Access Control Encryption for Equality, Comparison, and More

Georg Fuchsbauer[1], Romain Gay[1], Lucas Kowalczyk[2], and Claudio Orlandi[3]

[1] ENS, CNRS, Inria and PSL Research University, France, first.last@ens.fr
[2] Columbia University, USA, luke@cs.columbia.edu
[3] Aarhus University, Aarhus, Denmark, orlandi@cs.au.dk

Abstract. Access Control Encryption (ACE) is a novel paradigm for encryption which allows to control not only what users in the system are allowed to *read* but also what they are allowed to *write*.
The original work of Damgård et al. [DHO16] introducing this notion left several open questions, in particular whether it is possible to construct ACE schemes with polylogarithmic complexity (in the number of possible identities in the system) from standard cryptographic assumptions.
In this work we answer the question in the affirmative by giving (efficient) constructions of ACE for an interesting class of predicates which includes equality, comparison, interval membership, and more.
We instantiate our constructions based both on standard pairing assumptions (SXDH) or more efficiently in the generic group model.

1 Introduction

Access Control Encryption (ACE) is a novel paradigm for encryption that was introduced by Damgård, Haagh and Orlandi [DHO16]. (A similar concept had previously been introduced in [IPV10].) The main difference between ACE and other advanced encryption primitives (such as *identity-based* [Sha84, BF01, Sak00], *attribute-based* [SW05] or *functional encryption* [BSW11]) is that while previous concepts for encryption prevent parties from *receiving* messages (or functions of these) that are not meant for them, ACE also prevents unauthorized parties from *sending* messages to others they are not allowed to communicate with.

In a nutshell, ACE considers a set of senders $\{S_i\}_{i\in\{0,1\}^n}$ and a set of receivers $\{R_j\}_{j\in\{0,1\}^n}$. An ACE scheme is parameterized by a predicate P and $P(i,j)=1$ indicates that S_i is allowed to communicate with R_j while $P(i,j)=0$ means that no communication should be possible. All communication is assumed to be routed through a special party, called the *sanitizer*, which is assumed to be *semi-honest*; in particular, the sanitizer will follow the protocol specification but might try to learn additional information by colluding with other parties in the system.

During the key distribution phase each sender S_i is given an encryption key ek_i while each receiver is given a decryption key dk_j. A sender can then create a ciphertext $c=\mathsf{Enc}(ek_i,m)$ which is sent to the sanitizer. The sanitizer need not know (nor does he learn) the message which is being transmitted nor the

Table 1. Comparison of the construction in this work and in [DHO16], for predicates $P : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. The ciphertext size dominates the complexity in all three constructions, and is therefore used as a metric for comparison.

Construction	Predicate	Ciphertext Size	Assumption
[DHO16, Section 3]	any	$O(2^n)$	DDH or DCR
[DHO16, Section 4]	any	$\mathsf{poly}(n)$	iO
This work	$\mathsf{P}_{\mathsf{eq}}, \mathsf{P}_{\mathsf{comp}}, \ldots$	$O(n)$	SXDH

identity of the sender, but performs a simple sanitization of the ciphertext and broadcasts the output $c' = \mathsf{San}(pp, c)$ to all receivers. *Correctness* of the ACE scheme guarantees that if $P(i,j) = 1$ then $\mathsf{Dec}(dk_j, c) = m$ i.e., authorized receivers should be able to recover the message.

ACE also imposes two security requirements: the first, called the *no-read rule*, requires any set of unauthorized receivers (even colluding with the sanitizer) to be unable to learn any information from ciphertexts that they are not allowed to decrypt. The second (and more interesting) one is called the *no-write rule* and guarantees that no set of corrupt senders $\{S_i\}$ can transfer any information to any set of corrupt receivers $\{R_j\}$ under the condition that $P(i,j) = 0$ for each combination of sender-receiver pair.

In [DHO16] the authors present two ACE schemes which can implement any predicate $P : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. However, both constructions have severe limitations. The first construction can be instantiated under standard number-theoretic assumptions, such as the decisional Diffie-Hellman (DDH) assumption or the decisional composite residuosity (DCR) assumption underlying Paillier encryption. However, its complexity, e.g. in terms of key and ciphertext size, is exponential in n and can therefore only be used when the number of identities in a system is very small. The second construction, whose complexity is polynomial in n, relies on a special flavor of general-purpose functional encryption (defined in [DHO16]) that, to the best of our knowledge, can only be instantiated using indistinguishability obfuscation [GGH+13]; the scheme is therefore not practically useful at this time.

The authors of [DHO16] left as an open question whether it is possible to construct asymptotically efficient ACE schemes without obfuscation, even for limited classes of predicates. In this work we answer this question in the affirmative by showing asymptotically efficient constructions for interesting predicates such as equality, comparison, and interval membership, as summarized in Table 1 which are based on standard pairing assumptions (SXDH). (The construction can be instantiated even more efficiently in the generic group model, see Table 2 for the exact constants involved in the constructions).

Technical Overview of Our Contributions Our first technical contribution is an ACE scheme for the equality predicate i.e.,

$$\mathsf{P}_{\mathsf{eq}}(i,j) = 1 \Leftrightarrow i = j$$

The scheme can be instantiated using generic assumptions (see Section 3.2) and very efficiently using cryptographic pairings and in particular *structure-preserving signatures on equivalence classes* [HS14] (see Section 3.3). We show how to instantiate this construction based on standard pairing assumptions (SDXH) or more efficiently in the generic group model. See Table 2 for a detailed efficiency comparison.

We then show how to use the scheme for equality in a black-box way to implement ACE for a predicate defined in the following way. Let S and R be two efficient functions which map identities into sets of identities:

$$\mathsf{S}\colon \{0,1\}^n \to 2^{\{0,1\}^n} \text{ and } \mathsf{R}\colon \{0,1\}^n \to 2^{\{0,1\}^n}$$

under the constraint that $\max_{i,j}\{|\mathsf{S}(i)|, |\mathsf{R}(j)|\} = \mathsf{poly}(n)$. Then we can construct efficient ACE for the predicate defined by

$$\mathsf{P}_{\mathsf{disj}}(i,j) = 1 \Leftrightarrow \mathsf{S}(i) \cap \mathsf{R}(j) \neq \emptyset \ .$$

We show that this class of predicates is quite rich (using results from [SBC+07] and [GMW15]) and includes useful predicates such as comparison (i..e, the predicate $\mathsf{P}_{\mathsf{comp}}(i,j) = 1 \Leftrightarrow i \leq j$) and interval membership (i.e., the predicate $\mathsf{P}_{\mathsf{range}}$ defined for all points $z \in [N]$ and intervals $I \subset [N]$ as $\mathsf{P}_{\mathsf{range}}(z, I) = 1 \Leftrightarrow z \in I$).

In a nutshell, the composed ACE scheme works as follows: assuming an ACE for equality, sender i is given all the encryption keys corresponding to the identities contained in the set $\mathsf{S}(i)$ and receiver j is given all the decryption keys for identities contained in the set $\mathsf{R}(j)$. To encrypt a message, the sender encrypts it under all his encryption keys (padding to the size of the largest possible set). Now if the intersection of $\mathsf{S}(i)$ and $\mathsf{R}(j)$ is not empty, the receiver can decrypt at least one of the ciphertexts and therefore learn the message; the scheme thus satisfies *correctness*. Intuitively, the scheme also satisfies the *no-read* and *no-write rule* since $\mathsf{P}_{\mathsf{disj}}(i,j) = 0 \Rightarrow \mathsf{S}(i) \cap \mathsf{R}(j) = \emptyset$, which allows us to use the security property of the underlying equality ACE scheme.

For correctness, the receiver must be able to tell when decryption of the underlying ACE succeeds. This can be achieved using standard techniques, e.g., by using a sparse message space. The trivial implementation of decryption, where the receiver tries all keys on all ciphertexts, would lead to a decryption complexity *quadratic* in the size of $\mathsf{R}(j)$. In Section 4 we overcome this shortcoming by defining the overall predicate with *disjunction of equalities* instead of *disjointness of sets*.

We note that the *linear construction* from [DHO16] might at first glance look similar to the one proposed here, with $\mathsf{R}(j) = \{j\}$ (each receiver is given a single key) and $\mathsf{S}(i) = \{j \mid P(i,j) = 1\}$ (each sender is given a key for every receiver she is allowed to talk to). Note however that the complexity of this construction is *inherently* exponential, due to the way that ciphertexts are constructed and sanitized: in the *linear construction* of [DHO16], ciphertexts contain one entry for *every possible receiver in the system* (senders encrypt the message using the keys of all the receivers they are allowed to talk to and add random ciphertexts for the other receivers), and the sanitization process treats each component of

the ciphertext differently (i.e., the sanitizer sanitizes each component of the ciphertext using a receiver-dependent procedure). Our approach is to start with an ACE for equality with the property that the sanitizer's algorithm is oblivious of the identity of the sender/receiver.

Finally, we note that all constructions in [DHO16] require the sanitizer to store some *secret* information, the knowledge of which would allow the adversary to break the *no-write* rule. In contrast, for the schemes presented in this paper, the sanitizer does not need to store any secret information, thereby significantly reducing the chances for an adversary to break the security of the system. In particular, the adversary must perform an *active* corruption of the sanitizer in order to break the *no-write rule*.

2 Defining ACE

ACE Notation. An *access control encryption* (ACE)[4] scheme is defined by the following PPT algorithms:

Setup: Setup is a randomized algorithm that on input the security parameter κ and a policy $P\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ outputs a master secret key msk and public parameters pp (which include the message space $\mathcal{M}$ and ciphertext spaces $\mathcal{C}, \mathcal{C}'$).

Key Generation: Gen is a deterministic algorithm[5] that on input the master secret key msk, a type $t \in \{\mathsf{sen}, \mathsf{rec}\}$ and an identity $i \in \{0,1\}^n$, outputs a key k. We use the following notation for the two kinds of keys in the system:
- $ek_i \leftarrow \mathsf{Gen}(msk, i, \mathsf{sen})$ and call it an *encryption key for* $i \in \{0,1\}^n$
- $dk_j \leftarrow \mathsf{Gen}(msk, j, \mathsf{rec})$ and call it a *decryption key for* $j \in \{0,1\}^n$

We remark that, as opposed to [DHO16], there is no need for a private *sanitizer key* in our schemes.

Encrypt: Enc is a randomized algorithm that, on input an encryption key ek_i and a message m, outputs a ciphertext c.

Sanitizer: San is a randomized algorithm that using the public parameters pp transforms an incoming ciphertext $c \in \mathcal{C}$ into a sanitized ciphertext $c' \in \mathcal{C}'$.

Decryption: Dec is a deterministic algorithm that recovers a message $m' \in \mathcal{M} \cup \{\perp\}$ from a ciphertext $c' \in \mathcal{C}'$ using a decryption key dk_j.

Definition 1 (Correctness). *For all $m \in \mathcal{M}$, $i, j \in \{0,1\}^n$ with $P(i,j) = 1$:*

$$\Pr\left[\mathsf{Dec}\left(dk_j, \mathsf{San}\left(pp, \mathsf{Enc}\left(ek_i, m\right)\right)\right) \neq m\right] \leq \mathsf{negl}\left(\kappa\right)$$

with $(pp, msk) \leftarrow \mathsf{Setup}(1^\kappa, P)$, $ek_i \leftarrow \mathsf{Gen}(msk, i, \mathsf{sen})$, $dk_j \leftarrow \mathsf{Gen}(msk, j, \mathsf{rec})$, and the probability is taken over the random coins of all algorithms.

Complementary to correctness, we require that it is detectable when decryption does not succeed, formalized as follows.

[4] This section is taken almost verbatim from [DHO16].

[5] This is without loss of generality, since we can always add a PRF key to msk and derive the randomness for Gen from the PRF and the identity of the party.

Definition 2 (Detectability). *For all $m \in \mathcal{M}$, $i, j \in \{0,1\}^n$ with $P(i,j) = 0$:*

$$\Pr\left[\mathsf{Dec}\left(dk_j, \mathsf{San}\left(pp, \mathsf{Enc}\left(ek_i, m\right)\right)\right) \neq \perp\right] \leq \mathsf{negl}\left(\kappa\right)$$

with $(pp, msk) \leftarrow \mathsf{Setup}(1^\kappa, P)$, $ek_i \leftarrow \mathsf{Gen}(msk, i, \mathsf{sen})$, $dk_j \leftarrow \mathsf{Gen}(msk, j, \mathsf{rec})$, and the probability is taken over the random coins of all algorithms.

Definition 3 (No-Read Rule). *Consider the following game between a challenger C and a stateful adversary A:*

No-Read Rule	
Game Definition	**Oracle Definition**
1. $(pp, msk) \leftarrow \mathsf{Setup}(1^\kappa, P)$*;* *2.* $(m_0, m_1, i_0, i_1) \leftarrow A^{\mathcal{O}_G(\cdot), \mathcal{O}_E(\cdot)}(pp)$*;* *3.* $b \leftarrow \{0,1\}$*;* *4.* $c \leftarrow \mathsf{Enc}(\mathsf{Gen}(msk, i_b, \mathsf{sen}), m_b)$*;* *5.* $b' \leftarrow A^{\mathcal{O}_G(\cdot), \mathcal{O}_E(\cdot)}(c)$*;*	$\mathcal{O}_G(j,t)$*:* *1. Output* $k \leftarrow \mathsf{Gen}(msk, j, t)$*;* $\mathcal{O}_E(i,m)$*:* *1.* $ek_i \leftarrow \mathsf{Gen}(msk, i, \mathsf{sen})$*;* *2. Output* $c \leftarrow \mathsf{Enc}(ek_i, m)$*;*

We say that A wins the No-Read game *if $b = b'$, $|m_0| = |m_1|$, $i_0, i_1 \in \{0,1\}^n$ and for all queries q to $\mathcal{O}_G$ with $q = (j, \mathsf{rec})$ it holds that*

$$P(i_0, j) = P(i_1, j) = 0 \ .$$

We say an ACE scheme satisfies the No-Read rule *if for all PPT A*

$$\mathsf{adv}^A_{\mathsf{No\text{-}Read}}(ACE) = \Pr[A \textit{ wins the No-Read game}] - \tfrac{1}{2} \leq \mathsf{negl}(\kappa) \ .$$

Remark: The definition in [DHO16] requires $2 \cdot |\Pr[A \text{ wins the No-Read game}] - \frac{1}{2}| \leq \mathsf{negl}(\kappa)$, which is unachievable, since any A whose output satisfies $|m_0| \neq |m_1|$ has advantage $= 1$ (the same also applies to their version of Definition 4).

