香港城市大學
City University of Hong Kong

Privacy-enhancing Technologies

Prof. Cong Wang
Department of Computer Science
CityUHK

Slides credit in part from A. Juels, D, Song, A. Miller, E. Ben-Sasson, D. Dziembowski, A. Judmayer, F. Zhang, I. Eyal, J. Poon, F. Greenspan, and S. Halevi

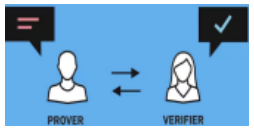
What's "Advanced Cryptography"?

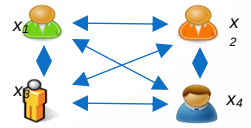
- Cryptography beyond encryption, signatures
  - **Protecting computation, not just data**

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 (HE) 

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 (HE)

Not in this talk:

- Searchable Encryption
- 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

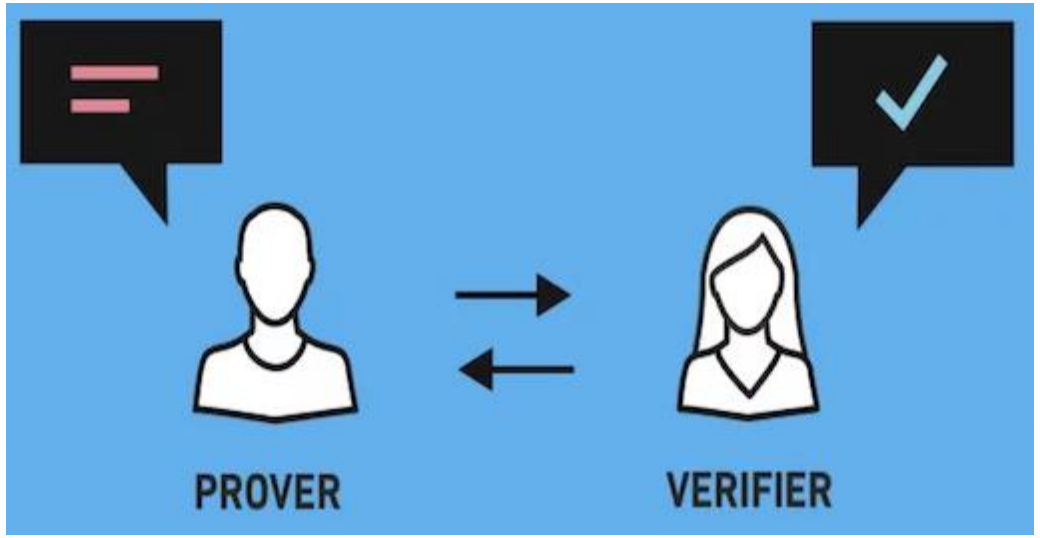
Not "generally usable" yet

Advanced Crypto Tools

- Zero-Knowledge (ZK)
- Secure Multi-Party Computation (MPC)
- Homomorphic Encryption (HE)

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

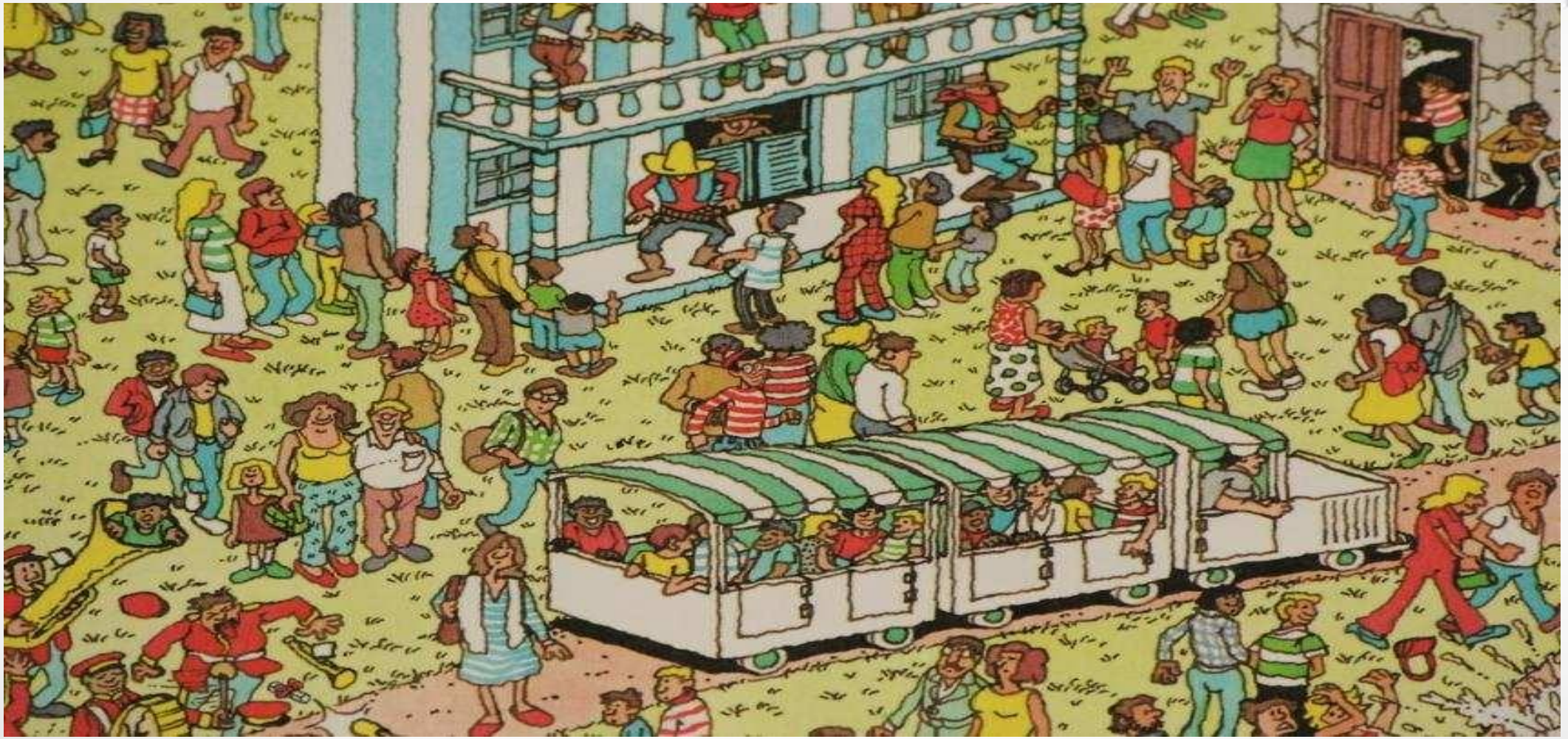
- Example: my secret is my purchase history
  - I can prove to Hood that I bought
    10 gallons of milk this month
    - so I can get a coupon
  - Without revealing anything else

Example of ZK: Where is Charlie?

Goal:

- find the reporter Charlie in a big picture,
- convince the verifier (me) that you have the solution without revealing it (neither to me, nor to the others).

Where is Charlie?



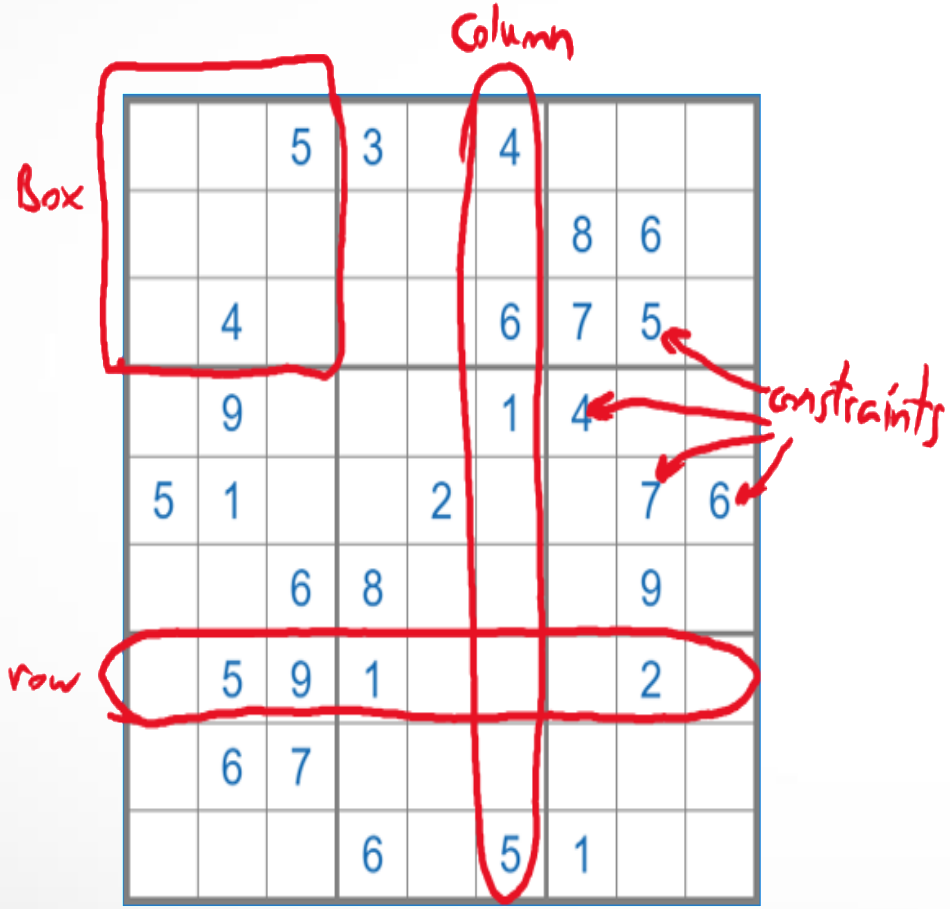
Where is Charlie?

How can you prove that you know where is Charlie without saying nothing about where he is?

Solutions:

1. get a copy of the picture, cut out Charlie and show it to me.
2. put a big mask with a window having the shape of Charlie and show me Charlie through the window.

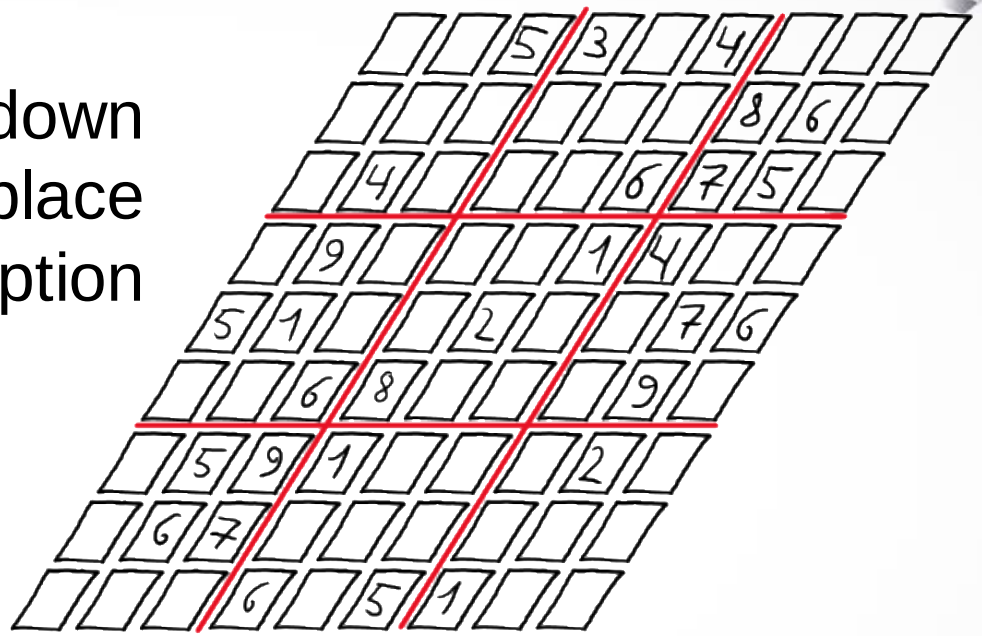
A more useful example



- Alice want to prove to Bob he knows the solution;
- Alice does not want to show the solution directly to Bob such that Bob can show it to Charlie.

Interactive proof

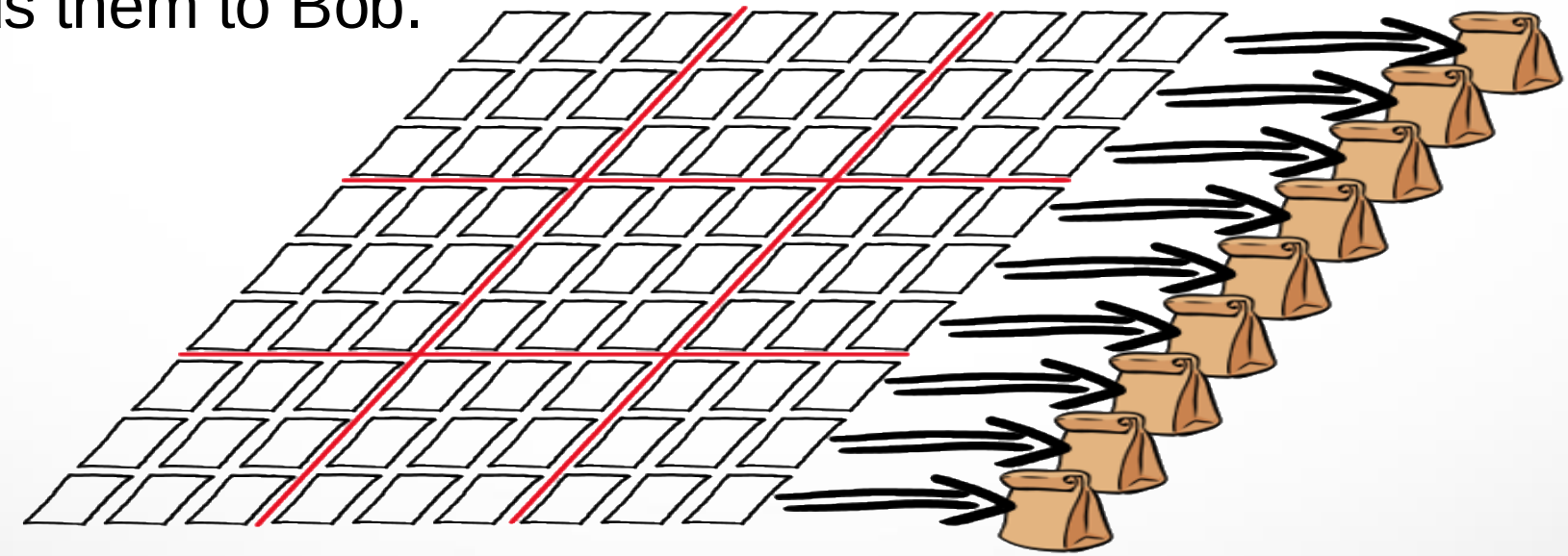
- The commitment: Alice writes down the solution on each card and place them face down, with the exception of the constraints



- The challenge: Bob can choose whether he wants to check the rows, the columns, or the blocks. Eventually he picks one of these three conditions at random.

Interactive proof

- Bob decides to choose the row
- Prove: Alice proceeds to place the cards from each row inside an opaque bag—one bag per row. She gives each bag a thorough shake, making sure the index cards inside are mixed well and hands them to Bob.



Interactive proof

- Verify: Bob opens each bag and verifies that they should each have exactly 9 cards with all the numbers 1 through 9

- Bob says this proves nothing and he can also do it by placing the numbers 1 through 9 in each row in any order he'd like
- Alice explains that she couldn't have known in advance that Bob would pick the rows. She's not a mind reader.

Making the proof more convincing

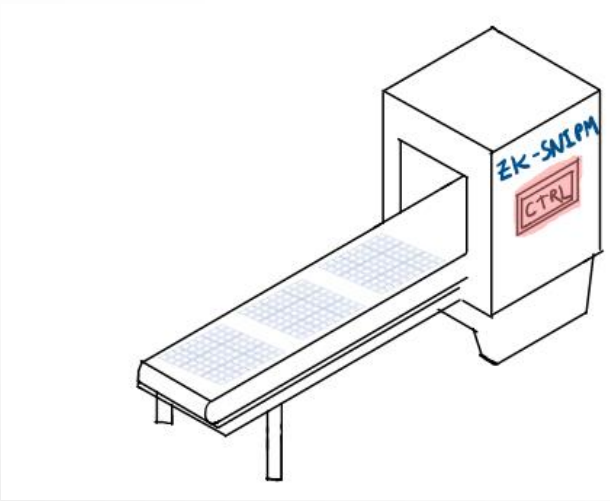
- Bob thinks that there is still a 2 in 3 chance for Alice to cheat. How can Alice assure Bob with higher probability?

1. Rinse and repeat: Alice repeats the procedure with Bob—each time placing the same cards for the same Sudoku problem face down, but letting Bob pick a different test at random.

2. After a long series of tests, Bob is forced to admit that Alice is either an extremely lucky person, or, that she simply has a solution to the Sudoku problem (or perhaps she could read his thoughts after all).

Non-interactive proof

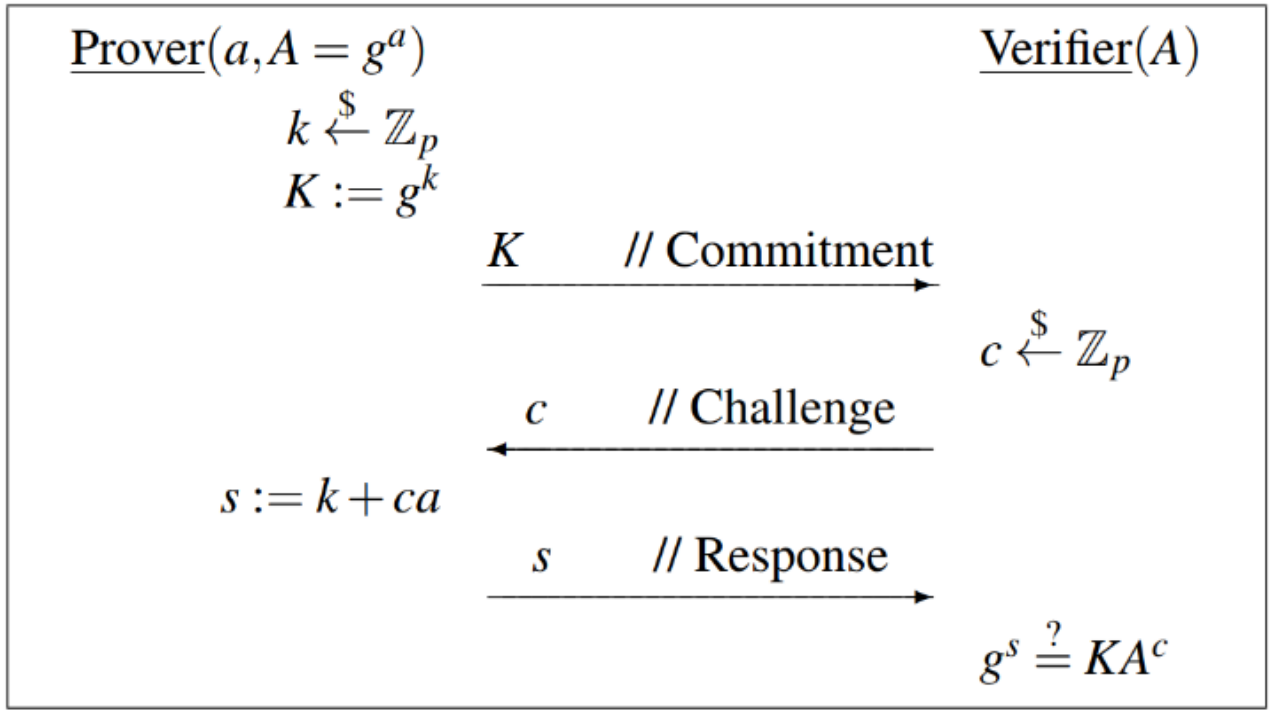
- How about Alice collude with Bob?
  - Imagine that they start a Youtube channel live streaming the Sudoku solving. How can they prove to the audience that they are not cheating?
- A cryptographer Dave presents a incredible new invention : "The Zero-Knowledge Sudoku Non-Interactive Proof Machine" (or zk-SNIPM as he like to refer to it).



