Pattern Matching in Encrypted Stream from Inner Product Encryption

Élie Bouscatié[1,2], Guilhem Castagnos[2], and Olivier Sanders[1]

[1] Orange Labs, Applied Crypto Group, Cesson-Sévigné, France
[2] Université de Bordeaux, INRIA, CNRS, IMB UMR 5251, F-33405 Talence, France

Abstract. Functional encryption features secret keys, each associated with a key function f, which allow to directly recover $f(x)$ from an encryption of x, without learning anything more about x. This property is particularly useful when delegating data processing to a third party as it allows the latter to perfom its task while ensuring minimum data leakage. However, this generic term conceals a great diversity in the cryptographic constructions that strongly differ according to the functions f they support.

A recent series of works has focused on the ability to search a pattern within a data stream, which can be expressed as a function f. One of the conclusions of these works was that this function f was not supported by the current state-of-the-art, which incited their authors to propose a new primitive called Stream Encryption supporting Pattern Matching (SEPM). Some concrete constructions were proposed but with some limitations such as selective security or reliance on non-standard assumptions.

In this paper, we revisit the relations between this primitive and two major subclasses of functional encryption, namely Hidden Vector Encryption (HVE) and Inner Product Encryption (IPE). We indeed first exhibit a generic transformation from HVE to SEPM, which immediately yields new efficient SEPM constructions with better features than existing ones. We then revisit the relations between HVE and IPE and show that we can actually do better than the transformation proposed by Katz, Sahai and Waters in their seminal paper on predicate encryption. This allows to fully leverage the vast state-of-the-art on IPE which contains adaptively secure constructions proven under standard assumptions. This results in countless new SEPM constructions, with all the features one can wish for. Beyond that, we believe that our work sheds a new light on the relations between IPE schemes and HVE schemes and in particular shows that some of the former are more suitable to construct the latter.

Keywords: Pattern Matching · Functional Encryption · Hidden Vector Encryption · Inner Product Encryption

1 Introduction

Outsourcing IT services has become very common worldwide[3] for multiple reasons ranging from costs reduction to improved services. Whatever the actual reason is, the concrete consequence for the company that delegates such services is that a third party ends up with its data in clear because of the well-known limitations of standard encryption.

Ideally, this third party should only learn the minimal information necessary for performing the requested processing, which has motivated the design of countless encryption schemes compatible with specific processing. Such schemes belong to the realm of functional encryption [BSW11], where the third party recovers a function $f(x)$ from an encryption of x without learning anything else about x, with minimal interaction. Of course, the function f, and hence the encryption scheme, strongly depends on the considered application, which explains the profusion of papers related to this topic.

1.1 Related Works

As functional encryption schemes supporting a large set of functions (e.g. [JLS21, AMVY21]) tend to be quite complex, a variety of schemes have been tailored to a specific function and therefore to the requirements of specific use-cases. In this paper, we will focus on the ability to detect specific patterns within an encrypted string (also called *pattern matching*), which is very useful for many scenarios such as Intrusion Detection Systems (IDS) or search on genomic data.

At first glance, this problem seems to be directly related to the area of *searchable* encryption (e.g. [BDOP04, DPP18]) where one can decide if a ciphertext C encrypts some data x provided that it has received a trapdoor T_x specific to x. Unfortunately, as noted in [DFOS18], this does not solve the problem of pattern matching because there is a huge difference between deciding whether C encrypts x or whether C encrypts a string y that contains x as a substring. One could try to follow the tokenization approach of [SLPR15], which consists in splitting the encrypted string into many overlapping substrings that will be individually encrypted using searchable encryption. However, this only works if all searched patterns are strings of a unique same length, which is not true in practice[4]. Adaptations of this approach are possible but lead to other problems, as also discussed in [DFOS18]. We also note that techniques tailored to use-cases related to external storage (*e.g.* [CS15, LL18]) do not work in our context as, in the latter, the entity performing the test is the data owner which allows to reveal more information. In our case, the test is performed by a third entity which should only learn the result of the pattern matching.

Similarly, previous papers on pattern matching (*e.g.* [SLPR15, DFOS18], [BCC20, BCS21]) dismissed so-called predicate encryption [KSW08], a sub-class

[3] https://sumatosoft.com/blog/it-outsourcing-2019-overview-trends
[4] See e.g. the length distribution of Snort rules https://snort.org/downloads#rules

of functional encryption where f is essentially a boolean function, as they noticed, here again, that this primitive does not exactly answer our problem. More specifically, they considered two related primitives, namely Inner Product Encryption (IPE) [KSW08] and Hidden Vector Encryption (HVE) [BW07] that seem to provide the kind of features one needs for pattern matching. The former allows to test if the inner product of some vector associated with the ciphertext and some other vector associated with the secret key is zero whereas the latter allows to test if a ciphertext is associated with a vector of attributes, potentially with wildcards. However, they noted that using such schemes for pattern matching on data streams require to provide, for each searched pattern, a secret key linear in the size of the stream, which quickly becomes cumbersome. This is truly unfortunate as this area of cryptography has been extensively studied, with very impressive results. For example, if we focus on the specific case of Inner Product Encryption (*e.g.* [KSW08, OT11, OT12a, OT12b, Ram16, CGW18]), one can find schemes with remarkable features such as adaptive security, proofs under standard assumptions, etc.

This state of affairs led very recent papers [DFOS18, BCC20, BCS21] to define a new primitive called Stream Encryption supporting Pattern Matching (SEPM), directly tailored to the pattern matching use-case. Conceptually, this primitive is close to predicate encryption but aims at providing constant size secret keys that yet allow to search the patterns anywhere in the stream. As this feature seemed incompatible with IPE or HVE, the authors of these papers started from scratch with constructions only achieving selective security and, for most of them, under very strong interactive assumptions.

1.2 Our Contributions

In this paper we completely revisit SEPM by identifying generic and efficient transformations linking IPE and SEPM through HVE. The direct consequence of our work is that it allows to leverage all the state-of-the art related to IPE and HVE to directly build SEPM with new features. More specifically, we proceed in two main steps, as follows.

Our natural starting point is HVE for two reasons. Firstly, by identifying the caracters of our data streams with the attributes of HVE, one gets the ability to search patterns while ensuring data privacy. Secondly, HVE supports wildcards, that is, a special caracter $\star$ that matches all caracters. This allows to detect more advanced patterns such as `ab`$\star\star$`cd`, meaning `ab` followed by `cd` with an offset of 2. This kind of patterns is necessary in many applications such as IDS, as illustrated by the Snort data rules mentionned above. Moreover, when it comes to data sream, this allows to test the presence of some pattern `abc` at any position within the stream by providing secret keys for the patterns `abc`$\star\star\ldots$, $\star$`abc`$\star\ldots$, etc. Obviously, the natural downside of this approach is that one must issue secret keys for any possible position of the pattern, which quickly becomes cumbersome. This is actually the reason why [DFOS18] dismissed HVE as a potential solution. The latter paper managed to have constant-size secret keys allowing to search a pattern *everywhere* in the data stream but at the cost of a very large public key.

In a follow-up work, [BCC20] addressed the problem of the large public key through a technique called fragmentation which consists in splitting the stream into overlapping and redundant substrings. The same technique was used in [BCS21] to construct a scheme with better complexity and security.

In this work we show that this fragmentation technique is actually much more powerful than initially thought because, intuitively, it allows to reduce the problem of finding a pattern anywhere in a stream to the one of searching this pattern within fixed-length substrings, called fragments, which limits the consequences of the problem mentioned above. This allows us to propose a generic transformation from HVE to SEPM which automatically improves the state-of-the-art of SEPM.

Once this is done, we try to further improve our result by trying to connect SEPM to IPE, which has been much more studied than HVE.

Here, we do not start from scratch as Katz, Sahai and Waters [KSW08] already showed a relation between IPE and HVE. More specifically, they noted that if one encrypts a vector $(\mathbf{xr}, -\mathbf{r}) \in \mathbf{F}_p^{2n}$ with an IPE scheme, where $\mathbf{xr}$ denotes the element-wise product of the vector $\mathbf{x}$ and some random vector $\mathbf{r}$, then one can test if $\mathbf{x} = \mathbf{k}$ (and thus get an HVE scheme) given an IPE secret key for $(1, \dots, 1, \mathbf{k}) \in \mathbf{F}_p^{2n}$. Indeed, one can note that the scalar product between these two vectors is 0. Obviously, the opposite must be true and this is the purpose of the vector $\mathbf{r}$. Without this randomness $\mathbf{r}$ in the ciphertext, one could indeed easily construct another secret key that would cancel $\mathbf{x}$ without being equal to $\mathbf{x}$. This, combined with the way it handles wildcards, described in the body of this paper, explains why this transformation doubles the size of the original vectors. We stress the importance of $\mathbf{r}$ being hidden in the ciphertext by the security of the IPE scheme. Surprisingly, this fact is not mentioned by [KSW08] to prove the security of their transformation. Actually, the arguments they provide still apply to our next transformation but we show a counterexample in this case, *i.e.* an secure IPE scheme whose conversion is not secure. As a warm-up, we provide a complete proof of security for their transformation, which allows to identify the subtleties that arise in the process.

As the KSW transformation entails a doubling of the ciphertext size, we propose in this paper a new conversion with a ratio very close to 1. Our core idea, which allows us to handle wildcards with fewer coordinates, is to move the randomness $\mathbf{r}$ to the secret key in the following way. We set our ciphertext as $(\mathbf{x}, -1) \in \mathbf{F}_p^{n+1}$ whereas the secret key is $(\mathbf{r}, \langle \mathbf{k}, \mathbf{r} \rangle)$. Here again, we get an HVE scheme that allows to test whether $\mathbf{k} = \mathbf{x}$, but with a better efficiency. However, proving security of the resulting transformation is much more complex. Intuitively, the problem stems from the fact that security inherently depends on the secrecy of $\mathbf{r}$. When $\mathbf{r}$ is embedded in the ciphertext, as in the KSW transformation, one can rely on the security of the encryption scheme itself. In our case, this is no longer possible as there is no equivalent property for the secret key itself. Theoretically, one could learn $\mathbf{r}$ from the secret key and thus break security of the conversion. We study this problem more thoroughly and show that it actually depends on the exact model we consider.

In the case of selective security, we show that an adversary is unable to exploit this problem and so that our conversion IPE to HVE remains secure for all schemes.

In the adaptive case, we cannot prove such a result in general, and actually show a counterexample with an IPE scheme from the litterature. Fortunately, we show that we can circumvent this problem if the underlying IPE scheme satisfies a new property that we formalize. This property concerns the secret keys of the IPE schemes and we show that many such constructions naturally achive it under the discrete logarithm assumption. With this additional property, we are at last able to prove adaptive security of the HVE schemes resulting from this conversion. This allows to leverage the whole state-of-the-art of IPE schemes with a better efficiency than with the KSW conversion. Besides that, this shows that some IPE schemes are more suitable to design HVE schemes, which clarifies the relation between these two primitives.

In a last section, we draw the consequences of our generic conversions. Where as all known SEPM proposals only achieved selective security under strong assumptions, we show that it is possible to achieve adaptive security under DLIN by loosing only a constant factor on efficiency.

2 Definitions

In this section, we first give useful notations for the context of pattern matching and then review notions of functional encryption still in this context. In particular, we consider Hidden Vector Encryption and Inner Product Encryption, two primitives that we will use to construct Stream Encryption supporting Pattern Matching. As we shall see, we will consider predicate only versions of these two primitives, viewing attributes as messages.

