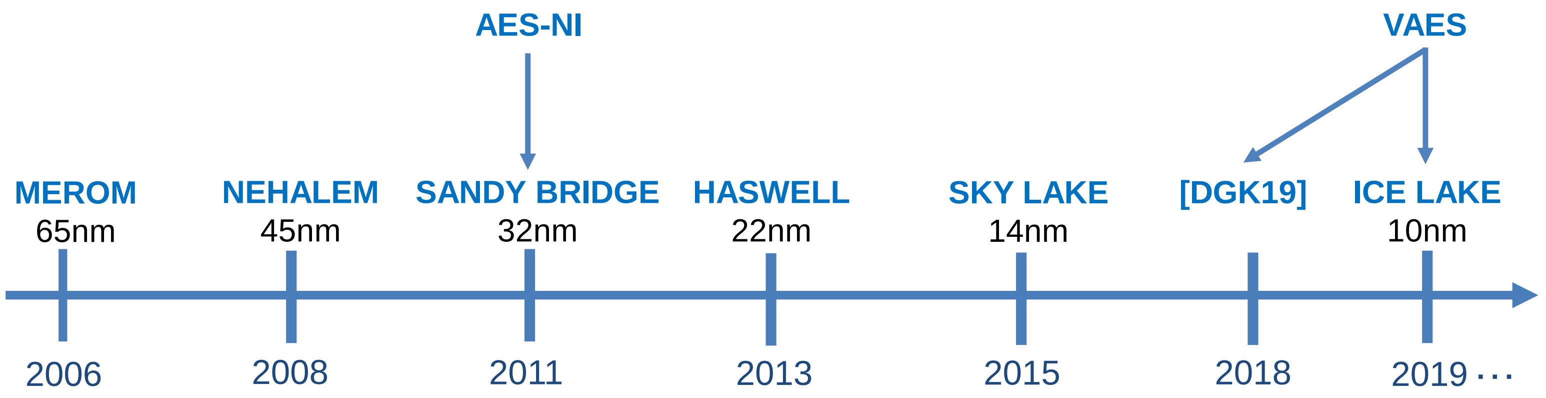
VASA: Vector AES Instructions for Security Applications

Jean-Pierre Münch, Thomas Schneider, Hossein Yalame

ACSAC'21

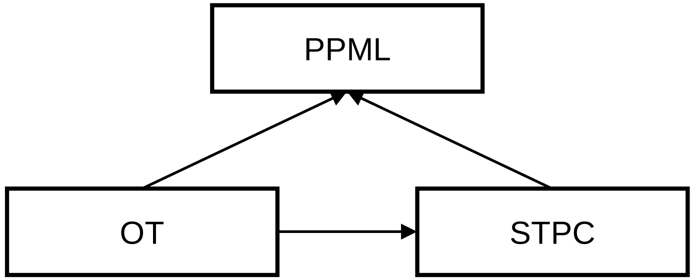
TECHNISCHE UNIVERSITÄT DARMSTADT

ENCRYPTO CRYPTOGRAPHY AND PRIVACY ENGINEERING

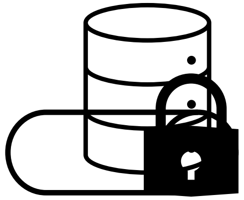


Motivation – Security Applications

- AES is the most ubiquitous symmetric cipher and used in many applications
  - Disk encryption / Transmission encryption
  - Post-quantum signature schemes [DGK21]
  - Secure two-party computation (STPC)
- STPC protocols are implemented with AES



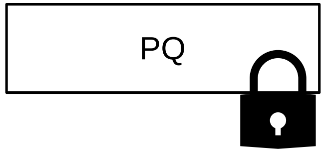
Motivation - Some Use Cases of Parallel AES



Disk Encryption / Transmission Encryption [DGK19]

Trivial parallelism → Solve once

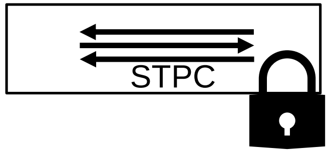
Difficulty?



Post-Quantum Signatures [DG19a, DG19b, DGK21]

Accelerate PRFs → and PRGs

This work:



Secure Two Party Computation

Complex data dependencies from circuits →

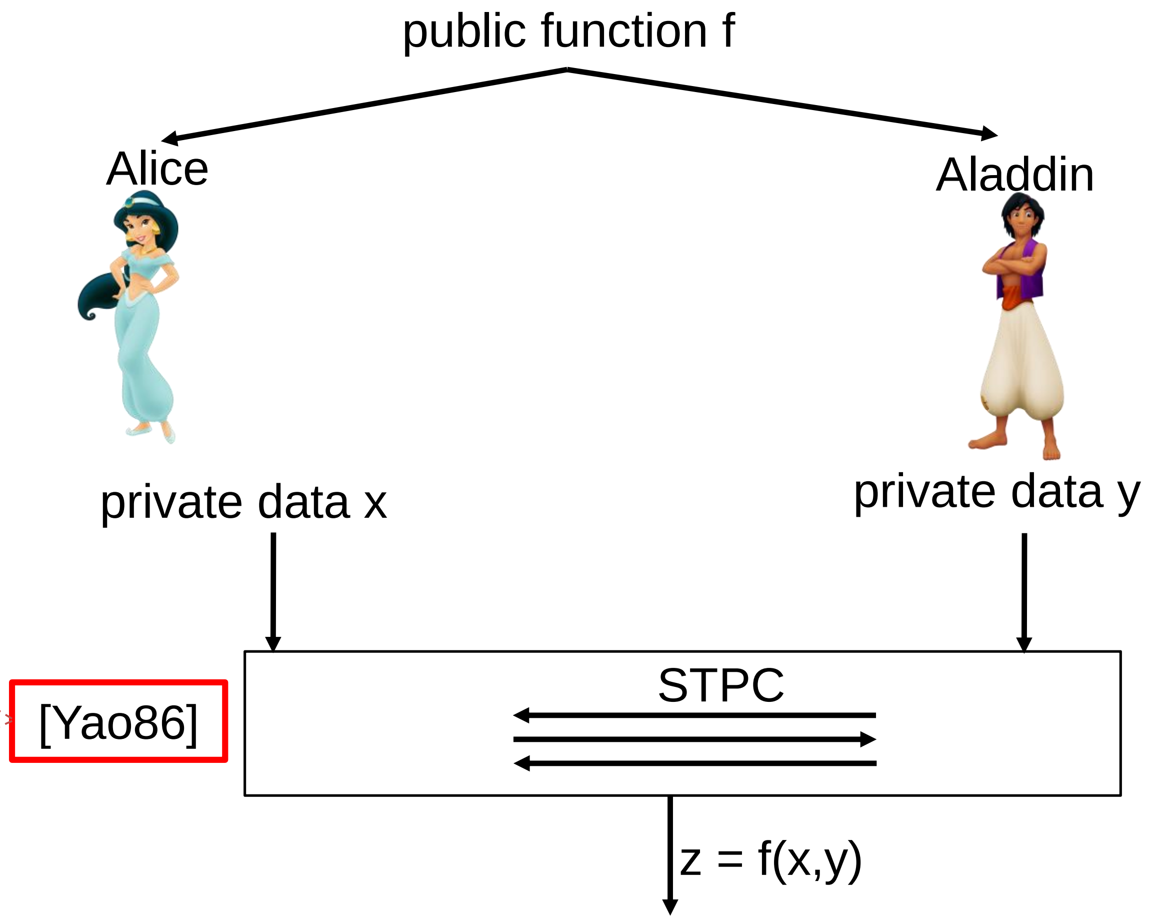
Our challenge: Batch enough independent AES calls together for the AES hardware units
Our goal: Improve efficiency of STPC protocol implementations using VAES