- ✓ The Machine was basically an automated version of Alice's test.
- ✓ Alice, Bob, Dave can set up the secret series of tests with their inputs while nobody could fully control the final result.

ZKPs: A Concrete Example

Schnorr protocol for the proof scheme ZKPoK{(a) : $g^a = A$}, where *g* is the generator of a group *G* of prime order *p*



Advanced Crypto Tools

- Zero-Knowledge (ZK)
- Secure Multi-Party Computation (MPC)
- Homomorphic Encryption (HE)

Yao's Millionaires Problem

Two millionaires want to know who is richer on between them

However, they are not willing to reveal the amount of their wealth

Average Salary Problem

Alice

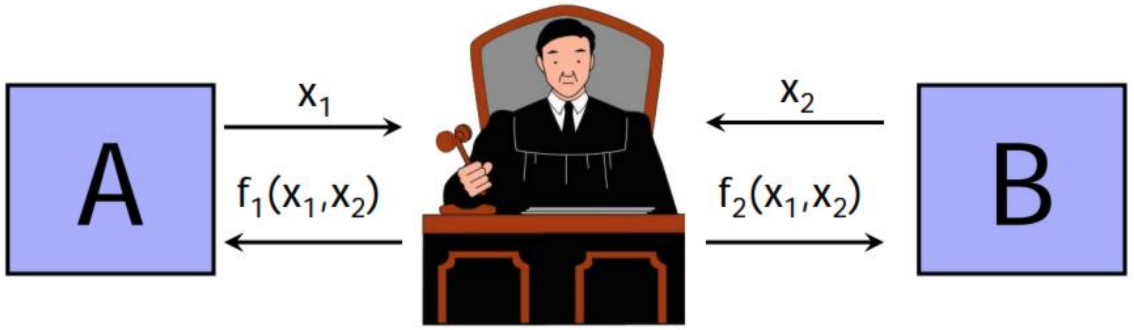
Bob

Charlie

A group of cryptographers want to compute the average of their salaries,
However, they also do not want to reveal the amount of their own salaries!

An Ideal Situation

- What would the ideal situation be?
  - A trusted and incorruptible third party
  - All parties send inputs to trusted party
  - Trusted party computes and sends output



  - **However, what if we do not have such an ideal, trusted party?**

Let's Play A Dating Game…

A guy and a girl want to check if they are both interested in going out

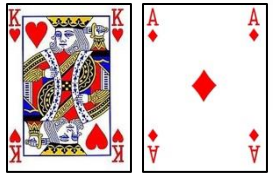
- If they both are, then output is YES
- If at least one is not, then output is NO

YES or NO?

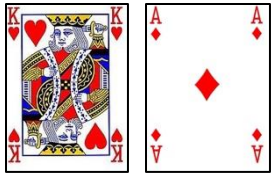
- If Alice says YES and Bob says NO, then the result is NO and Bob doesn't know if Alice said YES or not
- Alice doesn't lose face…

Example: Dating Game with Cards

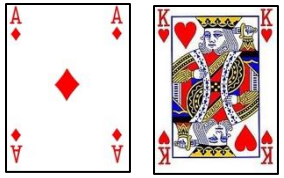
- Alice and Bob each get two cards



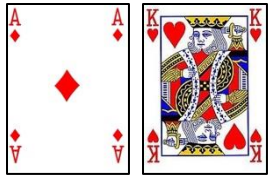
- If Alice likes Bob:



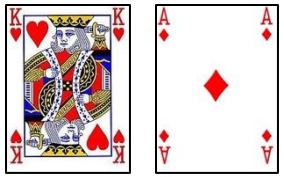
and if not:



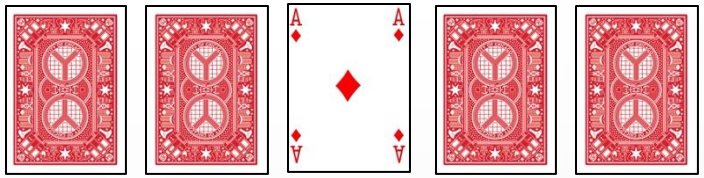
- If Bob likes Alice:



and if not:



- Each turns their cards over, with an Ace in the middle

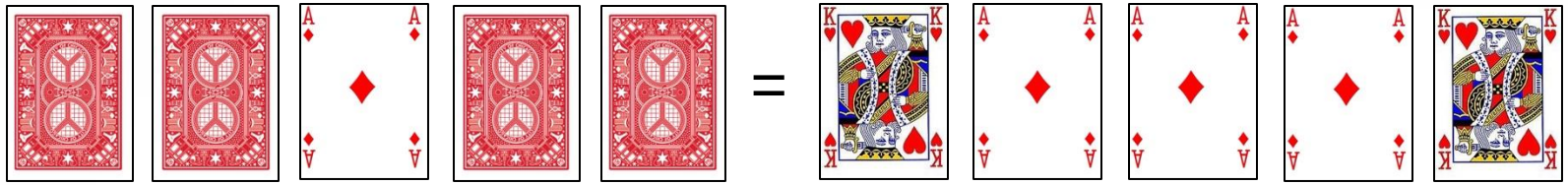


Alice's cards

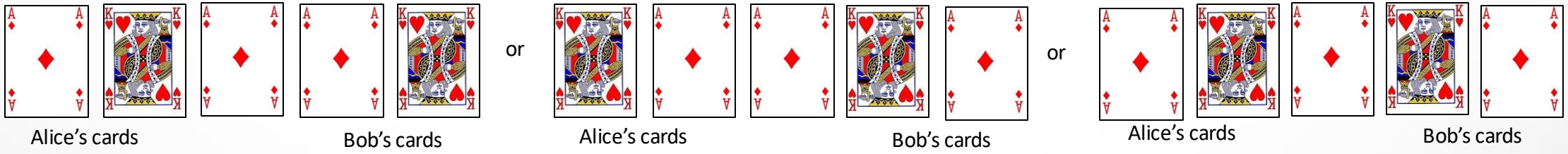
Bob's cards

Example: Dating Game with Cards

- If Alice and Bob like each other



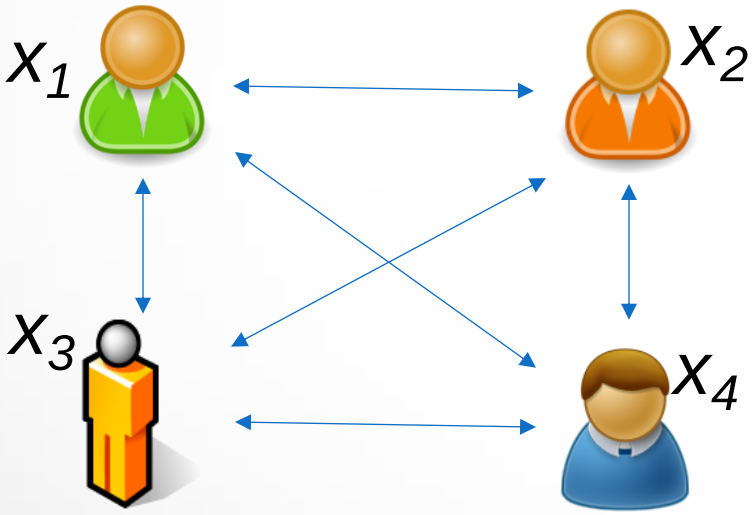
- Otherwise,



- Parties turn over middle card and randomly rotate
- If three Aces in a row then YES; else NO

Secure Multi-Party Computation

- We all have our individual secrets
  - We can compute a function of these secrets
  - Without revealing 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

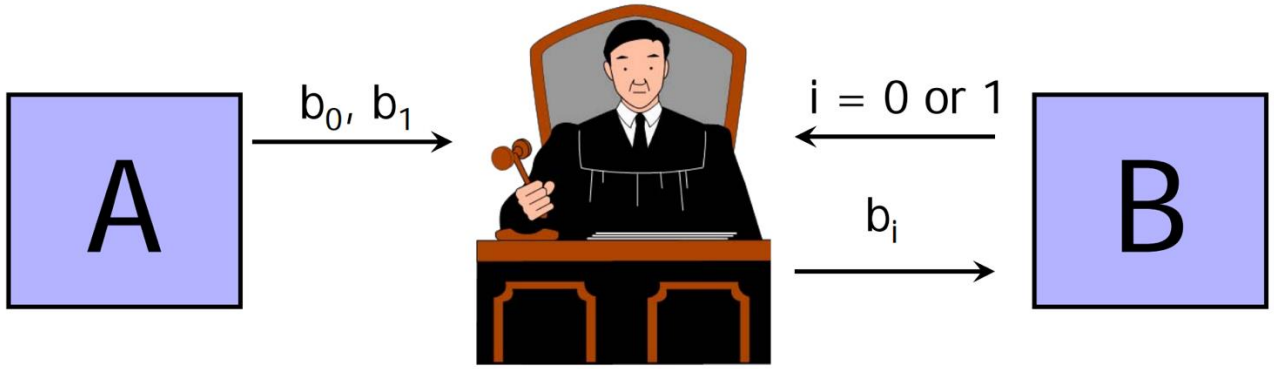
- Available (in principle) since the 80's [Yao'86, GMW'86]

Adversary Model

- Some of protocol participants may be corrupt
  - If all were honest, would not need MPC
- Semi-honest (aka passive; honest-but-curious)
  - Follows protocol, but tries to learn more from received messages than expected.
- Malicious (aka active)
  - Deviates from the protocol in arbitrary ways, lies about his inputs, may quit at any point
- For now, we focus on semi-honest two-party protocols

Oblivious Transfer [Rabin'81]

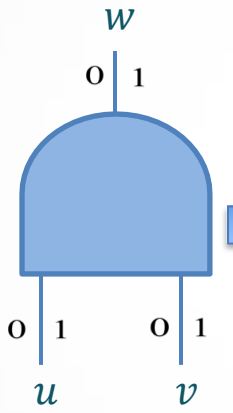
- Fundamental MPC primitive



- A inputs two bits, B inputs the index of one of A's bits
- B learns his chosen bit, A learns nothing
  - A does not learn which bit B has chosen;
  - B does not learn the value of the bit that he did not choose.

Yao's Garbled Circuit [Yao'86]

- Garbling a single Boolean gate



u	v	$w = u \wedge v$
0	0	0
0	1	0
1	0	0
1	1	1

u	v	$w = u \wedge v$
k_u^0	k_v^0	k_w^0
k_u^0	k_v^1	k_w^0
k_u^1	k_v^0	k_w^0
k_u^1	k_v^1	k_w^1

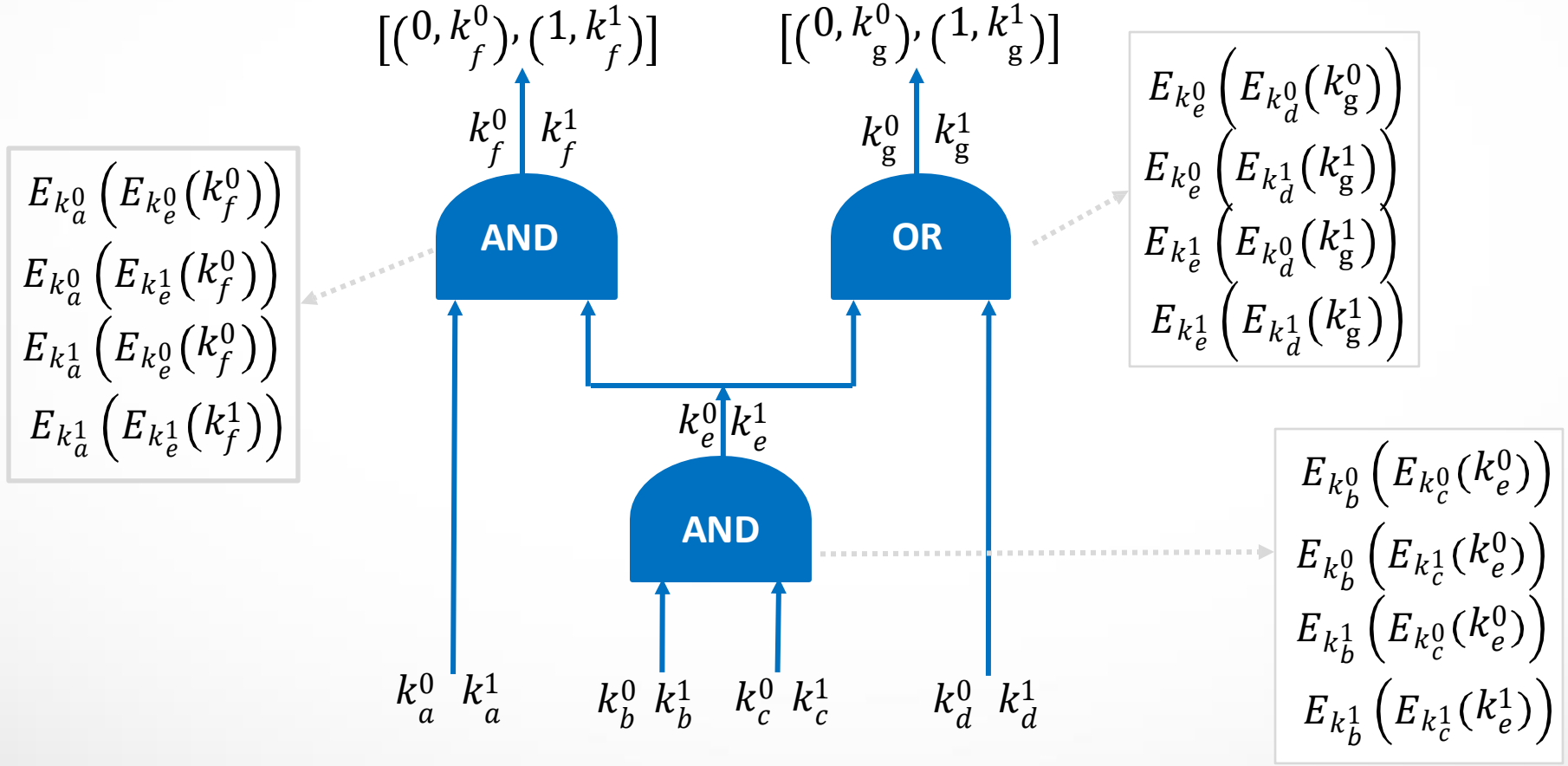
$$E_{k_u^0}\left(E_{k_v^0}(k_w^0)\right)$$
$$E_{k_u^0}\left(E_{k_v^1}(k_w^0)\right)$$
$$E_{k_u^1}\left(E_{k_v^0}(k_w^0)\right)$$
$$E_{k_u^1}\left(E_{k_v^1}(k_w^1)\right)$$

In random order

- Given one key on each input wire, compute on the output wire, learning nothing about the input values.
- Keys on input wires are called garbled inputs

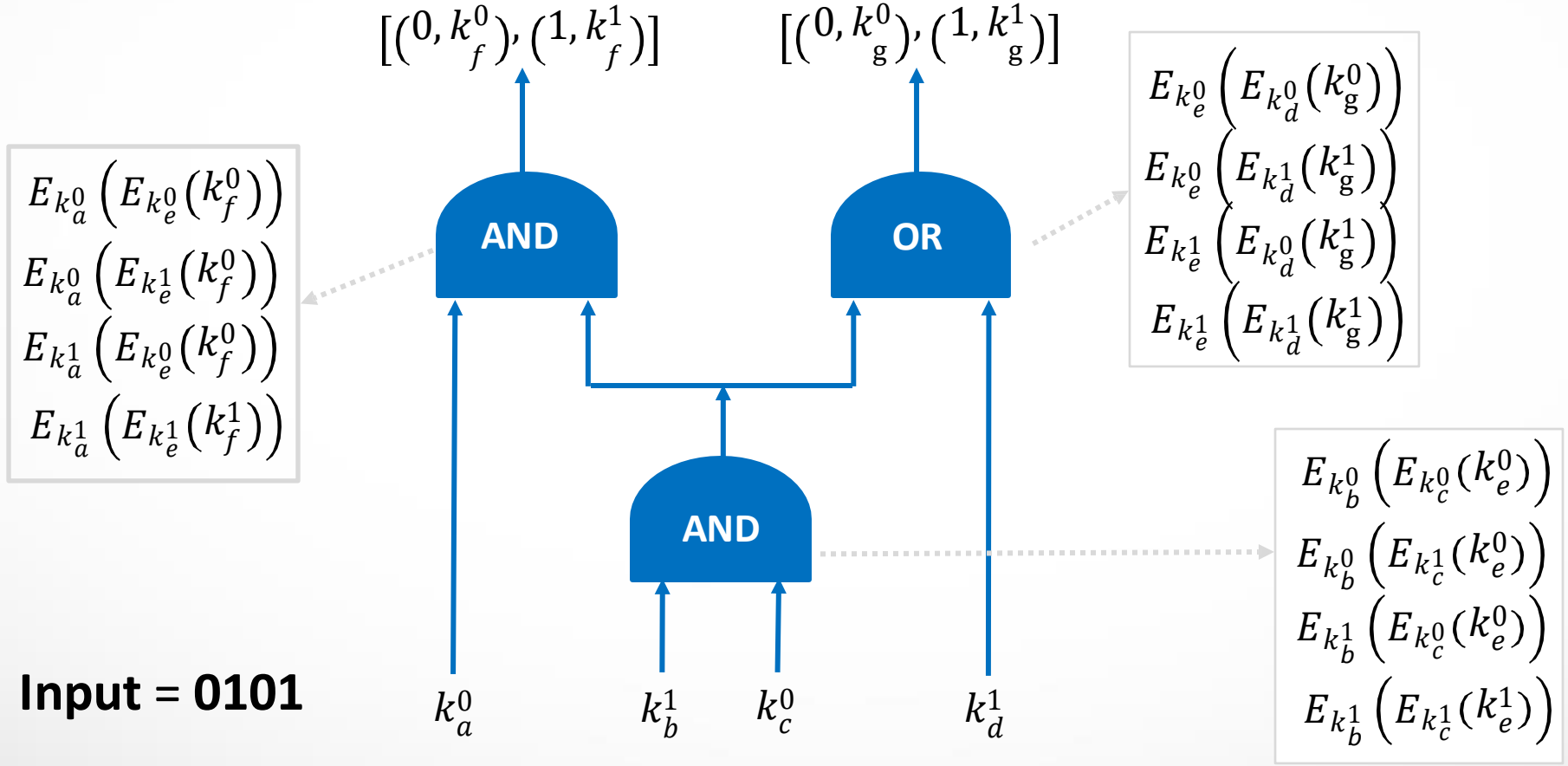
Yao's Garbled Circuit [Yao'86]