Notations and vocabulary. We denote by $\mathbf{N}$ the set of positive integers and for any $n \in \mathbf{N}$, we note $[\![n]\!] := \{1, \dots, n\}$. For any set A, we write $x \xleftarrow{\$} A$ to say that x is chosen uniformly at random in A, we note $A^* := \bigcup_{i \geq 1} A^i$ where A^i is the usual Cartesian product $A \times \cdots \times A$ and for an element $\mathbf{a} \in A^*$, we note $\mathsf{len}(\mathbf{a})$ the non negative integer such that $\mathbf{a} \in A^{\mathsf{len}(\mathbf{a})}$ and call the *length* of $\mathbf{a}$.
Let $\star$ be the wildcard symbol and Σ a finite alphabet that does not contain $\star$. An element $\mathbf{x} \in \Sigma^*$ is called a *string*, an element $\mathbf{k} \in (\Sigma \cup \{\star\})^*$ a *pattern* and the set $\mathsf{supp}(\mathbf{k}) := \{1 \leq i \leq \mathsf{len}(\mathbf{k}) : k_i \neq \star\}$ is called the *support* of $\mathbf{k}$.
We say that the pattern $\mathbf{k}$ *matches* the string $\mathbf{x}$ if $\mathsf{len}(\mathbf{k}) = \mathsf{len}(\mathbf{x})$, and

$$\forall i \in \mathsf{supp}(\mathbf{k}), k_i = x_i.$$

More generally, if $\mathsf{len}(\mathbf{k}) \leq \mathsf{len}(\mathbf{x})$, then for any $1 \leq i \leq \mathsf{len}(\mathbf{x}) - \mathsf{len}(\mathbf{k}) + 1$, we say that the pattern $\mathbf{k}$ *matches* the string $\mathbf{x}$ *at position* i if

$$\forall j \in \mathsf{supp}(\mathbf{k}), k_j = x_{i+j-1}.$$

Other notations are differed to the beginning of section 4 where the choice of Σ becomes more specific.

2.1 Functional Encryption

Syntax. We recall the general definition of functional encryption as introduced in [BSW11]. A functionality F defined over (K, X) is a function $F : K \times X \rightarrow \{0,1\}^*$ described as a (deterministic) Turing Machine. The set K is called the key space and the set X is called the plaintext space.

A functional encryption scheme for the functionality F enables one to evaluate $F(\mathbf{k}, \mathbf{x})$ given the encryption of $\mathbf{x}$ and a secret key $\mathsf{sk}_{\mathbf{k}}$ for $\mathbf{k}$. The algorithm for evaluation $F(\mathbf{k}, \mathbf{x})$ using $\mathsf{sk}_{\mathbf{k}}$ is called decrypt. More precisely, a functional encryption scheme is defined as a tuple of four PPT algorithms $(\mathtt{setup}, \mathtt{keygen}, \mathtt{enc}, \mathtt{dec})$ as follows:

- $(pp, \mathsf{pk}, \mathsf{mk}) \leftarrow \mathtt{setup}(1^\lambda)$, generates public parameters that are implicit inputs of the other algorithms, a public key and a master secret key;
- $\mathsf{sk}_{\mathbf{k}} \leftarrow \mathtt{keygen}(\mathsf{mk}, \mathbf{k})$, generates a secret key for $\mathbf{k}$;
- $\mathbf{c} \leftarrow \mathtt{enc}(\mathsf{pk}, \mathbf{x})$, encrypts the message $\mathbf{x}$;
- $y \leftarrow \mathtt{dec}(\mathsf{sk}_{\mathbf{k}}, \mathbf{k}, \mathbf{c})$, uses sk to compute $y \in \{0,1\}^*$ from $\mathbf{c}$.

Correctness. As we are essentially interested in pattern matching applications, the definition of correctness that we give will be associated with the notion of *false positive* (a pattern is mistakenly detected). However, for all the schemes that we consider there is no false negative, patterns that are present, will always be detected. Moreover, although we could provide a generic definition of correctness, we choose to distinguish two relevant cases in our context, the one where the output of F is 0 or 1 and the one where this output can be parsed as some finite subset of $\mathbf{N}$. It will lead to more intuitive definitions. These definitions are similar to those in [ABC$^+$08] but we consider a slightly weaker notion of false positive.
A functional encryption scheme with functionality F such that $F(\mathbf{k}, \mathbf{x}) \in \{0,1\}$ is correct if for all $\mathbf{k} \in K, \mathbf{x} \in X$,

- $F(\mathbf{k}, \mathbf{x}) = 1 \implies \mathtt{dec}(\mathtt{keygen}(\mathsf{mk}, \mathbf{k}), \mathbf{k}, \mathtt{enc}(\mathsf{pk}, \mathbf{x})) = 1$.
- $F(\mathbf{k}, \mathbf{x}) = 0$ and $\mathtt{dec}(\mathtt{keygen}(\mathsf{mk}, \mathbf{k}), \mathbf{k}, \mathtt{enc}(\mathsf{pk}, \mathbf{x})) = 1$ (*i.e.* a false positive) occurs with negligible probability $\mu(\lambda)$ over the coins of all the algorithms.

A functional encryption scheme with functionality F such that $F(\mathbf{k}, \mathbf{x})$ is a finite subset of $\mathbf{N}$ is correct if for all $\mathbf{k} \in K, \mathbf{x} \in X, i \in \mathbf{N}$,

- $i \in F(\mathbf{k}, \mathbf{x}) \implies i \in \mathtt{dec}(\mathtt{keygen}(\mathsf{mk}, \mathbf{k}), \mathbf{k}, \mathtt{enc}(\mathsf{pk}, \mathbf{x}))$.
- $i \notin F(\mathbf{k}, \mathbf{x})$ and $i \in \mathtt{dec}(\mathtt{keygen}(\mathsf{mk}, \mathbf{k}), \mathbf{k}, \mathtt{enc}(\mathsf{pk}, \mathbf{x}))$ (*i.e.* a false positive) occurs with negligible probability $\mu(\lambda)$ over the coins of all the algorithms.

Security. We here recall the classical $\mathsf{IND-CPA}$ security for functional encryption schemes.

Definition 1 **($\mathsf{IND-CPA}$ for functional encryption).** *A Functional Encryption scheme is $\mathsf{IND-CPA}$ secure if no probabilistic polynomial time adversary $\mathcal{A}$ has a non-negligible advantage in the following game, $\mathtt{Exp}_{\mathcal{A}}^{\mathsf{IND-CPA}}$:*

Setup: *run $(pp, \mathsf{pk}, \mathsf{mk}) \leftarrow \mathtt{setup}(1^\lambda)$ and give pp, pk to $\mathcal{A}$.*
Query Phase 1: *$\mathcal{A}$ submits queries $\mathbf{k} \in K$ and gets $\mathsf{sk}_{\mathbf{k}} \leftarrow \mathtt{keygen}(\mathsf{mk}, \mathbf{k})$*
Challenge: *$\mathcal{A}$ submits two messages $\mathbf{m}^{(0)}, \mathbf{m}^{(1)} \in X$ such that every queried pattern $\mathbf{k}$ follows the natural restriction:*

$$F(\mathbf{k}, \mathbf{m}^{(0)}) = F(\mathbf{k}, \mathbf{m}^{(1)}). \tag{1}$$

The challenger chooses $\beta \xleftarrow{\$} \{0,1\}$ and gives $c \leftarrow \mathtt{enc}(\mathsf{pk}, \mathbf{m}^{(\beta)})$ to $\mathcal{A}$.
Query Phase 2: *$\mathcal{A}$ can issue key queries as before but subject to restriction (1).*
Guess: *$\mathcal{A}$ eventually outputs a bit β' in $\{0,1\}$.*

The advantage of $\mathcal{A}$ is defined as $\left|\Pr[\beta = \beta'] - \frac{1}{2}\right|$.

This definition is sometimes called *adaptive* $\mathsf{IND-CPA}$ *security*. In a weaker model, *selective security*, the adversary $\mathcal{A}$ has to choose $\mathbf{m}^{(0)}, \mathbf{m}^{(1)}$ at the beginning of the game, before seeing the public key and public parameters and before Query Phase 1.

2.2 Some classes of Functional Encryption

Hidden Vector Encryption. This primitive, HVE for short, was introduced in [BW07]. The original definition follows the paradigm of predicate encryption. A secret key encapsulates a key pattern (a string with possible wildcards) while a ciphertext encrypts both an attribute string and a payload message. A first security notion, called *payload hiding*, ensures that a ciphertext hides all information about the payload message unless one has a secret key for a key pattern that matches the attribute string, in this case, he recovers the payload message. An additional security notion, called *attribute hiding* (cf. [KSW08]), ensures that a ciphertext hides all information about the attribute string and decryption does not reveal any information about the attribute string other than the fact that it matches the key pattern or not.

It was noted in [DFOS18] that an attribute hiding HVE can be used for pattern matching on the attribute without revealing extra information about it but with the strong limitations recalled in our introduction. While other applications may not consider attribute hiding or weaker versions of it, this notion is crucial to achieve this purpose.

In many works on HVE (*e.g.* [KSW08, DIP13]) a first building block is presented, called a *predicate-only* HVE. This focuses on the attribute, not considering the payload message. The reason behind this is that attribute hiding is the hardest part to achieve (especially when adaptive security is targeted as in [OT10]) and the full-fledged HVE is then obtained using a key encapsulation mechanism. In the following, we will abuse the terminology of [KSW08], as in [DIP13], and simply refer to *predicate-only* HVE as HVE. Thus what we called an attribute will be seen as the message and the attribute hiding security notion will coincide with the classical $\mathsf{IND-CPA}$ security notion for functional encryption.

This gives the following definition.

Definition 2 ((Predicate-Only) Hidden Vector Encryption). *An n-HVE scheme for some integer n is formally described as a functional encryption scheme where:*

1. *The key space K is $(\Sigma \cup \{\star\})^n$.*
2. *The plaintext space X is Σ^n.*
3. *The functionality is $F_{\mathrm{HVE}} \colon K \times X \longrightarrow \{0,1\}$*

$$(\mathbf{k}, \mathbf{x}) \longmapsto F_{\mathrm{HVE}}(\mathbf{k}, \mathbf{x}) = \begin{cases} 1 & \textit{if } \mathbf{k} \textit{ matches } \mathbf{x}, \\ 0 & \textit{otherwise.} \end{cases}$$

Inner Product Encryption. We will also consider Inner Product Encryption (IPE), a primitive introduced by [KSW08] who additionally noted a relation to HVE (we will review and improve this result in Sections 4 and 5). Here keys and attributes are vectors and one tests whether their inner product is zero, instead of testing matching. Again we consider a predicate-only version of this primitive which is sufficient for our purposes. We adapt the definition from [BSW11] that uses the vector space $\mathbf{F}_p^n$ to define an IPE.

Definition 3 ((Predicate-Only) Inner product Encryption). *An n-IPE scheme for some integer n is formally described as a functional encryption scheme where:*

1. *The setup algorithm defines a randomly chosen prime p of length λ, where λ is the security parameter.*
2. *The key space K and plaintext space X are $\mathbf{F}_p^n$.*
3. *The functionality is $F_{\mathrm{IPE}} \colon K \times X \longrightarrow \{0,1\}$*

$$(\mathbf{u}, \mathbf{v}) \longmapsto F_{\mathrm{IPE}}(\mathbf{u}, \mathbf{v}) = \begin{cases} 1 & \textit{if } \langle \mathbf{u}, \mathbf{v} \rangle = 0, \\ 0 & \textit{otherwise.} \end{cases}$$

Stream Encryption supporting Pattern Matching. This primitive has been recently considered in [BCC20,BCS21]. It can be formalized as a functional encryption scheme as follows. A plaintext x is a stream, an element of Σ^*. Given a pattern $\mathbf{k}$ of length upper bounded by n, the functionality returns all the integers i such that $\mathbf{k}$ matches $\mathbf{x}$ at position i.

Definition 4 (Stream Encryption supporting Pattern Matching). *An n-SEPM scheme for some integer n is formally described as a functional encryption scheme where:*

1. *The key space K is $\bigcup_{i=1}^{n}(\Sigma \cup \{\star\})^i$ completed by the empty key ϵ.*
2. *The plaintext space X is Σ^*.*
3. *The functionality is defined as*

$$\begin{aligned} F_{\text{SEPM}} \colon (K \setminus \{\epsilon\}) \times X &\longrightarrow \{S \subset \mathbf{N} : S \text{ is finite}\} \\ (\mathbf{k}, \mathbf{x}) &\longmapsto F_{\text{SEPM}}(\mathbf{k}, \mathbf{x}) = \{i : \mathbf{k} \text{ matches } \mathbf{x} \text{ at position } i\}. \end{aligned}$$

And we let $F_{\text{SEPM}}(\epsilon, \mathbf{x}) = \mathsf{len}(\mathbf{x})$ to leak the length of the message intentionally.

