Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products

Jonathan Katz[1⋆], Amit Sahai[2⋆⋆], and Brent Waters[3⋆⋆⋆]

[1] University of Maryland, jkatz@cs.umd.edu
[2] UCLA, sahai@cs.ucla.edu
[3] SRI International, bwaters@csl.sri.com

Abstract. Predicate encryption is a new paradigm generalizing, among other things, identity-based encryption. In a predicate encryption scheme, secret keys correspond to predicates and ciphertexts are associated with attributes; the secret key SK_f corresponding to a predicate f can be used to decrypt a ciphertext associated with attribute I if and only if $f(I) = 1$. Constructions of such schemes are currently known for relatively few classes of predicates.
We construct such a scheme for predicates corresponding to the evaluation of *inner products* over $\mathbb{Z}_N$ (for some large integer N). This, in turn, enables constructions in which predicates correspond to the evaluation of disjunctions, polynomials, CNF/DNF formulae, or threshold predicates (among others). Besides serving as a significant step forward in the theory of predicate encryption, our results lead to a number of applications that are interesting in their own right.

1 Introduction

Traditional public-key encryption is rather coarse-grained: a sender encrypts a message M with respect to a given public key PK, and only the owner of the (unique) secret key associated with PK can decrypt the resulting ciphertext and recover the message. These straightforward semantics suffice for point-to-point communication, where encrypted data is intended for one particular user who is known to the sender in advance. In other settings, however, the sender may instead want to define some complex *policy* determining who is allowed to recover the encrypted data. For example, classified data might be associated with certain keywords; this data should be accessible to users who are allowed to read *all*

⋆ Research supported in part by NSF CAREER award #0447075 and the U.S. Army Research Laboratory.
⋆⋆ Supported in part by the NSF ITR and CyberTrust programs (including grants 0627781, 0456717, 0716389, and 0205594), a subgrant from SRI as part of the Army Cyber-TA program, an equipment grant from Intel, an Okawa Research Award, and an Alfred P. Sloan Foundation Research Fellowship.
⋆⋆⋆ Supported by NSF CNS-0524252, CNS-0716199; the US Army Research Office under the CyberTA Grant No. W911NF-06-1-0316; and the U.S. Department of Homeland Security under Grant Award Number 2006-CS-001-000001.

classified information, as well as to users allowed to read information associated with the particular keywords in question. Or, in a health care application, a patient's records should perhaps be accessible only to a physician who has treated the patient in the past.

Applications such as those sketched above require new cryptographic mechanisms that provide more fine-grained control over access to encrypted data. *Predicate encryption* offers one such tool. At a high level (formal definitions are given in Section 2), secret keys in a predicate encryption scheme correspond to predicates in some class $\mathcal{F}$, and a sender associates a ciphertext with an attribute in a set Σ; a ciphertext associated with the attribute $I \in \Sigma$ can be decrypted by a secret key SK_f corresponding to the predicate $f \in \mathcal{F}$ if and only if $f(I) = 1$.

The "basic" level of security achieved by such schemes guarantees, informally, that a ciphertext associated with attribute I hides all information about the underlying message unless one holds a secret key giving the explicit ability to decrypt. I.e., if an adversary $\mathcal{A}$ holds keys $SK_{f_1}, \ldots, SK_{f_\ell}$, then $\mathcal{A}$ learns nothing about a message encrypted using attribute I if $f_1(I) = \cdots = f_\ell(I) = 0$. We refer to this security notion as *payload hiding*. A stronger notion of security, that we call *attribute hiding*, requires further that a ciphertext hides all information about the associated attribute I except that which is explicitly leaked by the keys in one's possession; i.e., an adversary holding secret keys as above learns only the values $f_1(I), \ldots, f_\ell(I)$ (in addition to the message in case one of these evaluates to '1'). Formal definitions are given in Section 2.

Much recent work aimed at constructing different types of fine-grained encryption schemes can be cast in the framework of predicate encryption. Identity-based encryption (IBE) [21, 8, 13, 4, 5, 23] can be viewed as predicate encryption for the class of equality tests; the standard notion of security for IBE [8, 12] corresponds to payload-hiding, while *anonymous* IBE [7, 11, 14] corresponds to the stronger notion of attribute hiding. Attribute-based encryption schemes [20, 15, 3, 19] can also be cast in the framework of predicate encryption, though in this case all the listed constructions achieve payload hiding only. Boneh and Waters [10] construct a predicate encryption scheme that handles conjunctions (of, e.g., equality tests) and range queries; their scheme satisfies the stronger notion of attribute hiding. Shi et al. [22] also construct a scheme for range queries, but prove security in a weaker model where attribute hiding is required to hold only if the adversary holds keys that do *not* allow recovery of the message.

Other work introducing concepts related to predicate encryption includes [2, 1]. In contrast to the present work, however, the threat model in those works do not consider *collusion* among users holding different secret keys. The problem becomes significantly more difficult when security against collusion is required.

1.1 Our Results

An important research direction is to construct predicate encryption schemes for predicate classes $\mathcal{F}$ that are as expressive as possible, with the ultimate goal being to handle all polynomial-time predicates. We are still far from this goal. Furthermore, most of the existing work (listed above) yields only payload-hiding schemes, and existing techniques for obtaining attribute-hiding schemes seem limited to enforcing *conjunctions* (indeed, handling disjunctions was left as an open question in [10]). Getting slightly technical, this is because the underlying cryptographic mechanism used in the above schemes is to pair components of the secret key with corresponding components of the ciphertext and then multiply the intermediate results together; a "cancelation" occurs if everything "matches", but a random group element results if there is any "mismatch". Thus, the holder of a non-matching secret key learns only that there was a mismatch in *at least one* position, but does not learn the number of mismatches or their locations. Very different cryptographic techniques seem needed to support disjunctions, since a mismatch in a single position cannot result in a completely random group element but rather must somehow allow for a "cancelation" if there is a match in some other position. (We stress that what makes this difficult is that attribute hiding requires correct decryption to hide the position of a match and only reveal that there was a match in at least one position.)

The aim of our work is to construct an attribute-hiding scheme handling disjunctions. As a stepping stone toward this goal, we first focus on predicates corresponding to the computation of inner products over $\mathbb{Z}_N$ (for some large integer N). Formally, we take $\Sigma = \mathbb{Z}_N^n$ as our set of attributes, and take our class of predicates to be $\mathcal{F} = \{f_{\boldsymbol{x}} \mid \boldsymbol{x} \in \mathbb{Z}_N^n\}$ where $f_{\boldsymbol{x}}(\boldsymbol{y}) = 1$ iff $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = 0$. (Here, $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ denotes the dot product $\sum_{i=1}^n x_i \cdot y_i \bmod N$ of two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$.) We construct a predicate encryption scheme for this $\mathcal{F}$ without random oracles, based on two new assumptions in composite-order groups equipped with a bilinear map. Our assumptions are non-interactive and of fixed size (i.e., not "q-type"), and can be shown to hold in the generic group model. A pessimistic interpretation of our results would be that we prove security in the generic group model, but we believe it is of importance that we are able to distill our necessary assumptions to ones that are compact and falsifiable. Our construction uses new techniques, including the fact that we work in a bilinear group whose order is a product of *three* primes.

