Advanced Cryptography: Promise and Challenges

Shai Halevi, IBM Research

ACM-CCS, October 2018

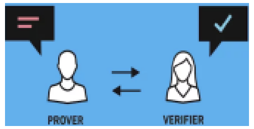
What's "Advanced Cryptography"?

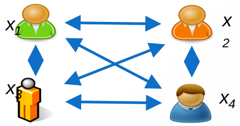
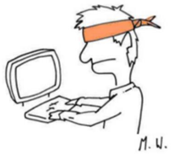
- Cryptography beyond encryption, signatures
  - **Protecting computation, not just data**

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

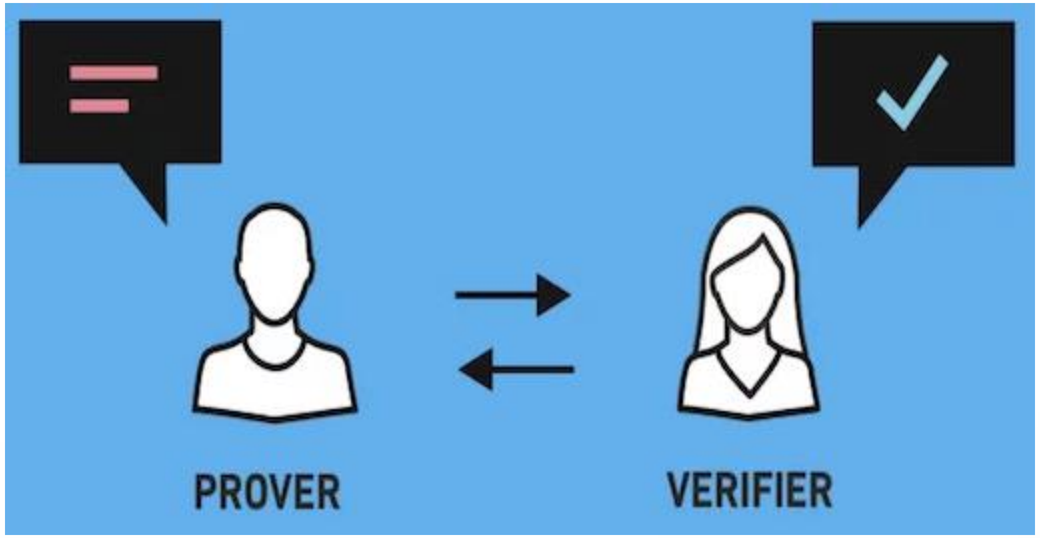
Not "generally usable" yet

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

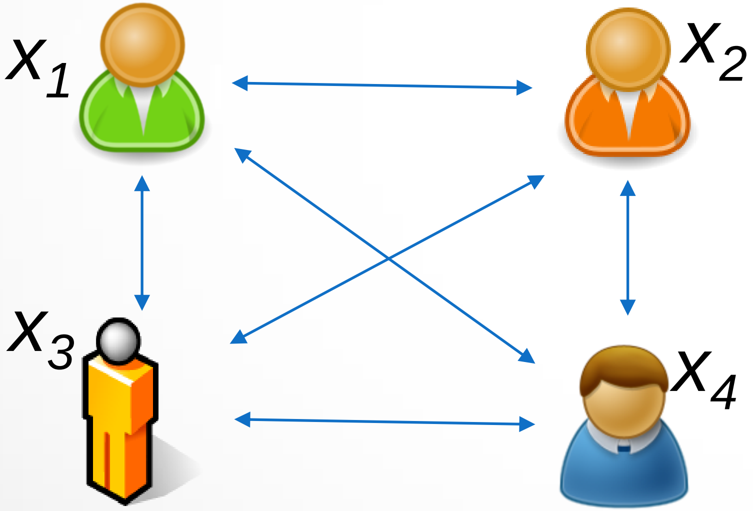
Zero Knowledge Proofs

- 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

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]

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)

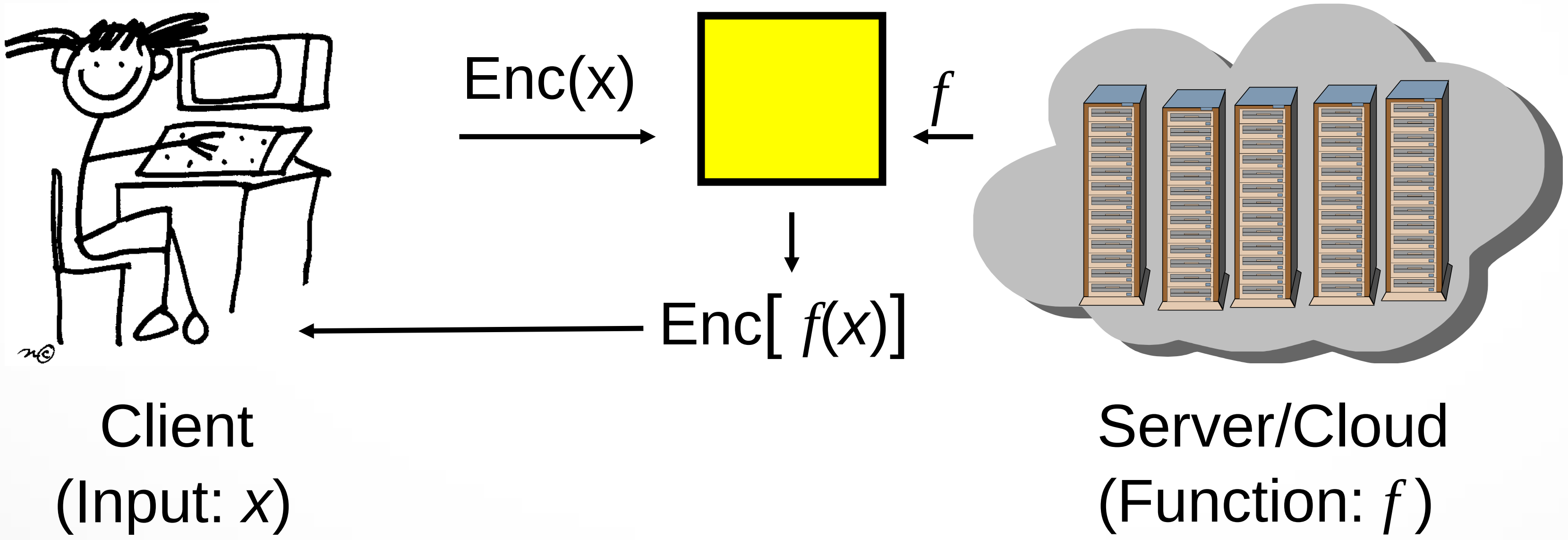
- Example: medical data
  - Evaluating the effectivness of a treatment