- Garbling an entire circuit



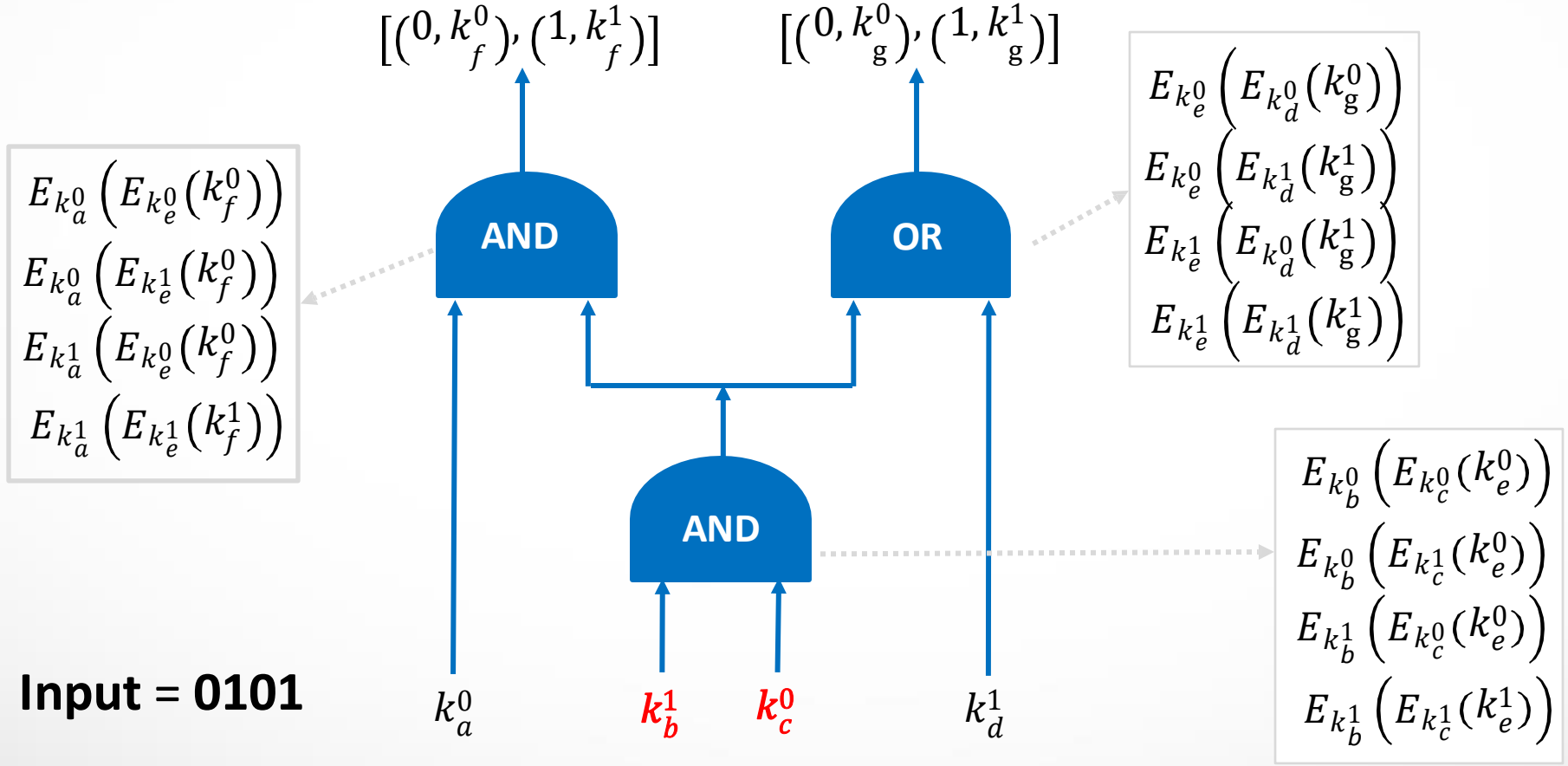
Yao's Garbled Circuit [Yao'86]

- Garbling an entire circuit



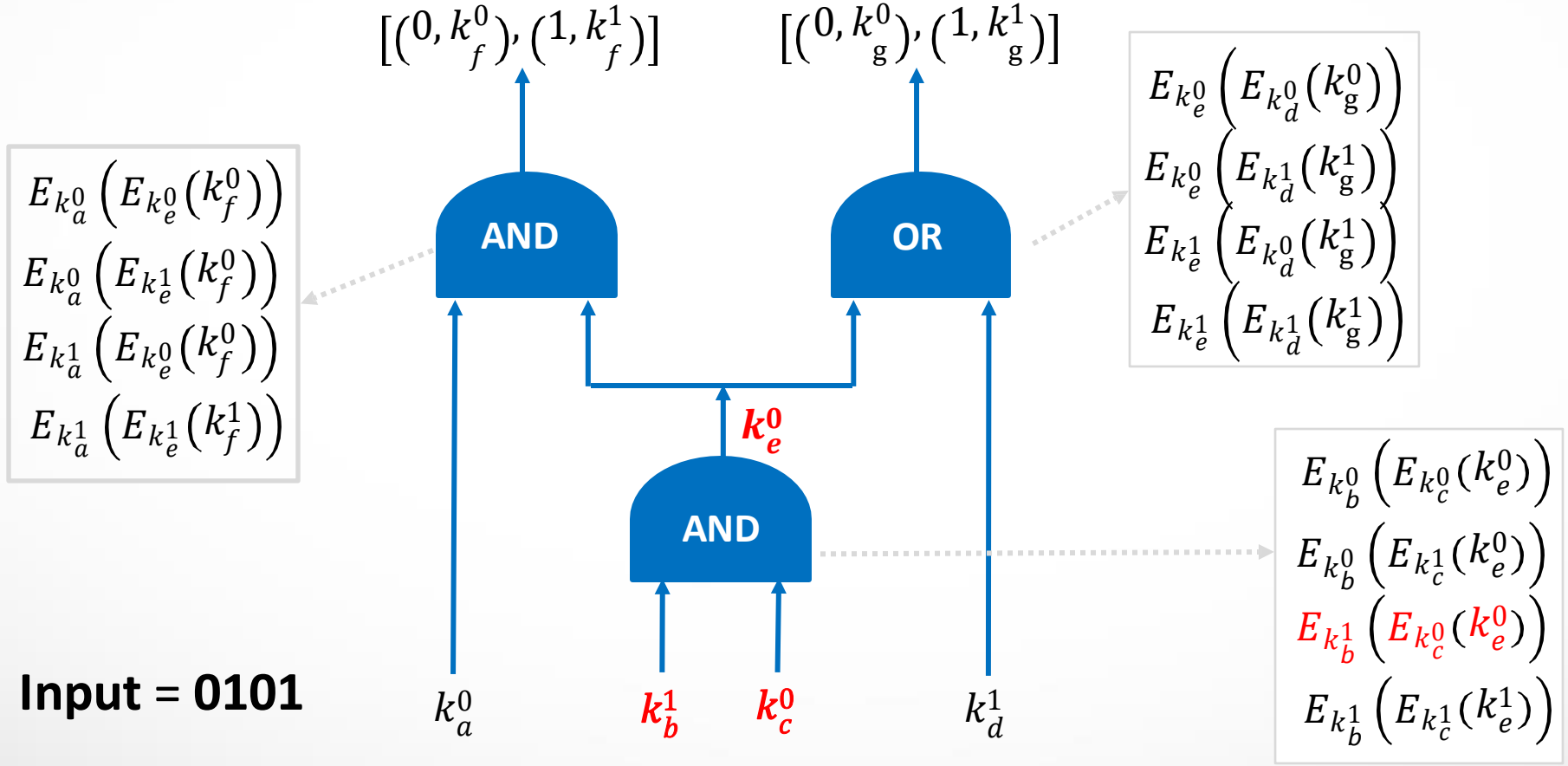
Yao's Garbled Circuit [Yao'86]

- Evaluating an entire circuit (with OT)



Yao's Garbled Circuit [Yao'86]

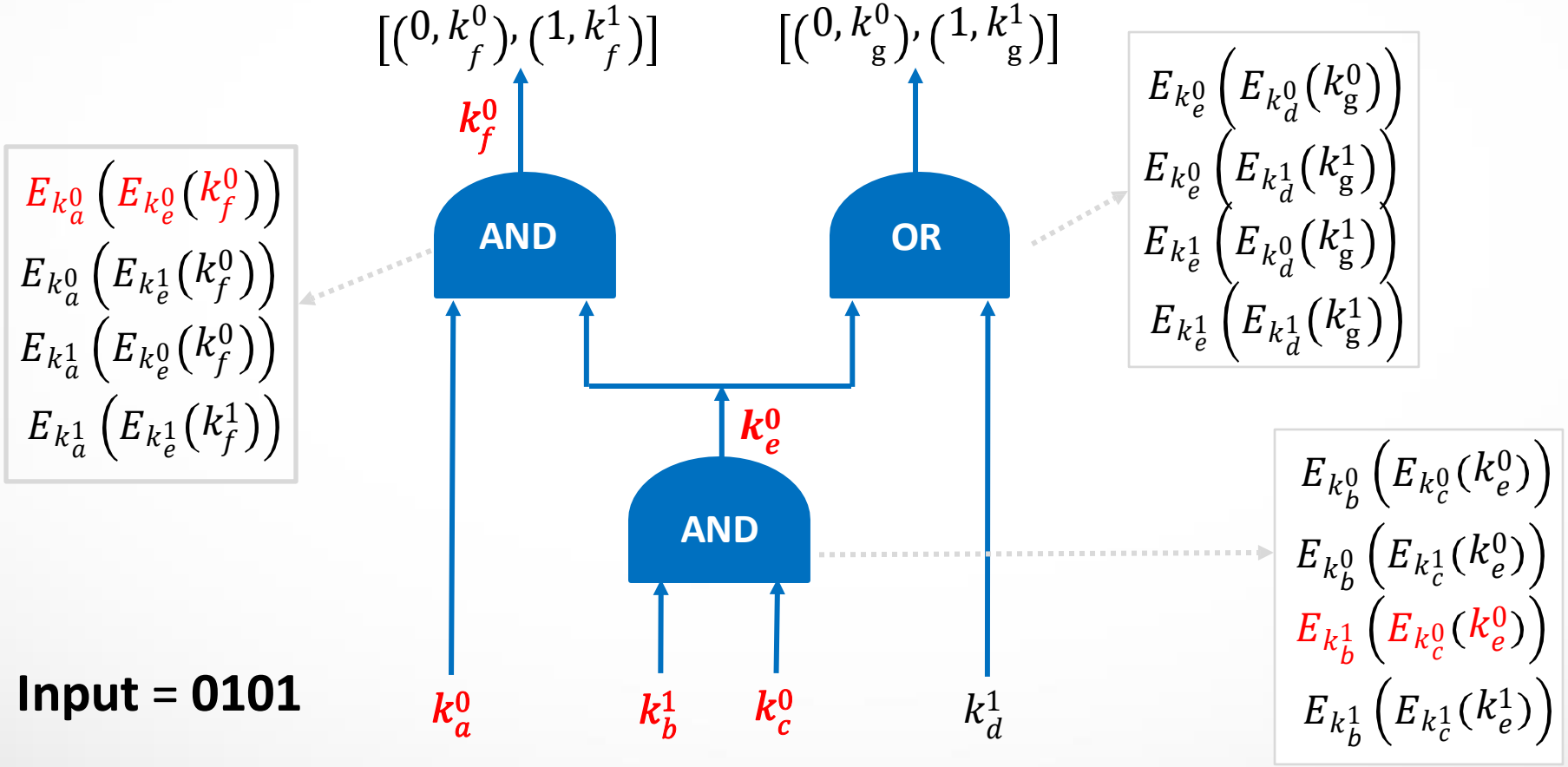
- Evaluating an entire circuit (with OT)



Input = 0101

Yao's Garbled Circuit [Yao'86]

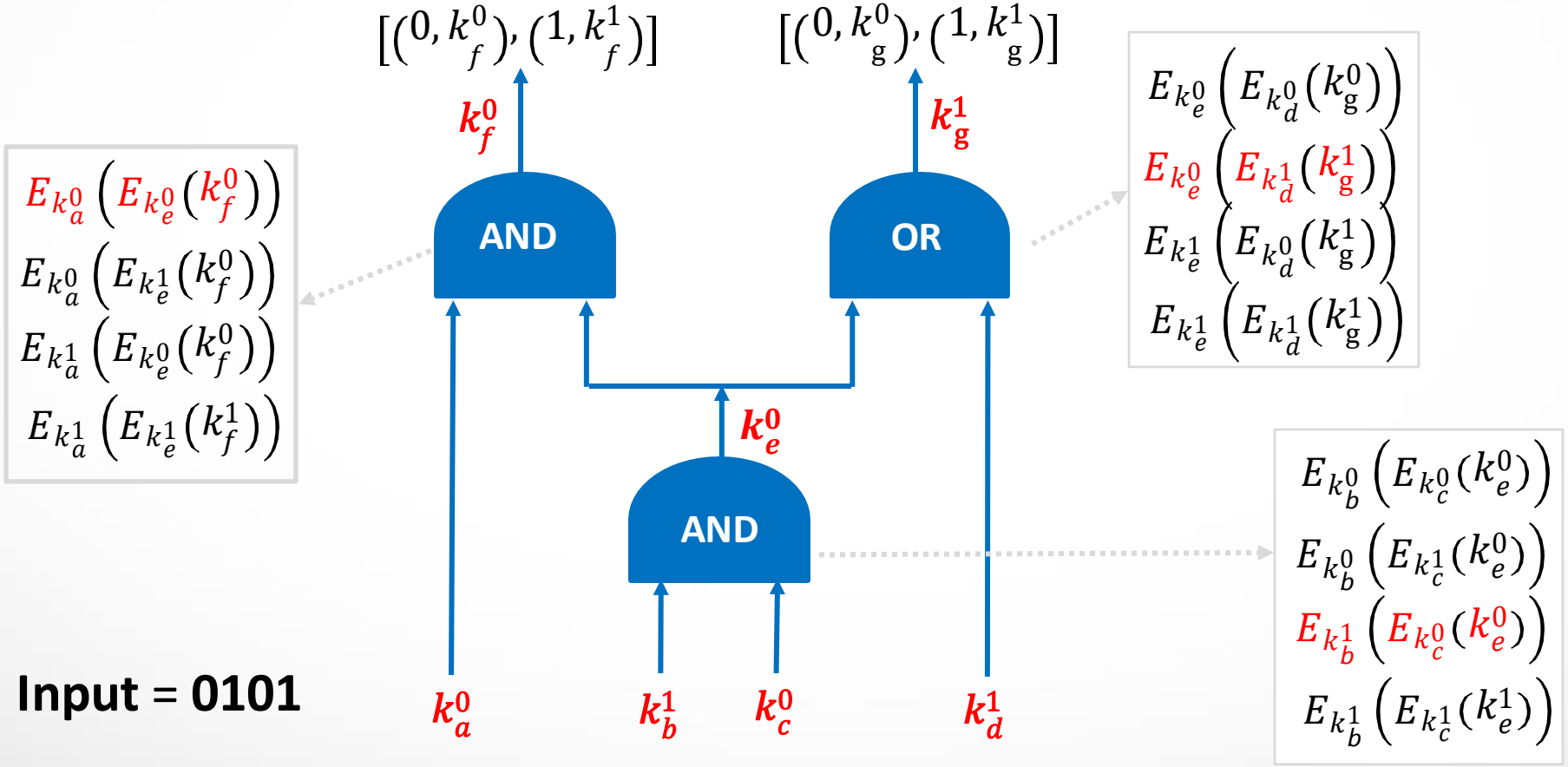
- Evaluating an entire circuit (with OT)



