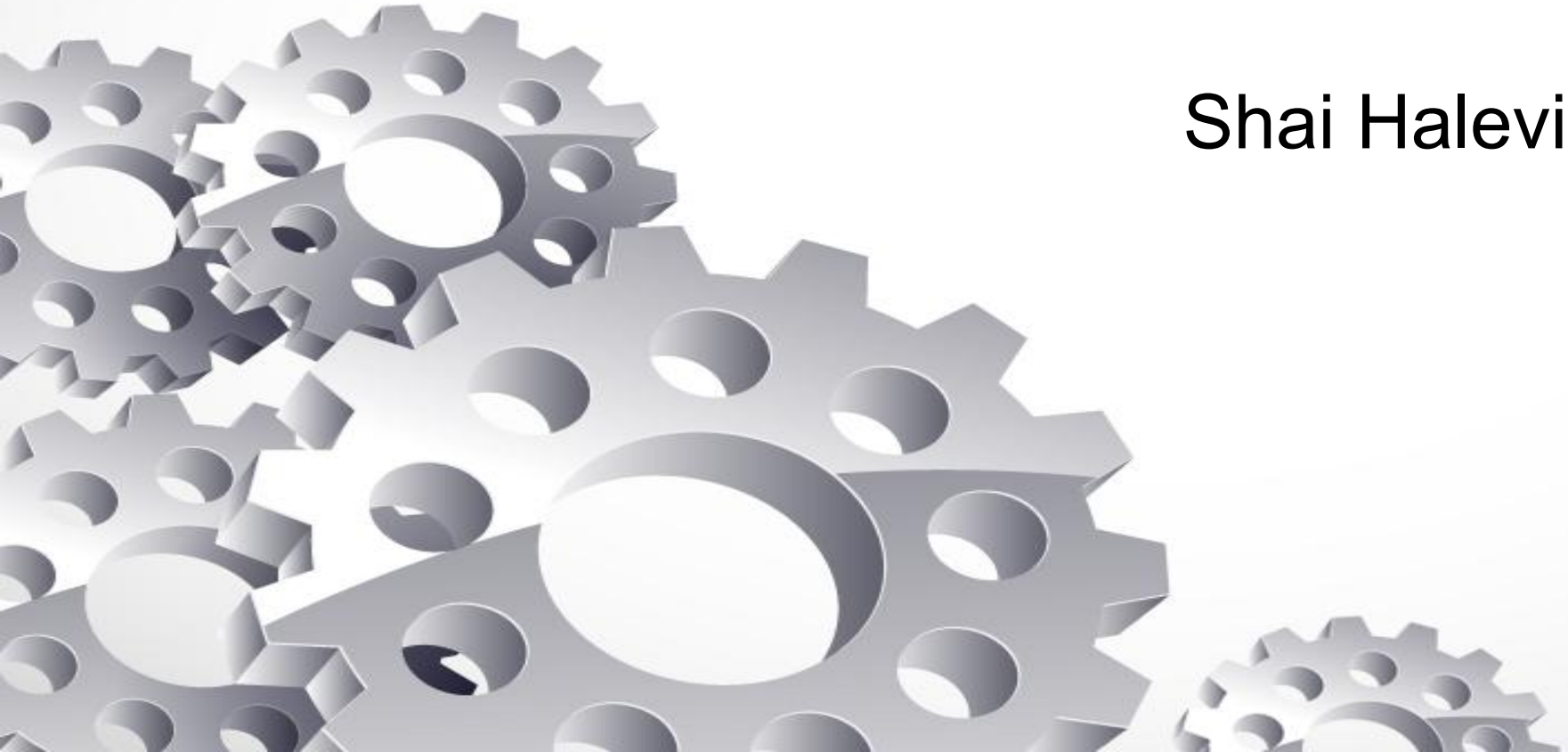
Advanced Cryptorgaphy: Promise and Challenges

Shai Halevi, IBM Research

ICMC, May 2018

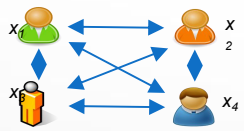


What's "Advanced Cryptography"?

