Attribute-Based Cryptography

Lecture 21
And Pairing-Based Cryptography
Identity-Based Encryption
Identity-Based Encryption

In PKE, KeyGen produces a random (PK, SK) pair
Identity-Based Encryption

- In PKE, KeyGen produces a random (PK,SK) pair
- Can I have a “fancy public-key” (e.g., my name)?
Identity-Based Encryption

- In PKE, KeyGen produces a random (PK, SK) pair
- Can I have a “fancy public-key” (e.g., my name)?
  - But no one should be able to pick a PK and find an SK for it
Identity-Based Encryption

In PKE, KeyGen produces a random (PK,SK) pair

Can I have a “fancy public-key” (e.g., my name)?

But no one should be able to pick a PK and find an SK for it

But suppose a trusted authority for key generation
Identity-Based Encryption

- In PKE, KeyGen produces a random (PK,SK) pair
- Can I have a “fancy public-key” (e.g., my name)?
  - But no one should be able to pick a PK and find an SK for it
- But suppose a trusted authority for key generation
- Then: Can it generate a valid (PK,SK) pair for any PK?
Identity-Based Encryption

- In PKE, KeyGen produces a random (PK,SK) pair
- Can I have a “fancy public-key” (e.g., my name)?
  - But no one should be able to pick a PK and find an SK for it
- But suppose a trusted authority for key generation
- Then: Can it generate a valid (PK,SK) pair for any PK?
- Identity-Based Encryption: a key-server (with a master secret-key) that can generate such pairs
Identity-Based Encryption

- In PKE, KeyGen produces a random (PK, SK) pair
- Can I have a “fancy public-key” (e.g., my name)?
  - But no one should be able to pick a PK and find an SK for it
- But suppose a trusted authority for key generation
  - Then: Can it generate a valid (PK, SK) pair for any PK?
- Identity-Based Encryption: a key-server (with a master secret-key) that can generate such pairs
  - Encryption will use the master public-key, and the receiver’s “identity” (i.e., fancy public-key)
Identity-Based Encryption

- In PKE, KeyGen produces a random (PK,SK) pair
- Can I have a “fancy public-key” (e.g., my name)?
  - But no one should be able to pick a PK and find an SK for it
- But suppose a trusted authority for key generation
- Then: Can it generate a valid (PK,SK) pair for any PK?
- Identity-Based Encryption: a key-server (with a master secret-key) that can generate such pairs
  - Encryption will use the master public-key, and the receiver’s “identity” (i.e., fancy public-key)
  - In PKE, sender has to retrieve PK for every party it wants to talk to (from a trusted public directory)
Identity-Based Encryption

- In PKE, KeyGen produces a random (PK,SK) pair
- Can I have a “fancy public-key” (e.g., my name)?
  - But no one should be able to pick a PK and find an SK for it
- But suppose a trusted authority for key generation
- Then: Can it generate a valid (PK,SK) pair for any PK?
- Identity-Based Encryption: a key-server (with a master secret-key) that can generate such pairs
  - Encryption will use the master public-key, and the receiver’s “identity” (i.e., fancy public-key)
  - In PKE, sender has to retrieve PK for every party it wants to talk to (from a trusted public directory)
  - In IBE, receiver has to obtain its SK from the authority
Identity-Based Encryption
Identity-Based Encryption

Security requirement for IBE (will skip formal statement):
Identity-Based Encryption

Security requirement for IBE (will skip formal statement):

- Environment/adversary can decide the ID of the honest parties (in the beginning, or (if supported) adaptively)
Identity-Based Encryption

Security requirement for IBE (will skip formal statement):

- Environment/adversary can decide the ID of the honest parties (in the beginning, or (if supported) adaptively)
- Adversary can adaptively request SK for any number of IDs (which are not used for honest parties)
Identity-Based Encryption

Security requirement for IBE (will skip formal statement):

- Environment/adversary can decide the ID of the honest parties (in the beginning, or (if supported) adaptively)
- Adversary can adaptively request SK for any number of IDs (which are not used for honest parties)
- "Semantic security" for encryption with the ID of honest parties (CPA: with no access to decryption)
Identity-Based Encryption

