COS433/Math 473: Cryptography

Mark Zhandry
Princeton University
Spring 2018
Secret Sharing
Vault should only open if both Alice and Bob are present.
Vault should only open if Alice, Bob, and Charlie are all present
Vault should only open if any two of Alice, Bob, and Charlie are present.
(Threshold) Secret Sharing

Syntax:
\[
\text{Share}(k,t,n) \; \text{outputs} \; (sh_1,\ldots,sh_n)
\]
\[
\text{Recon}(\; (sh_i)_{i \in S} \; ) \; \text{outputs} \; k'
\]

Correctness: \( \forall S \; \text{s.t.} \; |S| \geq t \)

If \((sh_i)_{i=1,\ldots,n} \leftarrow \text{Share}(k,t,n)\), then
\[
\Pr[\text{Recon}(\; (sh_i)_{i \in S} \; ) = k] = 1
\]
(Threshold) Secret Sharing

Security:

For any $S$, $|S| < t$, given $(sh_i)_{i \in S}$, should be impossible to recover $k$

$$(sh_i)_{i \in S} : (sh_i)_{i=1,\ldots,n} \leftarrow Share(k_0,t,n)$$

$$\approx$$

$$(sh_i)_{i \in S} : (sh_i)_{i=1,\ldots,n} \leftarrow Share(k_1,t,n)$$
**n-out-of-n** Secret Sharing

Share secret $k$ so that can only reconstruct $k$ if all $n$ users get together.

Ideas?
Shamir Secret Sharing

Let $p$ be a prime $> n$, $\geq \#(k)$

$\text{Share}(k,t,n)$:
- Choose a random polynomial $P$ of degree $t-1$ where $P(0) = k$
- $sh_i = P(i)$

$\text{Recon}\left( (sh_i)_{i \in S} \right)$: use shares to interpolate $P$, then evaluate on $0$
Shamir Secret Sharing

Correctness:
• \( t \) input/outputs (shares) are enough to interpolate a degree \( t-1 \) polynomial

Security:
• Given just \( t-1 \) inputs/outputs, \( P(0) \) is equally likely to be any value
Multiparty Computation
Multiparty Computation

\[ f(a, b, c) \]

\[ f(a, b, c) \]

\[ f(a, b, c) \]
Multiparty Computation

Observation 1: Shamir secret sharing is additively homomorphic:

Given shares $sh_1$ of $x_1$ and $sh_2$ of $x_2$, $r \times sh_1 + s \times sh_2$ is a share of $r \times x_1 + s \times x_2$

• $sh_1 = P_1(i)$, $sh_2 = P_2(i)$, so $r \times sh_1 + s \times sh_2 = (r \times P_1 + s \times P_2)(i)$
• $r \times P_1 + s \times P_2$ has same degree
MPC for linear $f$

- Secret share $a$
  - Alice
  - Bob
  - Charlie
MPC for linear $f$

Secret share $b$

a -> c

b

c

MPC for linear $f$

Secret share $c$
MPC for linear $f$

Locally compute shares of $f(a,b,c)$
MPC for linear $f$

Broadcast shares, then reconstruct

$a$  $b$  $c$
MPC for General $f$

Observation 2: Shamir Secret Sharing is sort of multiplicatively homomorphic

Given shares $sh_1$ of $x_1$ and $sh_2$ of $x_2$, $sh_1 \times sh_2$ is a share of $x_1 \times x_2$, but with a different threshold

- $sh_1 = P_1(i)$, $sh_2 = P_2(i)$, so $sh_1 \times sh_2 = (P_1 \times P_2)(i)$

- $P_1 \times P_2$ has degree $2d$

Idea: can do multiplications locally, and then some additional interaction to get degree back to $d$
MPC for General $f$

To maintain correctness, need threshold to stay at most $n$:

• But multiplying doubles threshold, so need $t \leq n/2$

• This means scheme broken if adversary corrupts $n/2$ users.