- Cryptography beyond encryption, signatures
  - Protecting computation, not just data

I'll mention three technologies:

- Zero-Knowledge Proofs (ZKP)
- Secure Multi-Party Computation (MPC)
- Homomorphic Encryption (FHE)

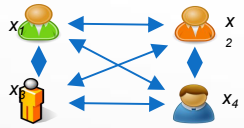


What's "Advanced Cryptography"?

- Cryptography beyond encryption, signatures
  - Protecting computation, not just data

I'll mention three technologies:

- Zero-Knowledge Proofs (ZKP)
- Secure Multi-Party Computation (MPC)
- Homomorphic Encryption (FHE)



Not in this talk:

- Oblivious RAM (ORAM)
- Attribute-Based Encryption (ABE)
- ...

Advanced Cryptography is Needed

Advanced Cryptography is

Needed

Fast enough to be useful

Advanced Cryptography is

Needed

Fast enough to be useful

Not "generally usable" yet

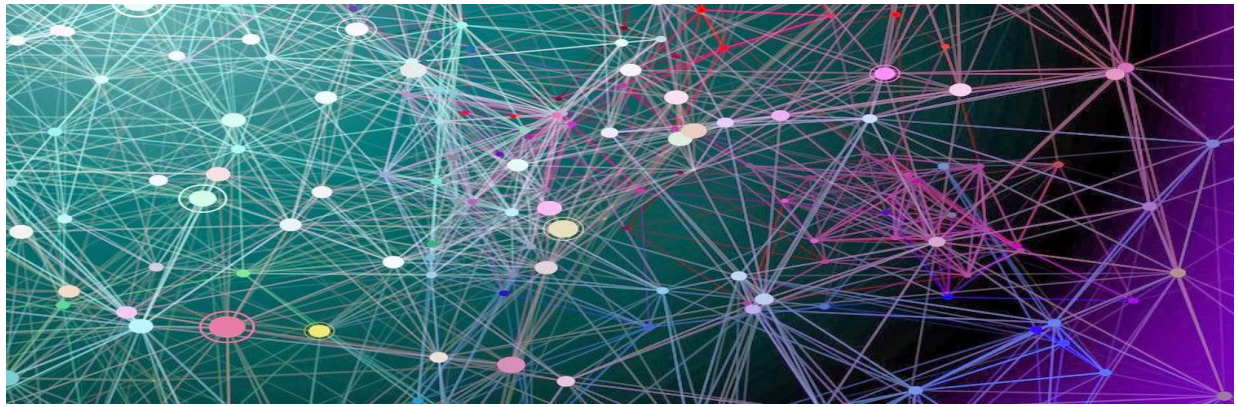
The Need for Advanced Cryptography

Your Privacy for Sale

- We give up privacy in return for services
  - location for directions, restaurant recommendation
  - health data for "personalized medicine"
  - financials for tax, investment services
  - purchace history for better ads, coupons
  - ...
- Personalized services **require** personal information
  - or so we are told

Data Abuse in the New Normal

- The entire IT industry is busy making it easier
  - Larger collections, better ways to process them



- Make no mistake: it will only get worse
  - We cannot provide the opportunity for easy abuse, and seriously expect it not to happen

The Promise of Advanced Cryptography

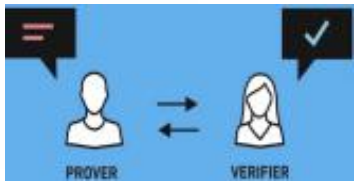
Blindfold Computation

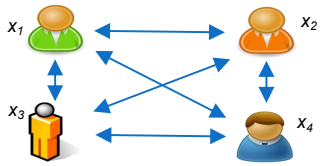
- The ability to process data without ever seeing it
  - Personalized services without access to private information

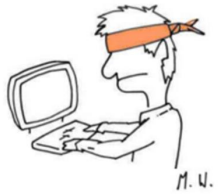
The Promise of Advanced Cryptography

Blindfold Computation

- Also useful for more traditional security issues
  - E.g., key and credential management, many more ...
  - Not the focus of this talk, though