    ***f* (patient1Data, patient2Data, ...) = effective/not-effective**

  - Data for different patients held by different clinics
  - Can compute this without 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#] = reviews for nearby coffee shops
  - I get back encrypted recommendation for coffee shops within two blocks

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

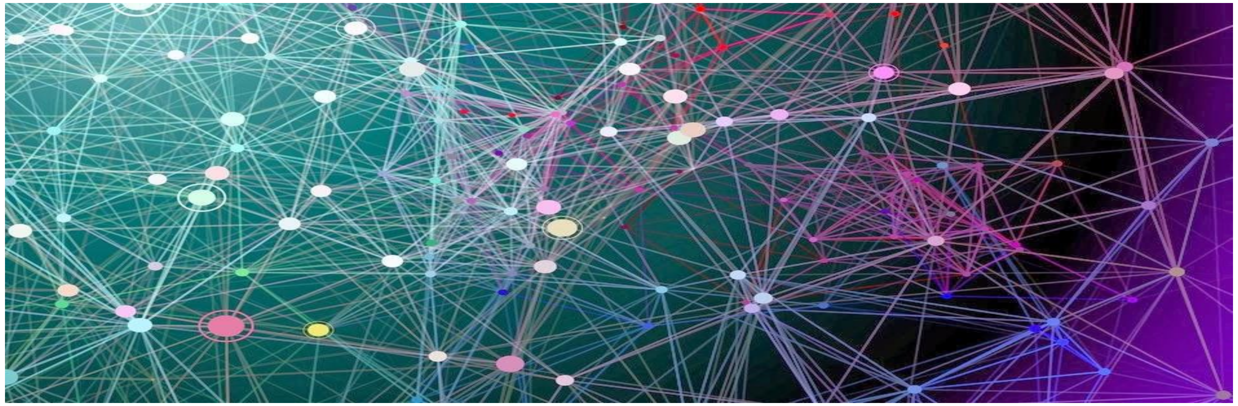
Your Privacy for Sale

- 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

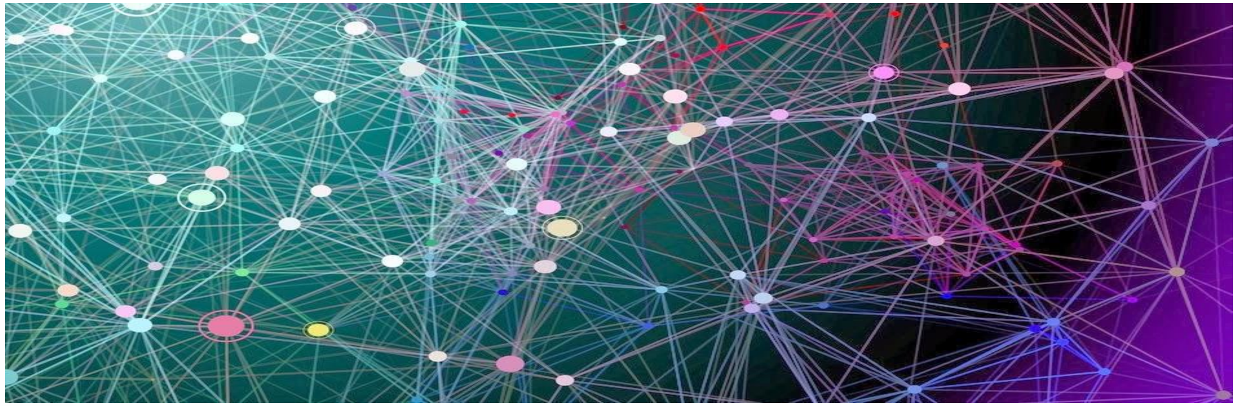
- The entire IT industry is busy making it easier
  - Larger collections, better ways to link, process them