Security requirement for IBE (will skip formal statement):

- Environment/adversary can decide the ID of the honest parties (in the beginning, or (if supported) adaptively)
- Adversary can adaptively request SK for any number of IDs (which are not used for honest parties)
- "Semantic security" for encryption with the ID of honest parties (CPA: with no access to decryption)
- Or, CCA security: also gets (guarded) access to decryption for honest parties’ IDs
Identity-Based Encryption

Security requirement for IBE (will skip formal statement):

- Environment/adversary can decide the ID of the honest parties (in the beginning, or (if supported) adaptively)
- Adversary can adaptively request SK for any number of IDs (which are not used for honest parties)
- “Semantic security” for encryption with the ID of honest parties (CPA: with no access to decryption)
- Or, CCA security: also gets (guarded) access to decryption for honest parties’ IDs
- IBE (even CPA-secure) can easily give CCA-secure PKE!
Identity-Based Encryption

Security requirement for IBE (will skip formal statement):

- Environment/adversary can decide the ID of the honest parties (in the beginning, or (if supported) adaptively)
- Adversary can adaptively request SK for any number of IDs (which are not used for honest parties)
- “Semantic security” for encryption with the ID of honest parties (CPA: with no access to decryption)
- Or, CCA security: also gets (guarded) access to decryption for honest parties’ IDs

IBE (even CPA-secure) can easily give CCA-secure PKE!

IBE: Can’t malleate ciphertext for one ID into one for another
Identity-Based Encryption

Security requirement for IBE (will skip formal statement):
- Environment/adversary can decide the ID of the honest parties (in the beginning, or (if supported) adaptively)
- Adversary can adaptively request SK for any number of IDs (which are not used for honest parties)
- “Semantic security” for encryption with the ID of honest parties (CPA: with no access to decryption)
- Or, CCA security: also gets (guarded) access to decryption for honest parties’ IDs

IBE (even CPA-secure) can easily give CCA-secure PKE!

IBE: Can’t malleate ciphertext for one ID into one for another

\[ \text{PKE} \text{nc}_{\text{MPK}}(m) = (\text{verkey}, C=\text{IBE} \text{nc}_{\text{MPK}}(\text{id}=\text{verkey}; m), \text{sign}_{\text{signkey}}(C) ) \]
Identity-Based Encryption

Security requirement for IBE (will skip formal statement):

- Environment/adversary can decide the ID of the honest parties (in the beginning, or (if supported) adaptively)
- Adversary can adaptively request SK for any number of IDs (which are not used for honest parties)
- "Semantic security" for encryption with the ID of honest parties (CPA: with no access to decryption)
- Or, CCA security: also gets (guarded) access to decryption for honest parties' IDs

IBE (even CPA-secure) can easily give CCA-secure PKE!

- IBE: Can't malleate ciphertext for one ID into one for another
- \( \text{PKE}_{\text{MPK}}(m) = (\text{verkey}, C=\text{IBE}_{\text{MPK}}(\text{id}=\text{verkey}; m), \text{sign}_{\text{signkey}}(C) ) \)
Identity-Based Encryption
Identity-Based Encryption

Notion of IBE suggested by Shamir in 1984 (but no construction)
Identity-Based Encryption

- Notion of IBE suggested by Shamir in 1984 (but no construction)
Identity-Based Encryption

- Notion of IBE suggested by Shamir in 1984 (but no construction)
- But no formal proof of security
Identity-Based Encryption

- Notion of IBE suggested by Shamir in 1984 (but no construction)
  - But no formal proof of security
- Quadratic Residuosity based scheme by Cocks (2001)
Identity-Based Encryption

- Notion of IBE suggested by Shamir in 1984 (but no construction)

  - But no formal proof of security

- Quadratic Residuosity based scheme by Cocks (2001)
  - But long ciphertexts (Shorter, but slower scheme by Boneh-Gentry-Hamburg (2007) )
Identity-Based Encryption

