CPSC 367: Cryptography and Security

Michael J. Fischer

Lecture 20
April 9, 2019

Zero Knowledge Interactive Proofs (ZKIP)

- ZKIP for graph isomorphism
- Abstraction from two ZKIP examples

Information Splitting

Multishare Secret Splitting

- Threshold secret splitting scheme
- Secret splitting with dishonest parties

Zero Knowledge Interactive Proofs (ZKIP)

Zero knowledge interactive proofs (continued)

We have seen two examples of zero knowledge interactive proofs:

- Secret cave protocol.
- Simplified Feige-Fiat-Shamir authentication protocol.

We now look at ZKIP's in greater detail.

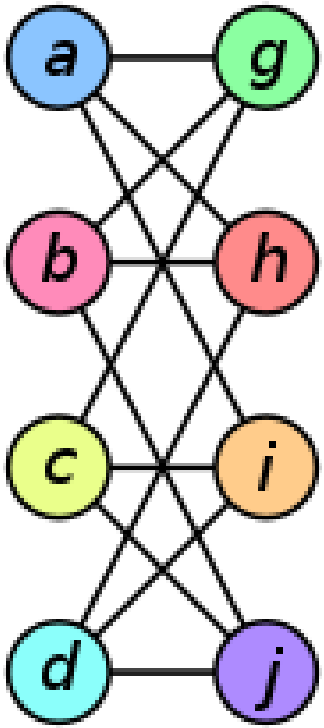
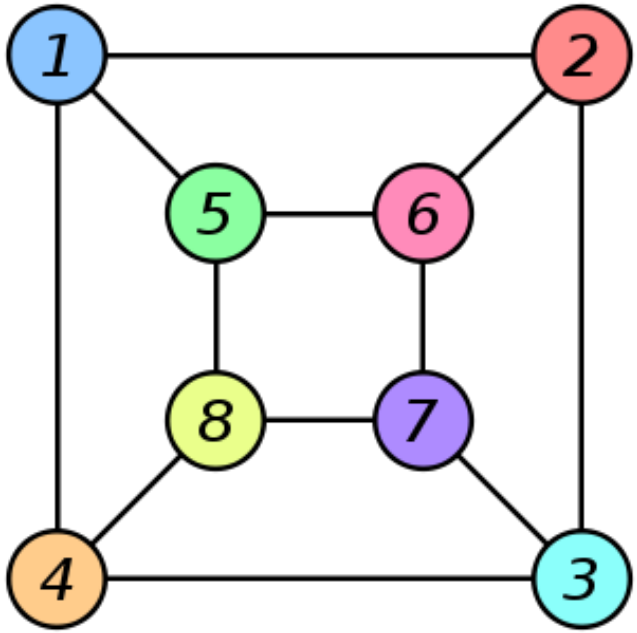
Graph isomorphism problem

Two undirected graphs G and H are said to be *isomorphic* if there exists a bijection π from vertices of G to vertices of H that preserves edges.

That is, $\{x, y\}$ is an edge of G iff $\{\pi(x), \pi(y)\}$ is an edge of H.

The *graph isomorphism problem* is, given graphs G and H, to determine whether or not G and H are isomorphic.

Graph Isomorphism

Graph G	Graph H	Isomorphism π
		$\pi(a) = 1$ $\pi(b) = 6$ $\pi(c) = 8$ $\pi(d) = 3$ $\pi(g) = 5$ $\pi(h) = 2$ $\pi(i) = 4$ $\pi(j) = 7$

From Wikipedia, https://en.wikipedia.org/wiki/Graph_isomorphism

Testing versus finding

No polynomial time algorithm is known for **testing** if two graphs G and H are isomorphic, but this problem is also not known to be NP-hard.

It follows that there is no known polynomial time algorithm for **finding** the isomorphism π given two isomorphic graphs G and H.
Why?
If there were such a polynomial time algorithm, we could test isomorphism as follows:

> *Given G and H, use A to find an isomorphism π from G to H. If A succeeds, answer "yes"; otherwise answer "no".*

Complexity of graph isomorphism

László Babai claims that the graph isomorphism problem is in *quasipolynomial time*, that is, time of the form

$$2^{O(\log(n)^c)}$$

for some constant c. This is a huge improvement over the best prior result. This result is still unverified (see László Babai Graph Isomorphism).

A zero-knowledge proof for isomorphism

Suppose G_0 and G_1 are public graphs, and Alice knows an isomorphism $\pi : G_0 \to G_1$.

Using a zero-knowledge proof, Alice can prove to Bob that she knows π *without revealing any information about* π. In particular, she convinces Bob that the graphs really are isomorphic.

