function of their private inputs while learning nothing beyond the result. SPDZ [3, 4] is a widely used family of MPC protocols in the malicious security setting. Its security rests on the following invariant: every piece of preprocessing used in a computation must be generated under one shared authentication key. In principle, the preprocessing protocols enforce this invariant internally.

We first show that a broken invariant enables a simple selective-failure attack that leaks an honest party's secret inputs. We then show that leading implementations fail to enforce it, letting a corrupted party present different keys across threads or runs and mount a practical attack.

**How SPDZ protects secrets.** SPDZ stores a secret $x$ as additive shares $x = x_0 + x_1$,[1] one per party. Each secret carries an authentication tag (MAC) $m = \alpha x$, shared as $m = m_0 + m_1$, under a global key $\alpha = \alpha_0 + \alpha_1$ that no single party knows. A value is *valid* when $m = \alpha x$. An offline *preprocessing* phase supplies correlated random values, authenticated under $\alpha$, for the online phase to consume; validity is enforced by a lightweight MPC protocol that checks each value against the key of the preprocessing it uses, and aborts on failure.

**The attack.** Suppose a corrupted party obtains a second batch of preprocessing under a different key $\alpha + 1$, while a secret $x$ was authenticated under $\alpha$. Since each value is checked against the key of the preprocessing it uses, feeding $x$ into an operation that consumes this batch checks its validity under $\alpha + 1$. Adding a guess $g$ to its local tag share $m_0$ then gives

$$m = (m_0 + m_1 + g) = \alpha\, x + g = (\alpha + 1)\, x + (g - x),$$

so $x$ passes as valid under $\alpha + 1$ exactly when $g = x$. Whether the run aborts thus reveals whether the guess is right: the check is now a selective-failure oracle that leaks information about the secret $x$.

**In practice.** Three influential frameworks enforce key consistency within a single preprocessing session, but not across the multiple sessions their execution paths allow:

- **SCALE-MAMBA** [2] and its fork **FANNG-MPC** [1] allow multiple "runs" to process large circuits, and each run re-reads the key from a plaintext file and re-commits to it.
- **MP-SPDZ** [5]: in multi-threaded execution, preprocessing draws fresh key material per thread/run, so the attacker can authenticate under $\alpha$ in one thread/run and $\alpha + 1$ in another.

**Fix and disclosure.** The preprocessing protocols already commit to a key within each session; implementations must persist that commitment across sessions rather than re-deriving it each time. We reported it to all three: MP-SPDZ (maintainer a co-author) patched it in v0.4.3, SCALE-MAMBA is unmaintained, and FANNG-MPC's maintainers acknowledged it and explicitly cleared us to publish.

[1] For simplicity we describe SPDZ in the two-party setting, but the attack naturally extends to more parties.

## References


[1] Najwa Aaraj, Abdelrahaman Aly, Tim Güneysu, Chiara Marcolla, Johannes Mono, Rogerio Paludo, Iván Santos-González, Mireia Scholz, Eduardo Soria-Vazquez, Victor Sucasas, and Ajith Suresh. FANNG-MPC: framework for artificial neural networks and generic MPC. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2025(1):1–36, 2025.

[2] Abdelrahaman Aly, Kelong Cong, Daniele Cozzo, Marcel Keller, Emmanuela Orsini, Dragos Rotaru, Oliver Scherer, Peter Scholl, Nigel P. Smart, Titouan Tanguy, and Tim Wood. SCALE–MAMBA v1.14: Documentation. `https://nigelsmart.github.io/SCALE/Documentation-SCALE.pdf`, 2021.

[3] Ivan Damgård, Marcel Keller, Enrique Larraia, Valerio Pastro, Peter Scholl, and Nigel P. Smart. Practical covertly secure MPC for dishonest majority - or: Breaking the SPDZ limits. In *ESORICS*, pages 1–18. Springer, 2013.

[4] Ivan Damgård, Valerio Pastro, Nigel P. Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption. In *CRYPTO*, pages 643–662. Springer, 2012.

[5] Marcel Keller. MP-SPDZ: A versatile framework for multi-party computation. In *CCS*, pages 1575–1590. ACM, 2020.