# Introduction to Modern Cryptography
# Lecture 10[2]

December 17, 2013

Instructor: Benny Chor

Teaching Assistant: Nir Bitansky

School of Computer Science
Tel-Aviv University

Fall Semester, 2013–14, 15:00–18:00
Dan David 201

Course site: http://tau-crypto-f13.wikidot.com/

[2]some slides courtesy of Benny Appelbaum

## Lecture/Recitation 9: Reminder

- Secret sharing.

## Lecture 10: Plan

- Secret sharing: more details, and a proof.

- Interactive proof systems.
- Zero knowledge proof systems

Much of this class will be presented using the decade old technique of whiteboard and markers.

## $t$-out-of-$n$ Secret Sharing: Setting

- A value $S \in \mathcal{U}$ is the secret key allowing access or activation of an extremely critical and sensitive device.
- A "trusted dealer" holds $S$, but does not wish to activate the device right now.
- This dealer wants to delegate the secret to $n$ parties.
- The parties can only be partially trusted: We seek a mechanism that will enable any $t$ parties to reconstruct $S$, but any subset of $t-1$ parties (or less) cannot get any partial information on $S$.
- Computational requirements: Reconstruction should be efficient (as a function of length of $S$ and of number of parties $n$).
- Like always, we want impossibility of achieving any partial information by $t-1$ parties to be information theoretical, not computational.
- But exceptionally, here we can achieve it.

## $t$-out-of-$n$ Secret Sharing

A simple solution is to construct $\binom{n}{t}$ independent $t$-out-of-$t$ secret sharing scheme, and share the secret $S$ in each.

This works, but has a large overhead, which becomes exponential in $n$ as $t$ grows (think of $t = n/2$).

A far more efficient scheme, based on polynomial interpolation, was given by Adi Shamir. We describe it next.

## Shamir's $t$-out-of-$n$ Secret Sharing: Intuition

Preliminaries: Without loss of generality, the size of the secrets' universe satisfies $n + 1 \leq |U|$, and furthermore $U = Z_p$ for some prime number $p$ (we can always extend $U$, and simply not use some of the values in $Z_p$).

Intuition: Suppose we have a degree one univariate polynomial $f[x] = ax + b$ over $Z_p$, and we give each participant $i$ $(i = 1, \ldots, n)$ the value $f(i)$.

Question 1: What does a single participant, $i$, learn about $f(0) = b$?

Answer 1: Nothing!

Question 2: What can two participants, $i, j$ $(i \neq j)$ learn about $f(0) = b$?

Answer 2: Everything! Having $f(i) = ai + b$ and $f(j) = aj + b$, they can solve two linear equations in two variables ($a$ and $b$) and recover both $a, b$. In fact what they do is polynomial interpolation, in this case of a degree one polynomial.

## A (Slight) Detour: Lagrange Polynomial Interpolation

We are given a set of $t$ pairs $(x_1, y_1), \ldots, (x_t, y_t)$. Furthermore, we are told there is a univariate, degree $t-1$ polynomial, $f[x]$, satisfying $f(x_i) = y_i$ $(i = 1, \ldots, t)$.

How can we find this polynomial (namely find its $t$ coefficients $f[x] = a_{t-1}x^{t-1} + \ldots + a_1 x + a_0$)?

Define

$$f_1[x] = y_1 \cdot \frac{x - x_2}{x_1 - x_2} \cdot \frac{x - x_3}{x_1 - x_3} \cdots \frac{x - x_t}{x_1 - x_t} \; .$$

Then $f_1[x]$ is a degree $t-1$ polynomial, satisfying $f(x_1) = y_1$, and for all other $i \neq 1$, $f(x_i) = 0$. (Note that all terms $x_1 - x_j$ in the denominator are non-zero.)

We can define $f_2[x], \ldots, f_t[x]$ analogously. Then the desired degree $t-1$ polynomial is

$$f[x] = f_1[x] + f_2[x] + \ldots + f_t[x] \; .$$

## Lagrange Polynomial Interpolation: Uniqueness

We can define $f_2[x], \ldots, f_t[x]$ analogously. Then the desired degree $t-1$ polynomial is

$$f[x] = f_1[x] + f_2[x], \ldots, f_t[x] \ .$$

Note that there is a unique $f[x]$ with these properties. For suppose $g[x]$ also satisfies these properties. Then $f[x] - g[x]$ is a degree $t-1$ polynomial with $t$ different roots. So it must be the zero polynomial, thus $f[x] = g[x]$.