- 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

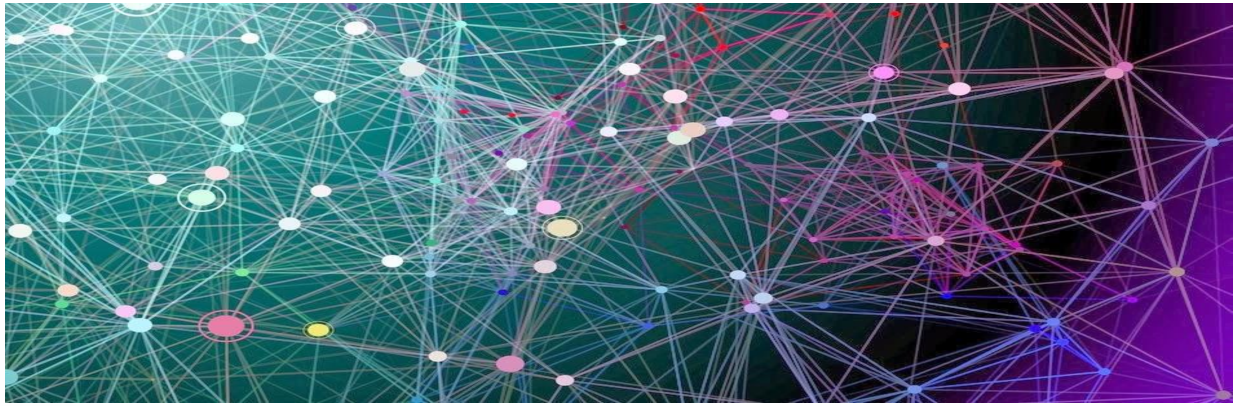
- The entire IT industry is busy making it easier
  - Larger collections, better ways to link, process them