We view our main construction as a significant step toward increasing the expressiveness of predicate encryption. Moreover, we show that any predicate encryption scheme supporting "inner product" predicates as described above can be used as a building block to construct predicates of more general types:

- As an easy warm-up, we show that it implies (anonymous) identity-based encryption as well as hidden-vector encryption [10]. As a consequence, our work implies all the results of [10].
- We can also construct predicate encryption schemes supporting polynomial evaluation. Here, we take $\mathbb{Z}_N$ as our set of attributes, and predicates correspond to polynomials in $\mathbb{Z}_N[x]$ of some bounded degree; a predicate evaluates to 1 iff the corresponding polynomial evaluates to 0 on the point in question. We can also extend this to include multi-variate polynomials (in some bounded number of variables). A "dual" of this construction allows the attributes to be polynomials, and the predicates to correspond to evaluation at a fixed point.

– Given the above, we can fairly easily support predicates that are *disjunctions* of other predicates (e.g., equality), thus achieving our main goal. In the context of identity-based encryption, this gives the ability to issue secret keys corresponding to a *set* of identities that enables decryption whenever a ciphertext is encrypted to any identity in this set (without leaking which identity was actually used to encrypt).
– We also show how to handle predicates corresponding to DNF and CNF formulae of some bounded size.
– Working directly with our "inner product" construction, we can derive a scheme supporting threshold queries of the following form: Attributes are subsets of $A = \{1, \ldots, \ell\}$, and predicates take the form $\{f_{S,t} \mid S \subseteq A\}$ where $f_{S,t}(S') = 1$ iff $S \cap S' = t$. This is useful in the "fuzzy IBE" setting of Sahai and Waters [20], and improves on their work in that we achieve attribute hiding (rather than only payload hiding) and handle *exact* thresholds.

We defer further discussion regarding the above until Section 5.

2 Definitions

We formally define the syntax of predicate encryption and the security properties discussed informally in the Introduction. (Our definitions follow the general framework of those given in [10].) Throughout this section, we consider the general case where Σ denotes an arbitrary set of attributes and $\mathcal{F}$ denotes an arbitrary set of predicates over Σ. Formally, both Σ and $\mathcal{F}$ may depend on the security parameter and/or the master public parameters; for simplicity, we leave this implicit.

Definition 1. *A* predicate encryption scheme *for the class of predicates $\mathcal{F}$ over the set of attributes Σ consists of four* PPT *algorithms* Setup, GenKey, Enc, Dec *such that:*

- Setup *takes as input the security parameter 1^n and outputs a (master) public key PK and a (master) secret key SK.*
- GenKey *takes as input the master secret key SK and a (description of a) predicate $f \in \mathcal{F}$. It outputs a key SK_f.*
- Enc *takes as input the public key PK, an attribute $I \in \Sigma$, and a message M in some associated message space. It returns a ciphertext C. We write this as $C \leftarrow \mathsf{Enc}_{PK}(I, M)$.*
- Dec *takes as input a secret key SK_f and a ciphertext C. It outputs either a message M or the distinguished symbol $\perp$.*

For correctness, we require that for all n, all (PK, SK) generated by $\mathsf{Setup}(1^n)$*, all $f \in \mathcal{F}$, any key $SK_f \leftarrow \mathsf{GenKey}_{SK}(f)$, and all $I \in \Sigma$:*

- *If $f(I) = 1$ then $\mathsf{Dec}_{SK_f}(\mathsf{Enc}_{PK}(I, M)) = M$.*
- *If $f(I) = 0$ then $\mathsf{Dec}_{SK_f}(\mathsf{Enc}_{PK}(I, M)) = \perp$ with all but negligible probability.*

We will also consider a variant of the above that we call a *predicate-only scheme*. Here, Enc takes only an attribute I (and no message); the correctness requirement is that $\mathsf{Dec}_{SK_f}(\mathsf{Enc}_{PK}(I)) = f(I)$ and so all the receiver learns is whether the predicate is satisfied. A predicate-only scheme can serve as a useful building block toward a full-fledged predicate encryption scheme.

Our definition of attribute-hiding security corresponds to the notion described informally earlier. Here, an adversary may request keys corresponding to the predicates $f_1, \ldots, f_\ell$ and is then given either $\mathsf{Enc}_{PK}(I_0, M_0)$ or $\mathsf{Enc}_{PK}(I_1, M_1)$ for attributes I_0, I_1 such that $f_i(I_0) = f_i(I_1)$ for all i. Furthermore, if $M_0 \neq M_1$ then it is required that $f_i(I_0) = f_i(I_1) = 0$ for all i. The goal of the adversary is to determine which attribute-message pair was encrypted, and the stated conditions ensure that this is not trivial. Observe that when specialized to the case when $\mathcal{F}$ consists of equality tests on strings, this notion corresponds to *anonymous* identity-based encryption (with selective-ID security).

Definition 2. *A predicate encryption scheme with respect to $\mathcal{F}$ and Σ is* attribute hiding *(or simple* secure*) if for all* PPT *adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in the following experiment is negligible in the security parameter n:*

1. *$\mathcal{A}(1^n)$ outputs $I_0, I_1 \in \Sigma$.*
2. $\mathsf{Setup}(1^n)$ *is run to generate PK, SK, and the adversary is given PK.*
3. *$\mathcal{A}$ may adaptively request keys for any predicates $f_1, \ldots, f_\ell \in \mathcal{F}$ subject to the restriction that $f_i(I_0) = f_i(I_1)$ for all i. In response, $\mathcal{A}$ is given the corresponding keys $SK_{f_i} \leftarrow \mathsf{GenKey}_{SK}(f_i)$.*
4. *$\mathcal{A}$ outputs two equal-length messages M_0, M_1. If there is an i for which $f_i(I_0) = f_i(I_1) = 1$, then it is required that $M_0 = M_1$.*
5. *A random bit b is chosen, and $\mathcal{A}$ is given the ciphertext $C \leftarrow \mathsf{Enc}_{PK}(I_b, M_b)$.*
6. *The adversary may continue to request keys for additional predicates, subject to the same restrictions as before.*
7. *$\mathcal{A}$ outputs a bit b', and succeeds if $b' = b$.*

The advantage of $\mathcal{A}$ is the absolute value of the difference between its success probability and $1/2$.

We remark that our definition uses the "selective" notion of security introduced in [12]. One could also consider the definition where the adversary is allowed to specify I_0, I_1 adaptively, based on PK and any secret keys it obtains.

Payload hiding, a strictly weaker notion of security, is defined by forcing $I_0 = I_1 = I$ in the above (in which case $\mathcal{A}$ has no possible advantage if $f_i(I) = 1$ for some i). For predicate-only encryption schemes we simply omit the messages in the above experiment. For convenience, we include in Appendix A a re-statement of the definition of security given above for the particular inner-product predicate we use in our main construction.

3 Background on Pairings and Complexity Assumptions

We review some general notions about bilinear groups of *composite order*, first used in cryptographic applications by [9]. In contrast to all prior work using composite-order bilinear groups, however, we use groups whose order N is a product of *three* (distinct) primes. This is for simplicity only, since a variant of our construction can be proven secure based on a "decisional linear"-type assumption [6] in a group of composite order N which is a product of two primes.[4]

Let $\mathcal{G}$ be an algorithm that takes as input a security parameter 1^n and outputs a tuple $(p, q, r, \mathbb{G}, \mathbb{G}_T, \hat{e})$ where p, q, r are distinct primes, $\mathbb{G}$ and $\mathbb{G}_T$ are two cyclic groups of order $N = pqr$, and $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is *bilinear* (i.e., $\forall u, v \in \mathbb{G}$ and $\forall a, b \in \mathbb{Z}$ we have $\hat{e}(u^a, v^b) = \hat{e}(u, v)^{ab}$) and *non-degenerate* (i.e., if g generates $\mathbb{G}$ then $\hat{e}(g, g)$ generates $\mathbb{G}_T$). We assume the group operations in $\mathbb{G}$ and $\mathbb{G}_T$ as well as the bilinear map $\hat{e}$ are all computable in time polynomial in n. Furthermore, we assume that the description of $\mathbb{G}$ and $\mathbb{G}_T$ includes generators of these groups.

