Attribute-based encryption (ABE): more fine-grained control to encrypted data

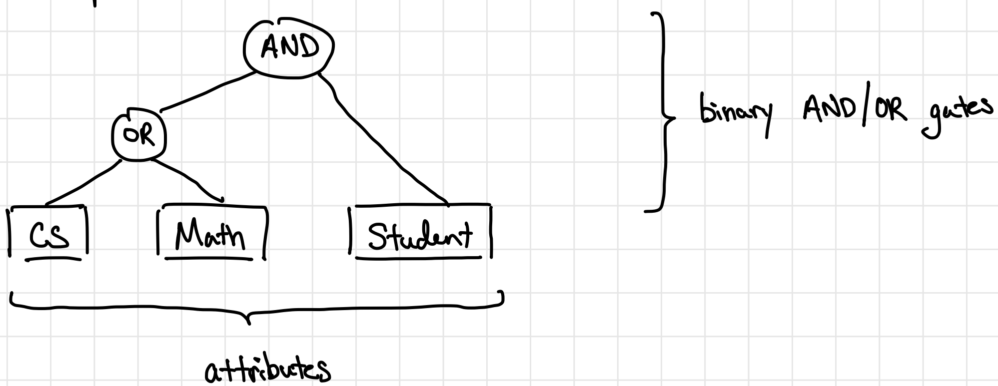
IBE: can encrypt to an identity (policy is basically checking equality)
ABE: can encrypt to general policies

Key-policy ABE: ciphertexts are associated with attributes
secret keys are associated with policy

<u>Example</u>: attribute could be a classification level (unclassified, secret, top secret)
policy could be (unclassified or secret)

attribute could describe a role (e.g., CS, math, physics, etc.)
policy could be (CS or math)

We will describe policies as a Boolean formula:



In a formula, every gate has fan-out 1 (each output of a gate can only be used once)

Will assume a polynomial number of attributes today

We will label the attributes by integers $1, 2, \ldots, n$

Goyal - Pandey - Sahai - Waters ABE scheme:

Setup: For each $i \in [n]$, sample $t_i \xleftarrow{R} \mathbb{Z}_p$
Sample $\gamma \xleftarrow{R} \mathbb{Z}_p$

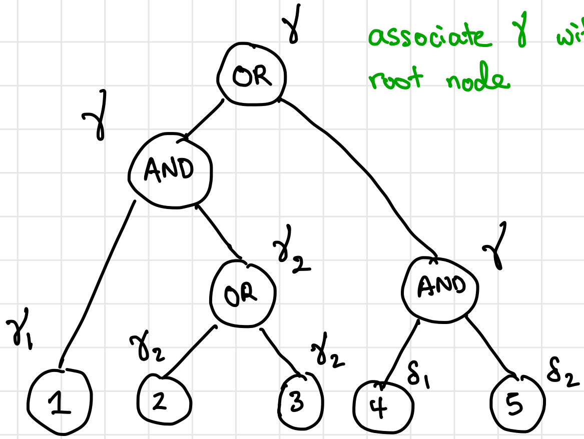
Output attribute keys $T_1 = g^{t_1}, \ldots, T_n = g^{t_n}, \quad h = e(g,g)^{\gamma}$

$$mpk = (T_1, \ldots, T_n, h) \qquad msk = (t_1, \ldots, t_n, \gamma)$$

Encrypt (mpk, S, m): On input $mpk = (T_1, \ldots, T_n, h)$ and a set of attributes $S \subseteq [n]$, and the message $m \in \mathbb{G}_T$, sample $r \xleftarrow{R} \mathbb{Z}_p$ and output

$$ct = (S, T_i^r, h^r \cdot m)$$

KeyGen (msk, P): We will start with an example, and formalize later. Idea is we will secret share the "secret key" γ according to the policy P.