Our definition of the *no-read rule* is also weaker in that it does not guarantee anonymity of the sender against an adversary who can decrypt the ciphertext (in the context of attribute-based encryption a similar property is called *weak attribute hiding* [OT12]). However, none of the applications of ACE described in [DHO16] require this property.

Definition 4 (No-Write Rule). *Consider the following game between a challenger C and a stateful adversary A:*

No-Write Rule	
Game Definition	**Oracle Definition**
1. $(pp, msk) \leftarrow \mathsf{Setup}(1^\kappa, P)$*;* *2.* $m' \leftarrow \mathcal{M}$*;* $b \leftarrow \{0,1\}$*;* *3.* $(c_0, i') \leftarrow A^{\mathcal{O}_E(\cdot), \mathcal{O}_S(\cdot)}(pp)$*;* *4.* $c_1 \leftarrow \mathsf{Enc}(\mathsf{Gen}(msk, i', \mathsf{sen}), m')$*;* *5.* $b' \leftarrow A^{\mathcal{O}_E(\cdot), \mathcal{O}_R(\cdot)}(\mathsf{San}(pp, c_b))$*;*	$\mathcal{O}_S(j,t)$ *and* $\mathcal{O}_R(j,t)$*:* *1. Output* $k \leftarrow \mathsf{Gen}(msk, j, t)$*;* $\mathcal{O}_E(i,m)$*:* *1.* $ek_i \leftarrow \mathsf{Gen}(msk, i, \mathsf{sen})$*;* *2. Output* $c \leftarrow \mathsf{San}(pp, \mathsf{Enc}(ek_i, m))$*;*

Let Q_S (resp. Q) be the set of all queries $q = (j, t)$ that A issues to $\mathcal{O}_S$ (resp. both $\mathcal{O}_S$ and $\mathcal{O}_R$). Let I_S be the set of all $i \in \{0,1\}^n$ such that $(i, \mathsf{sen}) \in Q_S$ and let J be the set of all $j \in \{0,1\}^n$ such that $(j, \mathsf{rec}) \in Q$. Then we say that A wins the No-Write game *if $b' = b$ and all of the following hold:*

1. $i' \in I_S \cup \{0\}$;
2. $\forall i \in I_S, j \in J,\ P(i, j) = 0$;
3. $\mathsf{San}(pp, c_0) \neq \bot$.

We say an ACE scheme satisfies the No-Write rule *if for all PPT A*

$$\mathsf{adv}^{A}_{\mathsf{No\text{-}Write}}(ACE) = \Pr[A \textit{ wins the No-Write game}] - \tfrac{1}{2} \leq \mathsf{negl}(\kappa) \ .$$

Remark: Note that the *no-write rule* as defined in [DHO16] does not require the third condition above, which essentially just requires the ciphertext output by the adversary to be *well-formed* relative to the public parameters pp (which crucially means that the adversary already *knows* if the ciphertext is well-formed or not). The constructions in [DHO16] deal with this by letting the sanitizer output a random encryption when running on an malformed ciphertext instead. We find the notion presented here to be more natural.

3 ACE for Equality

Here, we show two how to build an ACE for the *equality predicate* defined by $\mathsf{P}_{\mathsf{eq}}\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ and

$$\mathsf{P}_{\mathsf{eq}}(x, y) = 1 \ \Leftrightarrow \ x = y \ .$$

We present two constructions, one based on generic assumptions and a second (more efficient) one based on cryptographic pairings.

3.1 Generic Construction Preliminaries

We start with reviewing the notation we will use for standard cryptographic building blocks and we refer to standard textbooks in cryptography (such as [Gol09, KL14]), for formal definitions of security. For real functions f and g, we write $f(\kappa) \approx g(\kappa)$ if $|f(\kappa) - g(\kappa)| \leq \mathsf{negl}(\kappa)$, where negl is a negligible function in κ.

Non-Interactive Zero-Knowledge Proofs. Let L be a language and R a relation s.t. $x \in L$ if and only if there exists a witness w such that $(x, w) \in R$. A non-interactive proof system [BFM88] for a relation R is defined by the PPT algorithms $(\mathsf{NIZK.Gen}, \mathsf{NIZK.Prove}, \mathsf{NIZK.Ver})$ with $crs \leftarrow \mathsf{NIZK.Gen}(1^\kappa, L)$, $\pi \leftarrow \mathsf{NIZK.Prove}(crs, x, w)$ and $\mathsf{NIZK.Ver}(crs, x, \pi) \in \{0,1\}$. We require *correctness*, *(perfect) soundness*, *knowledge extraction*, and *zero-knowledge*.

Correctness. For all PPT adversaries A:

$$\Pr\left[\begin{array}{l} crs \leftarrow \mathsf{NIZK.Gen}(1^\kappa, L); \\ (x,w) \leftarrow A(crs); \\ \pi \leftarrow \mathsf{NIZK.Prove}(crs, x, w) \end{array} : \mathsf{NIZK.Ver}(crs, x, \pi) = 1 \text{ if } (x,w) \in R\right] \approx 1 \ .$$

Soundness. For all PPT adversaries A:

$$\Pr\left[\begin{array}{l} crs \leftarrow \mathsf{NIZK.Gen}(1^\kappa, L); \\ (x,\pi) \leftarrow A(crs) \end{array} : \mathsf{NIZK.Ver}(crs, x, \pi) = 0 \text{ if } x \notin L\right] \approx 1 \ .$$

Knowledge Extraction. We say that a system $(\mathsf{NIZK.Gen}, \mathsf{NIZK.Prove}, \mathsf{NIZK.Ver})$ has knowledge-extraction security if there exists a knowledge extractor, which is a pair of PPT algorithms (E_1, E_2) with the following two properties:

1. For all PPT adversaries A:

$$\begin{aligned} \Pr[crs \leftarrow \mathsf{NIZK.Gen}(1^\kappa, L) \ : \ & A(crs) = 1] \\ & \approx \Pr[(crs, \tau) \leftarrow E_1(1^\kappa, L) \ : \ A(crs) = 1] \ . \end{aligned}$$

2. For all PPT adversaries A:

$$\Pr\left[\begin{array}{l} (crs, \tau) \leftarrow E_1(1^\kappa, L); \\ (x,\pi) \leftarrow A(crs); \\ w \leftarrow E_2(crs, \tau, x, \pi) \end{array} : \mathsf{NIZK.Ver}(crs, x, \pi) = 0 \text{ or } (x,w) \in R\right] \approx 1 \ .$$

Zero-Knowledge. We say that proof system $(\mathsf{NIZK.Gen}, \mathsf{NIZK.Prove}, \mathsf{NIZK.Ver})$ has zero-knowledge security if there exists a simulator, which is a pair of PPT algorithms (S_1, S_2) with the following property: For all PPT adversaries A:

$$\begin{aligned} \Pr[crs \leftarrow \mathsf{NIZK.Gen}(1^\kappa, L) \ : \ & A^{\mathsf{NIZK.Prove}(crs,\cdot,\cdot)}(crs) = 1] \\ & \approx \Pr[(crs, \tau) \leftarrow S_1(1^\kappa, L) \ : \ A^{S'(crs,\tau,\cdot,\cdot)}(crs) = 1] \ , \end{aligned}$$

where $S'(crs, \tau, x, w) = S_2(crs, \tau, x)$ if $(x,w) \in R$ and outputs *failure* otherwise.

We speak of perfect correctness, perfect soundness, perfect knowledge extraction, and perfect zero-knowledge if for sufficiently large security parameters, and for all adversaries (unbounded, and not just PPT), we have equalities in the respective definitions.

Digital Signatures. A signature scheme is a tuple of PPT algorithms $(\mathsf{Sig.Gen}, \mathsf{Sig.Sig}, \mathsf{Sig.Ver})$ with $(sk, vk) \leftarrow \mathsf{Sig.Gen}(1^\kappa)$, $\sigma = \mathsf{Sig.Sig}(sk, m)$, and $\mathsf{Sig.Ver}(vk, m, \sigma) \in \{0,1\}$. We require *correctness* and *existential unforgeability under chosen-message attacks*. (Note that we defined the signature algorithm to be deterministic. Any randomized signature scheme can be de-randomized using a pseudo-random tape generated with a PRF on the message).

Anonymous and Weakly Sanitizable Public-Key Encryption. We use a public-key encryption (PKE) scheme which must satisfy *semantic security*, *anonymity*, and which must be *weakly sanitizable*. The syntax is as follows: $pp \leftarrow \mathsf{PKE.Par}(1^\kappa)$ outputs public parameters, $(ek, dk) \leftarrow \mathsf{PKE.Gen}(pp)$ outputs an encryption/decryption key pair, $c \leftarrow \mathsf{PKE.Enc}(ek, m)$ outputs an encryption of m, $c' \leftarrow \mathsf{PKE.San}(pp, c)$ outputs a sanitized version of c and $m' \leftarrow \mathsf{PKE.Dec}(dk, c)$ decrypts ciphertext c.

Anonymity can be formalized as in [BBDP01] via a game where the adversary receives (pp, ek_0, ek_1), chooses a message m, receives $c \leftarrow \mathsf{PKE.Enc}(ek_b, m)$ and must guess b.

In [DHO16] the notion of *sanitizable encryption* is introduced as a relaxation of *rerandomizable encryption*. Here we only require an even weaker property: we define an encryption scheme to be *weakly sanitizable* if the adversary cannot win a game where he is given (pp, ek), chooses $(m_0, r_0), (m_1, r_1)$, receives

$$c' = \mathsf{PKE.San}(pp, \mathsf{PKE.Enc}(ek, m_b; r_b); r')$$

with uniform randomness r' and must guess b.

The weakening lies in the fact that sanitizations only have to be computationally indistinguishable, whereas in the sanitizable PKE of [DHO16], sanitizations of encryptions of the same message must be statistically indistinguishable.

An anonymous and weakly sanitizable scheme. An encryption scheme that satisfies the above properties under the DDH assumption is the following simple variation of ElGamal [Gam85]. As for the original scheme, the parameters $pp = (\mathbb{G}, p, g)$ consist of the description of a DDH-hard group $\mathbb{G}$ of order p generated by g; the decryption key is a random element $dk \in \mathbb{Z}_p$ and the encryption key is defined as $ek = g^{dk}$. Encryption of a message $m \in \mathbb{G}$ is now defined as picking random $r \in \mathbb{Z}_p^*$ and $s \in \mathbb{Z}_p$ and defining a ciphertext as

$$\mathsf{Enc}(ek, m; (r, s)) = (d_0, d_1, c_0, c_1) = (g^r, ek^r, g^s, ek^s \cdot m) \ .$$

A ciphertext (d_0, d_1, c_0, c_1) is sanitized by first checking if $d_0 = 1$ or $d_1 = 1$, in which case the sanitizer outputs two random group elements; otherwise it picks a random $t \in \mathbb{Z}_p^*$ and returns $(d_0^t \cdot c_0,\ d_1^t \cdot c_1) = (g^{rt+s},\ ek^{rt+s} \cdot m)$, which is (statistically close to) a fresh encryption of m.

This scheme can be made detectable (see Definition 2) using standard techniques, e.g. by choosing a sparse message space $\mathcal{M}'$, that is, with $\frac{|\mathcal{M}'|}{p} \leq \mathsf{negl}(\kappa)$, where p is the order of $\mathbb{G}$. Decryption of a sanitized ciphertext $(d_0^t c_0, d_1^t c_1) = (g^{rt+s}, ek^{rt+s} \cdot m)$ with a different key $dk' \neq dk$ yields: $(g^{rt+s})^{-dk'} \cdot ek^{rt+s} \cdot m = (g^{dk-dk'})^{rt+s} \cdot m$, which is statistically close to a random element of $\mathbb{G}$. If $\frac{|\mathcal{M}'|}{p} \leq \mathsf{negl}(\kappa)$, the probability that this element is in $\mathcal{M}'$ is negligible and so:

$$\Pr\left[\mathsf{Dec}\left(dk', \mathsf{San}\left(pp, \mathsf{Enc}\left(ek, m\right)\right)\right) \neq \perp\right] \leq \mathsf{negl}\left(\kappa\right) \ \ ,$$

meeting our definition for detectability.

Proposition 1. *The above encryption scheme is anonymous and weakly sanitizable.*

Proof. For anonymity, notice that we can define a hybrid anonymity game where the adversary is given an encryption of its message m under a new encryption key $e_x = g^x$ (where x is a random element of $\mathbb{Z}_p$) instead of ek_0 or ek_1 and move from the game that uses ek_0 to encrypt the challenge ciphertext to one that uses ek_x using DDH. Given a DDH challenge g, g^a, g^b, and g^{ab+x} where either $x = 0$ or x is a random element in $\mathbb{Z}_p$, one can play the game using $ek_0 = g^a$ and create the challenge ciphertext as $((g^b)^{\tilde{r}}, (g^{ab+x})^{\tilde{r}}, g^b, g^{ab+x} \cdot m)$. If $x = 0$, then this is distributed like the game that uses ek_0 (where $dk = a$, $r = b\tilde{r}$ and $s = b$). If x is a random element of $\mathbb{Z}_p$, then this is distributed like the game that uses ek_x. (Moving from from ek_x to ek_1 follows symmetrically).

To see that the variant is weakly sanitizable, notice that we can similarly define a hybrid sanitizability game where the adversary is given two random group elements as its challenge sanitized ciphertext. In such a game, the challenge sanitized ciphertext is independent of b, so the adversary cannot achieve any advantage. We can move to this game using DDH. Given a DDH challenge g, g^a, g^b, and g^{ab+x} where either $x = 0$ or x is a random element in $\mathbb{Z}_p$, one can play the game using $ek = g^a$ and create the challenge sanitized ciphertext as $(g^b, g^{ab+x} \cdot m_b)$ (unless r_b causes d_0 or d_1 to be the identity, in which case it uses two random group elements). If $x = 0$, then this is distributed like the normal game (a sanitized ciphertext $(d_0^t \cdot c_0,\ d_1^t \cdot c_1) = (g^{rt+s},\ ek^{rt+s} \cdot m_b)$ looks like an ElGamal encryption of m_b when $d_0 \neq 1$ and $d_1 \neq 1$). If x is a random element of $\mathbb{Z}_p$, then the challenge sanitized ciphertext is distributed as two random group elements. □

3.2 Generic Construction

Construction 1 (ACE for Equality – Generic)**.** *We construct an ACE scheme* $\mathsf{ACE} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ *defined by the following algorithms:*

Setup: Compute $pp^{\mathsf{pke}} \leftarrow \mathsf{PKE.Par}(1^\kappa)$ and $(vk, sk) \leftarrow \mathsf{Sig.Gen}(1^\kappa)$.
Let L be the language defined by the following NP relation: for $x = (vk, c)$ and $w = (pk, \sigma, m, r)$, define $R(x, w) = 1$ iff

$$\mathsf{Sig.Ver}(vk, pk, \sigma) = 1 \ \wedge \ c = \mathsf{PKE.Enc}(pk, m; r) \ .$$

Compute $crs \leftarrow \mathsf{NIZK.Gen}(1^\kappa, L)$. Pick a random PRF key K for a PRF F. Output $pp = (pp^{\mathsf{pke}}, vk, crs)$ and $msk = (sk, K)$.

