Predicate Encryption Systems
No Query Left Unanswered

Summary of a Ph.D. Thesis presented at the Università di Salerno

Vincenzo Iovino

Dipartimento di Informatica, Università di Salerno, Italia.
iovino@dia.unisa.it

Supervisor: Giuseppe Persiano

29 April 2011

Predicate Encryption Schemes

- Predicate encryption (PE) schemes [Boneh-Waters07] are encryption schemes in which each ciphertext Ct is associated with a attribute vector $\vec{x} = (x_1, \ldots, x_n)$ and keys K are associated with predicates.
- A key K can decrypt a ciphertext Ct iff the attribute vector of the ciphertext satisfies the predicate of the key.
- PE $\rightarrow$ fine-grained access control on encrypted data.

Examples (Antispam filter)

- Your antispam filter should discard emails containing some prohibited words.
- With classical PKE you give the antispam the secret key.
- It learns all the content of the email.
- With predicate encryption you give the antispam a special key relative to the words.
- It only learns whether the words are in the email.

Examples (Credit card transactions)

- A gateway (G) observes a stream of encrypted transactions.
- It must flag transactions whose values is $>$ \$1000.
- With PKE, Visa must give G the Sk.
- With Predicate Encryption, Visa can give G a special key T.
- By using T, G only learns whether the transaction is for a value $>$ \$1000.

Definition of Predicate Encryption Schemes

- A predicate encryption scheme for a class $\mathcal{F}$ of predicates (boolean functions) over attributes in Σ is quadruple of probabilistic polynomial-time algorithms $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ such that:
- Setup takes as input the security parameter 1^k and outputs the *master public key* Pk and the *master secret key* Msk.
- KeyGen takes as input the master secret key Msk and a predicate $f \in \mathcal{F}$ and outputs the decryption key K_f associated with f.
- Enc takes as input the public key Pk and an *attribute string* $\vec{x} \in \Sigma$ and a message M in some associated message space and returns ciphertext $\mathsf{Ct}_{\vec{x}}$.
- Dec takes as input a secret key K_f and a ciphertext $\mathsf{Ct}_{\vec{x}}$ and outputs a message M.

Correctness of Predicate Based Encryption Schemes

We require that for all attributes $\vec{x} \in \Sigma$ and predicates $f \in \mathcal{F}$ such that $f(\vec{x}) = 1$, it holds that:

$$\mathrm{Prob}[(\mathsf{Pk}, \mathsf{Msk}) \leftarrow \mathsf{Setup}(1^k); K_f \leftarrow \mathsf{KeyGen}(\mathsf{Msk}, f);$$

$$\mathsf{Ct}_{\vec{x}} \leftarrow \mathsf{Enc}(\mathsf{Pk}, \vec{x}, M) : \mathsf{Dec}(K_f, \mathsf{Ct}_{\vec{x}}) = M] \geq 1 - neg(k).$$

Viceversa if $f(\vec{x}) = 0$, then the previous probability should be negligible.

Predicate-only schemes

- Encrypt only with respect to the attribute string.
- There is no message M to encrypt (alternatively you can set it to 1).
- Dec procedure is substituted with a Test procedure which returns 0 or 1 indicating whether the predicate is satisfied.
- Useful for encrypted databases and many other applications.

Security of Predicate Based Encryption Schemes

- A PE scheme $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ has ξ-ecurity, where $\xi \subset \{0, 1\}$, if all PPT adversaries $\mathcal{A}$ have negligible advantage in the following experiment.
- Setup. The public and the secret key $(\mathsf{Msk}, \mathsf{Pk})$ are generated using the Setup procedure and $\mathcal{A}$ receives Pk.
- Query Phase I. $\mathcal{A}$ requests and gets private keys K_f relative to predicates f. Key $K_{\vec{y}}$ is computed using the KeyGen procedure.
- Challenge. $\mathcal{A}$ returns two different pairs attribute/message (x_0, M_0) and (x_1, M_1) of the same length, subject to the constraint that $f(\vec{x}_0) = f(\vec{x}_1) \in \xi$ for any f queried to the key oracle in both query phases. η is chosen at random from $\{0, 1\}$. $\mathcal{A}$ is given ciphertext $\mathsf{Ct}_{\vec{x}} \leftarrow \mathsf{Enc}(\mathsf{Pk}, \vec{x}_\eta, M_\eta)$.
- Query Phase II. Identical to Query Phase I.
- Output. $\mathcal{A}$ returns η'. If $\eta = \eta'$ then return 1 else return 0.