## Lagrange Polynomial Interpolation: Linearity

Let $x_1, x_2, \ldots, x_t$ and $x_0$ be $t+1$ different points in $Z_p$.
Let $y_1, y_2, \ldots, y_t$ be any $t$ points in $Z_p$.
We know there is a unique polynomial of degree $t-1$ (or less) such that $f(x_i) = y_i$, $i = 1, 2, \ldots, t$.

The value $f(x_0)$ (“the secret”) can be expressed as a linear combination of $f(x_1) = y_1, f(x_2) = y_2, \ldots, f(x_t) = y_t$ (“the shares”), with coefficients that depend on $x_1, x_2, \ldots, x_t$ alone (but not on the shares).

## Lagrange Polynomial Interpolation in Sage

Good old Sage naturally supports Lagrange interpolation. We just got to be a bit careful so it understands the numbers we input are $Z_p$ elements, rather than integers (which are the default).

```
F = GF(19)
R = PolynomialRing(F, x)
g=R.lagrange_polynomial([(F(0),F(4)),(F(2),F(12)),(F(6),F(6))])
       # F(b) is the conversion of integer b to a GF(19) element
       # so g(0)=4, g(2)=12, g(6)=6
g
```

```
>    7*x^2 + 9*x + 4
```

## $t$-out-of-$n$ Secret Sharing: Important Clarification

The dealer defines a degree $t-1$ polynomial whose free term equals the secret: $f[x] = r_{t-1}x^{t-1} + \ldots + r_1 x + S$.

The leading coefficient of the polynomial can be $0$. So the polynomial is of degree at most $t-1$.

## Shamir $t$-out-of-$n$ Secret Sharing: Construction

- Let $S \in Z_p$ be the secret. We take $p$ satisfying $p > n + 1$. If the domain of secrets is smaller, some values in $Z_p$ will just not be used.
- The dealer choses at random $t - 1$ values $r_{t-1}, \ldots, r_1$ uniformly and independently in the domain $Z_p$.
- Dealer defines a degree $t - 1$ polynomial whose free term equals the secret: $f[x] = r_{t-1}x^{t-1} + \ldots + r_1 x + S$.
- Shares $s_1, s_2, \ldots, s_n$ of players $1, \ldots, n$ are values of $f[x]$ at $n$ corresponding points,
  $s_1 = f(1), \ldots, s_{n-1} = f(n-1), s_n = f(n)$.

We remark that we allow the leading coefficient (and additional coefficients) to be zero. Each coefficient other than the free term is chosen at random and independently over $Z_p$.

## Shamir $t$-out-of-$n$ Secret Sharing: Construction

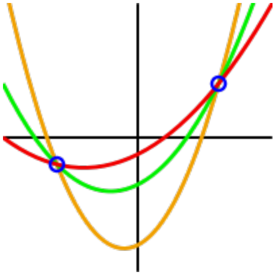
- Dealer defines a degree $t-1$ polynomial whose free term equals the secret: $f[x] = r_{t-1}x^{t-1} + \ldots + r_1 x + S$.
- Shares $s_1, s_2, \ldots, s_n$ of players $1, \ldots, n$ are values of $f[x]$ at $n$ corresponding points,
  $s_1 = f(1), \ldots, s_{n-1} = f(n-1), s_n = f(n)$.

- Reconstruction: Any set of $t$ players (or more) apply Langrange interpolation formula to their shares, find the coefficients of the unique degree $t-1$ polynomial $f[x] = r_{t-1}x^{t-1} + \ldots + r_1 x + S$. The free term of this polynomial, $S$, is the desired secret.
- Note that no matter which $t$ (or more) shares are used, reconstruction yields the same polynomial (and hence, secret).