• Known to be optimal for “information-theoretic” MPC

Using crypto (e.g. one-way functions), can get threshold all the way up to $n$
MPC for Malicious Adversaries

So far, everything assumes players act honestly, and just want to learn each other’s inputs

But what if honest players deviate from protocol?

Idea: use ZK proofs to prove that you followed protocol without revealing your inputs
Cryptocurrency
Features of Physical Cash

Essentially anonymous

Hard to counterfeit

Easy to verify
Limitations of Physical Cash

Cannot be used online
• Instead, need to involve banks
• Banks see all transactions
• Merchants can also track you

Requires central government to issue
• Ok for most people, but maybe you don’t trust the government
Digital Cash

Currency is now 1s and 0s

Crypto can make digital currency easy to verify, hard to mint

Major challenge: prevent double spending (Also decentralizing minting process)
Solution: Public Ledger

Bank transfers $$ to Alice

Each bill has unique serial number
Solution: Public Ledger

<table>
<thead>
<tr>
<th>Bank transfers $$ to Alice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice transfers $$ to Bob</td>
</tr>
</tbody>
</table>
Solution: Public Ledger

Bank transfers $$ to Alice

Alice transfers $$ to Bob
Solution: Public Ledger

Bank maintain ledger?
• But then bank must be involved in every transaction
• How does bank prevent malicious Bob from claiming Alice transferred money to him?

Anonymity also lost, since all transactions public
Solution: Use Signatures

\[ \sigma_1 = \text{Sign}(\text{sk}_{\text{Bank}}, \text{"pk}_{\text{Bank}} \text{ transfers } \$$$ \text{ to } \text{pk}_A) \]
Solution: Use Signatures

\[
\sigma_2 = \text{Sign}(sk_A, \text{"pk}_A\text{ transfers }\$\text{ to pk}_B\text{"})
\]

pk<sub>Bank</sub> transfers $\$\$\$\$ to pk<sub>A</sub>, \(\sigma_1\)

pk<sub>A</sub> transfers $\$\$\$\$ to pk<sub>B</sub>, \(\sigma_2\)
Solution: Use Signatures

By using public key as identity, transactions not immediately traced to individual
• Though can still trace sequences of transactions

By signing, prevents Bob from claiming Alice gave him money when she didn’t
Decentralized Currency

Removing the bank is hard:

• How is ledger maintained?

• How to prevent ledger from being tampered with?

• Who mints new currency?

• How do we limit supply?
Proofs of Work

Prove that some amount of computation has been performed

Ex:
• Let $H$ be a hash function (modeled as a RO)
• An input $x$ such that $H(x) = 0^{t\ast\ast\ast\ast\ast\ast}$ is a “proof” that you computed approximately $2^t$ hashes
Proofs of Work and Cryptocurrency

Idea: currency is a proof of work

• Limits supply of money, so keeps inflation in check
• Now, anyone can mint new money
Blockchain

Immutable public ledger

Block:

<table>
<thead>
<tr>
<th>$\text{pk}_A$ transfers $$$ \text{ to pk}_B, \sigma_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_1$</td>
</tr>
<tr>
<td>$\tau_1$</td>
</tr>
</tbody>
</table>

Hashes to $0^*\text{*****}$
Blockchain

Immutable public ledger

Block:

\[ pk_E \text{ transfers } \$1 \text{ to } pk_F, \sigma_0 \]

\[ \begin{array}{c|c}
  h_0 & \tau_0 \\
\end{array} \]

\[ pk_A \text{ transfers } \$1 \text{ to } pk_B, \sigma_1 \]

\[ \begin{array}{c|c}
  h_1 & \tau_1 \\
\end{array} \]

\[ pk_C \text{ transfers } \$2 \text{ to } pk_D, \sigma_2 \]

\[ \begin{array}{c|c}
  h_2 & \tau_2 \\
\end{array} \]

Hashes to 0+****
Blockchain

By making each block a proof of work, hard to modify blockchain

So proofs of work used to:

• Mint new money
• Add transactions to blockchain

Why would anyone go through the effort of adding transactions to the blockchain?
Blockchain

