Cryptography

Lecture 1

Cryptography

Lecture 1

Our first encounter with secrecy:
Secret-Sharing

Secrecy

Secrecy

- Cryptography is all about "controlling access to information"
  - Access to learning and/or influencing information

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 Game

- A “dealer” and two “players” Alice and Bob

A Game

- A “dealer” and two “players” Alice and Bob
- Dealer has a message, say two bits m_1m_2

A Game

- A "dealer" and two "players" Alice and Bob
- Dealer has a message, say two bits m_1m_2
- She wants to "share" it among the two players so that neither player by herself/himself learns anything about the message, but together they can find it

A Game

- A "dealer" and two "players" Alice and Bob
- Dealer has a message, say two bits m_1m_2
- She wants to "share" it among the two players so that neither player by herself/himself learns anything about the message, but together they can find it
- Bad idea: Give m_1 to Alice and m_2 to Bob

A Game

- A "dealer" and two "players" Alice and Bob
- Dealer has a message, say two bits $m_1 m_2$
- She wants to "share" it among the two players so that neither player by herself/himself learns anything about the message, but together they can find it
- Bad idea: Give m_1 to Alice and m_2 to Bob
- Other ideas?

Sharing a bit

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

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)

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)
  - Alice learns nothing either: for each possible value of m (0 or 1), a is a random bit (0 w.p. ½, 1 w.p. ½)

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)
  - Alice learns nothing either: 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 $(0,1)$

Sharing a bit

- To share a bit m, Dealer picks a uniformly random bit b and gives a := m⊕b to Alice and b to Bob
  - Bob learns nothing (b is a random bit)
  - Alice learns nothing either: for each possible value of m (0 or 1), a is a random bit (0 w.p. ½, 1 w.p. ½)
    - Her view is independent of the message

m = 0 → (a,b) = (0,0) or (1,1)
m = 1 → (a,b) = (1,0) or (0,1)

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)
  - Alice learns nothing either: 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 $a \oplus b$

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)
  - Alice learns nothing either: for each possible value of m (0 or 1), a is a random bit (0 w.p. ½, 1 w.p. ½)
    - Her view is independent of the message
  - Together they can recover m as $a \oplus b$
- Multiple bits can be shared independently: as, $m_1m_2 = a_1a_2 \oplus b_1b_2$

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

Sharing a bit

- To share a bit m, Dealer picks a uniformly <u>random</u> bit b and gives $a := m \oplus b$ to Alice and b to Bob
  - Bob learns nothing (b is a random bit)
  - Alice learns nothing either: for each possible value of m (0 or 1), a is a random bit (0 w.p. ½, 1 w.p. ½)
    - Her view is <u>independent</u> of the message
  - Together they can recover m as $a \oplus b$
- Multiple bits can be shared independently: as, $\underline{m_1 m_2} = \underline{a_1 a_2} \oplus \underline{b_1 b_2}$
- Note: any one share can be chosen before knowing the message [why?]

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

Secrecy

Secrecy

- Is the message m really secret?

Secrecy

- Is the message m really secret?
- Alice or Bob can correctly find the bit m with probability ½, by randomly guessing

Secrecy

- Is the message m really secret?
- 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!)

Secrecy

- Is the message m really secret?
- 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

Secrecy

- Is the message m really secret?
- 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 additional information to either party

Secrecy

- Is the message m really secret?
- 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 additional information to either party
- Typical crypto goal: **preserving** secrecy

Secrecy

Secrecy

- Goal: What Alice (or Bob) knows about the message after seeing her share is the same as what she knew a priori

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: probability distribution over the message

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: probability distribution over the message
  - For each message m, Pr[msg=m]

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: probability distribution over the message
  - For each message m, Pr[msg=m]
- What she knows after seeing her share (a.k.a. her view)

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: 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]

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: 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]
- Secrecy: $\forall$ v, $\forall$ m, Pr[msg=m | view = v] = Pr[msg = m]

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: 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]
- Secrecy: $\forall$ v, $\forall$ m, Pr[msg=m | view = v] = Pr[msg = m]
  - i.e., view is independent of message

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: 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]
- Secrecy: $\forall$ v, $\forall$ m, Pr[msg=m | view = v] = Pr[msg = m]
  - i.e., view is independent of message
  - Equivalently, $\forall$ v, $\forall$ m, Pr[view=v | msg=m] = Pr[view = v]

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: 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]
- Secrecy: ∀ v, ∀ m, Pr[msg=m | view = v] = Pr[msg = m]
  - i.e., view is independent of message
  - Equivalently, ∀ v, ∀ m, Pr[view=v | msg=m] = Pr[view = v]

Determined by the scheme

Secrecy

Secrecy

- Secrecy: $\forall$ v, $\forall$ m, Pr[msg=m | view = v] = Pr[msg = m]

Secrecy

- Secrecy: $\forall$ v, $\forall$ m, Pr[msg=m | view = v] = Pr[msg = m]
  - i.e., view is independent of message

Secrecy

- Secrecy: $\forall$ v, $\forall$ m, Pr[msg=m | view = v] = Pr[msg = m]
  - i.e., view is independent of message
  - Equivalently, $\forall$ v, $\forall$ m, Pr[view=v | msg=m] = Pr[view = v]

Secrecy

- Secrecy: $\forall$ v, $\forall$ m, Pr[msg=m | view = v] = Pr[msg = m]
  - i.e., view is independent of message
  - Equivalently, $\forall$ v, $\forall$ m, Pr[view=v | msg=m] = Pr[view = v]
  - i.e., for all possible values of the message, the view is distributed the same way

Secrecy

- Secrecy: $\forall$ v, $\forall$ m, Pr[msg=m | view = v] = Pr[msg = m]
  - i.e., view is independent of message
  - Equivalently, $\forall$ v, $\forall$ m, Pr[view=v | msg=m] = Pr[view = v]
  - i.e., for all possible values of the message, the view is distributed the same way
- Equivalently (why?), $\forall$ v, $\forall m_1, m_2$, Pr[view=v | msg=m_1] = Pr[view=v | msg=m_2]

Secrecy

- Secrecy: $\forall$ v, $\forall$ m, Pr[msg=m | view = v] = Pr[msg = m]
  - i.e., view is independent of message
  - Equivalently, $\forall$ v, $\forall$ m, Pr[view=v | msg=m] = Pr[view = v]
  - i.e., for all possible values of the message, the view is distributed the same way
- Equivalently (why?), $\forall$ v, $\forall m_1, m_2$, Pr[view=v | msg=m_1] = Pr[view=v | msg=m_2]
- Important: can't say Pr[msg=m_1 | view=v] = Pr[msg=m_2 | view=v] (unless the prior is uniform)

Secrecy

- Secrecy: $\forall$ v, $\forall$ m, Pr[msg=m | view = v] = Pr[msg = m]
  - i.e., view is independent of message
  - Equivalently, $\forall$ v, $\forall$ m, Pr[view=v | msg=m] = Pr[view = v]
  - i.e., for all possible values of the message, the view is distributed the same way
- Equivalently (why?), $\forall$ v, $\forall m_1$, m_2, Pr[view=v | msg=m_1] = Pr[view=v | msg=m_2]

Doesn't involve message distribution at all.

- Important: can't say Pr[msg=m_1 | view=v] = Pr[msg=m_2 | view=v] (unless the prior is uniform)

Exercise

Exercise

- Consider the following secret-sharing scheme

Exercise

- Consider the following secret-sharing scheme
  - Message space = { buy, sell, wait }

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

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

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

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

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

Secret-Sharing

- More general secret-sharing

Secret-Sharing

- More general secret-sharing
  - Allow more than two parties (how?)

Secret-Sharing

- More general secret-sharing
  - Allow more than two parties (how?)
  - Privileged <u>subsets</u> of parties should be able to reconstruct the secret (not necessarily just the entire set of parties)

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

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)

Secret-Sharing

- More general secret-sharing
  - Allow more than two parties (how?)
  - Privileged <u>subsets</u> 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

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

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

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

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

Threshold Secret-Sharing

- (n,t)-secret-sharing

Threshold Secret-Sharing

- (n,t)-secret-sharing
  - Divide a message m into n shares $s_1,\ldots,s_n$, such that

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

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

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

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

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

Threshold Secret-Sharing

- Construction: (n,n) secret-sharing

Threshold Secret-Sharing

- Construction: (n,n) secret-sharing
  - Message-space = share-space = G, a finite group

Threshold 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)

Threshold 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)

Threshold 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)

Threshold 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):

Threshold 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

Threshold 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
    - Let $s_n = -(s_1 + \ldots + s_{n-1}) + M$

Threshold 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
    - Let $s_n = -(s_1 + \ldots + s_{n-1}) + M$
  - Reconstruct($s_1,\ldots,s_n$): $M = s_1 + \ldots + s_n$

Threshold 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
    - 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?]