3 From HVE to SEPM through fragmentation

There are several ways of designing a public key encryption scheme supporting pattern matching but, as explained in [DFOS18], they usually lead to systems suffering from very concrete limitations, such as the restriction of the set of possible patterns to strings of a unique same length, secret keys (called trapdoors in [DFOS18]) whose size is linear in the maximum size of the encryption stream, etc. The authors of [DFOS18] proposed an alternative primitive that theoretically addresses these issues but with rather poor performance. Two follow-up works [BCC20] and [BCS21] improved this by introducing SEPM schemes. They both extensively rely on a technique called *fragmention* that enables to circumvent the need for shiftable trapdoors identified in [DFOS18] by splitting the stream into fragments with some redundancy.

We first recall this technique and then show, as a first contribution, that fragmentation creates a strong relation between SEPM and HVE: we expose a generic conversion from a $2d$-HVE scheme to a n-SEPM with $n = d + 1$ and its security.

3.1 Fragmentation

Let n be an upper bound on the length of the patterns supported by the SEPM scheme. To enable pattern matching for string $\mathbf{x}$ of any length, the fragmentation technique splits the latter into overlapping substrings $\mathbf{x}_i$ of size $2d$, where $d := n - 1$, as follows

$$\mathbf{x}_i = (x_{(i-1)d+1}, \ldots, x_{(i+1)d}) \qquad \text{for } i = 1, \ldots, \left\lceil \frac{\mathrm{len}(\mathbf{x})}{d} \right\rceil.$$

This leads to this decomposition of $\mathbf{x}$:

$$\mathbf{x} = \overbrace{x_1, \ldots, x_d, x_{d+1}, \ldots, x_{2d}}^{\mathbf{x}_1}, \overbrace{x_{2d+1}, \ldots, x_{3d}, x_{3d+1}, \ldots, x_{4d}}^{\mathbf{x}_3}, x_{4d+1}, \ldots, x_{5d}, \ldots$$

$$\underbrace{x_{d+1}, \ldots, x_{2d}, x_{2d+1}, \ldots, x_{3d}}_{\mathbf{x}_2}, \underbrace{x_{3d+1}, \ldots, x_{4d}, x_{4d+1}, \ldots, x_{5d}}_{\mathbf{x}_4}$$

Our first contribution in this paper is to revisit this notion of fragmentation to show that it actually creates a strong relation between SEPM and HVE. More specifically, we show that once a string has been fragmented in this way, one can divert the use of any $2d$-HVE scheme to build an SEPM scheme. This observation allows to leverage the work that has already been done on Hidden Vector Encryption and even on Inner Product Encryption as we will explain in Section 4. In particular, it avoids the need to build a new system from scratch, as was done in [BCC20] and [BCS21]. Actually, one can show that the constructions of these works implicitly define a $2d$-HVE scheme.

Remark 1. Note that if d does not divide $\mathrm{len}(\mathbf{x})$, the last fragment is not completely defined. A generic solution is to complete it with padding but this could create a problem if the streaming resumes. However, in the latter case, one can simply retransmit this last fragment, completed with the new data. Alternatively, several HVE schemes, such as the ones implicitly used in [BCS21], allow to produce the encryption of an incomplete message that may later be completed consistently. Our point here is that incomplete fragments can easily be handled and so that we can, from now on, assume that d divides $\mathrm{len}(\mathbf{x})$.

3.2 Conversion

We first remark that fragmentation reduces the problem of encrypting a string of arbitrary length into the one of encrypting several substrings of fixed length. We can therefore run the encryption algorithm of any fixed-length primitive such as a $2d$-HVE scheme. However, this is true for any splitting of $\mathbf{x}$. The specificity of fragmentation is that it avoids the problem of patterns straddling fragments by ensuring that any searchable pattern will always be entirely contained in at least one fragment. Thanks to this feature, and by appropriately generating the secret keys of the underlying HVE system, one can ensure that any pattern will be detected.

We depict in Fig. 1 our generic conversion from a $2d$-HVE scheme to a n-SEPM with $n = d + 1$.

Correctness of the SEPM scheme of Fig. 1. Suppose that a pattern $\mathbf{k}$ matches $\mathbf{x}$ at some position ℓ, meaning that $\ell \in F_{\mathrm{SEPM}}(\mathbf{k}, \mathbf{x})$ where F_{SEPM} is defined in Def. 4. Let $i = \lceil \frac{\ell}{d} \rceil$. By construction, $\mathbf{k}$ is fully contained inside $\mathbf{x}_i = (x_{(i-1)d+1}, \ldots, x_{(i+1)d})$. We indeed have both:

1. $(i-1)d + 1 \leq \ell$
2. $\ell + \mathrm{len}(\mathbf{k}) - 1 \leq \ell + n - 1 = \ell + d \leq (i+1)d$

where the last inequality stands as $\ell \leq i \cdot d$, by definition of i. Therefore, within the fragment $\mathbf{x}_i$, $\mathbf{k}$ starts at some position $1 \leq j \leq d$ and will be detected by sk_j with probability 1 by correctness of $\mathcal{E}_{\mathrm{HVE}}$. We thus have that $\ell \in S$ with probability 1 where S is the output of $\mathtt{dec}_{\mathrm{SEPM}}(\mathsf{sk}, \mathbf{c})$.

Conversely suppose that $\ell \notin F_{\mathrm{SEPM}}(\mathbf{k}, \mathbf{x})$, but $\ell \in S$. This means that $\mathtt{dec}_{\mathrm{HVE}}(\mathsf{sk}_j, \mathbf{c}_i)$ has returned 1. By correctness of $\mathcal{E}_{\mathrm{HVE}}$, this can occur only with negligible probability as the pattern is not present in the fragment $\mathbf{x}_i$. Consequently, the probability of false positives of $\mathcal{E}_{\mathrm{SEPM}}$ is the same as the one of $\mathcal{E}_{\mathrm{HVE}}$. $\square$

- Let $n \in \mathbf{N}$, $n \geq 2$, $d = n - 1$.
- Let $\mathcal{E}_{\mathrm{HVE}} := (\mathtt{setup}_{\mathrm{HVE}}, \mathtt{keygen}_{\mathrm{HVE}}, \mathtt{enc}_{\mathrm{HVE}}, \mathtt{dec}_{\mathrm{HVE}})$ be a $2d$-HVE scheme.
- $\mathtt{setup}_{\mathrm{SEPM}}$ is $\mathtt{setup}_{\mathrm{HVE}}$.
- $\mathtt{keygen}_{\mathrm{SEPM}}(\mathsf{mk}, \mathbf{k})$ takes as input a pattern $\mathbf{k}$ with $\mathrm{len}(\mathbf{k}) \leq n$ and computes

$$\mathsf{sk}_j \leftarrow \mathtt{keygen}_{\mathrm{HVE}}(\mathsf{mk}, \mathbf{k}_j)$$

  for all $j \in [\![d]\!]$, where $\mathbf{k}_j$ is the $2d$ long pattern $(\overbrace{\star, \ldots, \star}^{j-1}, \mathbf{k}, \star, \ldots, \star)$.
  It then returns $\mathsf{sk} := (\mathsf{sk}_1, \ldots, \mathsf{sk}_d)$.
- $\mathtt{enc}_{\mathrm{SEPM}}(\mathsf{pk}, \mathbf{x})$ takes a string $\mathbf{x}$ of length $f \cdot d$ for some $f \in \mathbf{N}$ (see Remark 1) and computes

$$\mathbf{c}_i \leftarrow \mathtt{enc}_{\mathrm{HVE}}(\mathsf{pk}, (x_{(i-1)d+1}, \ldots, x_{(i+1)d}))$$

  for all $i \in [\![f]\!]$, to return $\mathbf{c} := (\mathbf{c}_1, \ldots, \mathbf{c}_f)$.
- $\mathtt{dec}_{\mathrm{SEPM}}(\mathsf{sk}, \mathbf{c})$ parses sk as $(\mathsf{sk}_1, \ldots, \mathsf{sk}_d)$ and $\mathbf{c}$ as $(\mathbf{c}_1, \ldots, \mathbf{c}_f)$ for some $f \in \mathbf{N}$. Then it sets $S = \varnothing$ and, for every $i \in [\![f]\!]$ and $j \in [\![d]\!]$, it tests whether $1 = \mathtt{dec}_{\mathrm{HVE}}(\mathsf{sk}_j, c_i)$, in which case it updates $S \leftarrow S \cup \{(i-1)d + j\}$.
  Finally, it outputs S.

Fig. 1. Generic construction of an n-SEPM scheme $\mathcal{E}_{\mathrm{SEPM}}$ from a $2d$-HVE scheme

Remark 2. Our conversion described in Fig.1 leads to secret keys sk whose size is *independent* of the length of $\mathbf{x}$. Technically, our conversion would also work for symmetric HVE schemes but in such a case we would have to change the encryption key, and therefore the secret keys, for each fragment, which would be rather cumbersome.

3.3 Security

Theorem 1. *The conversion in Fig.1 transforms an* $\mathsf{IND-CPA}$ *secure* $2(n-1)$-*HVE scheme* $\mathcal{E}_{\mathrm{HVE}}$ *into an* $\mathsf{IND-CPA}$ *secure* n-*SEPM* $\mathcal{E}_{\mathrm{SEPM}}$ *scheme.*

Proof. For the sake of clarity, we slightly adapt $\mathtt{Exp}_{\mathcal{A}}^{\mathsf{IND-CPA}}(\mathcal{E}_{\mathrm{SEPM}})$ for SEPM by defining $\mathtt{Exp}_{\mathcal{A}}^{\mathsf{IND-CPA-0}}(\mathcal{E}_{\mathrm{SEPM}})$ (resp. $\mathtt{Exp}_{\mathcal{A}}^{\mathsf{IND-CPA-1}}(\mathcal{E}_{\mathrm{SEPM}})$) as the original experiment where the challenger always choose $\beta = 0$ (resp. $\beta = 1$). We must then show that

$$\left|\Pr[\mathtt{Exp}_{\mathcal{A}}^{\mathsf{IND-CPA-0}} \to 0] - \Pr[\mathtt{Exp}_{\mathcal{A}}^{\mathsf{IND-CPA-1}} \to 0]\right| \text{ is negligible.}$$

In other words, the behaviour of $\mathcal{A}$ must be the same in both games.

Let $\mathbf{x}^0$ and $\mathbf{x}^1$ be the challenge messages submitted by the adversary $\mathcal{A}$ and $\mathbf{x}^0_1, \ldots, \mathbf{x}^0_f$ and $\mathbf{x}^1_1, \ldots, \mathbf{x}^1_f$ their respective fragments. In this proof we will proceed through a sequence of games where we will progressively replace $\mathbf{x}^0_i$ by $\mathbf{x}^1_i$ in the challenge ciphertext. Any discrepancy in the behaviour of $\mathcal{A}$ would then imply that it has been able to distinguish two HVE ciphertexts, and so an attack against the $\mathsf{IND}-\mathsf{CPA}$ security of the HVE scheme.

More formally, we define the following sequence of games:

- $\mathbf{game}_0$ is $\mathtt{Exp}^{\mathsf{IND-CPA}-0}_{\mathcal{A}}(\mathcal{E}_{\mathrm{SEPM}})$,
- for $i = 1, \ldots, f$, $\mathbf{game}_i$ is the same game as $\mathbf{game}_{i-1}$ except that,

$$c = \big(\mathsf{enc}_{\mathrm{HVE}}(\mathsf{pk}, \mathbf{x}^1_1), \ldots, \mathsf{enc}_{\mathrm{HVE}}(\mathsf{pk}, \mathbf{x}^1_i),\\ \mathsf{enc}_{\mathrm{HVE}}(\mathsf{pk}, \mathbf{x}^0_{i+1}), \ldots, \mathsf{enc}_{\mathrm{HVE}}(\mathsf{pk}, \mathbf{x}^0_f)\big),$$

so $\mathbf{game}_f$ is exactly $\mathtt{Exp}^{\mathsf{IND-CPA}-1}_{\mathcal{A}}(\mathcal{E}_{\mathrm{SEPM}})$.