However, the proof is *non-transferrable*, so Bob cannot turn around and convince Carol of that fact.

Interactive proof of graph isomorphism

	Alice		Bob
1.	Simultaneously choose a random isomorphic copy H of G_0 and an isomorphism $\tau : G_0 \to H$.		
		$\xrightarrow{H}$	
2.		$\xleftarrow{b}$	Choose random $b \in \{0, 1\}$.
3.	If $b = 0$, let $\sigma = \tau$. If $b = 1$, let $\sigma = \tau \circ \pi^{-1}$.	$\xrightarrow{\sigma}$	Check $\sigma(G_b) = H$.

Validity of isomorphism IP

The protocol is similar to the simplified Feige-Fiat-Shamir protocol

If both Alice and Bob follow this protocol, Bob's check always succeeds.

- When $b = 0$, Alice send τ in step 3, and Bob checks that τ is an isomorphism from G_0 to H.
- When $b = 1$, the function σ that Alice computes is an isomorphism from G_1 to H. This is because π^{-1} is an isomorphism from G_1 to G_0 and τ is an isomorphism from G_0 to H. Composing them gives an isomorphism from G_1 to H, so again Bob's check succeeds.

Isomorphism IP is zero knowlege

The protocol is zero knowledge (at least informally) because all Bob learns is a random isomorphic copy H of either G_0 or G_1 and the corresponding isomorphism.

He could have obtained this information by himself without Alice's help.

What convinces him that Alice really knows π is that in order to repeatedly pass his checks, the graph H of step 1 must be isomorphic to *both* G_0 and G_1.

Moreover, Alice knows isomorphisms $\sigma_0 : G_0 \to H$ and $\sigma_1 : G_1 \to H$ since she can produce them upon demand.

Hence, she also knows an isomorphism π from G_0 to G_1, since $\sigma_1^{-1} \circ \sigma_0$ is such a function.

FFS authentication and isomorphism IP

We have seen two examples of zero knowledge interactive proofs of knowledge of a secret.

In the simplified Feige-Fiat-Shamir authentication scheme, Alice's secret is a square root of v^{-1}.

In the graph isomorphism protocol, her secret is the isomorphism π.

In both cases, the protocol has the form that Alice sends Bob a "commitment" string x, Bob sends a query bit b, and Alice replies with a response y_b.

Bob then checks the triple (x, b, y_b) for validity.

FFS/Isomorphism comparison (continued)

In both protocols, neither triple $(x, 0, y_0)$ nor $(x, 1, y_1)$ alone give any information about Alice's secret, but y_0 and y_1 can be combined to reveal her secret.

In the FFS protocol, $y_1 y_0^{-1} \bmod n$ is a square root of v^{-1}.
(Note: Since v^{-1} has four square roots, the revealed square root might not be the same as Alice's secret, but it is equally valid as a means of impersonating Alice.)

In the graph isomorphism protocol, $y_1^{-1} \circ y_0$ is an isomorphism mapping G_0 to G_1.

Another viewpoint

One way to view zero knowledge protocols is that Alice splits her secret into two parts, y_0 and y_1.

By randomization, Alice is able to convince Bob that she really has (or could produce on demand) both parts, but in doing so, she is only forced to reveal one of them.

Each part by itself is statistically independent of the secret and hence gives Bob no information about the secret.

Together, they can be used to recover the secret.

Other materials on zero knowledge

Here are some links to other interesting materials on zero knowledge.

- How to explain zero-knowledge protocols to your children gives a different version of the Secret Cave protocol along with other stories illustrating other aspects of zero knowledge, such as non-transferrability of proof.
- Using a zero-knowledge protocol to prove you can solve a sudoku is a video of a Skype session in which Katie Steckles proves her sudoku-solving ability to Christian Perfect.
- Cryptographic and Physical Zero-Knowledge Proof Systems for Solutions of Sudoku Puzzles is the paper describing the sudoku solution protocol upon which the video above is based.

Information Splitting

Two-key locks

There are many situations in which one wants to grant access to a resource only if a sufficiently large group of agents cooperate.

For example, the office safe of a supermarket might require both the manager's key and the armored car driver's key in order to be opened.

This protects the store against a dishonest manager or armored car driver, and it also prevents an armed robber from coercing the manager into opening the safe.

A similar 2-key system is used for safe deposit boxes in banks.

The Big Picture

Much of cryptography is concerned with splitting a piece of information s into a collection of *shares* $s_1, \ldots, s_r$.

Certain subsets of shares allow s to be easily recovered; other subsets are insufficient to allow any useful information about s to be easily obtained.

