Introduction to Modern Cryptography

Benny Chor

Secret Sharing Schemes
Threshold Cryptography
Hard Core Bits
Coin Flipping Over the Phone

Lecture 10

Tel-Aviv University

20 December 2009; Typos fixed December 27

n-out-of-n Secret Sharing (revisited)

A not so hallucinated scenario:

- 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 all n parties to reconstruct S, but any subset of $n-1$ parties (or less) cannot get any partial information on S.
- Computational requirements: Reconstruction should be efficient (polynomial in the length of S and of number of parties n).
- Impossibility of obtaining any partial information by any subset with $n-1$ parties (or less) should be information theoretical – not based on any complexity assumptions.

Formal Definition: n-out-of-n Secret Sharing

- The trusted dealer holds $S \in \mathcal{U}$, picks a random value r by some distribution over a finite space, and generates the shares $s_1, s_2, \ldots, s_n$ by applying a known function $F(S, r) = (s_1, s_2, \ldots, s_n)$.
- Note: F is deterministic. All randomness comes from r.
- Party i receives share s_i.
- Reconstruction: There is a reconstruction function H such that applying it to the n shares always gives back the secret. Formally, for all S, r, $H(F(S, r)) = S$.

Formal Definition (cont.): n-out-of-n Secret Sharing

- Secrecy against coalitions: For any two possible values a, b of the secret $S \in \mathcal{U}$, and for any subset (coalition) of $n-1$ players,
- The distribution of the shares of the coalition members, given that $S = a$ is exactly equal to the distribution of the shares of the coalition members, given that $S = b$.
- Remark: This formulation is slightly different from the one given on board last week, but the two are equivalent.

n-out-of-n Secret Sharing: Construction

- Denote the size of the secrets universe, $|U|$, by m.
- Without loss of generality, the possible values of S are a subset of $Z_m = \{0, 1, \ldots, m-1\}$.
- The dealer choses at random $n-1$ values $r_1, \ldots, r_{n-1}$ uniformly and independently.
- Shares $s_1, s_2, \ldots, s_{n-1}$ of players $1, \ldots, n-1$ are $r_1, \ldots, r_{n-1}$, correspondingly.
- Share s_n of player n is $S - \left(\sum_{i=1}^{n-1} r_i\right) \mod m$.

n-out-of-n Secret Sharing: Validity

Claim: This is a valid n-out-of-n secret sharing scheme.

- Reconstruction by n players is easy:
  Simply express $S = \sum_{i=1}^{n} s_i \mod m$.
- Secrecy against coalitions: Will demonstrate for the coalition $\{2, \ldots, n-1, n\}$. Suppose the value of the secret is a. Let $c_2, \ldots, c_{n-1}, c_n$ be any $n-1$ values in Z_m.
- Claim: There is exactly one choice of $r_1, \ldots, r_{n-1}$ that yields these $n-1$ shares. Specifically, $r_2 = c_2, \ldots, r_{n-1} = c_{n-1}$, and $r_n = a - \left(\sum_{i=1}^{n-1} c_i\right) \mod m$.
- Since $r_1, \ldots, r_{n-1}$ are chosen uniformly and independently in Z_m, the probability of getting these $n-1$ shares, given the that the secret equals a, is exactly $1/m^{n-1}$.
- Obviously this probability (but not the values of the random r_i) is exactly the same if the secret equals b. ♠

t-out-of-n Secret Sharing

Another, not so hallucinated scenario:

- 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 before, impossibility of achieving any partial information by $t-1$ parties should be information theoretical, not computational.

t-out-of-n Secret Sharing

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

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

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

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

Preliminaries: Without loss of generality, the size of the secrets's 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

