Cryptography

and Network Security

Lecture 1

Our first encounter with secrecy:

Secret-Sharing

Secrecy

- Cryptography is all about "controlling access to information"
  - Access to learning and/or influencing information
- One of the aspects of access control is secrecy

A Game

- A "dealer" and two "players" Alice and Bob
- Dealer has a message m
- She wants to "share" it among the two players so that neither player by herself/himself learns <u>anything</u> about the message, but together they can find it
- Bad idea: If m is a two-bit message m_1m_2, give m_1 to Alice and m_2 to Bob
- Other ideas?

Sharing a bit

- To share a bit m, Dealer picks a uniformly random bit b and gives $a := m \oplus b$ to Alice and b to Bob
  - Bob learns nothing (b is a random bit)
  - Neither does Alice: for each possible value of m (0 or 1), a is a random bit (0 w.p. ½, 1 w.p. ½)

    $m = 0 \rightarrow (a,b) = (0,0)$ or $(1,1)$
    $m = 1 \rightarrow (a,b) = (1,0)$ or $(0,1)$

    - Her view is independent of the message
  - Together they can recover m as $a \oplus b$
- Multiple bits can be shared independently: e.g., $\underline{m_1 m_2} = \underline{a_1 a_2} \oplus \underline{b_1 b_2}$
- Note: any one share can be chosen before knowing the message [why?]

Secrecy

- Is the message m really <u>secret</u>?
- Alice or Bob can correctly find the bit m with probability ½, by randomly guessing
  - Worse, if they already know something about m, they can do better (Note: we didn't say m is uniformly random!)
- But they could have done this without obtaining the shares
  - The shares didn't leak any <u>additional</u> information to either party
- Typical crypto goal: **<u>preserving</u>** secrecy

Preserving Secrecy

- Goal: What Alice (or Bob) knows about the message after seeing her share is the same as what she knew a priori
- What she knows about the message a priori: a probability distribution over the message
  - For each message m, Pr[msg=m]
- What she knows after seeing her share (a.k.a. her view)
  - Say view is v. Then new distribution: Pr[msg=m | view=v]
- Formally: $\forall$ possible v, $\forall$ m, Pr[msg=m | view = v] = Pr[msg = m]
  - i.e., view is independent of message
    - $\forall$ v, $\forall$ m, Pr[view=v, msg=m] = Pr[view = v] · Pr[msg=m]

Preserving Secrecy

- What Alice (or Bob) knows about the message after seeing her share is the same as what she knew a priori:
  - $\forall$ possible v, $\forall$ m, Pr[msg=m | view = v] = Pr[msg = m]
  - $\forall$ v, $\forall$ m, Pr[view=v, msg=m] = Pr[view = v] · Pr[msg=m]
  - $\forall$ v, $\forall$ possible m, Pr[view = v | msg = m] = Pr[view = v]

Determined by the scheme

- $\forall$ v, $\forall$possible m, m', Pr[view=v | msg=m] = Pr[view=v | msg=m']
  - i.e., for all possible messages, the view is distributed the same way
  - The view could be **simulated** without knowing the message

Doesn't involve message distribution at all!

- Important: can't say Pr[msg=m | view=v] = Pr[msg=m' | view=v] (unless the prior is uniform)

Exercise

- Consider the following secret-sharing scheme
  - Message space = { buy, sell, wait }
  - buy $\rightarrow$ (00,00), (01,01), (10,10) or (11,11) w/ prob 1/4 each
  - sell $\rightarrow$ (00,01), (01,00), (10,11) or (11,10) w/ prob 1/4 each
  - wait $\rightarrow$ (00,10), (01,11), (10,00), (11,01), (00,11), (01,10), (10,01) or (11,00) w/ prob 1/8 each
  - Reconstruction: Let $\beta_1\beta_2 = \text{share}_{\text{Alice}} \oplus \text{share}_{\text{Bob}}$. Map $\beta_1\beta_2$ as follows: 00 $\rightarrow$ buy, 01 $\rightarrow$ sell, 10 or 11 $\rightarrow$ wait
- Is it secure?