Input = 0101

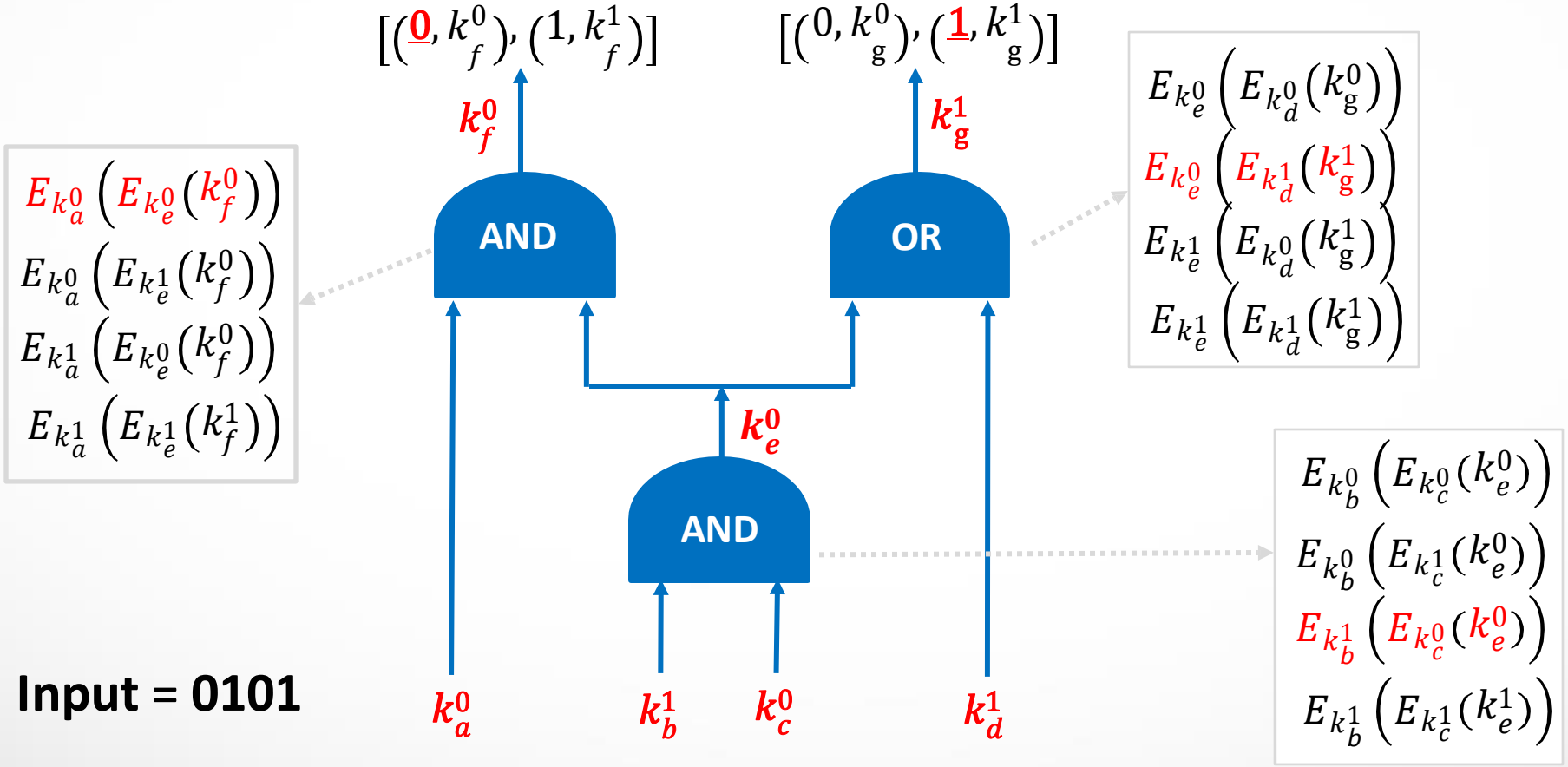
Yao's Garbled Circuit [Yao'86]

- Evaluating an entire circuit (with OT)



Yao's Garbled Circuit [Yao'86]

- Evaluating an entire circuit (with OT)



A Standard Case

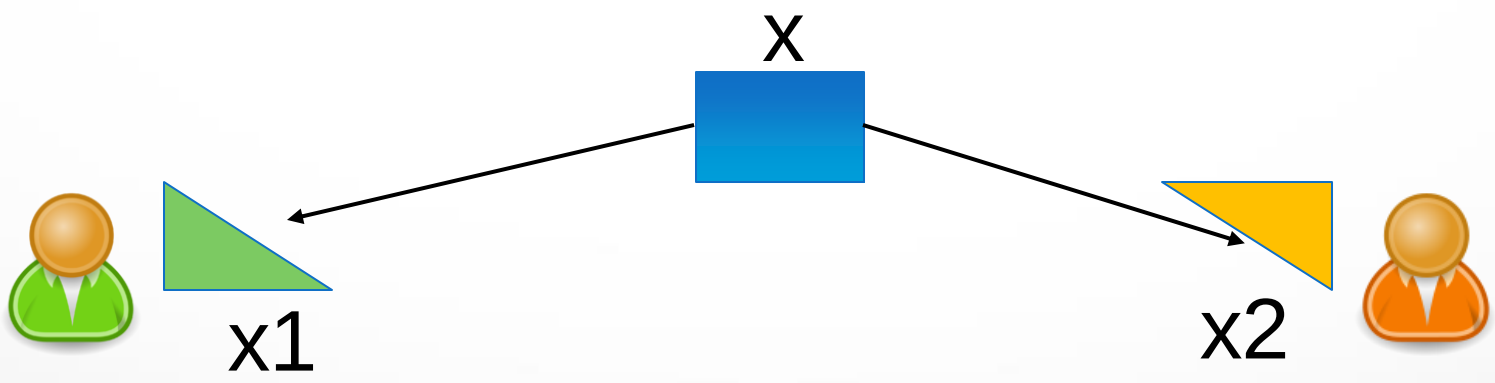
P1: Contructing garbled circuit

OT

P2: Evaluating garbled circuit

Secret Sharing

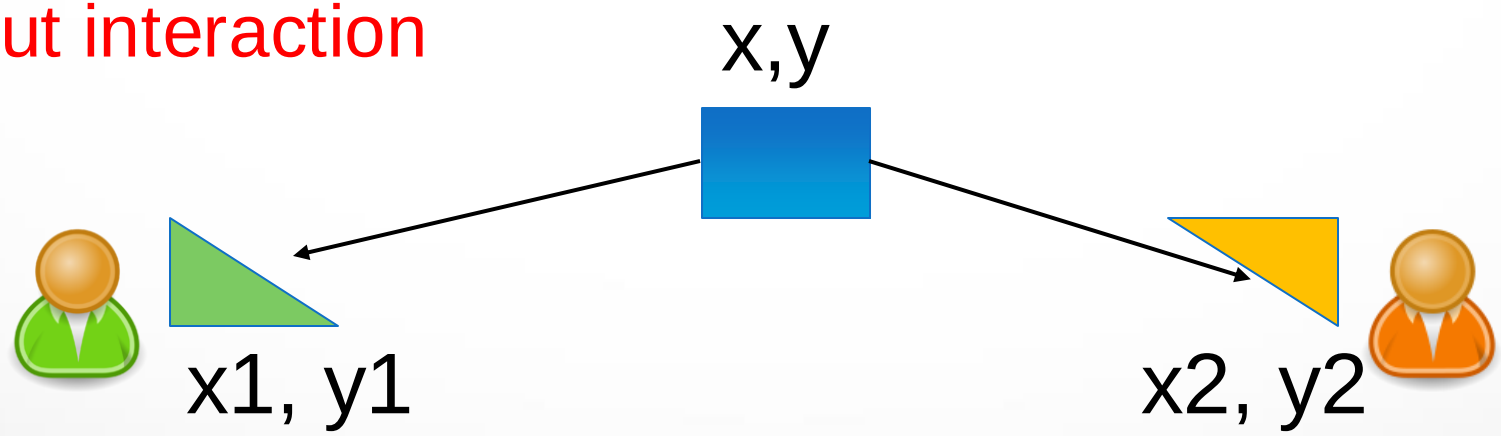
- Another fundamental MPC primitive
- Suppose an additive sharing of input x held by two parties.
  - P1 holds x1 and P2 holds x2.
  - x = x1+x2
  - None of them know the value of secret x



Compute with Secret Sharing

- Addition of the values of x and y
  - P1 computes z1 = x1+y1
  - P2 computes z2 = x2+y2

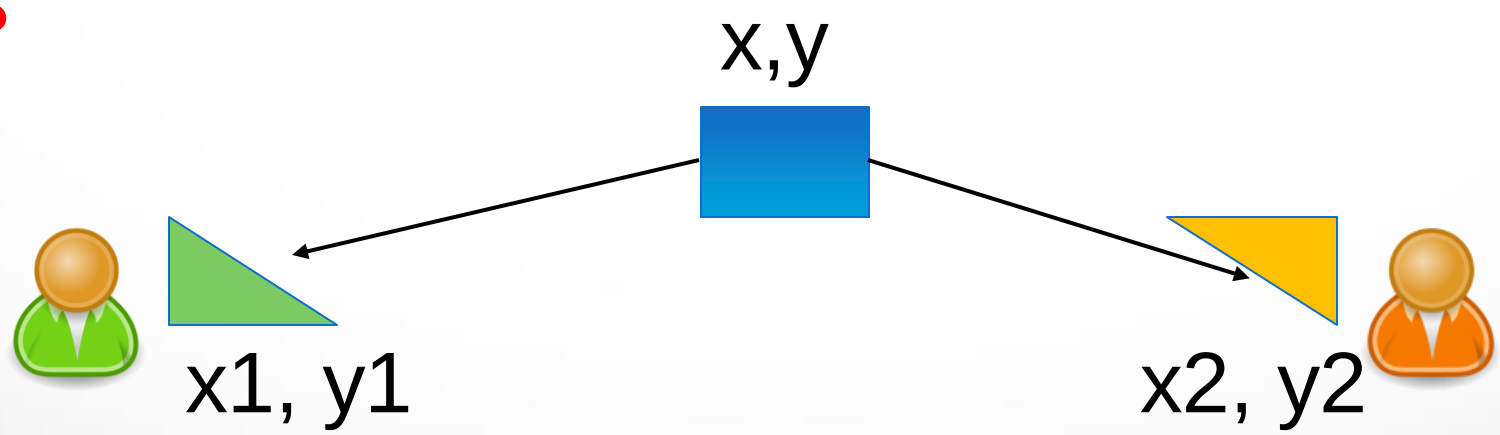
- z = x + y = (x1+x2) + (y1+y2) = (x1+y1) + (x2+y2) = z1+z2
  - Without interaction



Compute with Secret Sharing

- Multiplication of the values of x and y
  - If P1 computes z1 = x1*y1
  - If P2 computes z2 = x2*y2

- **x*y ≠ z1+z2**
  - **How?**



Beaver Triple [Beaver'91]

- A Beaver triple is a tuple of (a,b,c), where c=a*b.
  - P1 obtains a1,b1,c1.
  - P2 obtains a2,b2,c2.
  - c = a*b → c1+c2 = (a1+a2)*(b1+b2).

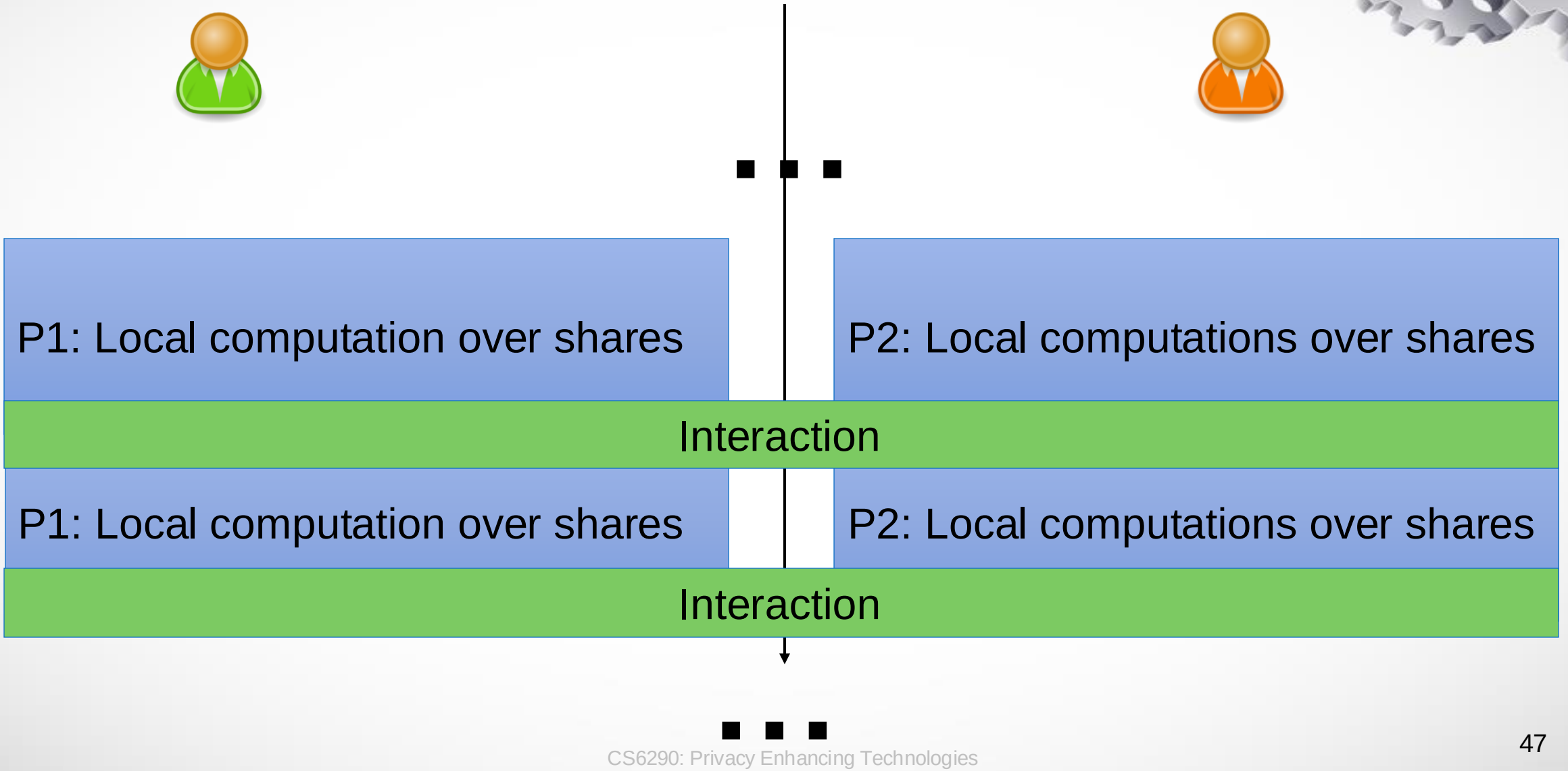
Beaver Triple [Beaver'91]

- A Beaver triple is a special tuple of (a,b,c).
  - c = a*b → c1+c2 = (a1+a2)*(b1+b2).
  - P1 obtains a1,b1,c1; P2 obtains a2,b2,c2.
- To securely compute x*y, we need a Beaver triple.
  - P1 computes a1+x1 = A1; P2 computes a2+x2=A2; they exchange A1, A2 to obtain A=A1+A2=x+a.
  - Similarly, they compute B = y+b.
  - P1 computes z1 = A*y1-B*a1+c1; P2 computes z2 = A*y2-B*a2+c2; they exchange to obtain z = z1+z2.
  - With interaction

Compute with Secret Sharing

- Arbitrary function
  - Taylor expansion: All function can be approximated via `add` and `mult`.
  - The intermediate results also exist as the additive share of inputs for the subsequent function evaluations.

A Standard Case

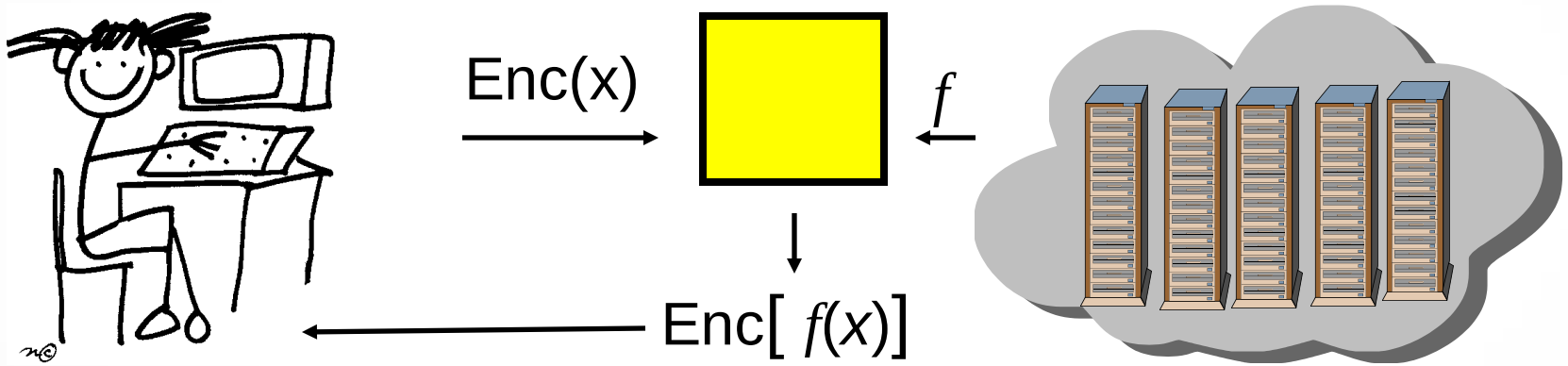


Advanced Crypto Tools

- Zero-Knowledge (ZK)
- Secure Multi-Party Computation (MPC)
- Homomorphic Encryption (HE)

Homomorphic Encryption

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



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

Homomorphic Encryption

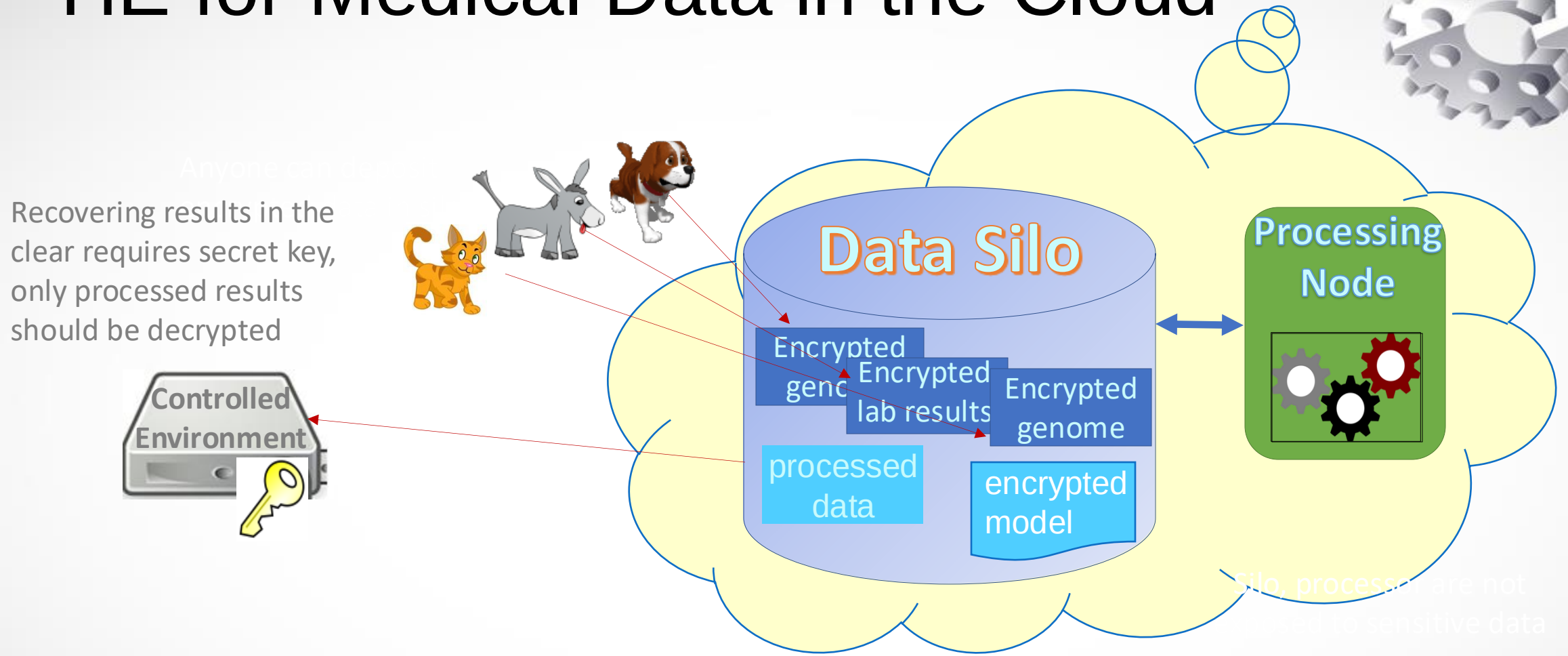
- Data can be processed in encrypted form
  - Result is also encrypted

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

Classification of Homomorphic Encryption

- Partially homomorphic encryption (PHE)
  - Support **limited types** of homomorphic operations
    - Additively homomorphic *or* multiplicatively homomorphic
    - E.g., Paillier cryptosystem (additively)[Paillier1999], ElGamal encryption (multiplicatively) [ElGamal1985]
  - Not quite expensive
- Full homomorphic encryption (FHE)
  - Support **full types** of homomorphic operations
  - e.g., Gentry's scheme [Gentry2009]
  - Highly expensive
- Somewhat homomorphic encryption (SHE)
  - Support **limited number** of full homomorphic operations
    - E.g., [BrakerskiV2011]
  - Cost between PHE and FHE

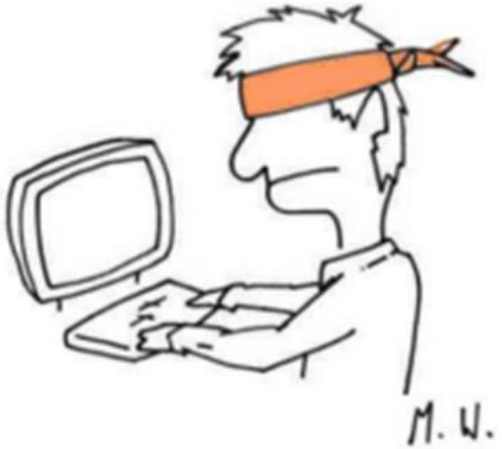
HE for Medical Data in the Cloud



- "Silos" of encrypted data, each controlled by a key
  - Lots of stored data, small parts of it are in process at any time

The Promise of Advanced Cryptography

Blindfold Computation



- The ability to process data without ever seeing it

The Need for Advanced Cryptography

Your Privacy for Sale

- We give up information in return for services
  - E.g., location for directions, restaurant recommendation, health data for "personalized medicine", financials for tax and investment services, purchace history for better ads and coupons, ...

