ENCRYPTO
CRYPTOGRAPHY AND
PRIVACY ENGINEERING

When Trying to Catch Cheaters Breaks the MPC

Breaking and Fixing Delayed Consistency Checks in Trident, Fantastic Four, SWIFT, and Quad

Andreas Brüggemann & Thomas Schneider

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Contact

https://encrypto.de

Me, Thinking About *SWIFT* (MPC Protocols) ...

What Happened?

ENCRYPTO
CRYPTOGRAPHY AND
PRIVACY ENGINEERING

Protocol		#Parties	Flawed Proof	Insecure	Vulnerable Implementation(s)
Trident	NDSS [CRS20]	$3+1$	**Yes**	**Yes**	–
Fantastic Four	USENIX Sec. [DEK21]	4	**Yes**	**Yes**	**Yes**
SWIFT	USENIX Sec. [KPP+21]	3	**Yes**	**Maybe**	–
Quad	PETs [HKSW+25]	$3+1$	**Yes**	**Yes**	**Yes**
Tetrad	NDSS [KPR+22]	$3+1$	**Yes**	**No**	**Yes**

[CRS20] H. Chaudhari, R. Rachuri, and A. Suresh. “Trident: Efficient 4PC Framework for Privacy Preserving Machine Learning”. NDSS. 2020

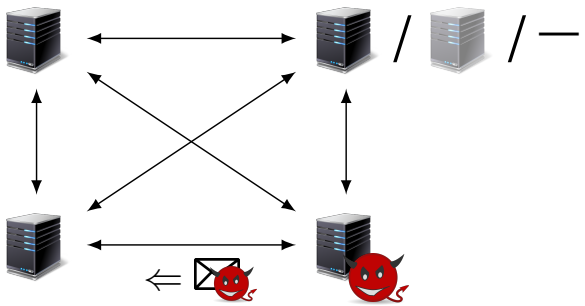
[DEK21] A. P. K. Dalskov, D. Escudero, and M. Keller. “Fantastic Four: Honest-Majority Four-Party Secure Computation With Malicious Security”. USENIX Security. 2021

[KPP+21] N. Koti, M. Pancholi, A. Patra, and A. Suresh. “SWIFT: Super-fast and Robust Privacy-Preserving Machine Learning”. USENIX Security. 2021

[HKSW+25] C. Harth-Kitzerow, A. Suresh, Y. Wang, H. Yalame, G. Carle, and M. Annavaram. “High-Throughput Secure Multiparty Computation with an Honest Majority in Various Network Settings”. PoPETs 1 (2025)

[KPR+22] N. Koti, A. Patra, R. Rachuri, and A. Suresh. “Tetrad: Actively Secure 4PC for Secure Training and Inference”. NDSS. 2022

What's the Setting?



- 4 / 3+1 / 3 parties…
- …running MPC
- One corruption (honest majority)
- Active adversary
- Standard simulation-based security

Crash Course in Trident [CRS20]

Secret Sharing Semantics

λ_v^1, λ_v^3

m_v

$\lambda_v^1, \lambda_v^2, \lambda_v^3$

λ_v^3, λ_v^2

m_v

- $\langle \lambda_v \rangle$: $\lambda_v^1 + \lambda_v^2 + \lambda_v^3 = \lambda_v, \lambda_v^i \leftarrow\$ \mathcal{R}$
- $m_v = v + \lambda_v$

λ_v^2, λ_v^1

m_v

[CRS20] H. Chaudhari, R. Rachuri, and A. Suresh. “Trident: Efficient 4PC Framework for Privacy Preserving Machine Learning”. NDSS. 2020