## Shamir $n$-out-of-$n$ Secret Sharing: Secrecy

- Dealer defines a degree $t-1$ polynomial whose free term equals the secret: $f[x] = r_{t-1}x^{t-1} + \ldots + r_1 x + S$.
- Shares $s_1, s_2, \ldots, s_n$ of players $1, \ldots, n$ are values of $f[x]$ at $n$ corresponding points,
  $s_1 = f(1), \ldots, s_{n-1} = f(n-1), s_n = f(n)$.

- Secrecy: Should show that any set of $t-1$ players (or less) learns nothing about the secret.


(image from Wikipedia)

## $t$-out-of-$n$ Secret Sharing: Proof of Secrecy

- The secret $S$ and of a set of $t-1$ pairs $(x_1, y_1), \ldots, (x_{t-1}, y_{t-1})$ determine uniquely a polynomial $f[x] = r_{t-1}x^{t-1} + \ldots + r_1 x + S$.
- The correspondence is one-to-one and onto – there are $p^{t-1}$ such polynomials and exactly the same number of pairs $(x_1, y_1), \ldots, (x_t, y_t)$ (once the $t-1$ participants $x_1, x_2, \ldots, x_t$ are fixed).
- Thus for any given secret, any set of $t-1$ participants, and any $t-1$ corresponding shares, there is exactly one polynomial giving rise to these share
- So the probability that these shares will be distributed is exactly $1/p^{t-1}$ for any secret $S$.
- Smaller sets of participants are just projections of $t$ participants onto fewer ones, hence for each share are distributed uniformly ♠

## $t$-out-of-$n$ Secret Sharing: Some Remarks

- For the scheme to work, we need a finite field with at least $n+1$ elements (where $n$ is the number of parties). We worked in $Z_p$, but might as well work inside $GF(p^k)$, including the case where $p=2$, since Lagrange interpolation is applicable to any field.
- For the case where the domain of the secrets is larger than $n$, the size of secrets is the same as size of each share. Such scheme is termed ideal secret sharing scheme.
- The set of minimal subsets that can reconstruct the secret is termed the access structure of the scheme. We dealt with threshold access structure, but there are efficient schemes for other access structures as well.
- Most access structures are not known to possess efficient secret sharing schemes. Inefficient ones (large size of shares) are easy to come by. No "substantial" (exponential) lower bounds on shares sizes were shown so far.

## One More Remark: Why prime?

- For the interpolation to work, we had to assume that $p$ is prime (there are no “Zero divisors”)

## One More Remark: Why prime?

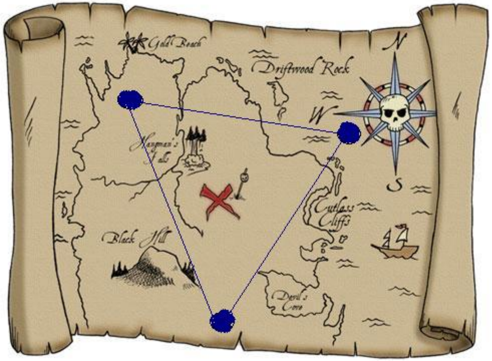
- For the interpolation to work, we had to assume that $p$ is prime (there are no "Zero divisors")
- In general, we can work over any finite field as long as it has $n + 1$ elements (where $n$ is the number of parties).

## One More Remark: Why prime?

- For the interpolation to work, we had to assume that $p$ is prime (there are no "Zero divisors")
- In general, we can work over any finite field as long as it has $n+1$ elements (where $n$ is the number of parties).
- For example, we can work over an extension field $GF(p^k)$, including the case where $p=2$, since Lagrange interpolation is applicable to any field.

## The Dying Pirate: Solution

The pirate draws 3 random points whose center (as a triangle) hits the location of the treasure. Each son gets the coordinates of a single point.
The info of a single point, or even a pair of points, reveals no information about the true location.



## Application 1: PGP key recovery mechanism

- Pretty Good Privacy[3] is a program to provide encrypted email, employing public key cryptography.
- It is risky to store the key on the PC of the user
- But the user cannot remember the key $k$
- So users remember a very long password $p$ and store only $H(p) \oplus k$ where $H$ is a hash function
- What if the user forgets the password?