Your Privacy for Sale

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

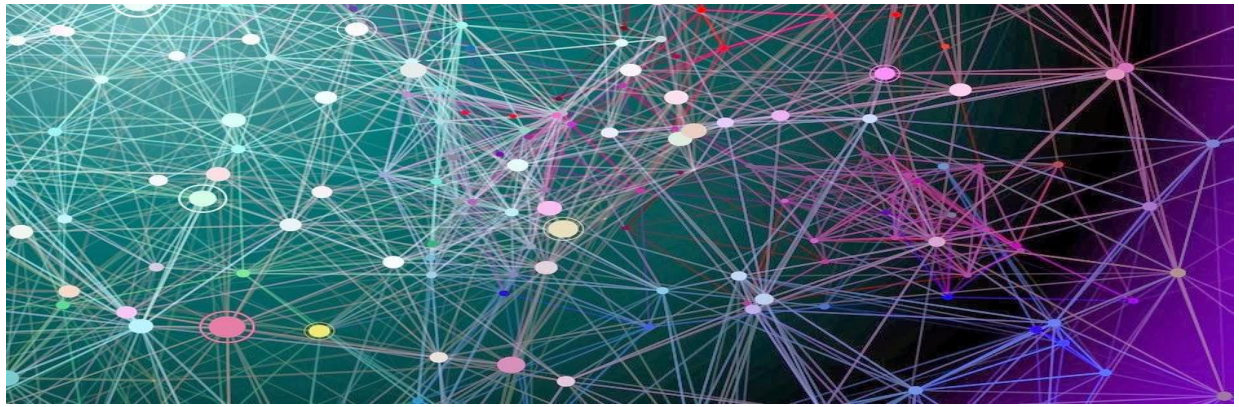
Your Privacy for Sale

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

- What happens once we give away this information?

Data Abuse is the New Normal

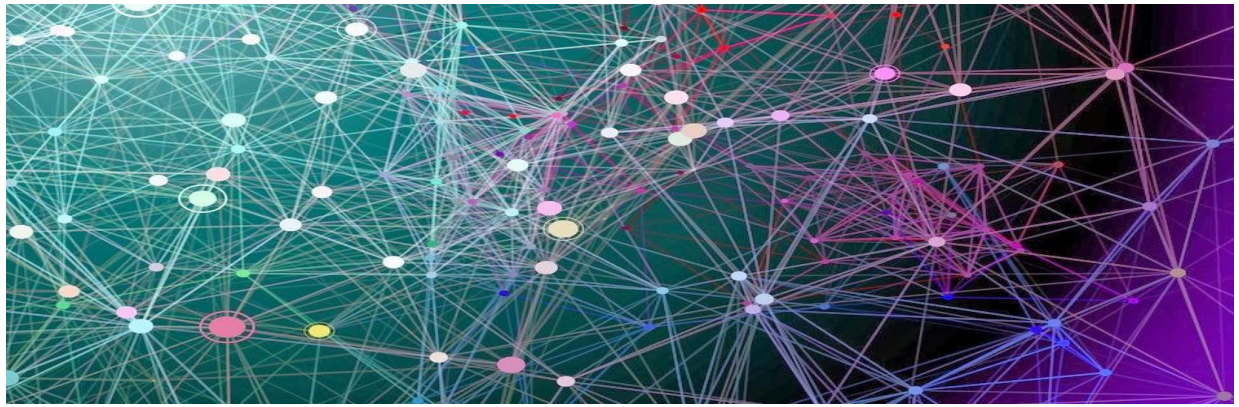
- The entire IT industry is busy making it easier
  - Larger collections, better ways to link, process them



- Data abuse, not “data breach”
  - Overwhelming motivation to use whatever data can be found
  - If the data is available, it will be (ab)used

Data Abuse is the New Normal

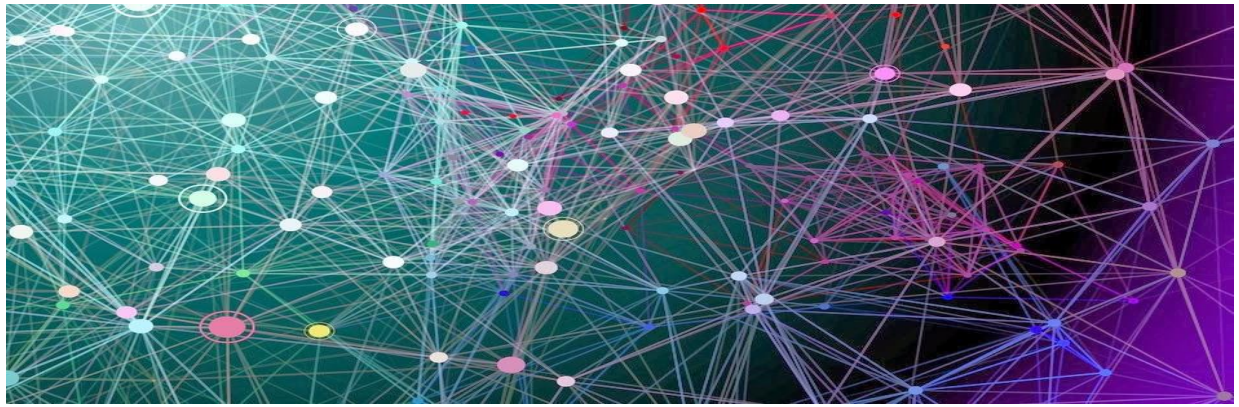
- The entire IT industry is busy making it easier
  - Larger collections, better ways to link, process them



- It will only get worse
  - We cannot provide opportunity for easy abuse, seriously expect it not to happen

Data Abuse is the New Normal

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

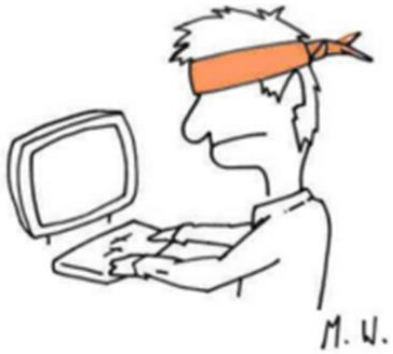


IT, security professionals

- We need all the tools we can get to push back
  - "Advanced Crypto" is an under-utilized tool in our box

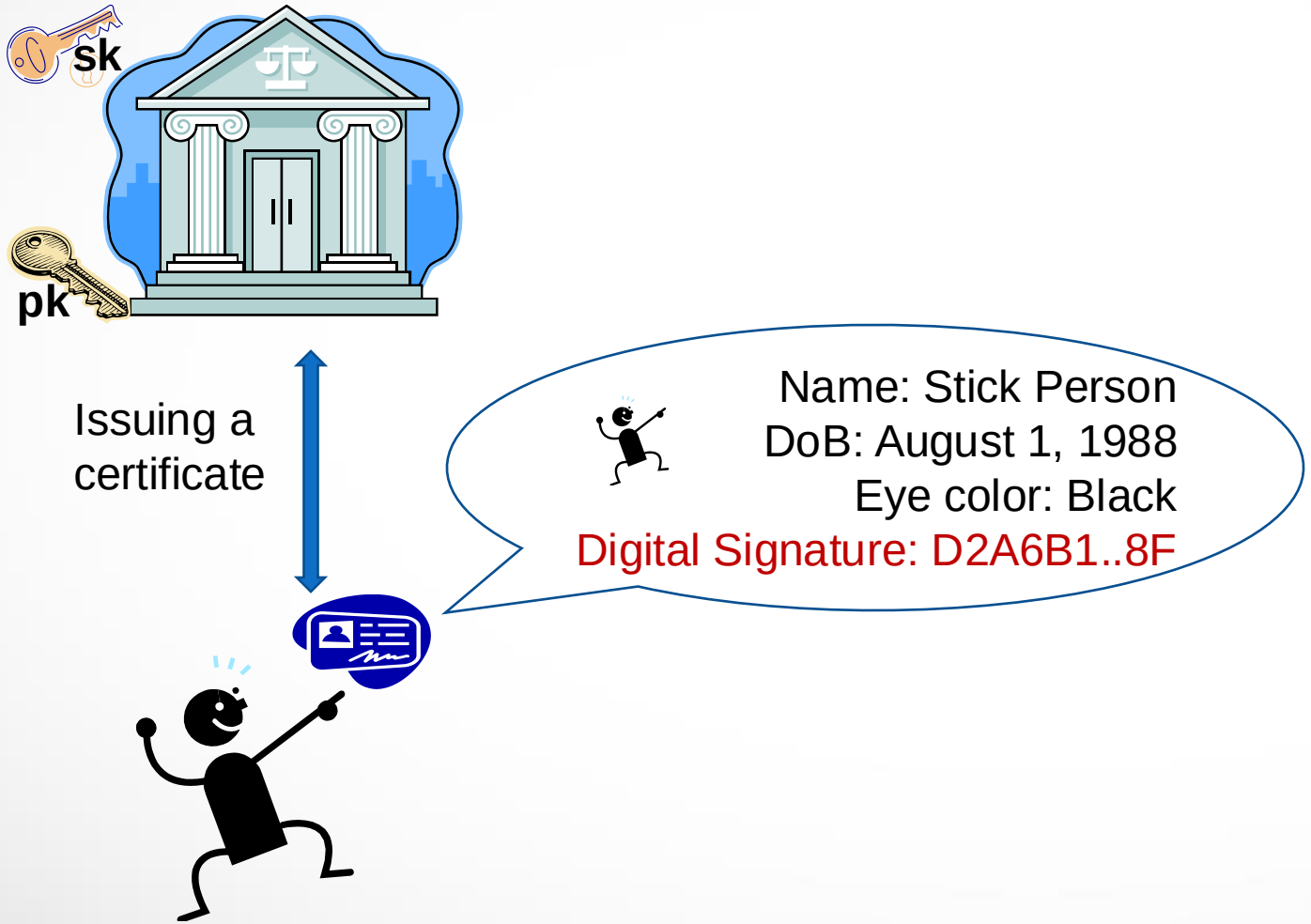
The Promise of Advanced Cryptography

Blindfold Computation

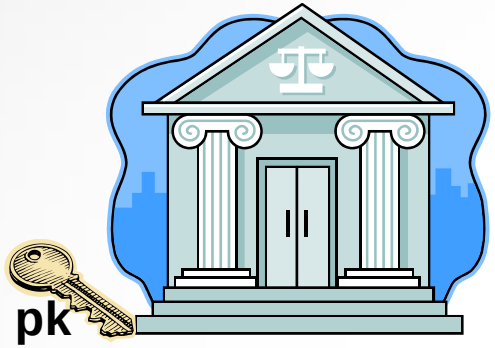


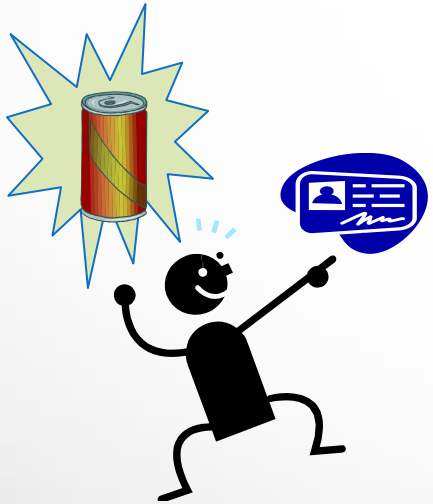
- The ability to process data without ever seeing it
  - Personalized services without access to private information
  - You cannot abuse data that's not there

Example: Anonymous Credentials using ZK



Example: Anonymous Credentials using ZK





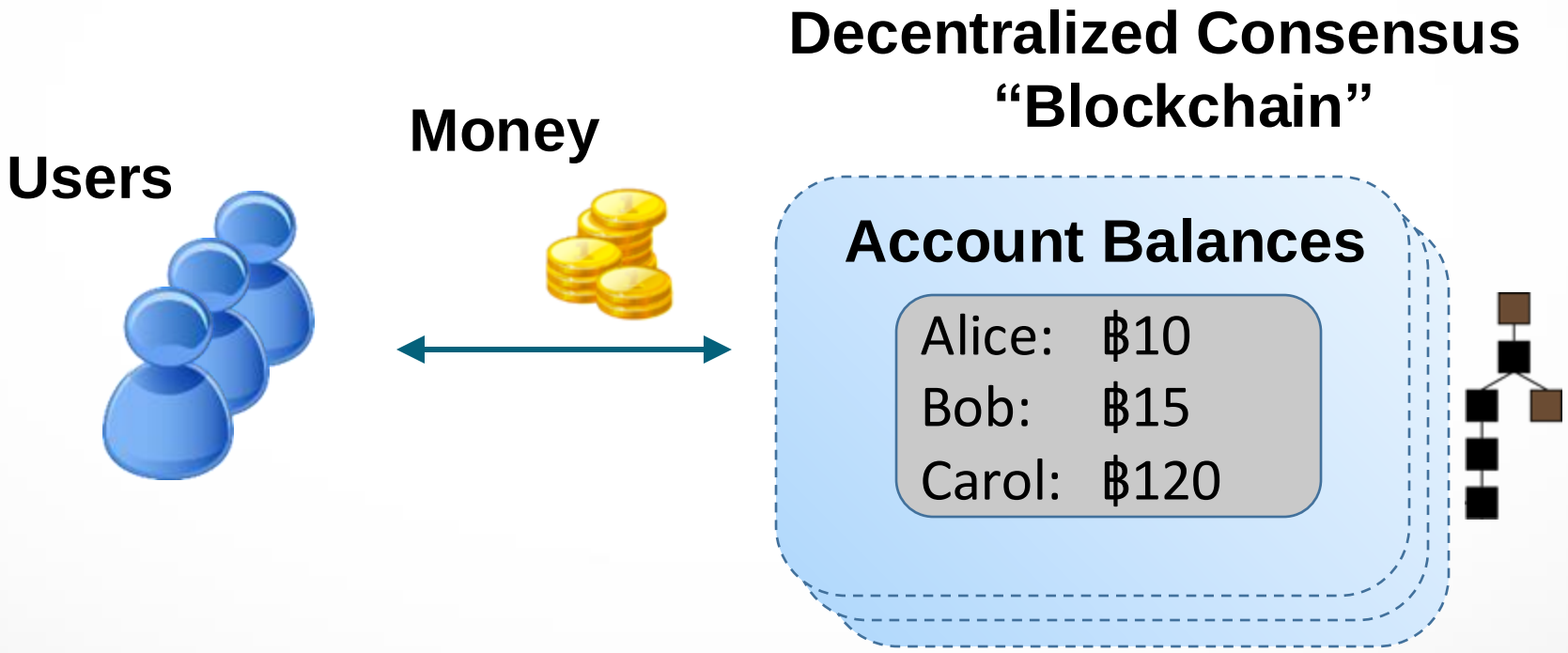
"D2A6B1..8F is a valid signature wrt **pk** on a statement that includes a birthdate before 2000 and the picture "

Prove in zero-knowledge

ZKP for Blockchain Privacy

Bitcoin transparency

- All transactions on the Bitcoin blockchain are visible to the public



Nakamoto's dilemma

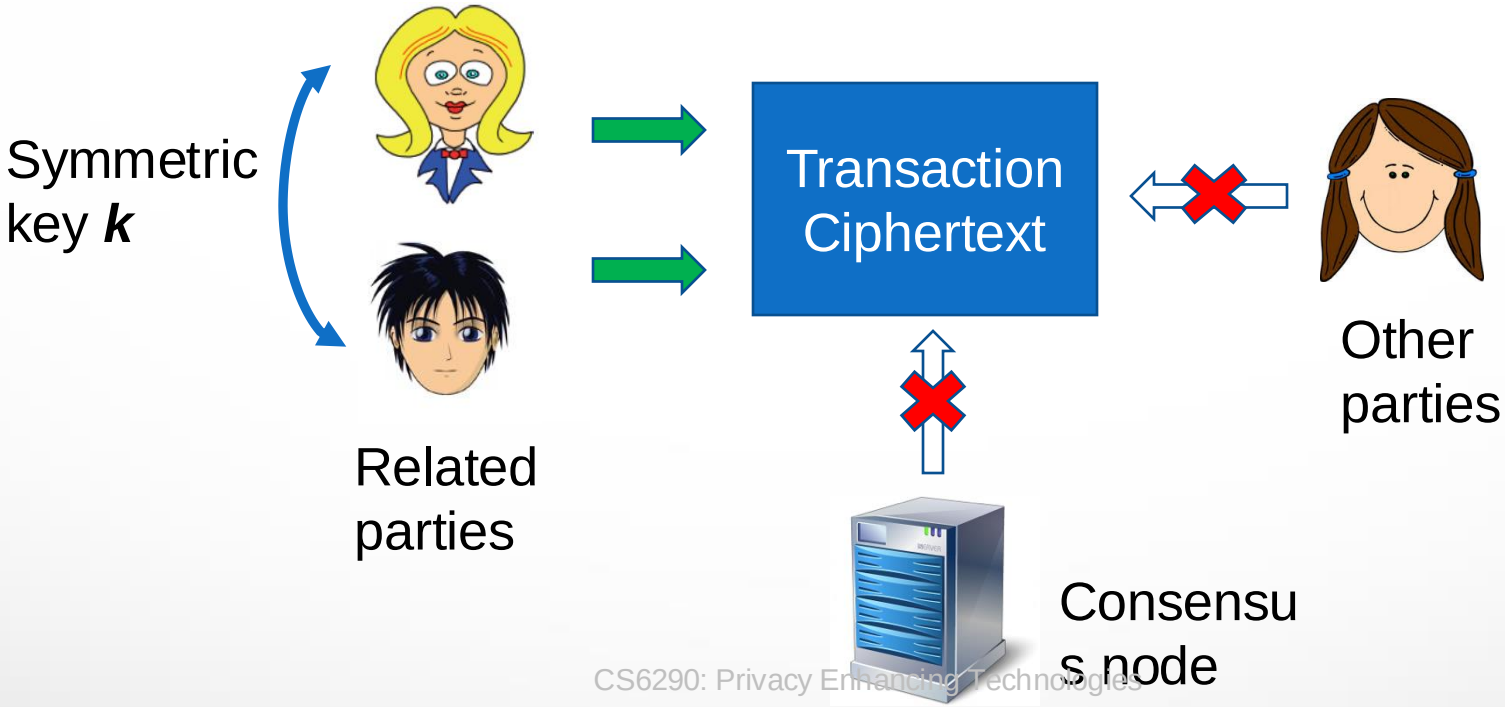
- Suppose you are Satoshi Nakamoto and want to spend your 1,624,500 BTC without attracting any attention…

How?