For $i = 0, \ldots, f$, let Z_i be the event that the adversary outputs 0 in $\mathbf{game}_i$. We thus have,

$$\big|\Pr[\mathtt{Exp}^{\mathsf{IND-CPA}-0}_{\mathcal{A}}(\mathcal{E}_{\mathrm{SEPM}}) \to 0] - \Pr[\mathtt{Exp}^{\mathsf{IND-CPA}-1}_{\mathcal{A}}(\mathcal{E}_{\mathrm{SEPM}}) \to 0]\big|\\ \leq \sum_{i=1}^{f} \big|\Pr[Z_i] - \Pr[Z_{i-1}]\big|.$$

Let us assume that there exists $i^* \in [\![f]\!]$ such that $\big|\Pr[Z_{i^*}] - \Pr[Z_{i^*-1}]\big|$ is not negligible. We describe an adversary $\mathcal{B}$ that uses $\mathcal{A}$ against the $\mathsf{IND}-\mathsf{CPA}$ security of $\mathcal{E}_{\mathrm{HVE}}$. Let $\mathcal{C}(\mathcal{E}_{\mathrm{HVE}})$ be the challenger of $\mathtt{Exp}^{\mathsf{IND-CPA}}(\mathcal{E}_{\mathrm{HVE}})$.

Setup. $\mathcal{B}$ runs $\mathcal{C}(\mathcal{E}_{\mathrm{HVE}})$ to get the public parameters and keys of the system and forwards them to $\mathcal{A}$.

Query Phase 1. When $\mathcal{A}$ makes a query for a pattern k, $\mathcal{B}$ proceeds as in Fig. 1 and then queries d times $\mathcal{C}(\mathcal{E}_{\mathrm{HVE}})$ to build the associated secret key sk.

Challenge. The algorithm $\mathcal{B}$ uses the public key to set the ciphertext elements

$$\mathbf{c}_i \leftarrow \mathsf{enc}_{\mathrm{HVE}}(\mathsf{pk}, \mathbf{x}^1_i) \qquad \text{for } i = 1, \ldots, i^* - 1,\\ \text{and } \mathbf{c}_i \leftarrow \mathsf{enc}_{\mathrm{HVE}}(\mathsf{pk}, \mathbf{x}^0_i) \qquad \text{for } i = i^* + 1, \ldots, f.$$

It then submits $\mathbf{x}^0_{i^*}$ and $\mathbf{x}^1_{i^*}$ to $\mathcal{C}(\mathcal{E}_{\mathrm{HVE}})$ as challenge messages which returns a ciphertext element used by $\mathcal{B}$ as $\mathbf{c}_{i^*}$. The algorithm $\mathcal{B}$ can then send $(\mathbf{c}_1, \ldots, \mathbf{c}_f)$ to $\mathcal{A}$ as the challenge ciphertext.

Query phase 2. The algorithm $\mathcal{B}$ proceeds as in the first phase.

Guess. $\mathcal{B}$ finally forwards the bit β' issued by the adversary to $\mathcal{C}(\mathcal{E}_{\mathrm{HVE}})$.

First note that the restrictions placed on pattern queries in the SEPM experiment implies that all key queries to $\mathcal{C}(\mathcal{E}_{\mathrm{HVE}})$ are valid. In other words, if a pattern $\mathbf{k}$ matched $\mathbf{x}^0_{i^*}$ but not $\mathbf{x}^1_{i^*}$ in the HVE game, then the same would be true for $\mathbf{x}^0$ and $\mathbf{x}^1$ in the SEPM game, which is not possible.

Finally, the challenge ciphertext returned by $\mathcal{C}(\mathcal{E}_{\mathrm{HVE}})$ is either an encryption of $\mathbf{x}^0_{i^*}$ or $\mathbf{x}^1_{i^*}$. In the first case, we are playing $\mathbf{game}_{i^*-1}$. In the second case, this is exactly $\mathbf{game}_{i^*}$. Any adversary such that $\left|\Pr[Z_{i^*}] - \Pr[Z_{i^*-1}]\right|$ is non-negligible can then be used against the $\mathsf{IND}-\mathsf{CPA}$ experiment of HVE. □

Remark 3. This proof readily adapts to the case of selective security.

4 Hidden Vector Encryption from Inner Product Encryption

In the previous section, we have shown that any HVE scheme could be used to construct an SEPM scheme. However, we note that there is not many HVE schemes in the literature, in particular when one wants specific properties such as adaptive security. This stands in sharp contrast with a related primitive, Inner Product Encryption, for which countless constructions exist. Actually, the relatively low number of publications on HVE can perhaps be explained by a subsection of [KSW08] where the authors explain how one can generically build a n-HVE scheme from a $2n$-IPE scheme (referred as KSW conversion in the following). To our knowledge, this result has not been formally proven and [KSW08] seems to only consider selective security. In this section, we first show, as a warm-up that this conversion is secure, even in the adaptive case, although the proof is not that straightforward. In particular, we will see that subtleties appear in the proof, concerning the conversions of valid key queries made by an HVE adversary to valid key queries for an IPE adversary.

As this conversion doubles ciphertext size, we then revisit the links between these two primitives to show that one can achieve a much better ratio (almost 1) through a new conversion that we introduce. We then show that one can prove that this construction is secure in the selective case, using similar information theoretic arguments than in the proof of the KSW conversion. We defer the case of adaptive security to Section 5.

Notations. In the following conversions, we suppose $\Sigma = \mathbf{F}_p^\times$ and $\star = 0$ which implies $\Sigma \cup \{\star\} = \mathbf{F}_p$. For two vectors of same length $\mathbf{u} = (u_1, \ldots, u_n)$ and $\mathbf{v} = (v_1, \ldots, v_n)$, we denote by $\mathbf{uv}$ the vector of same length obtained by element-wise product

$$\mathbf{uv} := (u_1v_1, \ldots, u_nv_n)$$

For a vector $\mathbf{k} \in \mathbf{F}_p^n$, we denote by $\mathbb{1}_{\mathbf{k}}$ the vector $(s_1, \ldots, s_n)$ where for all $i \in [\![n]\!]$, $s_i = 1$ if $i \in \mathsf{supp}(\mathbf{k})$ and $s_i = 0$ if $i \notin \mathsf{supp}(\mathbf{k})$.

4.1 KSW Conversion

In [KSW08, Subsection 5.2], Katz, Sahai and Waters give the following conversion from a $2n$-IPE scheme to an n-HVE scheme. We define the applications

$$\begin{array}{rccc} f\colon & \Sigma^n \times \mathbf{F}_p^n & \longrightarrow & \mathbf{F}_p^{2n} \\ & (\ \mathbf{x}\ ,\ \mathbf{r}\ ) & \longmapsto & (\ \mathbf{xr}\ ,\ -\mathbf{r}\ ) \end{array}$$

$$\begin{array}{rccc} g\colon & \mathbf{F}_p^n & \longrightarrow & \mathbf{F}_p^{2n} \\ & \mathbf{k} & \longmapsto & (\ \mathbb{1}_{\mathbf{k}}\ ,\ \mathbf{k}\ ) \end{array}$$

where we have put the coordinates in a different order than [KSW08], simplifying our notations without fundamentally changing the original conversion. The construction of Katz, Sahai and Waters is depicted in Fig. 2.

- Let $\mathcal{E}_{\mathrm{IPE}} := (\mathtt{setup}_{\mathrm{IPE}}, \mathtt{keygen}_{\mathrm{IPE}}, \mathtt{enc}_{\mathrm{IPE}}, \mathtt{dec}_{\mathrm{IPE}})$ be a $(2n)$-IPE scheme.
- $\mathtt{setup}_{\mathrm{HVE}}(1^\lambda, n)$ is $\mathtt{setup}_{\mathrm{IPE}}(1^\lambda, 2n)$.
- $\mathtt{keygen}_{\mathrm{HVE}}(\mathsf{mk}, \mathbf{k})$ returns $\mathsf{sk}_{\mathbf{k}} \leftarrow \mathtt{keygen}_{\mathrm{IPE}}(\mathsf{mk}, g(\mathbf{k}))$.
- $\mathtt{enc}_{\mathrm{HVE}}(\mathsf{pk}, \mathbf{x})$ chooses $\mathbf{r} \overset{\$}{\leftarrow} \mathbf{F}_p^n$ and returns $\mathbf{c} \leftarrow \mathtt{enc}_{\mathrm{IPE}}(\mathsf{pk}, f(\mathbf{x}, \mathbf{r}))$.
- $\mathtt{dec}_{\mathrm{HVE}}(\mathsf{sk}_{\mathbf{k}}, \mathbf{c})$ is $\mathtt{dec}_{\mathrm{IPE}}(\mathsf{sk}_{\mathbf{k}}, \mathbf{c})$.

Fig. 2. KSW construction of an n-HVE scheme $\mathcal{E}_{\mathrm{HVE}}$ from a $2n$-IPE scheme

Correctness. Let us remark that $\langle f(\mathbf{x}, \mathbf{r}), g(\mathbf{k})\rangle = \langle(\mathbf{xr}, -\mathbf{r}), (\mathbb{1}_{\mathbf{k}}, \mathbf{k})\rangle = \langle \mathbf{xr}, \mathbb{1}_{\mathbf{k}}\rangle - \langle \mathbf{r}, \mathbf{k}\rangle = \langle \mathbb{1}_{\mathbf{k}}\mathbf{x}, \mathbf{r}\rangle - \langle \mathbf{k}, \mathbf{r}\rangle = \langle \mathbb{1}_{\mathbf{k}}\mathbf{x} - \mathbf{k}, \mathbf{r}\rangle$. Moreover, if a pattern $\mathbf{k}$ matches $\mathbf{x}$, then with the notations that we have just introduced, we have $\mathbb{1}_{\mathbf{k}}\mathbf{x} - \mathbf{k} = \mathbf{0}$.

As a result, if $\mathbf{k}$ matches $\mathbf{x}$, this inner product in 0 for all choices of r. By correctness of $\mathcal{E}_{\mathrm{IPE}}$, with probability 1 decryption of a ciphertext for $\mathbf{x}$ with $\mathsf{sk}_{\mathbf{k}}$ will return 1.

Conversely, if $\mathbf{k}$ does not match $\mathbf{x}$, then $\mathbb{1}_{\mathbf{k}}\mathbf{x} - \mathbf{k} \neq \mathbf{0}$. The probability for a random vector $\mathbf{r}$ in $\mathbf{F}_p^n$ of being orthogonal to a non-zero vector is $1/p$. So decryption will return 1 with negligible probability $1/p + (1 - 1/p) \cdot \mu(\lambda)$ where $\mu(\lambda)$ is the probability of false positive for $\mathcal{E}_{\mathrm{IPE}}$. This proves the correctness of the conversion. □

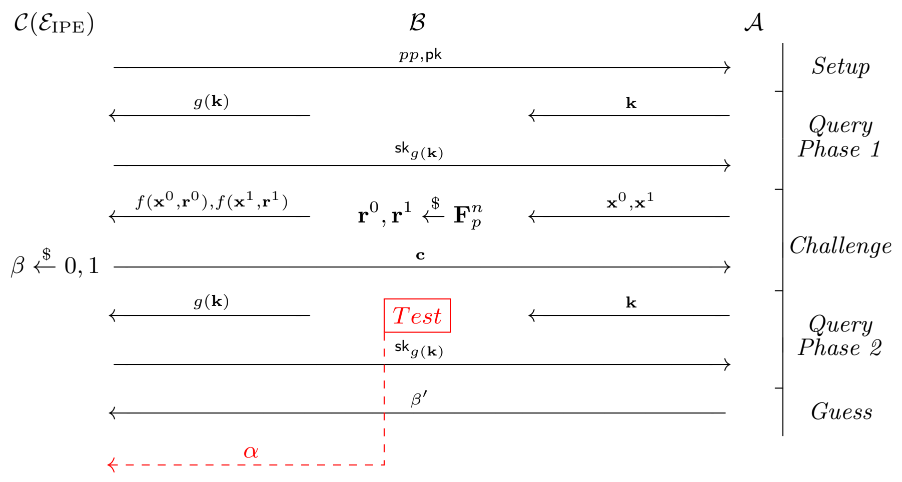
As we explain above, the authors of [KSW08] do not prove the security of this conversion but only provide a very informal argument to support this claim from correctness and seem to consider only selective security. Below, we show that this conversion indeed results in an adaptively (resp. selectively) secure HVE scheme if the IPE scheme is adaptively (resp. selectively) secure but also that this result is not that straightforward. Intuitively, the problem stems from the fact that there is some discrepancy between the restriction on key queries in an HVE experiment and the one in the IPE experiment. More concretely, an adversary in the HVE experiment may submit a key query for a vector $\mathbf{k}$ that does not match any of the challenge messages of the HVE experiment (this is thus a valid query) but that yet results, through this conversion, in an illicit query for the IPE experiment. The status of a query (illicit or not) will depend on the randomness $\mathbf{r}$ that is included in the challenge ciphertext. Before the challenge ciphertext is revealed, this randomness is still unknown and, as suggested in [KSW08], one can use the same argument that we saw when we proved correctness, for stating that false positives are unlikely. However, this argument does not hold in phase 2. Once the challenge ciphertext has been revealed, the randomness is no longer perfectly hidden. Fortunately, we can rely on another information theoretic argument as we explain in the proof.