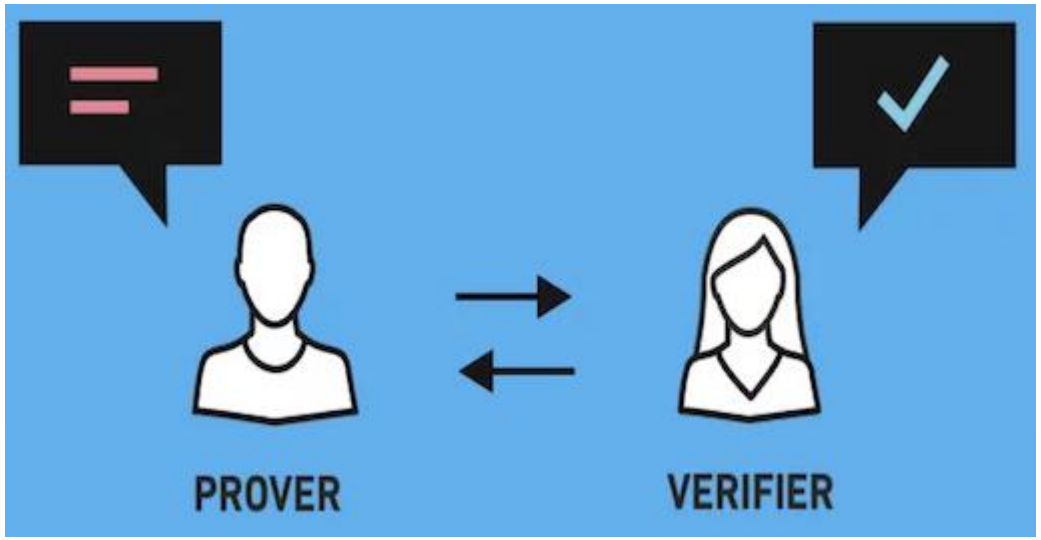




Types of Advanced Cryptography

Zero Knowledge Proofs

- I have a secret
  - I can convince you of some properties of my secret
  - Without revealing it



- Available (in principle) since the 80's [GMR'85]

Zero Knowledge Proofs

- I have a secret
  - I can convince you of some properties of my secret
  - Without revealing it

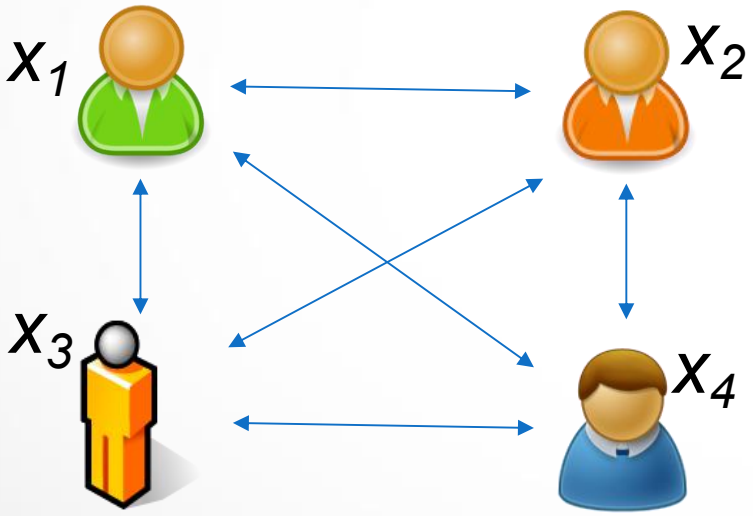
- Example: my secret is my purchase history

Zero Knowledge Proofs

- I have a secret
  - I can convince you of some properties of my secret
  - Without revealing it
- Example: my secret is my purchase history
  - I can prove that I bought 10 gallons of milk this month
    - so I can get a coupon
  - Without revealing anything else

Secure Multi-Party Computation

- We all have our individual secrets
  - We can compute a function of these secrets
  - Without revaling them to each other (or anyone else)



Goal:
Correctness: Everyone computes $y=f(x_1,\ldots,x_n)$
Privacy: Nothing but the output is revealed

- Available (in principle) since the 80's [Yao'86, GMW'86]