Notions of Security

- Selective security: the adversary chooses the challenge attributes before seeing the public-key.
- Why? The model is weaker (see separation in the thesis) but it is easier to prove the security
- The simulator can build the public-key basing it on the challenges so that it can answer all the queries easily.
- In the case that $\xi = \{0\}$ we talk about security against *restricted adversaries.*
- If $\xi = \{0, 1\}$ we have the best security we can guarantee. In this case we talk about security against *unrestricted adversaries.*
- Recently, Boneh,Sahai and Waters showed impossibility result for simulation-based security.
- Main Result of This Thesis: First PE system for HVE (to define..) secured against *unrestricted adversaries.*

Trivial construction for every predicate

- Let $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{Dec}')$ be a PK system. Let $\mathcal{F} = (P_1, \ldots, P_t)$. We build a PE $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ as follows.
- $\mathsf{Setup}(1^k)$: runs t-times $\mathsf{Setup}'(1^k)$ to obtain $\mathsf{Pk} = (\mathsf{Pk}_1, \ldots, \mathsf{Pk}_t)$ and $\mathsf{Msk} = (\mathsf{Sk}_1, \ldots, \mathsf{Sk}_t)$.
- $\mathsf{KeyGen}(\mathsf{Msk}, f)$: (here f is a index j of a predicate in the list $(P_1, \ldots, P_t)$ outputs $K_f = (j, \mathsf{Sk}_j)$.
- $\mathsf{Enc}(\mathsf{Pk}, M, \vec{x})$: define $C_j = \mathsf{Enc}'(\mathsf{Pk}_j, M)$ if $P_j(\vec{x}) = 1$ or $C_j = \mathsf{Enc}'(\mathsf{Pk}_j, \perp)$ otherwise. Outputs $\mathsf{Ct}_{\vec{x}} = (C_1, \ldots, C_t)$.
- $\mathsf{Dec}(K_f, \mathsf{Ct}_{\vec{x}})$: Let K_f be (j, Sk_j) and $\mathsf{Ct}_x = (C_1, \ldots, C_t)$. Outputs $\mathsf{Dec}(\mathsf{Sk}_j, C_j)$.

Trivial construction for every predicate - continued

- The construction is higly inefficient (super-exponential time and space).
- We do not know whether it is possible to construct PE for any poly-time predicates.
- Despite of this, we have efficient constructions for some interesting predicates with many applications.

Definition of Hidden Vector Encryption Schemes

- Defined by [Boneh-Waters07].
- HVE schemes are Predicate Encryption schemes for Match.
- Let $\vec{x}$ be a string over Σ and $\vec{y}$ be a string over $\Sigma \cup \{\star\}$; $\vec{x}$ and $\vec{y}$ of the same length n.
- Define predicate Match$(\vec{x}, \vec{y})$ to be true iff for each $1 \leq i \leq n$ we have $x_i = y_i$ or $y_i = \star$. Intuitively, $\star$ is the "don't care" symbol.
- Example: Match is true with 001 and 00$\star$ but not with 101 and $\star$11.

Applications of HVE (PEKS/SE, AIBE,Conjunctive queries on encrypted DB)

- Easy to see that HVE implies Searchable Encryption and Anonymous IBE.
- Analogously, you can see SE as predicate-only PE scheme for the equality predicate.
- Applications above do not use the $\star$ capabilites.
- Exploiting the $\star$'s, I could search in the encrypted DB of UNISA if there are other people with my name. Namely, search all tuples with 'Name=Vincenzo AND Campus=UNISA'.
- The last is not possible with PEKS/SE.
- Other applications: conjunctive comparison queries and subset queries.

Reduce k-CNF and k-DNF to HVE

Idea

Enumerates all the k-CNF clauses over n variables. They are $\Theta(n^k)$.

k-DNF

For k-DNF complement the result (valid for predicate-only schemes).

A more general predicate

- In Eurocrypt08, Katz-Sahai-Waters presented a scheme for a more general class of predicates.
- Keys and ciphertexts are relative to attribute vectors $\vec{x} \in \mathbb{Z}_N^w$.
- By using a key relative to $\vec{y}$ you can decrypt a ciphertex relative to $\vec{x}$ iff $\langle \vec{x}, \vec{y} \rangle = 0 \mod N$.
- Easy to see that inner-product $\rightarrow$ HVE.