Key Generation: Given the master secret key msk and an identity i, the encryption and decryption keys are computed as follows: run

$$(pk, dk) \leftarrow \mathsf{PKE.Gen}(pp^{\mathsf{pke}}; F_K(i)) \ \text{ and } \ \sigma = \mathsf{Sig.Sig}(sk, pk)$$

and define

$$ek_i = (pk, \sigma) \ \text{ and } \ dk_i = dk$$

Encryption: On input a message m and an encryption key $ek_i = (pk, \sigma)$ pick encryption randomness r, compute $c' = \mathsf{PKE.Enc}(pk, m; r)$, let $x = (vk, c')$, $w = (pk, \sigma, m, r)$ and compute $\pi \leftarrow \mathsf{NIZK.Prove}(crs, x, w)$. Output $c = (c', \pi)$.

Sanitizer: On input $pp = (pp^{\mathsf{pke}}, vk, crs)$ and a ciphertext $c = (c', \pi)$ the sanitizer outputs $\perp$ if $\mathsf{NIZK.Ver}(crs, x = (vk, c'), \pi) = 0$; otherwise it returns $c'' \leftarrow \mathsf{PKE.San}(pp, c')$.

Decryption: Given a ciphertext c' and a decryption key $dk_j = dk$ output

$$m' = \mathsf{PKE.Dec}(dk, c') \ .$$

Theorem 1. *Construction 1 satisfies the No-Read Rule if the underlying PKE scheme satisfies semantic security and anonymity, if the proof system is zero-knowledge and the PRF is pseudorandom.*

Proof. We assume that A makes queries $\mathcal{O}_G(i_0, \mathsf{sen})$ and $\mathcal{O}_G(i_1, \mathsf{sen})$ (this is w.l.o.g., as any A can be transformed into such an adversary without affecting its winning probability). We define a hybrid game which guesses A's oracle queries that lead to the creation of the encryption keys of users i_0 and i_1. If the guess was wrong, the game outputs a random bit. (The differences to the original game are items 0. and 6. below.) Let $q_{\max}$ be an upper bound on the number of $\mathcal{O}_G(\cdot, \mathsf{sen})$ plus the number $\mathcal{O}_E$ queries that A makes during the game. (The keys could also be first created during an encryption query.) Since A is PPT, it is clear that $q_{\max}$ is polynomial in κ.

Hybrid Game for No-Read Rule	
Game Definition	**Oracle Definition**
0. $q_0, q_1 \leftarrow \{1, \ldots, q_{\max}\}$; $\hat{q} \leftarrow 1$ 1. $(pp, msk) \leftarrow \mathsf{Setup}(1^\kappa, P)$; 2. $(m_0, m_1, i_0, i_1) \leftarrow A^{\mathcal{O}_G(\cdot), \mathcal{O}_E(\cdot)}(pp)$; 3. $b \leftarrow \{0, 1\}$; 4. $c \leftarrow \mathsf{Enc}(\mathsf{Gen}(msk, i_b, \mathsf{sen}), m_b)$; 5. $b' \leftarrow A^{\mathcal{O}_G(\cdot), \mathcal{O}_E(\cdot)}(c)$; 6. If $Q[q_0] = i_0$ and $Q[q_1] = i_1$ Return b'; Else return $b' \leftarrow \{0, 1\}$;	$\mathcal{O}_G(j, t)$: 0. If $t = \mathsf{sen}$ and $\mathsf{Gen}(msk, j, \mathsf{sen})$ has not been called yet, then $Q[\hat{q}] = j$; $\hat{q} = \hat{q} + 1$; 1. Output $ek_j \leftarrow \mathsf{Gen}(msk, j, t)$; $\mathcal{O}_E(i, m)$: 0. If $t = \mathsf{sen}$ and $\mathsf{Gen}(msk, i, \mathsf{sen})$ has not been called yet, then $Q[\hat{q}] = i$; $\hat{q} = \hat{q} + 1$; 1. $ek_i \leftarrow \mathsf{Gen}(msk, i, \mathsf{sen})$; 2. Output $c \leftarrow \mathsf{Enc}(ek_i, m)$;

Lemma 1. *An adversary that wins the no-read game with non-negligible advantage also wins the hybrid game with non-negligible advantage.*

Proof. Assume an adversary breaks the no-read rule, that is, there exists c s.t.

$$\mathsf{adv}^A_{\mathsf{No\text{-}Read}}(ACE) = \Pr[b' = b \text{ in the No-Read Game}] - \tfrac{1}{2} \geq \tfrac{1}{\kappa^c}$$

for infinitely many κ. Let E denote the event that in the hybrid game $Q[q_0] = i_0$ and $Q[q_1] = i_1$. Note that this event is independent of A's view; moreover,

conditioned on E occurring, the hybrid game and the original No-Read-Rule game are equivalent; finally $\Pr[E] = q_{\max}^{-2}$. We thus have

$$\begin{aligned}
&\mathsf{adv}^A_{\text{hybrid}}(ACE) = \Pr[b' = b \text{ in the hybrid game}] - \tfrac{1}{2} \\
&= \Pr[b' = b \text{ in hybrid} \mid E] \cdot \Pr[E] + \Pr[b' = b \text{ in hybrid} \mid \neg E] \cdot \Pr[\neg E] - \tfrac{1}{2} \\
&= \Pr[b' = b \text{ in No-Read} \mid E] \cdot \Pr[E] + \tfrac{1}{2} \cdot \Pr[\neg E] - \tfrac{1}{2} \\
&= \Pr[b' = b \text{ in No-Read}] \cdot \tfrac{1}{q_{\max}^2} + \tfrac{1}{2} \cdot (1 - \tfrac{1}{q_{\max}^2}) - \tfrac{1}{2} \\
&= \tfrac{1}{q_{\max}^2} \cdot \mathsf{adv}^A_{\text{hybrid}}(ACE) \geq \tfrac{1}{q_{\max}^2 \cdot \kappa^c}
\end{aligned}$$

for infinitely many κ. Thus, $\mathsf{adv}^A_{\text{hybrid}}(ACE)$ is not negligible in κ. □

Assuming an arbitrary PPT A, we will now show that H_0, the hybrid above with b fixed to 0, is computationally indistinguishable from H_1 (b fixed to 1). By Lemma 1, A cannot have won the original game, thus proving the theorem. We define a sequence of hybrid games between H_0 and H_1 and show that each one is computationally indistinguishable from the previous one (i.e., the probability that the hybrid game returns 1 only changes negligibly).

Game $H_{b,1}$ (for $b \in \{0,1\}$) is defined as H_b, except we use a truly random function instead of F to generate all secret keys.

$H_{b,0} \approx_c H_{b,1}$ (which we use as shorthand for $\Pr[A \text{ wins } H_{b,0}] \approx \Pr[A \text{ wins } H_{b,1}]$)**:** Indistinguishability follows from PRF security (as K is never revealed to A).

Game $H_{b,2}$ is the same as $H_{b,1}$, except *crs*, contained in *pp*, and π in the challenge ciphertext c are simulated.

$H_{b,1} \approx_c H_{b,2}$**:** Indistinguishability follows from the zero-knowledge property of the proof system.

Game $H_{0,3}$ is the same as $H_{0,2}$, except c is computed as encryption of m_1 (instead of m_0) under identity i_0's key.

$H_{0,2} \approx_c H_{0,3}$**:** Indistinguishability follows from semantic security of the encryption scheme: We construct a PPT reduction B that receives a challenge pk and simulates game $H_{0,2}$. When A makes the query that generates the q_0-th encryption key, B sets this key to pk. If A queries the corresponding decryption key, B aborts (outputting a random bit). When A outputs (m_0, m_1, i_0, i_1) and i_0 is not the identity corresponding to the q_0-th key, B aborts. Otherwise, B submits (m_0, m_1) as challenge to receive c from its challenger (which is either m_0 or m_1 encrypted under pk) and forwards c to A together with a simulated proof π. Reduction B perfectly simulates either $H_{0,2}$ or $H_{0,3}$, depending on its own challenge: if B guesses q_0 and q_1 correctly, it does not abort and otherwise it outputs a random bit anyway.

$H_{0,3} \approx_c H_{1,2}$**:** The two games differ in that m_1 is encrypted under i_0's key in $H_{0,3}$ and i_1's key in $H_{1,2}$. Indistinguishability follows from anonymity of the encryption scheme: We construct a PPT reduction B, which receives pk_0 and pk_1 and simulates $H_{1,2}$ for A, except that it sets the q_0th key to pk_0 and the q_1th key to pk_1. If A queries a corresponding decryption key or if in A's output (m_0, m_1, i_0, i_1), i_0 does not correspond to the q_0th key or i_1

does not correspond to the q_1th key then B aborts. Otherwise, B submits m_1 as a challenge to receive c from its challenger (which is m_1 encrypted under pk_0 or pk_1), which it forwards to A together with a simulated proof π. Depending on its own challenge, B perfectly simulates either $H_{1,2}$ or $H_{1,3}$: if B guesses q_0 and q_1 correctly, it does not abort and otherwise it outputs a random bit anyway.

We have thus shown $H_0 \approx_c H_{0,1} \approx_c H_{0,2} \approx_c H_{0,3} \approx_c H_{1,2} \approx_c H_{1,1} \approx_c H_1$, which concludes the proof. □

Theorem 2. *Construction 1 satisfies the No-Write Rule if the underlying PKE scheme is anonymous and weakly sanitizable, if the proof system is perfectly sound and has knowledge extraction security, the signature scheme is unforgeable and the PRF is pseudorandom.*

Proof. Let H_0 denote the No-Write-Rule game. W.l.o.g. we assume that A makes a query $\mathcal{O}_S(i', \mathsf{sen})$ and that $I_S \cap J = \emptyset$ (i.e., A satisfies the 2nd item in the winning condition in Definition 4). We start with defining two hybrid games whose indistinguishability from H_0 is immediate:

Game H_1 is defined as H_0, except we use a truly random function instead of F to generate all secret keys.

$H_0 \approx_c H_1$**:** Indistinguishability follows from PRF security.

Game H_2 is the same as H_1, except that crs is computed via the knowledge extractor: $(crs, \tau) \leftarrow E_1(1^\kappa, L)$ (where τ is the extraction trapdoor). When A outputs $c_0 = (c, \pi)$, we run the second part of the extractor: $w \leftarrow E_2(crs, \tau, x = (vk, c), \pi)$, where vk is contained in pp.

$H_1 \approx_c H_2$**:** Indistinguishability follows from the first property of knowledge extraction (i.e., a CRS output by E_1 is indistinguishable from one output by $\mathsf{NIZK.Gen}$) of the proof system. (Running E_2 has no effect on the outcome of the game.)

Hybrid Game H_2 for No-Write Rule

(Note that $\mathcal{O}_E$ need not compute the proof as it is discarded by San anyway.)

Game Definition	**Oracle Definition**
1. $pp^{\mathsf{pke}} \leftarrow \mathsf{PKE.Par}(1^\kappa)$; $(vk, sk) \leftarrow \mathsf{Sig.Gen}(1^\kappa)$; $(crs, \tau) \leftarrow E_1(1^\kappa, L)$; $pp = (pp^{\mathsf{pke}}, vk, crs)$ 2. $m^* \leftarrow \mathcal{M}$; $b \leftarrow \{0,1\}$; 3. $((c, \pi), i') \leftarrow A^{\mathcal{O}_E(\cdot), \mathcal{O}_S(\cdot)}((pp^{\mathsf{pke}}, vk, crs))$; $(pk, \sigma, m, r) \leftarrow E_2(crs, \tau, x = (vk, c), \pi)$; 4. $c_0 := (c, \pi)$; $c_1 \leftarrow \mathsf{Enc}(ek_{i'}, m^*)$; 5. $b' \leftarrow A^{\mathcal{O}_E(\cdot), \mathcal{O}_R(\cdot)}(\mathsf{San}(pp, c_b))$; 6. Return b';	$\mathcal{O}_S(j, t)$ and $\mathcal{O}_R(j, t)$: 1. If pk_j not yet defined, then $(pk_j, dk_j) \leftarrow \mathsf{PKE.Gen}(pp^{\mathsf{pke}})$; $\sigma_j = \mathsf{Sig.Sig}(sk, pk_j)$; If $t = \mathsf{rec}$ then return dk_j; Else return $ek_j = (pk_j, \sigma_j)$; $\mathcal{O}_E(i, m)$: 1. If pk_i not yet defined, then $(pk_j, dk_j) \leftarrow \mathsf{PKE.Gen}(pp^{\mathsf{pke}})$; 2. $c' \leftarrow \mathsf{PKE.Enc}(pk, m)$; Return $c'' \leftarrow \mathsf{PKE.San}(pp, c')$;

For a particular run of game H_2 (which is determined by the coins used by the adversary and the challenger when running the probabilistic algorithms), we now differentiate four types. We let $w = (pk, \sigma, m, r)$ denote the output of E_2.

Type 1: A outputs $c_0 = (c, \pi)$ with $\mathsf{NIZK.Ver}(crs, (vk, c), \pi) = 0$ (where crs, vk come from the public parameters used in the game).

Type 2: A outputs $c_0 = (c, \pi)$ with $\mathsf{NIZK.Ver}(crs, (vk, c), \pi) = 1$ but $R((vk, c), w) = 0$, i.e. $\mathsf{Sig.Ver}(vk, pk, \sigma) \neq 1$ or $c \neq \mathsf{PKE.Enc}(pk, m; r)$.

Type 3: A outputs $c_0 = (c, \pi)$ with $\mathsf{NIZK.Ver}(crs, (vk, c), \pi) = 1$, we have

$$\mathsf{Sig.Ver}(vk, pk, \sigma) = 1 \ \wedge \ c = \mathsf{PKE.Enc}(pk, m; r) \tag{1}$$

and pk was *not* issued in an oracle query by $\mathcal{O}_S$.

Type 4 is defined as Type 3 except pk *was* issued in an oracle query by $\mathcal{O}_S$.

The 4 types are a partitioning of the coin space of the experiment, which we denote by $T_1, \ldots, T_4$. Let W_2 denote the event that A wins hybrid game H_2.

Lemma 2. $\Pr[W_2 \wedge T_1] = 0$.