ENCRYPTO
CRYPTOGRAPHY AND
PRIVACY ENGINEERING

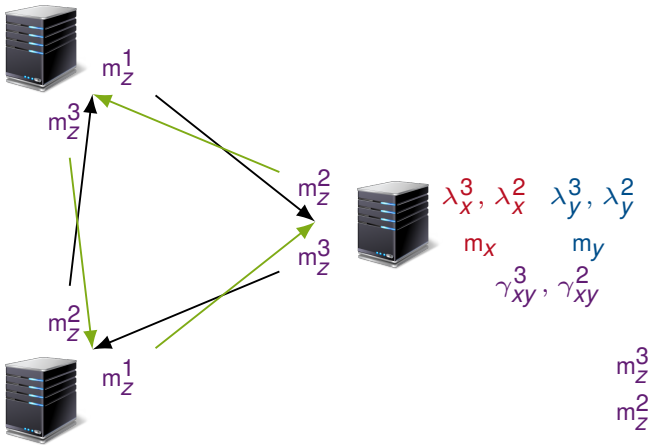
- Goal: Multiply $x \cdot y = z$
- Non-interactively sample $\langle \lambda_z \rangle$ for $\lambda_z \leftarrow\$ \mathcal{R}$.
- Compute m_z?

$$\begin{aligned} m_z &= z + \lambda_z \\ &= x \cdot y + \lambda_z \\ &= (m_x - \lambda_x) \cdot (m_y - \lambda_y) + \lambda_z \\ &= m_x m_y - m_x \lambda_y - m_y \lambda_x + \lambda_x \lambda_y + \lambda_z \end{aligned}$$

- Precompute $\langle \gamma_{xy} \rangle$ for $\gamma_{xy} = \lambda_x \lambda_y$.
- Compute $\langle m_z \rangle = m_x m_y - m_x \langle \lambda_y \rangle - m_y \langle \lambda_x \rangle + \langle \gamma_{xy} \rangle + \langle \lambda_z \rangle$.
- Recover m_z.

Crash Course in Trident [CRS20]

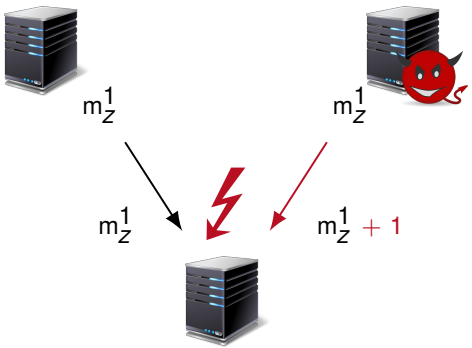
Multiplication II: Communication

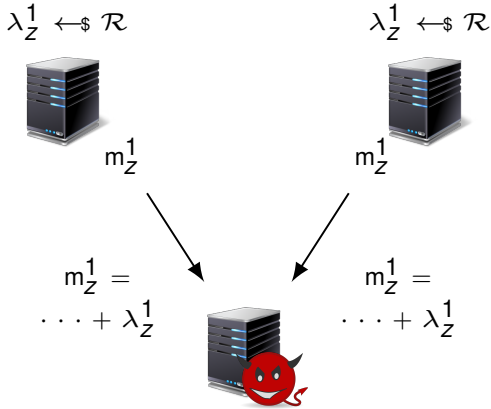


$$m_z^3 = -m_x \lambda_y^3 - m_y \lambda_x^3 + \gamma_{xy}^3 + \lambda_z^3$$
$$m_z^2 = -m_x \lambda_y^2 - m_y \lambda_x^2 + \gamma_{xy}^2 + \lambda_z^2$$

Crash Course in Trident [CRS20]

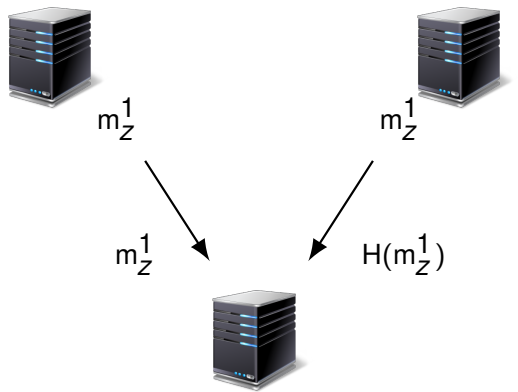
Multiplication III: Redundant Transmissions