Secret-Sharing

- More general secret-sharing
  - Allow more than two parties (how?)
  - Privileged subsets of parties should be able to reconstruct the secret (not necessarily just the entire set of parties)
- Very useful
  - Direct applications (distributed storage of data or keys)
  - Important component in other cryptographic constructions
    - Amplifying secrecy of various primitives
    - Secure multi-party computation
    - Attribute-Based Encryption
    - Leakage resilience ...

Threshold Secret-Sharing

- (n,t)-secret-sharing
  - Divide a message m into n shares $s_1,\ldots,s_n$, such that
    - any t shares are enough to reconstruct the secret
    - up to t-1 shares should have no information about the secret
- our previous example: (2,2) secret-sharing

e.g., $(s_1,\ldots,s_{t-1})$ has the same distribution for every m in the message space

Threshold Secret-Sharing

Additive Secret-Sharing

- Construction: (n,n) secret-sharing
  - Message-space = share-space = G, a finite group
    - e.g. $G = \mathbb{Z}_2$ (group of bits, with xor as the group operation)
    - or, $G = \mathbb{Z}_2^d$ (group of d-bit strings)
    - or, $G = \mathbb{Z}_p$ (group of integers mod p)
  - Share(m):
    - Pick $(s_1,\ldots,s_{n-1})$ uniformly at random from G^{n-1}
    - Let $s_n = -(s_1 + \ldots + s_{n-1}) + m$
  - Reconstruct$(s_1,\ldots,s_n)$: $m = s_1 + \ldots + s_n$
  - Claim: This is an (n,n) secret-sharing scheme [Why?]