- 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

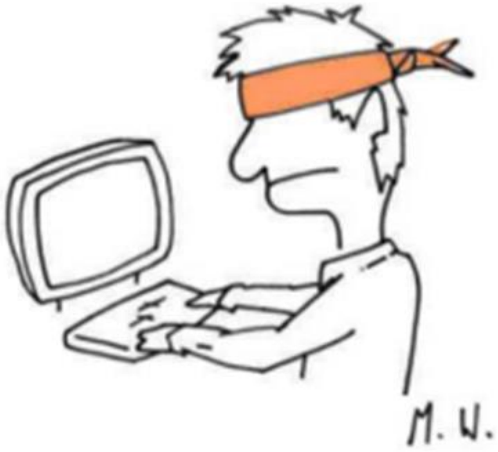


IT, security professionals

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

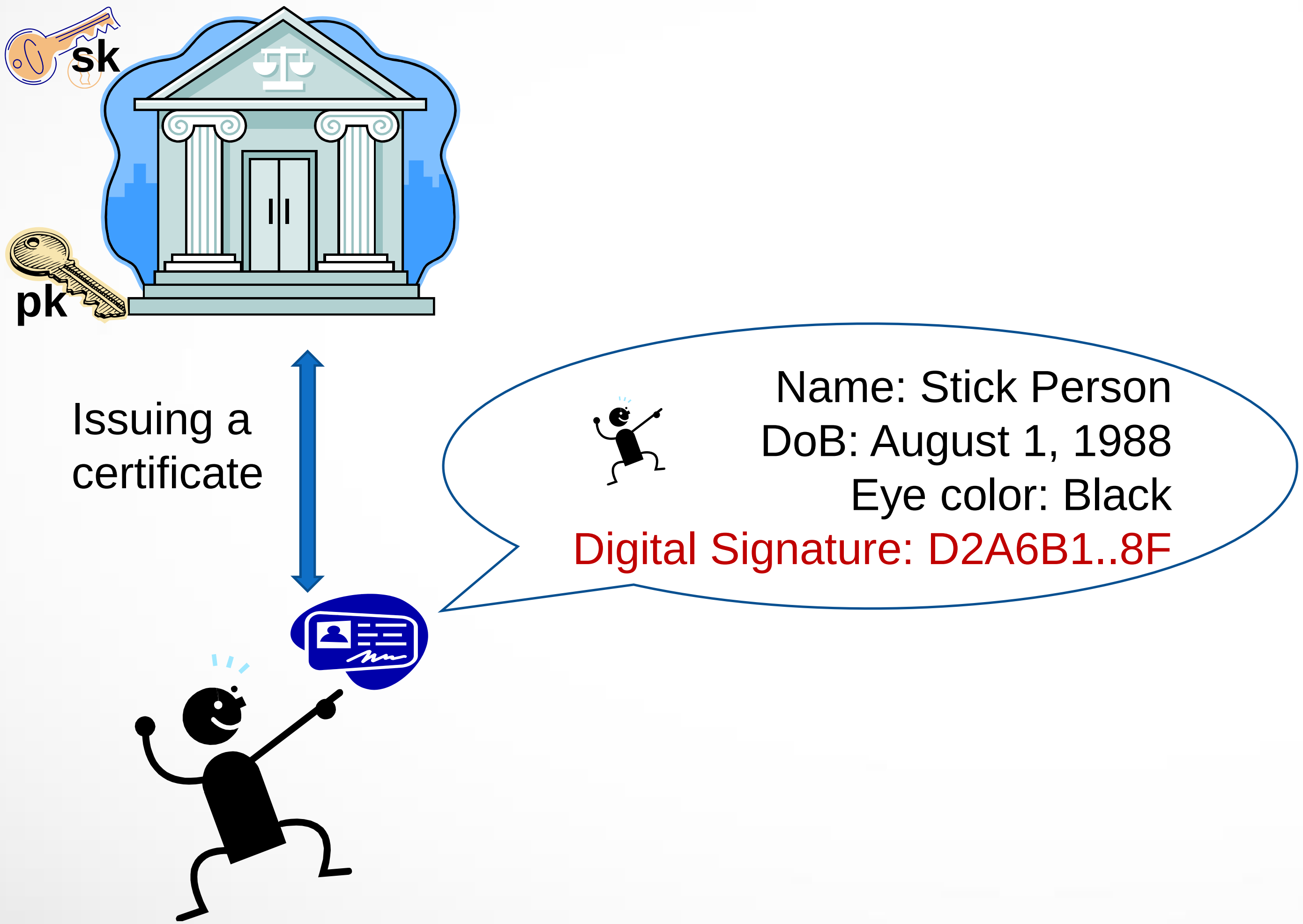
The Promise of Advanced Cryptography

Blindfold Computation

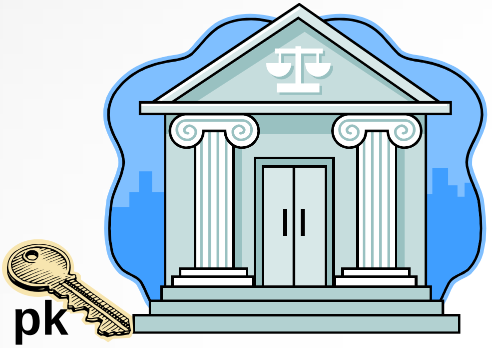


- The ability to process data without ever seeing it
  - Personalized services without access to private information
  - You cannot abuse data that's not there