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
    - 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?]

PROOF

Additive Secret-Sharing: Proof

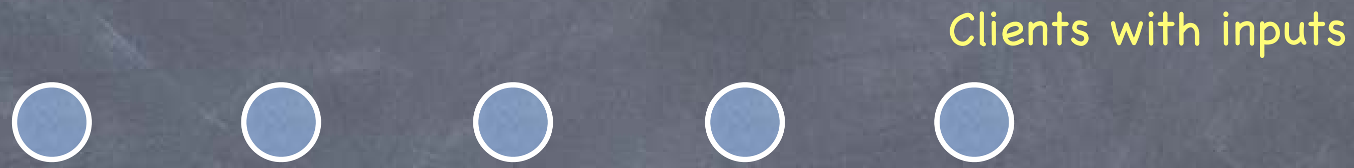
- Share(M):
  - Pick $s_1,\ldots,s_{n-1}$ uniformly at random from G
  - Let $s_n = M - (s_1 + \ldots + s_{n-1})$
- Reconstruct($s_1,\ldots,s_n$): $M = s_1 + \ldots + s_n$
- Claim: Upto n-1 shares give no information about M
- Proof: Let $T \subseteq \{1,\ldots,n\}$, $|T| = n-1$. We shall show that $\{ s_i \}_{i\in T}$ 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 $\Pr[\,(s_2,\ldots,s_n)=(g_1,\ldots,g_{n-1})\,] = \Pr[\,(s_1,\ldots,s_{n-1})=(a,g_1,\ldots,g_{n-2})\,]$, $a:=(M-(g_1+\ldots+g_{n-1}))$
  - But $\Pr[(s_1,\ldots,s_{n-1})=(a,g_1,\ldots,g_{n-2})] = 1/|G|^{n-1}$, since $(s_1,\ldots,s_{n-1})$ are picked uniformly at random from G
  - Hence $\Pr[\,(s_2,\ldots,s_n)=(g_1,\ldots,g_{n-1})\,] = 1/|G|^{n-1}$, irrespective of M. $\square$