Proof. T_1 means A outputs $c_0 = (c, \pi)$ with $\mathsf{NIZK.Ver}(crs, (vk.c), \pi) = 0$. In this case, the San procedure aborts, and by definition A loses the game. □

Lemma 3. $\Pr[T_2] \approx 0$.

Proof. In case T_2 occurs A broke property 2 of knowledge-extraction security of the proof system: it output a valid proof π for statement $x = (vk, c)$ but the extractor E_2 failed to extract a witness w with $R(x, w) = 1$. □

Lemma 4. $\Pr[T_3] \approx 0$.

Proof. T_3 implies that A output (c, π) from which E_2 extracted $w = (pk, \sigma, m, r)$ with $\mathsf{Sig.Ver}(vk, pk, \sigma) = 1$ and pk was not issued in an oracle query.

If T_3 occurred with non-negligible probability then we could construct a PPT adversary B that achieves the same advantage in the signature forging game as follows: B simulates H_2 for A, creating a crs with an extraction trapdoor τ and using its signature oracle to respond to *send key* queries, i.e., queries of the form $(\cdot, \mathsf{sen})$ to $\mathcal{O}_S$. When A outputs $c_0 = (c, \pi)$, B runs $(pk, \sigma, m, r) \leftarrow E_2(crs, \tau, (vk, c), \pi)$ and returns (pk, σ). If T_3 occurred then B did not query pk to its signing oracle, meaning B output a valid forgery. Assuming our signature scheme is unforgeable, this (and thus T_3) can only occur with negligible probability. □

Lemma 5. $\left|\Pr[W_2 \,|\, T_4] - \frac{1}{2}\right| \approx 0$.

Proof. T_4 implies that A outputs $c_0 = (c, \pi)$ from which E_2 extracted $w = (pk, \sigma, m, r)$ with $c = \mathsf{PKE.Enc}(pk, m; r)$ and pk *was* issued in an oracle query by $\mathcal{O}_S$.

Similarly to the proof of Theorem 1, we first define a hybrid game which guesses A's oracle queries that lead to the creation of the encryption keys of users i' (from A's output (c_0, i')) and i (the identity corresponding to pk extracted by E_2). If the guess was wrong, the game outputs a random bit.

Let $q_{\max}$ be an upper bound on the number of $\mathcal{O}_S(\cdot, \mathsf{sen})$ plus the number of $\mathcal{O}_E$ queries that A makes during the game. Since A is PPT, $q_{\max}$ is polynomial in κ. (The differences to the original game are items 0., 6., and 7. below.)

Hybrid Game H_3 for No-Write Rule	
Game Definition	**Oracle Definition**
0. $q, q' \leftarrow \{1, \ldots, q_{\max}\}$; $\hat{q} \leftarrow 1$ 1. $pp^{\mathsf{pke}} \leftarrow \mathsf{PKE.Par}(1^\kappa)$; $(vk, sk) \leftarrow \mathsf{Sig.Gen}(1^\kappa)$; $(crs, \tau) \leftarrow E_2(1^\kappa, L)$; $pp = (pp^{\mathsf{pke}}, vk, crs)$ 2. $m^* \leftarrow \mathcal{M}$; $b \leftarrow \{0,1\}$; 3. $((c, \pi), i') \leftarrow A^{\mathcal{O}_E(\cdot), \mathcal{O}_S(\cdot)}((pp^{\mathsf{pke}}, vk, crs))$; $(pk, \sigma, m, r) \leftarrow E_2(crs, \tau, x = (vk, c), \pi)$; 4. $c_0 := (c, \pi)$; $c_1 \leftarrow \mathsf{Enc}(ek_{i'}, m^*)$; 5. $b' \leftarrow A^{\mathcal{O}_E(\cdot), \mathcal{O}_R(\cdot)}(\mathsf{San}(pp, c_b))$; 6. Let i be s.t. $pk = pk_i$ 7. If $Q[q] = i$ and $Q[q'] = i'$ Return b'; Else return $b' \leftarrow \{0,1\}$;	$\mathcal{O}_S(j,t)$ and $\mathcal{O}_R(j,t)$: 1. If pk_j not yet defined $(pk_j, dk_j) \leftarrow \mathsf{PKE.Gen}(pp^{\mathsf{pke}})$; $\sigma_j = \mathsf{Sig.Sig}(sk, pk_j)$; $Q[\hat{q}] = j$; $\hat{q} = \hat{q} + 1$; If $t = \mathsf{rec}$ then return dk_j; Return $ek_j = (pk_j, \sigma_j)$; $\mathcal{O}_E(i,m)$: 1. If pk_i not yet defined, then $(pk_j, dk_j) \leftarrow \mathsf{PKE.Gen}(pp^{\mathsf{pke}})$; $Q[\hat{q}] = i$; $\hat{q} = \hat{q} + 1$; 2. $c' \leftarrow \mathsf{PKE.Enc}(pk, m)$; Return $c'' \leftarrow \mathsf{PKE.San}(pp^{\mathsf{pke}}, c')$;

Following the argument from Lemma 1, we have that an adversary that wins H_2 with non-negligible advantage also wins H_3 (event which we denote by W_3) with non-negligible advantage. Thus,

$$\left|\Pr[W_3 \,|\, T_4] - \tfrac{1}{2}\right| \approx 0 \;\Rightarrow\; \left|\Pr[W_2 \,|\, T_4] - \tfrac{1}{2}\right| \approx 0\ . \tag{2}$$

Assuming an arbitrary PPT A we will now show that if T_4 occurs then $H_3^{(0)}$, the hybrid H_3 with b fixed to 0, is indistinguishable from $H_3^{(1)}$ (b fixed to 1). Thus $|\Pr[W_3 \,|\, T_4] - \frac{1}{2}| \approx 0$ and the lemma follows via (2).

To show indistinguishability of $H_3^{(0)}$ and $H_3^{(1)}$, we define an intermediate hybrid game H_4 and show that, conditioned on T_4, it is computationally indistinguishable from both $H_3^{(0)}$ and $H_3^{(1)}$ (i.e., the probability that the hybrid game returns 1 only changes negligibly).

In $H_3^{(0)}$, A is given the challenge ciphertext $c' \leftarrow \mathsf{San}(pp, c_0)$. If T_4 occurs then (cf. (1)) the ciphertext contained in $c_0 = (c, \pi)$ satisfies $c = \mathsf{PKE.Enc}(pk, m; r)$ (with pk, m and r extracted by E_2). Moreover, T_4 implies that π is valid and A thus receives $c' \leftarrow \mathsf{PKE.San}(pp, \mathsf{PKE.Enc}(pk, m; r))$.

Game H_4 is the same as $H_3^{(0)}$, except that we define the ciphertext given to A as $c' \leftarrow \mathsf{PKE.San}(pp, \mathsf{PKE.Enc}(pk, m^*, r^*))$ where m^*, r^* are random.

$\Pr[1 \leftarrow H_3^{(0)} \,|\, T_4] \approx \Pr[1 \leftarrow H_4 \,|\, T_4]$**:** Indistinguishability follows from sanitizing security of the encryption scheme: We construct a PPT reduction B that receives a challenge pk and simulates game $H_3^{(0)}$. When A makes the query that generates the q-th encryption key, B sets this key to pk. If A queries the corresponding decryption key, B aborts (outputting a random bit). Note that B will never abort if it guesses q and q' correctly, since a correct guess means that pk will be given out as a call to $\mathcal{O}(i, \mathsf{sen})$, and the security game then prohibits a request for the decryption key for i.
Upon receiving $(c_0 = (c, \pi), i')$ from A, B runs $(pk', \sigma, m_0, r_0) \leftarrow E_2(crs, \tau, (vk, c), \pi)$ (where T_4 implies that pk' was queried in an oracle call).

If $pk' \neq pk$ or $i' \neq Q[q']$ (B has not guessed q, q' correctly), then B aborts. Otherwise B submits (m_0, r_0, m^*, r^*) for random m^*, r^* and receives a sanitized ciphertext c', which it gives to A. The received c' is a sanitization of either A's output c_0 (for which we have $c_0 = \mathsf{PKE.Enc}(pk, m_0; r_0)$) or of $\mathsf{PKE.Enc}(pk, m^*; r^*)$ (always assuming B's guesses were correct). B can answer *decryption key* oracle queries for all allowed queries.

Reduction B perfectly simulates either $H_3^{(0)}$ or H_4, depending on its own challenge: if B guesses q and q' correctly, it does not abort and otherwise it outputs a random bit anyway.

$\Pr[1 \leftarrow H_4 \,|\, T_4] \approx \Pr[1 \leftarrow H_3^{(1)} \,|\, T_4]$**:** Letting $((c, \pi), i')$ denote A's output, the two games differ in that m^* is encrypted under pk in H_4 (where pk is such that $c = \mathsf{PKE.Enc}(pk, m; r)$) and under i''s key in $H_3^{(1)}$. Indistinguishability follows from anonymity of the encryption scheme: We construct a PPT B, which receives pk_0 and pk_1, and simulates $H_3^{(1)}$ for A, except that it sets the q-th and the q'-th created keys to pk_0 and pk_1, respectively. (If $q = q'$ then it sets both to pk_0.) If A queries a corresponding decryption key, B aborts. Upon receiving $(c_0 = (c, \pi), i')$ from A, B runs $(pk, \sigma, m, r) \leftarrow E_2(crs, \tau, (vk, c), \pi)$ and aborts if $pk \neq pk_0$ or if i' does not correspond to the q'th key (B's guess was wrong). If $q \neq q'$ then B submits a random m^* as a challenge to receive $\hat{c}$ from its challenger (which is m^* encrypted under pk_0 or pk_1); if $q = q'$ then B sets $\hat{c} = \mathsf{PKE.Enc}(pk_0, m^*, r^*)$. Next, B gives $c' \leftarrow \mathsf{PKE.San}(pp, \hat{c})$ to A.

Reduction B perfectly simulates either $H_3^{(1)}$ or H_4 (which are the same if $q = q'$), depending on its own challenge: if B guesses q and q' correctly, it does not abort and otherwise it outputs a random bit anyway. □

The theorem now follows from Lemmas 2–5. Letting W_0 denote the event that A wins the No-Write game H_0, we have

$$\begin{aligned}
&\mathsf{adv}^A_{\mathsf{No\text{-}Write}}(ACE) = \Pr[W_0] - \tfrac{1}{2} \;\approx\; \Pr[W_2] - \tfrac{1}{2} \\
&\leq \underbrace{\Pr[W_2 \wedge T_1]}_{\overset{\text{Lemma 2}}{=} 0} + \underbrace{\Pr[T_2]}_{\overset{\text{Lemma 3}}{\approx} 0} + \underbrace{\Pr[T_3]}_{\overset{\text{Lemma 4}}{\approx} 0} + \Big(\underbrace{\Pr[W_2 \,|\, T_4] - \tfrac{1}{2}}_{\overset{\text{Lemma 5}}{\approx} 0}\Big) \Pr[T_4] + \underbrace{\tfrac{1}{2} \Pr[T_4] - \tfrac{1}{2}}_{\leq 0} \\
&\leq \mathsf{negl}\ . \qquad \square
\end{aligned}$$

Here we show how to instantiate the generic construction, based on the SXDH assumption (Corollary 1), or based on the generic group model (Corollary 2). Both instantiation use structure-preserving signatures (SPS) [AFG+10], Groth-Sahai proofs [GS08] and the weakly sanitizable version of ElGamal encryption [Gam85] described in Section 3.3. In Corollary 1, we use the most efficient SPS scheme from SXDH, namely the one from [KPW15]. In Corollary 2, we use the most efficient SPS scheme with a security proof in the generic group model, which is [AGHO11]. The exact efficiency of the resulting ACE schemes are given in Table 2 on p. 23.

Corollary 1. *If the SXDH assumption holds, then by Theorems 1 and 2, Construction 1 instantiated with the signature scheme from [KPW15], Groth-Sahai proofs [GS08] and the weakly sanitizable version of ElGamal encryption [Gam85] from Section 3.3 satisfies the No-Read and No-Write rules.*

Corollary 2. *Theorems 1 and 2 imply that Construction 1 instantiated with the signature scheme from [AGHO11], Groth-Sahai proofs [GS08] and the weakly sanitizable version of ElGamal encryption [Gam85] satisfies the No-Read and No-Write rules in the generic group model.*

3.3 A More Efficient Construction From Pairings

Our next construction is based on ElGamal encryption, which is anonymous and re-randomizable; however, re-randomization of a ciphertext requires knowledge of its public key, so the sanitizer, who will randomize ciphertexts before passing them on, would be able to link ciphertexts to receivers.

Under a public key $pk = g^{sk}$, a message m is encrypted as $c_0 = g^r,\ c_1 = pk^r \cdot m$. In order to enable randomization without revealing the public key, the sender will *randomize* the public key as $d = (g^s,\ pk^s)$ for some random $s \neq 0$. Given c and d, the sanitizer now picks a random t and defines $c' := (c_0 \cdot d_0^t,\ c_1 \cdot d_1^t)$. Since $c' = (g^{r+st},\ pk^{r+st} \cdot m)$ is an ElGamal encryption of m under pk, the receiver, who knows the corresponding secret key, can decrypt. On the other hand, t randomizes the ciphertext, thus to someone computationally bounded and not knowing sk, the pair looks random. This ensures anonymity towards the sanitizer and thus the no-read rule.

However, the no-write rule can easily be violated: a sender could send ciphertexts under any key and since the key is hidden, this would even be hard to detect. To enforce sending ciphertexts under legitimate keys, in the previous construction keys were signed; but without again resorting to proofs, it seems hard to verify that the key underlying the randomized key d was signed.

Fortunately, structure-preserving signatures on equivalence classes (SPS-EQ) [HS14] achieve precisely what is needed here, so the sketched construction goes through without including any proofs in the ciphertext. This primitives allows signing of pairs (d_0, d_1) of group elements and adapting such signatures to multiples of the message. In particular, given a signature σ on (d_0, d_1), anyone can adapt the signature to (d_0^s, d_1^s) for any s. On the other hand, unforgeability guarantees that these are the only transformations one can do. The signatures are thus valid on all messages from the equivalence class

$$[(d_0, d_1)]_{\mathcal{R}} := \{(m_0, m_1) \,|\, \exists s : m_0 = d_0^s \wedge m_1 = d_1^s\} \ .$$

Adaptivity of SPS-EQ requires that signatures that were adapted to a multiple of the original message are indistinguishable from a fresh signature on the multiple.