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

Example: No-Fly-List Using 2PC

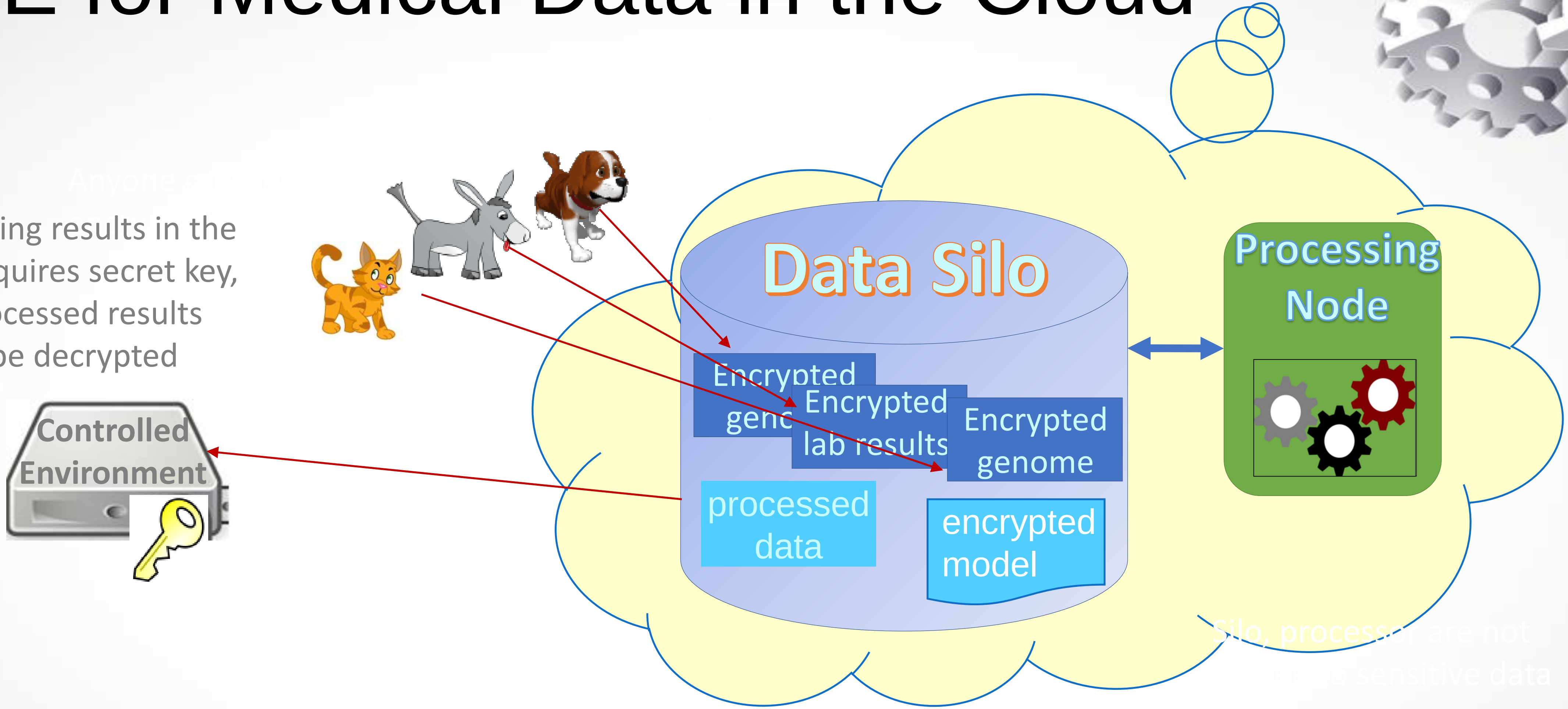
Police has a list of suspects

Airline has a list of passengers

Output is the intersection of the two lists

HE for Medical Data in the Cloud

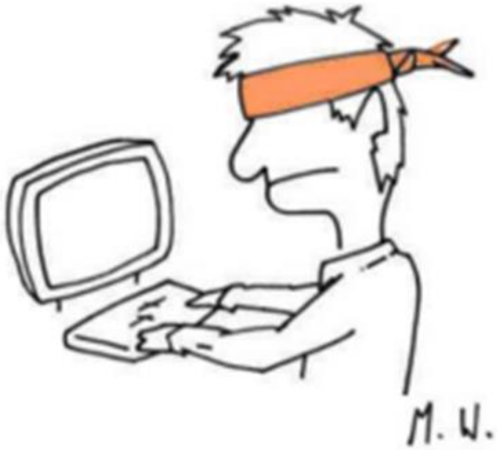
Recovering results in the clear requires secret key, only processed results should be decrypted



- “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



- 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

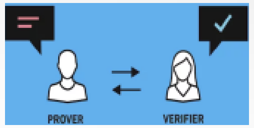
Some Speed Examples

- 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.

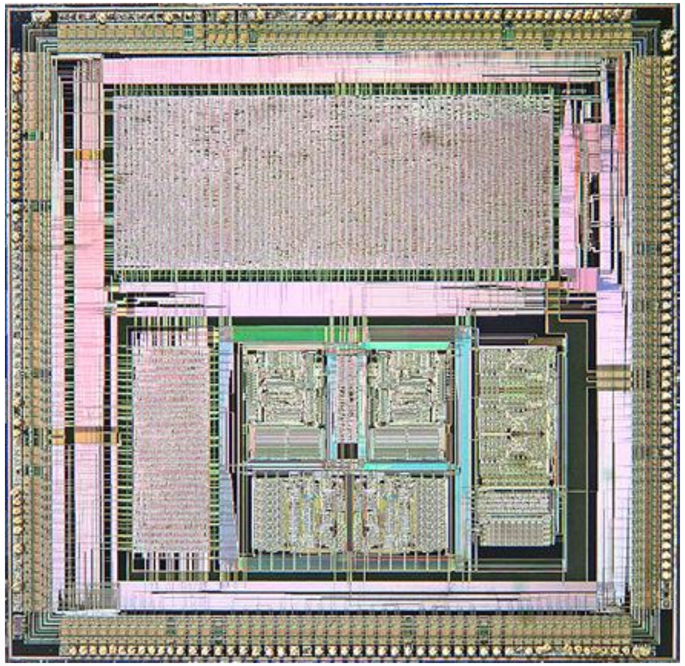
My apologies if I didn’t include your awesome work in this list

Some ZK Speed Examples



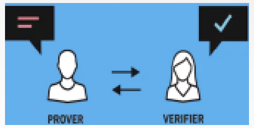
- Proving a 100,000-gate predicate in 1.8sec