We use the notation $\mathbb{G}_p, \mathbb{G}_q, \mathbb{G}_r$ to denote the subgroups of $\mathbb{G}$ having order p, q, and r, respectively. Observe that $\mathbb{G} = \mathbb{G}_p \times \mathbb{G}_q \times \mathbb{G}_r$. Note also that if g is a generator of $\mathbb{G}$, then the element g^{pq} is a generator of $\mathbb{G}_r$; the element g^{pr} is a generator of $\mathbb{G}_q$; and the element g^{qr} is a generator of $\mathbb{G}_p$. Furthermore, if, e.g., $h_p \in \mathbb{G}_p$ and $h_q \in \mathbb{G}_q$ then

$$\hat{e}(h_p, h_q) = \hat{e}\left((g^{qr})^{\alpha_1}, (g^{pr})^{\alpha_2}\right) = \hat{e}\left(g^{\alpha_1}, g^{r\alpha_2}\right)^{pqr} = 1,$$

where $\alpha_1 = \log_{g^{qr}} h_p$ and $\alpha_2 = \log_{g^{pr}} h_q$. Similar rules hold whenever $\hat{e}$ is applied to non-identity elements in distinct subgroups.

3.1 Our Assumptions

We now state the assumptions we use to prove security of our construction. As remarked earlier, these assumptions are new but we justify them by proving that they hold in the generic group model under the assumption that finding a non-trivial factor of N (the group order) is hard. (Details appear in the full version of this paper [17].) At a minimum, then, our construction can be viewed as secure in the generic group model. Nevertheless, we state our assumptions explicitly and highlight that they are non-interactive and of fixed size.

Assumption 1. Let $\mathcal{G}$ be as above. We say that $\mathcal{G}$ satisfies Assumption 1 if the advantage of any PPT algorithm $\mathcal{A}$ in the following experiment is negligible in the security parameter n:

1. $\mathcal{G}(1^n)$ is run to obtain $(p, q, r, \mathbb{G}, \mathbb{G}_T, \hat{e})$. Set $N = pqr$, and let g_p, g_q, g_r be generators of $\mathbb{G}_p$, $\mathbb{G}_q$, and $\mathbb{G}_r$, respectively.
2. Choose random $Q_1, Q_2, Q_3 \in \mathbb{G}_q$, random $R_1, R_2, R_3 \in \mathbb{G}_r$, random $a, b, s \in \mathbb{Z}_p$, and a random bit ν. Give to $\mathcal{A}$ the values $(N, \mathbb{G}, \mathbb{G}_T, \hat{e})$ as well as

$$g_p,\ \ g_r,\ \ g_q R_1,\ \ g_p^b,\ \ g_p^{b^2},\ \ g_p^a g_q,\ \ g_p^{ab} Q_1,\ \ g_p^s,\ \ g_p^{bs} Q_2 R_2.$$

[4] This is analogous to the "folklore" transformation (see, e.g., [16]) that converts a scheme based on a group whose order N is a product of two primes to a scheme that uses a prime-order group. Typically, using composite order groups gives a simpler scheme; since the group sizes are larger, however, group operations are less efficient.

If $\nu = 0$ give $\mathcal{A}$ the value $T = g_p^{b^2 s} R_3$, while if $\nu = 1$ give $\mathcal{A}$ the value $T = g_p^{b^2 s} Q_3 R_3$.
3. $\mathcal{A}$ outputs a bit ν', and succeeds if $\nu' = \nu$.

The advantage of $\mathcal{A}$ is the absolute value of the difference between its success probability and 1/2.

Assumption 2. Let $\mathcal{G}$ be as above. We say that $\mathcal{G}$ satisfies Assumption 2 if the advantage of any PPT algorithm $\mathcal{A}$ in the following experiment is negligible in the security parameter n:

1. $\mathcal{G}(1^n)$ is run to obtain $(p, q, r, \mathbb{G}, \mathbb{G}_T, \hat{e})$. Set $N = pqr$, and let g_p, g_q, g_r be generators of $\mathbb{G}_p$, $\mathbb{G}_q$, and $\mathbb{G}_r$, respectively.
2. Choose random $h \in \mathbb{G}_p$ and $Q_1, Q_2 \in \mathbb{G}_q$, random $s, \gamma \in \mathbb{Z}_q$, and a random bit ν. Give to $\mathcal{A}$ the values $(N, \mathbb{G}, \mathbb{G}_T, \hat{e})$ as well as
$$g_p, \ g_q, \ g_r, \ h, \ g_p^s, \ h^s Q_1, \ g_p^\gamma Q_2, \ \hat{e}(g_p, h)^\gamma.$$
If $\nu = 0$ then give $\mathcal{A}$ the value $\hat{e}(g_p, h)^{\gamma s}$, while if $\nu = 1$ then give $\mathcal{A}$ a random element of $\mathbb{G}_T$.
3. $\mathcal{A}$ outputs a bit ν', and succeeds if $\nu' = \nu$.

The advantage of $\mathcal{A}$ is the absolute value of the difference between its success probability and 1/2.

Note that both the above assumptions imply the hardness of factoring N.

4 Our Main Construction

Our main construction is a predicate-only scheme where the set of attributes is $\Sigma = \mathbb{Z}_N^n$, and the class of predicates is $\mathcal{F} = \{f_{\boldsymbol{x}} \mid \boldsymbol{x} \in \mathbb{Z}_N^n\}$ with $f_{\boldsymbol{x}}(\boldsymbol{y}) = 1$ iff $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = 0 \bmod N$. (We use vectors of length n for convenience only.) In this section we construct a predicate-only scheme and give some intuition about our proof. In Appendix B we show how our scheme can be extended to give a full-fledged predicate encryption scheme. All proofs of security appear in the full version of our paper [17].

Intuition. In our construction, each ciphertext has associated with it a (secret) vector $\boldsymbol{x}$, and each secret key corresponds to a vector $\boldsymbol{v}$. The decryption procedure must check whether $\boldsymbol{x} \cdot \boldsymbol{v} = 0$, and reveal nothing about $\boldsymbol{x}$ but whether this is true. To do this, we will make use of a bilinear group $\mathbb{G}$ whose order N is the product of three primes p, q, and r. Let $\mathbb{G}_p$, $\mathbb{G}_q$, and $\mathbb{G}_r$ denote the subgroups of $\mathbb{G}$ having order p, q, and r, respectively. We will (informally) assume, as in [9], that a random element in any of these subgroups is indistinguishable from a random element of $\mathbb{G}$.[5] Thus, we can use random elements from one subgroup to "mask" elements from another subgroup.