Enforcement of the no-read rule follows in a straightforward fashion from DDH (the tuple $(g^r,\ pk^r \cdot m,\ g^s,\ pk^s)$ is indistinguishable from random under DDH and an instance can be embedded by using the adaptivity property of

SPS-EQ). Enforcement of the no-write rule is harder to prove and relies on unforgeability for SPS-EQ (which precludes the attack sketched above). The latter ensures that the values (d_0, d_1) sent by the adversary must be multiples of (g, pk_i) for some pk_i obtained from the key oracle.

The tricky part is that once the reduction embeds a DDH challenge, it cannot find out *which* public key was used, and so cannot simulate the game. We thus rely on the knowledge-of-exponent assumption which implies that for any adversary that is given (g, pk) and returns (g^s, pk^s) there exists an extractor that extracts s from the adversary. Now the reduction can guess which public key pk_i the adversary randomizes and *efficiently check* whether its guess was correct. If it is not the case, the reduction can abort and output a random bit. (If the reduction does not abort when its simulation is incorrect, we do not have any guarantees as to the adversary's behavior.)

Bilinear groups. A bilinear-group generator $\mathsf{BG.Gen}$ is a PPT algorithm that takes input a security parameter 1^κ and outputs a description BG of a bilinear group $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, \hat{g})$, where p is a prime of length κ; $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are groups of order p; g generates $\mathbb{G}_1$, $\hat{g}$ generates $\mathbb{G}_2$ and $e\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a bilinear map that is non-degenerate, i.e. $e(g, \hat{g})$ generates $\mathbb{G}_T$.

We say that the DDH assumption holds in $\mathbb{G}_1$ for $\mathsf{BG.Gen}$ if no PPT adversary A, given $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, \hat{g}) \leftarrow \mathsf{BG.Gen}(1^\kappa)$, and (S, T, U) with $s, t, u \leftarrow \mathbb{Z}_p^*$, $b \leftarrow \{0, 1\}$ and $S = g^s$, $T = g^t$, $U = g^{(1-b)u+bst}$, can decide b with non-negligible advantage. It holds in $\mathbb{G}_2$ if the same is true when g is replaced by $\hat{g}$. We say that SXDH holds for $\mathsf{BG.Gen}$ if DDH holds in both $\mathbb{G}_1$ and $\mathbb{G}_2$.

SPS-EQ. A *structure-preserving signature scheme on equivalence classes* [HS14, FHS15] consists of the following PPT algorithms:

$\mathsf{EQS.Gen}$, on input a bilinear group BG and a vector length $\ell > 1$ (in unary) outputs a key pair (sk, pk). $\mathsf{EQS.Sig}$ takes a secret key sk and a representative $M = (m_1, \ldots, m_\ell) \in (\mathbb{G}_1^*)^\ell$ of class $[M]_\mathcal{R}$ and outputs a signature σ for the equivalence class $[M]_\mathcal{R}$. $\mathsf{EQS.Adp}$, on input a representative $M \in (\mathbb{G}_1^*)^\ell$, a signature σ for M, a scalar μ and a public key pk, returns an updated signature σ' for the new representative $M' = M^\mu := (m_1^\mu, \ldots, m_\ell^\mu)$. $\mathsf{EQS.Ver}$ takes a representative $M \in (\mathbb{G}_1^*)^\ell$, a signature σ and a public key pk and outputs 1 if σ is valid for M under pk and 0 otherwise. $\mathsf{EQS.VfK}$ checks if a secret key sk corresponds to a public key pk and if so returns 1 and 0 otherwise.

The scheme should satisfy correctness, existential unforgeability under chosen-message attacks (EUF-CMA) and perfect signature adaptation. Let $M \in \mathbb{G}_1^*$, $\mu \in \mathbb{Z}_p^*$, and (sk, pk) be output by $\mathsf{EQS.Gen}$; σ by $\mathsf{EQS.Sig}(sk, M)$; and σ' by $\mathsf{EQS.Adp}(M, \sigma, \mu, pk)$. Then the scheme is *correct* if $\mathsf{EQS.VfK}(sk, pk) = 1$, $\mathsf{EQS.Ver}(M, \sigma) = 1$ and $\mathsf{EQS.Ver}(M^\mu, \sigma') = 1$.

Unforgeability is defined w.r.t. equivalence classes, i.e., a forgery must be on a message from an equivalence class for which the forger has not seen signatures.

Definition 5 (EUF-CMA). *Consider the following game for an adversary A:*

EUF-CMA Game for SPS-EQ	
Game Definition	**Oracle Definition**
1. $BG \leftarrow \mathsf{BG.Gen}(1^\kappa)$; 2. $(sk, pk) \leftarrow \mathsf{EQS.Gen}(BG, 1^\ell)$; 3. $(M^*, \sigma^*) \leftarrow A^{\mathcal{O}(\cdot)}(pk)$;	$\mathcal{O}(M)$: 1. Return $\mathsf{EQS.Sig}(sk, M)$;

Let Q be the set of all queries that A issues to $\mathcal{O}$. Then we say that A wins the EUF-CMA game *if the following hold:*

1. *For all $M \in Q$: $[M^*]_\mathcal{R} \neq [M]_\mathcal{R}$;*
2. $\mathsf{EQS.Ver}(M^*, \sigma^*, pk) = 1$

An SPS-EQ scheme is EUF-CMA *if for all $\ell > 1$ and all PPT algorithms A*

$$\mathsf{adv}^A_{\mathsf{EUF\text{-}CMA}}(\textit{SPS-EQ}) = \Pr[A \textit{ wins the EUF-CMA game}] \leq \mathsf{negl}(\kappa) \ .$$

The final property requires that signatures adapted by $\mathsf{EQS.Adp}$ are distributed like fresh signatures from $\mathsf{EQS.Sig}$.

Definition 6 (Signature Adaptation). *An SPS-EQ scheme* perfectly adapts signatures *if for all tuples $\ell > 1$, (sk, pk, M, σ, μ) with*

$$\mathsf{EQS.VfK}(sk, pk) = 1 \qquad \mathsf{EQS.Ver}(M, \sigma, pk) = 1 \qquad M \in (\mathbb{G}_1^*)^\ell \qquad \mu \in \mathbb{Z}_p^*$$

$\mathsf{EQS.Adp}(M, \sigma, \mu, pk)$ *and* $\mathsf{EQS.Sig}(sk, M^\mu)$ *are identically distributed.*

The most efficient construction of SPS-EQ is the following from [FHS14]. It has perfect signature adaptation and satisfies EUF-CMA in the generic group model (GGM).

SPS-EQ Construction from [FHS14]	
$\mathsf{EQS.Gen}(BG, 1^\ell)$: Choose $(x_i)_{i\in[\ell]} \leftarrow (\mathbb{Z}_p^*)^\ell$; $sk \leftarrow (x_i)_{i\in[\ell]}$; $pk \leftarrow (\hat{g}^{x_i})_{i\in[\ell]}$; Return (sk, pk); $\mathsf{EQS.Sig}((x_i)_{i\in[\ell]}, M)$: //$M \in (\mathbb{G}_1^*)^\ell$; Choose $y \leftarrow \mathbb{Z}_p^*$; Return $\sigma = \big(\prod m_i^{x_i y}, g^{y^{-1}}, \hat{g}^{y^{-1}}\big)$; $\mathsf{EQS.Adp}(pk, M, \sigma = (Z, Y, \hat{Y}), \mu)$: //$\mu \in \mathbb{Z}_p^*$ if $\mathsf{EQS.Ver}(pk, M, \sigma) = 0$, return $\perp$; Choose $\psi \leftarrow \mathbb{Z}_p^*$; Return $\sigma' = (Z^{\psi\mu}, Y^{\psi^{-1}}, \hat{Y}^{\psi^{-1}})$;	$\mathsf{EQS.Ver}(pk, M, \sigma = (Z, Y, \hat{Y}))$ Return 1 if all of the following hold: $Y \neq 1$; $\prod_{i\in[\ell]} e(M_i, \hat{X}_i) = e(Z, \hat{Y})$; $e(Y, \hat{g}) = e(g, \hat{Y})$; Else return 0; $\mathsf{EQS.Ver}(sk = (x_i), pk = (\hat{X}_i))$: If for all $i \in [\ell] : \hat{X}_i = \hat{g}^{x_i}$; then return 1; Else return 0;

KEA. The knowledge of exponent assumption [BP04] for a bilinear group generator $\mathsf{BG.Gen}$ states that for every PPT algorithm A, which given the output $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, \hat{g})$ of $\mathsf{BG.Gen}$ and a random $h \leftarrow \mathbb{G}_1$ as input outputs g^s, h^s

for some s, there exists a PPT extractor which, when given the coins of A as input, extracts s with non-negligible probability. Note that KEA trivially holds in the GGM, and since for our most efficient construction we already work in the GGM to use SPS-EQ, this is not an extra assumption.

Construction 2 (ACE for Equality – Pairing)**.** *We construct an ACE scheme* $\mathsf{ACE} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ *defined by the following algorithms:*

Setup: Given a bilinear group $BG = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g, \hat{g}, e)$, run $(sk, vk) \leftarrow \mathsf{EQS.Gen}(BG)$, pick a PRF key K and return $pp = (BG, vk)$ and $msk = (sk, K)$.

Key Generation: Define $dk_i = F_K(0||i)$ and $pk_i = g^{dk_i}$, and compute $\sigma_i = \mathsf{EQS.Sig}(sk, (g, pk_i); F_K(1||i))$; Return $ek_i = (pk_i, \sigma_i)$ and dk_i.

Encryption: On input a message m and an encryption key $ek_i = (pk_i, \sigma_i)$, pick randomness $r, s \leftarrow \mathbb{Z}_p^*$ and compute $\sigma' \leftarrow \mathsf{EQS.Adp}(vk, (g, pk_i), \sigma_i, s)$ and return

$$c_0 = g^r, \quad c_1 = pk_i^r \cdot m, \quad c_2 = g^s, \quad c_3 = pk_i^s, \quad \sigma' \ .$$

Sanitizer: If $\mathsf{EQS.Ver}(vk, (c_2, c_3), \sigma') = 0$ then output $\perp$. Else choose a random t and return

$$c_0' = c_0 \cdot c_2^t \ , \quad c_1' = c_1 \cdot c_3^t \ .$$

Decryption: Return $m = c_1' \cdot (c_0')^{-dk_j}$.

Correctness follows by inspection, and detectability of the ACE follows from the detectability of the underlying PKE we use, namely ElGamal. We will now show that the scheme also satisfies the no-read and the no-write rule.

Theorem 3. *Construction 2 satisfies the No-Read Rule if the PRF is pseudo-random, the SPS-EQ scheme has perfect adaptivity and the DDH assumption holds in* $\mathbb{G}_1$*.*

Proof. Plugging Construction 2 into the security game yields the game in Figure 1 (where we replaced PRF values by consistent random values). The proof is similar to that of Theorem 3 also proceeds by a series of hybrid games.

Game H**:** As the original game but at the beginning the challenger makes a random guess q from $\{1, \ldots, q_{\max}\}$ where $q_{\max}$ is a bound on the number of $\mathcal{O}_G(\cdot, \mathsf{sen})$ queries plus the number of $\mathcal{O}_E(\cdot, \cdot)$ queries. Let $(j^*, \cdot)$ be the qth such query. If $j^* \neq i_b$, the challenger returns a random bit as the output of the game.

No-Write Game $\rightarrow H$**:** This results in a polynomial loss $\frac{1}{q_{\max}}$ in the adversary's winning probability, shown analogously to Lemma 1. If the latter was non-negligible before, it is so afterwards.

Game $H_{b,1}$**:** As hybrid H with b fixed and the values of the PRF replaced with (consistent) random values.

$H_b \approx_c H_{b,1}$**:** The games are indistinguishable by PRF security.

Game Definition	**Oracle Definition**
// assume w.l.o.g. A queries (i_0, sen) and (i_1, sen) to $\mathcal{O}_G$ 1. $(sk, vk) \leftarrow \mathsf{EQS.Gen}(BG)$; $pp = (BG, vk)$; 2. $(m_0, m_1, i_0, i_1) \leftarrow A^{\mathcal{O}_G(\cdot), \mathcal{O}_E(\cdot)}(pp)$; 4. $r, s \leftarrow \mathbb{Z}_p^*$; $\sigma' \leftarrow \mathsf{EQS.Adp}(vk, \sigma_{i_b}, s)$; $c = (g^r, pk_{i_b}^r m_b, g^s, pk_{i_b}^s, \sigma')$; 5. $b' \leftarrow A^{\mathcal{O}_G(\cdot), \mathcal{O}_E(\cdot)}(c)$;	$\mathcal{O}_G(j, t)$: // $t = \mathsf{rec} \Rightarrow j \notin \{i_0, i_1\}$ 1. If pk_j not defined: $dk_j \leftarrow \mathbb{Z}_p$; $pk_j = g^{dk_j}$; If $t = \mathsf{rec}$ then return dk_j; $\sigma_j \leftarrow \mathsf{EQS.Sig}(sk, (g, pk_j))$; Return $ek_j = (g, pk_j, \sigma_j)$; $\mathcal{O}_E(i, m)$: 1. If pk_i not yet defined: $dk_i \leftarrow \mathbb{Z}_p$; $pk_i = g^{dk_i}$; $\sigma_i \leftarrow \mathsf{EQS.Sig}(sk, (g, pk_i))$; 2. $r', s' \leftarrow \mathbb{Z}_p^*$; $\sigma' \leftarrow \mathsf{EQS.Adp}(vk, \sigma_i, s')$; Return $(g^{r'}, pk_i^{r'} \cdot m, g^{s'}, pk_i^{s'}, \sigma')$;

Fig. 1. No-Read Rule for Constr. 2 for fixed b and PRF outputs replaced by random

Game $H_{b,2}$: As $H_{b,1}$, but instead of running $\mathsf{EQS.Adp}$, σ' is computed as a fresh signature on $(g^s, pk_{i_b}^s)$.

$H_{b,1} \approx_c H_{b,2}$**:** The two games are equally distributed by the *perfect signature-adaptation* property of SPS-EQ.

Game $H_{b,3}$: Defined as $H_{b,2}$, except c is replaced by $c = (g^r, pk_{i_b}^r m_b, g^s, pk_{i_b}^t, \sigma')$, that is, the 4th component is random.

$H_{b,2} \approx_c H_{b,3}$**:** Indistinguishable under DDH. Note that pk_{i_b} is known in advance (as i_b is guessed as j^*) and that dk_i is not revealed and s is not used anywhere else (since $H_{b,2}$). The reduction can thus replace the values $(g, pk_{i_b}, g^s, pk_{i_b}^s)$ with a DDH challenge.

Game $H_{b,4}$: Defined as $H_{b,3}$, except c is replaced by $c = (g^r, pk_{i_b}^u m_b, g^s, pk_{i_b}^t, \sigma')$, that is, the 2nd component is random.