Secure Multi-Party Computation

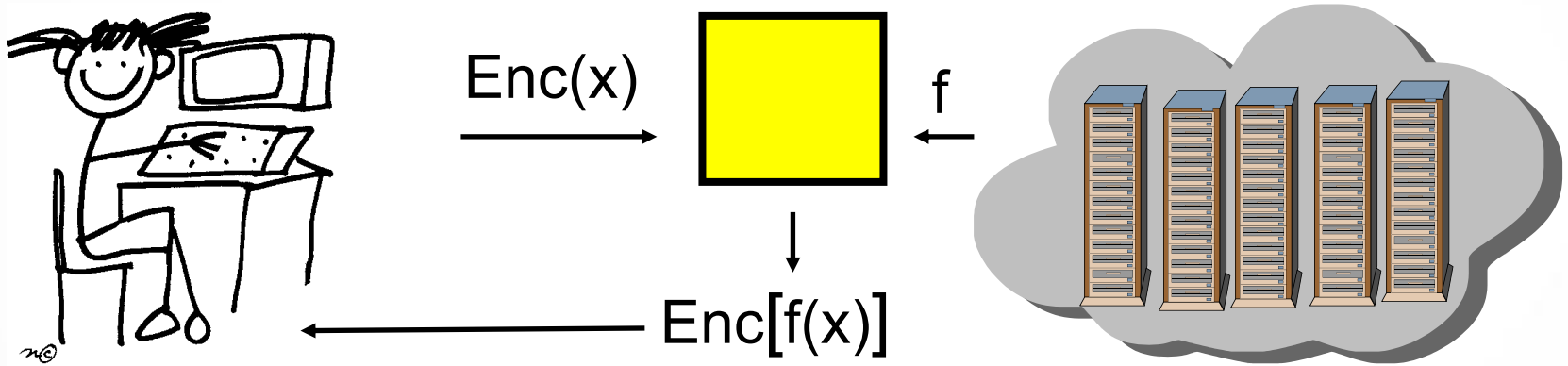
- We all have our individual secrets
  - We can compute a function of these secrets
  - Without revaling them to each other (or anyone else)

- Example: medical data
  - Evaluating the effectivness of a treatment

    ***f*(patient1Data, patient2Data, ...) = effective/not-effective**
  - Data for different patients held by different clinics
  - Can compute this with revealing any private data

Homomorphic Encryption

- Data can be processed in encrypted form
  - Result is also encrypted



- Available (in principle) for <10 years [Gen'09]

Homomorphic Encryption

- Data can be processed in encrypted form
  - Result is also encrypted

- Example: location services
  - I encrypt my location, send to yelp
  - Yelp compute an encrypted table lookup
    - T[cityBlock#] = ads for nearby coffee shops
  - I get back encrypted recommendation for coffee shops within two blocks

Fast Enough to be Useful

Performance of Advanced Cryptography

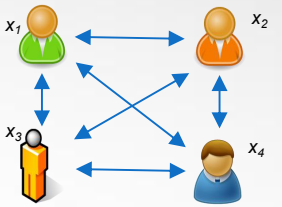
- Improving performence has been a major research topic over the last 30 years
  - Tremendous progress, many orders of magnitude

- For almost any realistic tasks, there is a cryptographic solution with adequate performance
  - Although designing it may take a team of experts

Speed Examples: Zero-Knowledge

- Digital currencies (zCoin, zCash, ...):
  - Proving that I have sufficiently many unspent coins on the ledger
  - Constructing proof in ~1min, verification in a few msec
- Anonymous credentials (idemix)
  - Proving that I possess a credential, takes 1-30 seconds
- Many other uses
  - Private payments in the Brave browser (using Anonize)
  - ...

Speed Examples: Secure MPC



- Biometric authentication in ~100ms
  - f(db, newSample) = match/noMatch
- Private set intersection in 1-30 seconds
  - Intersection of two sets with thousands of members
- Finding similar patients in a database in ~30sec
  - Best 5 (approx) matches against 4000 patients, 1000 markers
- "Generic computation": 1M gates in 1sec-5min
  - Depending on # of parties, LAN / WAN, adversary model

Speed Examples: HE

- Similarity of encrypted feature vectors
  - Computing similarity of two 1M-marker sequences in minutes
- Inference of simple NNets on encrypted data
  - Amortized 1000 perditions/minute on MNIST optical characters
- Computing a logistic-regression model on genome data
  - Under 10 minutes with 10-15 columns, ~1000 rows