- Notion of IBE suggested by Shamir in 1984 (but no construction)


  - But no formal proof of security

- Quadratic Residuosity based scheme by Cocks (2001)

  - But long ciphertexts (Shorter, but slower scheme by Boneh-Gentry-Hamburg (2007))

- Boneh-Franklin IBE (2001): similar to [SOK] ID-NIKD (but with a proof of security in the random oracle model)
Identity-Based Encryption

- Notion of IBE suggested by Shamir in 1984 (but no construction)

  - But no formal proof of security

- Quadratic Residuosity based scheme by Cocks (2001)
  - But long ciphertexts (Shorter, but slower scheme by Boneh-Gentry-Hamburg (2007))

- Boneh-Franklin IBE (2001): similar to [SOK] ID-NIKD (but with a proof of security in the random oracle model)

- Pairing-based, without RO: Boneh-Boyen (2004), Waters (2005), ...
Identity-Based Encryption

- Notion of IBE suggested by Shamir in 1984 (but no construction)
  - But no formal proof of security
- Quadratic Residuosity based scheme by Cocks (2001)
  - But long ciphertexts (Shorter, but slower scheme by Boneh-Gentry-Hamburg (2007))
- Boneh-Franklin IBE (2001): similar to [SOK] ID-NIKD (but with a proof of security in the random oracle model)
- Pairing-based, without RO: Boneh-Boyen (2004), Waters (2005), ...
- Without pairing: Using QR, Lattices, ...
Bilinear Pairing
Bilinear Pairing

A relatively new (and less understood) tool in cryptography
Bilinear Pairing

- A relatively new (and less understood) tool in cryptography

- Two (or three) groups with an efficient pairing operation, $e: G \times G \rightarrow G_T$ that is "bilinear"
Bilinear Pairing

- A relatively new (and less understood) tool in cryptography
- Two (or three) groups with an efficient pairing operation, $e: G \times G \rightarrow G_T$ that is "bilinear"
- Typically, prime order (cyclic) groups
Bilinear Pairing

A relatively new (and less understood) tool in cryptography

Two (or three) groups with an efficient **pairing operation**, $e: G \times G \rightarrow G_T$ that is “bilinear”

- Typically, prime order (cyclic) groups

  $e(g^a, h^b) = e(g, h)^{ab}$
Bilinear Pairing

- A relatively new (and less understood) tool in cryptography
- Two (or three) groups with an efficient pairing operation, $e: G \times G \rightarrow G_T$ that is "bilinear"
  - Typically, prime order (cyclic) groups
  - $e(g^a, h^b) = e(g, h)^{ab}$
  - Multiplication (once) in the exponent!
Bilinear Pairing

- A relatively new (and less understood) tool in cryptography
- Two (or three) groups with an efficient pairing operation, \( e: G \times G \rightarrow G_T \) that is “bilinear”
  - Typically, prime order (cyclic) groups
  - \( e(g^a, h^b) = e(g, h)^{ab} \)
  - Multiplication (once) in the exponent!
    - \( e(g^ag^{a'}, g^b) = e(g^a, g^b) e(g^{a'}, g^b) ; e(g^a, g^{bc}) = e(g^{ac}, g^b) ; \ldots \)
Bilinear Pairing

➤ A relatively new (and less understood) tool in cryptography

➤ Two (or three) groups with an efficient pairing operation, $e: G \times G \rightarrow G_T$ that is “bilinear”

➤ Typically, prime order (cyclic) groups

➤ $e(g^a, h^b) = e(g,h)^{ab}$

➤ Multiplication (once) in the exponent!

➤ $e(g^ag^{a'},g^b) = e(g^a,g^b) e(g^{a'},g^b)$ ; $e(g^a,g^{bc}) = e(g^{ac},g^b)$ ; ...

