How MPC Frameworks Use Threshold Cryptography

Marcella Hastings
University of Pennsylvania
Secure multi-party computation (MPC) in practice

- Blind auction [BCD+08]
- Fraud detection [BJSV16]
- Parameter computation [BGM17]
- Financial statistics [BLV17]
- Government applications
- Private companies
Modern end-to-end frameworks for MPC

- Goal: general-purpose tools that can execute any computation
Modern end-to-end frameworks for MPC

- Goal: general-purpose tools that can execute any computation
- Protocols assumed impractical until Fairplay [MNPS04]
Modern end-to-end frameworks for MPC

- Goal: general-purpose tools that can execute any computation
- Protocols assumed impractical until Fairplay [MNPS04]
- Performance improvements rapidly advanced state-of-the-art
Modern end-to-end frameworks for MPC

- Goal: general-purpose tools that can execute any computation
- Protocols assumed impractical until Fairplay [MNPS04]
- Performance improvements rapidly advanced state-of-the-art
Modern General-Purpose Frameworks

Questions for our survey

- Who are frameworks designed for?
- What types of MPC algorithms do they implement?
- Are they suitable for use in large-scale applications?
Modern General-Purpose Frameworks

Questions for our survey

- Who are frameworks designed for?
- What types of MPC algorithms do they implement?
- Are they suitable for use in large-scale applications?

Questions for this workshop

- Which frameworks already implement threshold schemes?
- Does this survey provide insight into what we should standardize?
Contributions
General purpose frameworks for secure multi-party computation [HHNZ19]

Survey
- Surveyed 9 frameworks and 2 circuit compilers
- Recorded protocol, feature, implementation details
- Evaluated usability criteria
Contributions
General purpose frameworks for secure multi-party computation [HHNZ19]

Survey

- Surveyed 9 frameworks and 2 circuit compilers
- Recorded protocol, feature, implementation details
- Evaluated usability criteria

Open-source framework repository

- Three sample programs in every framework
- Docker instances with complete build environments
- Documentation on compilation and execution

github.com/mpc-sok/frameworks
Findings

Our original questions

- Diverse set of threat models and protocols
- Expressive languages are suitable for real applications
- Engineering limitations
- Barriers to usability (documentation)
Findings

Our original questions

▶ Diverse set of threat models and protocols
▶ Expressive languages are suitable for real applications
▶ Engineering limitations
▶ Barriers to usability (documentation)

Threshold questions

▶ A growing proportion of frameworks support threshold operations
▶ They all do it via secret sharing
Frameworks and protocol families

Garbled circuit

EMP-toolkit
Obliv-C
ObliVM
TinyGarble

Multi-party circuit based

ABY

Hybrid

SCALE-MAMBA
Sharemind
PICCO

Wysteria

swanky

7 / 17
Frameworks and protocol families (2020)

- Garbled circuit
  - EMP-toolkit
  - Obliv-C
  - ObliVM
  - TinyGarble

- MP-SPDZ
  - ABY
  - MOTION-HyCC
  - ABY$^3$

- Hybrid
  - SCALE-MAMBA
  - Sharemind
  - PICCO
  - EzPC
  - JIFF

- Multi-party circuit based
  - Wysteria
  - MPyC
Frameworks and protocol families (2020)

- Garbled circuit
  - EMP-toolkit
  - Obliv-C
  - ObliVM
  - TinyGarble

- Multi-party circuit based
  - ABY
  - MOTION-HyCC
  - ABY^3

- Hybrid
  - SCALE-MAMBA
  - Sharemind
  - PICCO
  - EzPC
  - JIFF
  - MPyC
  - Wysteria
  - FRESCO
Garbled circuit protocols

Introduced by [Yao82, Yao86]

- Functions represented as Boolean circuits
- Often 2-party semi-honest, but exceptions are growing
Frameworks and protocol families (2020)

Garbled circuit

- EMP-toolkit
- Obliv-C
- ObliVM
- TinyGarble

Multi-party circuit based

- ABY
- MOTION-HyCC
- ABY³

Hybrid

- SCALE-MAMBA
- Sharemind
- PICCO
- EzPC
- JIFF

- MP-SPDZ
- MPyC
- Wysteria
Multi-party circuit-based protocols

Introduced by [GMW87, BGW88, CCD88]

- Functions represented as Boolean or arithmetic circuits
- Data represented as linear secret shares
- Various threat models and protocol types (information-theoretic or cryptographic)
Frameworks and protocol families (2020)

- Garbled circuit:
  - EMP-toolkit
  - Obliv-C
  - ObliVM
  - TinyGarble

- Multi-party circuit based:
  - ABY
  - MOTION-HyCC
  - ABY³

- Hybrid:
  - SCALE-MAMBA
  - Sharemind
  - PICCO
  - EzPC
  - JIFF
  - SWANKY
  - MP-SPDZ
  - FRESCO
  - Wysteria
  - MPyC
Hybrid protocols

- Integrates optimized subprotocols for common functions
  - Bitwise operators in arithmetic settings
  - Matrix operations
- Seamless front-end experience (no explicit protocol selection)
- Currently: One-to-one mapping from operations to protocols
Hybrid protocols

- Integrates optimized subprotocols for common functions
  - Bitwise operators in arithmetic settings
  - Matrix operations
- Seamless front-end experience (no explicit protocol selection)
- Currently: One-to-one mapping from operations to protocols
What does “threshold” mean for MPC?

Threshold adversary

- Up to $k$ corrupted parties cannot learn honest inputs
- They can block output (sometimes)
- This is a common threat model, so I didn’t survey it
What does “threshold” mean for MPC?

Threshold adversary

- Up to $k$ corrupted parties cannot learn honest inputs
- They can block output (sometimes)
- This is a common threat model, so I didn’t survey it

Threshold output

- A qualified group of $k$ parties can retrieve output
- This might only be true at certain points in the protocol
Threshold secret sharing schemes used in MPC frameworks

Shamir sharing

- Used by SCALE-MAMBA, PICCO, MP-SPDZ, MPyC, JIFF
- Mostly use classic [Shamir ’79]
- Standards: NISTIR 8214, ISO/IEC 19592-2

Replicated sharing

- Used by SCALE-MAMBA, MP-SPDZ, ABY³
- Schemes based on [Benaloh and Leichter ’09] [Araki et al. ’16]

Details of these findings are in the frameworks wiki
github.com/mpc-sok/frameworks/wiki
Performance evaluation

In theory
Measure circuit size
Measure rounds and volume of communication
Performance evaluation

In theory
Measure circuit size
Measure rounds and volume of communication

In practice
- Many frameworks don't produce traditional circuits

See [Keller '20] for performance comparison and caveats

Lesson for standardizers
Be careful about abstractions when you standardize a “whole” MPC scheme
Performance evaluation

In theory
Measure circuit size
Measure rounds and volume of communication

In practice
▶ Many frameworks don’t produce traditional circuits
▶ Non-crypto variables can wildly affect performance (network channels, message batching, IO, language)
▶ See [Keller ’20] for performance comparison and caveats

Lesson for standardizers
Be careful about abstractions when you standardize a “whole” MPC scheme
How MPC Frameworks Use Threshold Cryptography

Marcella Hastings
University of Pennsylvania

github.com/mpc-sok/frameworks