In the simplest form, s is split into two shares a and b. Neither share alone gives useful information about s, but together they reveal s.

One-time pad cryptosystem

The one-time pad cryptosystem in Lecture 3 can be viewed as an instance of secret splitting.

Here, Alice's secret is her message m.

The two shares are the ciphertext c and the key k.

Neither by themselves gives any information about m, but together they reveal $m = k \oplus c$.

Two-part secret splitting

We might like to achieve the same properties for cryptographic keys or other secrets.

Let k be the key for a symmetric cryptosystem. One might wish to split k into two *shares* k_1 and k_2 so that by themselves, neither k_1 nor k_2 by itself reveals any information about k, but when suitably combined, k can be recovered.

A simple way to do this is to choose k_1 uniformly at random and then let $k_2 = k \oplus k_1$.

Both k_1 and k_2 are uniformly distributed over the key space and hence give no information about k.

However, combined with XOR, they reveal k, since $k = k_1 \oplus k_2$.

Unequal length shares

In some kinds of secret splitting, the two shares are not the same length.

For example, in AES, the secret message m is broken into a short key k and a long ciphertext c, where $m = D_k(c)$.

Multishare Secret Splitting

Motivation for multishare secret splitting

Secret splitting generalizes to more than two shares.

Imagine a large company that restricts access to important company secrets to only its five top executives, say the president, vice-president, treasurer, CEO, and CIO.

They don't want any executive to be able to access the data alone since they are concerned that an executive might be blackmailed into giving confidential data to a competitor.

Motivation (cont.)

On the other hand, they also don't want to require that all five executives get together to access their data because

- this would be cumbersome;
- they worry about the death or incapacitation of any single individual.

They decide as a compromise that any three of them should be able to access the secret data, but one or two of them operating alone should not have access.

(τ, k) threshold secret spitting scheme

A (τ, k) *threshold secret splitting scheme* splits a secret s into *shares* $s_1, \ldots, s_k$.

Any subset of τ or more shares allows s to be recovered, but no subset of shares of size less than τ gives any information about s.

The executives of the previous example want a $(3, 5)$ threshold secret splitting scheme:

> *The secret key is to be split into 5 shares, any 3 of which allow the secret to be recovered.*

A threshold scheme based on polynomials

Shamir proposed a threshold scheme based on polynomials.

A *polynomial of degree d* is an expression

$$f(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_d x^d,$$

where $a_d \neq 0$.

The numbers $a_0, \ldots, a_d$ are called the *coefficients* of f.

A polynomial can be simultaneously regarded as a function and as an object determined by its vector of coefficients.

Interpolation

Interpolation is the process of finding a polynomial that goes through a given set of points.

Fact

Let $(x_1, y_1), \ldots, (x_k, y_k)$ be points, where all of the x_i's are distinct. There is a unique polynomial $f(x)$ of degree at most $k-1$ that passes through all k points, that is, for which $f(x_i) = y_i$ $(1 \leq i \leq k)$.

f can be found using *Lagrangian interpolation*. This statement generalizes the familiar statement from high school geometry that two points determine a line.

Lagrangian interpolation method

One way to understand Lagrangian interpolation is to consider the polynomial

$$\delta_i(x) = \frac{(x - x_1)(x - x_2)\dots(x - x_{i-1}) \cdot (x - x_{i+1})\dots(x - x_k)}{(x_i - x_1)(x_i - x_2)\dots(x_i - x_{i-1}) \cdot (x_i - x_{i+1})\dots(x_i - x_k)}$$

Although this looks at first like a rational function, it is actually just a polynomial in x since the denominator contains only the x-values of the given points and not the variable x.

$\delta_i(x)$ has the easily-checked property that $\delta_i(x_i) = 1$, and $\delta_i(x_j) = 0$ for $j \neq i$.

Lagrangian interpolation method (cont.)

Using $\delta_i(x)$, the polynomial

$$p(x) = \sum_{i=1}^{k} y_i \, \delta_i(x)$$

is the desired interpolating polynomial, since $p(x_i) = y_i$ for $i = 1, \ldots, k$.

To actually find the coefficients a_i of $p(x) = \sum_{i=0}^{k} a_i x^i$, it is necessary to expand $p(x)$ by multiplying out the factors and collect like terms.

Interpolation over finite fields

Interpolation also works over finite fields such as $\mathbf{Z}_p$ for prime p.

It is still true that any k points with distinct x coordinates determine a unique polynomial of degree at most $k-1$ over $\mathbf{Z}_p$.

Of course, we must have $k \leq p$ since $\mathbf{Z}_p$ has only p distinct coordinate values in all.

