Public-Key Encryption Resistant to Parameter Subversion and its Realization from Efficiently-Embeddable Groups

Benedikt Auerbach[1], Mihir Bellare[2], and Eike Kiltz[1]

[1] Horst-Görtz Institute for IT Security and Faculty of Mathematics, Ruhr-University Bochum, Germany
{ benedikt.auerbach, eike.kiltz }@rub.de
[2] Department of Computer Science & Engineering, University of California San Diego, 9500 Gilman Drive, La Jolla, California 92093, USA
mihir@eng.ucsd.edu.

Abstract. We initiate the study of public-key encryption (PKE) schemes and key-encapsulation mechanisms (KEMs) that retain security even when public parameters (primes, curves) they use may be untrusted and subverted. We define a strong security goal that we call ciphertext pseudorandomness under parameter subversion attack (CPR-PSA). We also define indistinguishability (of ciphertexts for PKE, and of encapsulated keys from random ones for KEMs) and public-key hiding (also called anonymity) under parameter subversion attack, and show they are implied by CPR-PSA, for both PKE and KEMs. We show that hybrid encryption continues to work in the parameter subversion setting to reduce the design of CPR-PSA PKE to CPR-PSA KEMs and an appropriate form of symmetric encryption. To obtain efficient, elliptic-curve-based KEMs achieving CPR-PSA, we introduce efficiently-embeddable group families and give several constructions from elliptic-curves.

1 Introduction

This paper initiates a study of public-key encryption (PKE) schemes, and key-encapsulation mechanisms (KEMs), resistant to subversion of public parameters. We give definitions, and efficient, elliptic-curve-based schemes. As a tool of independent interest, we define efficiently-embeddable group families and construct them from elliptic curves.

Parameter subversion. Many cryptographic schemes rely on some trusted, public parameters common to all users and implementations. Sometimes these are specified in standards. The Oakley primes [39], for example, are a small number of fixed prime numbers widely used for discrete-log-based systems. For ECC (Elliptic Curve Cryptography), the parameters are particular curves. Examples include the P-192, P-224, ... curves from the FIPS-186-4 [38] standard and Ed25519 [16].

There are many advantages to such broad use of public parameters. For example, it saves implementations from picking their own parameters, a task that can be error-prone and difficult to do securely. It also makes key-generation faster and allows concrete-security improvements in the multi-user setting [7]. Recent events indicate, however, that public parameters also bring a risk, namely that they can be *subverted*. The representative example is Dual-EC. We refer to [19] for a comprehensive telling of the story. Briefly, Dual EC was a PRG whose parameters consisted of a description of a cyclic group and two generators of the group. If the discrete logarithm of one generator to base the other were known, security would be compromised. The Snowden revelations indicate that NIST had adopted parameters provided by the NSA and many now believe these parameters had been subverted, allowing the NSA to compromise the security of Dual EC. Juniper's use of Dual EC further underscores the dangers [21].

Security in the face of parameter subversion. DGGJR [26] and BFS [9] initiated the study of cryptography that retains security in the face of subverted parameters, the former treating PRGs and the latter treating NIZKs, where the parameter is the common reference string. In this paper we treat encryption. We define what it means for parameter-using PKE schemes and KEMs to retain security in the face of subversion of their parameters. With regard to schemes, ECC relies heavily on trusted parameters. Accordingly we focus here, providing various efficient elliptic-curve-based schemes that retain security in the face of parameter subversion.

Current mitigations. In practice, parameters are sometimes specified in a verifiable way, for example derived deterministically (via a public algorithm) from publicly-verifiable coins. The coins could be obtained by applying a hash function like SHA1 to some specified constants (as is in fact done for the FIPS-186-4 curves [38] and in the ECC brainpool project), via the first digits of the irrational number π, or via lottery outcomes [5]. This appears to reduce the possibility of subversion, but BCCHLN [15] indicate that the potential of subverting elliptic curves still remains, so there is cause for caution even in this regard. Also, even if such mechanisms might "work" in some sense, we need definitions to understand what "work" means, and proofs to ensure definitions are met. Our work gives such definitions.