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

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 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')
f=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
f
```

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

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,
  $s_1 = f(1), \ldots, s_{n-1} = f(n-1), s_n = f(n)$.

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

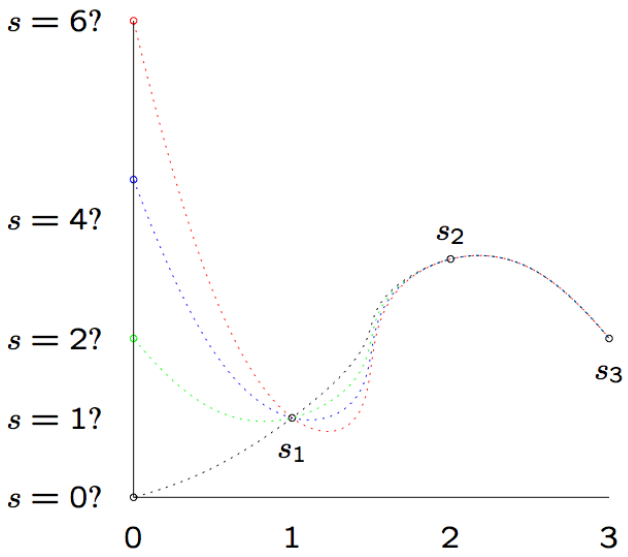
- Dealer defines a degree $t-1$ polynomial whose free term equals the secret: $f[x] = r_{t-1}x^{t-1} + \ldots + r_1 x + S$.
- Shares $s_1, s_2, \ldots, s_n$ of players $1, \ldots, n$ are values of $f[x]$ at n corresponding points,
  $s_1 = f(1), \ldots, s_{n-1} = f(n-1), s_n = f(n)$.

- 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.



(from http://www.tcs.hut.fi/Studies/T-79.159/2004/slides/L9.pdf)

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

Claim: This is a valid t-out-of-n secret sharing scheme.

- Secrecy against coalitions: Will demonstrate for the coalition $\{1, \ldots, t-1\}$ (but all others are the same). Suppose the value of the secret is a. Let $c_1, \ldots, c_{t-1}$ be any $t-1$ values in Z_p.
- Claim: There is exactly one choice of $r_1, \ldots, r_{t-1}$ that yields these $t-1$ shares. The $t-1$ shares $s_1 = f(1), \ldots, s_{t-1} = f(t-1)$ plus the secret a uniquely determine a a degree $t-1$ polynomial $f[x] = r_{t-1}x^{t-1} + \ldots + r_1 x + a$.
- Thus is the secret is a, the probability of getting the shares $s_1 = f(1), \ldots, s_{t-1} = f(t-1)$ is exactly $1/p^{t-1}$.
- Likewise, if is the secret is b, the probability of getting the shares $s_1 = f(1), \ldots, s_{t-1} = f(t-1)$ is exactly $1/p^{t-1}$.
- So the shares are distributed exactly the same given that the secret equals is a or is b. ♠

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.

Threshold Cryptography: RSA Signatures as Special Case

- Mixes threshold secret sharing and RSA signatures.
- Have n parties.
- Dealer publishes $N = pq$ and public "encryption key" e. Dealer knows factorization of $N = pq$ and secret "decryption key" d. Both are kept secret.
- Dealer wishes to distribute "pieces of d" to parties so that given a message M, any t of them can produce a valid signature of it, but any $t - 1$ or fewer cannot.
- Notice that simply doing secret sharing for d does not work. (why?)
- We'll do the n-out-of-n solution (easy) and maybe the t-out-of-n one on board.
- A crucial observation is the fact that the secret, $f(0)$, is a linear combination of the shares (t values, $f(i)$'s).

Hard Core Bits for One Way Functions

- Let $F : D \to D$ be a one-way, one-to-one function.
- Suppose $B : D \to \{0, 1\}$ is an efficiently computable predicate on elements of D.
- We say that B is a hard core bit for F if it is computationally hard, given $F(x)$, to determine $B(x)$.
- This task cannot be harder than inverting F.
- We want to formalize this notion.

Hard Core Bits for One Way Functions (2)

- We want to formalize this notion.
- Definition: We say that B is a hard core bit for F if the following holds:
- There is an efficient (possibly randomized) procedure $\mathcal{P}$, which gets the input $z = F(y)$,
- for any algorithm $\mathcal{A}(\cdot)$ that, on input $F(x)$, outputs $B(x)$,
- The procedure $\mathcal{P}$ can adaptively generate queries $z_1 = F(y_1)$, $z_2 = F(y_2), \ldots, z_k = F(y_k)$, and feed them to $\mathcal{A}(\cdot)$. The response to the queries are the bits $B(y_1), B(y_2), \ldots, B(y_k)$.
- The running time of $\mathcal{P}$ and the number of queries it makes, k, are both polynomial in $\log D$.
- Last but not least: The procedure $\mathcal{P}$ correctly inverts F, namely it outputs $y = F^{-1}(z)$. ♠

Hard Core Bits for One Way Functions: Intuition

If B is a hard core bit for F then the ability to efficiently infer $B(x)$ from $F(x)$ enables to invert F.

Any algorithm $\mathcal{A}(\cdot)$ that, on input $F(x)$, outputs $B(x)$, enables the inversion of $F(x)$ by employing the procedure $\mathcal{P}$.

Thus if F is indeed hard (we assumed it is one-way) then $B(x)$ cannot be computed from $F(x)$.

Such "hard core bit" can serve as the basis for coin tossing over the phone.

Coin Tossing Over the Phone

Two non trusting parties wish to toss coins over the phone.

- The parties do not deviate from the syntax of the protocol.
- They may try to cheat in different ways, provided the messages they send have the right format.
- For example, Alice could send the string 00000 whenever the protocol calls for five random bits. (There is no way Bob can convincingly argue she is cheating here.)
- For a different example, Alice could send a composite number where the protocol calls for a prime number. (This specific attempt is not very smart, since Bob will easily detect it.)
- For yet a third example, Alice could send a product of three numbers pqr, where the protocol calls for a product of two, pq. (It is not clear if Bob could detect this.)