An Application

- Gives a "private summation" protocol

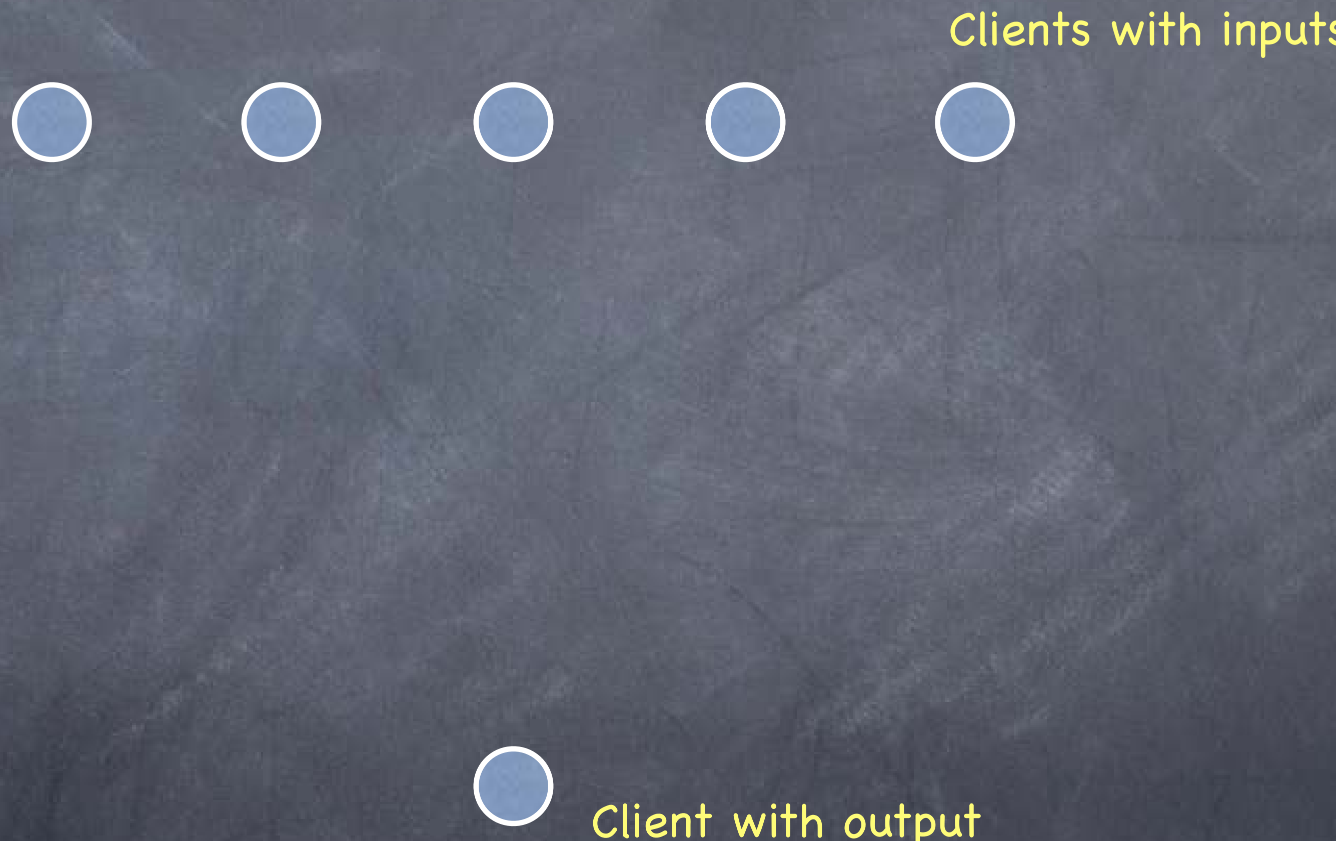
An Application

- Gives a "private summation" protocol



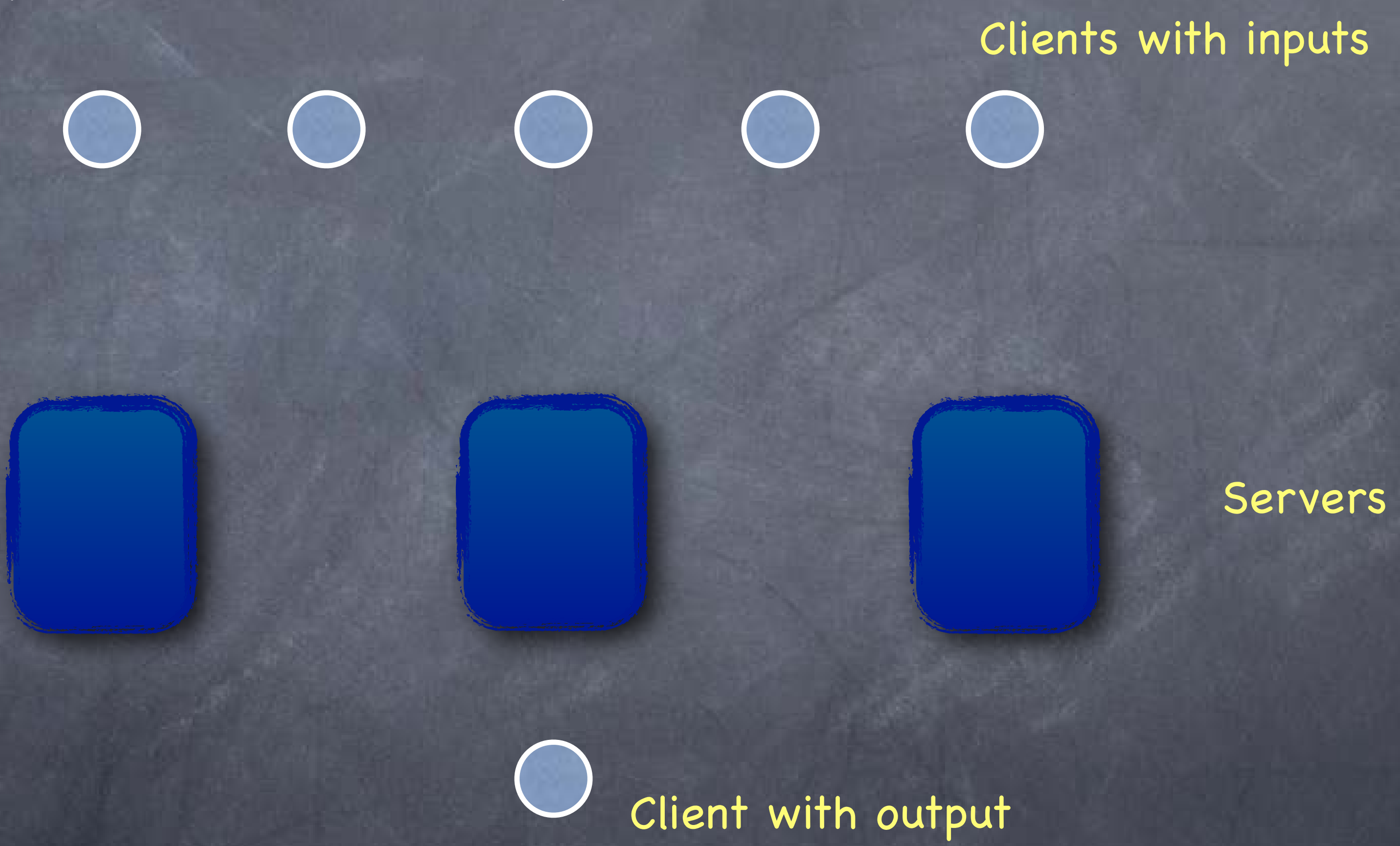
An Application

- Gives a "private summation" protocol



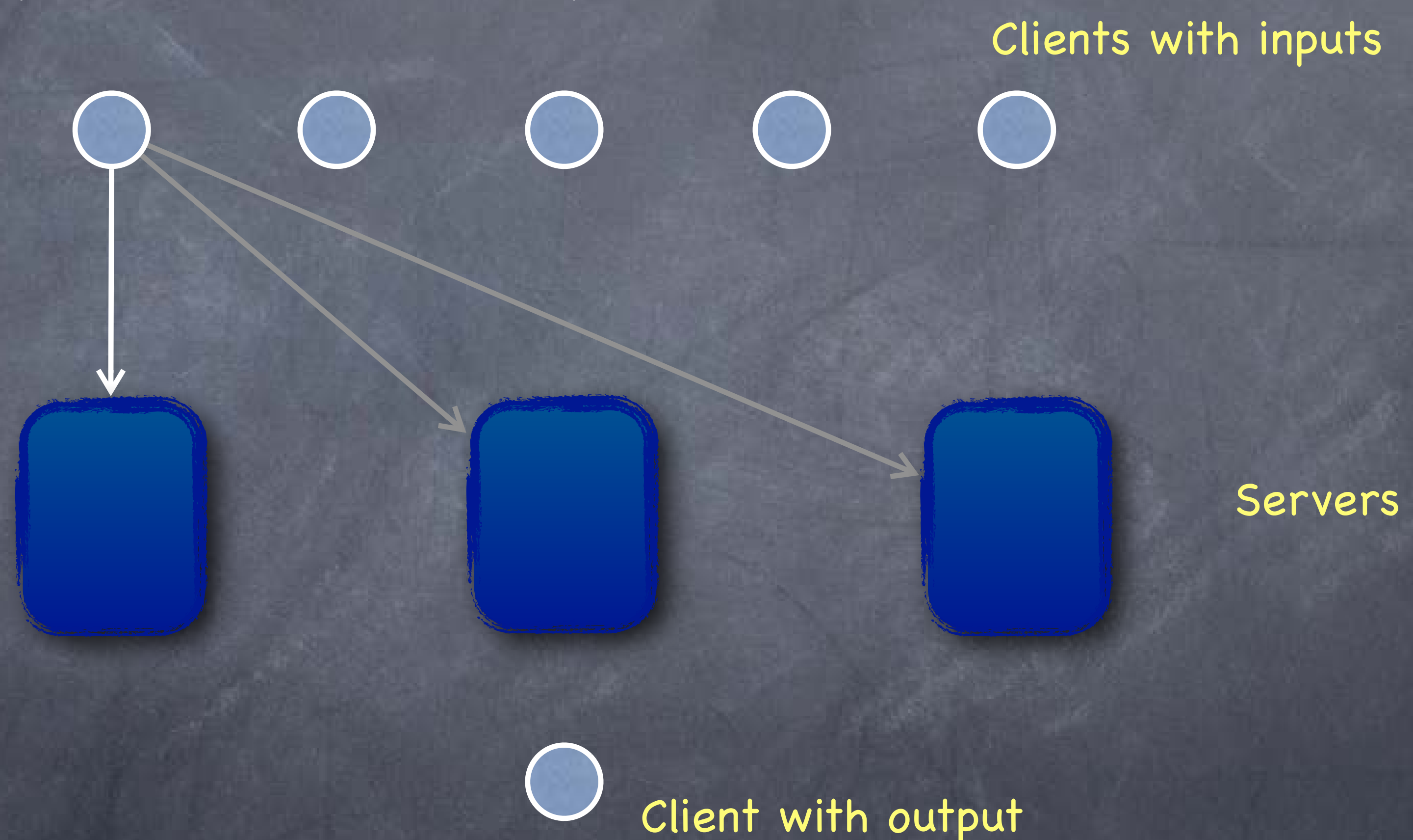
An Application

- Gives a “private summation” protocol



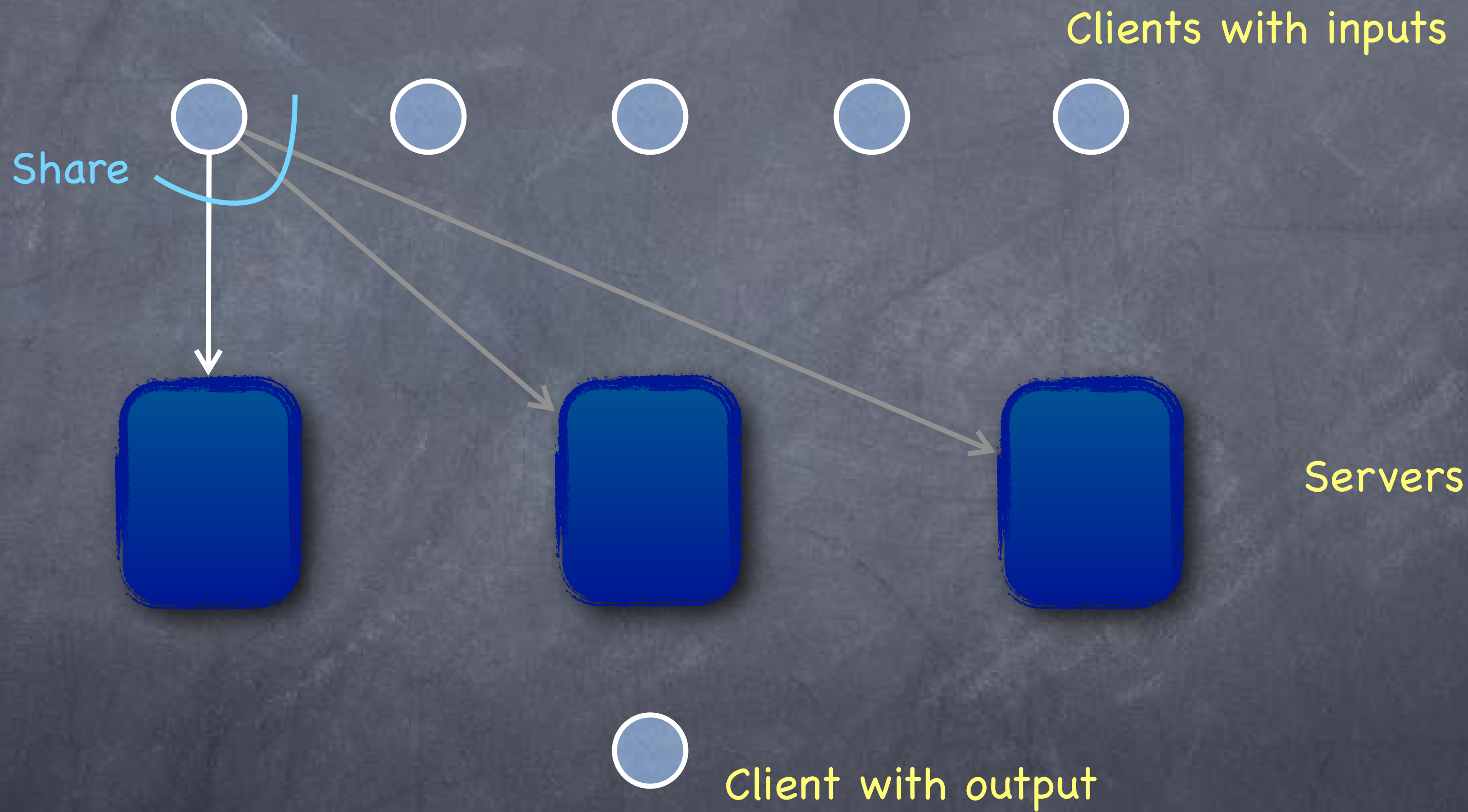
An Application

- Gives a "private summation" protocol



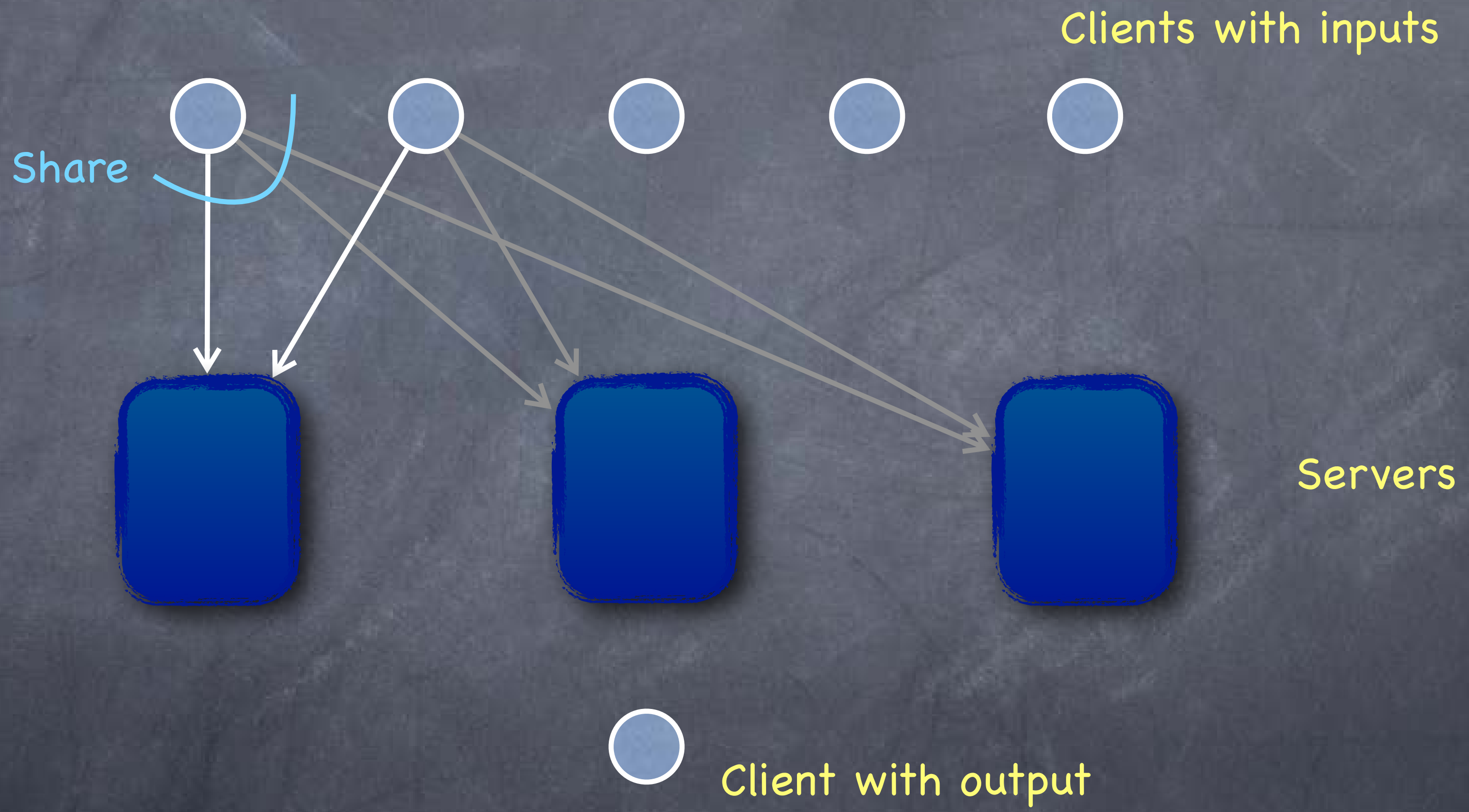
An Application

- Gives a "private summation" protocol



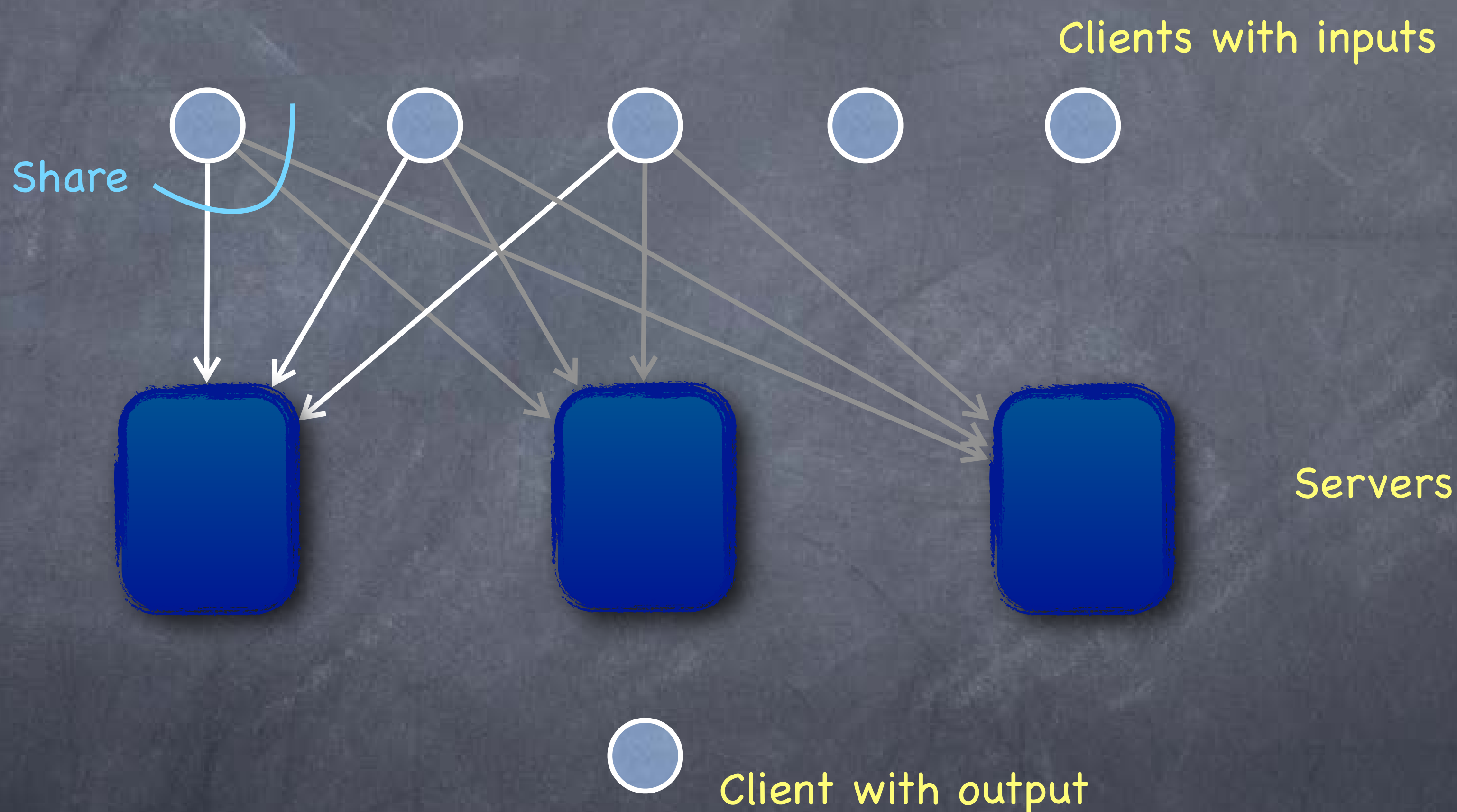
An Application

- Gives a "private summation" protocol



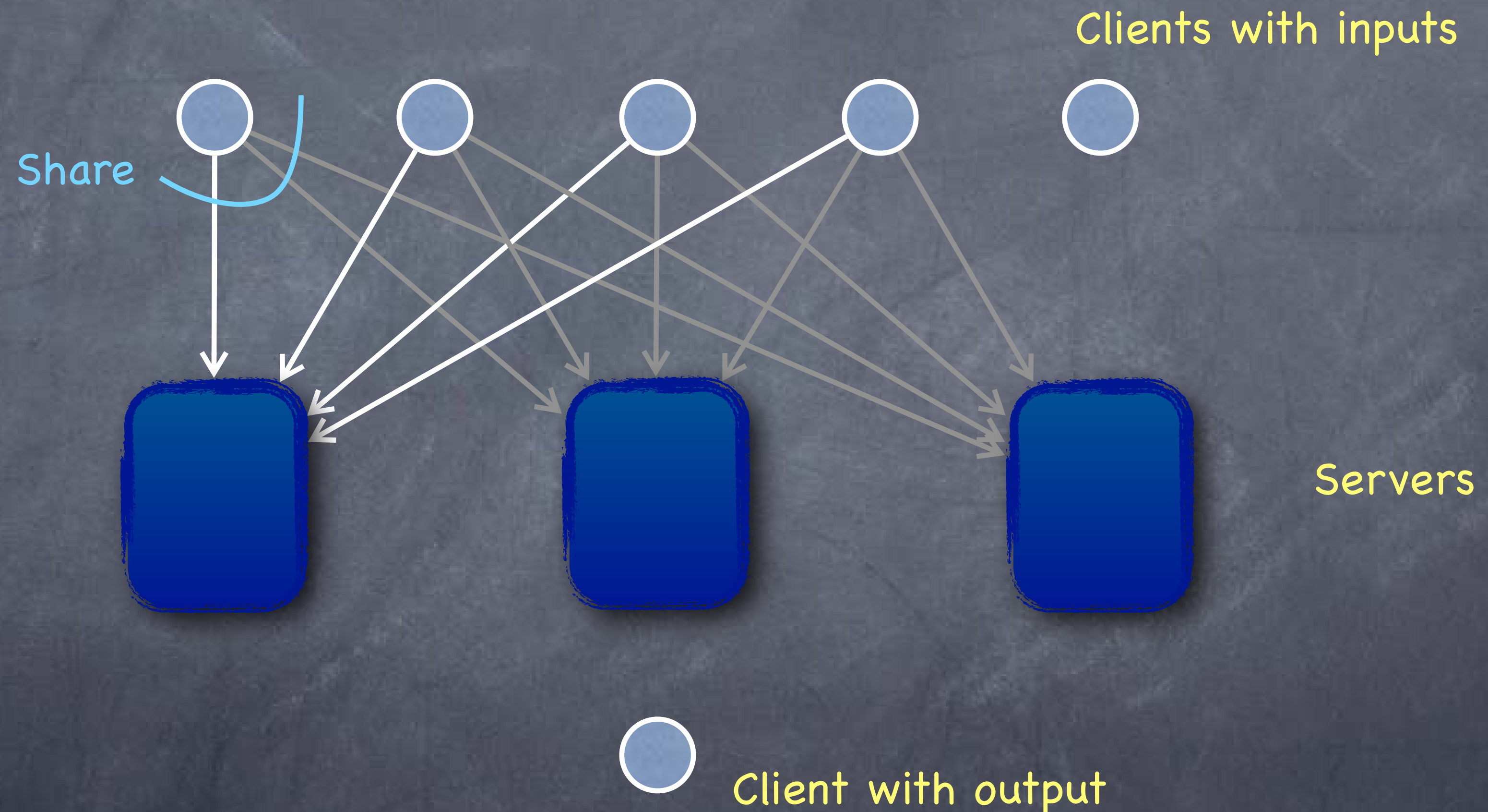
An Application

- Gives a “private summation” protocol



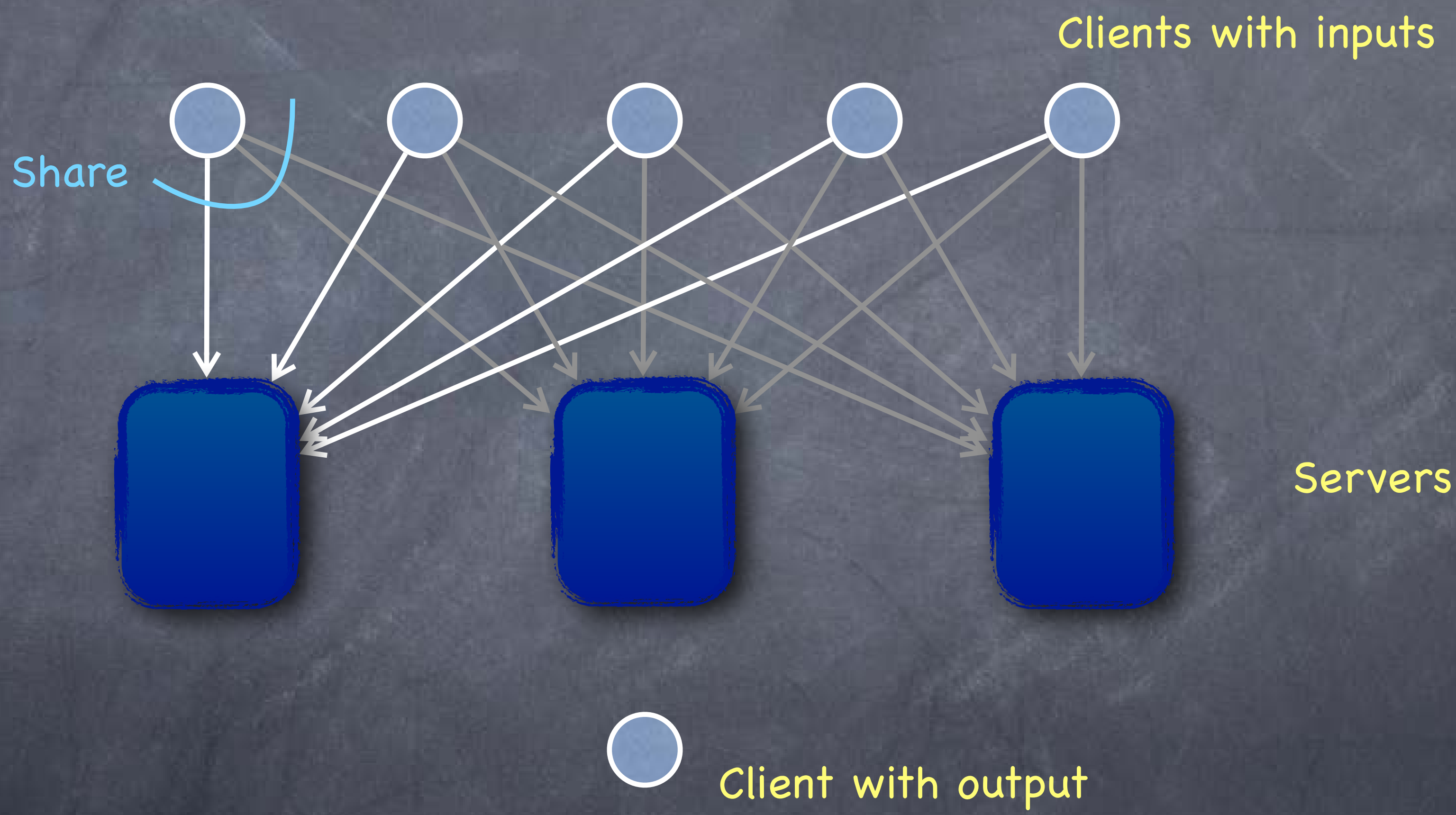
An Application

- Gives a "private summation" protocol



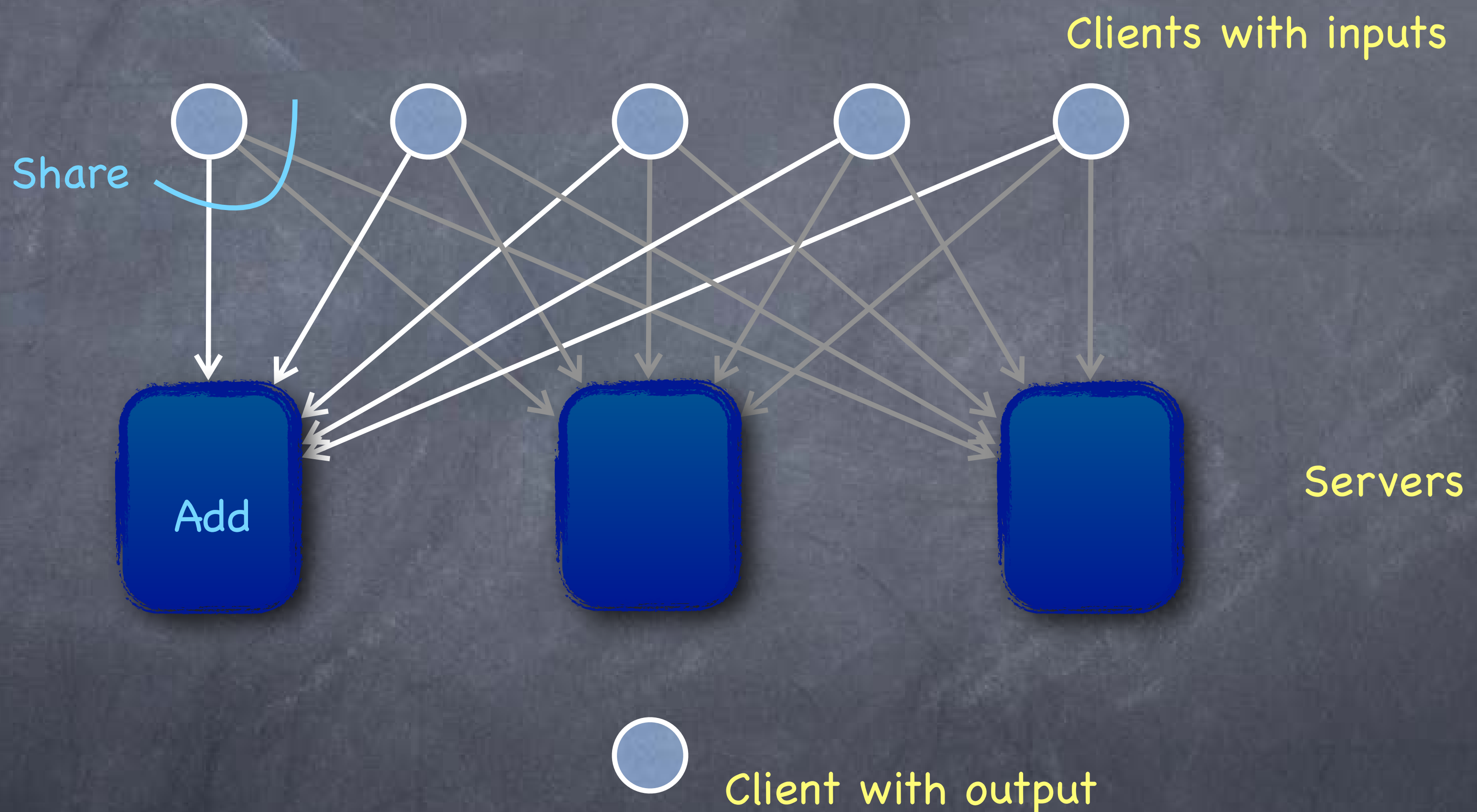
An Application