➤ Required to be not degenerate: $e(g,g) \neq 1$
Decisional Bilinear-Diffie-Hellman Assumption
Decisional Bilinear-Diffie-Hellman Assumption

- DDH is not hard in $G$, if there is a bilinear pairing
Decisional Bilinear-Diffie-Hellman Assumption

- DDH is not hard in $G$, if there is a bilinear pairing
  - Given $(g^a, g^b, g^z)$ check if $e(g^a, g^b) = e(g^z, g)$
Decisional Bilinear-Diffie-Hellman Assumption

- DDH is not hard in G, if there is a bilinear pairing
  - Given \((g^a, g^b, g^z)\) check if \(e(g^a, g^b) = e(g^z, g)\)

- **Decisional Bilinear DH assumption**: \((g^a, g^b, g^c, g^{abc})\) is indistinguishable from \((g^a, g^b, g^c, g^z)\). \((a, b, c, z\) random\)
IBE from Pairing
IBE from Pairing

\( \text{MPK: } g, h, Y = e(g, h)^y, \pi = (u, u_1, \ldots, u_n) \)
IBE from Pairing

- MPK: $g, h, Y = e(g, h)^y, \pi = (u, u_1, ..., u_n)$
- MSK: $h^y$
IBE from Pairing

MPK: \( g, h, Y = e(g, h)^y, \pi = (u, u_1, ..., u_n) \)

MSK: \( h^y \)

Enc(m; s) = \( (g^s, \pi(ID)^s, M.Y^s) \)
IBE from Pairing

- **MPK:** $g, h, Y = e(g, h)^y$, $\pi = (u, u_1, \ldots, u_n)$
- **MSK:** $h^y$
- **Enc(m,s):** $(g^s, \pi(ID)^s, M.Y^s)$

\[ \pi(ID) = u \prod_{i:ID_i=1} u_i \]
IBE from Pairing

- **MPK**: $g, h, Y = e(g, h)^y, \pi = (u, u_1, ..., u_n)$
- **MSK**: $h^y$
- **Enc**($m; s$) = ($g^s, \pi(ID)^s, M.Y^s$)
- **SK for ID**: ($g^t, h^y.\pi(ID)^t$) = ($d_1, d_2$)

\[ \pi(ID) = u \prod_{i: ID_i = 1} u_i \]
IBE from Pairing

- **MPK:** $g, h, Y = e(g, h)^y, \pi = (u, u_1, ..., u_n)$
- **MSK:** $h^y$
- **Enc(m;s):** $(g^s, \pi(ID)^s, M.Y^s)$
- **SK for ID:** $(g^t, h^y.\pi(ID)^t) = (d_1, d_2)$
- **Dec:** $(a, b, c; d_1, d_2) = c / [e(a, d_2) / e(b, d_1)]$

\[ \pi(ID) = u \prod_{i:ID_i=1} u_i \]
IBE from Pairing

- **MPK**: $g, h, Y = e(g, h)^y$, $\pi = (u, u_1, ..., u_n)$
- **MSK**: $h^y$
- **Enc**($m; s$) = ($g^s$, $\pi(ID)^s$, $M.Y^s$)
- **SK for ID**: ($g^t$, $h^y.\pi(ID)^t$) = ($d_1$, $d_2$)
- **Dec** ($a, b, c; d_1, d_2$) = $c / [ e(a,d_2) / e(b,d_1) ]$
- CPA security based on Decisional-BDH
Attribute-Based Encryption
Attribute-Based Encryption

Which users can decrypt a ciphertext will be decided by the attributes and policies associated with the message and the user.
Attribute-Based Encryption

- Which users can decrypt a ciphertext will be decided by the attributes and policies associated with the message and the user.

- A central authority will create secret keys for the users (like in IBE) based on attributes/policies for each user.
Attribute-Based Encryption

- Which users can decrypt a ciphertext will be decided by the attributes and policies associated with the message and the user.

- A central authority will create secret keys for the users (like in IBE) based on attributes/policies for each user.

- Ciphertexts can be created (by anyone) by incorporating attributes/policies.
Ciphertext-Policy ABE
Ciphertext-Policy ABE

Users in the system have attributes; receives a key (or “key bundle”) from an authority for its set of attributes
Ciphertext-Policy ABE

- Users in the system have attributes; receives a key (or “key bundle”) from an authority for its set of attributes

- Ciphertext contains a policy (a boolean predicate over the attribute space)
Ciphertext-Policy ABE

- Users in the system have attributes; receives a key (or "key bundle") from an authority for its set of attributes.