Background. A PKE scheme specifies a parameter generation algorithm that returns parameters π, a key-generation algorithm that takes π and returns a public key pk and matching secret key sk, an encryption algorithm that given π, pk and message m returns a ciphertext c, and a decryption algorithm that given π, sk, c recovers m. We denote the classical notions of security by IND —indistinguishability of ciphertexts under chosen-ciphertext attack [8, 22]— and PKH —public-key hiding, also called anonymity, this asks that ciphertexts not reveal the public key under which they were created [6]. For KEMs, parameter and key generation are the same, encryption is replaced by encapsulation —it takes π, pk to return an encapsulated key K and a ciphertext c that encapsulates K— and decryption is replaced by decapsulation —given π, sk, c it recovers K. We continue to denote the classical goals by IND —this now asks for indistinguishability of encapsulated keys from random under chosen-ciphertext

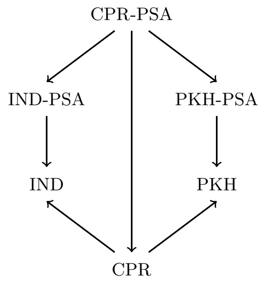

Fig. 1. Relations between notions of security. The notions are defined, and the relations hold, for both PKE schemes and KEMs. An arrow A → B is an implication: if a scheme meets A then it also meets B.

attack [23]— and PKH. We stress that these classical notions assume *honest parameter generation*, meaning *the parameters are trusted.*

We know that, in this setting, IND PKE is reduced, via hybrid encryption, to IND KEMs and ind-cpa symmetric encryption [23]. To the best of our knowledge, no analogous result exists for PKH.

Mass surveillance activities have made apparent the extent to which privacy can be violated purely by access to meta-data, including who is communicating with whom. PKE and KEMs providing PKH are tools towards systems that do more to hide identities of communicants. We will thus target this goal in the parameter subversion setting as well.

Definitions and relations. For both PKE and KEMs, we formulate a goal called ciphertext pseudorandomness under parameter subversion attack, denoted CPR-PSA. It asks that ciphertexts be indistinguishable from strings drawn randomly from the ciphertext space, even under a chosen-ciphertext attack (CCA). We also extend the above-discussed classical goals to the parameter subversion setting, defining IND-PSA and PKH-PSA. For both PKE (Proposition 1) and KEMs (Proposition 2) we show that CPR-PSA implies both IND-PSA and PKH-PSA. We thus get the relations between the new and classical notions summarized in Figure 1. (Here CPR is obtained by dropping the PSA in CPR-PSA, meaning it is our definition with honest parameter generation. This extends the notions of [37, 26] to allow a CCA.)

We ask whether we can reduce the design of CPR-PSA PKE to the design of CPR-PSA KEMs via hybrid encryption. Proposition 3 says the answer is yes, but, interestingly, requires that the KEM has an extra property of well-distributed ciphertexts that we denote WDC-PSA. (The symmetric encryption scheme is required to have pseudo-random ciphertexts. Such symmetric schemes are easily obtained.) We now have a single, strong target for constructions, namely CPR-PSA+WDC-PSA KEMs. (By the above they imply CPR-PSA PKE, which in turn implies IND-PSA PKE and PKH-PSA PKE.) Our goal thus becomes to build efficient KEMs that are CPR-PSA+WDC-PSA.

Parameter-free schemes. We say that a scheme (PKE or KEM) is parameter free if there are no parameters. (Formally, the parameters are the empty string ε.) Note that a parameter-free scheme that is XXX secure is trivially also XXX-PSA secure. (XXX $\in \{\mathrm{CPR}, \mathrm{IND}, \mathrm{PKH}\}$.) This is an important observation, and some of our schemes will indeed be parameter-free, but, as we discuss next, this observation does not trivialize the problem.