We need to craft privacy-preserving technique atop blockchains!

First attempt: encrypt the data?

- Leverage standard encryption schemes, e.g., AES
  - Protect privacy
  - But consensus node cannot **validate** immediately

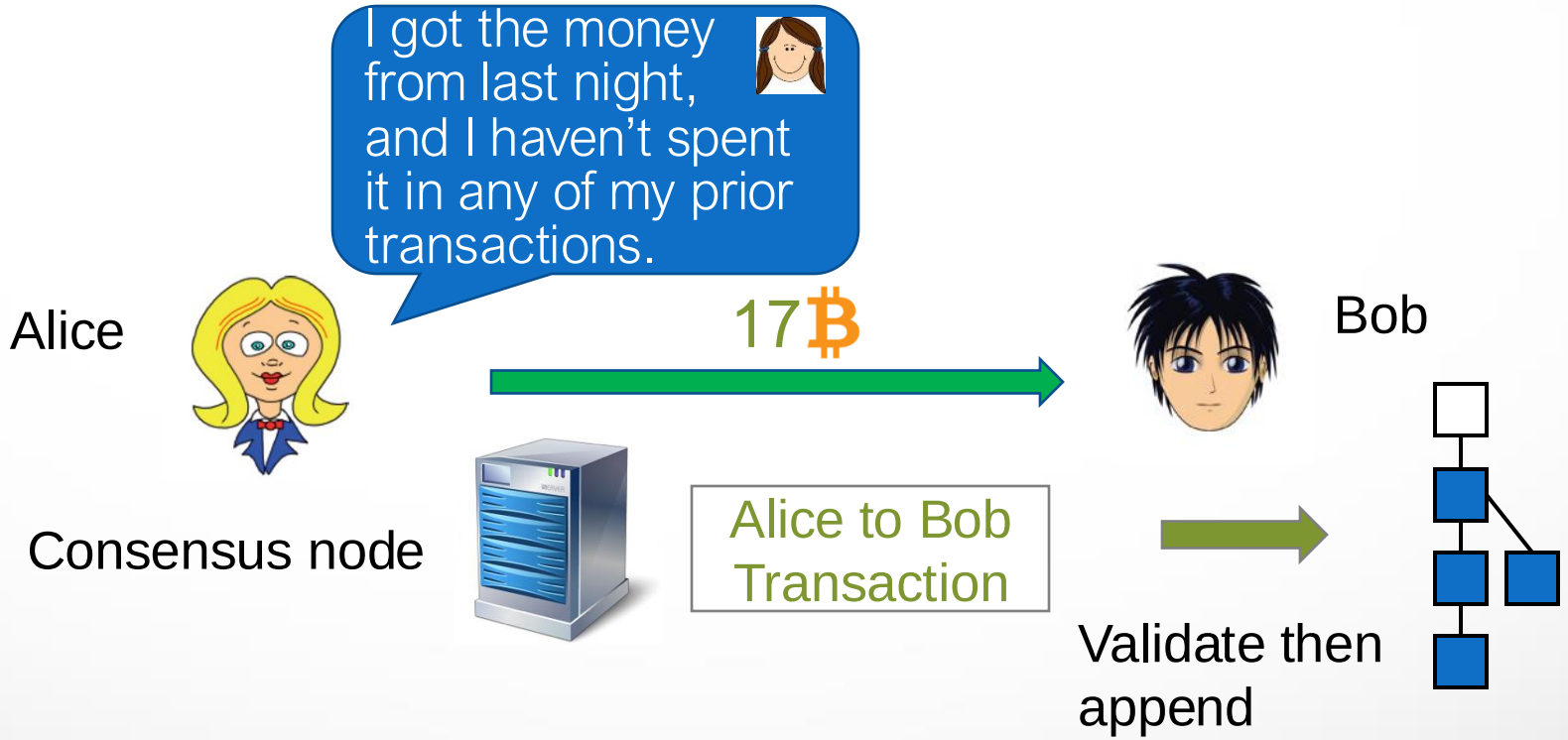


Applicable scenarios

- Transactions that **NOT** relate to on-chain assets (e.g., cryptocurrencies)
  - Store encrypted health record on Bitcoin
  - Store encrypted search index on Ethereum
  - Record an article on Ethereum
  - …
- For consortium/private blockchains that have no on-chain asset
  - Hyperledger Frabric
  - R3 Corda
  - …

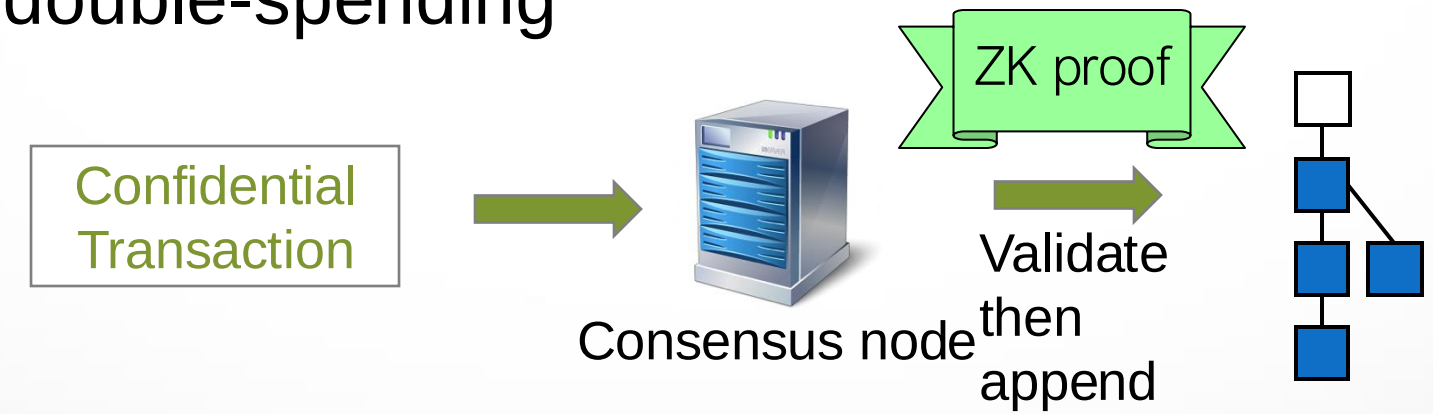
On-Chain-Asset-Related Transactions?

- Must be validated before recording
  - Validate coin ownership
  - Prevent double-spending



Requirements for confidential transactions

- Hide transaction history
  - Reveal neither the **sender**, **receiver**, or transaction **amount**
- But still able to
  - Validate coin ownership
  - Prevent double-spending



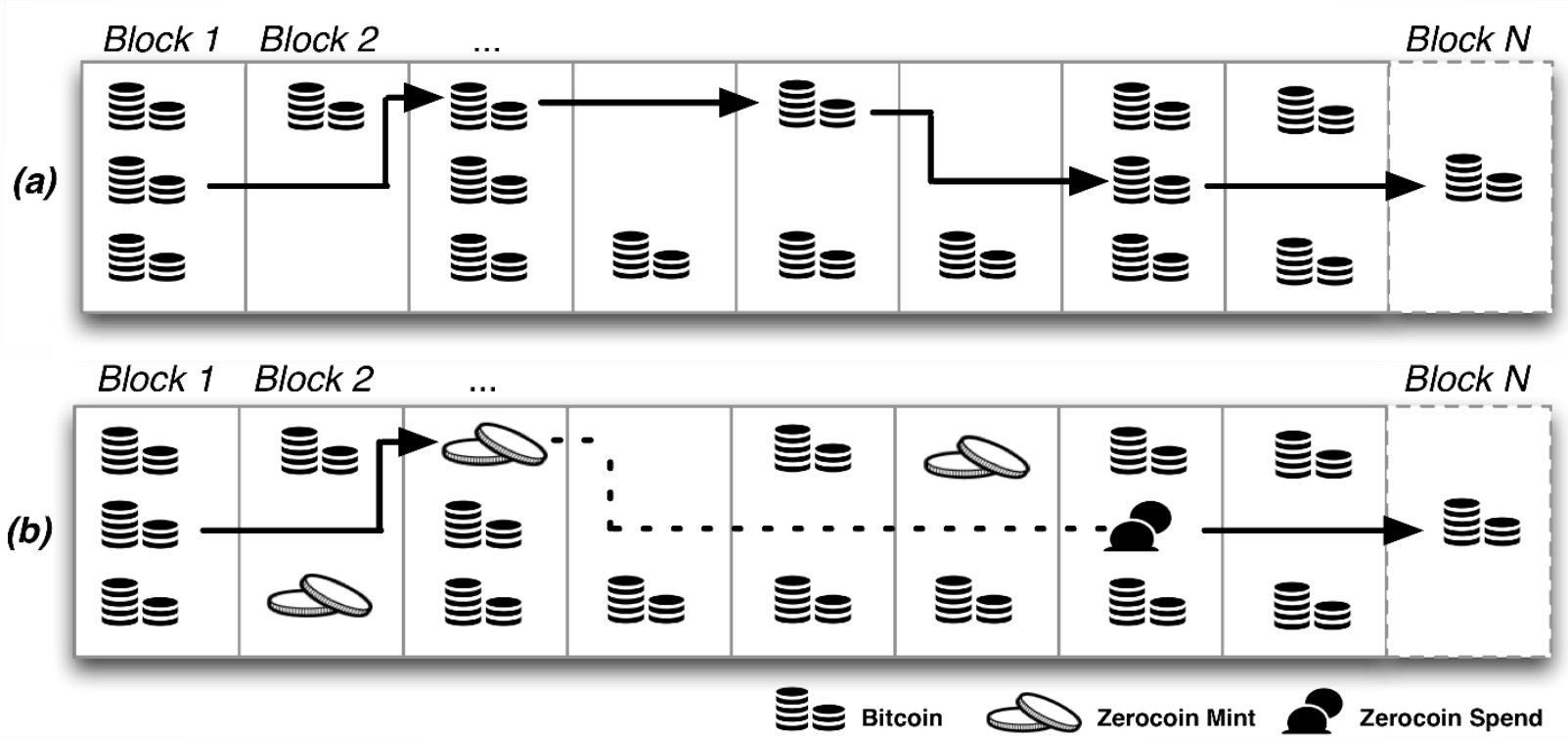
Now we are going to talk about two specific confidential transaction designs:

I. Miers et al.: “Zerocoin: Anonymous Distributed E-Cash from Bitcoin”, IEEE S&P’ 13

&

E. Ben-Sasson et al.: “Zerocash: Decentralized Anonymous Payments from Bitcoin”, IEEE S&P’ 14

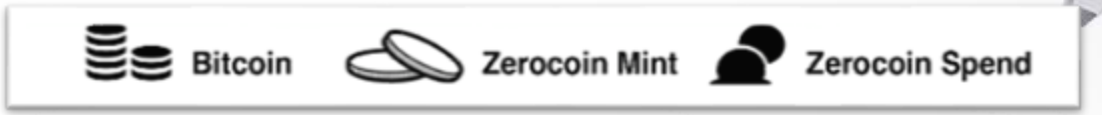
Basic *anonymous e-cash (Zerocoin[1])*



a) illustrates a normal Bitcoin transaction history, with transactions linked.
b) illustrates a Zerocoin chain. The linkage between mint and spend (dotted line) cannot be determined from the block chain data.

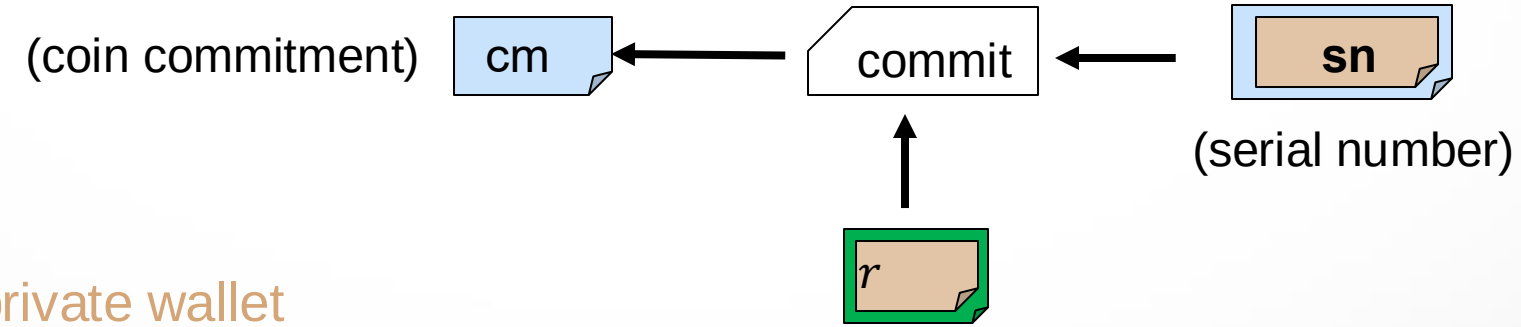
[1] I. Miers et al.: Zerocoin: Anonymous Distributed E-Cash from Bitcoin. Proc. of S&P' 13.

Zerocoin Overview



Minting: I hereby spend 1 BTC to create cm

Spending: I'm using up a coin with (unique) sn, and I know r, and a cm that match sn.

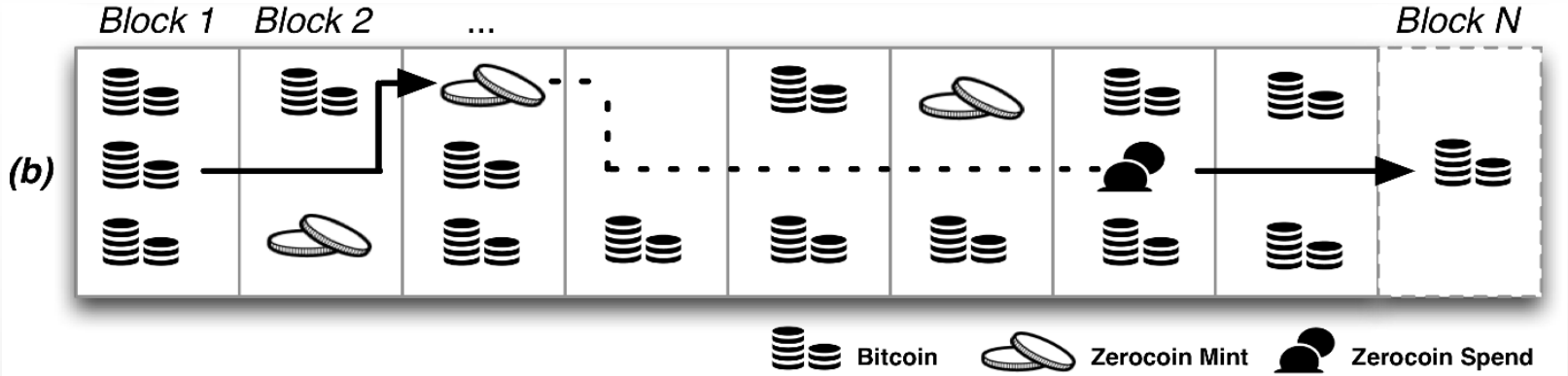


Legend:

- In private wallet
- In public ledger
- Proved to be known

Zerocoin limitations

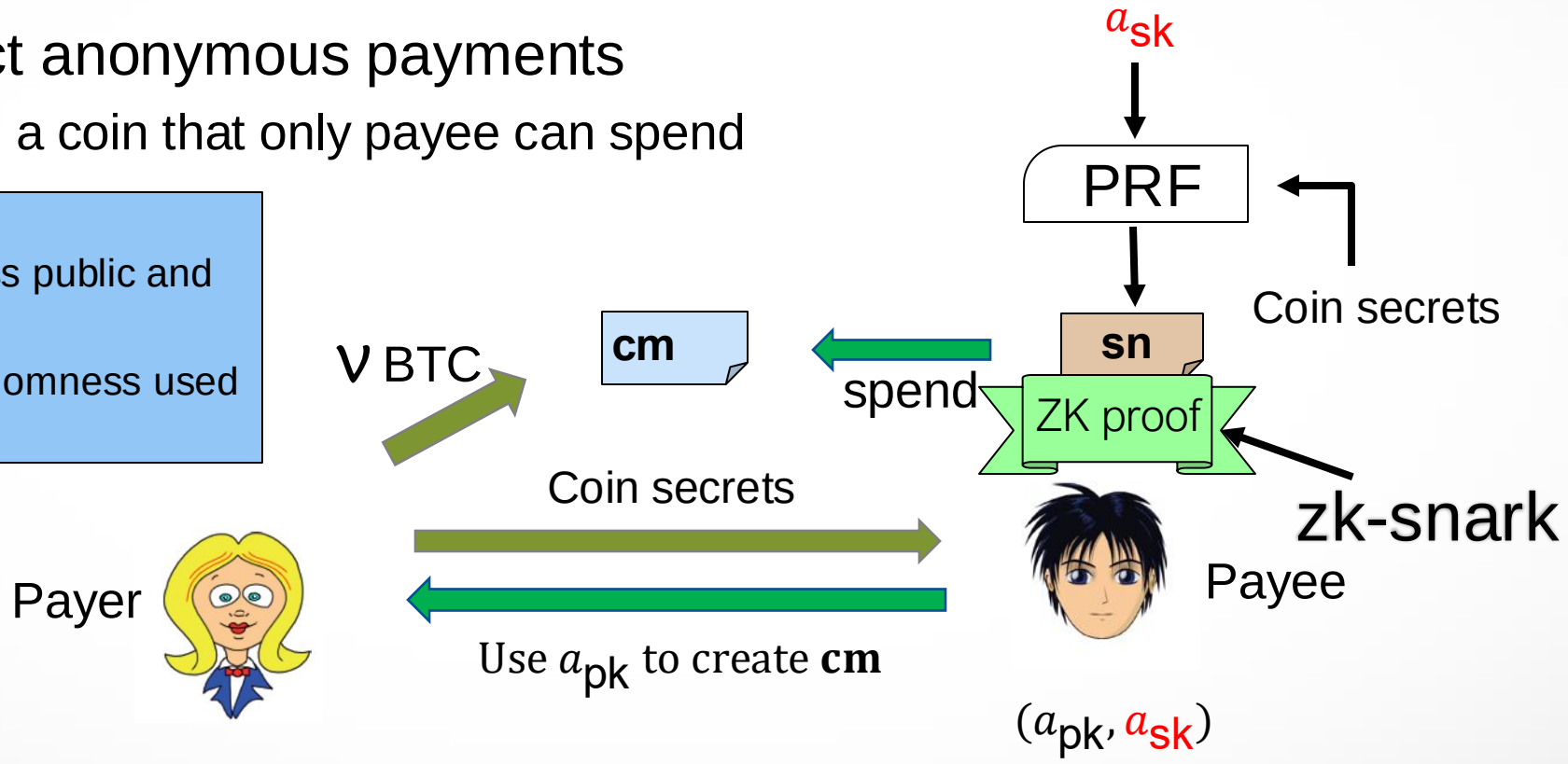
- Not an anonymous currency
  - Only creates a decentralized mix to “wash” bitcoins
  - The transaction amount is fixed
    - Leaks information, why?
  - Cannot support direct anonymous payments via “zerocoins”



Zerocash: Decentralized Anonymous Payments*

- Adding variable denomination values
- Adding direct anonymous payments
  - Payer build a coin that only payee can spend

ν: coin values
(a_{pk}, a_{sk}): address public and secret key
Coin secrets: randomness used for building **cm**



* E. Ben-Sasson et al.: Zerocash: Decentralized Anonymous Payments from Bitcoin. Proc. of S&P' 14.

Denomination is still a side-channel

- **Problem:**
- According to different value **v**, there are different subsets that revealing value information.
- Say I spend 5 BTC, it could only come from people that have minted 5 BTC, and that is a much smaller subset of all of the mints.

- **Solution: merge & split**

Pouring Zerocash coins

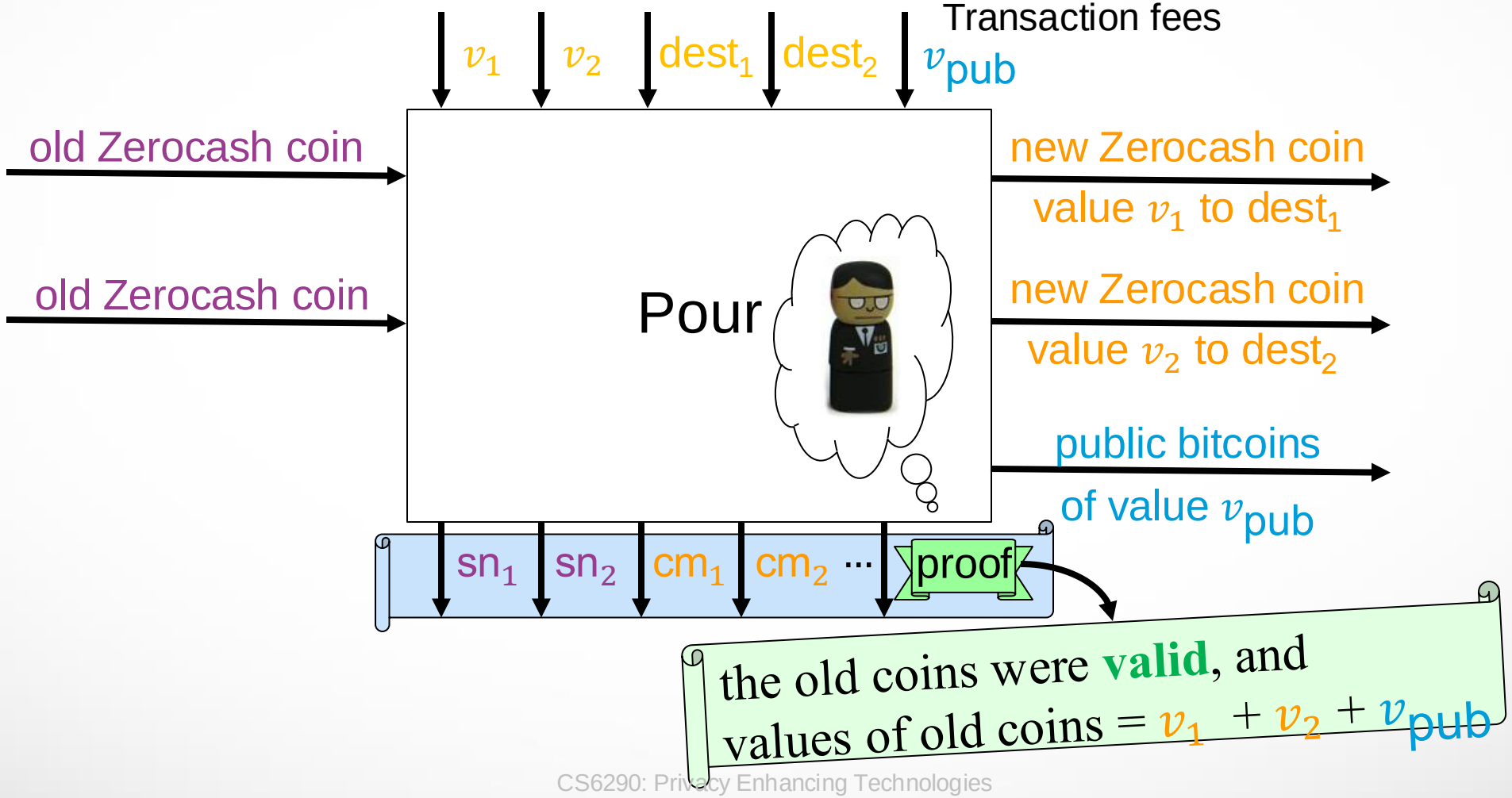
Simplified

Single transaction type capturing:

Sending payments

Exchanging into bitcoins

Transaction fees



The use of MPC and HE on Blockchains

MPC for Blockchain

- Secure randomness generation via MPC
  - E.g., for node selection in PoS protocols
- For achieving secure off-chain on-chain computation
  - Off-load on-chain executions to an off-chain committee, and secretly share the data to each committee member
    - Can preserve off-chain data confidentiality when there is at least one honest member (non-colluding assumption)
  - Require public verifiability for the computation results
- …

HE for Blockchain

- Can achieve general confidential computation on-chain
  - Put HE ciphertexts on-chain and let contract execute them
  - But currently, too expensive to be used on public blockchains like Ethereum…
- Another solution to achieve secure off-chain on-chain computation
  - Compute HE ciphertexts off-the-chain and return results on-chain
  - Verifiability should be enforced as well
- …

The Promise of Advanced Cryptography

Blindfold Computation

- Also useful for “more traditional” security issues
  - E.g., key and credential management, protecting commercial secrets, collaboration on sensitive data, ...

Fast Enough to be Useful

Performance of Advanced Cryptography

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

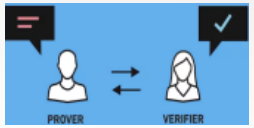
- For most tasks, there is a cryptographic solution with adequate performance
  - Although designing it may take a team of experts

Some Speed Examples

- Lots of examples, meant to demonstrate feasibility of doing “many things” with reasonable performance
  - It’s okay to feel a little dizzy after example #17,352…
- The point is not to compare them
  - They operate in very different settings: “general-purpose” vs. specific functions, different security guarantees, different performance profiles, etc.

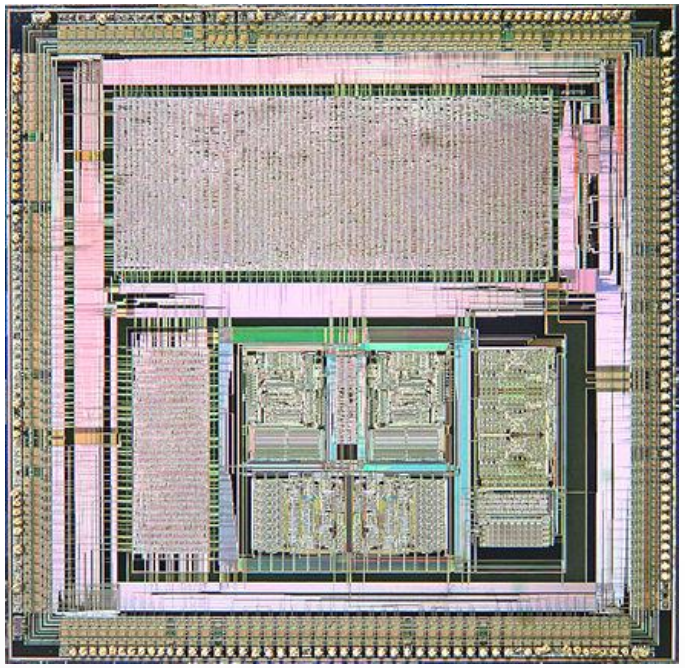
Not all awesome works are included in this list

Some ZK Speed Examples

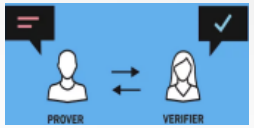


- Proving a 100,000-gate predicate in 1.8sec

  Improved Non-Interactive Zero Knowledge with Applications […] (KKW, CCS 2018)

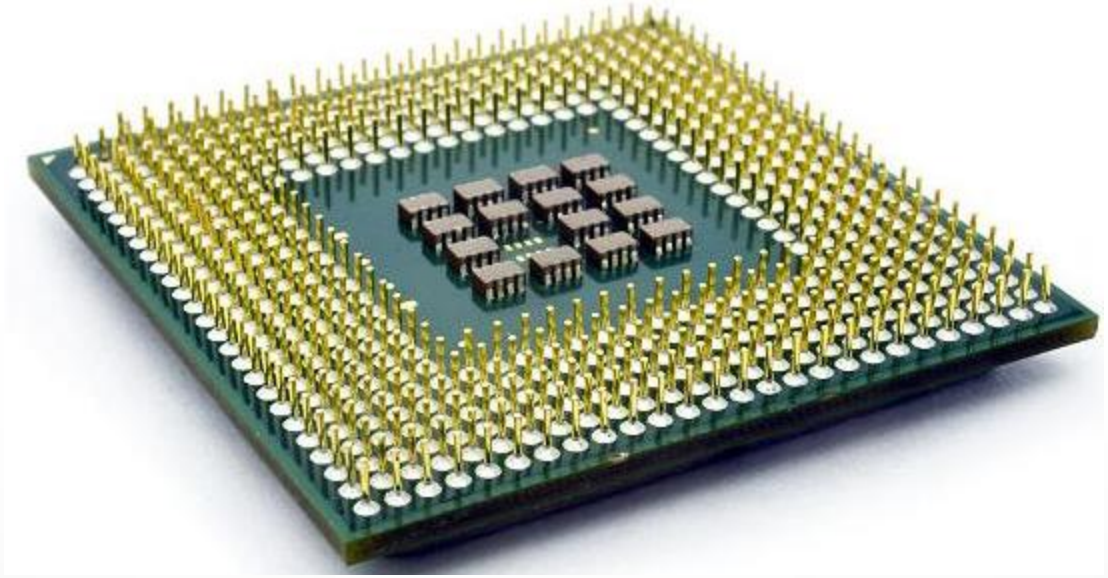


Some ZK Speed Examples

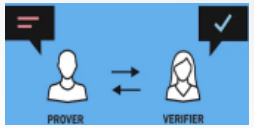


- Proving a 2^{27}-gate predicate on a 64-cluster in ~1.5 hours

  DIZK: A Distributed Zero Knowledge Proof System
  (WZCPS, USENIX Security 2018)



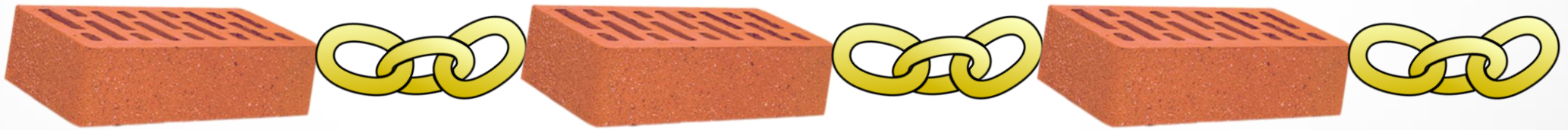
Some ZK Speed Examples



- "I know a pre-image of `0xA4E…1` under SHA2"
  - Proving a 511-node hash tree in 200sec, verifying in 0.7sec

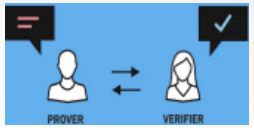
    Libra: Succinct Zero-Knowledge Proofs with Optimal Prover Computation (XZZPS, CRYPTO 2019)

- Useful, e.g., for blockchains



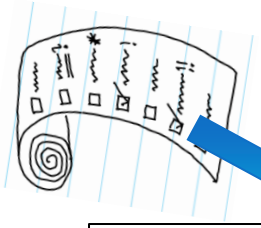
  - Can prove things about the hash values in the blocks

Some ZK Speed Examples



- DNA match against a database
  - zk-STARK, [BBHR, CRYPTO 2019]

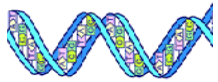
Police has a forensic DNA database



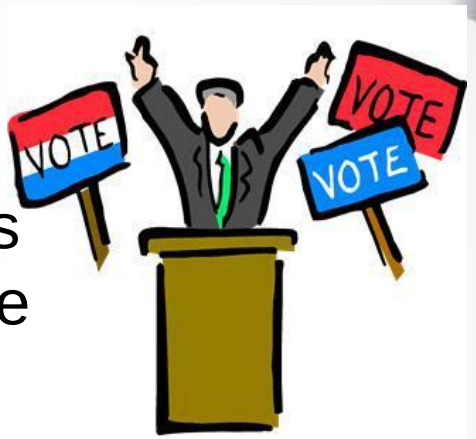
Public commitment 0x3b2a108a

"the sample whose hash is 0xe677d398 does not match anything in the database whose hash is 0x3b2a108a"

Presidential candidate has a DNA sample

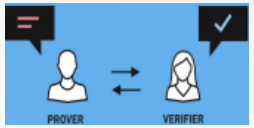


Public commitment 0xe677d398



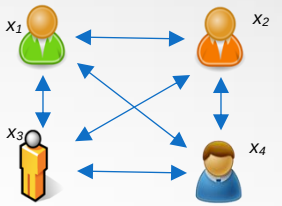
  - Size-100,000 DB, proving in ~1 hour, verifying in milliseconds

ZK Proofs in the Wild

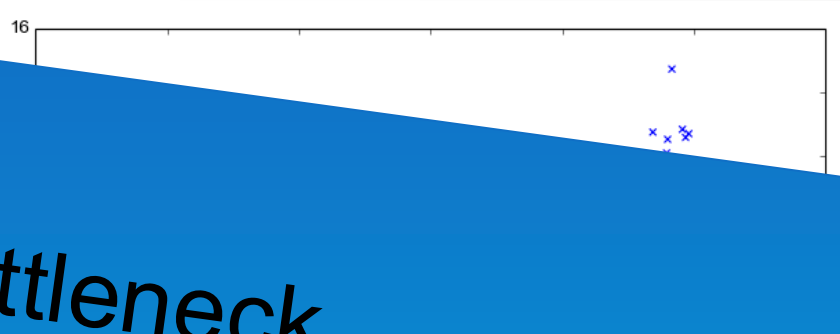


- Digital currencies (Zcash, Monero...)
  - Proving that I have sufficiently many unspent coins on the ledger
  - Constructing proof in seconds, verification in a few msec
- Anonymous credentials (e.g., idemix)
  - Proving that I possess a credential, takes 1-30 seconds
- Tax bracket proofs (Deloitte/QEDit)
  - Commitments to my financial data posted to ledger
  - Then I can prove that I belong to a certain tax bracket
- …

Some MPC Speed Examples

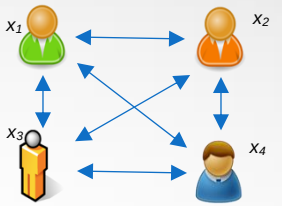


- 1 regression with



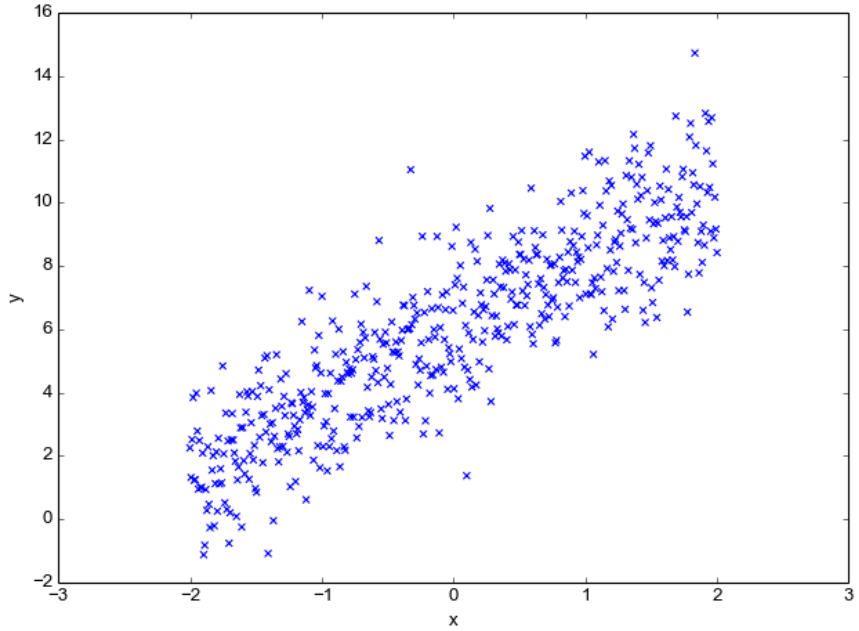
- For most protocols, the bottleneck is communication rather than computation
  - So performance is measures for LAN vs WAN