- Ciphertext contains a policy (a boolean predicate over the attribute space).

- If a user’s attribute set satisfies the policy, can use its key bundle to decrypt the ciphertext.
Ciphertext-Policy ABE

Users in the system have attributes; receives a key (or “key bundle”) from an authority for its set of attributes.

Ciphertext contains a policy (a boolean predicate over the attribute space).

If a user’s attribute set satisfies the policy, can use its key bundle to decrypt the ciphertext.

Multiple users cannot pool their attributes together.
Ciphertext-Policy ABE

- Users in the system have attributes; receives a key (or “key bundle”) from an authority for its set of attributes.

- Ciphertext contains a policy (a boolean predicate over the attribute space).

- If a user’s attribute set satisfies the policy, can use its key bundle to decrypt the ciphertext.

  - Multiple users cannot pool their attributes together.

- Application: End-to-End privacy in Attribute-Based Messaging.
Key-Policy ABE
Key-Policy ABE

- Attributes will be assigned to a ciphertext (when creating the ciphertext)
Key-Policy ABE

- Attributes will be assigned to a ciphertext (when creating the ciphertext)
- Policies will be assigned to users/keys by an authority (who creates the keys)
Key-Policy ABE

- Attributes will be assigned to a ciphertext (when creating the ciphertext)

- Policies will be assigned to users/keys by an authority (who creates the keys)

- A key can decrypt only those ciphertexts whose attributes satisfy the policy
Key-Policy ABE

- Attributes will be assigned to a ciphertext (when creating the ciphertext)

- Policies will be assigned to users/keys by an authority (who creates the keys)

  - A key can decrypt only those ciphertexts whose attributes satisfy the policy

- E.g. Applications
Key-Policy ABE

- Attributes will be assigned to a ciphertext (when creating the ciphertext)
- Policies will be assigned to users/keys by an authority (who creates the keys)
- A key can decrypt only those ciphertexts whose attributes satisfy the policy
- E.g. Applications
  - Fuzzy IBE: use a policy that allows receiver’s ID to be slightly different from an ID specified in the policy
Key-Policy ABE

- Attributes will be assigned to a ciphertext (when creating the ciphertext)

- Policies will be assigned to users/keys by an authority (who creates the keys)

  - A key can decrypt only those ciphertexts whose attributes satisfy the policy

- E.g. Applications

  - Fuzzy IBE: use a policy that allows receiver’s ID to be slightly different from an ID specified in the policy

  - Audit log inspection: grant auditor authority to read only messages with certain attributes
A KP-ABE Scheme
A KP-ABE Scheme

A construction that supports “linear policies”
A KP-ABE Scheme

- A construction that supports "linear policies"

- Policy corresponds to a (monotonic) access structure (sets of attributes that when pooled satisfy the policy)
A KP-ABE Scheme

- A construction that supports “linear policies”
- Policy corresponds to a (monotonic) access structure (sets of attributes that when pooled satisfy the policy)
- Linear: Matrix L with each row labeled by an attribute, such that a set of attributes S satisfies the policy iff
A KP-ABE Scheme

- A construction that supports “linear policies”

- Policy corresponds to a (monotonic) access structure (sets of attributes that when pooled satisfy the policy)

- **Linear**: Matrix $L$ with each row labeled by an attribute, such that a set of attributes $S$ satisfies the policy iff

  - there is a vector $v$ such that $v.L = [1 \ 1 \ \ldots \ 1]$
A KP-ABE Scheme

- A construction that supports “linear policies”

- Policy corresponds to a (monotonic) access structure (sets of attributes that when pooled satisfy the policy)

- Linear: Matrix $L$ with each row labeled by an attribute, such that a set of attributes $S$ satisfies the policy iff

  - there is a vector $v$ such that $v.L = [1 \ 1 \ \ldots \ 1]$?