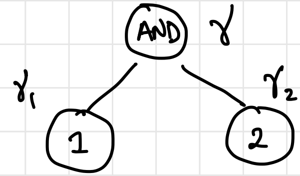
associate γ with root node

attributes

$$\gamma_1, \gamma_2 \xleftarrow{R} \mathbb{Z}_p \quad \text{s.t.} \quad \gamma_1 + \gamma_2 = \gamma$$

$$\delta_1, \delta_2 \xleftarrow{R} \mathbb{Z}_p \quad \text{s.t.} \quad \delta_1 + \delta_2 = \gamma$$

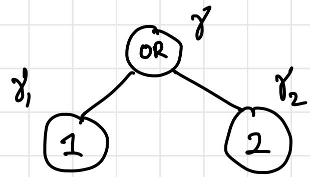
for AND gate



sample $\gamma_1, \gamma_2 \xleftarrow{R} \mathbb{Z}_p$ such that $\gamma = \gamma_1 + \gamma_2$

need both γ_1 <u>and</u> γ_2 to obtain γ

for OR gate



set $\gamma_1 = \gamma_2 = \gamma$

either γ_1 <u>or</u> γ_2 needed to obtain γ

We refer to the exponents associated with the leaf nodes as the shares of the master secret key associated with the attribute.

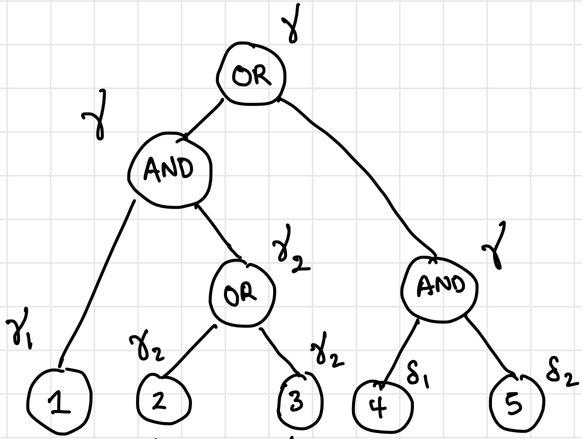
<u>Key for policy P</u>:

1. Secret share γ according to policy P
2. Let $\gamma_i \in \mathbb{Z}_p$ be the share associated with attribute i
3. The secret key for P is the set

   g^{γ_i / t_i} for each attribute i that appears in the policy

   along with the policy P

Decryption (we will defer formal description to later)



$\gamma_1 + \gamma_2 = \gamma$

$\delta_1 + \delta_2 = \gamma$

g^{γ_1/t_1} $\quad g^{\gamma_2/t_2}$ $\quad g^{\gamma_2/t_3}$ $\quad g^{\delta_1/t_4}$ $\quad g^{\delta_2/t_5}$ [attribute keys]

write each as K_i for each attribute appearing in the policy

ciphertext : $S, \{T_i^r\}_{r \in S}, \; h^r \cdot m$

$$T_i^r = g^{rt_i}, \qquad h^r \cdot m = e(g,g)^{r\gamma} \cdot m$$

Suppose ciphertext has attributes (2,4,5)

ciphertext contains $g^{rt_2}, g^{rt_4}, g^{rt_5}$

user can pair $e(K_2, g^{rt_2}) = e(g^{\gamma_2/t_2}, g^{rt_2}) = e(g,g)^{\gamma_2 r}$

can similarly obtain $e(g,g)^{\delta_1 r}$ and $e(g,g)^{\delta_2 r}$

↳ Observe that exponents $\gamma_2 r, \delta_1 r, \delta_2 r$ are secret shares of γr. If the attributes satisfy the policy, can use shares to reconstruct $e(g,g)^{\gamma r}$ and decrypt. This relies on fact that secret sharing scheme has a linear reconstruction algorithm.

Summary of approach: ElGamal-style encryption

secret key: γ
public key: $h = e(g,g)^{\gamma}$

ciphertext includes $h^{r} \cdot m$

Normally in ElGamal, ciphertext contains g^{r} and γ is used to decrypt.
With pairings, we can also use g^{γ} as the decryption key.
↳ But cannot just give out g^{γ}.