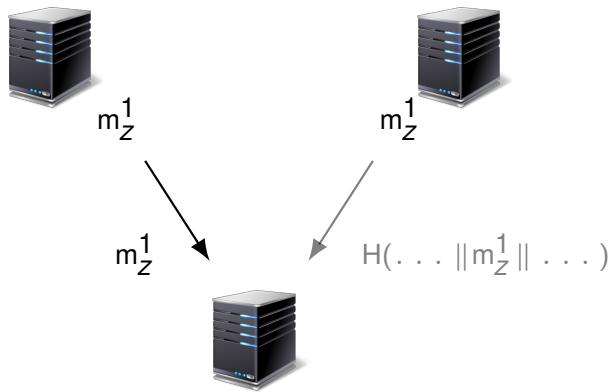
Optimizations in Trident [CRS20]

Hashing



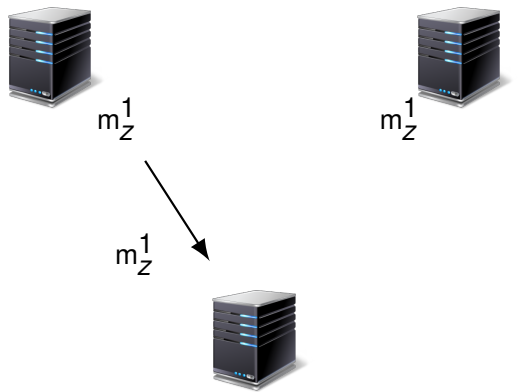
Optimizations in Trident [CRS20]

Batching



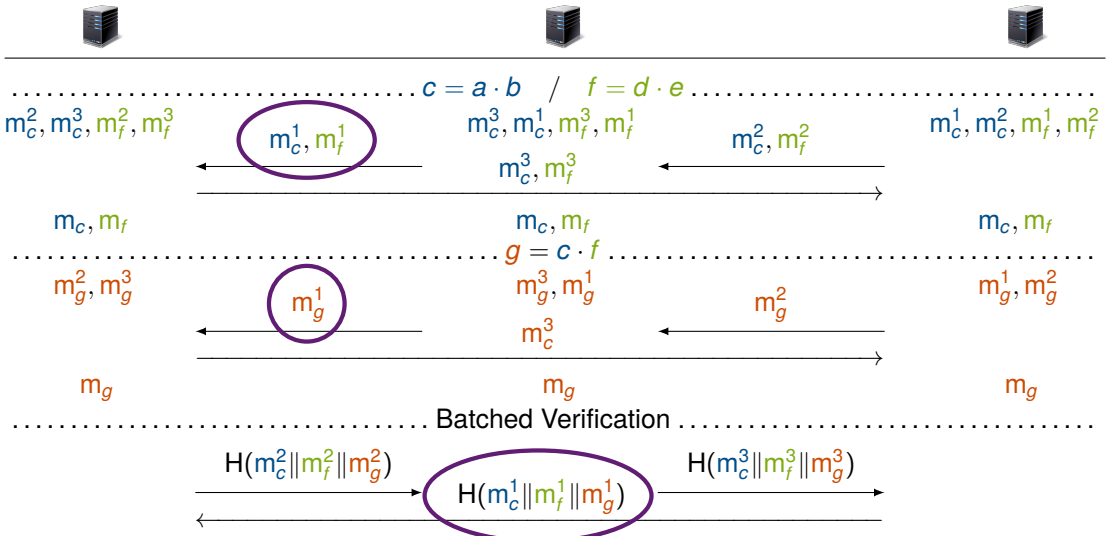
Optimizations in Trident [CRS20]