Known constructions for HVE

- First construction by Boneh-Waters07.
- It used bilinear group of composite order and thus assumed factoring.
- Iovino-Persiano08 show a more efficient construction based on groups of prime order.
- The latter construction is very simple and the security proof is based on Decision Linear.
- New schemes followed which add delegating capabilities, key privacy, short keys...
- This thesis: fully secure restricted and unrestricted HVE.

Groups endowed with bilinear maps

- We have multiplicative groups $\mathbb{G}$ and $\mathbb{G}_T$ of prime order p and a non-degenerate bilinear pairing function $\mathbf{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$.
- The pairing function has the property that, for all $g \in \mathbb{G}, g \neq 1$, we have $\mathbf{e}(g, g) \neq 1$ and $\mathbf{e}(g^a, g^b) = \mathbf{e}(g, g)^{ab}$.
- We denote by g and $\mathbf{e}(g, g)$ the generators of $\mathbb{G}$ and $\mathbb{G}_T$.
- We call a *symmetric bilinear* instance a tuple $\mathcal{I} = [p, \mathbb{G}, \mathbb{G}_T, g, \mathbf{e}]$ and assume that there exists an efficient generation procedure that, on input security parameter 1^k, outputs an instance with $|p| = \Theta(k)$.

Bilinear groups of composite-order

- We have multiplicative *cyclic groups* $\mathbb{G}$ *and* $\mathbb{G}_T$ *of* composite-order N product of more primes and a non-degenerate bilinear pairing function $\mathbf{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$.
- Since that the groups are cyclic, it follows that $\mathbf{e}(g, h) = 1$ when g and h belong to different subgroups of $\mathbb{G}$.
- This property is called orthogonality and is used in our fully secure constructions.
- (Maybe) we can convert schemes based on composite-order groups to schemes based on prime-order groups (see Freeman10).

Computational Assumptions

- In bilinear groups, standard assumptions like Decisional Diffie-Hellman are *false*
- No problem. We can formulate new assumptions believed to be true in this setting.
- Example 1: Decision BDH. Given a tuple $[g, g^{z_1}, g^{z_2}, g^{z_3}, Z]$ for random exponents $z_1, z_2, z_3 \in \mathbb{Z}_p$ it is hard to distinguish $Z = \mathbf{e}(g, g)^{z_1 z_2 z_3}$ from a random $Z \in \mathbb{G}_T$.
- Decision Linear. Given a tuple $[g, g^{z_1}, g^{z_2}, g^{z_1 z_3}, g^{z_1 z_4}, Z]$ for random exponents $z_1, z_2, z_3, z_4 \in \mathbb{Z}_p$ it is hard to distinguish $Z = g^{z_3 + z_4}$ from a random $Z \in \mathbb{G}$.
- In bilinear groups of composite-order we can formulate assumptions that essentially state the difficulty of distinguishing whether an element does or does not contain a given subgroup.

The selectively secure construction - First attempt

- We associate to each position of the attribute string $\vec{x} = x_1, \ldots, x_n$ a value t_i if $x_i = 1$ or r_i otherwise. These numbers are chosen at random in $\mathbb{Z}_p$ along with z and the public-key is g^{t_i}, g^{r_i} for each $i = 1, \ldots, n$.
- To generate private keys for a string $\vec{y}$, share z in a_i's such that the sum of a_i's is z. In the positions where $y_i = 1$ put g^{a_i/t_i} and where $y_i = 0$ put g^{a_i/r_i}.
- When encrypting the pair $(M, \vec{x})$, hide the message M with $M \cdot \mathbf{e}(g, g)^{zs}$ for a random s; also in the positions where $x_i = 1$ put $g^{t_i s}$ or $g^{r_i s}$ otherwise.
- To decrypt, we pair (for example) g^{a_i/t_i} with g^{st_i} to obtain $\mathbf{e}(g, g)^{a_i s}$. Multiply each such element to obtain $\mathbf{e}(g, g)^{zs}$ used to recover M.
- Intuition: to obtain z, you must get all a_i's, and that's possible only if you own the key for a string that matches with the ciphertext attribute.