Shamir's secret splitting scheme

Here's how Shamir's (τ, k) secret splitting scheme works.

Let Alice (also called the *dealer*) have secret s.

She first chooses a prime $p > k$ and announces it to all players.

Constructing the polynomial

She next constructs a polynomial

$$f = a_0 + a_1 x + a_2 x^2 \ldots a_{\tau-1} x^{\tau-1}$$

of degree at most $\tau - 1$ as follows:

- She sets $a_0 = s$ (the secret).
- She chooses $a_1, \ldots, a_{\tau-1} \in Z_p$ at random.

Constructing the shares

She constructs the k shares as follows:

- She chooses $x_i = i$. $(1 \leq i \leq k)$
- She chooses $y_i = f(i)$. $(1 \leq i \leq k)$[1]
- Share $s_i = (x_i, y_i) = (i, f(i))$.

[1] $f(i)$ is the result of evaluating the polynomial f at the value $x = i$. All arithmetic is over the field $\mathbf{Z}_p$, so we omit explicit mention of mod p.

Recovering the secret

Theorem

s can be reconstructed from any set T of τ or more shares.

Proof.

Suppose $s_{i_1}, \ldots, s_{i_\tau}$ are τ distinct shares in T.

By interpolation, there is a unique polynomial $g(x)$ of degree $d \leq \tau - 1$ that passes through these shares.

By construction of the shares, $f(x)$ also passes through these same shares; hence $g = f$ as polynomials.

In particular, $g(0) = f(0) = s$ is the secret. □

Protection from unauthorized disclosure

Theorem

For any set T' of fewer than τ shares and any possible secret $\hat{s}$, there is a polynomial $\hat{f}$ that interprets those shares and reveals $\hat{s}$.

Proof.

Let $T' = \{s_{i_1}, \ldots, s_{i_r}\}$ be a set of $r < \tau$ shares.

In particular, for each $\hat{s} \in \mathbf{Z}_p$, there is a polynomial $g_{\hat{s}}$ that interpolates the shares in $T' \cup \{(0, \hat{s})\}$.

Each of these polynomials passes through all of the shares in T', so each is a plausible candidate for f. Moreover, $g_{\hat{s}}(0) = \hat{s}$, so each $\hat{s}$ is a plausible candidate for the secret s. □

No information about secret

One can show further that the number of polynomials that interpolate $T' \cup \{(0, \hat{s})\}$ is the same for each $\hat{s} \in \mathbf{Z}_p$, so each possible candidate $\hat{s}$ is equally likely to be s.

Hence, the shares in T' give no information at all about s.

Secret splitting with semi-honest parties

Shamir's scheme is an example of a protocol that works assuming *semi-honest* parties.

These are players that follow the protocol but additionally may collude in an attempt to discover secret information.

We just saw that no coalition of fewer than τ players could learn anything about the dealer's secret, even if they pooled all of their shares.

Secret splitting with dishonest dealer

In practice, either the dealer or some of the players (or both) may be dishonest and fail to follow the protocol. The honest players would like some guarantees even in such situations.

A dishonest dealer can always lie about the true value of her secret. Even so, the honest players want assurance that their shares do in fact encode a unique secret, that is, the same secret s is recovered from every set of τ shares.

Failure of Shamir's scheme with dishonest dealer

Shamir's (τ, k) threshold scheme assumes that all k shares lie on a single polynomial of degree at most $\tau - 1$.

This might not hold if the dealer is dishonest and gives bad shares to some of the players.

The players have no way to discover that they have bad shares until later when they try to reconstruct s, and maybe not even then.

Verifiable secret sharing

In *verifiable secret sharing*, the sharing phase is an active protocol involving the dealer and all of the players.

At the end of this phase, either the dealer is exposed as being dishonest, or all of the players end up with shares that are consistent with a single secret.

Needless to say, protocols for verifiable secret sharing are quite complicated.

Dishonest players

Dishonest players present another kind of problem. These are players that fail to follow the protocol. During the reconstruction phase, they may fail to supply their share, or they may present a (possibly different) corrupted share to each other player.

With Shamir's scheme, a share that just disappears does not prevent the secret from being reconstructed, as long as enough valid shares remain.

But a player who lies about his share during the reconstruction phase can cause other players to reconstruct incorrect values for the secret.

Fault-tolerance in secret sharing protocols

A *fault-tolerant secret sharing scheme* should allow the secret to be correctly reconstructed, even in the face of a certain number of corrupted shares.

Of course, it may be desirable to have schemes that can tolerate dishonesty in both dealer and a limited number of players.

The interested reader is encouraged to explore the extensive literature on this subject.