At a high level, we will use these subgroups as follows. $\mathbb{G}_q$ will be used to encode the vectors $\boldsymbol{x}$ and $\boldsymbol{v}$ in the ciphertext and secret keys, respectively. Computation of the inner product $\langle \boldsymbol{v}, \boldsymbol{x} \rangle$ will be done in $\mathbb{G}_q$ (in the exponent), using

[5] This is only for intuition. Our actual computational assumption is given in Section 3.

the bilinear map. $\mathbb{G}_p$ will be used to encode an equation (again in the exponent) that evaluates to zero when decryption is done properly. This subgroup is used to prevent an adversary from improperly "manipulating" the computation (by, e.g., changing the ordering of components of the ciphertext or secret key, raising these components to some power, etc.). On an intuitive level, if the adversary tries to manipulate the computation in any way, then the computation occurring in the $\mathbb{G}_p$ subgroup will no longer yield the identity (i.e., will no longer yield 0 in the exponent), but will instead have the effect of "masking" the correct answer with a random element of $\mathbb{G}_p$ (which will invalidate the entire computation). Elements in $\mathbb{G}_r$ are used for "general masking" of terms in other subgroups; i.e., random elements of $\mathbb{G}_r$ will be multiplied with various components of the ciphertext (and secret key) in order to "hide" information that might be present in the $\mathbb{G}_p$ and $\mathbb{G}_q$ subgroups.

We now proceed to the formal description of our scheme.

4.1 The Construction

$\mathsf{Setup}(1^n)$ The setup algorithm first runs $\mathcal{G}(1^n)$ to obtain $(p, q, r, \mathbb{G}, \mathbb{G}_T, \hat{e})$ with $\mathbb{G} = \mathbb{G}_p \times \mathbb{G}_q \times \mathbb{G}_r$. Next, it computes g_p, g_q, and g_r as generators of $\mathbb{G}_p, \mathbb{G}_q$, and $\mathbb{G}_r$, respectively. It then chooses $R_{1,i}, R_{2,i} \in \mathbb{G}_r$ and $h_{1,i}, h_{2,i} \in \mathbb{G}_p$ uniformly at random for $i = 1$ to n, and $R_0 \in \mathbb{G}_r$ uniformly at random. The public parameters include $(N = pqr, \mathbb{G}, \mathbb{G}_T, \hat{e})$ along with:

$$PK = \big(g_p, \quad g_r, \quad Q = g_q \cdot R_0, \quad \{H_{1,i} = h_{1,i} \cdot R_{1,i}, \quad H_{2,i} = h_{2,i} \cdot R_{2,i}\}_{i=1}^n\big).$$

The master secret key SK is $\big(p, q, r, g_q, \{h_{1,i},\ h_{2,i}\}_{i=1}^n\big)$.

$\mathsf{Enc}_{PK}(\boldsymbol{x})$ Let $\boldsymbol{x} = (x_1, \ldots, x_n)$ with $x_i \in \mathbb{Z}_N$. This algorithm chooses random $s, \alpha, \beta \in \mathbb{Z}_N$ and $R_{3,i}, R_{4,i} \in \mathbb{G}_r$ for $i = 1$ to n. (Note: a random element $R \in \mathbb{G}_r$ can be sampled by choosing random $\delta \in \mathbb{Z}_N$ and setting $R = g_r^\delta$.) It outputs the ciphertext

$$C = \Big(C_0 = g_p^s, \quad \big\{\, C_{1,i} = H_{1,i}^s \cdot Q^{\alpha \cdot x_i} \cdot R_{3,i}, \quad C_{2,i} = H_{2,i}^s \cdot Q^{\beta \cdot x_i} \cdot R_{4,i} \big\}_{i=1}^n\Big).$$

$\mathsf{GenKey}_{SK}(\boldsymbol{v})$ Let $\boldsymbol{v} = (v_1, \ldots, v_n)$, and recall $SK = \big(p, q, r, g_q, \{h_{1,i},\ h_{2,i}\}_{i=1}^n\big)$. This algorithm chooses random $r_{1,i}, r_{2,i} \in \mathbb{Z}_p$ for $i = 1$ to n, random $R_5 \in \mathbb{G}_r$, random $f_1, f_2 \in \mathbb{Z}_q$, and random $Q_6 \in \mathbb{G}_q$. It then outputs

$$SK_{\boldsymbol{v}} = \left(\begin{array}{c} K = R_5 \cdot Q_6 \cdot \prod_{i=1}^n h_{1,i}^{-r_{1,i}} \cdot h_{2,i}^{-r_{2,i}}, \\ \Big\{ K_{1,i} = g_p^{r_{1,i}} \cdot g_q^{f_1 \cdot v_i}, \quad K_{2,i} = g_p^{r_{2,i}} \cdot g_q^{f_2 \cdot v_i} \Big\}_{i=1}^n \end{array} \right).$$

$\mathsf{Dec}_{SK_{\boldsymbol{v}}}(C)$ Let $C = \big(C_0, \{\, C_{1,i},\ C_{2,i}\}_{i=1}^n\big)$ and $SK_{\boldsymbol{v}} = \big(K, \{K_{1,i},\ K_{2,i}\}_{i=1}^n\big)$ be as above. The decryption algorithm outputs 1 iff

$$\hat{e}(C_0, K) \cdot \prod_{i=1}^n \hat{e}(C_{1,i}, K_{1,i}) \cdot \hat{e}(C_{2,i}, K_{2,i}) \stackrel{?}{=} 1.$$

Correctness. To see that correctness holds, let C and $SK_{\boldsymbol{v}}$ be as above. Then

$$
\begin{aligned}
&\hat{e}(C_0, K) \cdot \prod_{i=1}^{n} \hat{e}(C_{1,i}, K_{1,i}) \cdot \hat{e}(C_{2,i}, K_{2,i}) \\
&= \hat{e}\left(g_p^s, \quad R_5 Q_6 \prod_{i=1}^{n} h_{1,i}^{-r_{1,i}} h_{2,i}^{-r_{2,i}} \right) \\
&\quad \cdot \prod_{i=1}^{n} \hat{e}\left(H_{1,i}^s Q^{\alpha \cdot x_i} R_{3,i}, \quad g_p^{r_{1,i}} g_q^{f_1 \cdot v_i} \right) \cdot \hat{e}\left(H_{2,i}^s Q^{\beta \cdot x_i} R_{4,i}, \quad g_p^{r_{2,i}} g_q^{f_2 \cdot v_i} \right) \\
&= \hat{e}\left(g_p^s, \quad \prod_{i=1}^{n} h_{1,i}^{-r_{1,i}} h_{2,i}^{-r_{2,i}} \right) \\
&\quad \cdot \prod_{i=1}^{n} \hat{e}\left(h_{1,i}^s \cdot g_q^{\alpha \cdot x_i}, \quad g_p^{r_{1,i}} g_q^{f_1 \cdot v_i} \right) \cdot \hat{e}\left(h_{2,i}^s \cdot g_q^{\beta \cdot x_i}, \quad g_p^{r_{2,i}} g_q^{f_2 \cdot v_i} \right) \\
&= \prod_{i=1}^{n} \hat{e}(g_q, g_q)^{(\alpha f_1 + \beta f_2) x_i v_i} \quad = \quad \hat{e}(g_q, g_q)^{(\alpha f_1 + \beta f_2 \bmod q)\langle \boldsymbol{x}, \boldsymbol{v} \rangle},
\end{aligned}
$$

where α, β are random in $\mathbb{Z}_N$ and f_1, f_2 are random in $\mathbb{Z}_q$. If $\langle \boldsymbol{x}, \boldsymbol{v} \rangle = 0 \bmod N$, then the above evaluates to 1. If $\langle \boldsymbol{x}, \boldsymbol{v} \rangle \neq 0 \bmod N$ there are two cases: if $\langle \boldsymbol{x}, \boldsymbol{v} \rangle \neq 0 \bmod q$ then with all but negligible probability (over choice of α, β, f_1, f_2) the above evaluates to an element other than the identity. The other possibility is that $\langle \boldsymbol{x}, \boldsymbol{v} \rangle = 0 \bmod q$, in which case the above would always evaluate to 1; however, this would reveal a non-trivial factor of N and so this occurs with only negligible probability (recall, our assumptions imply hardness of finding a non-trivial factor of N).