Attribute Hiding

- The above scheme guarantees the security of the message M (under DBDH) but not of $\vec{x}$.
- In fact, in a such scheme we should include g^{t_i}, g^{r_i}'s as public parameters. This would break the security of previous scheme
- Indeed an adversary could test whether first two bits of the string associated to a ciphertext are 01 using this check:
- $\mathbf{e}(g^{t_1 s}, g^{r_2}) = \mathbf{e}(g^{t_1}, g^{r_2 s})$
- The previous scheme is unsecure!
- But there is a solution...

The splitting technique

- We solve the problem using a linear splitting technique.
- For each position we choose s_i at random and split $g^{t_i s}$ in $g^{t_i(s-s_i)}$ and $g^{v_i s_i}$, analogously split $g^{r_i s}$ in $g^{r_i(s-s_i)}$ and $g^{m_i s_i}$ (now include in Pk also g^{r_i} and g^{m_i}).
- Similarly for the private keys, change g^{a_i/t_i} with the pair $g^{a_i/t_i}, g^{a_i/v_i}$ and g^{a_i/r_i} with the pair $g^{a_i/v_i}, g^{a_i/r_i}$.
- The decryption works and the new scheme is secure!
- The splitting technique needs the Decision Linear assumption.

Dual System Encryption

- Fully-secure constructions for IBE or more general primitives required the random oracle model (Boneh-Franklin's IBE), ad-hoc solutions (efficient IBE of Waters in the standard model) or non-standard assumptions (Gentry's IBE).
- Waters09 presented a powerful tool to prove the full security of IBE-like primitives: the Dual System Encryption methodology.
- In DSE keys and ciphertexts can assume two forms: normal and semi-functional.
- Normal key (ciphertext) can be combined with a semi-functional ciphertext (key).
- Semi-functional ciphertexts can NOT decrypted by semi-functional keys!

Dual System Encryption - continued

- The security proof proceeds in the following steps.
- The challenge ciphertext is changed to semi-functional form: adversary can not detect it!.
- The keys are changed one by one to semi-functional.
- Idea: normal keys can not decrypt and if you change them to semi-functional form they continue to not decrypt: so adversary does not detect the change.
- By performing the change one key at a time we can exploit locality and the indistinguishability follows by simple assumptions.
- More paradoxes!

The Paradoxes of Dual System Encryption

- The simulator could use the assumption to create a ciphertext (for the same id) that is semi-functional and test if the key is normal or semi-functional.
- Waters09 avoids the paradox by using tags: it attaches a tag (that is function of the id) to each semi-functional ciphertext and to a key of both types and decryption works only if the tags are different.
- LewkoWaters10 avoids the paradox by using the concept of nominally semi-functional algorithms: a nominally semi-functional ciphertext and a nominally semi-functional key can be combined for decryption.

Our Fully-secure HVE Constructions

- Setup$(1^\lambda, 1^\ell)$: bilinear instance of groups of composite order $N = p_1 p_2 p_3 p_4$. Choose $(t_{i,b} \in_R \mathbb{Z}_N)_{i \in [\ell], b \in \{0,1\}}$.

$$\mathsf{Pk} = [N, g_3, (T_{i,b} = g_1^{t_{i,b}} \cdot R_{3,i,b})_{i \in [\ell], b \in \{0,1\}}]$$

$$\mathsf{Msk} = [g_{12} = g_1 \cdot g_2, g_4, (t_{i,b})_{i \in [\ell], b \in \{0,1\}}]$$

- KeyGen$(\mathsf{Msk}, \vec{y})$: Let $S_{\vec{y}} = \{i \in [\ell] | y_i \neq \star\}$. Choose $a_i \in_R \mathbb{Z}_N$ such that $\sum_{i \in S_{\vec{y}}} a_i = 0$.

$$Y_i = {g_{12}}^{a_i / t_{i,y_i}} W_{4,i}$$

- Enc$(\mathsf{Pk}, \vec{x})$: Choose $s \in_R \mathbb{Z}_N$.

$$X_i = {T_{i,x_i}}^s Z_{3,i}$$

- Test$(\mathsf{Ct}, \mathsf{Sk}_{\vec{y}})$: returns TRUE iff $T = 1$.

$$T = \prod_{i \in S_{\vec{y}}} \mathbf{e}(X_i, Y_i) = \prod_{i \in S_{\vec{y}}} \mathbf{e}(g_1^{s \cdot t_{i,x_i}}, {g_1}^{a_i / t_{i,y_i}}) = \prod_{i \in S_{\vec{y}}} \mathbf{e}(g_1, g_1)^{\frac{s \cdot t_{i,x_i} \cdot a_i}{t_{i,y_i}}}$$