Preliminaries - Secure Two-Party Computation (STPC)

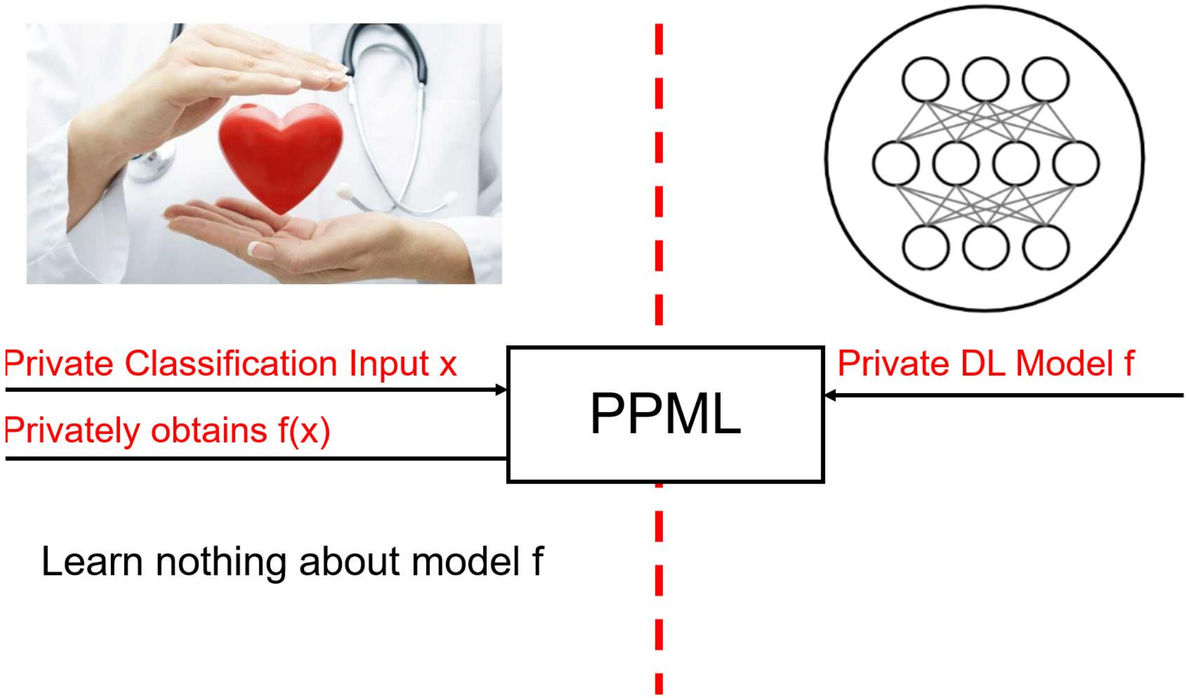
- Compute arbitrary function f
- On private data x, y
- Without trusted third party
- Reveal nothing but the result z = f(x,y)

Free-XOR [KS08]

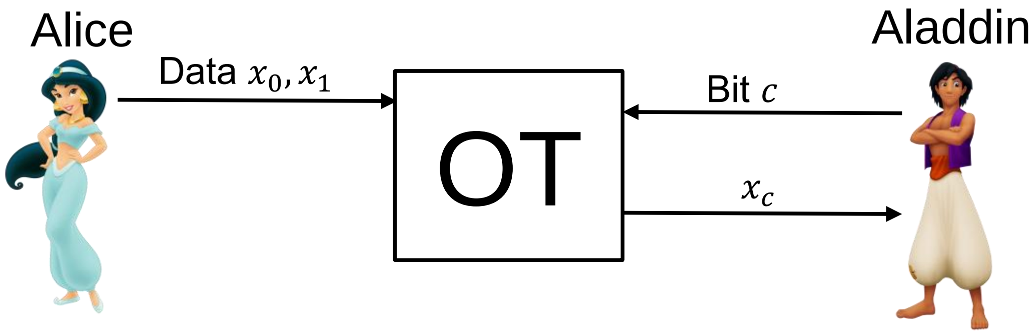
Fixed-key Garbling [BHK13]
Via AES



Privacy-preserving Machine Learning (PPML) and Oblivious Transfer (OT)