There may appear to be some redundancy in our construction; for instance, the $C_{1,i}$ and $C_{2,i}$ components play almost identical roles. In fact we can view the encryption system as two parallel sub-systems linked via the C_0 component (and the corresponding component of the secret key). This two sub-system approach was first used by Boyen and Waters [11]; it can be viewed as a complex generalization of the Naor-Yung [18] "two-key" paradigm to the predicate encryption setting. A natural question is whether this redundancy can be eliminated to achieve better performance. While such a construction appears to be secure, our current proof (that utilizes a *non-interactive* assumption) relies in an essential way on having two parallel subsystems.

4.2 Proof Intuition

The most challenging aspect to providing a proof of our scheme naturally arises from the disjunctive capabilities of our system. In previous conjunctive systems (such as the one by Boneh and Waters [10]) the authors proved security by moving through a sequence of hybrid games, in which an encryption of a vector $\boldsymbol{x}$ was changed component-by-component to the encryption of a vector $\boldsymbol{y}$. The adversary could only ask for queries that did not match either $\boldsymbol{x}$ or $\boldsymbol{y}$, or queries that did not "look at" the components in which $\boldsymbol{x}$ and $\boldsymbol{y}$ differed. Thus, it was relatively straightforward to perform hybrid experiments over the components of $\boldsymbol{x}$ and $\boldsymbol{y}$ that differed, since the private keys given to the adversary did not "look at" these components.

In our proof an adversary will again try to determine whether a given ciphertext was encrypted with respect to $\boldsymbol{x}$ or $\boldsymbol{y}$. However, in our case the adversary can legally request a secret key for a vector $\boldsymbol{v}$ such that $\langle \boldsymbol{x}, \boldsymbol{v} \rangle = 0$ and $\langle \boldsymbol{y}, \boldsymbol{v} \rangle = 0$; i.e., it may obtain a key that should enable correct decryption in either case. This means that we cannot use the same proof strategy as in previous, conjunctive schemes. For example, if we change just one component at a time, then the "hybrid" vector used in an intermediate step will likely not be orthogonal to $\boldsymbol{v}$ (and the adversary will be able to detect this because its secret key will no longer decrypt the given ciphertext). Therefore, we need to use a sequence of hybrid games in which entire vectors are changed in one step, instead of using a sequence of hybrid games where the vector is changed component-by-component.

To do this we take advantage of the fact that, as noted earlier, our encryption scheme contains two parallel sub-systems. In our proof we will use hybrid games where a challenge ciphertext will be encrypted with respect to one vector in the first sub-system and a *different* vector in the second sub-system. (Note that such a ciphertext is ill-formed, since any valid ciphertext will always use the same vector in each sub-system.) Let $(\boldsymbol{a}, \boldsymbol{b})$ denote a ciphertext encrypted using vector $\boldsymbol{a}$ in the first sub-system and $\boldsymbol{b}$ in the second sub-system. To prove indistinguishability when encrypting to $\boldsymbol{x}$ (which corresponds to $(\boldsymbol{x}, \boldsymbol{x})$) and when encrypting to $\boldsymbol{y}$ (which corresponds to $(\boldsymbol{y}, \boldsymbol{y})$), we will prove indistinguishability of the following sequence of hybrid games:

$$(\boldsymbol{x}, \boldsymbol{x}) \approx (\boldsymbol{x}, \boldsymbol{0}) \approx (\boldsymbol{x}, \boldsymbol{y}) \approx (\boldsymbol{0}, \boldsymbol{y}) \approx (\boldsymbol{y}, \boldsymbol{y}).$$

Using this structure in our proof allows us to use a simulator that will essentially be able to work in one sub-system without "knowing" what is happening in the other one. The simulator embeds a "subgroup decision-like" assumption into the challenge ciphertext for each experiment. The structure of the challenge will determine whether a sub-system encrypts the given vector or the zero vector. Details of our proof and further discussion are given in the full version of our paper [17].

5 Applications of Our Main Construction

In this section we discuss some applications of predicate encryption schemes of the type constructed in this paper. Our treatment here is general and can be based on any predicate encryption scheme supporting "inner product" queries; we do not rely on any specific details of our construction.

Given a vector $\boldsymbol{x} \in \mathbb{Z}_N^\ell$, we denote by $f_{\boldsymbol{x}} : \mathbb{Z}_N^\ell \to \{0, 1\}$ the function such that $f_{\boldsymbol{x}}(\boldsymbol{y}) = 1$ iff $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = 0$. We define $\mathcal{F}_\ell \stackrel{\text{def}}{=} \{f_{\boldsymbol{x}} \mid \boldsymbol{x} \in \mathbb{Z}_N^\ell\}$. An *inner product encryption scheme of dimension* ℓ is an attribute-hiding predicate encryption scheme for the class of predicates $\mathcal{F}_\ell$.

5.1 Anonymous Identity-Based Encryption

As a warm-up, we show how anonymous identity-based encryption (IBE) can be recovered from any inner product encryption scheme of dimension 2. To generate the master public and secret keys for the IBE scheme, simply run the setup algorithm of the underlying inner product encryption scheme. To generate secret keys for the identity $I \in \mathbb{Z}_N$, set $\boldsymbol{I} := (1, I)$ and output the secret key for the predicate $f_{\boldsymbol{I}}$. To encrypt a message M for the identity $J \in \mathbb{Z}_N$, set $\boldsymbol{J}' := (-J, 1)$ and encrypt the message using the encryption algorithm of the underlying inner product encryption scheme and the attribute $\boldsymbol{J}'$. Since $\langle \boldsymbol{I}, \boldsymbol{J}' \rangle = 0$ iff $I = J$, correctness and security follow.

5.2 Hidden-Vector Encryption

Given a set Σ, let $\Sigma_\star = \Sigma \cup \{\star\}$. Hidden-vector encryption (HVE) [10] corresponds to a predicate encryption scheme for the class of predicates $\Phi_\ell^{\mathsf{hve}} = \{\phi^{\mathsf{hve}}_{(a_1,\ldots,a_\ell)} \mid a_1, \ldots, a_\ell \in \Sigma_\star\}$, where

$$\phi^{\mathsf{hve}}_{(a_1,\ldots,a_\ell)}(x_1, \ldots, x_\ell) = \begin{cases} 1 \text{ if, for all } i, \text{ either } a_i = x_i \text{ or } a_i = \star \\ 0 \text{ otherwise} \end{cases}.$$

A generalization of the ideas from the previous section can be used to realize hidden-vector encryption with $\Sigma = \mathbb{Z}_N$ from any inner product encryption scheme $(\mathsf{Setup}, \mathsf{GenKey}, \mathsf{Enc}, \mathsf{Dec})$ of dimension 2ℓ:

- The setup algorithm is unchanged.
- To generate a secret key corresponding to the predicate $\phi^{\mathsf{hve}}_{(a_1,\ldots,a_\ell)}$, first construct a vector $\boldsymbol{A} = (A_1, \ldots, A_{2\ell})$ as follows:

  $$\begin{aligned} &\text{if } a_i \neq \star: \ A_{2i-1} := 1, \quad A_{2i} := a_i \\ &\text{if } a_i = \star: \ A_{2i-1} := 0, \quad A_{2i} := 0. \end{aligned}$$

  Then output the key obtained by running $\mathsf{GenKey}_{SK}(f_{\boldsymbol{A}})$.
- To encrypt a message M for the attribute $x = (x_1, \ldots, x_\ell)$, choose random $r_1, \ldots, r_\ell \in \mathbb{Z}_N$ and construct a vector $\boldsymbol{X_r} = (X_1, \ldots, X_{2\ell})$ as follows:

  $$X_{2i-1} := -r_i \cdot x_i, \quad X_{2i} := r_i$$

  (where all multiplication is done modulo N). Then output the ciphertext $C \leftarrow \mathsf{Enc}_{PK}(\boldsymbol{X_r}, M)$.