Optimistic Execution I



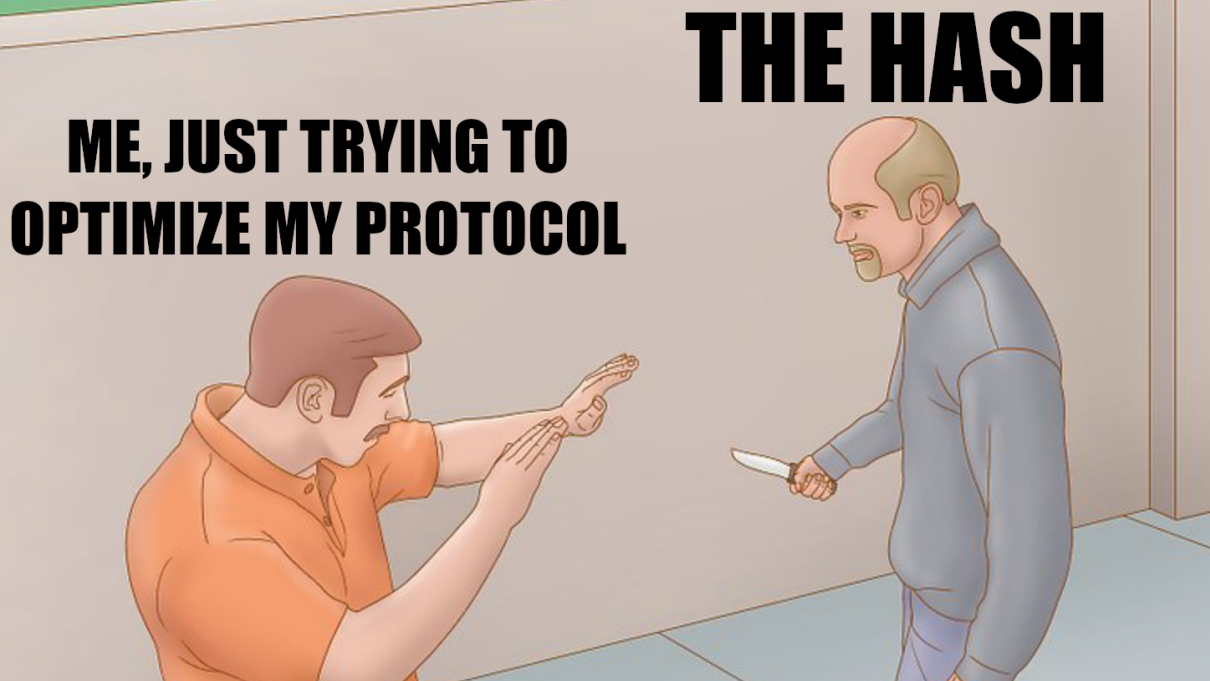
Optimizations in Trident [CRS20]