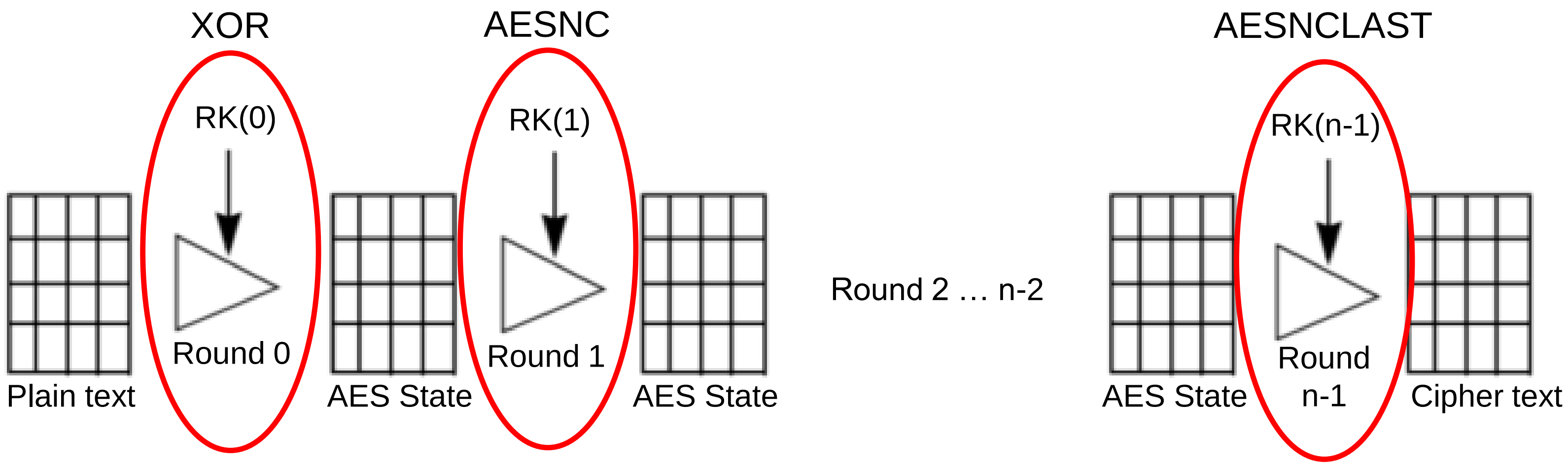


Privacy-preserving Machine Learning (PPML)



Oblivious Transfer (OT) Protocol

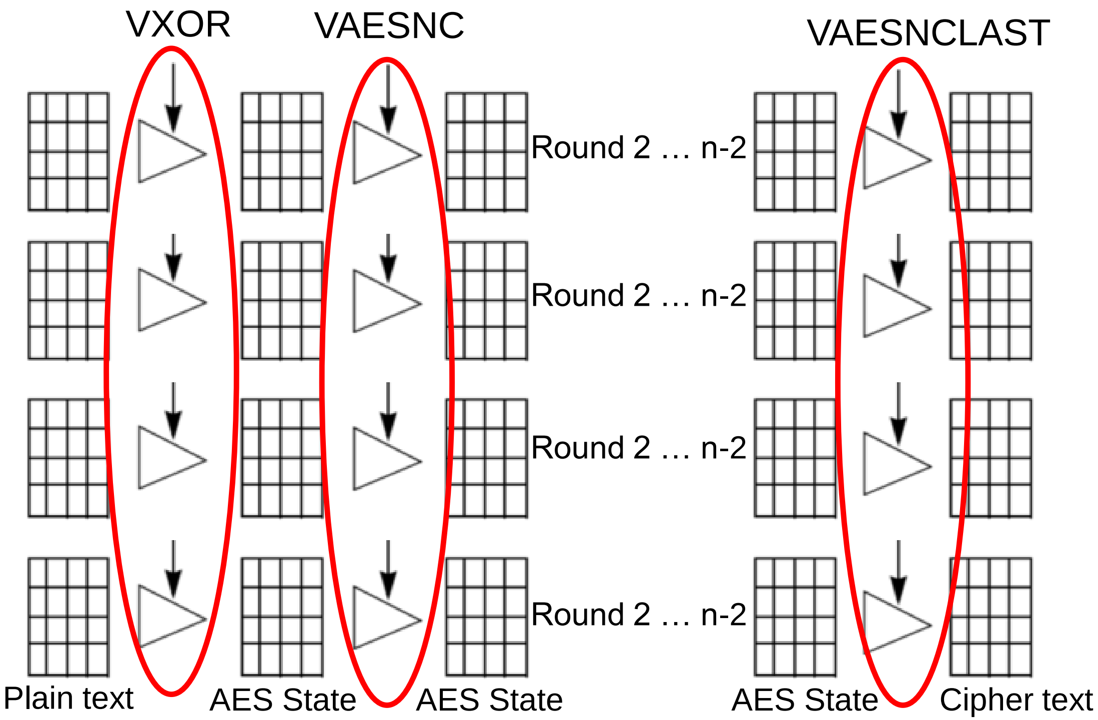
AES



AES-128: n=10
AES-192: n=12
AES-256: n=14

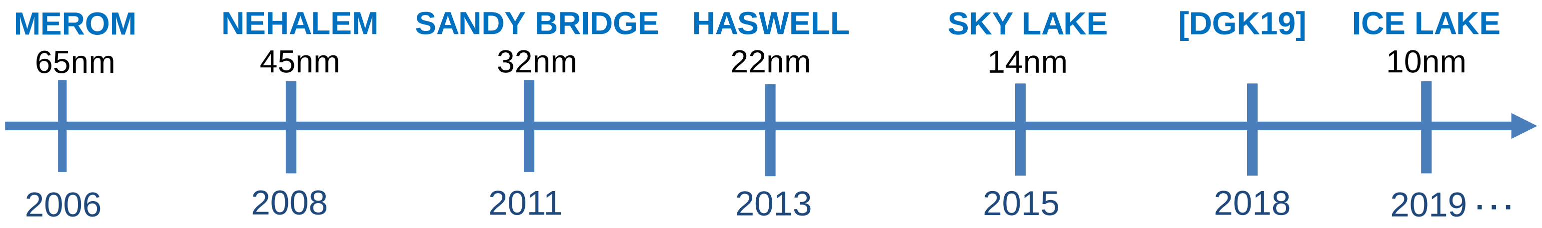
Vectorized AES (VAES) [DGK19]

- Importance of batching data and microarchitectural properties [DGK19]
  - Block ciphers: AES-CTR, AES-CBC, AES-GCM, and AES-GCM-SI.
    - Up to 4×performance improvements