$H_{b,3} \approx_c H_{b,4}$**:** Indistinguishable under DDH. The reduction replaces the values $(g, pk_{i_b}, g^r, pk_{i_b}^r)$ with a DDH challenge.

Since $H_{0,4} \equiv H_{1,4}$ (in both the adversary receives c which consists of 4 random group elements and a signature on the last 2), we showed that H_0 and H_1 are indistinguishable, which contradicts the assumption that A distinguishes them. □

Theorem 4. *Construction 2 satisfies the No-Write Rule if the PRF is pseudo-random, the SPS-EQ scheme is unforgeable, and KEA and DDH hold in* $\mathbb{G}_1$.

Proof. As it is straightforward to prove indistinguishability to the original game, let us immediately assume that all calls to the PRF are replaced by (consistent) random values, which yields the following game: (Note that in the definition of $\mathcal{O}_E$, we need not generate σ in Gen, as it is then discarded by San anyway.)

We first distinguish between two types of PPT adversaries:

Type 1 returns c_0, which contains an SPS-EQ forgery with non-negligible probability; that is, $(c_{0,2}, c_{0,3})$ is *not* a multiple of any (g, pk_i) where pk_i is the key obtained from oracle call $\mathcal{O}_G(i, \mathsf{sen})$.

Game Definition	**Oracle Definition**
// A queries (i', sen) in step 3 // No queries (i, sen), (i, rec) for same i // $\mathsf{EQS.Ver}(vk, (c_{0,2}, c_{0,3}), \sigma') = 1$ 1. $(sk, vk) \leftarrow \mathsf{EQS.Gen}(BG)$; $pp = (BG, vk)$; 2. $r \leftarrow \mathcal{M}$; $b \leftarrow \{0,1\}$; 3. $(c_0, i') \leftarrow A^{\mathcal{O}_E(\cdot), \mathcal{O}_S(\cdot)}(pp)$; 4. $r, s \leftarrow \mathbb{Z}_p^*$; $\sigma' \leftarrow \mathsf{EQS.Adp}(vk, \sigma_{i'}, s)$; $c_1 = (g^r, pk_{i'}^r \cdot m_b, g^s, pk_{i'}^s, \sigma')$; 5. $t \leftarrow \mathbb{Z}_p^*$; $c'_{b,0} = c_{b,0} \cdot c_{b,2}^t$; $c'_{b,1} = c_{b,1} \cdot c_{b,3}^t$ $b' \leftarrow A^{\mathcal{O}_E(\cdot), \mathcal{O}_R(\cdot)}(c'_b)$;	$\mathcal{O}_S(j,t)$ and $\mathcal{O}_R(j,t)$: 1. If pk_j not yet defined: $dk_j \leftarrow \mathbb{Z}_p$; $pk_j = g^{dk_j}$; If $t = \mathsf{rec}$ then return dk_j; If σ_j not yet defined: $\sigma_j \leftarrow \mathsf{EQS.Sig}(sk, (g, pk_j))$; Return $ek_j = (g, pk_j, \sigma_j)$; $\mathcal{O}_E(i,m)$: 1. If pk_i not yet defined: $dk_i \leftarrow \mathbb{Z}_p$; $pk_i = g^{dk_i}$; 2. $r', s', t' \leftarrow \mathbb{Z}_p^*$; // $c = (g^{r'}, pk_i^{r'} \cdot m, g^{s'}, pk_i^{s'}, \cdot)$; Return $(g^{r'+s't'}, pk_i^{r'+s't'} \cdot m)$;

Fig. 2. No-Write Rule for Constr. 2 and PRF outputs replaced by random

Type 2 returns such a forgery with negligible probability only.

Breaking EUF-CMA of SPS-EQ can be reduced to Type 1 forgeries in a straightforward fashion: the PPT reduction B simulates the no-write game using the given vk and replacing all calls of $\mathsf{EQS.Sig}(sk, (g, pk_i))$ by queries to its signature oracle; when A outputs $c_0 = (c_{0,0}, c_{0,1}, c_{0,2}, c_{0,3}, \sigma_0)$ then B returns σ_0 as a forgery on $M = (c_{0,2}, c_{0,3})$. By assumption (Type 1), with non-negligible probability M is not a multiple of the messages (g, pk_i) queried to the signing oracle; B thus breaks EUF-CMA.

We now show how to use Type 2 adversaries to break DDH assuming KEA. Let $q_{\max}$ denote an upper bound on the number of A's queries $(\cdot, \mathsf{sen})$ to $\mathcal{O}_S$ and $\mathcal{O}_R$ plus the number of queries to $\mathcal{O}_E$.

We first construct a PPT algorithm B with input (g, h). B picks two uniform values $q_0, q_1 \leftarrow [q_{\max}]$ and simulates the no-write game for A, except for the following changes: when the q_0th key pk is created during an oracle query $(j_0, \cdot)$, B sets $pk_{j_0} = h$. Let j_1 be the index of the q_1th key created. If A later queries (j_0, rec) or (j_1, rec) to $\mathcal{O}_S$ or $\mathcal{O}_R$ then B aborts. When A outputs (c_0, i') then B stops and returns $(c_{0,2}, c_{0,3})$.

Let us analyze B's behavior: Since A is of Type 2, we know that with overwhelming (i.e. all except with negligible) probability, A outputs (c_0, i') with $(c_{0,2}, c_{0,c}) = (g^a, pk_j^a)$, for some a and j s.t. (j, sen) was queried to $\mathcal{O}_S$ or $\mathcal{O}_R$. Now with probability $\frac{1}{q_{\max}^2}$, we have $j = j_0$ and $i' = j_1$. This event is independent of A's view and if it occurs then B's simulation does not abort: by assumption A makes queries (j, sen) and (i', sen) and can therefore not make queries (j, rec) and (i', rec).

With probability at least $\frac{1}{q_{\max}^2} - \mathsf{negl}(\kappa)$, B thus returns (g^a, h^a) for some a. Assuming KEA there exists thus an extractor X that, given B's coins, outputs a.

We now consider the following hybrid H of the no-write-rule game: first choose $h \leftarrow \mathbb{G}_1$ and $j_0, j_1 \leftarrow [q_{\max}]$ then run the game setting $pk_{j_1} = h$. On the same coins as used to run the game run X and let a be its output. If A's output (c_0, i') satisfies

$$(c_{0,2}, c_{0,3}) = (g^a, pk_{j_0}^a) \tag{3}$$

and $i' = j_1$ then return A's final output b'. Else return a random bit $b' \leftarrow \{0,1\}$

Since H, until the event that A outputs c_0 is defined as B, X's output a satisfies (3) with non-negligible probability, as shown above. The probability that hybrid game H outputs A's bit b' is thus non-negligible.

Further note that setting $h = pk_{j_0}$ is only a syntactical change, so H differs from the original game only in the event that the latter aborts (outputting a random bit). An analysis analogue to "$0 \rightarrow 1$" in the proof of Theorem 3 shows that if A wins the original game with non-negligible probability then it wins H with non-negligible probability.

Define H_β as H with b fixed to β. Our last step is now to show that under DDH A cannot distinguish H_0 from H_1, which contradicts A winning H and concludes the proof.

For this, we define another hybrid H'_β which modifies H_β in that c'_β is defined as $(c_{\beta,0} \cdot c_{\beta,2}^t,\ c_{\beta,1} \cdot U)$, where U is a uniform group element. Thus, c'_β is a uniformly random pair and so the game H'_β is independent of β. Therefore H'_0 is distributed as H'_1. What remains to show is that H_β is indistinguishable from H'_β.

We first show that H_0 is indistinguishable from H'_0. The games only differ when X returns a satisfying (3) (otherwise both output a random bit). In this case $h = pk_{j_0}$. Consider a DDH adversary D_0 that receives a challenge $(P, T = g^t, U)$ where either $U = P^t$ or U is random. D_0 simulates H_0 setting $h = P$ and associating the values t from the challenge and the game: it sets $c_{0,2}^t = T^a$ and $c_{0,3} = U^a$. If $U = P^t$ then D_0 simulates H_0; otherwise it simulates H'_0.

Finally, H'_1 is shown indistinguishable from H_1 by a similar reduction: on input a DDH challenge (P, T, U), D_1 simulates H_1, except that it sets $pk_{j_1} = P$ and $c_{1,2}^t = T^s$ and $c_{1,3} = U^s$. If $U = P^t$ then D_1 simulates H_1; otherwise it simulates H'_1. □

Using Theorem 3 and Theorem 4 with the SPS-EQ from [FHS14], which has perfect signature adaptation and satisfies EUF-CMA in the generic group model (GGM), we obtain the following corollary. The concrete efficiency of the resulting scheme is given in Table 2.

Corollary 3. *In the generic group model, Construction 2 instantiated with the SPS-EQ from [FHS14] satisfies the No-Read and No-Write rules.*

3.4 Comparing the Two Constructions

In Table 2 we compare the efficiency and the assumptions required for our constructions. The most efficient way to instantiate the generic construction from Section 3.2 is via structure-preserving signatures (SPS) [AFG+16], Groth-Sahai proofs [GS08] and the weakly sanitizable version of ElGamal encryption [Gam85]

Table 2. Comparison of the constructions in Section 3.2 and 3.3. In all cases pp also includes the description of the group. A ciphertext produced by Enc is denoted by c while c' denotes a sanitized ciphertext, output of San.

Construction	pp	ek	dk	c	c'	Assumpt'n
Generic$[\cdot,\cdot]$	$vk + crs$	$1\mathbb{G}_1 + sig$	$1\mathbb{Z}_p$	$4\mathbb{G}_1 + \pi$	$2\mathbb{G}_1$	
Generic[[KPW15],[GS12]]	$4\mathbb{G}_1 + 11\mathbb{G}_2$	$7\mathbb{G}_1 + 1\mathbb{G}_2$	$1\mathbb{Z}_p$	$34\mathbb{G}_1 + 16\mathbb{G}_2$	$2\mathbb{G}_1$	SXDH
Generic[[AGHO11],[GS12]]	$5\mathbb{G}_1 + 7\mathbb{G}_2$	$3\mathbb{G}_1 + 1\mathbb{G}_2$	$1\mathbb{Z}_p$	$20\mathbb{G}_1 + 14\mathbb{G}_2$	$2\mathbb{G}_1$	GGM
Construction 2 (Sect. 3.3)	$2\mathbb{G}_2$	$3\mathbb{G}_1 + 1\mathbb{G}_2$	$1\mathbb{Z}_p$	$6\mathbb{G}_1 + 1\mathbb{G}_2$	$2\mathbb{G}_1$	GGM

described in Section 3.3. The security of the latter two relies on the SXDH assumption. The most efficient SPS scheme from SXDH is the one from [KPW15] (signatures from $\mathbb{G}_1^6 \times \mathbb{G}_2$, public keys from $\mathbb{G}_2^7$). The most efficient SPS scheme with a security proof in the generic group model (GGM) is from [AGHO11] (signatures from $\mathbb{G}_1^2 \times \mathbb{G}_2$, public keys from $\mathbb{G}_1 \times \mathbb{G}_2^3$). See Corollaries 1 and 2. We also include Construction 2 from Section 3.3, which does not require zero-knowledge proofs, and which we proved secure in the GGM.

4 ACE for Disjunction of Equalities

In this section we show how to use the equality ACE scheme in a black-box way to implement more interesting predicates. Intuitively, as stated in the introduction, this is done by assigning sets of identities for the ACE scheme to each sender and receiver, in such a way that the intersection between the set $\mathsf{S}(i)$ of identities given to sender i and the set $\mathsf{R}(j)$ of identities given to receiver j is non-empty if and only if $P(i,j) = 1$. Note however that in this case a receiver, to be able to decrypt, would have to try each decryption key on each ciphertext, thus resulting in quadratic complexity. To avoid this, we compose our scheme using the following *disjunction of equalities* predicate instead: here each sender is assigned a *vector* of identities $\mathbf{x}$ and each receiver a vector of identities $\boldsymbol{y}$, and the predicate is defined as $\mathsf{P}_{\mathsf{or\text{-}eq}} : \mathcal{D}^\ell \times \mathcal{D}^\ell \to \{0,1\}$, and

$$\mathsf{P}_{\mathsf{or\text{-}eq}}(\mathbf{x},\mathbf{y}) = 1 \;\Leftrightarrow\; \bigvee_{i=1}^{\ell} (x_i = y_i) \ .$$

We give a generic construction that relies on any ACE for equality, namely, for the predicate $\mathsf{P}_{\mathsf{eq}} : (\mathcal{D} \times [\ell]) \times (\mathcal{D} \times [\ell]) \to \{0,1\}$, defined by

$$\mathsf{P}_{\mathsf{eq}}((x,i),(y,j)) = 1 \;\Leftrightarrow\; x = y \text{ and } i = j \ ,$$

such as those of Section 3. [6]

[6] To use an ACE for predicate $\mathsf{P}_{\mathsf{eq}} \colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, such as those in Section 3, one uses an injective hash function from $\mathcal{D} \times [\ell]$ to $\{0,1\}^n$, which exists as long as $2^n \geq |\mathcal{D}| \cdot [\ell]$.

Construction 3 (ACE for Disjunction of Equality – Generic)**.** *We construct an ACE scheme* $\mathsf{ACE}_{\mathsf{or\text{-}eq}}$ *for* $\mathsf{P}_{\mathsf{or\text{-}eq}}$ *from an ACE scheme* $\mathsf{ACE}_{\mathsf{eq}} = (\mathsf{Setup}_{\mathsf{eq}}, \mathsf{Gen}_{\mathsf{eq}}, \mathsf{Enc}_{\mathsf{eq}}, \mathsf{San}_{\mathsf{eq}}, \mathsf{Dec}_{\mathsf{eq}})$ *for* P_{eq}*.* $\mathsf{ACE}_{\mathsf{or\text{-}eq}}$ *is defined by the following algorithms:*

Setup: Output $(pp, msk) \leftarrow \mathsf{Setup}_{\mathsf{eq}}(1^\kappa)$.

Key Generation: Given the master secret key msk and vectors $\mathbf{x}, \mathbf{y} \in \mathcal{D}^\ell$, the encryption and decryption keys are computed as follows:

$ek_{\mathbf{x}} = (ek_{(x_1,1)}, \ldots, ek_{(x_\ell,\ell)})$ with $ek_{(x_i,i)} \leftarrow \mathsf{Gen}_{\mathsf{eq}}(msk, (x_i, i), \mathsf{sen})$ for $i \in [\ell]$;
$dk_{\mathbf{y}} = (dk_{(y_1,1)}, \ldots, dk_{(y_\ell,\ell)})$ with $dk_{(y_i,i)} \leftarrow \mathsf{Gen}_{\mathsf{eq}}(msk, (y_i, i), \mathsf{rec})$ for $i \in [\ell]$.