Any (n-1)-tuple of shares is uniformly distributed, irrespective of the message

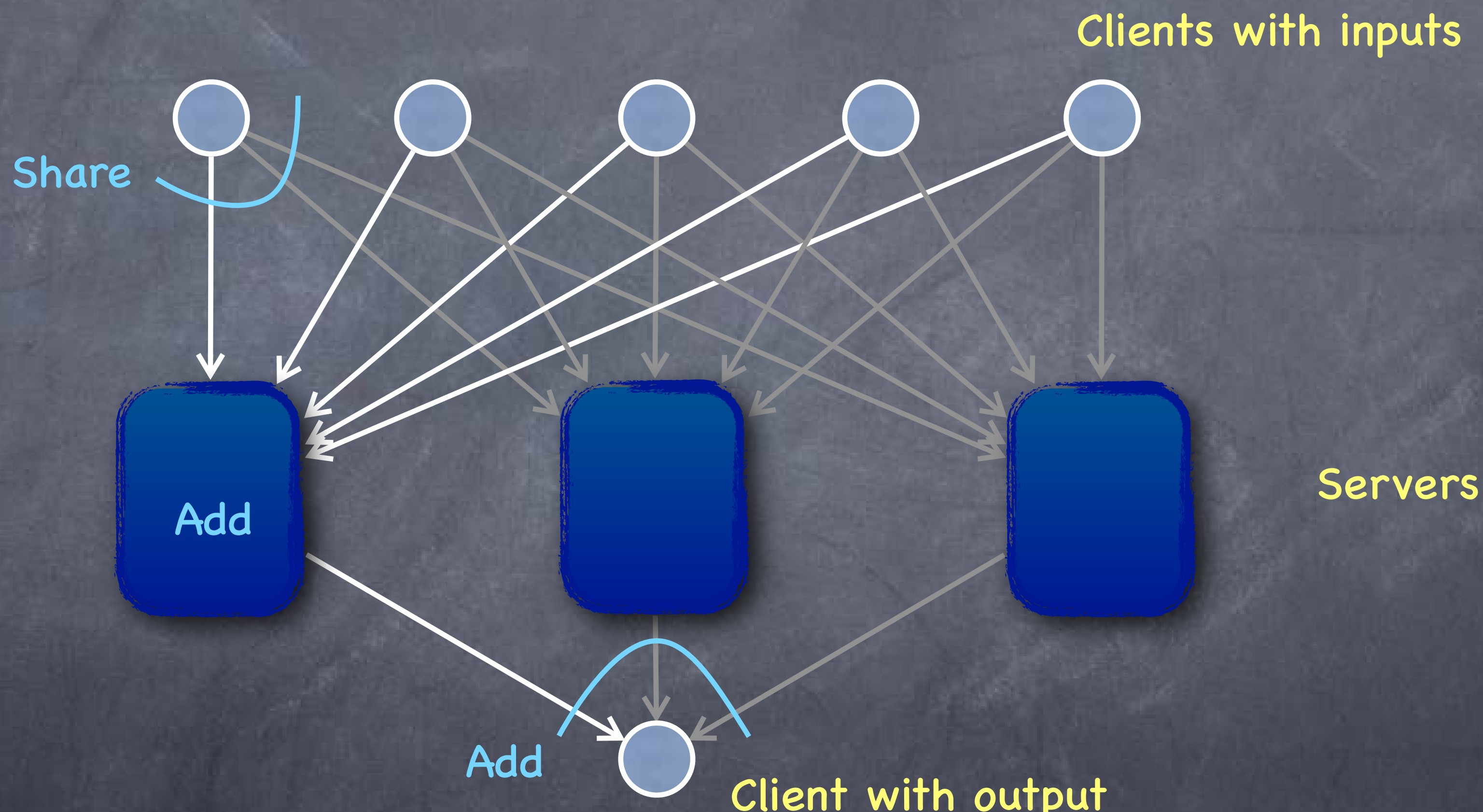
PROOF

Additive Secret-Sharing: Proof

- Share(m):
  - Pick $(s_1,\ldots,s_{n-1})$ uniformly at random from G^{n-1}
  - Let $s_n = m - (s_1 + \ldots + s_{n-1})$
- Claim: Upto n-1 shares give no information about m
- Proof: Let $T \subseteq \{1,\ldots,n\}$, $|T| = n-1$. We shall show that $\{ s_i \}_{i \in T}$ is distributed the same way (in fact, uniformly) irrespective of what m is.
  - For concreteness consider $T = \{2,\ldots,n\}$. Fix any (n-1)-tuple of elements in G, $(g_1,\ldots,g_{n-1}) \in G^{n-1}$. To prove $\Pr[(s_2,\ldots,s_n)=(g_1,\ldots,g_{n-1})]$ is same for all m.
  - Fix any m.
  - $(s_2,\ldots,s_n) = (g_1,\ldots,g_{n-1}) \Leftrightarrow (s_2,\ldots,s_{n-1}) = (g_1,\ldots,g_{n-2})$ and $s_1 = m - (g_1+\ldots+g_{n-1})$.
  - So $\Pr[(s_2,\ldots,s_n) = (g_1,\ldots,g_{n-1})] = \Pr[(s_1,\ldots,s_{n-1}) = (a,g_1,\ldots,g_{n-2})]$ where $a := m - (g_1+\ldots+g_{n-1})$
  - But $\Pr[(s_1,\ldots,s_{n-1}) = (a,g_1,\ldots,g_{n-2})] = 1/|G|^{n-1}$, since $(s_1,\ldots,s_{n-1})$ is picked uniformly at random from G^{n-1}
  - Hence $\Pr[(s_2,\ldots,s_n) = (g_1,\ldots,g_{n-1})] = 1/|G|^{n-1}$, irrespective of m. □

An Application

- Gives a "private summation" protocol



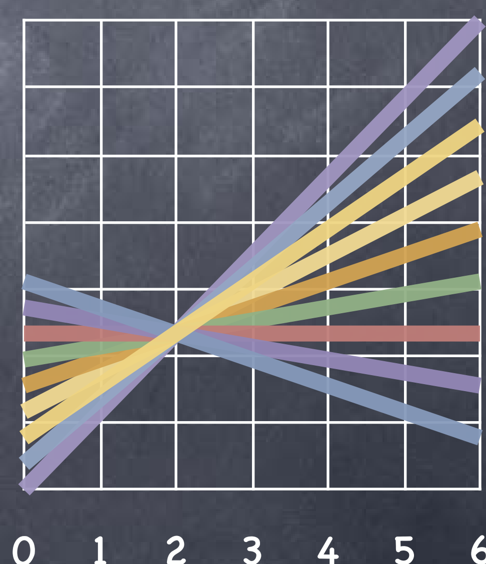
- No colluding set of servers/clients will learn more than the inputs/output of the clients in the collusion, provided that at least one server stays out of the collusion