AES-NI vs. VAES [Fog]

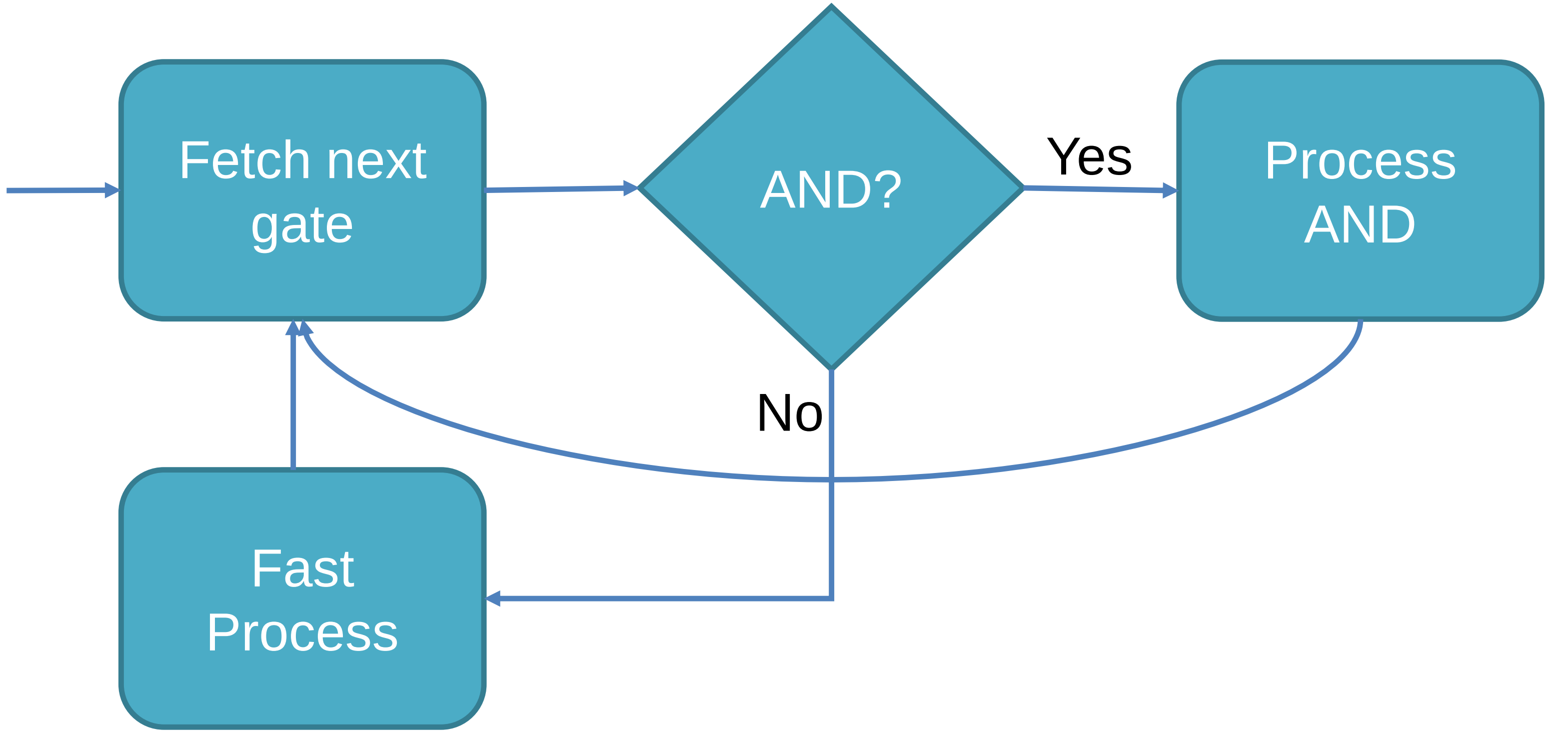
Year	Architecture	Acceleration	Width [op]	Latency [cycles]	Throughput [CPI]	Minimal Batch Size
2013	Haswell	AES-NI	1	7	1	7
2015	Sky lake	AES-NI	1	4	1	4
2019	Ice lake	VAES	1/2/4	3/3/3	0.5/0.5/1	6/12/12