[3] also termed Pretty Good Piracy in the early days of commercial public key encryption

## Application 1: PGP key recovery mechanism

Solution of PGP: Secret Sharing

- The key is shared in a $3$-out-of-$5$ scheme to $5$ shares $s_1, \ldots, s_5$.
- The user selects $5$ personal questions to which he knows the answers $a_1, \ldots, a_5$.
- The information $H(a_1) \oplus s_1$ , . . . , $H(a_5) \oplus s_5$ is stored in the computer (or actually, on a special purpose key reconstruction server)
- If the user remembers the answers to at least $3$ of the questions, he can reconstruct the key.

## And Now to Something Completely Different: Interactive Proof Systems

תֶּחֱזַקְנָה יְדֵי כָל-אַחֵינוּ הַמְחוֹנְנִים
עַפְרוֹת אַרְצֵנוּ בַּאֲשֶׁר הֵם שָׁם;
אַל יִפֹּל רוּחֲכֶם – עַלִּיזִים, מִתְרוֹנְנִים
בֹּאוּ שְׁכֶם אֶחָד לְעֶזְרַת הָעָם!

– חיים נחמן ביאליק, **ברכת עם**

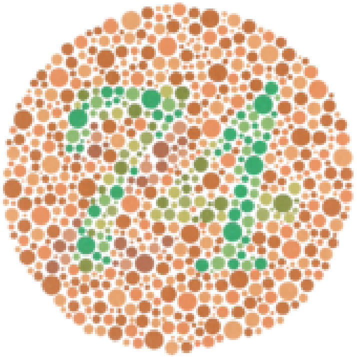
## Interactive and Zero Knowledge Proofs: Plan

- Traditional proof systems and their relations to the class NP.
- Interactive proof systems.
- Will show an interactive proof for graph non isomorphism (a problem in coNP that is not known to be in NP).
- Zero knowledge interactive proof systems.
- Will show a zero knowledge proof for graph isomorphism.
- Will show a zero knowledge proof for graph 3 colorability (an NP complete problem), using commitment schemes[4].

[4]which can be built using pseudo random generators, or, equivalently, one way functions.

## Interactive Proofs: The Two Apples Story

Benny wants to convince the color blind[5] Nir (staaaaaaam) that two apples with identical physical characteristic (shape, mass, etc.) have different colors.



[5] Deuteranopia (1% of males), Tritanopia (less than 1% of males), Protanomaly (1% of males), Deuteranomaly (most common – 6% of males).

## Interactive Proofs: The Two Apples Story

Benny wants to convince the color blind Nir (staaaaaaam) that two apples with identical physical characteristic (shape, mass, etc.) have different colors.

## Interactive Proofs

We will use a ppt presentation by Prof. Muli Safra (much fancier than those you are used to in our humble course).

## Zero Knowledge Proofs

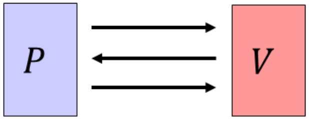
Can I convince you that some statement $\mathcal{S}$ holds, without giving you any hint about its proof?

- Not by authority or intimidation!
- By some proof system, possibly randomized, where
- If $\mathcal{S}$ is false, the probability that I convince you is smaller than $\varepsilon$ (a small parameter).
- If $\mathcal{S}$ is true, the probability that I convince you is larger than $1-\varepsilon$.
- The text of the conversations, assuming $\mathcal{S}$ is true, gives you nothing more – you could have generated very similar text on your own.

- We just saw a zero knowledge proof for graph 3 colorability.
- We will present (on board) a zero knowledge proof of graph isomorphism.
- We will present (on board) a zero knowledge proof of knowledge for an instance of the discrete logarithm problem.

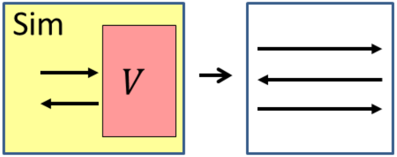
## Zero Knowledge: Ideal Scenario vs. Real Scenario (images by the same anonymous referee)

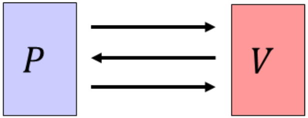
The real scenario: What does the verifier observe in an interactive proof system?