To see that correctness holds, let $(a_1, \ldots, a_\ell)$, $\boldsymbol{A}$, $(x_1, \ldots, x_\ell)$, $\boldsymbol{r}$, and $\boldsymbol{X_r}$ be as above. Then:

$$\phi^{\mathsf{hve}}_{(a_1,\ldots,a_\ell)}(x_1, \ldots, x_\ell) = 1 \ \Rightarrow \ \forall \boldsymbol{r} : \langle \boldsymbol{A}, \boldsymbol{X_r} \rangle = 0 \ \Rightarrow \ \forall \boldsymbol{r} : f_{\boldsymbol{A}}(\boldsymbol{X_r}) = 1.$$

Furthermore, assuming $\gcd(a_i - x_i, N) = 1$ for all i:

$$\begin{aligned} \phi^{\mathsf{hve}}_{(a_1,\ldots,a_\ell)}(x_1, \ldots, x_\ell) = 0 &\Rightarrow \Pr_{\boldsymbol{r}}\left[\langle \boldsymbol{A}, \boldsymbol{X_r} \rangle = 0\right] = 1/N \\ &\Rightarrow \Pr_{\boldsymbol{r}}\left[f_{\boldsymbol{A}}(\boldsymbol{X_r}) = 1\right] = 1/N, \end{aligned}$$

which is negligible. Using this, one can prove security of the construction as well.

A straightforward modification of the above gives a scheme that is the "dual" of HVE, where the set of attributes is $(\Sigma_\star)^\ell$ and the class of predicates is $\bar{\Phi}_\ell^{\mathsf{hve}} = \{\bar{\phi}^{\mathsf{hve}}_{(a_1,\ldots,a_\ell)} \mid a_1, \ldots, a_\ell \in \Sigma\}$ with

$$\bar{\phi}^{\mathsf{hve}}_{(a_1,\ldots,a_\ell)}(x_1, \ldots, x_\ell) = \begin{cases} 1 \text{ if, for all } i, \text{ either } a_i = x_i \text{ or } x_i = \star \\ 0 \text{ otherwise} \end{cases} .$$

5.3 Predicate Encryption Schemes Supporting Polynomial Evaluation

We can also construct predicate encryption schemes for predicates corresponding to polynomial evaluation. Let $\Phi^{\text{poly}}_{\leq d} = \{f_p \mid p \in \mathbb{Z}_N[x], \deg(p) \leq d\}$, where

$$\phi_p(x) = \begin{cases} 1 \text{ if } p(x) = 0 \\ 0 \text{ otherwise} \end{cases}$$

for $x \in \mathbb{Z}_N$. Given an inner product encryption scheme $(\mathsf{Setup}, \mathsf{GenKey}, \mathsf{Enc}, \mathsf{Dec})$ of dimension $d+1$, we can construct a predicate encryption scheme for $\Phi^{\text{poly}}_{\leq d}$ as follows:

- The setup algorithm is unchanged.
- To generate a secret key corresponding to the polynomial $p(x) = a_d x^d + \cdots + a_0 x^0$, set $\boldsymbol{p} := (a_d, \ldots, a_0)$ and output the key obtained by running $\mathsf{GenKey}_{SK}(f_{\boldsymbol{p}})$.
- To encrypt a message M for the attribute $w \in \mathbb{Z}_N$, set $\boldsymbol{w} := (w^d \bmod N, \ldots, w^0 \bmod N)$ and output the ciphertext $C \leftarrow \mathsf{Enc}_{PK}(\boldsymbol{w}, M)$.

Since $p(w) = 0$ iff $\langle \boldsymbol{p}, \boldsymbol{w} \rangle = 0$, correctness and security follow.

The above shows that we can construct predicate encryption schemes where predicates correspond to univariate polynomials whose degree d is polynomial in the security parameter. This can be generalized to the case of polynomials in t variables, and degree at most d in each variable, as long as d^t is polynomial in the security parameter.

We can also construct schemes that are the "dual" of the above, in which attributes correspond to polynomials and predicates involve the evaluation of the input polynomial at some fixed point.

5.4 Disjunctions, Conjunctions, and Evaluating CNF and DNF Formulas

Given the polynomial-based constructions of the previous section, we can fairly easily build predicate encryption schemes for disjunctions of equality tests. For example, the predicate OR_{I_1,I_2}, where $\mathsf{OR}_{I_1,I_2}(x) = 1$ iff either $x = I_1$ or $x = I_2$, can be encoded as the univariate polynomial

$$p(x) = (x - I_1) \cdot (x - I_2),$$

which evaluates to 0 iff the relevant predicate evaluates to 1. Similarly, the predicate $\overline{\mathsf{OR}}_{I_1,I_2}$, where $\overline{\mathsf{OR}}_{I_1,I_2}(x_1,x_2) = 1$ iff either $x_1 = I_1$ or $x_2 = I_2$, can be encoded as the bivariate polynomial

$$p'(x_1, x_2) = (x_1 - I_1) \cdot (x_2 - I_2).$$

Conjunctions can be handled in a similar fashion. Consider, for example, the predicate AND_{I_1,I_2} where $\mathsf{AND}_{I_1,I_2}(x_1,x_1) = 1$ if both $x_1 = I_1$ and $x_2 = I_2$. Here, we determine the relevant secret key by choosing a random $r \in \mathbb{Z}_N$ and letting the secret key correspond to the polynomial

$$p''(x_1, x_2) = r \cdot (x_1 - I_1) + (x_2 - I_2).$$

If $\mathsf{AND}_{I_1,I_2}(x_1,x_1) = 1$ then $p''(x_1,x_2) = 0$, whereas if $\mathsf{AND}_{I_1,I_2}(x_1,x_1) = 0$ then, with all but negligible probability over choice of r, it will hold[6] that $p''(x_1,x_2) \neq 0$.

The above ideas extend to more complex combinations of disjunctions and conjunctions, and for boolean variables this means we can handle arbitrary CNF or DNF formulas. (For non-boolean variables we do not know how to directly handle negation.) As pointed out in the previous section, the complexity of the resulting scheme depends polynomially on d^t, where t is the number of variables and d is the maximum degree (of the resulting polynomial) in each variable.