  - and, labels corresponding to non-zero entries of $v$ are all contained in $S$
A KP-ABE Scheme

A construction that supports “linear policies”

Policy corresponds to a (monotonic) access structure (sets of attributes that when pooled satisfy the policy)

Linear: Matrix $L$ with each row labeled by an attribute, such that a set of attributes $S$ satisfies the policy iff

- there is a vector $v$ such that $v.L = [1 \ 1 \ \ldots \ 1]$
- and, labels corresponding to non-zero entries of $v$ are all contained in $S$

Linear algebra over some finite field (e.g. $GF(p)$)
A KP-ABE Scheme

- A construction that supports “linear policies”

- Policy corresponds to a (monotonic) access structure (sets of attributes that when pooled satisfy the policy)

- **Linear**: Matrix L with each row labeled by an attribute, such that a set of attributes S satisfies the policy iff
  - there is a vector v such that \( v.L = [1 \ 1 \ ... \ 1] \)
  - and, labels corresponding to non-zero entries of v are all contained in S

- Linear algebra over some finite field (e.g. GF(p) )

- For efficiency need a small matrix
Example of a “Linear Policy”
Example of a “Linear Policy”

Consider this policy, over 7 attributes
Example of a “Linear Policy”

Consider this policy, over 7 attributes
Example of a “Linear Policy”

Consider this policy, over 7 attributes

$L$: OR AND AND OR
Example of a “Linear Policy”

Consider this policy, over 7 attributes

L:

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

OR

AND

AND

AND

OR
Example of a “Linear Policy”

Consider this policy, over 7 attributes

L:

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Can allow threshold gates too
A KP-ABE Scheme
A KP-ABE Scheme

MPK: $g, Y = e(g, g)^y, T = (g^{t_1}, ..., g^{t_n})$ (n attributes)
A KP-ABE Scheme

- **MPK:** \( g, Y = e(g, g)^y, T = (g^{t_1}, \ldots, g^{t_n}) \) (n attributes)
- **MSK:** \( y \) and \( t_a \) for each attribute \( a \)
A KP-ABE Scheme

- MPK: $g, Y = e(g, g)^y, T = (g^{t_1},..., g^{t_n})$ (n attributes)
- MSK: $y$ and $t_a$ for each attribute $a$
- Enc($m, A; s$) = $(A, \{ T_a^s \}_{a \in A}, M.Y^s)$
A KP-ABE Scheme

- MPK: \( g, Y = e(g, g)^y \), \( T = (g^{t_1}, \ldots, g^{t_n}) \) (n attributes)
- MSK: \( y \) and \( t_a \) for each attribute \( a \)
- \( \text{Enc}(m, A; s) = (A, \{T_a^s\}_{a \in A}, M.Y^s) \)

SK for policy \( L \) (with \( d \) rows): Let \( u = (u_1 \ldots u_d) \) s.t. \( \sum_i u_i = y \). For each row \( i \), let \( x_i = <L_i, u>/t_{\text{label}(i)} \). Let \( \text{Key} \ X = \{g^{x_i}\}_{i=1}^d \)
A KP-ABE Scheme

- **MPK**: $g, Y = e(g, g)^y$, $T = (g^{t_1}, ..., g^{t_n})$ (n attributes)
- **MSK**: $y$ and $t_a$ for each attribute $a$
- **Enc**(m,A;s) = ( A, $\{ T_a^s \}_{a \in A}$, M.Y$^s$)
- **SK** for policy L (with d rows): Let $u = (u_1 ... u_d)$ s.t. $\Sigma_i u_i = y$. For each row $i$, let $x_i = <L_i, u>/t_{\text{label}(i)}$. Let Key $X = \{ g^{x_i} \}_{i=1 \text{ to } d}$
- **Dec** ((A,{$Z_a$}$\{a \in A, c\}$; {$X_i$}row $i$)) : Get $Y^s = \prod_{i:\text{label}(i) \in A} e(Z_{\text{label}(i)}, X_i)^{V_i}$
  where $v = [v_1 ... v_d]$ s.t. $v_i=0$ if label(i) $\notin A$, and $v_L=[1...1]$
A KP-ABE Scheme