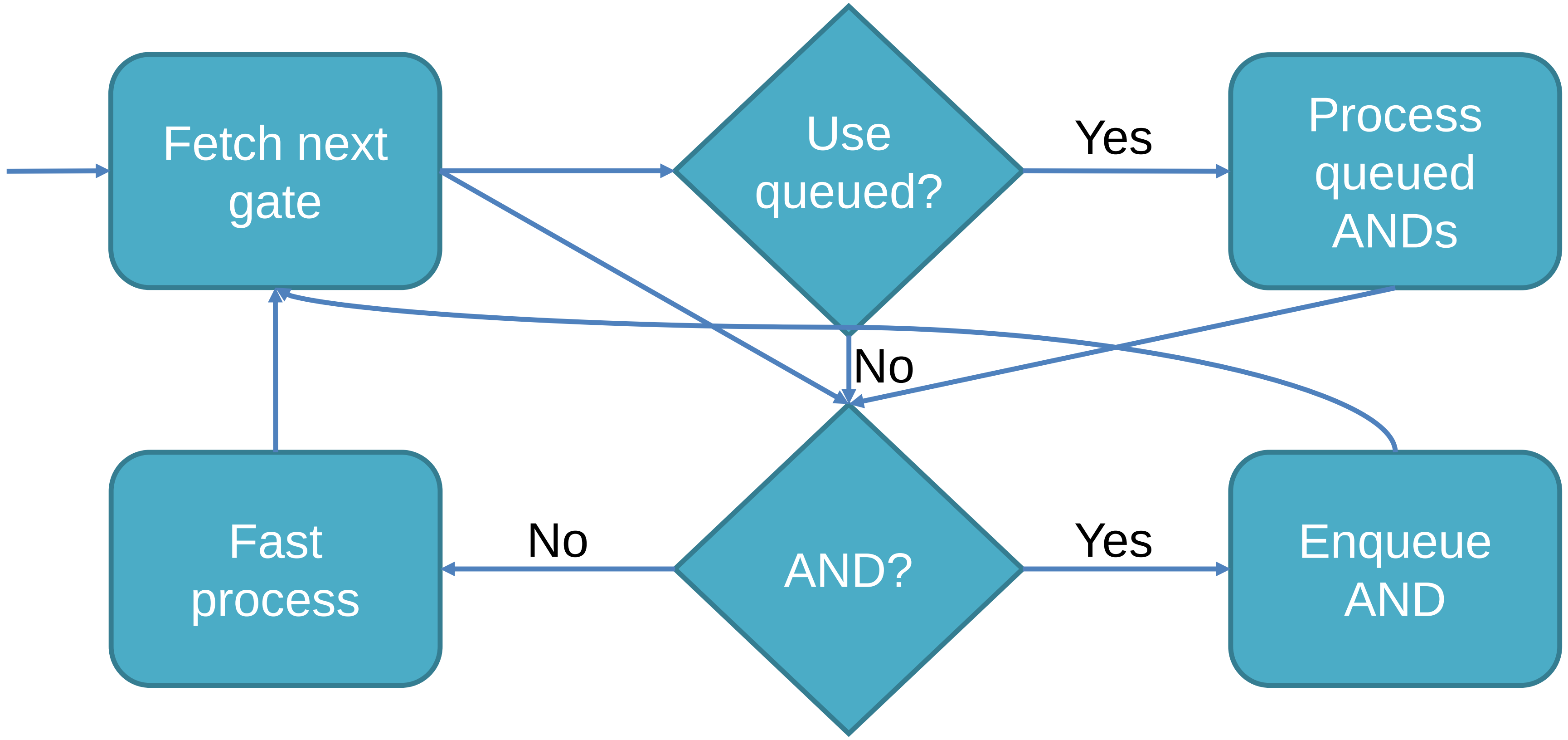
Our Contributions

- Automatic batch identification and computation techniques for efficient use of AES in complex security applications
- First performance measurements for VAES in the area of STPC
  - STPC frameworks: ABY [DSZ15], EMP-OT [Emptool]
  - PPML framework: CrypTFlow2 [RRKC+20]
- Open-source implementation at https://encrypto.de/code/VASA