4.2 Security Analysis of KSW Conversion

We prove in this section the following theorem.

Theorem 2. *The KSW conversion transforms a selective (resp. adaptive)* $\mathsf{IND-CPA}$ *secure* $2n$*-IPE scheme* $\mathcal{E}_{\mathrm{IPE}}$ *into a selective (resp. adaptive)* $\mathsf{IND-CPA}$ *secure* n*-HVE* $\mathcal{E}_{\mathrm{HVE}}$ *scheme.*

Proof. Consider an adversary $\mathcal{A}$ against the $\mathsf{IND-CPA}$ security of $\mathcal{E}_{\mathrm{HVE}}$. Let $\mathcal{C}(\mathcal{E}_{\mathrm{IPE}})$ be the challenger of the $\mathsf{IND-CPA}$ experiment for $\mathcal{E}_{\mathrm{IPE}}$. We build an adversary $\mathcal{B}$ against the $\mathsf{IND-CPA}$ security of $\mathcal{E}_{\mathrm{IPE}}$, using $\mathcal{A}$. An overview of the adversary $\mathcal{B}$ is given in Fig. 3.



$\mathcal{C}(\mathcal{E}_{\mathrm{IPE}})$
$\mathcal{B}$
$\mathcal{A}$
pp,pk
Setup
$g(\mathbf{k})$
$\mathbf{k}$
$\mathsf{sk}_{g(\mathbf{k})}$
Query Phase 1
$f(\mathbf{x}^0,\mathbf{r}^0), f(\mathbf{x}^1,\mathbf{r}^1)$
$\mathbf{r}^0,\mathbf{r}^1 \xleftarrow{\$} \mathbf{F}_p^n$
$\mathbf{x}^0,\mathbf{x}^1$
$\beta \xleftarrow{\$} 0,1$
$\mathbf{c}$
Challenge
$g(\mathbf{k})$
Test
$\mathbf{k}$
$\mathsf{sk}_{g(\mathbf{k})}$
Query Phase 2
β'
Guess
α

$\boxed{\textit{Test}}$ checks if conditions (2) and (3) are met in which case it returns the value α of condition (3) and stops the simulation.

Fig. 3. Overview of the adversary $\mathcal{B}$ in the proof of Theorem 2.

Setup. The adversary $\mathcal{B}$ simply forwards the public parameters and public key from $\mathcal{C}(\mathcal{E}_{\text{IPE}})$ to the adversary $\mathcal{A}$.

Query Phase 1. As there is no restriction on the possible queries in this phase in either security game, $\mathcal{B}$ simply answers a query $\mathbf{k}$ of $\mathcal{A}$ by submitting $g(\mathbf{k})$ as a query to $\mathcal{C}(\mathcal{E}_{\text{IPE}})$ and forwards the secret key to $\mathcal{A}$.

Challenge. In this phase, $\mathcal{A}$ submits two messages $\mathbf{x}^0$ and $\mathbf{x}^1$. We only have to handle the case where this pair satisfies the restriction of the HVE game. In this case, $\mathcal{B}$ chooses $\mathbf{r}^0, \mathbf{r}^1 \in \mathbf{F}_p^n$, submits the messages $f(\mathbf{x}^0, \mathbf{r}^0)$ and $f(\mathbf{x}^1, \mathbf{r}^1)$ and forwards the resulting challenge ciphertext to $\mathcal{A}$.

However a problem can occur if $f(\mathbf{x}^0, \mathbf{r}^0)$ and $f(\mathbf{x}^1, \mathbf{r}^1)$ do not satisfy the restriction of the IPE game. Let $\mathbf{k}$ be a pattern queried by $\mathcal{A}$. From the restriction of the HVE game, there are two cases.

First, $\mathbf{k}$ matches both x^0 and x^1. In this case, as we have seen for correctness, $\forall \alpha \in \{0,1\}$, $\langle f(\mathbf{x}^\alpha, \mathbf{r}^\alpha), g(\mathbf{k})\rangle = 0$, for all choices of randomness. As a result the messages $f(\mathbf{x}^0, \mathbf{r}^0)$ and $f(\mathbf{x}^1, \mathbf{r}^1)$ always satisfy the restriction of the IPE game.

The second case is a little more complex. The problematic conditions are

$$\forall \alpha \in \{0,1\}, \mathbf{k} \text{ does not match } \mathbf{x}^\alpha, \text{ and,}$$

$$\exists \alpha \in \{0,1\}, \langle f(\mathbf{x}^\alpha, \mathbf{r}^\alpha), g(\mathbf{k})\rangle = 0 \text{ and } \langle f(\mathbf{x}^{1-\alpha}, \mathbf{r}^{1-\alpha}), g(\mathbf{k})\rangle \neq 0.$$

As seen before, this can be rewritten as follows:

$$\forall \alpha \in \{0,1\}, \mathbb{1}_{\mathbf{k}}\mathbf{x}^\alpha - \mathbf{k} \neq \mathbf{0} \tag{2}$$

$$\exists \alpha \in \{0,1\}, \langle \mathbb{1}_{\mathbf{k}}\mathbf{x}^\alpha - \mathbf{k}, \mathbf{r}^\alpha\rangle = 0 \text{ and } \langle \mathbb{1}_{\mathbf{k}}\mathbf{x}^{1-\alpha} - \mathbf{k}, \mathbf{r}^{1-\alpha}\rangle \neq 0 \tag{3}$$

At this stage, we can still rely on the correctness argument as in [KSW08] because $\mathbf{r}^0$ and $\mathbf{r}^1$ were still unknown at the time of the queries. As a result, if equation (2) holds, equation (3) holds with negligible probability $2 \cdot \left(\frac{1}{p} \cdot \left(1 - \frac{1}{p}\right)\right) = \frac{2}{p} - \frac{2}{p^2} < \frac{2}{p}$.

Query Phase 2. Unlike in phase 1, we can no longer argue that $\mathbf{r}^0$ and $\mathbf{r}^1$ are unknown to the adversary and this is where we cannot rely only on the arguments developed for correctness as suggested in [KSW08]. Concretely, the HVE adversary $\mathcal{A}$ could get information on one of these random values from the challenge ciphertext and so submit a query $\mathbf{k}$ satisfying conditions (2) and (3). It might perhaps be possible to exclude such cases by assuming some appropriate computational assumption but this could only be done on a case-by-case basis and so would be irrelevant for this generic conversion.

Fortunately we can proceed differently: if the adversary $\mathcal{A}$ submits a query $\mathbf{k}$ that does not match either challenge message but such that $\langle \mathbb{1}_{\mathbf{k}}\mathbf{x}^\alpha - \mathbf{k}, \mathbf{r}^\alpha\rangle = 0$ for some $\alpha \in \{0,1\}$, and $\langle \mathbb{1}_{\mathbf{k}}\mathbf{x}^{1-\alpha} - \mathbf{k}, \mathbf{r}^{1-\alpha}\rangle \neq 0$, then $\mathcal{B}$ returns α to $\mathcal{C}(\mathcal{E}_{\text{IPE}})$ and stops. The intuition here is that the probability that $\mathcal{A}$ submits such a query with $\alpha = 1 - \beta$ is negligible because it has no information about $\mathbf{r}^{1-\beta}$. Formally, for any query $\mathbf{k}$ that does not match either challenge message, $\langle \mathbb{1}_{\mathbf{k}}\mathbf{x}^{1-\beta} - \mathbf{k}, \mathbf{r}^{1-\beta}\rangle = 0$ happens with probability $\frac{1}{p}$ and in the other case, $\mathcal{B}$ wins the security game.

Guess. Finally, $\mathcal{B}$ forwards the guess of $\mathcal{A}$ to $\mathcal{C}(\mathcal{E}_{\mathrm{IPE}})$. $\square$

4.3 Our conversion

A clear downside of the previous approach is that it requires a $2n$-IPE scheme to build a n-HVE scheme, which does not seem optimal. In this section, we propose a new generic transformation that halves this cost. We keep the same overall idea, of testing if a pattern $\mathbf{k}$ matches $\mathbf{x}$ by testing if the inner product $\langle \mathbb{1}_{\mathbf{k}}\mathbf{x} - \mathbf{k}, \mathbf{r}\rangle$ is 0 for a random $\mathbf{r}$, but we add this randomness during key generation instead of encryption which allows us to handle the wildcards of $\mathbf{k}$ without doubling the coordinates. However, as we will see, this change has profound consequences on the security proofs.

$$\begin{array}{rccc} f\colon & \Sigma^n & \longrightarrow & \mathbf{F}_p^{n+1} \\ & \mathbf{x} & \longmapsto & (\ \mathbf{x}\ ,\ -1\ ) \end{array}$$

$$\begin{array}{rccc} g\colon & \mathbf{F}_p^n \times \mathbf{F}_p^n & \longrightarrow & \mathbf{F}_p^{n+1} \\ & (\ \mathbf{k}\ ,\ \mathbf{r}\ ) & \longmapsto & (\ \mathbb{1}_{\mathbf{k}}\mathbf{r}\ ,\ \langle \mathbf{k}, \mathbf{r}\rangle\ ) \end{array}$$

We depict in Fig. 4 our generic conversion from an $(n+1)$-IPE scheme to a n-HVE scheme.

- Let $\mathcal{E}_{\mathrm{IPE}} := (\mathsf{setup}_{\mathrm{IPE}}, \mathsf{keygen}_{\mathrm{IPE}}, \mathsf{enc}_{\mathrm{IPE}}, \mathsf{dec}_{\mathrm{IPE}})$ be a $(n+1)$-IPE scheme.
- $\mathsf{setup}_{\mathrm{HVE}}(1^\lambda, n)$ is $\mathsf{setup}_{\mathrm{IPE}}(1^\lambda, n+1)$.
- $\mathsf{keygen}_{\mathrm{HVE}}(\mathsf{mk}, \mathbf{k})$ chooses $\mathbf{r} \xleftarrow{\$} \mathbf{F}_p^n$ and returns $\mathsf{sk}_{\mathbf{k}} \leftarrow \mathsf{keygen}_{\mathrm{IPE}}(\mathsf{mk}, g(\mathbf{k}, \mathbf{r}))$.
- $\mathsf{enc}_{\mathrm{HVE}}(\mathsf{pk}, \mathbf{x})$ returns $\mathbf{c} \leftarrow \mathsf{enc}_{\mathrm{IPE}}(\mathsf{pk}, f(\mathbf{x}))$.
- $\mathsf{dec}_{\mathrm{HVE}}(\mathsf{sk}_{\mathbf{k}}, \mathbf{c})$ is $\mathsf{dec}_{\mathrm{IPE}}(\mathsf{sk}_{\mathbf{k}}, \mathbf{c})$.

