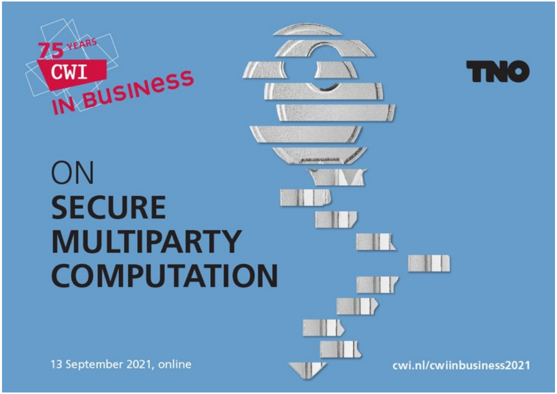
Introduction to Secure Multiparty Computation

Ronald Cramer[1,2] & Lisa Kohl[1]

[1]CWI Cryptology Group, [2]Math. Inst. Leiden U.



Part I: What is Secure Multiparty Computation?

Classical Cryptography vs. Secure MPC (1/2)

Classical cryptographic tasks pertain to data communication*:*

- *Data Confidentiality:* `(public key) encryption`
- *Data Authenticity:* `message authentication codes`
- *Non-Repudiation:* `digital signatures`

These are all part of the realm of **uni-lateral security**:

"*protecting the good guys from the bad guys*"

Note: Bad guys *outside* the system (e.g. eavesdropper)

Classical Cryptography vs. Secure MPC (2/2)

Multi-Lateral Security*:*

- multi-party processing *on mutually private data*
- *with the purpose of enabling* controlled release *of information*
- *in the face of* mutual mistrust or conflicting interests
- *and in the* absence *of "trusted arbiter".*

Area is *fundamentally different* from uni-lateral security:

- Meaningful in *world-of-two*!

  Indeed: security of communication is w.r.t. "a third".
- Requires *dedicated crypto*; not just encryption, signatures

  E.g., just encrypting bids in auction is not a solution.

Examples

- (TOY) *Two-party Dating*:

  Goal (part 1):
  X, Y jointly determine possible *mutual attraction* and each of X, Y learns the outcome: yes/no.

  Unavoidable:

  1. *fancying* party infers other's position from outcome.
  2. *non-fancying* party knows outcome in advance.

  Goal (part 2): *face-saving*, i.e., *non-fancying* party remains *ignorant* about other's position.

- *Historical Toy Example* (1st, 1982): *Millionaires Problem*.

- *Voting/Elections*:

  Goal:
  tally but keep individual votes secret.

- *Auctions*:

  Goal:
  reveal winner but keep bids secret
  (even from auctioneer).

- *Benchmarking*:

  Goal:
  determine "best-practise" without revealing trade-secrets.

  e.g., companies jointly compute average salaries or other statistics without revealing anything else to each other.

- Goldwasser/Micali/Rackoff (1985): *zero knowledge proofs*

  Goal:
  convincing sceptic of theorem yet proof remains secret.

The General Secure Multiparty Computation Problem

Let f be an arbitrary function in n-variables $X_1, \ldots, X_n$ s.t.

1. each variable takes value in a *finite* domain D
2. the function f takes value in a *finite* range R.

Now, there are n parties $P_1, \ldots, P_n$.

Each party P_i has a *private* input $x_i \in D$.

Problem: How can they jointly correctly compute the outcome

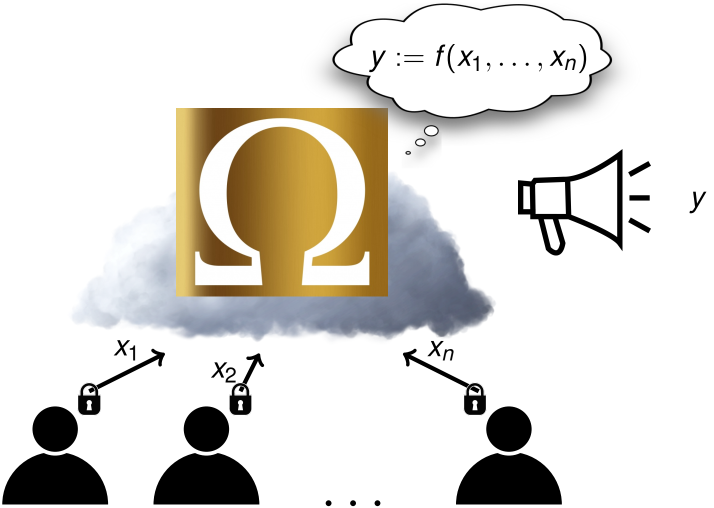
$$y := f(x_1, \ldots, x_n) \in R$$

without revealing anything about their respective private inputs?
[except for what others infer from outcome and their own inputs]

Example ("dating"): $f(x_1, x_2) = x_1 \cdot x_2 \in \{0, 1\}$ with
$x_1, x_2 \in \{0, 1\}$.