Optimistic Execution II

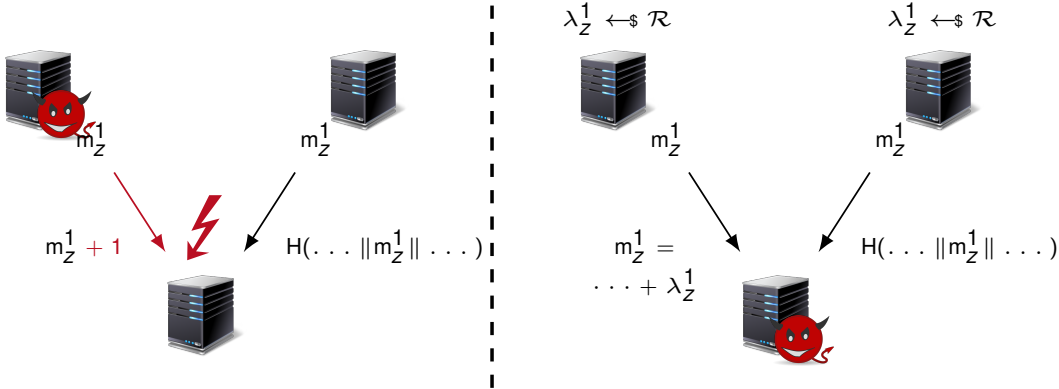


THE HASH

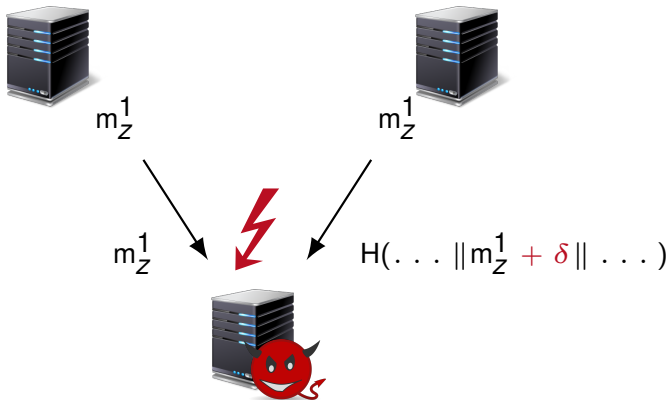
ME, JUST TRYING TO
OPTIMIZE MY PROTOCOL



Recap: Redundant Transmissions in Trident [CRS20]



The Issue



Breaking the Protocol

$c = a \cdot b \quad / \quad f = d \cdot e$

$m_c^2, m_c^3, m_f^2, m_f^3$ | m_c^1, m_f^1 | $m_c^3, m_c^1, m_f^3, m_f^1$ | m_c^2, m_f^2 | $m_c^1, m_c^2, m_f^1, m_f^2$

$m_c^3 + \mathbf{1}, m_f^3$

m_c, m_f | $m_g^1 - \lambda_f^1 = -(m_c + 1)\lambda_f^1 - m_f \lambda_c^1 + \gamma_{cf}^1 + \lambda_g^1$ | $m_c + \mathbf{1}, m_f$

m_g^2, m_g^3 | m_g^1 | m_g^3, m_g^1 | ... | $m_g^1 - \lambda_f^1, \ldots$

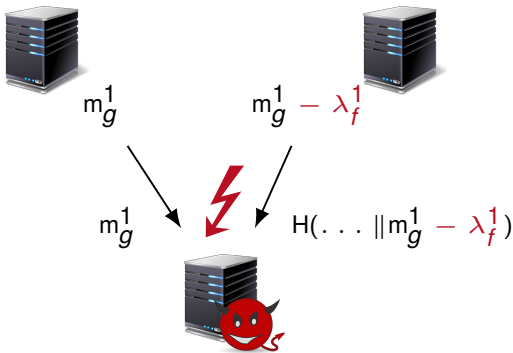
m_c^3

m_g | ... | ...

Batched Verification

$H(m_c^2 \| m_f^2 \| m_g^2)$ | $H(m_c^1 \| m_f^1 \| m_g^1 - \lambda_f^1)$ | $H(m_c^3 \| m_f^3 \| m_g^3)$

Extracting Private Data



$$m_f = f + \lambda_f^1 + \lambda_f^2 + \lambda_f^3$$

$$\widetilde{\lambda_f^1} = m_f - \widetilde{f} - \lambda_f^2 - \lambda_f^3$$

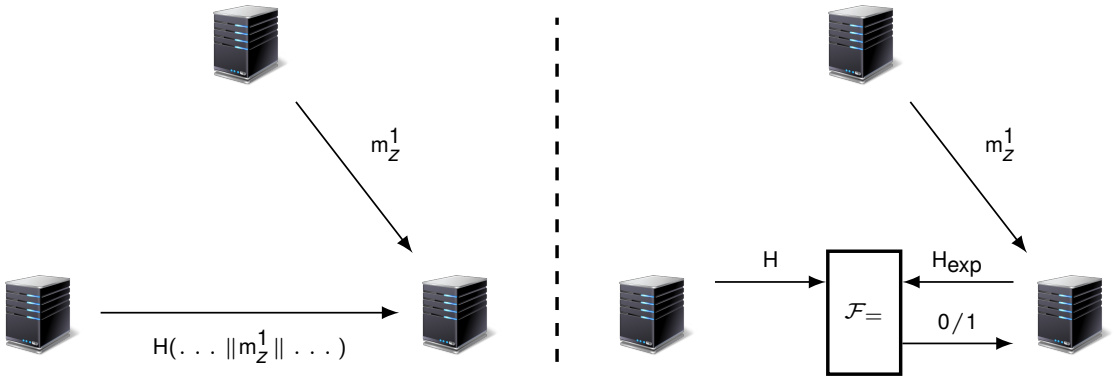
$$H(\ldots \| m_g^1 - \widetilde{\lambda_f^1}) \stackrel{?}{=} H(\ldots \| m_g^1 - \lambda_f^1)$$