Encryption: On input a message m and an encryption key $ek_{\mathbf{x}} = (ek_{(x_1,1)}, \ldots, ek_{(x_\ell,\ell)})$ pick some independent randomness $r_1, \ldots, r_\ell$, compute

$$c_i = \mathsf{Enc}(ek_{(x_i,i)}, m; r_i) \ ,$$

for $i \in [\ell]$, and output $c = (c_1, \ldots, c_\ell)$.

Sanitizer: Given a ciphertext $c = (c_1, \ldots, c_\ell)$, apply $\mathsf{San}_{\mathsf{eq}}$ component-wise.

Decryption: Given a ciphertext $c = (c_1, \ldots, c_\ell)$ and a decryption key $dk_{\mathbf{y}} = (dk_{(y_1,1)}, \ldots, dk_{(y_\ell,\ell)})$ for $\mathbf{y} \in \mathcal{D}^\ell$, compute $\mathsf{Dec}(dk_{(y_i,i)}, c_i)$ for $i \in [\ell]$. Let $m_i = \mathsf{Dec}(dk_{(y_i,i)}, c_i)$, then output the first $m_i \neq \perp$ or $\perp$ if there is no such successful decryption.

Remark: Note that the complexity of the composed scheme, including the decryption algorithm, is linear in ℓ.

Lemma 6 (Correctness and Detectability). *Construction 3 is correct, according to Definition 1.*

Proof. For all $i \in [\ell]$ and $x_i, y_i \in \mathcal{D}$ such that $x_i = y_i$,

$$\Pr[\mathsf{Dec}_{\mathsf{eq}}(dk_{(y_i,i)}, \mathsf{San}_{\mathsf{eq}}(\mathsf{Enc}_{\mathsf{eq}}(ek_{(x_i,i)}, m))) = m] \geq 1 - \mathsf{negl}(\kappa) \ ,$$

by correctness of $\mathsf{ACE}_{\mathsf{eq}}$. Moreover, by detectability of $\mathsf{ACE}_{\mathsf{eq}}$, for all $x_i, y_i \in \mathcal{D}$ such that $x_i \neq y_i$, we have:

$$\Pr[\mathsf{Dec}_{\mathsf{eq}}(dk_{(y_i,i)}, \mathsf{San}_{\mathsf{eq}}(\mathsf{Enc}_{\mathsf{eq}}(ek_{(x_i,i)}, m))) = \perp] \geq 1 - \mathsf{negl}(\kappa) \ .$$

Therefore, by a union bound over the ℓ disjunctions, we obtain that for all $\mathbf{x}, \mathbf{y} \in \mathcal{D}^\ell$ such that $\mathsf{P}_{\mathsf{or\text{-}eq}}(\mathbf{x}, \mathbf{y}) = 1$:

$$\Pr[\mathsf{Dec}(dk_{\mathbf{x}}, \mathsf{San}(pp, \mathsf{Enc}(ek_{\mathbf{x}}, m))) = m] \geq 1 - \mathsf{negl}(\kappa) \ ,$$

that is, $\mathsf{ACE}_{\mathsf{or\text{-}eq}}$ is correct. A similar argument is used to show that $\mathsf{ACE}_{\mathsf{or\text{-}eq}}$ is detectable. □

Lemma 7 (No-Read-Rule). *If the underlying* $\mathsf{ACE}_{\mathsf{eq}}$ *for* P_{eq} *satisfies the No-Read-Rule from Definition 3, then so does* $\mathsf{ACE}_{\mathsf{or\text{-}eq}}$ *from Construction 3. In particular, for any PPT adversary A against the No-Read-Rule for* $\mathsf{ACE}_{\mathsf{or\text{-}eq}}$*, there exists a PPT adversary B such that*

$$\mathsf{adv}^A_{\mathsf{No\text{-}Read}}(\mathsf{ACE}_{\mathsf{or\text{-}eq}}) \leq \ell \cdot \mathsf{adv}^B_{\mathsf{No\text{-}Read}}(\mathsf{ACE}_{\mathsf{eq}}) \ .$$

Proof. We define $\ell+1$ hybrid games, where for all $i \in [\ell+1]$, Hybrid i is defined as in the table below.

Hybrid i	**Oracle Definition**
1. $(pp, msk) \leftarrow \mathsf{Setup}(1^\kappa, \mathsf{P}_{\mathsf{or\text{-}eq}})$; 2. $(m_0, m_1, \mathbf{x}^{(0)}, \mathbf{x}^{(1)}) \leftarrow A^{\mathcal{O}_G(\cdot),\mathcal{O}_E(\cdot)}(pp)$; 3. For $j \le i-1$: $c_j \leftarrow \mathsf{Enc}_{\mathsf{eq}}(ek_{(x_j^{(1)},j)}, m_1)$. For $j \ge i$: $c_j \leftarrow \mathsf{Enc}_{\mathsf{eq}}(ek_{(x_j^{(0)},j)}, m_0)$. 4. $b' \leftarrow A^{\mathcal{O}_G(\cdot),\mathcal{O}_E(\cdot)}(c_1, \ldots, c_\ell)$;	$\mathcal{O}_G(j,t)$: 1. Output $k \leftarrow \mathsf{Gen}(msk, j, t)$; $\mathcal{O}_E(i,m)$: 1. $ek_i \leftarrow \mathsf{Gen}(msk, i, \mathsf{sen})$; 2. Output $c \leftarrow \mathsf{Enc}(ek_i, m)$;

We say an adversary A wins hybrid i if it returns 1 and $|m_0| = |m_1|$, $\mathbf{x}^{(0)}$, $\mathbf{x}^{(1)} \in \mathcal{D}^\ell$, and for all queries q to $\mathcal{O}_G$ with $q = (\mathbf{y}, \mathsf{rec})$ it holds that

$$\mathsf{P}_{\mathsf{or\text{-}eq}}(\mathbf{x}^{(0)}, \mathbf{y}) = \mathsf{P}_{\mathsf{or\text{-}eq}}(\mathbf{x}^{(1)}, \mathbf{y}) = 0 \ .$$

Note that for any PPT adversary A,

$$\mathsf{adv}^A_{\mathsf{No\text{-}Read}}(\mathsf{ACE}_{\mathsf{or\text{-}eq}}) \le \tfrac{1}{2} \, |\Pr[A \text{ wins Hybrid } \ell+1] - \Pr[A \text{ wins Hybrid } 1]| \ .$$

For all $i \in [\ell]$, we build a PPT adversary B_i, such that:

$$|\Pr[A \text{ wins Hybrid } i+1] - \Pr[A \text{ wins Hybrid } i]| \le 2 \cdot \mathsf{adv}^{B_i}_{\mathsf{No\text{-}Read}}(\mathsf{ACE}_{\mathsf{eq}}) \ ,$$

thereby proving the lemma. This comes from the facts that $\mathsf{ACE}_{\mathsf{eq}}$ satisfies the No-Read Rule, and that for all $\mathbf{y} \in \mathcal{D}^\ell$, $\mathsf{P}_{\mathsf{or\text{-}eq}}(\mathbf{x}^{(0)}, \mathbf{y}) = \mathsf{P}_{\mathsf{or\text{-}eq}}(\mathbf{x}^{(1)}, \mathbf{y}) = 0$ implies $\mathsf{P}_{\mathsf{eq}}((x_i^{(0)}, i), (y_j, j)) = \mathsf{P}_{\mathsf{eq}}((x_i^{(1)}, i), (y_j, j)) = 0$ for all $i, j \in [\ell]$. □

Lemma 8 (No-Write Rule). *If the underlying* $\mathsf{ACE}_{\mathsf{eq}}$ *for* P_{eq} *satisfies the No-Write Rule from Definition 4 and the No-Read-Rule from Definition 3, then* $\mathsf{ACE}_{\mathsf{or\text{-}eq}}$ *from Construction 3 satisfies the No-Write rule. In particular, for any PPT adversary A against the No-Write Rule for* $\mathsf{ACE}_{\mathsf{or\text{-}eq}}$*, there exist PPT adversaries B_1 and B_2 such that*

$$\mathsf{adv}^A_{\mathsf{No\text{-}Write}}(\mathsf{ACE}_{\mathsf{or\text{-}eq}}) \le \ell \cdot \mathsf{adv}^{B_1}_{\mathsf{No\text{-}Write}}(\mathsf{ACE}_{\mathsf{eq}}) + 2\ell \cdot \mathsf{adv}^{B_2}_{\mathsf{No\text{-}Read}}(\mathsf{ACE}_{\mathsf{eq}}) \ .$$

Proof. As for the No-Read rule, we use a hybrid argument; for $i \in [2\ell]$ Hybrid i is defined in the tables below, where $m_0 \in \mathcal{M}$ is an arbitrary, fixed message:

Hybrid i**, for** $i \in [\ell+1]$	**Oracle Definition**
1. $(pp, msk) \leftarrow \mathsf{Setup}(1^\kappa, \mathsf{P}_{\mathsf{or\text{-}eq}})$; 2. $((c_1^{(0)}, \ldots, c_\ell^{(0)}), \mathbf{x}') \leftarrow A^{\mathcal{O}_E(\cdot),\mathcal{O}_S(\cdot)}(pp)$; 3. $(c_1^{(1)}, \ldots, c_\ell^{(1)}) \leftarrow \mathsf{Enc}(\mathsf{Gen}(msk, \mathbf{x}', \mathsf{sen}), m_0)$; 4. $b' \leftarrow A^{\mathcal{O}_E(\cdot),\mathcal{O}_R(\cdot)}(\mathsf{San}(pp, c_1^{(1)}, \ldots, c_i^{(1)}, c_{i+1}^{(0)}, \ldots, c_\ell^{(0)}))$;	$\mathcal{O}_S(j,t)$ and $\mathcal{O}_R(j,t)$: 1. Output $k \leftarrow \mathsf{Gen}(msk, j, t)$; $\mathcal{O}_E(i,m)$: 1. $ek_i \leftarrow \mathsf{Gen}(msk, i, \mathsf{sen})$; 2. Output $\mathsf{San}(pp, \mathsf{Enc}(ek_i, m))$;

Hybrid $\ell+i$, for $i \in [\ell+1]$	**Oracle Definition**
1. $(pp, msk) \leftarrow \mathsf{Setup}(1^\kappa, \mathsf{P}_{\mathsf{or\text{-}eq}})$; 2. $m_1 \leftarrow \mathcal{M}$ 3. $((c_1^{(0)}, \ldots, c_\ell^{(0)}), \mathbf{x}') \leftarrow B^{\mathcal{O}_E(\cdot), \mathcal{O}_S(\cdot)}(pp)$; 4. $(c_1^{(1)}, \ldots, c_\ell^{(1)}) \leftarrow \mathsf{Enc}(\mathsf{Gen}(msk, \mathbf{x}', \mathsf{sen}), m_0)$; 4. $(c_1^{(2)}, \ldots, c_\ell^{(2)}) \leftarrow \mathsf{Enc}(\mathsf{Gen}(msk, \mathbf{x}', \mathsf{sen}), m_1)$; 5. $b' \leftarrow B^{\mathcal{O}_E(\cdot), \mathcal{O}_R(\cdot)}(\mathsf{San}(pp, c_1^{(2)}, \ldots, c_i^{(2)}, c_{i+1}^{(1)}, \ldots, c_\ell^{(1)}))$;	$\mathcal{O}_S(j,t)$ and $\mathcal{O}_R(j,t)$: 1. Output $k \leftarrow \mathsf{Gen}(msk, j, t)$; $\mathcal{O}_E(i,m)$: 1. $ek_i \leftarrow \mathsf{Gen}(msk, i, \mathsf{sen})$; 2. Output $\mathsf{San}(pp, \mathsf{Enc}(ek_i, m))$;

Let I_S and J be defined as in the No-Write Rule game from Definition 4. We say an adversary A wins Hybrid i, for $i \in [2\ell]$, if it returns 1 and all of the following hold:

1. $\mathbf{x}' \in I_S \cup \{0\}$;
2. $\forall \mathbf{x} \in I_S, \mathbf{y} \in J$, $\mathsf{P}_{\mathsf{or\text{-}eq}}(\mathbf{x}, \mathbf{y}) = 0$.

We denote by ε_i the probability that A wins Hybrid i, for $i \in [2\ell]$. Note that for any PPT adversary A:

$$\mathsf{adv}^A_{\mathsf{No\text{-}Write}}(\mathsf{ACE}_{\mathsf{or\text{-}eq}}) \leq \tfrac{1}{2} \left| \varepsilon_{2\ell} - \varepsilon_1 \right| .$$

The proof proceeds in two steps:

First step: for all $i \in [\ell]$, we build PPT adversaries $B_{1.i}$ and $B_{2.i}$ such that $|\varepsilon_{i-1} - \varepsilon_i| \leq 2 \cdot \mathsf{adv}^{B_{1.i}}_{\mathsf{No\text{-}Write}}(\mathsf{ACE}_{\mathsf{eq}}) + 2 \cdot \mathsf{adv}^{B_{2.i}}_{\mathsf{No\text{-}Read}}(\mathsf{ACE}_{\mathsf{eq}})$.

First, the No-Write Rule allows to switch the sanitized ciphertext in Hybrid $i-1$ from

$$\mathsf{San}(pp, c_1^{(1)}, \ldots, c_{i-1}^{(1)}, \boxed{c_i^{(0)}}, c_{i+1}^{(0)}, \ldots, c_\ell^{(0)}) \text{ to}$$

$$\mathsf{San}(pp, c_1^{(1)}, \ldots, c_{i-1}^{(1)}, \boxed{c_i^{(2)}}, c_{i+1}^{(0)}, \ldots, c_\ell^{(0)}) ,$$

where $c_i^{(2)} := \mathsf{Enc}(\mathsf{Gen}(msk, (x_i', i), \mathsf{sen}), m^*)$ and $m^* \leftarrow \mathcal{M}$.

Namely, adversary $B_{1.i}$ playing against the No-Write Rule for P_{eq}, after receiving the public parameters pp, sends them to A and simulates all the queries to $\mathcal{O}_E(\cdot)$ and $\mathcal{O}_S(\cdot)$ in the straightforward way: using its own oracles $\mathcal{O}_E(\cdot)$ and $\mathcal{O}_S(\cdot)$ for P_{eq}, coordinate-wise. Note that the restriction on A's queries, namely $\forall \mathbf{x} \in I_S, \mathbf{y} \in J$, $\mathsf{P}_{\mathsf{or\text{-}eq}}(\mathbf{x}, \mathbf{y}) = 0$, implies that $\mathsf{P}_{\mathsf{eq}}((x_i, i), (y_j, j)) = 0$ for all $i, j \in [\ell]$. Thus, $B_{1.i}$ can answer valid queries from A by valid queries to its own oracles.