Not “Generally Usable" Yet

Advanced Crypto Implementations

- Many software libraries for ZKP / MPC / FHE
  - Most of them open-source, a few proprietary
- Very hard to compare them, decide which technology/implementation to use for what purpose
  - Different tools, data & computation models, performance profiles, security guarantees, ...
  - Hardly any accepted benchmarks
- Most libraries are written for speed, not usability

Complexity of Advanced Cryptography

- Distributed computing is already complex
- “Advanced cryptography” also needs oblivious computation
  - Usually cannot do

    ```
    if (secretCondition) then do X else do Y
    ```

  - Instead must do both `X,Y`, then `MUX(X,Y;secretCondition)`
- Good performence requires extreme optimizations
  - Straightforward implementation will be exceedingly slow
- Tension between simplicity/usability and performance

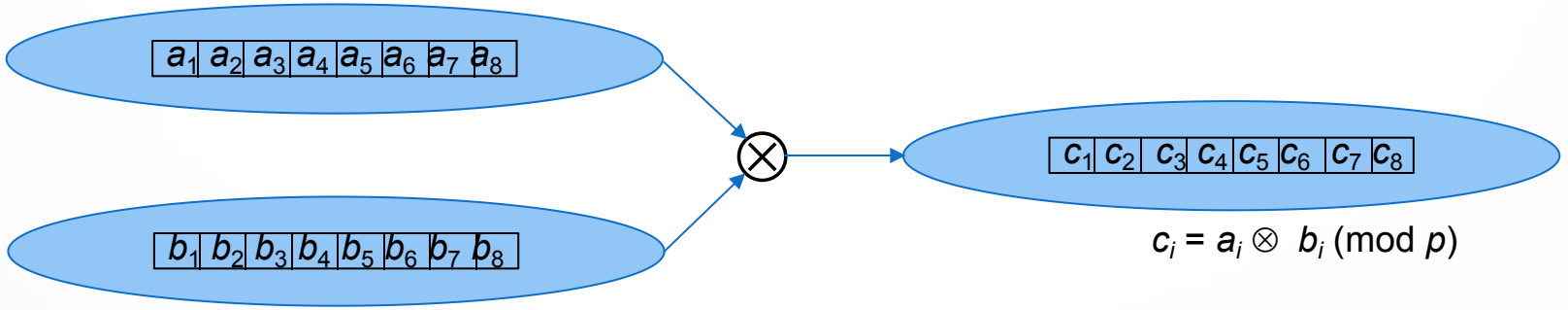
Example: Communication in Secure-MPC

- Communication between parties is a bottleneck in many protocols for secure multi-party computation
  - To optimize, many MPC libraries work with sockets
    - The library expects to be “in charge” of IP-address:port
- What if my system has its own communication layer?
  - E.g. working over https, gRPC, …
- Retrofitting existing libraries to use “abstract channels” is a lot of work, may degrade performance

Example: Data Encoding for FHE

- Ciphertext operations in contemporary FHE is slow

- "Ciphertext packing" to gain in performance
  - Each ciphertext encrypts a vector of plaintext element
  - Ciphertext operations effect element-wise operations



- Vector-size is a parameter, depends on the algebra

Example: Data Encoding for FHE (2)

- Lots of flexibility in setting the parameters
  - Determine plaintext modulus, vector-size, more
  - Choosing the right parameters is an art form
- Even with parameters set, where to put each piece of data requires a careful design
  - Could get orders-of-magnitude performance difference between different packing schemes
- No tool support for making any of these choices

Taming the Complexity

- How to make advanced cryptography usable to non-expert programmers?
- Frameworks, compiler support
  - Initial research work along these lines (Fabric, Obliv-C)
- Usable “toolboxes” for common tasks
  - Low level: arithmetic, sorting, linear algebra, …
  - Mid level: graphs algorithms, set intersection, ML tools, …
  - Domain specific tasks (medical, financial, …)
- We must shift focus from speed to usability

Summary: Advanced Cryptography is

Needed

Fast enough to be useful

Not "generally usable" yet

We need engagement of cryptographers and system builders

Summary: Advanced Cryptography is

Needed

Fast enough to be useful

Not "generally usable" yet

Questions?

We need engagement of cryptographers and system builders