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



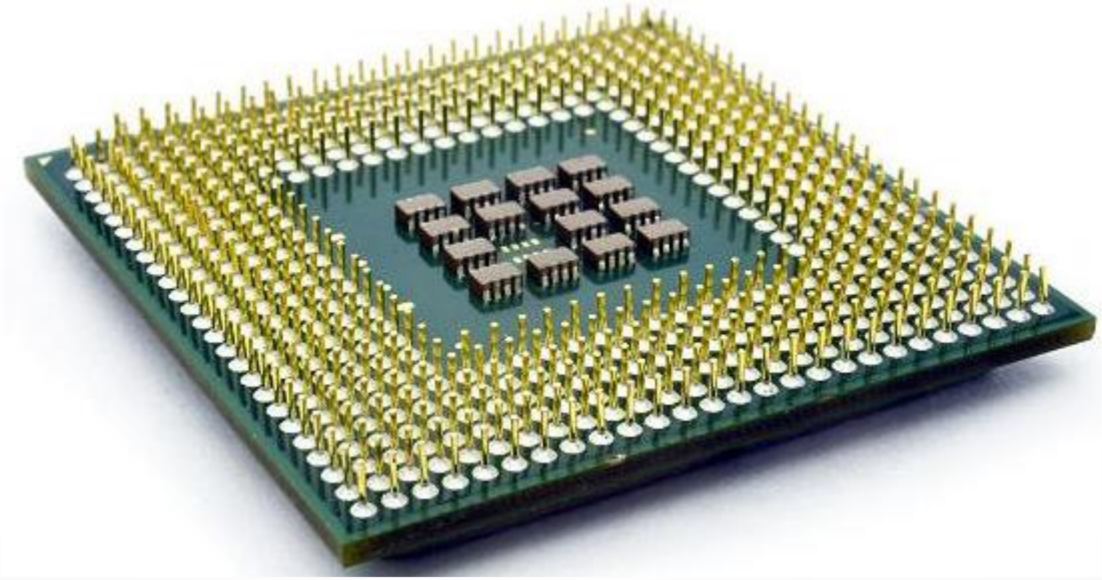
From this conference

Some ZK Speed Examples

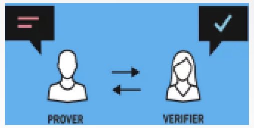


- Proving a 2^{27}-gate predicate on a 64-cluster in ~1.5 hours

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



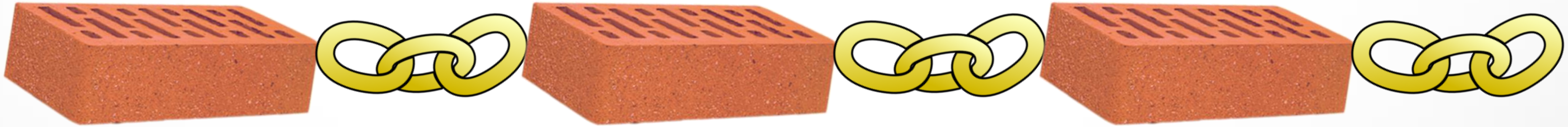
Some ZK Speed Examples



- “I know a pre-image of `0xA4E…1` under SHA”
  - Proving at 100 pre-images/sec, verifying at 5000/sec

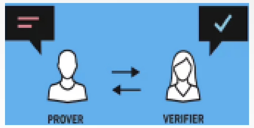
    Ligero: Lightweight Sublinear Arguments Without a Trusted Setup (AHIV, CCS 2017)

- Useful, e.g., for blockchains



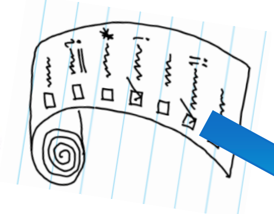
  - Can prove things about the hash values in the blocks

Some ZK Speed Examples



- DNA match against a database (zk-STARK, [BBHR, 2018])

Police has a forensic DNA database



Public commitment 0x3b2a108a

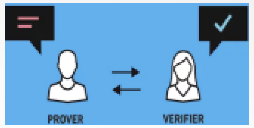
"the sample whose hash is 0xe677d398 does not match anything in the database whose hash is 0x3b2a108a"

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

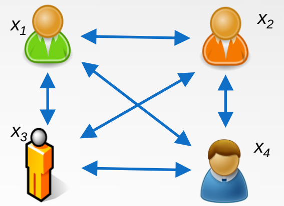


- Digital currencies (zCoin, Zcash, ...)
  - Proving that I have sufficiently many unspent coins on the ledger
  - Constructing proof in ~1min*, verification in a few msec

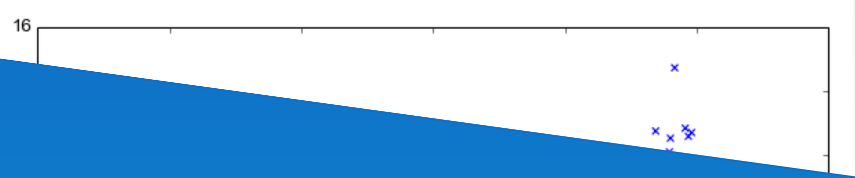
* Soon to be much faster

- Anonymous credentials (e.g., idemix)
  - Proving that I possess a credential, takes 1-30 seconds
- Private payments in the Brave browser (using Anonize)
- 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

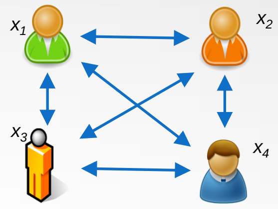


- 1 … regression with



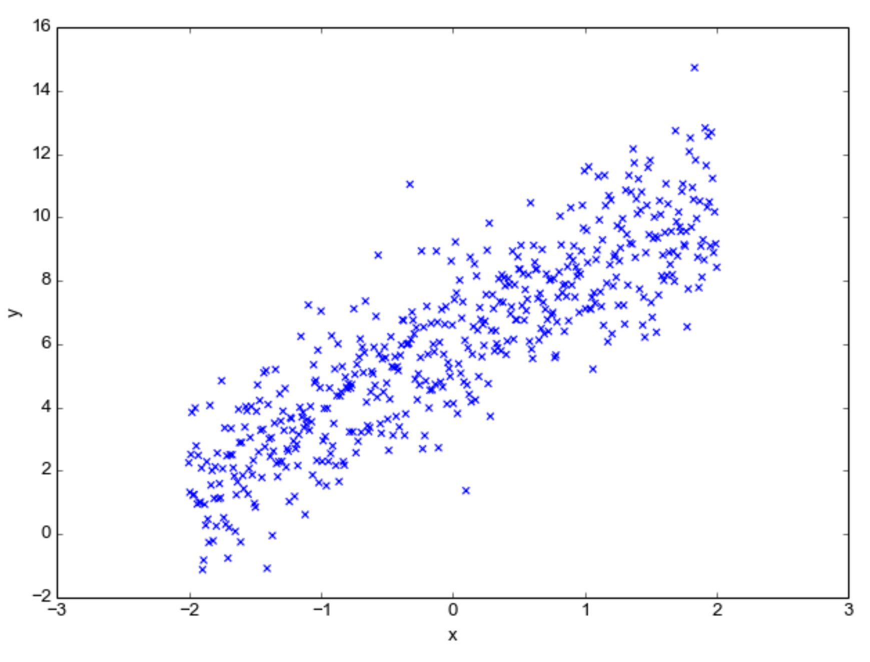
- For most protocols, the bottleneck is communication rather than computation
  - ➢ So performance is measures for LAN vs WAN