(Anton Shigurov, the cleanest coin tosser around.)

Coin Tossing Over the Phone: General Scheme

- Alice chooses an instance of a one-way, one-to-one function $F : D \to D$.
- Let $B : D \to \{0, 1\}$ be a hard core predicate for F.
- Alice starts by sending a description of F and of B to Bob.
- Alice picks an $x \in D$, computes $y = F(x)$ and $b = B(x)$.
- She sends y to Bob. This is supposed to create a commitment to the value x.
- Bob send to Alice his guess for $B(x)$, namely a bit $c \in \{0, 1\}$.
- The bit c could either be the result of a coin flip or the outcome of some efficient algorithm applied by Bob in an attempt to guess $B(x)$.
- After receiving c, Alice sends x to Bob, who can now compute $b = B(x)$ on his own.
- If $c = b$ then Bob wins the coin toss. Else ($c \neq b$) Alice wins.

Specific Examples (Good and Bad)

- $D = Z_p^*$, $F(x) = g^x \bmod p$ where p is a prime number, and g is a primitive element in Z_p^*. Let $B(x) = \text{Half}_p(x)$ (namely 0 if $1 \leq x < (p-1)/2$, and 1 if $(p-1)/2 \leq x \leq p-1$). Good, provided g is a primitive element.
- $D = Z_p^*$, $F(x) = g^x \bmod p$ where p is a prime number, the factorization of $p-1$ is known, and g is a primitive element in Z_p^*. Let $B(x) =$ be the least significant bit of x. Bad.
- Let $D = Z_N^*$, $F(x) = x^e \bmod N$ where $N = pq$, both p and q are prime numbers, and e is relatively prime to $(p-1)(q-1)$. The numbers e and N are sent to Bob, but N's factorization is not given. Let $B(x) =$ the least significant bit of x. Good, provided e is relatively prime to $\phi(N)$.
- Let $F : D \to D$ be any one-way, one-to-one function, let n be the number of bits of elements in D. Define a new one to one, one way function $G : D \times D \to D \times D$ by $G(x, r) = (F(x), r)$. Define $B(x, r) = \sum_{i=1}^{n} x_i \cdot r_i \pmod 2$. Good – this is the generic hard core bit of Goldreich and Levin, 1989.

Coin Tossing Over the Phone and Hard Core Bits

Question: Does it suffice that B is a hard core bit for F in order to guarantee a fair coin toss? Or should the definition be strengthened?

In assignment 4 you are asked to think about this problem, and provide an educated answer.

Coin Tossing Over the Phone: Can Alice Cheat?

- Alice chooses an instance of a one-way, one-to-one function $F : D \to D$.
- Suppose Alice managed to choose a two-to-one function $F : D \to D$, but poor Bob could not tell the difference.
- Furthermore, suppose there are $x_0, x_1 \in D$ such that $F(x_0) = F(x_1)$ but $B(x_0) = 0, B(x_1) = 1$.
- In such case, Alice can win the coin toss with certainty.

- Bob is justified in being worried.
- But what can he do? He has neither time nor patience to go exhaustively over D and verify that F is one-to-one over it.

Coin Tossing Over the Phone: Can Alice Cheat?

- Bob is justified in being worried.
- But what can he do? He has neither time nor patience to go exhaustively over D and verify that F is one-to-one over it.
- In the specific examples we considered, there is a short proof that F is one-to-one.
- For example, $D = Z_N^*$, $F(x) = x^e \bmod N$ where $N = pq$, both p and q are prime numbers, and e is relatively prime to $\phi(N)$.
- If e is not relatively prime to $\phi(N)$, then $F(x) = x^e \bmod N$ is k-to-one for some $k \geq 2$.
- This means that for every $y \in Z_N^*$ in the range of F there are distinct $x_1, x_2, \ldots, x_k \in Z_N^*$ such that $F(x_1) = F(x_2) = \ldots = F(x_k) = y$.

Coin Tossing Over the Phone: Calming Down Bob

- Bob is justified in being worried.
- But what can he do? He has neither time nor patience to go exhaustively over D and verify that $F(x) = x^e \bmod N$ is one-to-one over it.
- If F is not one-to-one, then for every $y \in Z_N^*$ in the range of F there are distinct $z_1, z_2, \ldots, z_k \in Z_N^*$ such that $F(z_1) = F(z_2) = \ldots = F(z_k) = y$.
- Just after receiving the description of F, Bob picks one hundred elements in Z_N^* at random, $z_1, z_2, \ldots, z_{100}$. He computes $y_i = F(z_i)$ $(i = 1, \ldots, 100)$, and sends $y_1, y_2, \ldots, y_{100}$ to Alice.
- Alice computes $F^{-1}(y_1), F^{-1}(y_2), \ldots, F^{-1}(y_{100})$ and sends them to Bob. If for all of them $F^{-1}(y_i) = z_i$, then Bob is convinced that F is one-to-one. Otherwise, he knows that Alice is cheating, and quits the game.

Some Observations on Verifying F Is One-To-One

- It was crucial that Bob makes his queries before Alice committed to x by sending $F(x)$ (why?).
- The "protocol for convincing Bob" relied on two very specific properties of $F(x) = x^e \bmod N$.
- First, we used the fact that this is a trapdoor function and Alice should have the trapdoor information.
- Second, the k-to-one property is very specific to exponentiation modulo N.
- In general, we may want to convince a suspicious verifier that a certain claim holds, without giving away any additional information.
- Leads to zero knowledge proofs (Goldwasser Micaly Rackoff 1985).