- Gives a "private summation" protocol



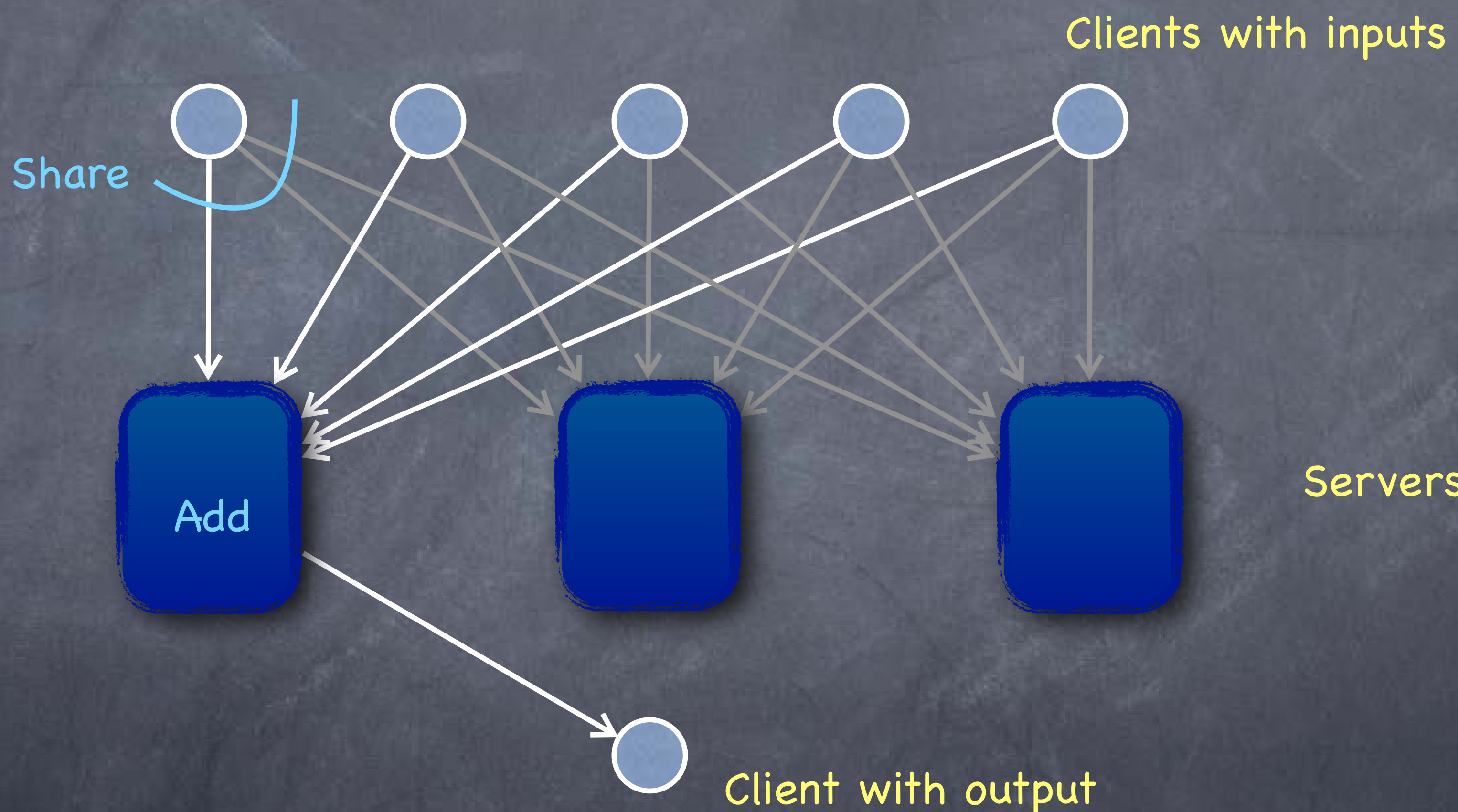
An Application

- Gives a "private summation" protocol



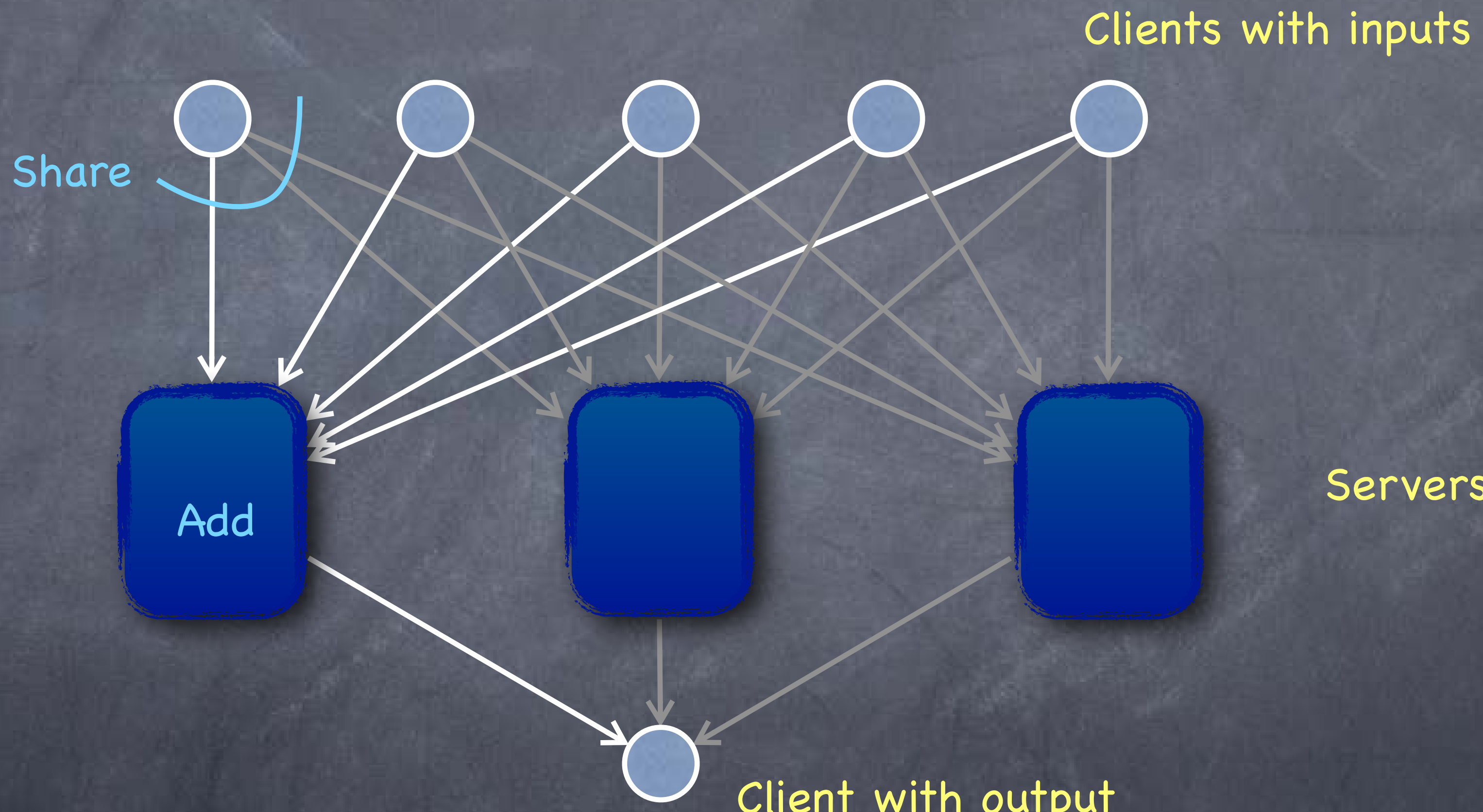
An Application

- Gives a “private summation” protocol



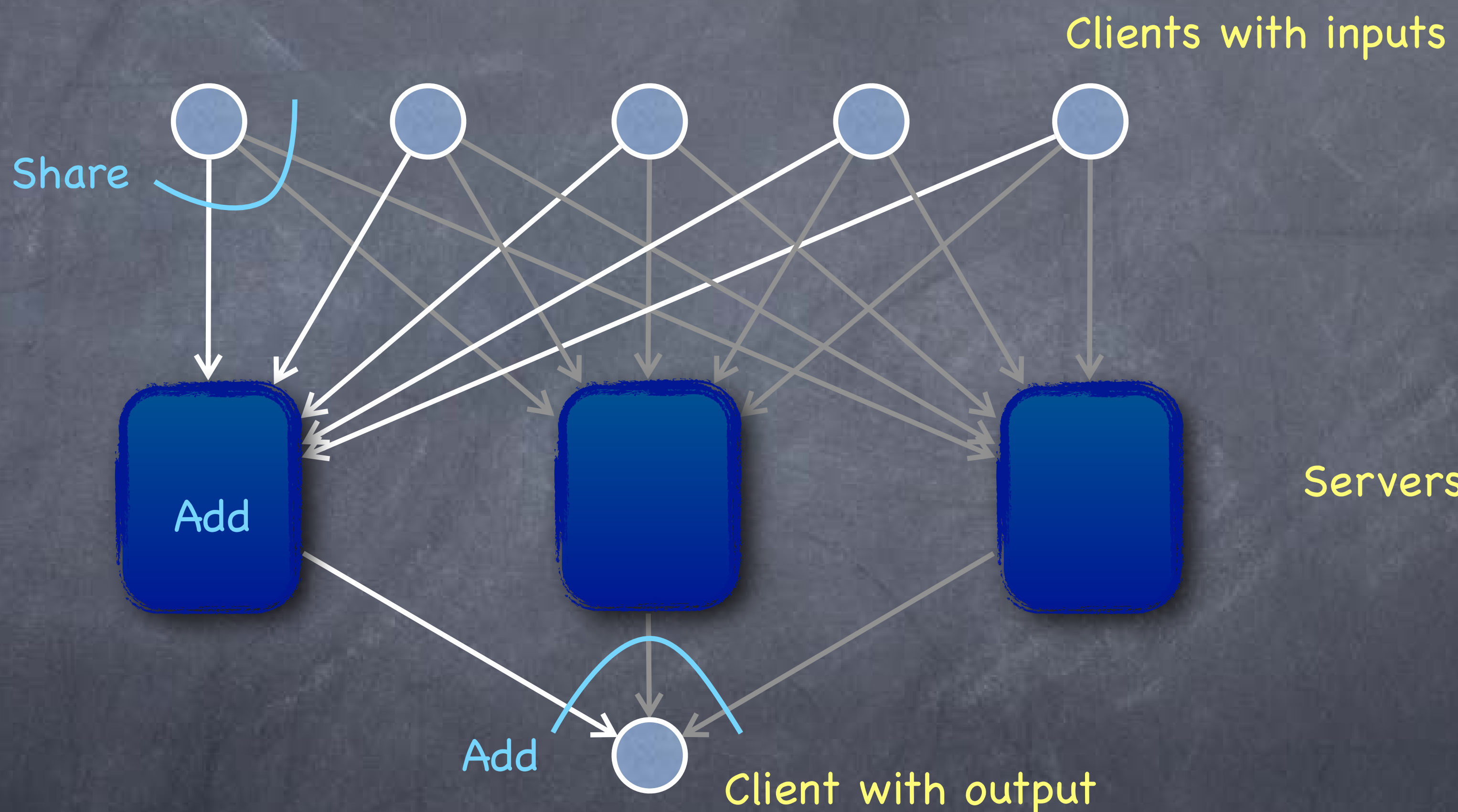
An Application

- Gives a "private summation" protocol



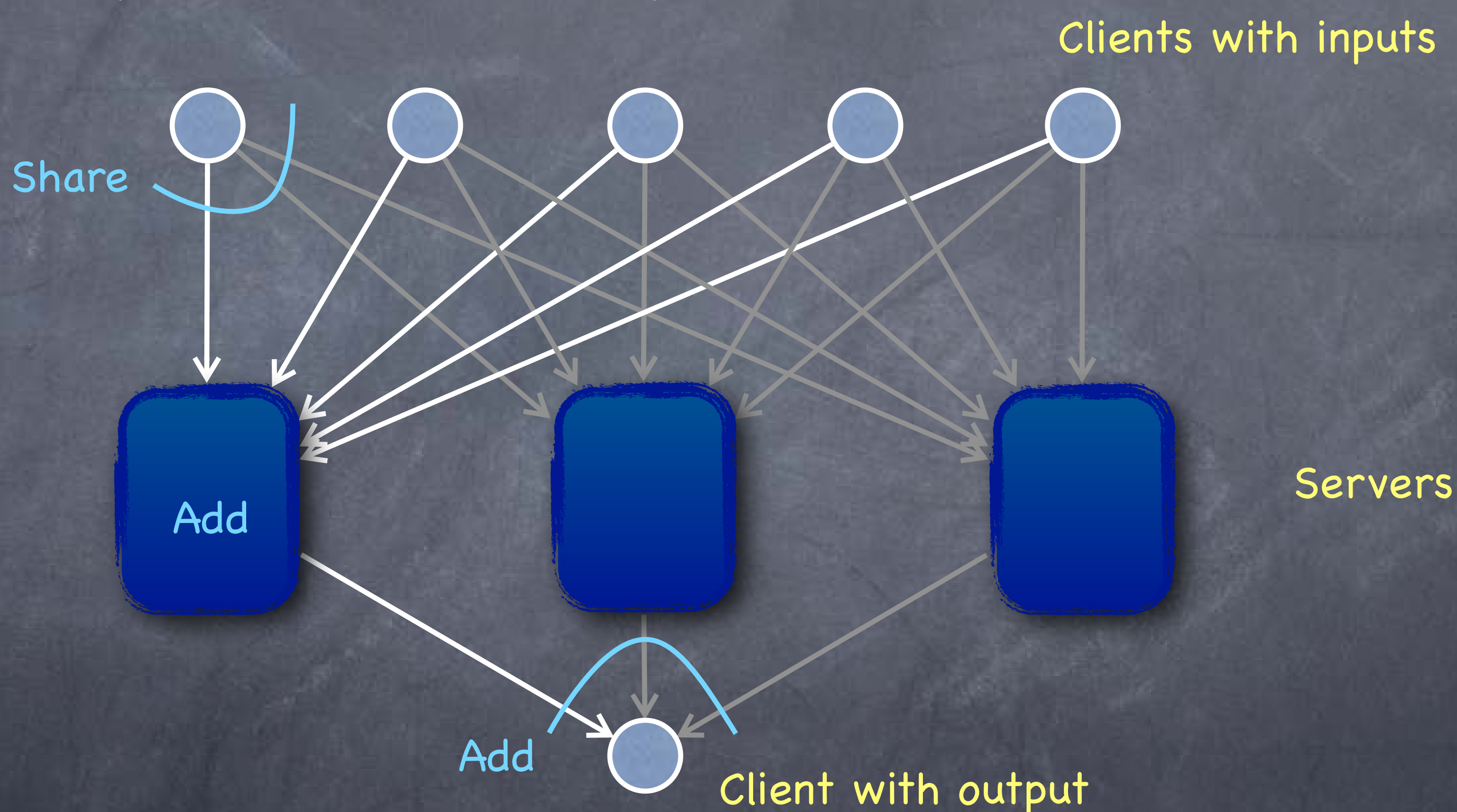
An Application

- Gives a "private summation" protocol



An Application

- Gives a "private summation" protocol



- Secure against passive corruption (no colluding set of servers/clients learn more than what they must) if at least one server stays out of the collusion

Threshold Secret-Sharing

Threshold Secret-Sharing

- Construction: (n,2) secret-sharing

Threshold Secret-Sharing

- Construction: (n,2) secret-sharing
- Message-space = share-space = F, a field (e.g. integers mod a prime)

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 $i=1,\ldots,n < |F|$)