Some MPC Speed Examples



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

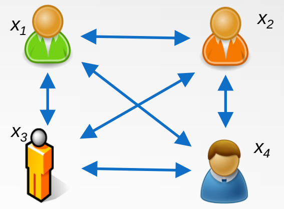
  An End-to-End System for Large Scale P2P MPC-as-a-Service […] (BHKL, CCS 2018)



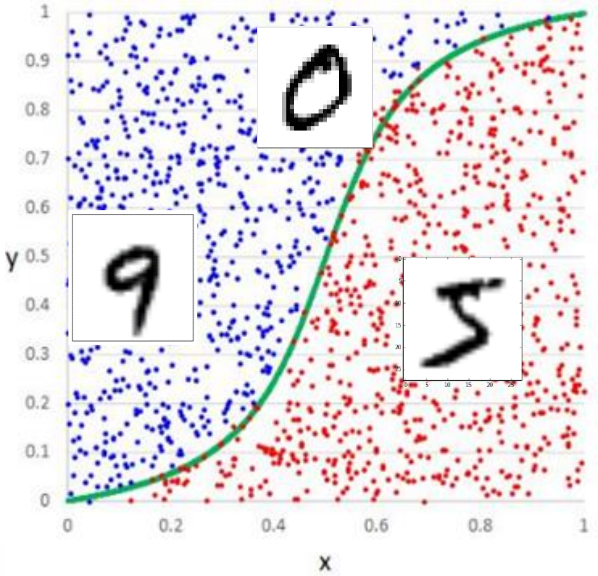
Data is shared among the parties, each holding 400,000 points

Cherry picked from this conference

Some MPC Speed Examples



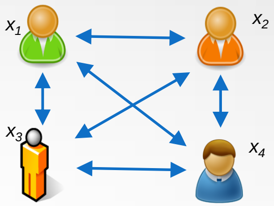
- 10-party regression with 4M inputs in 5sec over LAN
- 4-party logistic regression training in ~5 days over WAN
  - NANOPI: Extreme-Scale Actively-Secure Multi-Party Computation (ZCSH, CCS 2018)



Benchmarked on MNIST data:
1K rows x 784 columns

Cherry picked from this conference

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

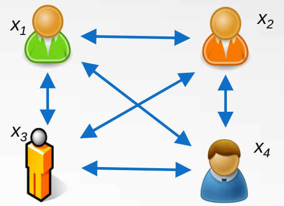
HyCC: Compilation of Hybrid Protocols for Practical Secure Computation (BDK, CCS 2018)

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

The matrix is shared between the two parties

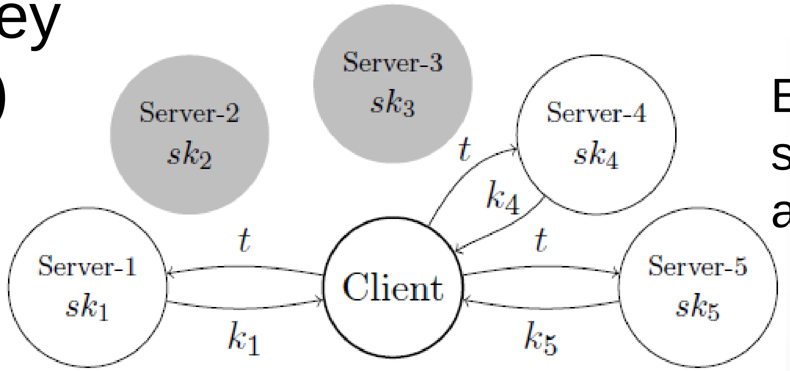
Cherry picked from this conference

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

Cherry picked from this conference

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

    "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
- Similar patients in a genomic database (iDASH 2016)
  - Best 5 matches against 4000 patients, 1000 markers, in ~30sec

    "Privacy-Preserving Search of Similar Patients in Genomic Data" (AHLR, PoPETS 2018)

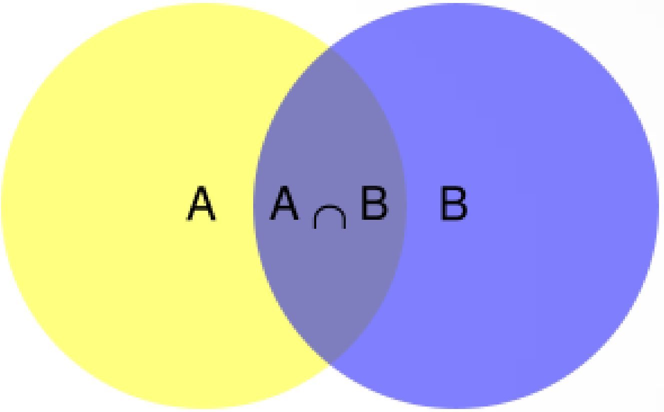
- Clearing-price auction on Hyperledger Fabric, 10-20sec

  "Initial Public Offering (IPO) on Permissioned Blockchain using Secure Multiparty Computation" (BDHHJMZ 2018)

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)