Instead: secret key contain secret share of g^{γ} according to policy P, but need a way to combine with attributes.

Approach: set $K_i = g^{d_i / t_i}$ where d_i is a share of γ and t_i is the attribute key.

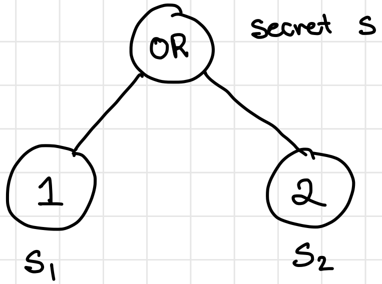
Pair with $g^{t_i r}$ from ciphertext to obtain share of $e(g,g)^{\gamma r}$.

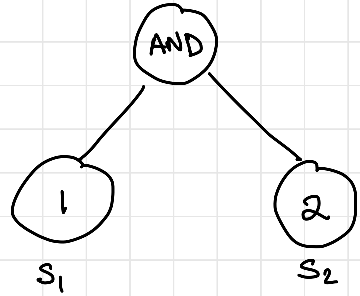
Secret shares for different keys are independent (no mixing and matching)

To describe the scheme more generally, we describe a linear secret sharing schem

Suppose we have a secret s. For each attribute t, we can associate with it a share s_t. Given a policy P and shares $\{s_t\}_{t \in T}$, if $P(T) = 1$, then should be able to recover s.

Idea: secret share gate-by-gate:

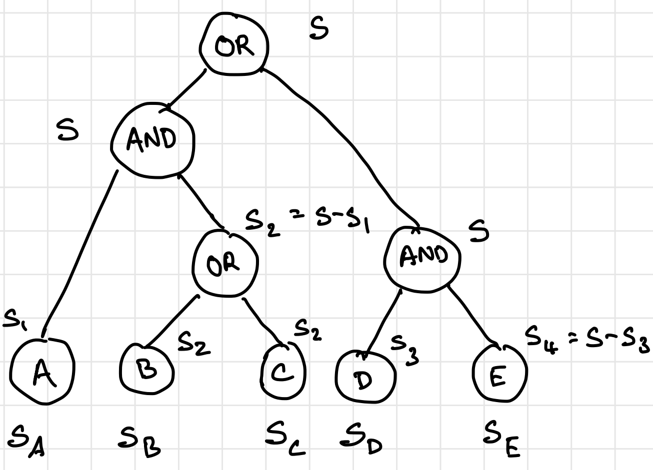




recover s if you have either s_1 or s_2
↳ set $s_1 = s_2 = s$

recover s if you have s_1 and s_2
↳ sample $s_1 \xleftarrow{R} \mathbb{F}_p$ and set $s_2 = s - s_1$
↳ reconstruct by computing $s_1 + s_2 = s$
↳ no information given out with just 1 share

Compose to support general policies



$s_1, s_3 \xleftarrow{R} \mathbb{Z}_p$

$s_A = s_1$
$s_B = s_C = s - s_1$
$s_D = s_3$
$s_E = s - s_3$

For the policy, we can associate with it a "share generating" matrix

For an AND gate, we can secret share s by sampling $\alpha \xleftarrow{R} \mathbb{F}_p$ and setting $s_1 = s + \alpha$ and $s_2 = -\alpha$:

$$\begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} s \\ \alpha \end{bmatrix}$$