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 $i=1,\ldots,n < |F|$)

a_i are n distinct, non-zero field elements

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): $r = (s_i-s_j)/(a_i-a_j)$; $M = s_i - r \cdot a_i$

a_i are n distinct, non-zero field elements

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?]

a_i are n distinct, non-zero field elements

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?]

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

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

a_i are n distinct, non-zero field elements

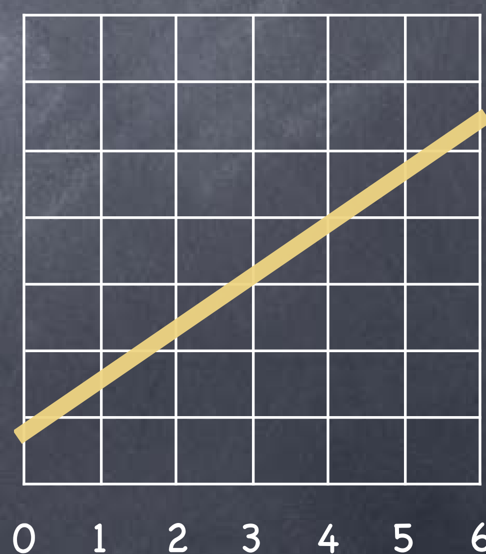
Since a_i^{-1} exists, exactly one solution for $r \cdot a_i + M = d$, for every value of d

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 $s_i = f(a_i)$.

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

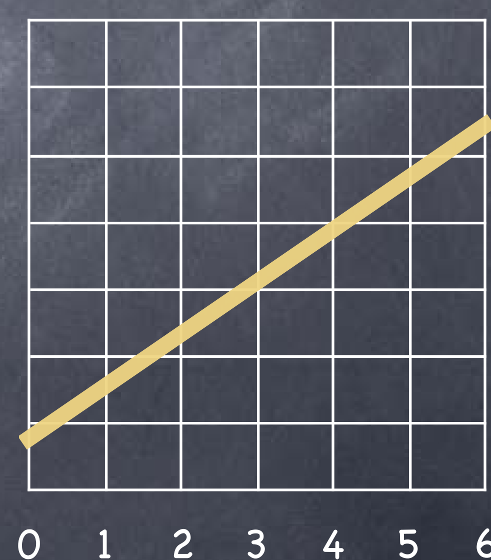