Fig. 4. Our construction of an n-HVE scheme $\mathcal{E}_{\mathrm{HVE}}$ from a $(n+1)$-IPE scheme

Correctness. Let $\mathbf{x} = (x_1, \ldots, x_n) \in \Sigma^n$ be a message and $\mathbf{c}$ be an encryption of $\mathbf{x}$. Let $\mathbf{k} = (k_1, \ldots, k_n) \in (\Sigma \cup \{\star\})^n$ be a pattern and $\mathsf{sk}_{\mathbf{k}}$ be the secret key of $\mathbf{k}$ generated as $\mathsf{keygen}_{\mathrm{IPE}}(\mathsf{mk}, g(\mathbf{k}, \mathbf{r}))$ for some vector $\mathbf{r} \in \mathbf{F}_p^n$. A calculation similar to the one we did before gives $\langle f(\mathbf{x}), g(\mathbf{k}, \mathbf{r})\rangle = \langle \mathbb{1}_{\mathbf{k}}\mathbf{x} - \mathbf{k}, \mathbf{r}\rangle$. This leads to one the following two cases:

- If $\mathbf{k}$ matches $\mathbf{x}$, then $\mathbb{1}_{\mathbf{k}}\mathbf{x} - \mathbf{k} = \mathbf{0}$, and $\langle f(\mathbf{x}), g(\mathbf{k}, \mathbf{r})\rangle = 0$ for all choices of r. By correctness of the IPE scheme, with probability 1, $\mathsf{dec}_{\mathrm{HVE}}(\mathsf{sk}_{\mathbf{k}}, \mathbf{c})$ will return 1.
- If $\mathbf{k}$ does not match $\mathbf{x}$, then $\mathbb{1}_{\mathbf{k}}\mathbf{x} - \mathbf{k} \neq \mathbf{0}$ and we only have $\langle \mathbb{1}_{\mathbf{k}}\mathbf{x} - \mathbf{k}, \mathbf{r}\rangle = 0$ with probability $1/p$ from the uniformity of $\mathbf{r}$. As the probability of a false positive for $\mathcal{E}_{\mathrm{IPE}}$ is some negligible function $\mu(\lambda)$, the probability of a false positive for the HVE scheme is less than $1/p + \mu(\lambda)$ which is negligible. $\square$

4.4 Selective Security

We now make a first assessment of the security of our new conversion. Unlike the KSW conversion, we need to distinguish the case of selective security from the case of adaptive security. Indeed, as we shall see, in the selective case, there is no problem of conversion of key queries from the HVE scheme to the IPE scheme: as the adversary chooses the challenge message at the beginning of the security game, the choice of the randomness will be always made *after* the choice at these messages and the choice of the query $\mathbf{k}$. As a result, the randomness is independent of the choices of the HVE adversary and we can still rely on an information theoretic argument. However for full adaptive security, this will no longer be the case. In the next section, we will show that we can rely on a computational argument related to the IPE scheme in order to go through the proof.

Theorem 3. *The conversion in Fig. 4 transforms a selective* $\mathsf{IND-CPA}$ *secure* $(n+1)$*-IPE scheme* $\mathcal{E}_{\mathrm{IPE}}$ *into a selective* $\mathsf{IND-CPA}$ *secure* n*-HVE scheme* $\mathcal{E}_{\mathrm{HVE}}$.

Remark 4. This theorem could actually be slightly extended in the sense that the result still holds if the adversary has access to the public key at the beginning of the security game but is not allowed to make key queries before committing to the challenge messages (*i.e.*, there is no Query Phase 1).

Proof. We denote by $\mathcal{C}_{\mathrm{sel}}(\mathcal{E}_{\mathrm{IPE}})$ the challenger of the selective $\mathsf{IND-CPA}$ security game for $\mathcal{E}_{\mathrm{IPE}}$. Again we build an adversary $\mathcal{B}$ that interacts with this challenger, using an adversary $\mathcal{A}$ against the selective security of $\mathcal{E}_{\mathrm{HVE}}$.

Setup. The adversary $\mathcal{B}$ receives the challenge messages $\mathbf{x}^0, \mathbf{x}^1 \in \Sigma^n$ from $\mathcal{A}$ and then forwards $f(\mathbf{x}^0), f(\mathbf{x}^1)$ to $\mathcal{C}_{\mathrm{sel}}(\mathcal{E}_{\mathrm{IPE}})$. Then $\mathcal{B}$ forwards the public key received from $\mathcal{C}_{\mathrm{sel}}(\mathcal{E}_{\mathrm{IPE}})$ to $\mathcal{A}$.

Query Phase 1. On key query $\mathbf{k} \in \mathbf{F}_p^n$, $\mathcal{B}$ chooses $\mathbf{r}$ at random in $\mathbf{F}_p^n$ and submits the key query $g(\mathbf{k}, \mathbf{r})$ to $\mathcal{C}_{\mathrm{sel}}(\mathcal{E}_{\mathrm{IPE}})$ and forwards the secret key to $\mathcal{A}$.

Challenge. The challenger $\mathcal{C}_{\mathrm{sel}}(\mathcal{E}_{\mathrm{IPE}})$ sends a ciphertext $\mathbf{c}$, encrypting either $f(\mathbf{x}^0)$ of $f(\mathbf{x}^1)$, which is forwarded to $\mathcal{A}$.

Query Phase 2. The adversary $\mathcal{B}$ proceeds as in the first query phase.

Guess. Finally, $\mathcal{B}$ forwards the guess of $\mathcal{A}$ to $\mathcal{C}_{\mathrm{sel}}(\mathcal{E}_{\mathrm{IPE}})$.

By construction, the guess of the adversary $\mathcal{A}$ can be used straightforwardly against the selective $\mathsf{IND-CPA}$ security of $\mathcal{E}_{\mathrm{IPE}}$ if $\mathcal{B}$ is a valid adversary. The only issue we need to consider here is the validity of the key queries. We actually face a situation rather similar to the one of the proof of Theorem 2. The problematic case is a query by $\mathcal{A}$ for a pattern $\mathbf{k}$ such that

$$\forall \alpha \in \{0,1\}, \mathbb{1}_{\mathbf{k}} \mathbf{x}^{\alpha} - \mathbf{k} \neq \mathbf{0}$$

$$\exists \alpha \in \{0,1\}, \langle \mathbb{1}_{\mathbf{k}} \mathbf{x}^{\alpha} - \mathbf{k}, \mathbf{r}^{\alpha} \rangle = 0 \text{ and } \langle \mathbb{1}_{\mathbf{k}} \mathbf{x}^{1-\alpha} - \mathbf{k}, \mathbf{r}^{1-\alpha} \rangle \neq 0.$$

This corresponds to a valid query for the HVE security experiment but not from the IPE security experiment. Fortunately, in the case of selective security, we can easily rule out this scenario. Indeed, in this case, the randomness $\mathbf{r}$ is selected by $\mathcal{B}$ *after* the choice of $\mathbf{x}^{\alpha}$ and $\mathbf{k}$ by $\mathcal{A}$. We can then rely on the same argument as in phase 1 of the proof of Theorem 2 to bound the probability of this event by $\frac{2}{p}$, which concludes the proof. $\square$

5 Adaptive Security

A natural question at this stage is how one can extend the previous result regarding our conversion to the case of adaptive security. Clearly, the approach of the selective security proof above cannot be generalised to this setting as we strongly relied on the fact that the adversary committed to the challenge messages *before* requesting any secret keys. Fortunately, we show in this section that we can prove the adaptive security of this generic conversion at the cost of imposing an additional requirement on the IPE scheme. We formalize this requirement as a property that we call *key privacy*. This notion is related to function privacy but implies weaker requirements on the secret key itself. This allows us to circumvent well-known limitations (see *e.g.* [BRS13]) of function-privacy for public key encryption, in particular the reliance on the entropy of the key space. As a consequence, one can decide once and for all if a given IPE scheme achieves this property, regardless of the distribution of the secret keys (and so of the context). We then consider some of the most popular constructions of IPE schemes and show that some of them achieve this property under very reasonable assumption (*e.g.* the Discrete Logarithm (DL) assumption) whereas some others do not. This highlights the fact that all IPE schemes are not equally suitable to construct HVE schemes.

5.1 Key privacy

Definition 5 (Key privacy for IPE). *An IPE scheme has key privacy if no probabilistic polynomial time adversary $\mathcal{A}$ has a non-negligible success in the following game, $\mathtt{Exp}_{\mathcal{A}}^{\mathsf{sk}-priv}$:*

Setup: *Run $(pp, \mathsf{pk}, \mathsf{mk}) \leftarrow \mathtt{setup}(1^{\lambda})$ and give pp, pk to $\mathcal{A}$.*
Query Phase 1: *$\mathcal{A}$ submits queries $\mathbf{k} \in \mathbf{F}_p^n$ and gets $sk_{\mathbf{k}} \leftarrow \mathtt{keygen}(\mathsf{mk}, \mathbf{k})$*
Challenge: *$\mathcal{A}$ submits a vector $\mathbf{y} \in \mathbf{F}_p^n$. The challenger chooses uniformly at random $\mathbf{u} \xleftarrow{\$} \{\mathbf{v} \in \mathrm{Vect}(\mathbf{y})^{\perp} : \mathsf{supp}(\mathbf{v}) \subset \mathsf{supp}(\mathbf{y})\}$ and gives $\mathsf{sk}_{\mathbf{u}} \leftarrow \mathtt{keygen}(\mathsf{mk}, \mathbf{u})$ to $\mathcal{A}$.*
Query Phase 2: *This phase is identical to Query Phase 1.*
Guess: *$\mathcal{A}$ eventually outputs a vector $\mathbf{z} \in \mathbf{F}_p^n \setminus \mathrm{Vect}(\mathbf{y})$ such that $\mathsf{supp}(\mathbf{z}) \subset \mathsf{supp}(\mathbf{y})$ and wins if $\langle \mathbf{z}, \mathbf{u} \rangle = 0$.*

Intuitively, this notion states the hardness of finding a non-trivial vector $\mathbf{z}$ that is orthogonal to a vector $\mathbf{u}$, given only $\mathsf{sk}_{\mathbf{u}}$. Obviously, we cannot reveal $\mathbf{u}$ to the adversary but we allow it to take part in the choice of $\mathbf{u}$ by submitting a vector $\mathbf{y}$ such that $\langle \mathbf{y}, \mathbf{u} \rangle = 0$. Very concretely, this models the fact that, in practice, the adversary may have some information about $\mathbf{u}$ since it knows (and may even choose) the pattern $\mathbf{y}$ that $\mathsf{sk}_{\mathbf{u}}$ allows to detect. However, for some schemes, this should essentially be the only information leaking about $\mathbf{u}$ from $\mathsf{sk}_{\mathbf{u}}$. As we do not want to reason in terms of entropy, we choose to define a computational goal (output $\mathbf{z} \in \mathbf{F}_p^n \setminus \mathrm{Vect}(\mathbf{y})$) which is much more convenient and yet sufficient to prove the adaptive security of our generic conversion. In particular, this allows us to evaluate this property for a given scheme independently of any application, as illustrated below.

5.2 Examples of key private IPE schemes

Before showing how this new security notion can be used to prove adaptive security of our generic conversion, we show in this subsection that it is naturally satisfied by some of the most popular IPE schemes, namely those from [KSW08] and [OT12a]. More generally, the technique we use below to prove this fact tends to show that IPE schemes where the components of the vector $\mathbf{u}$ associated with $\mathsf{sk}_{\mathbf{u}}$ appear as exponents in the latter key should satisfy this property. Intuitively, this stems from the fact that the vector $\mathbf{z}$ returned by the adversary provides a non-trivial relation between secret exponents, which can be used to solve a DL problem.

In the following, we use the same notations as in the original papers to facilitate verification of our claims without having to recall all the description of these schemes.

Katz-Sahai-Waters IPE Scheme. Our proof does not require the full knowledge of the IPE scheme of [KSW08] but will only use the fact that a secret key $\mathsf{sk}_{\mathbf{u}}$ associated with $\mathbf{u}$ contains two elements $K_{1,i} = g_p^{r_{1,i}} g_q^{f_1 u_i}$ and $K_{2,i} = g_p^{r_{2,i}} g_q^{f_2 u_i}$ where $r_{1,i}, r_{2,i}, f_1, f_2$ are random scalars. In our proof, the reduction will insert the DL challenge g_q^a in $g_q^{u_i}$, for $i \in \{i_1, i_2\}$, and generate all the other elements as usual. As $g_q^{u_i}$ is only involved in the elements $K_{1,i}$ and $K_{2,i}$ above we just have to explain how the reduction can proceed to construct them without knowing a. Actually, we only explain it for $K_{1,i}$ as $K_{2,i}$ has exactly the same structure.