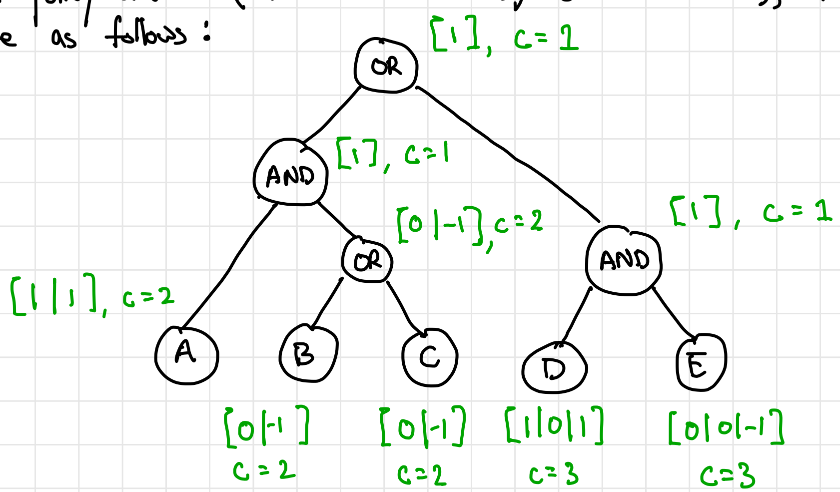
For an OR gate, we secret share s by setting $s_1 = s_2 = s$

$$\begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} s \end{bmatrix}$$

General procedure:

1. Associate vector $[1]$ with root node and initialize counter $c \leftarrow 1$
2. Proceed down the tree.
   - If a node is an OR with label v, the children are associated with label v (i.e., they are identical to the parent).
   - If a node is an AND with label v, we label the children with label $v \| 0^{\text{len}(v)-c} \| 1$ and the other label with $0^c \| -1$. Observe that the sum of the shares is $v \| 0^{\text{len}(v)-c}$. Finally, increment $c \leftarrow c+1$.
3. Pad all vectors with 0s to dimension c

<u>Example</u>: For policy above $(A \text{ AND } (B \text{ OR } C))$ OR $(D \text{ AND } E)$, the vectors are as follows:



Share-generation matrix:

same shares we had before!

$$\begin{array}{c} S_A \\ S_B \\ S_C \\ S_D \\ S_E \end{array} \underbrace{\begin{bmatrix} 1 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & -1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & -1 \end{bmatrix}}_{M_P} \begin{bmatrix} s \\ \alpha_1 \\ \alpha_2 \end{bmatrix} = \begin{bmatrix} s + \alpha_1 \\ -\alpha_1 \\ -\alpha_1 \\ s + \alpha_2 \\ -\alpha_2 \end{bmatrix}$$

<u>Main observation</u>: if attributes satisfy policy, then vector $e_1^T = [1, 0, \ldots, 0]$ will be in the row span of M_P

can show inductively: for every node v in the tree, if attributes satisfy the node v, then vector associated with v is in the row span of M_P

true for every leaf node and inductive step follows by construction

<u>Converse also holds</u>: if attributes do not satisfy policy, then e_1^T is not in the row span of M_P

Can express Boolean formula access structure as a linear secret sharing scheme

Boolean formula policy P on k attributes
$\longrightarrow$ share generation matrix M with k rows

If a set of attributes $S \subseteq [k]$ satisfies the policy, then there exists a vector $v \in \mathbb{Z}_p^k$ where $v_i = 0$ for all $i \notin S$ such that

$$v^T S = [1 \ 0 \ \cdots \ 0]$$

(i.e., the first basis vector is in spanned by the rows associated with the attributes in S).

If S does <u>not</u> satisfy the policy, then $[1 \ 0 \ \cdots \ 0]$ is not spanned by the rows of S.

↳ Suppose $v_1, \ldots, v_n$ are the vectors associated with S
Since $e_1 \notin \text{span}(v_1, \ldots, v_n)$, it must be the case that e_1 has a component in the orthogonal complement of $\{v_1, \ldots, v_n\}$
Thus, there exists a vector w such that $w^T v_i = 0$ and $w^T e_1 \neq 0$.
Without loss of generality, we assume the first component of w is 1.