Threshold Secret-Sharing

- 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 $i=1,\ldots,n < |F|$)

    a_i are n distinct, non-zero field elements
  - 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?]

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

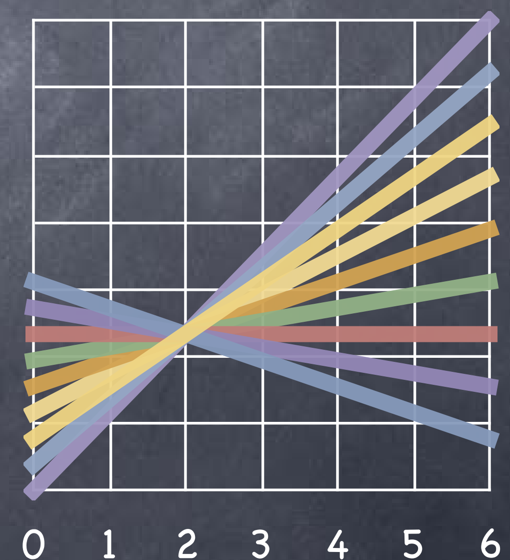


Threshold Secret-Sharing

- 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'

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

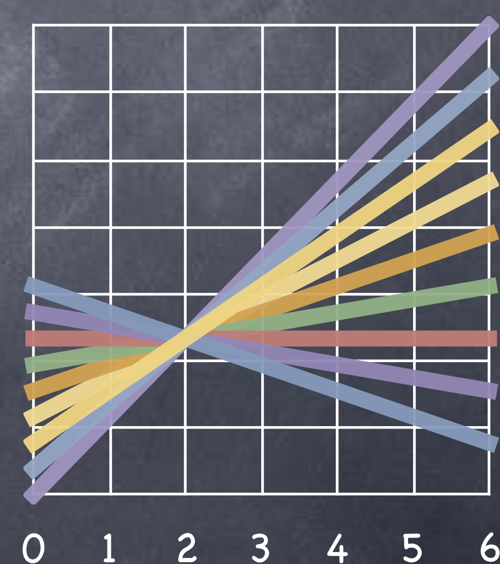


Threshold Secret-Sharing

- 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 $i=1,\ldots,n < |F|$)

    a_i are n distinct, non-zero field elements
  - 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?]

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



PROOF

(n,2) Secret-Sharing: Proof

- Share(M): pick random $r \leftarrow F$. Let $s_i = r \cdot a_i + M$ (for $i=1,\ldots,n < |F|$)