Then, $B_{1.i}$ receives the challenge $((c_1^{(0)}, \ldots, c_\ell^{(0)}), \mathbf{x}')$ from A, and it sends $(c_i^{(0)}, (x_i', i))$ to the challenger for P_{eq}, to receive ct_i^b where $b \leftarrow \{0,1\}$, and

$$\mathsf{ct}_i^0 := \mathsf{San}(pp, c_i^{(0)}) \text{ and } \mathsf{ct}_i^1 := \mathsf{San}(pp, \mathsf{Enc}(\mathsf{Gen}(msk, (x_i', i), \mathsf{sen}), m^*))$$

for $m^* \leftarrow \mathcal{M}$. Since $B_{1.i}$ knows m_0 (here we crucially rely on the fact that m_0 is a fixed message, and not a random message as in the No-Write Rule experiment, since it would be unknown to $B_{1.i}$), it can compute

$$\mathsf{ct}_j := \mathsf{San}(pp, \mathsf{Enc}(\mathsf{Gen}(msk, (x_j', j), \mathsf{sen}), m_0)) \text{ for } j < i ,$$

using its $\mathcal{O}_E$ oracle on input $((x'_j, j), m_0)$. Finally, it sets $\mathsf{ct}_j := c_j^{(0)}$ for $j > i$, and sends the sanitized ciphertext $(\mathsf{ct}_1, \ldots, \mathsf{ct}_{i-1}, \mathsf{ct}_i^b, \mathsf{ct}_{i+1}, \ldots, \mathsf{ct}_\ell)$ to A, and keeps simulating the oracles $\mathcal{O}_E(\cdot)$ and $\mathcal{O}_R(\cdot)$ as before.

Then, because $\mathsf{ACE}_{\mathsf{eq}}$ satisfies the No-Read Rule, and because for all $\mathbf{y} \in J$, $\mathsf{P}_{\mathsf{or\text{-}eq}}(\mathbf{x}', \mathbf{y}) = 0$, which implies $\mathsf{P}_{\mathsf{eq}}((x'_i, i), (y_j, j)) = 0$ for all $i, j \in [\ell]$, we can switch a sanitized ciphertext from

$$\mathsf{San}(pp, c_1^{(1)}, \ldots, c_{i-1}^{(1)}, \boxed{c_i^{(2)}}, c_{i+1}^{(0)}, \ldots, c_\ell^{(0)}) \text{ to}$$
$$\mathsf{San}(pp, c_1^{(1)}, \ldots, c_{i-1}^{(1)}, \boxed{c_i^{(1)}}, c_{i+1}^{(0)}, \ldots, c_\ell^{(0)}) ,$$

where $c_i^{(1)} = \mathsf{Enc}(\mathsf{Gen}(msk, (x'_j, j), \mathsf{sen}), m_0)$, as in Hybrid i. Namely, adversary $B_{2.i}$ simulates pp, $\mathcal{O}_E(\cdot)$, $\mathcal{O}_S(\cdot)$, $\mathcal{O}_R(\cdot)$, and computes sanitized ciphertexts ct_j for $j > i$ as described previously for $B_{1.i}$. For the ciphertexts ct_j for $j < i$, $B_{2.i}$ uses its oracle $\mathcal{O}_E$, and then, applies San to obtain the sanitized ciphertexts. It can do so since applying San only requires to know pp. Then, $B_{2.i}$ sends $(m_0, m_1, (x'_i, i), (x'_i, i))$ to the No-Read Rule experiment, where $m_1 \leftarrow \mathcal{M}$, to get back $c \leftarrow \mathsf{Enc}(\mathsf{Gen}(msk, (x'_i, i)), m_b)$. It sets $\mathsf{ct}_i := \mathsf{San}(pp, c)$, and sends the sanitized $(\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$ to A.

Second step: we build a PPT adversary $B_{3.i}$ such that $|\varepsilon_{\ell+i-1} - \varepsilon_{\ell+i}| \leq 2 \cdot \mathsf{adv}^{B_{3.i}}_{\mathsf{No\text{-}Read}}(\mathsf{ACE}_{\mathsf{eq}})$.

We use the No-Read Rule as for the first step. Namely, $B_{3.i}$ simulates pp, $\mathcal{O}_E(\cdot)$, $\mathcal{O}_S(\cdot)$, $\mathcal{O}_R(\cdot)$ as descried previously for $B_{2.i}$. Then, $B_{3.i}$ ignores the challenge $((c_1^{(0)}, \ldots, c_\ell^{(0)}), \mathbf{x}')$ sent by A, samples $m_1 \leftarrow \mathcal{M}$, computes

$$ct_j := \mathsf{San}(pp, \mathsf{Enc}(\mathsf{Gen}(msk, (x'_j, j), \mathsf{sen}), m_1)) \text{ for } j < i ,$$
$$ct_j := \mathsf{San}(pp, \mathsf{Enc}(\mathsf{Gen}(msk, (x'_j, j), \mathsf{sen}), m_0)) \text{ for } j > i ,$$

thanks to its oracle $\mathcal{O}_E$. Then, $B_{3.i}$ sends $(m_0, m_1, (x'_i, i), (x'_i, i))$ to the No-Read Rule experiment, to get back $c \leftarrow \mathsf{Enc}(\mathsf{Gen}(msk, (x'_i, i)), m_b)$. It sets $\mathsf{ct}_i := \mathsf{San}(pp, c)$, and sends the sanitized ciphertext $(\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$ to A. □

5 Predicates in Disjunction of Equalities

We show how to reduce the predicate $\mathsf{P}_{\mathsf{range}}$ defined for all points $z \in [N]$ and intervals $I \subset [N]$ as:

$$\mathsf{P}_{\mathsf{range}}(z, I) = 1 \Leftrightarrow z \in I$$

to $\mathsf{P}_{\mathsf{or\text{-}eq}}$ described in Section 4. This requires writing intervals I and points z as vectors, using a standard tree structure [DVOS00].

Lemma 9 (Interval to Vector [DVOS00]). *There is an efficient PPT algorithm* IntVec*, that on input an interval $I \subset [N]$ outputs*

$$(w_1, w_2, \ldots, w_{2n}) \in \left(\{0,1\}^* \cup \{\bot\}\right)^{2n} ,$$

where $n := \lceil \log N \rceil$, with the following properties:

– *for each* $i = 1, \ldots, n$, *we have* $w_{2i-1}, w_{2i} \in \{0,1\}^i \cup \{\perp\}$;
– *for all* $z \in [N]$, *we have* $z \in I$ *iff one of* $w_1, \ldots, w_{2t}$ *is a prefix of* z.

Here, $\perp$ *is special symbol such that* $\perp \notin \bigcup_{i=1}^{n} \{0,1\}^i$.

For instance, $\mathsf{IntVec}([010, 110]) = (\perp, \perp, 01, 10, 110, \perp)$.

Remark 1 (Hashing bit strings into $\mathcal{D}$*).* We want to use the ACE of Section 4, which requires finding an injective map from $\bigcup_{i=1}^{n} \{0,1\}^i \cup \{\perp\}$ into $\mathcal{D}$, where $n := [\lceil \log N \rceil]$. Such map exists as long as $|\mathcal{D}| \geq 2^{n+1} - 1$.

Now we give the description of algorithm PtVec, used to map points to vectors.

PtVec: On input $z \in [N]$, output $(v_1, \ldots, v_{2n})$, where

$$v_{2i-1} = v_{2i} := i\text{'th bit prefix of } z, \ i = 1, \ldots, n \ .$$

For instance, $\mathsf{PtVec}(011) = (0, 0, 01, 01, 011, 011)$.

Remark 2 (Duplicate Entries). Note that some strings appear more than once in the vector. This is necessary since the predicate is a function of both the *entries in the vector* and *their positions*.

Lemma 10. *For any point* $z \in [N]$ *and any interval* $I \subseteq [N]$,

$$z \in I \text{ iff } \mathsf{P}_{\mathit{or\text{-}eq}}(\mathsf{PtVec}(z), \mathsf{IntVec}(I)) = 1 \ .$$

Lemma 10 follows readily from Lemma 9.

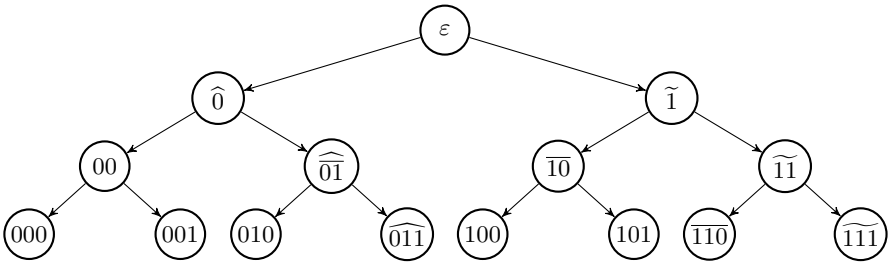

Fig. 3. Tree structure [DVOS00] for interval $[010, 110]$ (bar nodes), point 011 (hat nodes) and point 111 (tilde nodes). The common node 01 allows to decrypt for $011 \in [010, 110]$. No such node exists for $111 \notin [010, 110]$, which prevents decryption.

Acknowledgements. We would like to thank the reviewers of PKC'17 for their diligent proofreading and valuable remarks, helping us to improve the paper.

Fuchsbauer is supported in part by the French ANR ALAMBIC project (ANR-16-CE39-0006). Gay is supported by ERC Project aSCEND (639554).

Kowalczyk is supported in part by the Defense Advanced Research Project Agency (DARPA) and Army Research Office (ARO) under Contract #W911NF-15-C-0236; NSF grants #CNS-1445424, #CNS-1552932, and #CCF-1423306; an NSF Graduate Research Fellowship #DGE-16-44869; and ERC Project aSCEND (639554). Any opinions, findings, and conclusions or recommendations expressed are those of the authors and do not necessarily reflect the views of the Defense Advanced Research Projects Agency, Army Research Office, the National Science Foundation, or the U.S. Government.

Orlandi is supported by the Danish National Research Foundation (grant 61361136003) and COST Action IC1306.

References

AFG+10. Masayuki Abe, Georg Fuchsbauer, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. Structure-preserving signatures and commitments to group elements. In *Advances in Cryptology - CRYPTO 2010. Proceedings*, pages 209–236, 2010.

AFG+16. Masayuki Abe, Georg Fuchsbauer, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. Structure-preserving signatures and commitments to group elements. *J. Cryptology*, 29(2):363–421, 2016.

AGHO11. Masayuki Abe, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. Optimal structure-preserving signatures in asymmetric bilinear groups. In *Advances in Cryptology - CRYPTO 2011, Proceedings*, pages 649–666, 2011.

BBDP01. Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-privacy in public-key encryption. In *Advances in Cryptology - ASIACRYPT 2001, Proceedings*, pages 566–582, 2001.

BF01. Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In *Advances in Cryptology - CRYPTO 2001, Proceedings*, pages 213–229, 2001.

BFM88. Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *20th Annual ACM Symposium on Theory of Computing*, pages 103–112, 1988.

BP04. Mihir Bellare and Adriana Palacio. The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols. In *Advances in Cryptology - CRYPTO 2004, Proceedings*, pages 273–289, 2004.

BSW11. Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In *Theory of Cryptography - 8th Theory of Cryptography Conference, TCC 2011. Proceedings*, pages 253–273, 2011.

DHO16. Ivan Damgård, Helene Haagh, and Claudio Orlandi. Access control encryption: Enforcing information flow with cryptography. In *Theory of Cryptography - 14th International Conference, TCC 2016-B, Proceedings, Part II*, pages 547–576, 2016.

DVOS00. Mark De Berg, Marc Van Kreveld, Mark Overmars, and OtfriedCheong Schwarzkopf. *Computational Geometry*. Springer Berlin Heidelberg, 2000.

FHS14. Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Structure-preserving signatures on equivalence classes and constant-size anonymous credentials. *IACR Cryptology ePrint Archive*, 2014:944, 2014.

FHS15. Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Practical round-optimal blind signatures in the standard model. In *Advances in Cryptology - CRYPTO 2015 , Proceedings, Part II*, pages 233–253, 2015.

Gam85. Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

GGH+13. Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS*, pages 40–49, 2013.

GMW15. Romain Gay, Pierrick Méaux, and Hoeteck Wee. Predicate encryption for multi-dimensional range queries from lattices. In *Public-Key Cryptography - PKC 2015, Proceedings*, pages 752–776, 2015.

Gol09. Oded Goldreich. *Foundations of cryptography: volume 2, basic applications*. Cambridge university press, 2009.

GS08. Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *Advances in Cryptology - EUROCRYPT 2008, Proceedings*, pages 415–432, 2008.

GS12. Jens Groth and Amit Sahai. Efficient noninteractive proof systems for bilinear groups. *SIAM J. Comput.*, 41(5):1193–1232, 2012.

HS14. Christian Hanser and Daniel Slamanig. Structure-preserving signatures on equivalence classes and their application to anonymous credentials. In *Advances in Cryptology - ASIACRYPT 2014, Proceedings, Part I*, pages 491–511, 2014.

IPV10. Malika Izabachène, David Pointcheval, and Damien Vergnaud. Mediated traceable anonymous encryption. In Michel Abdalla and Paulo S. L. M. Barreto, editors, *Progress in Cryptology - LATINCRYPT 2010, Proceedings*, volume 6212 of *LNCS*, pages 40–60. Springer, 2010.

KL14. Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography*. CRC press, 2014.

KPW15. Eike Kiltz, Jiaxin Pan, and Hoeteck Wee. Structure-preserving signatures from standard assumptions, revisited. In *Advances in Cryptology - CRYPTO 2015, Part II*, pages 275–295, 2015.

OT12. Tatsuaki Okamoto and Katsuyuki Takashima. Adaptively attribute-hiding (hierarchical) inner product encryption. In *Advances in Cryptology - EUROCRYPT 2012*, pages 591–608, 2012.

Sak00. Ryuichi Sakai. Cryptosystems based on pairings. In *Symposium on Cryptography and Information Security 2000, SCIS2000*, 2000.

SBC+07. Elaine Shi, John Bethencourt, Hubert T.-H. Chan, Dawn Xiaodong Song, and Adrian Perrig. Multi-dimensional range query over encrypted data. In *2007 IEEE Symposium on Security and Privacy (S&P 2007), 20-23 May 2007, Oakland, California, USA*, pages 350–364, 2007.

Sha84. Adi Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology, Proceedings of CRYPTO '84*, pages 47–53, 1984.

SW05. Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *Advances in Cryptology - EUROCRYPT 2005*, pages 457–473, 2005.