MPK: $g, Y = e(g, g)^y, T = (g^{t_1}, ..., g^{t_n})$ (n attributes)

MSK: $y$ and $t_a$ for each attribute $a$

Enc$(m, A; s) = (A, \{ T_a^s \}_{a \in A}, M.Y^s)$

SK for policy $L$ (with d rows): Let $u=(u_1 ... u_d)$ s.t. $\Sigma_i u_i = y$. For each row $i$, let $x_i = <L_i, u>/t_{\text{label}(i)}$. Let Key $X = \{ g^{x_i} \}_{i=1 \text{ to } d}$

Dec $( (A, \{ Z_a \}_{a \in A}, c); \{ X_i \}_{\text{row } i} )$: Get $Y^s = \prod_{i: \text{label}(i) \in A} e(Z_{\text{label}(i)}, X_i)^{v_i}$ where $v = [v_1 ... v_d]$ s.t. $v_i=0$ if $\text{label}(i) \notin A$, and $v_L=[1...1]$

CPA security based on Decisional-BDH
A KP-ABE Scheme

- **MPK**: \( g, Y=e(g,g)^y, T = (g^{t_1}, ..., g^{t_n}) \) (n attributes)
- **MSK**: \( y \) and \( t_a \) for each attribute \( a \)
- **Enc**(\( m,A;s \)) = ( \( A, \{ T_a^s \}_{a \in A}, M.Y^s \) )
- **SK** for policy \( L \) (with \( d \) rows): Let \( u=(u_1 ... u_d) \) s.t. \( \Sigma_i u_i = y \). For each row \( i \), let \( x_i = \langle L_i,u \rangle/t_{\text{label}(i)} \). Let Key \( X = \{ g^{x_i} \}_{i=1 \text{ to } d} \)
- **Dec** (\( (A,\{Z_a\}_{a \in A},c); \{X_i\}_{\text{row } i} \)) : Get \( Y^s = \prod_{i: \text{label}(i) \in A} e(Z_{\text{label}(i)},X_i)^{V_i} \) where \( v = [v_1 ... v_d] \) s.t. \( v_i=0 \) if \( \text{label}(i) \notin A \), and \( v_L=[1...1] \)

**CPA security based on Decisional-BDH**

- Choosing a random vector \( u \) for each key helps in preventing collusion
Predicate Encryption
Predicate Encryption

Similar to ABE, but the ciphertext hides the attributes/policy
Predicate Encryption

- Similar to ABE, but the ciphertext hides the attributes/policy.
- Decryption reveals only whether a condition is satisfied by the ciphertext, and if it is, reveals the message too.
Predicate Encryption

- Similar to ABE, but the ciphertext hides the attributes/policy.
- Decryption reveals only whether a condition is satisfied by the ciphertext, and if it is, reveals the message too.
- e.g.: ciphertext contains a vector \(c\), and key a vector \(d\). Predicate: whether \(<c,d> = 0\) or not.
Predicate Encryption

Similar to ABE, but the ciphertext hides the attributes/policy

Decryption reveals only whether a condition is satisfied by the ciphertext, and if it is, reveals the message too

e.g.: ciphertext contains a vector $c$, and key a vector $d$. Predicate: whether $\langle c, d \rangle = 0$ or not

A building block for other predicates
Predicate Encryption

Similar to ABE, but the ciphertext hides the attributes/policy

Decryption reveals only whether a condition is satisfied by the ciphertext, and if it is, reveals the message too

e.g.: ciphertext contains a vector $c$, and key a vector $d$. Predicate: whether $\langle c, d \rangle = 0$ or not

A building block for other predicates

Constructions based on the Decision Linear assumption
Predicate Encryption

Similar to ABE, but the ciphertext hides the attributes/policy.

Decryption reveals only whether a condition is satisfied by the ciphertext, and if it is, reveals the message too.

E.g.: ciphertext contains a vector $c$, and key a vector $d$. Predicate: whether $<c,d> = 0$ or not.