Our proof strategy

- We project the PK in the $\mathbb{G}_{p_2}$ subgroup: the adversary does not detect the change because the keys share a $\mathbb{G}_{p_2}$ part (but not the challenge ciphertext).
- The simulator will know the trapdoors to create the $\mathbb{G}_{p_2}$ part of PK and keys.
- We change the $\mathbb{G}_{p_1}$ part of the keys one by one.
- In each key game we change the $\mathbb{G}_{p_1}$ part of the keys to random.
- We make this by guessing where the challenge key differs from the challenge ciphertext.

Our proof strategy - continued

- We solve the paradox of DSE by using an all-but-one simulation.
- The assumption allows us to simulate a key that differs from the challenge ciphertext in the guessed position but not keys that match it.
- In the last key game the $\mathbb{G}_{p_1}$ part of the key is random and the challenge ciphertext does not contain the $\mathbb{G}_{p_2}$ part. Recalling that the PK lives on $\mathbb{G}_{p_2}$ we conclude that the challenge attribute is information-theoretically hidden from the adversary.

A Paradox Left Unsolved

- The dual system encryption was formulated for (H)IBE where restricted and unrestricted security coincides.
- In PE, the adversary can ask queries for predicates that match both the challenges.
- In this case, a naive use of DSE induces a new paradox: a matching query would allow to distinguish if the key is semi-functional or normal.
- If it is semi-functional, the decryption with the semi-functional challenge ciphertext won't work but if it is normal it will do!

Our Solution: The Main Result of The Thesis

- We use $q \cdot \ell$ games instead of q games.
- We view the proof as a Down-Right-Up trip on the queries.
- Down Phase. For the first (in general i-th, for $i = 1$ to ℓ) position of the challenge ciphertext we change the distribution of the keys.
- Right Phase. We change the value of the first (in general i-th) position of the challenge ciphertex if it corresponds to a position where the two challenge attributes differ.
- The value is changed by setting it to random.

Our Solution: The Main Result of The Thesis - continued

- Up Phase. We come back to the situation where the keys were all well-formed but the challenge ciphertext remains changed.
- Right Phase. We iterate the process incrementing i and stepping to the Down Phase.
- Idea: In the Down Phase, when we receive a query for a vector $\vec{y}$ such that it has $\star$ in position i, we can simulate it correctly!
- It could be matching or non-matching query but we are sure that ALL matching keys have $\star$ in position i!
- During the Right Phase we observe the following situation: the matching keys have $\star$ in position i and the remaining keys (that can be either matching or non-matching) have a random $\mathbb{G}_{p_1}$ part.

Our Solution: The Main Result of The Thesis - Conclusion

- Therefore the *i*-th position of the challenge ciphertext is information-theoretically hidden from the adversary!
- In the last game the challenge ciphertext is independent from the challenge attributes: it is random where they differ and equal elsewhere.
- Some troubles: we can perform the simulation only for the positions where the challenge attributes differ.
- We use an abort technique in the bad case. Our analysis shows that the adversary cannot exploit this abort for its advantage.
- We loose a factor $\ell \cdot q$ in the reduction but we proved security against unrestricted adversaries!

Other results of this thesis

- Hierarchical IBE and PE: given a key for predicate P, derive a key for more specialized predicate (i.e., a predicate that satisfies less attributes).
- For HVE: given a key for $1 * 0$, you could derive a key for 100 or 110.
- For example, the University owns a key that decrypt everything and gives to the department of CS a key to decrypt only the ciphertexts that begin with 'CS Dept'.
- Previous Hierarchical HVE system of Shi-Waters08: super-linear computational complexity and selective security. Ours is linear and fully secure.
- First Fully secure Anonymous (H)IBE, Secret-key IBE/HVE, Partial Public-Key model.

Future directions and open problems

- Big open problem: PE for arbitrary poly-size circuits.
- Limits of bilinear maps: which classes of PE systems can we build from bilinear maps?
- PE schemes from other assumptions (lattices, QR, code-based...).
- Tight security proofs: if the adversary breaks the system in time t with probability p, build an adversary that breaks some simple assumption in approximatively the same time and probability.
- Efficiency: short ciphertexts and keys, constant-size PK, etc.

THE END

Questions Left Unanswered?