~~Optimistic Execution~~

Layer-Wise Verification

Repairing the Protocol

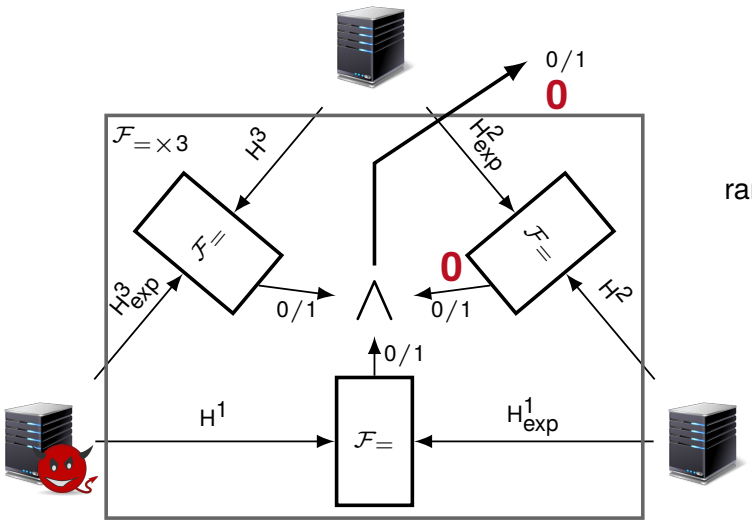
The Insufficient Fix



$\Rightarrow$ still allows testing for one value

Repairing the Protocol

The Efficient Fix



random lin. combination check:

$$\sum_{i=1}^{3}[r^i] \cdot \left([\mathrm{H}^i] - [\mathrm{H}^i_{\mathrm{exp}}]\right)$$

using Trident [CRS20] with layer-wise verification

How Could This Happen?

Aren't MPC Protocols to be Proven Secure?

ENCRYPTO
CRYPTOGRAPHY AND
PRIVACY ENGINEERING

1. Protocol Design
2. Security Proof ✔
3. Small, unsuspicious-looking Optimization
4. (no new proof) 🗲

Vulnerable Implementations

All Disclosed to Maintainers in Advance, now Fixed

ENCRYPTO
CRYPTOGRAPHY AND
PRIVACY ENGINEERING

Protocol	Insecure?	Vulnerable Implementation in …		
		MP-SPDZ [Kel20]	HPMPC [HK22]	Orq [BBM+25]
Trident [CRS20]	**Yes**	–	–	–
Fantastic Four [DEK21]	**Yes**	✔	✔	✔
SWIFT [KPP+21]	**Depends**	–	–	–
Quad [HKSW+25]	**Yes**	–	✔	–
Tetrad [KPR+22]	**No**	–	✔	–

⇒ **all vulnerabilities fixed**

[Kel20] M. Keller. “MP-SPDZ: A Versatile Framework for Multi-Party Computation”. CCS. 2020

[HK22] C. Harth-Kitzerow. “HPMPC: High-Performance Implementation of Secure Multiparty Computation (MPC) Protocols”. GitHub. `https://github.com/chart21/hpmpc/`. 2022

[BBM+25] E. Baum, S. Buxbaum, N. Mathai, M. Faisal, V. Kalavri, M. Varia, and J. Liagouris. “ORQ: Complex Analytics on Private Data with Strong Security Guarantees”. SOSP. 2025

ENCRYPTO
CRYPTOGRAPHY AND PRIVACY ENGINEERING

Without affecting security,
we delay all **questions** until the end (now!)

Paper

https://ia.cr/2026/234

MP-SPDZ PoC

https://encrypto.de/code/F4attack

Artifact Evaluated

Website

https://encrypto.de