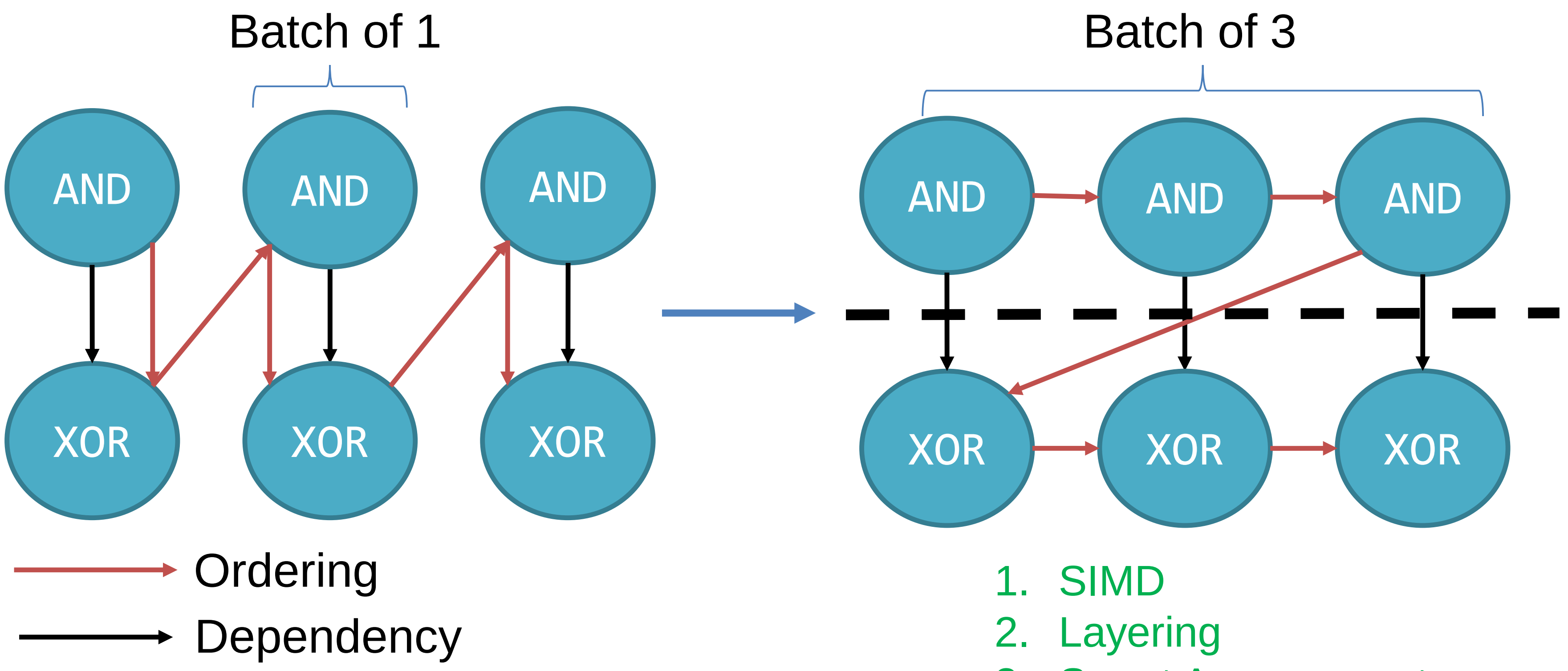
Parallelization - Baseline Scenario



Parallelization - Dynamic Batching



Parallelization - Static Batching



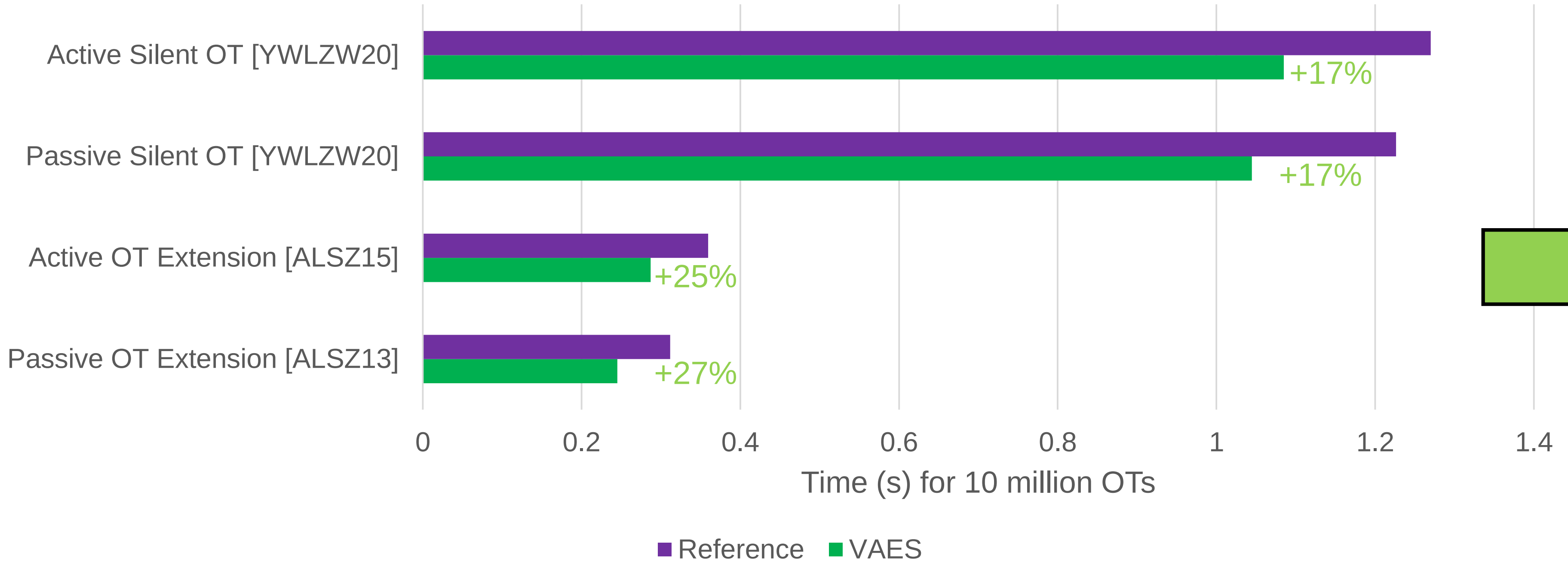
1. SIMD
2. Layering
3. Smart Arrangement

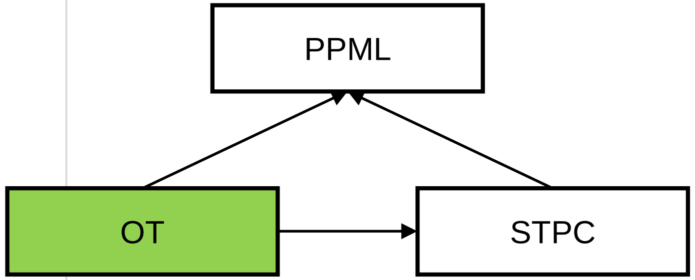
Benchmarking – Evaluation Platform

- Apple Macbook Pro
  - Intel Core i7-1068NG7, 2x16GB dual rank RAM

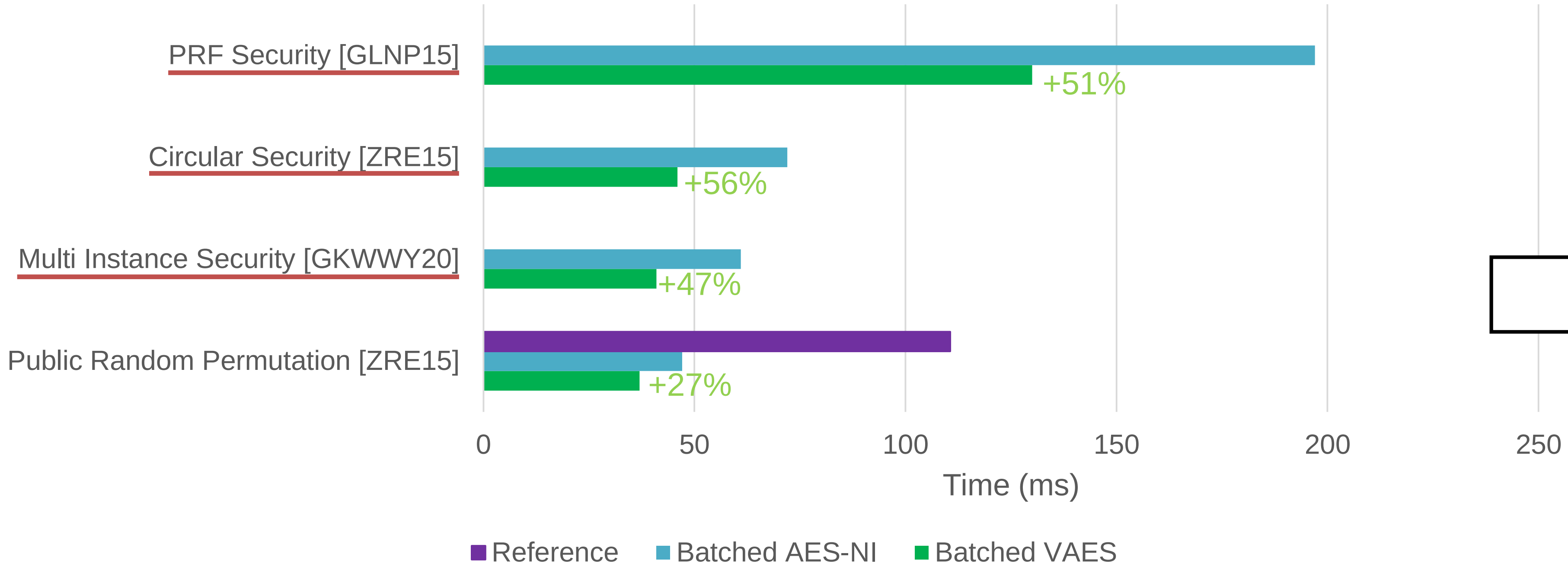
- Oblivious Transfer in EMP [EMPtool]
- Yao's Garbled Circuit in ABY [DSZ15]
- PPML in CrypTFlow2 [RRKC+20]