A building block for other predicates.

Constructions based on the Decision Linear assumption

$(f,g,h,f^x,g^y,h^{x+y})$ and $(f,g,h,f^x,g^y,h^z)$ indistinguishable for random $f, g, h, x, y, z$. 
Attribute-Based Signatures
Attribute-Based Signatures

“Claim-and-endorse”: Claim to have attributes satisfying a certain policy, and sign a message
Attribute-Based Signatures

“Claim-and-endorse”: Claim to have attributes satisfying a certain policy, and sign a message

Soundness: can’t forge, even by colluding
Attribute-Based Signatures

“Claim-and-endorse”: Claim to have attributes satisfying a certain policy, and sign a message

Soundness: can’t forge, even by colluding

Hiding: Doesn’t reveal how the policy was satisfied (beyond what is implied by the fact that it was)
Attribute-Based Signatures

“Claim-and-endorse”: Claim to have attributes satisfying a certain policy, and sign a message

Soundness: can’t forge, even by colluding

Hiding: Doesn’t reveal how the policy was satisfied (beyond what is implied by the fact that it was)

Also unlinkable: cannot link multiple signatures as originating from the same signer
An ABS Construction
An ABS Construction

Using “Credential Bundles” and NIZK proofs (in fact, NIWI proofs)
An ABS Construction

- Using “Credential Bundles” and NIZK proofs (in fact, NIWI proofs)
- Credential Bundle for a set of attributes:
An ABS Construction

- Using “Credential Bundles” and NIZK proofs (in fact, NIWI proofs)

- **Credential Bundle** for a set of attributes:
  - Security: Given multiple credential bundles, can’t create a credential bundle for a new set, unless it is a subset of attributes in a single given credential bundle
An ABS Construction

Using “Credential Bundles” and NIZK proofs (in fact, NIWI proofs)

Credential Bundle for a set of attributes:

- Security: Given multiple credential bundles, can’t create a credential bundle for a new set, unless it is a subset of attributes in a single given credential bundle

- Map each (claim, message) to a “pseudo-attribute”
An ABS Construction

Using “Credential Bundles” and NIZK proofs (in fact, NIWI proofs)

**Credential Bundle** for a set of attributes:

- **Security**: Given multiple credential bundles, can’t create a credential bundle for a new set, unless it is a subset of attributes in a single given credential bundle

- Map each (claim,message) to a “pseudo-attribute”

- **Signing key**: credential bundle for (real) attributes possessed
An ABS Construction

Using “Credential Bundles” and NIZK proofs (in fact, NIWI proofs)

Credential Bundle for a set of attributes:

- Security: Given multiple credential bundles, can’t create a credential bundle for a new set, unless it is a subset of attributes in a single given credential bundle

- Map each (claim,message) to a “pseudo-attribute”

- Signing key: credential bundle for (real) attributes possessed

- Signature: a NIZK proof of knowledge of a credential-bundle for attributes satisfying the claim, or a credential for the pseudo-attribute corresponding to (claim,message)
An ABS Construction

Using “Credential Bundles” and NIZK proofs (in fact, NIWI proofs)

Credential Bundle for a set of attributes:

- Security: Given multiple credential bundles, can’t create a credential bundle for a new set, unless it is a subset of attributes in a single given credential bundle

- Map each (claim, message) to a “pseudo-attribute”

- Signing key: credential bundle for (real) attributes possessed

- Signature: a NIZK proof of knowledge of a credential-bundle for attributes satisfying the claim, or a credential for the pseudo-attribute corresponding to (claim, message)

Today
Today

IBE, ABE and ABS
Today

IBE, ABE and ABS

Pairing-based cryptography
Today

- IBE, ABE and ABS
- Pairing-based cryptography
- Next up:
Today

IBE, ABE and ABS

Pairing-based cryptography

Next up:

Some more applications of pairing-based cryptography
Today

- IBE, ABE and ABS
- Pairing-based cryptography
- Next up:
  - Some more applications of pairing-based cryptography
  - Generic groups