## Zero Knowledge: Ideal Scenario vs. Real Scenario, cont. (images by the same anonymous referee)

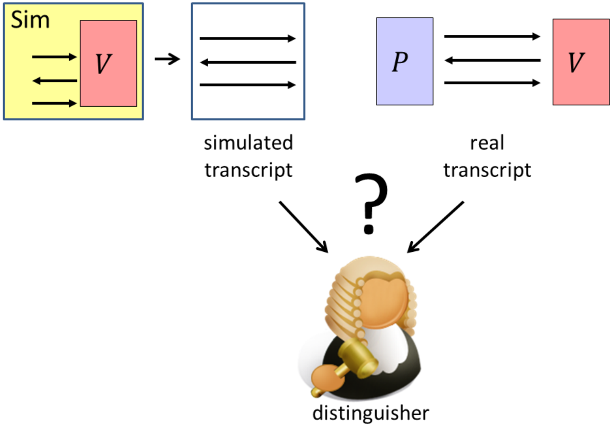
The ideal scenario: The simulator (which can call the verifier, run it, make it work, probe its cortex, etc.) produces a simulated transcript of the communication with the prover.





## Zero Knowledge: Ideal Scenario vs. Real Scenario, cont.

Actual communication vs. simulated communication:
A distinguisher cannot tell the two ensembles of communication apart.



Distinguisher is either computationally bound, or (stronger) unbounded (statistically indistinguishability).

## Honest Verifier vs. Malicious Verifier

Often honest verifier enables a simpler simulation strategy.

In schemes involving commitments, can use hiding to argue simulator succeeds with non negligible probability even for malicious adversary.

## Zero Knowledge Proof for Graph Isomorphism (sketch)

Setting: Two graphs $G_1, G_2$ such that $G_2 = \varphi(G_1)$.

The prover knows the homomorphism $\varphi$. Wishes to convince the verifier that the two are indeed isomorphic without revealing $\varphi$.

Repeat 100 times:

Prover: Choose at random a permutation $\psi$ of the nodes of $G_1$. Generate the induced graph, $H = \psi(G_1)$. Send $H = \psi(G_1)$ to verifier.
Verifier: Flip a coin. If heads, send $1$ to prover;
if tails, send $2$ to prover.
Prover: Send accordingly either a mapping (permutation) $\psi^{-1}$, mappling $H \mapsto G_1$, or $\varphi \circ \psi^{-1}$, mapping $H \mapsto G_2$ to verifier.
Verifier: Checks and accepts/rejects.
Note that in both cases, verifier sees a random permutation.

For the honest verifier, simulator can generate a transcript which is distributed exactly as the interaction above (without knowledge of the homomorphism).

## Zero Knowledge Proof of Knowledge of a Discrete Log (sketch)

Setting: Public prime $p$, multiplicative generator $g \in Z_p^*$. Bob publishes $g^x \pmod p$ and tells the post office to hand deliver packages addressed to Bob to anyone who can prove s/he knows $x$.

Bob is the prover. The post office is the verifier.

Repeat 100 times:

Prover: Choose at random $r, 0 \le r \le p-2$, send $g^r \pmod p$ to verifier.
Verifier: Flip a coin. If heads, send $y = g^r \pmod p$ to prover;
if tails, send $y = g^r g^x \pmod p$ to prover.
Prover: Check that indeed $y = g^r \pmod p$ or $y = g^r g^x \pmod p$. Send accordingly either $r$ or $r + x \pmod{p-1}$ to verifier.

For the honest verifier, simulator can generate a transcript which is distributed exactly as the interaction above (without knowledge of the discrete logarithm, $x$).

## Interactive Proofs: The Two Apples Story, cont.

Benny wants to convince the color blind Nir (staaaaaaam) that two apples with identical physical characteristic (shape, mass, etc.) have different colors.

Challenge: Modify the protocol so that it is also zero knowledge, namely poor, color blind Nir only learns the two apples have different colors, but not which color each apple is.