More enlightening and workable view:

How can the parties *jointly*, *without external help*, **emulate a virtual incorruptible mediator** Ω solving it for them:



Example: two-party zero knowledge proof

NB: just one party has private input.

- Prover P *privately* submits proof of theorem to Ω.
- Ω checks it.
- Ω announces to verifier V whether proof is valid.

So: how can P and V jointly simulate Ω such that

1. misbehaving P cannot lead V to accept false theorem.
2. misbehaving V remains ignorant about the proof.

Part II: How does Secure Multiparty Computation Work?

Early Major Milestones

- **Yao (1982):** *general secure two-party computation*.
  (NB) *any* two-party problem but *passive* security
- **Goldwasser/Micali/Rackoff (1985):**
  zero-knowledge proofs for NP.

Theorem (Ben-Or/Goldwasser/Wigderson, Chaum/Crépeau/Damgård 1988)

Suppose $n \geq 4$ parties arranged in complete, synchronous communication network with pair-wise secure channels.

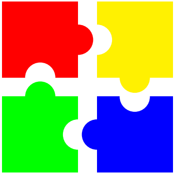
Suppose a computationally unbounded adversary corrupts $t < n/3$ parties, fully controlled towards its malicious purposes*.*

Then a virtual incorruptible mediator Ω can be emulated perfectly *and* efficiently*.*

Basic Protocol Layout (1/2)

Fact: function f can be given as "algorithmic network" of *additions* and *multiplications*, an *arithmetic circuit* C.

Basic Primitive: *dedicated "encryption" scheme* such that:

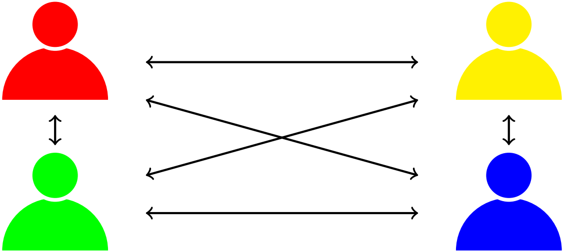


- $\leq t$ pieces: *perfectly hiding*. (**Example:** $n = 4$, $t = 1$)
  Particularly: *joint action required for decryption*

Secure Processing:

- Generation of "encryptions" of *sums and products* of "encrypted" secret values, while keeping them secret.
  NB: may require *interaction*.

The Protocol:



- Initially, each party "encrypts" its input x_i.
- Next, they recurse through circuit, keeping "encryption" of intermediate computation-results as *invariant*.
- Finally, from "encryption" of the outcome $y = f(x_1, \ldots, x_n)$, the parties "decrypt" to get y (*and only* y!).

Some Remarks

- *There is a version for $n/3 \leq t < n/2$.*

  NB: small positive error probability.
- *No computational intractability assumption required (but necessary for $t \geq n/2$).*

Specialized post-quantum crypto (e.g., SPDZ, FHE, ...):
Efficient post-quantum secure MPC for $t = n - 1$ ("only trust yourself").

Corollary

If the function f admits an efficient computer program, then the function f can be computed (post-quantum) securely and *efficiently.*

Is Secure Multiparty Computation Used in Industry?

- Auctions (2008–, Danisco)
- Voting (2011–, Helios)
- Micro-auctions on the Internet (2011 Google; back-up)
- Auctions in the electricity markets (Denmark, 2014–)
- Secure Statistical Analysis (Estonian Govt., 2014–)

Remarks:

- Basic protocol layout is universal
  (for 2-party: also Yao's garbled circuits)
- *Efficiency* $\longrightarrow$ emerging area of **secure algorithmics**
- Danisco auctions employs CDI05 pseudorandom secret sharing (CWI/Aarhus/Technion)
- Helios voting employs CDS97 scheme (CWI/IBM)
- Estonian application uses BGW88/CCD88 MPC.

Next...

- Benchmarking, credit-rating, fraud-detection, threat-intelligence analysis: under development
- Machine Learning
- Research-data-mining:

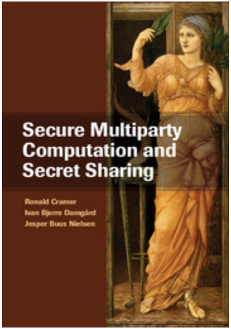
  Pharmaceutical: collaborative drug-to-drug interaction discovery
- *Distributed security*?

  micro-chips from multiple providers emulate a single one.

CAVEAT:

- *Theory: efficient "computer programs" $\Longrightarrow$ efficient circuits.*
- *But there may be substantial* **overhead**.
- *There is an additional issue, even though there are very efficient SMP protocols today: circuit must be* **oblivious**.

  I.e., computation path independent of inputs.
  This makes e.g. while loops *expensive for MPC.*
- *So: "MPC programming" is still a* **skillful art**



Secure Multiparty Computation and Secret Sharing
Ronald Cramer, Ivan Damgård, Jesper Nielsen
Cambridge University Press (July 2015)