More formally, our reduction is given a DL challenge $A = g_q^a$ in $\mathbb{G}_q$ and will interact with an adversary $\mathcal{A}$ against the key privacy of the IPE scheme to extract a.

In the security experiment, our reduction generates the secret key as usual and so is perfectly able to answer any query. The adversary will then eventually output a challenge $\mathbf{y}$ that will be managed as explained below.

But first we need to state some facts about $\mathbf{y}$. The goal of the adversary is to output $\mathbf{z}$ such that (1) $\mathsf{supp}(\mathbf{z}) \subset \mathsf{supp}(\mathbf{y})$ and (2) $\mathbf{y}$ and $\mathbf{z}$ are not colinear. There are therefore at least two indices $1 < i_1 < i_2 < n$ such that $y_{i_1} z_{i_2} \neq y_{i_2} z_{i_1}$, which implies that $\mathsf{supp}(y)$ contains at least one element. Actually, the latter set must contain at least two elements, otherwise $\mathbf{z}$ could not satisfy both (1) and (2). If $\mathbf{y}$ has only two non-zero components then a simple computation shows that the only possibility to meet (1) and (2) is when $\mathbf{u} = \mathbf{0}$, which does not occur with probability greater than $\frac{1}{q}$.

So, any adversary succeeding with non-negligible probability must output a vector $\mathbf{y}$ with at least three non-zero components. Let i_3 be the index of one of them, different from i_1 and i_2 defined above.

Upon receiving $\mathbf{y}$, our reduction chooses $s_{i_1}, s_{i_2} \in \mathbf{F}_q$ and implicitly sets

$$\begin{aligned} u_{i_1} =& s_{i_1} + y_{i_2} a \\ u_{i_2} =& s_{i_2} - y_{i_1} a. \end{aligned}$$

For all $i \in \llbracket n \rrbracket \setminus \{i_1, i_2, i_3\}$, the reduction explicitly sets $u_i \xleftarrow{\$} \mathbf{F}_q$ if $i \in \mathsf{supp}(\mathbf{y})$ and $u_i = 0$ otherwise. Finally it defines

$$u_{i_3} = -(y_{i_1} s_{i_1} + y_{i_2} s_{i_2} + \sum_{\substack{i=1 \\ i \neq i_1, i_2, i_3}}^{n} y_i u_i).$$

Thus, we have $\langle \mathbf{y}, \mathbf{u} \rangle = 0$ and the distribution of $\mathbf{u}$ generated this way is exactly the uniform distribution over $\{\mathbf{v} \in \mathrm{Vect}(\mathbf{y})^{\perp} : \mathsf{supp}(\mathbf{v}) \subset \mathsf{supp}(\mathbf{y})\}$.

The reduction can then compute K_{1,i_1} as

$$K_{1,i_1} = g_p^{r_{i_1,1}} (g_q^{s_{i_1}} A^{y_{i_2}})^{f_1}$$

and proceeds similarly for K_{1,i_2}. All the other elements $K_{1,i}$ can be computed directly as the reduction knows all the involved exponents.

At some stage, the adversary returns a guess $\mathbf{z}$. If the latter is valid, then we must have:

$$\sum_{i=1}^{n} (y_i - z_i) u_i = 0.$$

If we group the components in i_1 and i_2 together, we get

$$(y_{i_1} - z_{i_1}) u_{i_1} + (y_{i_2} - z_{i_2}) u_{i_2} = K$$

where K is a known scalar. As $y_{i_1} u_{i_1} + y_{i_2} u_{i_2} = s_{i_1} + s_{i_2}$, we can actually write the previous equation as

$$z_{i_1} u_{i_1} + z_{i_2} u_{i_2} = K'$$

where $K' = K - (s_{i_1} + s_{i_2})$ is still a known scalar. We thus have

$$z_{i_1}(s_{i_1} + y_{i_2} a) + z_{i_2}(s_{i_2} - y_{i_1} a) = K'.$$

which can be written as

$$z_{i_1} y_{i_2} a - z_{i_2} y_{i_1} a = K''$$

for a known K''. This gives

$$a(z_{i_1}y_{i_2} - z_{i_2}y_{i_1}) = K''.$$

As the factor $(z_{i_1}y_{i_2} - z_{i_2}y_{i_1})$ is assumed to be different from zero, we can recover a as $K''(z_{i_1}y_{i_2} - z_{i_2}y_{i_1})^{-1}$, which concludes the proof. $\square$

Okamoto-Takashima IPE Scheme. We show that the previous proof adapts very well to the construction of Section 4 of [OT12a]. Here, we use the additive notation of this paper. The setup of this scheme chooses a group $\mathbf{G}$ of order p and a generator $G \in \mathbf{G}$, so we use a DL challenge $A \in \mathbf{G}$ where implicitly $A = aG$ for some $a \in \mathbf{F}_p$ and the reduction sets the vector $\mathbf{u}$ as in the previous proof with respect to this challenge. This setup also generates a dual orthonormal basis among which the vectors $\mathbf{b}^*_{i_1}$ and $\mathbf{b}^*_{i_2}$ should be used to encode the positions i_1 and i_2 of the vector $\mathbf{u}$. These vectors are set as

$$\mathbf{b}^*_i = \sum_{j=1}^{4n+2} \vartheta_{i,j}(\overbrace{0,\ldots,0}^{j-1}, G, \overbrace{0,\ldots,0}^{4n+2-j}) \qquad \text{for } i = i_1, i_2$$

where $\vartheta_{i,j}$ are chosen by the reduction. The vector $\mathsf{sk}_{\mathbf{u}}$ generated by KeyGen is a sum of vectors that can all be computed regularly by the reduction except the two vectors $\sigma u_{i_1}\mathbf{b}^*_{i_1}$ and $\sigma u_{i_2}\mathbf{b}^*_{i_2}$ where σ is chosen regularly by the reduction. Instead, the reduction computes these two vectors as

$$\sigma[s_{i_1}\mathbf{b}^*_{i_1} + y_{i_2}\sum_{j=1}^{4n+2} \vartheta_{i_1,j}(\overbrace{0,\ldots,0}^{j-1}, A, \overbrace{0,\ldots,0}^{4n+2-j})]$$
$$\text{and } \sigma[s_{i_2}\mathbf{b}^*_{i_2} + y_{i_1}\sum_{j=1}^{4n+2} \vartheta_{i_2,j}(\overbrace{0,\ldots,0}^{j-1}, A, \overbrace{0,\ldots,0}^{4n+2-j})]$$

and sums them together with the other ones to generate the secret key. The rest of the proof is identical to the previous one. $\square$

5.3 Examples of non key private IPE schemes

Here we show that some techniques used to build IPE with constant secret keys size in [OT11, CGW18] do not allow key privacy, and in fact do not allow adaptive security using our conversion. In this case, one must use the KSW conversion. More specifically, this incompatibility stems from the fact that these constructions necessitate sharing the coordinates of the key vector inside its secret key *as scalars*.

We first show that such a scheme is not key private. In the key privacy game with $n \geq 3$, this allows for a winning strategy which consists in submitting $\mathbf{y} = (1,\ldots,1)$, learning $\mathbf{u}$ from $\mathsf{sk}_{\mathbf{u}}$ and solving the linear equation $u_1z_1,\ldots,u_nz_n =$

0 whose space of solutions has at least dimension $3 - 1 = 2$ and allows to successfully return $\mathbf{z}$. $\square$

We now show that using our conversion on such IPE schemes can actually not give adaptively secure HVE schemes. Indeed, an adversary against the adaptive security game of the resulting HVE has the following strategy. It first issues a query with key $\mathbf{k} = (1, \ldots, 1)$ and receives a secret key $\mathsf{sk}_{\mathbf{k}}$ which is also a secret key for the vector $g(\mathbf{k}, \mathbf{r}) = (\mathbf{r}, \langle \mathbf{k}, \mathbf{r} \rangle)$ in the underlying IPE scheme and contains the coordinates of this vector as scalars. With overwhelming probability, r_1 and r_2 are distinct from each other and from 0. Thus an adversary can build the n long vectors $\mathbf{x}^1 = (\frac{r_2}{r_1}, \frac{r_1}{r_2}, 1, \ldots, 1)$ and $\mathbf{x}^0 = (2, \ldots, 2)$. As $\mathbf{k}$ does not match either of these vectors, our adversary can submit them as challenge vectors and receives $\mathbf{c} \leftarrow \mathtt{enc}_{\mathrm{IPE}}(\mathsf{pk}, f(\mathbf{x}^\beta))$ for some unkown $\beta \leftarrow \{0, 1\}$. However we have $\langle f(\mathbf{x}^1), g(\mathbf{k}, \mathbf{r}) \rangle = 0 \neq \langle f(\mathbf{x}^0), g(\mathbf{k}, \mathbf{r}) \rangle$ with overwhelming probability and running $\mathtt{dec}_{\mathrm{HVE}}(\mathsf{sk}_{\mathbf{k}}, \mathbf{c})$ which is the same as $\mathtt{dec}_{\mathrm{IPE}}(\mathsf{sk}_{\mathbf{k}}, \mathbf{c})$ returns β. $\square$

5.4 Security Result

We now have all we need to state the adaptive security of a HVE scheme resulting from our conversion applied to an adaptive IPE scheme.

Theorem 4. *The conversion in Fig. 4 transforms an adaptive* $\mathsf{IND-CPA}$ *secure* $(n+1)$*-IPE scheme* $\mathcal{E}_{\mathrm{IPE}}$ *achieving key privacy into an adaptive* $\mathsf{IND-CPA}$ *secure* n*-HVE scheme* $\mathcal{E}_{\mathrm{HVE}}$.

Proof. We use the same reduction as in the proof of selective security of Theorem 3, using an adversary $\mathcal{A}$ against the HVE scheme to attack the security of the IPE scheme. The core issue is still the discrepancy between the key queries restrictions of these two primitives. Concretely, compared to the selective proof, a problem occurs in the reduction if $\mathcal{A}$ submits challenge messages $\mathbf{x}^0, \mathbf{x}^1 \in \Sigma$ such that there exists a query for a pattern $\mathbf{k}$ in Query Phase 1 satisfying the two following equations:

$$\forall \alpha \in \{0, 1\}, \mathbb{1}_{\mathbf{k}} \mathbf{x}^\alpha - \mathbf{k} \neq \mathbf{0} \tag{4}$$

$$\exists \alpha \in \{0, 1\}, \langle \mathbb{1}_{\mathbf{k}} \mathbf{x}^\alpha - \mathbf{k}, \mathbf{r}^\alpha \rangle = 0 \text{ and } \langle \mathbb{1}_{\mathbf{k}} \mathbf{x}^{1-\alpha} - \mathbf{k}, \mathbf{r}^{1-\alpha} \rangle \neq 0 \tag{5}$$

For Query Phase 2, as the challenge messages have been committed, the arguments of the selective proof hold again and the reduction can always use $\mathcal{C}(\mathcal{E}_{\mathrm{IPE}})$ to obtain the appropriate secret keys except with negligible probability.

Our strategy will then be to consider two types of adversary $\mathcal{A}$. Type 1 adversaries are those that *do not* output challenge messages satisfying conditions (4) and (5) for a queried $\mathbf{k}$ in phase 1. The selective proof of Theorem 3 readily adapts in this case. Conversely, Type 2 adversaries output challenge messages satisfying those conditions and we show in the following that they can be used to attack the key privacy notion of the IPE scheme contradicting the hypothesis that the IPE scheme has key privacy.

We denote by q a bound on the number of key queries that the adversary $\mathcal{A}$ may submit before the challenge phase. Let $\mathcal{C}_{\mathsf{sk}}(\mathcal{E}_{\mathrm{IPE}})$ be the challenger of the key privacy game of $\mathcal{E}_{\mathrm{IPE}}$. We now explicit a reduction that uses $\mathcal{A}$ to solve the key privacy experiment.