$\quad\quad$ affribute universe n

Setup (n): Sample $t_1, \ldots, t_n \xleftarrow{R} \mathbb{Z}_p$ $\quad$ Set $T_i = g^{t_i}$

$\gamma \xleftarrow{R} \mathbb{Z}_p$ $\quad\quad$ $h = e(g,g)^{\gamma}$

Set $mpk = (T_1, \ldots, T_n, h)$

Set $msk = (t_1, \ldots, t_n, \gamma)$

Encrypt (mpk, S, m): sample $r \xleftarrow{R} \mathbb{Z}_p$ and set $ct = (\{(i, T_i^r)\}_{i \in S}, h^r \cdot m)$

KeyGen (msk, M): here $M \in \mathbb{Z}_p^{k \times \ell}$ is the share-generating matrix associated with the policy

sample $\alpha_1, \ldots, \alpha_{\ell-1} \xleftarrow{R} \mathbb{Z}_p$ and compute the shares of γ:

$$\begin{bmatrix} s_1 \\ \vdots \\ s_k \end{bmatrix} = M \cdot \begin{bmatrix} \gamma \\ \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_{\ell-1} \end{bmatrix}$$

$\rho(1), \ldots, \rho(t)$ are the attributes the policy is looking at

let $\rho : [k] \to [n]$ be the mapping from the row index of M to the attribute index in $[n]$.

output $sk = M, \{(i, g^{s_i / t_{\rho(i)}})\}_{i \in [k]}$

Decrypt (sk, ct): Parse $sk = (M, \{(i, D_i)\}_{i \in [k]})$

$$ct = (\{(i, C_i)\}_{i \in S}, Z)$$

If S satisfies the policy, then there exists a vector $v \in \mathbb{Z}_p^k$ such that $v^T M = [1, 0, \ldots, 0]$ and $v_i = 0$ for all $\rho(i) \in S$

Compute $$Z \Big/ \prod_{i \in [t]} e(C_{\rho(i)}, D_i)^{v_i}$$

[Note: treat $C_{\rho(i)} = g^0$ if $\rho(i) \notin S$]

Correctness: $\prod_{i\in[t]} e(C_{\rho(i)}, D_i)^{v_i} = \prod_{i\in[t]} e\left(g^{r t_{\rho(i)}}, g^{s_i/t_{\rho(i)}}\right)^{v_i}$

[holds since $v_i = 0$ when $\rho(i) \notin S$]

$\sum s_i v_i = v^T s = v^T M \cdot \begin{bmatrix} \gamma \\ \alpha_1 \\ \vdots \\ \alpha_{\ell-1} \end{bmatrix}$

$$= \prod_{i\in[t]} e(g,g)^{r s_i v_i}$$

$$= e(g,g)^{r v^T M \cdot \begin{bmatrix} \gamma \\ \alpha_1 \\ \vdots \\ \alpha_{\ell-1} \end{bmatrix}} = e(g,g)^{\gamma r} = h^r$$

Security: We will show that ciphertext

$$\{(i, T_i^r)\}_{i\in S},\ h^r \cdot m$$

is indistinguishable from

$$\{(i, T_i^r)\}_{i\in S},\ e(g,g)^\alpha \quad \text{where } \alpha \xleftarrow{R} \mathbb{Z}_p$$

We will consider selective security where adversary has to choose S ahead of time (before seeing public key).

Hardness will rely on DBDH. $e(g,g)^{abc}$ or $e(g,g)^\alpha$

DBDH challenge: $(g, \underset{g^a}{\underset{\shortparallel}{A}}, \underset{g^b}{\underset{\shortparallel}{B}}, \underset{g^c}{\underset{\shortparallel}{C}}, T)$

Reduction strategy:

Algorithm A chooses a set of attributes $S \subseteq [n]$

We need to simulate the public key mpk and secret keys for policies that S does not satisfy

Approach: Challenge ciphertext will use T $\leftarrow e(g,g)^{\gamma r}$

If $T = e(g,g)^{abc}$, we will need to map abc to γr and also publish $e(g,g)^{\gamma}$ in public key