5.5 Exact Thresholds

We conclude with an application that relies directly on inner product encryption. Here, we consider the setting of "fuzzy IBE" [20], which can be mapped to the predicate encryption framework as follows: fix a set $A = \{1, \ldots, \ell\}$ and let the set of attributes be all subsets of A. Predicates take the form $\Phi = \{\phi_S \mid S \subseteq A\}$ where $\phi_S(S') = 1$ iff $|S \cap S'| \geq t$, i.e., S and S' overlap in *at least* t positions. Sahai and Waters [20] show a construction of a payload-hiding predicate encryption scheme for this class of predicates.

We can construct a scheme where the attribute space is the same as before, but the class of predicates corresponds to overlap in *exactly* t positions. (Our scheme will also be attribute hiding.) Namely, set $\Phi' = \{\phi'_S \mid S \subseteq A\}$ with $\phi'_S(S') = 1$ iff $|S \cap S'| = t$. Then, given any inner product encryption scheme of dimension $\ell + 1$:

- The setup algorithm is unchanged.
- To generate a secret key for the predicate ϕ'_S, first define a vector $\boldsymbol{v} \in \mathbb{Z}_N^{\ell+1}$ as follows:

$$\begin{aligned} \text{for } 1 \leq i \leq \ell:\quad & v_i = 1 \text{ iff } i \in S \\ & v_{\ell+1} = 1. \end{aligned}$$

  Then output the key obtained by running $\mathsf{GenKey}_{SK}(f_{\boldsymbol{v}})$.

[6] In general, the secret key may leak the value of r in which case the adversary will be able to find x_1, x_2 such that $\mathsf{AND}_{I_1,I_2}(x_1,x_1) \neq 1$ yet $p''(x_1,x_2) = 0$. Since, however, we consider the "selective" notion of security (where the adversary must commit to x_1, x_2 at the outset of the experiment), this is not a problem in our setting.

- To encrypt a message M for the attribute $S' \subseteq A$, define a vector $\boldsymbol{v}'$ as follows:
$$\text{for } 1 \le i \le \ell\text{:} \quad v_i = 1 \text{ iff } i \in S'$$
$$v_{\ell+1} = -t \bmod N.$$
Then output the ciphertext $C \leftarrow \mathsf{Enc}_{PK}(\boldsymbol{v}', M)$.

Since $|S \cap S'| = t$ exactly when $\langle \boldsymbol{v}, \boldsymbol{v}' \rangle = 0$, correctness and security follow.

Disclaimer

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Department of Homeland Security.

References

1. S. Al-Riyami, J. Malone-Lee, and N. Smart. Escrow-free encryption supporting cryptographic workflow. *Intl. J. Information Security*, 5(4):217–229, 2006.
2. W. Bagga and R. Molva. Policy-based cryptography and applications. In *Financial Cryptography*, 2005.
3. J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy*, 2007.
4. D. Boneh and X. Boyen. Efficient selective-ID identity based encryption without random oracles. In *Advances in Cryptology — Eurocrypt 2004*.
5. D. Boneh and X. Boyen. Secure identity based encryption without random oracles. In *Advances in Cryptology — Crypto 2004*.
6. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *Advances in Cryptology — Crypto 2004*.
7. D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. Public-key encryption with keyword search. In *Advances in Cryptology — Eurocrypt 2004*.
8. D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. *SIAM J. Computing*, 32(3):586–615, 2003.
9. D. Boneh, E.-J. Goh, and K. Nissim. Evaluating 2-DNF formulas on ciphertexts. In *Theory of Cryptography Conference*, 2005.
10. D. Boneh and B. Waters. Conjunctive, subset, and range queries on encrypted data. In *Theory of Cryptography Conference*, 2007.
11. X. Boyen and B. Waters. Anonymous hierarchical identity-based encryption (without random oracles). In *Advances in Cryptology — Crypto 2006*.
12. R. Canetti, S. Halevi, and J. Katz. A forward-secure public-key encryption scheme. In *Advances in Cryptology — Eurocrypt 2003*.
13. C. Cocks. An identity based encryption scheme based on quadratic residues. In *Proc. IMA Intl. Conf. on Cryptography and Coding*, 2001.
14. C. Gentry. Practical identity-based encryption without random oracles. In *Advances in Cryptology — Eurocrypt 2006*.
15. V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM CCCS*, 2006.
16. J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *Advances in Cryptology — Eurocrypt 2008*.

17. J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. Cryptology ePrint Archive, Report 2007/404.
18. M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *STOC*, pages 427–437, 1990.
19. R. Ostrovsky, A. Sahai, and B. Waters. Attribute-based encryption with non-monotonic access structures. In *ACM CCCS*, 2007.
20. A. Sahai and B. Waters. Fuzzy identity-based encryption. In *Advances in Cryptology — Eurocrypt 2005.*
21. A. Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology — Crypto '84.*
22. E. Shi, J. Bethencourt, H. T.-H. Chan, D. X. Song, and A. Perrig. Multi-dimensional range queries over encrypted data. In *IEEE Symposium on Security and Privacy*, 2007.
23. B. Waters. Efficient identity-based encryption without random oracles. In *Advances in Cryptology — Eurocrypt 2005.*

A Security Definition for Inner-Product Encryption

Here, we re-state Definition 2 in the particular setting of our main construction, which is a predicate-only scheme where the set of attributes[7] is $\Sigma = \mathbb{Z}_N^n$ and the class of predicates is $\mathcal{F} = \{f_{\boldsymbol{x}} \mid \boldsymbol{x} \in \mathbb{Z}_N^n\}$ such that $f_{\boldsymbol{x}}(\boldsymbol{y}) = 1 \Leftrightarrow \langle \boldsymbol{x}, \boldsymbol{y} \rangle = 0$.

Definition 3. *A predicate-only encryption scheme for $\Sigma, \mathcal{F}$ as above is* attribute-hiding *if for all* PPT *adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in the following experiment is negligible in the security parameter n:*

1. $\mathsf{Setup}(1^n)$ *is run to generate keys PK, SK. This defines a value N which is given to $\mathcal{A}$.*
2. *$\mathcal{A}$ outputs $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{Z}_N^n$, and is then given PK.*
3. *$\mathcal{A}$ may adaptively request keys corresponding to vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_\ell \in \mathbb{Z}_N^n$, subject to the restriction that, for all i, $\langle \boldsymbol{v}_i, \boldsymbol{x} \rangle = \langle \boldsymbol{v}_i, \boldsymbol{y} \rangle$. In response, $\mathcal{A}$ is given the corresponding keys $SK_{\boldsymbol{v}_i} \leftarrow \mathsf{GenKey}_{SK}(f_{\boldsymbol{v}_i})$.*
4. *A random bit b is chosen. If $b = 0$ then $\mathcal{A}$ is given $C \leftarrow \mathsf{Enc}_{PK}(\boldsymbol{x})$, and if $b = 1$ then $\mathcal{A}$ is given $C \leftarrow \mathsf{Enc}_{PK}(\boldsymbol{y})$.*
5. *The adversary may continue to request keys for additional vectors, subject to the same restriction as before.*
6. *$\mathcal{A}$ outputs a bit b', and succeeds if $b' = b$.*

The advantage of $\mathcal{A}$ is the absolute value of the difference between its success probability and $1/2$.

B A Full-Fledged Predicate Encryption Scheme

In Section 4, we showed a construction of a *predicate-only* scheme. Here, we extend the previous scheme to obtain a full-fledged predicate encryption scheme in the sense of Definition 1. The construction follows.

[7] Technically speaking, both Σ and $\mathcal{F}$ depend on the public parameters (since N is generated as part of PK), but we ignore this technicality.