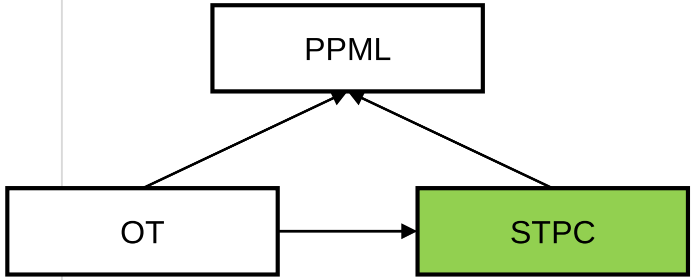
Benchmarking – OT in EMP Framework





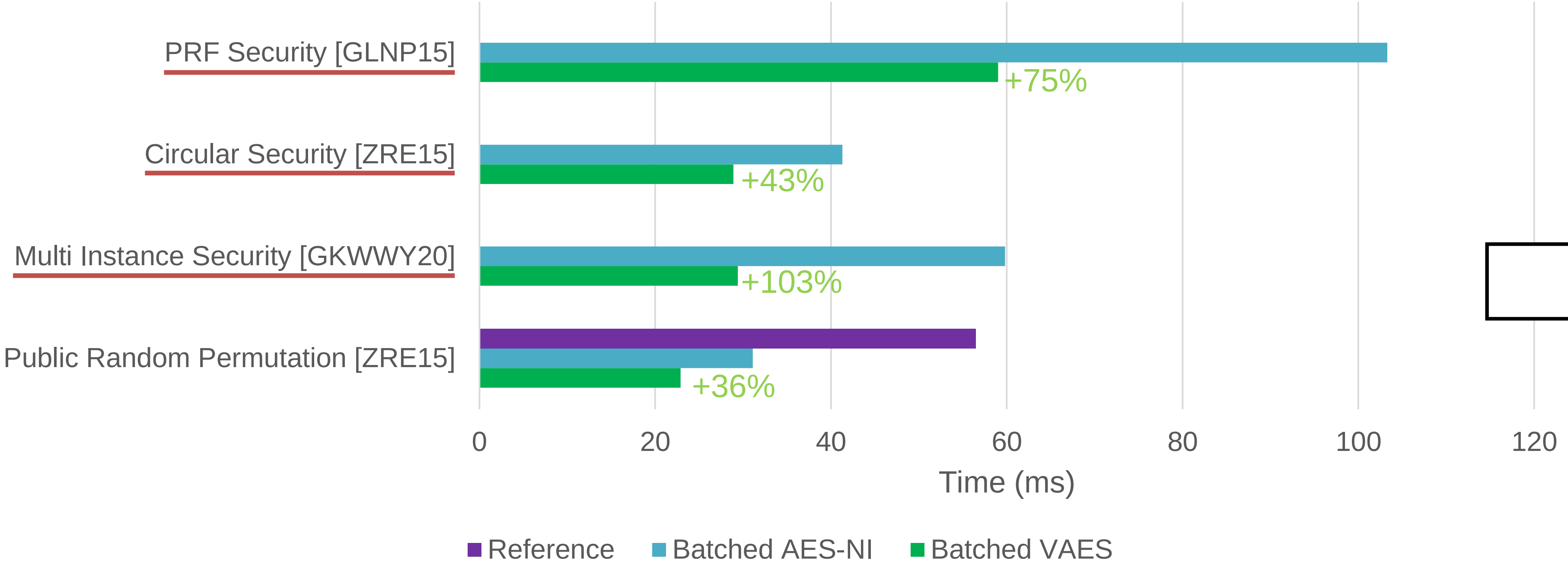
Benchmarking - Yao Garbling in ABY

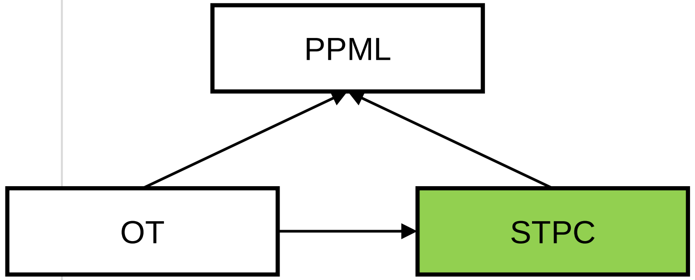




Average runtimes for applications AES, SHA-1, SCS-PSI, and Phasing-PSI.