Threshold Secret-Sharing

- Construction: (n,2) secret-sharing
- Message-space = share-space = F, a field (e.g. integers mod a prime)
  - Share(m): pick random r. Let $s_i = r \cdot a_i + m$ (for $i=1,\ldots,n < |F|$)
  - Reconstruct(s_i, s_j): $r = (s_i-s_j)/(a_i-a_j)$; $m = s_i - r \cdot a_i$
  - Each s_i by itself is uniformly distributed, irrespective of m [Why?]
  - "Geometric" interpretation
    - Sharing picks a random "line" $y = f(x)$, such that $f(0) = m$. Shares $s_i = f(a_i)$.
    - s_i is independent of m: exactly one line passing through (a_i,s_i) and $(0,m')$ for any secret m'
    - But can reconstruct the line from two points!

a_i are n distinct, non-zero field elements

Since a_i^{-1} exists, exactly one solution for $r \cdot a_i+m=d$, for every value of d



PROOF

(n,2) Secret-Sharing: Proof

- Share(m): pick random $r \leftarrow F$. Let $s_i = r \cdot a_i + m$ (for $i=1,\ldots,n < |F|$)
- Claim: Any one share gives no information about m
- Proof: For any $i \in \{1,\ldots,n\}$ we shall show that s_i is distributed the same way (in fact, uniformly) irrespective of what m is.
- Consider any $g \in F$. We shall show that $\Pr[\, s_i = g \,]$ is independent of m.
- Fix any m.
- For any $g \in F$, $s_i = g \Leftrightarrow r \cdot a_i + m = g \Leftrightarrow r = (g - m) \cdot a_i^{-1}$ (since $a_i \neq 0$)
- So, $\Pr[\, s_i = g \,] = \Pr[\, r = (g - m) \cdot a_i^{-1} \,] = 1/|F|$, since r is chosen uniformly at random $\square$

Threshold Secret-Sharing

Shamir Secret-Sharing

- (n,t) secret-sharing in a field F
- Generalizing the geometric/algebraic view: instead of lines, use polynomials
  - Share(m): Pick a random degree t-1 polynomial f(X), such that f(0) = m. Shares are $s_i = f(a_i)$.
    - Random polynomial with f(0) = m: $c_0 + c_1X + c_2X^2 + \ldots + c_{t-1}X^{t-1}$ by picking $c_0 = m$ and $c_1,\ldots,c_{t-1}$ at random.
  - Reconstruct($s_1,\ldots,s_t$): Lagrange interpolation to find $m = c_0$
    - Need t points to reconstruct the polynomial. Given t-1 points, out of $|F|^{t-1}$ polynomials passing through (0,m') (for any m') there is exactly one that passes through the t-1 points

Lagrange Interpolation

- Given t distinct points on a degree t-1 polynomial (univariate, over some field of more than t elements), reconstruct the entire polynomial (i.e., find all t co-efficients)
  - t variables: $c_0,\dots,c_{t-1}$.
    t equations: $1.c_0 + a_i.c_1 + a_i^2.c_2 + \dots a_i^{t-1}.c_{t-1} = s_i$
  - A linear system: W**c**=**s**, where W is a txt matrix with i^{th} row, $W_i= (1\ \ a_i\ \ a_i^2\ \dots\ a_i^{t-1})$
  - W (called the Vandermonde matrix) is invertible
    - $\mathbf{c} = W^{-1}\mathbf{s}$

Today

- Preserving secrecy: view is independent of the message
  - i.e., $\forall$ view, $\forall$ msg_1,msg_2, Pr[view | msg_1] = Pr[view | msg_2]
    - View does not give any additional information about the message, than what was already known (the prior)
  - The view could be **simulated** without any knowledge of the message
  - Holds even against unbounded computational power (a la Shannon)
- Achieved in additive and threshold secret-sharing schemes
- Such secrecy not always possible (e.g., no public-key encryption against computationally unbounded adversaries)