Reduction will use c for the exponent r and ab for the exponent γ. Then,

$$e(g,g)^{\gamma} = e(g,g)^{ab} = e(g^a, g^b)$$
$$= e(A,B)$$

For the attribute-specific components, if $i \in S$, pick $t_i \xleftarrow{R} \mathbb{Z}_p$ and set $T_i = g^{t_i}$ must know t_i to simulate ciphertext components

If $i \notin S$, will need to pick carefully

In the ciphertext, reduction can set $C_i = C^{t_i} = g^{t_i c}$

Suffices to construct secret keys. Problem: do not (and cannot know secret key ab)

Let $M \in \mathbb{Z}_p^{k \times \ell}$ be share-generation matrix associated with a policy.
Let $\rho : [k] \to [n]$ be labeling function.

Normally, we would compute

$$\begin{bmatrix} s_1 \\ \vdots \\ s_k \end{bmatrix} = M \cdot \begin{bmatrix} ab \\ \alpha_1 \\ \vdots \\ \alpha_\ell \end{bmatrix} = \begin{bmatrix} — & m_1^T & — \\ & \vdots & \\ — & m_k^T & \longrightarrow \end{bmatrix} \begin{bmatrix} ab \\ \alpha_1 \\ \vdots \\ \alpha_\ell \end{bmatrix}$$

and give out $D_i = g^{s_i / t_{\rho(i)}}$ for each $i \in [k]$

<u>Problem:</u> Each s_i is a linear function of ab and we do <u>not</u> have g^{ab}

Let $u = \begin{bmatrix} ab \\ \alpha_1 \\ \vdots \\ \alpha_\ell \end{bmatrix}$. Normally we compute $s = Mu$.

We will instead write $u = b \underbrace{\begin{bmatrix} 0 \\ \alpha_1 \\ \vdots \\ \alpha_\ell \end{bmatrix}}_{z \,:\, \text{share of } 0} + ab \cdot w$

recall first component of w is 1 and w is orthogonal to rows of M corresponding to attributes in S
[$m_i^T w = 0$ if $\rho(i) \in S$]

Same distribution as before!

Consider two possibilities for D_i:

1) if $\rho(i) \in S$: $s_i = m_i^T u = b\, m_i^T z$

$$D_i = g^{s_i / t_{\rho(i)}} = g^{b m_i^T z / t_{\rho(i)}} = \underline{B^{m_i^T z / t_{\rho(i)}}}$$

reduction can compute since it knows M, z and $t_{\rho(i)}$

2) if $\rho(i) \notin S$: $s_i = m_i^T u = b\, m_i^T z + ab\, m_i^T w$
$= b\,(m_i^T z + a\, m_i^T w)$

$$D_i = g^{s_i / t_{\rho(i)}} = g^{b(m_i^T z + a\, m_i^T w) / t_{\rho(i)}}$$

Seems to rely on knowledge of ab.
But... we have one degree of freedom. We can pick $t_{\rho(i)}$ strategically here!

Reduction does <u>not</u> need to know $t_{\rho(i)}$ since these are <u>not</u> present in challenge ciphertext. It just needs to give out $T_{\rho(i)} = g^{t_{\rho(i)}}$ in public parameters.

<u>Approach</u>: choose T_i for $i \notin S$ to force cancellation.

Set $T_i := B^{d_i}$ where $d_i \xleftarrow{R} \mathbb{Z}_p$ in public parameters.

$$B^{d_i} = g^{b d_i}$$

Then, in the secret key for $\rho(i) \notin S$

$$D_i = g^{s_i / t_{\rho(i)}} = g^{b(m_i^T z + a m_i^T w) / t_{\rho(i)}}$$

$$= g^{m_i^T z + a m_i^T w / d_i}$$

$$= \underline{g^{m_i^T z / d_i} A^{m_i^T w / d_i}}$$

reduction can compute since it knows M, z, w, and d_i