Some MPC Speed Examples



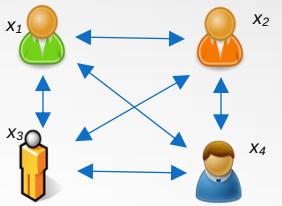
- 10-party linear regression with 4M inputs in 5sec over LAN

  An End-to-End System for Large Scale P2P MPC-as-a-Service […] (BHKL, CCS 2018)

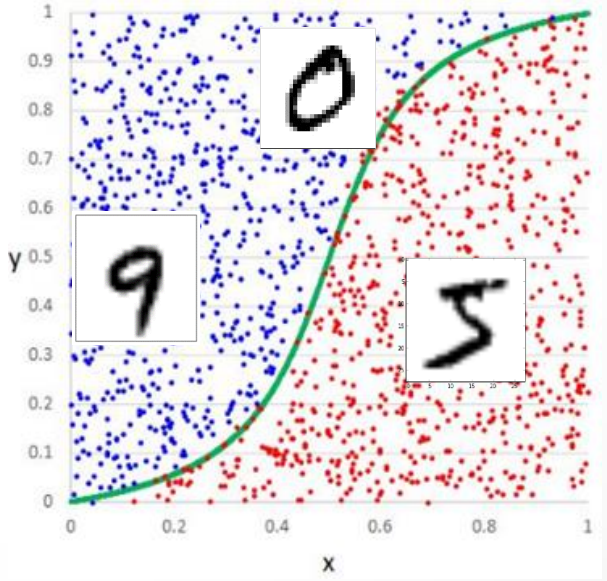


Data is shared among the parties, each holding 400,000 points

Some MPC Speed Examples

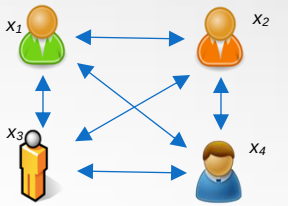


- 10-party regression with 4M inputs in 5sec over LAN
- 4-party logistic regression training in ~5 days over WAN
  - NANOPI: Extreme-Scale Actively-Secure Multi-Party Computation (ZCSH, CCS 2018)



Benchmarked on MNIST data:
1K rows x 784 columns

Some MPC Speed Examples



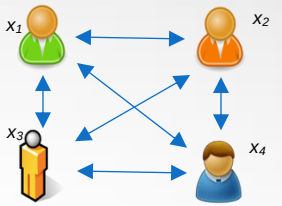
- 10-party regression with 4M inputs in 5sec over LAN
- 4-party logistic regression training in ~5 days over WAN
- 2-party 16x16 Gaussian elimination in 16sec over WAN

HyCC: Compilation of Hybrid Protocols for Practical Secure Computation (BDK, CCS 2018)

$$\begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} & b_1 \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} & b_2 \\ \vdots & \vdots & & \vdots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} & b_m \end{bmatrix}$$

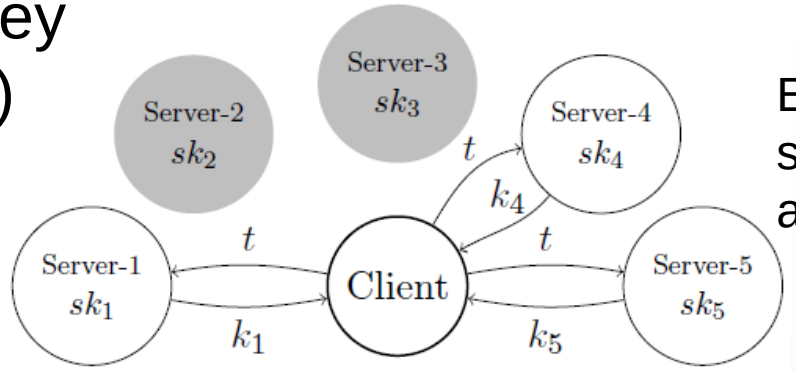
The matrix is shared between the two parties

Some MPC Speed Examples



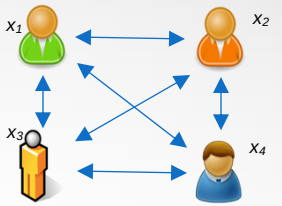
- 10-party regression with 4M inputs in 5sec over LAN
- 4-party logistic regression training in ~5 days over WAN
- 2-party 16x16 Gaussian elimination in 16sec over WAN
- 12-party distributed AES >50,000 enc/sec on WAN

DiSE: Distributed Symmetric-key Encryption (AMMP, CCS 2018)



Encryption key is secret-shared among the servers

Some MPC Speed Examples



- 10-party regression with 4M inputs in 5sec over LAN
- 4-party logistic regression training in ~5 days over WAN
- 2-party 16x16 Gaussian elimination in 16sec over WAN
- 12-party distributed AES >50,000 enc/sec on WAN
- 1000-party distributed RSA key-generation (2048 bits), 5 minutes on a WAN (CHIKMRsVW'19)
  - 35 minutes with 10,000 parties

More MPC Systems, Use-Cases

- Tax Fraud Detection System (Sharemind)
  - Analyzing one month of the Estonian economy in ten days

    "How the Estonian Tax and Customs Board Evaluated a Tax Fraud Detection System Based on Secure Multi-party Computation" (BJSV, FC 2015)
- Virtual HSMs (Unbound), MPC replacing hardware
  - RSA, ECDSA, AES,…, comparable speed to hardware HSM

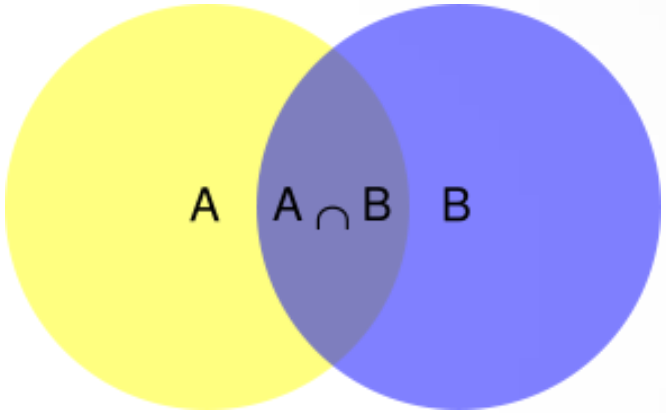
More MPC Systems, Use-Cases

- Tax Fraud Detection System (Sharemind)
  - Analyzing one month of the Estonian economy in ten days
- Virtual HSMs (Unbound), MPC replacing hardware
  - RSA, ECDSA, AES,…, comparable speed to hardware HSM
- Similar patients in a genomic database (iDASH 2016)
  - Best 5 matches against 4000 patients, 1000 markers, in ~30sec
- Clearing-price auction on Hyperledger Fabric, 10-20sec
- Set intersection + impact aggregation between Google and its advertising clients
  - 100,000 ads/clicks in ~400sec, 10MB bandwidth

HE Speed Examples

- Set intersection, size-2^{20} by size-512 sets in 1 sec

  Labeled PSI from Fully Homomorphic Encryption with Malicious Security (CHLR, CCS 2018)



HE Speed Examples

- Set intersection, size-2^{20} by size-512 sets in 1 sec
- Multiplying two 64x64 “real matrices” in ~9 seconds

  Secure Outsourced Matrix Computation and Application to Neural Networks (JKLS, CCS 2018)

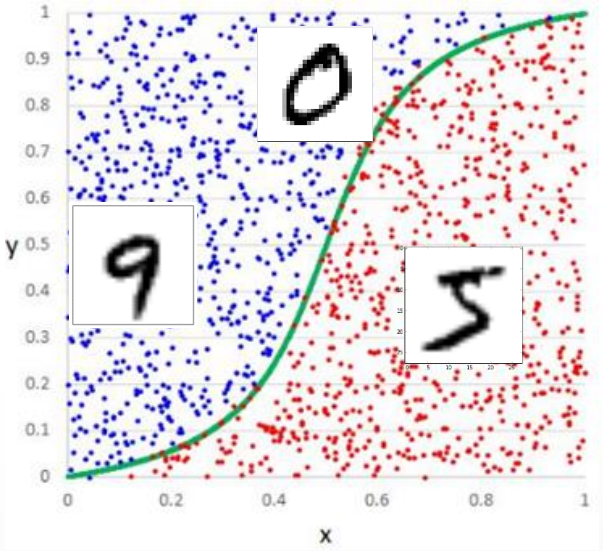
$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix}$$

$$\mathbf{A} \qquad \mathbf{B} \qquad \mathbf{C}$$

More HE Speed Examples

- Inference of simple models on encrypted data
  - 1000 perditions/minute, CNN on MNIST optical characters

    "Crypto-Nets: Neural Networks over Encrypted Data" (DGLLNW, ICML 2016)
  - 8000 predictions/second on 100-feature LR model



More HE Speed Examples

- Inference of simple models on encrypted data
  - 1000 perditions/minute, CNN on MNIST optical characters
  - 8000 predictions/second on 100-feature LR model
- Training a logistic-regression model on genome data
  - Under 10 minutes with 10-15 features, ~1000 rows (iDASH 2017)

    "Logistic Regression Model Training based on the Approximate Homomorphic Encryption" (KSKLC, BMC Medical Genomics 2018)
  - 15-30 minutes to train 30,000 models w/ 5 features (iDASH 2018)

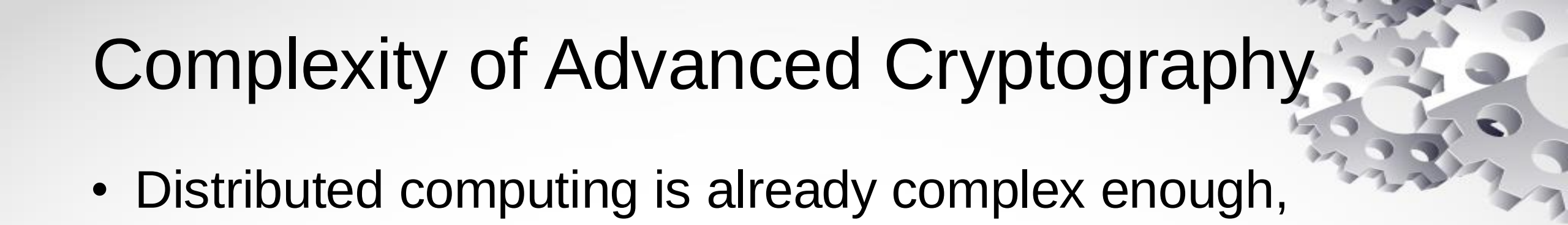
HE Use Cases

- Private cross-organization KYC/AML queries
  - Multiple banks pull encrypted data, run (encrypted) queries
    - Two of the top three entries in the 2019 Global AML and Financial Crime TechSprint Hackathon in London used these techniques
- Predicting client needs based on encrypted past data (MHSCBRQASF, eprint 2019/1113)
  - Sharing encrypted data between business units
  - Proof-of-concept project in Bradesco Bank (Brazil)

Such awesome performance, how come we're not seeing these tools everywhere?

Not “Generally Usable" Yet

Complexity of Advanced Cryptography



- Distributed computing is already complex enough, “advanced crypto” adds secrecy considerations
- Good performence requires extreme optimizations
  - Straightforward implementation will be exceedingly slow
  - Small application-level changes can make a big difference in how to best optimize for it
- Tension between simplicity/usability and performance

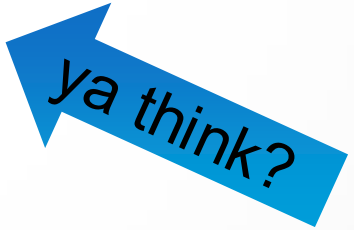
Implementations

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

Code Quality

- Most code written in C/C++
  - By researchers with limited C/C++ experience

```
parts.push_back(CtxtPart(*ptr,handle));
if (negative) parts.back().Negate(); // not thread-safe??
```



Example: Secure-MPC Communication

- 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

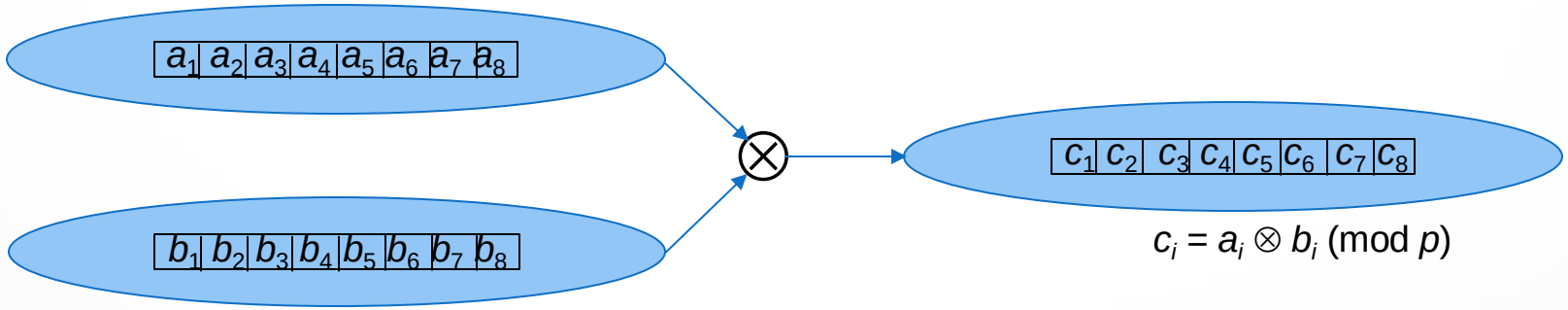
```
int main(int argc, char** argv)
{
  const char* addr = "127.0.0.1";
  int port = 7766;

  if (m_nPID == SERVER_ID) { //Play as OT sender
    InitSender(addr, port, glock);
    OTExtSnd* sender = InitOTExtSnd(prot, m_nBaseOTs, m_nChecks, usemecr, ftype, crypt);
    [...]
  }
  else { //Play as OT receiver
    InitReceiver(addr, port, glock);
    OTExtRec* receiver = InitOTExtRec(prot, m_nBaseOTs, m_nChecks, usemecr, ftype, crypt);
    [...]
  }
}
```

Example: Secure-MPC Communication

- Communication between parties is a bottleneck in many protocols for secure multi-party computation
  - To optimize, many MPC libraries work with sockets
- 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
  - Your best option is to look for another library

Example: Data Encoding for HE

- Ciphertext operations in contemporary HE are 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 HE

- 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
- Almost no tool support for making these choices

Taming the Complexity

- How to make advanced cryptography usable to non-expert programmers?
- Usable “toolbox libraries” for common tasks
  - Low level: arithmetic, sorting, linear algebra, …
  - Mid level: graphs algorithms, set intersection, ML tools, …
  - Domain specific tasks (medical, financial, …)
- Design libraries as “middleware”
  - One component in larger systems
  - Don’t assume that the library “owns” the relevant resources

Taming the Complexity

- How to make advanced cryptography usable to non-expert programmers?
- Frameworks, compiler support
  - Some work over last 10+years
    - e.g., Fairplay, Sharemind, Obliv-C, …
  - Considerable work reported in ACM CCS 2018
    - *An End-to-End System for Large Scale P2P MPC-as-a-Service[…]* (BHKL)
    - *HyCC: Compilation of hybrid protocols for Practical Secure[…] (BDK)*
    - *Generalizing the SPDZ CompilerFor Other Protocols* (ABFKLOT)
    - ALCHEMY: A Language and Compiler for HE […] (CPS)

Time to Put These Tools to Use

- The need is acute
- Push back against IT systems that put us in a fishbowl

- **Personalized services are possible without access to personal information**
  - Don’t believe people telling you they’re too slow

Time to Put These Tools to Use

- Cryptographers must put emphasis on usability and “mundane” software engineering aspects
  - Although improving performance is still important
- System builders should try to use what tools exist
  - Complain bitterly to your fellow cryptographers if their tools are too hard to use
- For now, keep designing one-off systems
  - Hopefully, some generalizations will emerge
  - These technologies are best suited for that type of applications

Time to Put These Tools to Use

- Some starting points to access these technologies:
  - Zero-Knowledge: https://zkp.science/
  - Secure-MPC: https://github.com/rdragos/awesome-mpc and http://www.multipartycomputation.com
  - HE: http://homomorphicencryption.org/
- We really need HOWTO documents
  - With application focus
  - Any volunteers to write them?

Incentives for Blindfold Computation?

- Customer demand?
  - Seems unlikely
- Government regulation?
  - Maybe, in some cases
- Developers wanting to do the right thing?
  - That's us, we have some choice in the systems that we build
  - Don't build systems that require users to hand over their data
    - It will be abused

Where To Go Next?

- Much work remains to improve efficiency
  - Especially for private machine learning
- But usability is where the main challenge lies
  - Building usable libraries, framework, compilers, …
  - System builders should try to use whatever tools exist
- This is a long-term project

A Few Useful Links

- Some starting points to access these technologies:
  - Zero-Knowledge: https://zkp.science/
  - MPC: https://github.com/rdragos/awesome-mpc
  - HE: https://homomorphicencryption.org/

Summary: Advanced Cryptography is

Needed

- Can help prevent data abuse
- An under-utilized tool

Fast enough to be useful

Not "generally usable" yet

We are making some progress

Related References

- O. Goldreich. Foundations of Cryptography — Volume I (Basic Tools), in Cambridge University Press, 2001.
- O. Goldreich. “Zero knowledge twenty years after their invention.” Tutorial at FOCS `02.
- Eli Ben-Sasson et al. “Succint non-interactive zero knowledge for a von Neumann Architecture”, in Proc. of USENIX Security 14.
- Jens Groth. “On the size of pairing-based non-interactive arguments”, in Proc. of EUROCRYPT 16.
- Craig Gentry: Fully homomorphic encryption using ideal lattices. STOC 2009: 169-178
- Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. CRYPTO 2011, 2011.
- Andrew Chi-Chih Yao: How to Generate and Exchange Secrets (Extended Abstract). FOCS 1986: 162-167