From this conference

HE Speed Examples

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

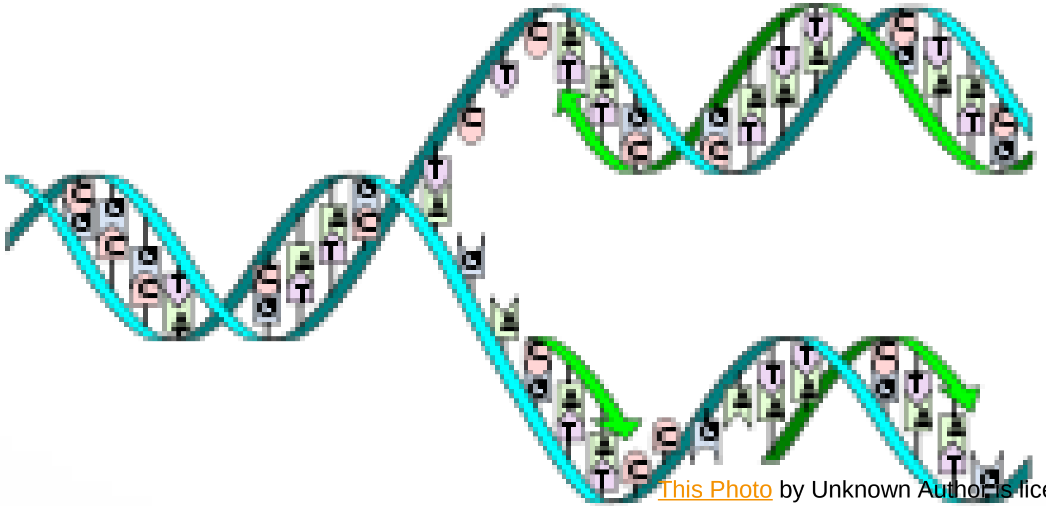
  Secure Outsourced Matrix Computation and Application to Neural Networks (JKLS, CCS 2018)

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

From this conference

More HE Speed Examples

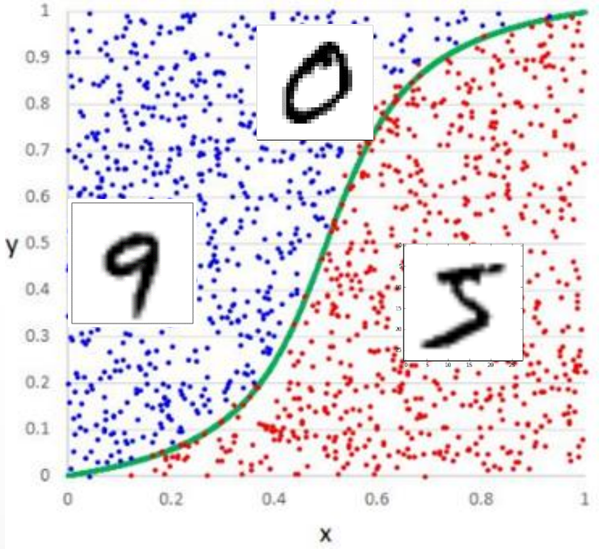
- Computing similarity of two 1M-vectors in minutes
  - Similarity of encrypted genome sequences (iDASH 2015)



More HE Speed Examples

- Computing similarity of two 1M-vectors in minutes
- 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

- Computing similarity of two 1M-vectors in minutes
- 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)

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??
```

ya think?

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

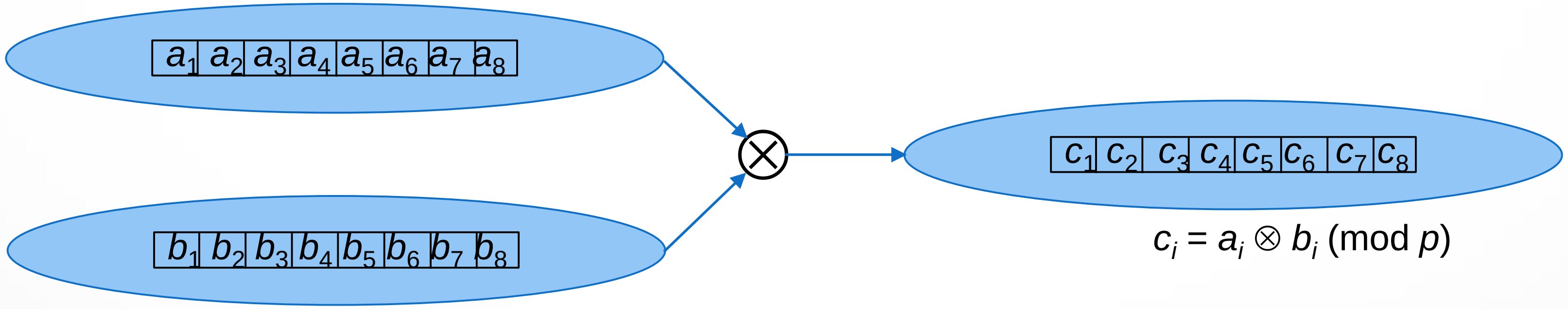
```
int main(int argc, char** argv)
{
  const char* addr = "127.0.0.1”;
  int port = 7766;

  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



$c_i = a_i \otimes b_i \pmod{p}$

- 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 this conference
    - *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

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

Summary: Advanced Cryptography is

Needed

- Can help prevent data abuse
- Still an under-utilized tool

Fast enough to be useful

Not "generally usable" yet

We are making some progress

Questions?

Thanks to many people for their input:
Eli Ben-Sasson, Alessandro Chiesa, Jonathan Katz,
Yehuda Lindell, Thomas Schneider, Elaine Shi,
Muthu Venkitasubaramaniam, Xiao Wang