- Reconstruct(s_i, s_j): $r = (s_i - s_j)/(a_i - a_j)$; $M = s_i - r \cdot a_i$

- Claim: Any one share gives no information about M
- Proof: For any $i \in \{1,..,n\}$ 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 $g \in F$, $s_i = g \Leftrightarrow r \cdot i + M = g \Leftrightarrow r = (g-M) \cdot a_i^{-1}$ (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 $\square$

Today

Today

- Secrecy: if view is independent of the message

Today

- Secrecy: if view is independent of the message
  - i.e., $\forall$ view, $\forall$ msg_1,msg_2, Pr[view | msg_1] = Pr[view | msg_2]

Today

- Secrecy: if view is independent of the message
  - i.e., $\forall$ view, $\forall$ msg_1, msg_2, $\Pr[view \mid msg_1] = \Pr[view \mid msg_2]$
  - View does not give any additional information about the message, than what was already known (prior)

Today

- Secrecy: if 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 <u>additional</u> information about the message, than what was already known (prior)
  - Allows for unbounded computational power

Today

- Secrecy: if view is independent of the message
  - i.e., $\forall$ view, $\forall$ msg_1, msg_2, $\Pr[\text{view} \mid msg_1] = \Pr[\text{view} \mid msg_2]$
  - View does not give any <u>additional</u> information about the message, than what was already known (prior)
  - Allows for unbounded computational power
- Such secrecy not always possible (e.g., no public-key encryption)

Today

- Secrecy: if view is independent of the message
  - i.e., $\forall$ view, $\forall$ msg_1, msg_2, $Pr[view \mid msg_1] = Pr[view \mid msg_2]$
  - View does not give any additional information about the message, than what was already known (prior)
  - Allows for unbounded computational power
- Such secrecy not always possible (e.g., no public-key encryption)
- Next: secrecy against computationally bounded players