Setup. The reduction simply forwards the public parameters and public key from $\mathcal{C}_{\mathsf{sk}}(\mathcal{E}_{\mathrm{IPE}})$ to the Type 2 adversary $\mathcal{A}$ of the $\mathcal{E}_{\mathrm{HVE}}$ security game. The reduction chooses uniformly at random an integer $m \in [\![q]\!]$.

Query phases 1 and 2. The reduction handles any key query $\mathbf{k}$ other than the m^{th} one by choosing $\mathbf{r} \overset{\$}{\leftarrow} \mathbf{F}_p^n$ and submitting $g(\mathbf{k}, \mathbf{r})$ to $\mathcal{C}_{\mathsf{sk}}(\mathcal{E}_{\mathrm{IPE}})$ and forwarding the received secret key.

For the m^{th} key query $\mathbf{k} \in \mathbf{F}_p^n$, the reduction chooses the vector $\mathbf{y} = (\mathbf{k}, -1)$ as the challenge vector for $\mathcal{C}_{\mathsf{sk}}(\mathcal{E}_{\mathrm{IPE}})$ and forwards the received secret key sk for some implicit vector $\mathbf{u} \in \mathbf{F}_p^{n+1}$ to the adversary. We show that this secret key is well distributed as there exists a well distributed vector $\mathbf{r}^{(\mathbf{u})}$ such that $u = g(\mathbf{k}, \mathbf{r}^{(\mathbf{u})})$.

By definition, $\mathbf{u}$ is a vector from $\mathrm{Vect}(\mathbf{y})^{\perp}$ with $\mathsf{supp}(\mathbf{u}) \subset \mathsf{supp}(\mathbf{y})$. Let us consider a vector $\mathbf{r}^{(\mathbf{u})} \in \mathbb{F}_p^n$ such that $r_i^{(\mathbf{u})} = u_i$ for all $i \in \mathsf{supp}(\mathbf{k})$, and random values elsewhere. As $\langle \mathbf{y}, \mathbf{u} \rangle = 0$ and $y_{n+1} = -1$, this means that $u_{n+1} = \langle \mathbf{k}, \mathbf{r}^{(\mathbf{u})} \rangle$ and we have indeed $u = g(\mathbf{k}, \mathbf{r}^{(\mathbf{u})})$. Finally, the distribution of $\mathbf{u}$ as defined in the key privacy experiment implies that $\mathbf{r}^{(\mathbf{u})}$ (and so the associate secret key) is well distributed.

Challenge. When the adversary submits the challenge messages $\mathbf{x}^0, \mathbf{x}^1 \in \Sigma^n$, the reduction checks if the m^{th} key query satisfies (4) and (5) for an $\alpha \in \{0, 1\}$. If this is the case, the reduction returns $\mathbf{z} = (\mathbb{1}_{\mathbf{k}} \mathbf{x}^{\alpha}, -1)$ otherwise, it returns $\perp$.

For a Type 2 adversary, this vector $\mathbf{z}$ is indeed a valid answer in the key privacy security game because:

- $\mathsf{supp}(\mathbf{z}) \subset \mathsf{supp}(\mathbf{y})$ by construction.
- $z_{n+1} = y_{n+1} = -1$ but $\mathbf{z} \neq \mathbf{y}$ because of (4), which means that these two vectors are not colinear.
- $\langle \mathbf{z}, \mathbf{u} \rangle = \langle f(\mathbf{x}^{\alpha}), g(\mathbf{k}, \mathbf{r}^{\mathbf{u}}) \rangle = 0$ because of (5).

Therefore, any type 2 adversary against the adaptive $\mathsf{IND}-\mathsf{CPA}$ security of the HVE scheme can be converted into an adversary against the key privacy of the IPE scheme provided that the guess on m is valid, which occurs with probability at least $\frac{1}{q}$. $\square$

6 Consequences

In this section we draw the practical consequences of our generic conversions which allows to leverage the remarkable results obtained for Inner Product Encryption. The most significant results regarding complexity and security are presented in Fig. 5. Among other things, the latter shows that our conversions lead to the first SEPM schemes with adaptive security under standard assumptions, without significant performance loss compared to the underlying IPE scheme. In particular, starting from the IPE scheme of [CGW18, Subsection 3.4], we obtain, under the DLIN assumption, an adaptively secure SEPM scheme whose test complexity does not depend on the length of the fragment or the pattern.

As a first example illustrating our conversion, we choose to start from [DIP13] which is, to our knowledge, the only adaptively secure HVE scheme in the literature. At first glance, it seems to yield the SEPM scheme with the lowest number of elements in the ciphertext but we stress that one cannot directly compare this scheme with others as it is defined over bilinear groups of composite orders which are larger and lead to slow implementations compared to prime order bilinear groups. Conversion from one setting to another is possible (see *e.g.* [Fre10, Lew12]) but has a great impact on the number of group elements.

We showed in Subsection 5.2 that the IPE scheme from [OT12a] satisfies our key privacy property and can therefore be converted using either the KSW conversion (Fig. 2) or our shorter conversion (Fig. 4). We highlight the differences between the schemes resulting from these conversions in the next two columns of Fig. 5 and show that our conversion gives the adaptively secure SEPM scheme with the most succinct ciphertext in prime order groups. Moreover, it shows that our new IPE to HVE conversion decreases complexity by a factor up to 4 compared to the KSW conversion.

Our last example uses the IPE scheme from [CGW18]. This yields the adaptively secure SEPM scheme with the most compact public key. Moreover, decryption time in one position is constant.

For completeness, Fig. 5 also recalls the properties of the three most recent SEPM schemes. We stress that the comparison is not very meaningful as those schemes only achieved selective security under non-standard assumptions, something that we wanted to avoid with our conversions.

Notations. In Fig. 5, we assume fragments of size $2d$ and plaintexts containing elements from an alphabet Σ of size $|\Sigma|$. To retain legibility of the table, we only keep the terms of highest order in d and thus make some minor approximations in our complexity evaluation. Our first two rows indicate the generic transformations we apply. PK indicates the number of group elements in the public key to support fragments of size $2d$. CT is the number of group elements to encrypt an element from Σ. $\mathrm{SK}_{\mathbf{k}}$ is the number of group elements in the secret key allowing to search a pattern $\mathbf{k}$ at a given position. TEST refers to the number of pairings necessary to test the presence of a pattern $\mathbf{k}$ at a given position. We refer to the original papers for a definition of the computational assumptions underlying their security.

	Existing SEPM schemes			New SEPM schemes built by conversions			
	[BCC20, 3]	[BCS21, 4.3]	[BCS21, 4.4]	[DIP13]	[OT12a, 4.2]	[OT12a, 4.2]	[CGW18, 3.4]
IPE→HVE					Fig. 2	Fig. 4	Fig. 2
HVE→SEPM				Fig. 1	Fig. 1	Fig. 1	Fig. 1
PK	$2d \cdot \lvert\Sigma\rvert$	$4d$	$6d$	$2d \cdot \lvert\Sigma\rvert$	$64d^2$	$16d^2$	$40d$
CT	4	2	4	2	16	8	20
$\mathrm{SK}_{\mathbf{k}}$	2	2	3	$\mathrm{len}(\mathbf{k})$	$16d$	$8d$	8
TEST	2	2	3	$\mathrm{len}(\mathbf{k})$	$16d$	$8d$	8
Group Order	Prime	Prime	Prime	Composite	Prime	Prime	Prime
Security	Selective	Selective	Selective	**Adaptive**	**Adaptive**	**Adaptive**	**Adaptive**
Assumption	i-GDH	i-GDH	EXDH	CSD, CDDH	**DLIN**	**DLIN**	**DLIN**

Fig. 5. Comparison table of SEPM schemes

References

ABC$^+$08. Michel Abdalla, Mihir Bellare, Dario Catalano, Eike Kiltz, Tadayoshi Kohno, Tanja Lange, John Malone-Lee, Gregory Neven, Pascal Paillier, and Haixia Shi. Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions. *Journal of Cryptology*, 21(3):350–391, July 2008.

AMVY21. Shweta Agrawal, Monosij Maitra, Narasimha Sai Vempati, and Shota Yamada. Functional encryption for turing machines with dynamic bounded collusion from LWE. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 239–269, Virtual Event, August 2021. Springer, Heidelberg.

BCC20. Anis Bkakria, Nora Cuppens, and Frédéric Cuppens. Privacy-preserving pattern matching on encrypted data. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 191–220. Springer, Heidelberg, December 2020.

BCS21. Elie Bouscatié, Guilhem Castagnos, and Olivier Sanders. Public key encryption with flexible pattern matching. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part IV*, volume 13093 of *LNCS*, pages 342–370. Springer, Heidelberg, December 2021.

BDOP04. Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 506–522. Springer, Heidelberg, May 2004.

BRS13. Dan Boneh, Ananth Raghunathan, and Gil Segev. Function-private identity-based encryption: Hiding the function in functional encryption. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 461–478. Springer, Heidelberg, August 2013.

BSW11. Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 253–273. Springer, Heidelberg, March 2011.

BW07. Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In Salil P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 535–554. Springer, Heidelberg, February 2007.

CGW18. Jie Chen, Junqing Gong, and Hoeteck Wee. Improved inner-product encryption with adaptive security and full attribute-hiding. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part II*, volume 11273 of *LNCS*, pages 673–702. Springer, Heidelberg, December 2018.

CS15. Melissa Chase and Emily Shen. Substring-searchable symmetric encryption. *PoPETs*, 2015(2):263–281, April 2015.

DFOS18. Nicolas Desmoulins, Pierre-Alain Fouque, Cristina Onete, and Olivier Sanders. Pattern matching on encrypted streams. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part I*, volume 11272 of *LNCS*, pages 121–148. Springer, Heidelberg, December 2018.

DIP13. Angelo De Caro, Vincenzo Iovino, and Giuseppe Persiano. Fully secure hidden vector encryption. In Michel Abdalla and Tanja Lange, editors, *PAIRING 2012*, volume 7708 of *LNCS*, pages 102–121. Springer, Heidelberg, May 2013.

DPP18. Ioannis Demertzis, Dimitrios Papadopoulos, and Charalampos Papamanthou. Searchable encryption with optimal locality: Achieving sublogarithmic read efficiency. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 371–406. Springer, Heidelberg, August 2018.

Fre10. David Mandell Freeman. Converting pairing-based cryptosystems from composite-order groups to prime-order groups. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 44–61. Springer, Heidelberg, May / June 2010.

JLS21. Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, page 60–73, New York, NY, USA, 2021. Association for Computing Machinery.

KSW08. Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 146–162. Springer, Heidelberg, April 2008.

Lew12. Allison B. Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 318–335. Springer, Heidelberg, April 2012.

LL18. Iraklis Leontiadis and Ming Li. Storage efficient substring searchable symmetric encryption. In *Proceedings of the 6th International Workshop on Security in Cloud Computing*, SCC '18, page 3–13. Association for Computing Machinery, 2018.

OT10. Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 191–208. Springer, Heidelberg, August 2010.

OT11. Tatsuaki Okamoto and Katsuyuki Takashima. Achieving short ciphertexts or short secret-keys for adaptively secure general inner-product encryption. In Dongdai Lin, Gene Tsudik, and Xiaoyun Wang, editors, *CANS 11*, volume 7092 of *LNCS*, pages 138–159. Springer, Heidelberg, December 2011.

OT12a. Tatsuaki Okamoto and Katsuyuki Takashima. Adaptively attribute-hiding (hierarchical) inner product encryption. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 591–608. Springer, Heidelberg, April 2012.

OT12b. Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure unbounded inner-product and attribute-based encryption. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 349–366. Springer, Heidelberg, December 2012.

Ram16. Somindu C. Ramanna. More efficient constructions for inner-product encryption. In Mark Manulis, Ahmad-Reza Sadeghi, and Steve Schneider, editors, *ACNS 16*, volume 9696 of *LNCS*, pages 231–248. Springer, Heidelberg, June 2016.

SLPR15. Justine Sherry, Chang Lan, Raluca Ada Popa, and Sylvia Ratnasamy. Blindbox: Deep packet inspection over encrypted traffic. In Steve Uhlig, Olaf Maennel, Brad Karp, and Jitendra Padhye, editors, *SIGCOMM 2015*, pages 213–226, 2015.