$\mathsf{Setup}(1^n)$ The setup algorithm first runs $\mathcal{G}(1^n)$ to obtain $(p, q, r, \mathbb{G}, \mathbb{G}_T, \hat{e})$ with $\mathbb{G} = \mathbb{G}_p \times \mathbb{G}_q \times \mathbb{G}_r$. Next, it computes g_p, g_q, and g_r as generators of $\mathbb{G}_p, \mathbb{G}_q$, and $\mathbb{G}_r$, respectively. It then chooses $R_{1,i}, R_{2,i} \in \mathbb{G}_r$ and $h_{1,i}, h_{2,i} \in \mathbb{G}_p$ uniformly at random for $i = 1$ to n, and $R_0 \in \mathbb{G}_r$ uniformly at random. It also chooses random $\gamma \in \mathbb{Z}_p$ and $h \in \mathbb{G}_p$. The public parameters include $(N = pqr, \mathbb{G}, \mathbb{G}_T, \hat{e})$ along with:

$$PK = \left(\begin{array}{c} g_p, \quad g_r, \quad Q = g_q \cdot R_0, \quad P = \hat{e}(g_p, h)^\gamma, \\ \left\{ H_{1,i} = h_{1,i} \cdot R_{1,i}, \quad H_{2,i} = h_{2,i} \cdot R_{2,i} \right\}_{i=1}^{n} \end{array} \right).$$

(This is identical to the public parameters of the predicate-only scheme with the exception of the additional element $P = \hat{e}(g_p, h)^\gamma$.) The master secret key SK is $\big(p, q, r, g_q, h^{-\gamma}, \{h_{1,i},\ h_{2,i}\}_{i=1}^n\big)$.

$\mathsf{Enc}_{PK}(\boldsymbol{x}, M)$ View M as an element of $\mathbb{G}_T$, and let $\boldsymbol{x} = (x_1, \ldots, x_n)$ with $x_i \in \mathbb{Z}_N$. This algorithm chooses random $s, \alpha, \beta \in \mathbb{Z}_N$ and $R_{3,i}, R_{4,i} \in \mathbb{G}_r$ for $i = 1$ to n. It outputs the ciphertext

$$C = \left(\begin{array}{c} C' = M \cdot P^s, \quad C_1 = g_p^s, \\ \left\{ C_{1,i} = H_{1,i}^s \cdot Q^{\alpha \cdot x_i} \cdot R_{3,i}, \quad C_{2,i} = H_{2,i}^s \cdot Q^{\beta \cdot x_i} \cdot R_{4,i} \right\}_{i=1}^{n} \end{array} \right).$$

$\mathsf{GenKey}_{SK}(\boldsymbol{v})$ Let $\boldsymbol{v} = (v_1, \ldots, v_n)$. This algorithm chooses random $r_{1,i}, r_{2,i} \in \mathbb{Z}_p$ for $i = 1$ to n, random $f_1, f_2 \in \mathbb{Z}_q$, random $R_5 \in \mathbb{G}_r$, and random $Q_6 \in \mathbb{G}_q$. It then outputs

$$SK_{\boldsymbol{v}} = \left(\begin{array}{c} K = R_5 \cdot Q_6 \cdot h^{-\gamma} \cdot \prod_{i=1}^n h_{1,i}^{-r_{1,i}} \cdot h_{2,i}^{-r_{2,i}}, \\ \left\{ K_{1,i} = g_p^{r_{1,i}} \cdot g_q^{f_1 \cdot v_i}, \quad K_{2,i} = g_p^{r_{2,i}} \cdot g_q^{f_2 \cdot v_i} \right\}_{i=1}^{n} \end{array} \right).$$

$\mathsf{Dec}_{SK_{\boldsymbol{v}}}(C)$ Let C and $SK_{\boldsymbol{v}}$ be as above. The decryption algorithm outputs

$$C' \cdot \hat{e}(C_1, K) \cdot \prod_{i=1}^{n} \hat{e}(C_{1,i}, K_{1,i}) \cdot \hat{e}(C_{2,i}, K_{2,i}).$$

As we have described it, decryption never returns an error (i.e., even when $\langle \boldsymbol{v}, \boldsymbol{x} \rangle \neq 0$). We will show below that when $\langle \boldsymbol{v}, \boldsymbol{x} \rangle \neq 0$ then the output is essentially a random element in the order-q subgroup of $\mathbb{G}_T$. By restricting the message space to some efficiently-recognizable set of negligible density in this subgroup, we recover the desired semantics by returning an error if the recovered message does not lie in this space.

Correctness. Let C and $SK_{\boldsymbol{v}}$ be as above. Then

$$
\begin{aligned}
& C' \cdot \hat{e}(C_1, K) \cdot \prod_{i=1}^{n} \hat{e}(C_{1,i}, K_{1,i}) \cdot \hat{e}(C_{2,i}, K_{2,i}) \\
&= M \cdot P^s \cdot \hat{e}\left(g_p^s, \quad R_5 Q_6 h^{-\gamma} \prod_{i=1}^{n} h_{1,i}^{-r_{1,i}} h_{2,i}^{-r_{2,i}} \right) \\
&\quad \cdot \prod_{i=1}^{n} \hat{e}\left(H_{1,i}^s Q^{\alpha \cdot x_i} R_{3,i}, \quad g_p^{r_{1,i}} g_q^{f_1 \cdot v_i} \right) \cdot \hat{e}\left(H_{2,i}^s Q^{\beta \cdot x_i} R_{4,i}, \quad g_p^{r_{2,i}} g_q^{f_2 \cdot v_i} \right) \\
&= M \cdot P^s \cdot \hat{e}\left(g_p^s, \quad h^{-\gamma} \prod_{i=1}^{n} h_{1,i}^{-r_{1,i}} h_{2,i}^{-r_{2,i}} \right) \\
&\quad \cdot \prod_{i=1}^{n} \hat{e}\left(h_{1,i}^s g_q^{\alpha \cdot x_i}, \quad g_p^{r_{1,i}} g_q^{f_1 \cdot v_i} \right) \cdot \hat{e}\left(h_{2,i}^s g_q^{\beta \cdot x_i}, \quad g_p^{r_{2,i}} g_q^{f_2 \cdot v_i} \right) \\
&= M \cdot P^s \cdot \hat{e}(g_p, h)^{-\gamma s} \cdot \prod_{i=1}^{n} \hat{e}(g_q, g_q)^{(\alpha f_1 + \beta f_2) x_i v_i} \quad = \quad M \cdot \hat{e}(g_q, g_q)^{(\alpha f_1 + \beta f_2) \langle \boldsymbol{x}, \boldsymbol{v} \rangle}.
\end{aligned}
$$

If $\langle \boldsymbol{x}, \boldsymbol{v} \rangle = 0 \bmod N$, then the above evaluates to M. If $\langle \boldsymbol{x}, \boldsymbol{v} \rangle \neq 0 \bmod N$ there are two cases: if $\langle \boldsymbol{x}, \boldsymbol{v} \rangle \neq 0 \bmod q$ then the above evaluates to an element whose distribution is statistically close to uniform in the order-q subgroup of $\mathbb{G}_T$. (Recall that α, β are chosen at random.) It is possible that $\langle \boldsymbol{x}, \boldsymbol{v} \rangle = 0 \bmod q$, in which case the above always evaluates to M; however, this reveals a non-trivial factor of N and so an adversary can cause this condition to occur with only negligible probability.