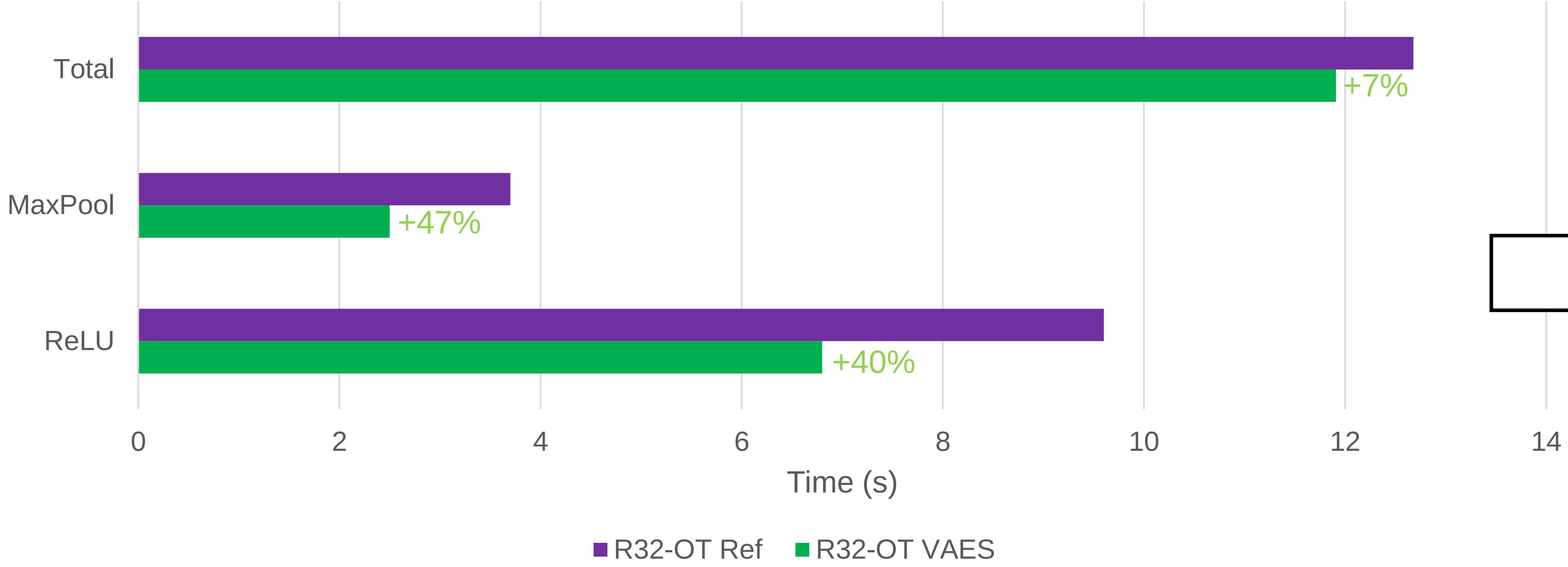
Benchmarking - Yao Evaluation in ABY

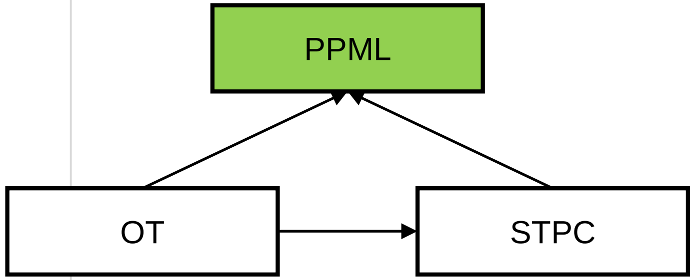




Average runtimes for applications AES, SHA-1, SCS-PSI, and Phasing-PSI.

Benchmarking – PPML in CrypTFlow2 Framework





Geometric mean of run-times using the SqueezeNetImgNet, SqueezeNetCIFAR, ResNet50, and DenseNet121 networks.

Conclusion

- ✓ Computation in STPC protocols can be accelerated with VAES
- ✓ Automatic batching of AES calls to the hardware units
- ✓ VAES yields significant performance improvements for parallel circuits

Future Work

- ✓ VAES in further MPC protocols

Thank You!

That was VASA:
Vector AES Instructions for Security Applications

Paper: https://ia.cr/2021/1493

Code: https://encrypto.de/code/VASA

Questions?

https://encrypto.de/yalame

Bibliography

[ALSZ13] Gilad Asharov, Yehuda Lindell, Thomas Schneider, and Michael Zohner. **More Efficient Oblivious Transfer and Extensions for Faster Secure Computation.** In CCS, 2013.

[ALSZ15] Gilad Asharov, Yehuda Lindell, Thomas Schneider, and Michael Zohner. **More Efficient Oblivious Transfer Extensions with Security for Malicious Adversaries.** In EUROCRYPT, 2015.

[BHKR13] Mihir Bellare, Viet Tung Hoang, Sriram Keelveedhi, and Phillip Rogaway. **Efficient Garbling from a Fixed-key Blockcipher.** In IEEE S&P, 2013.

[DG19a] Nir Drucker and Shay Gueron. **Generating a Random String with a Fixed Weight.** In CSCML, 2019.

[DG19b] Nir Drucker and Shay Gueron. **CTR DRBG with Vector AES NI.** Code: https://github.com/aws-samples/ctr-drbg-with-vector-aes-ni

[DG21] Nir Drucker, Shay Gueron. **Speed Up Over the Rainbow.** In ITNG, 2021.

[DGK19] Nir Drucker, Shay Gueron, and Vlad Krasnov. **Making AES Great Again: The Forthcoming Vectorized AES Instruction.** In ITNG, 2019.

[DSZ15] Daniel Demmler, Thomas Schneider, and Michael Zohner. **ABY– A Framework for Efficient Mixed Protocol Secure Two-party Computation.** In NDSS, 2015.

[Emptool] Xiao Wang, Alex J. Malozemoff, and Jonathan Katz. **EMP-toolkit: Efficient MultiParty Computation Toolkit.** Code: https://github.com/emp-toolkit

[Fog] Agner Fog. **Lists of instruction latencies, throughputs and microoperation breakdowns for Intel, AMD, and VIA CPUs.** https://www.agner.org/optimize/instruction_tables.pdf.

[GKWWY20] Chun Guo, Jonathan Katz, Xiao Wang, Chenkai Weng, and Yu Yu. **Better Concrete Security for Half-GatesGarbling (in the Multi-instance Setting).** In CRYPTO, 2020.

[GLNP15] Shay Gueron, Yehuda Lindell, Ariel Nof, and Benny Pinkas. **Fast Garbling of Circuits Under Standard Assumptions.** In CCS, 2015.

[Intel] Agner Fog. 2021. **Lists of instruction latencies, throughputs and microoperation breakdowns for Intel, AMD, and VIA CPUs**.

[MSY21] Jean-Pierre Münch, Thomas Schneider, Hossein Yalame. **VASA: Vector AES Instructions for Security Applications.** In ACSAC, 2021.

[RR21] Mike Rosulek and Lawrence Roy. **Three Halves Make a Whole? Beating theHalf-Gates Lower Bound for Garbled Circuits.** In CRYPTO, 2021.

[RRKC+20] Deevashwer Rathee, Mayank Rathee, Nishant Kumar, Nishanth Chandran, Divya Gupta, Aseem Rastogi, and Rahul Sharma. **CrypTFlow2: Practical 2-Party Secure Inference.** In CCS, 2020.

[Yao86] Andrew Chi-Chih Yao. **How to Generate and Exchange Secrets**. In FOCS, 1986.

[YWLZW20] Kang Yang, Chenkai Weng, Xiao Lan, Jiang Zhang, and Xiao Wang. **Ferret: Fast Extension for Correlated OT with Small Communication.** In CCS, 2020.

[ZRE15] Samee Zahur, Mike Rosulek, and David Evans. **Two halves make a whole: Reducing data transfer in garbled circuits using half gates.** In EUROCRYPT, 2015.