Idea: combine minting and adding blocks

Block:

| $\text{pk}_A$ transfers $\$1$ to $\text{pk}_B$, $\sigma_1$ |
|---|---|
| $h_1$ | $\tau_1$ |

$\text{pk}_M$ mined $\$M$

Hashes to $0^{\uparrow\uparrow\uparrow\uparrow\uparrow}$
Double Spending

$pk_A$ transfers $\$1$ to $pk_B$, $\sigma_1$

$pk_A$ transfers $\$1$ to $pk_C$, $\sigma_2$

What if Alice now claims this is the blockchain?
Double Spending

To prevent double spending, everyone always uses the longest chain as the blockchain.

If Alice tries to double spend, she will need to create a separate chain that is as long as the main chain:
- As long as she has $<50\%$ of the computing power of the mining power, it will not be possible.
Beyond COS 433
Elliptic Curves

\[ y^2 = a x^3 + b x^2 + c x + d \]

1. \[ y^2 = x^3 - x \]
2. \[ y^2 = x^3 - x + 1 \]
Group Law on ECs
ECs for Crypto

Consider EC over finite field

Set of solutions form a group

Dlog in group appears hard

• Given $aP = (P+P+...+P)$, find $a$
• Can use in crypto applications
Bilinear Maps

On some Elliptic curves, additional useful structure

Map \( e: G \times G \rightarrow G_2 \)
• \( e(g^a, g^b) = e(g, g)^{ab} \)
3-party Key Exchange

\[ \text{Shared key} = e(g,g)^{abc} \]
Bilinear Maps

Extremely powerful tool, many applications beyond those in COS 433
• 3 party *non-interactive* key exchange
• Identity-based encryption
• Broadcast encryption
Multilinear Maps

Map $e: G^n \rightarrow G_2$
• $e(g^a, g^b, \ldots) = e(g, g, \ldots)^{ab\ldots}$

Many more applications that bilinear maps:
• n+1 party non-interactive key exchange
• Obfuscation
• ...

Unfortunately, don’t know how to construct from elliptic curves
• Recently, constructions based on other math
Lattices
Lattices

Basis:
- Linearly independent
- Every point in lattice is integer combination of basis vectors
Lattices

Hard problems in lattices:
• Given a basis, find the shortest vector in the lattice
• Given a basis an a point not in the lattice, find the closest lattice point

Can base much crypto on approximation versions of these problems
• Basically everything we’ve seen in COS433, then some
Fully Homomorphic Encryption

In homework, you saw additively/multiplicatively homomorphic encryption:

\[
\text{Enc}(pk, x) + \text{Enc}(pk, y) = \text{Enc}(pk, x+y)
\]

OR

\[
\text{Enc}(pk, x) \times \text{Enc}(pk, y) = \text{Enc}(pk, x \times y)
\]

What if you could do both simultaneously?

• Arbitrary computations on encrypted data
Delegation

Doesn’t want Amazon to learn sensitive data
Delegation

Now, Alice wants Amazon to run expensive computation on data
Delegation

Homomorphic evaluation
Quantum Computing

Computers that take advantage of quantum physics

Turns out, good at solving certain problems
• Dlog in any group ($\mathbb{Z}_p^*$, ECs)
• Factor integers

Also can speed up brute force search:
• Invert OWF in time $2^{n/2}$
• Find collisions in time $2^{n/3}$
Quantum Computing

To protect against quantum attacks, must:
• Must increase key size
  • 256 bits for one-way functions
  • 384 bits for collision resistance
• Must not use DDH/Factoring
  • Lattices instead

Quantum computers still at least a few years away, but coming
Final Exam Details

Slightly longer than homework, but slightly shorter questions

Pick any **48 hour** period during the dates **May 16 – May 21**
• Will send out more comprehensive instructions

Example exam on course webpage
Reminders

HW 8 Due May 8

Project 3 Due Dean’s date

No more Monday OH – Mark’s OH by appointment