Issues and challenges. In an attempt to achieve PSA security through the above observation, we could consider the following simple way to eliminate parameters. Given a XXX-secure parameter-using scheme, build a parameter-free version of it as follows: the new scheme sets its parameters to the empty string; key generation runs the old parameter generation algorithm to get π, then the old key generation algorithm to get *pk* and *sk*, setting the new public and secret keys to (π, pk) and (π, sk), respectively; encryption and decryption can then follow the old scheme. This trivial construction, however, has drawbacks along two dimensions that we expand on below: (1) security and (2) efficiency.

With regard to security, the question is, if the old scheme is XXX, is the new one too? (If so, it is also XXX-PSA, since it is parameter free, so we only need to consider the classical notions.) The answer to the question is yes if XXX = IND, but *no if* XXX $\in \{\mathrm{PKH}, \mathrm{CPR}\}$. Imagine, as typical, that the parameters describe a group. Then in the new scheme, different users use different, independent groups. This will typically allow violation of PKH [6]. For example, in the El Gamal KEM, a ciphertext is a group element, so if two users have groups $\mathbb{G}_0$ and $\mathbb{G}_1$, respectively, one can determine which user generated a ciphertext by seeing to which of the two groups it belongs. The same is true for RSA where the group $\mathbb{G}_i = \mathbb{Z}_{N_i}$ is determined by the modulus N_i in the key of user i. Even when the moduli have the same bit length, attacks in [6] show how to violate PKH-security of the simple RSA KEM.

With regard to efficiency, the drawback is that we lose the benefits of parameter-using schemes noted above. In particular, key-generation is less efficient (because it involves parameter generation for the old scheme, which can be costly), and public keys are longer (because they contain the parameters of the old scheme). We aim to retain, as much as possible, the efficiency benefits of parameters while adding resistance to PSA.

BBDP [6] give (1) parameter-free IND+PKH RSA-based PKE schemes and (2) parameter-using discrete-log based IND+PKH PKE schemes. The former, since parameter-free, are IND-PSA+PKH-PSA, but they are not CPR-PSA and they are not as efficient as ECC-based schemes. The latter, while ECC-based and fast, are not secure against PSA.

The open question that emerges is thus to design efficient, ECC-based KEMs that are CPR-PSA+WDC-PSA. The technical challenge is to achieve CPR-PSA (and thus PKH-PSA) even though the groups of different users may be different.

Overview of the approach. We introduce and formalize *efficiently-embeddable group (eeg) families* and identify desirable security properties for them. We give a transform constructing CPR-PSA+WDC-PSA KEMs from secure eeg families. This reduces our task to finding secure eeg families. We propose several

eeg family	**Transform**	**Parameter**	**Assumption**	**Efficiency**			**Key size**
				$\mathsf{KE.G}$	$\mathsf{KE.E}$	$\mathsf{KE.D}$	
$\mathsf{EG}_{\mathrm{twist}}$	**eegToKE1**	p	sCDH-PSA	t_{TGen}	2,2	2	$10k$
$\mathsf{EG}_{\mathrm{twist}}$	**eegToKE2**	p	CDH-PSA	t_{TGen}	3,3	3	$12k$
$\mathsf{EG}^{\ell}_{\mathrm{twist\text{-}rs}}$	**eegToKE1**	—	sCDH-PSA	t_{TGen}	$3,\ \ell{+}1$	1	$9k$
$\mathsf{EG}^{\ell}_{\mathrm{twist\text{-}rs}}$	**eegToKE2**	—	CDH-PSA	t_{TGen}	$4,\ \ell{+}2$	2	$11k$
$\mathsf{EG}_{\mathrm{twist\text{-}re}}$	**eegToKE1**	—	sCDH-PSA	t_{TGen}	3, 3	1	$9k$
$\mathsf{EG}_{\mathrm{twist\text{-}re}}$	**eegToKE2**	—	CDH-PSA	t_{TGen}	4, 4	2	$11k$
$\mathsf{EG}^{\ell}_{\mathrm{ell1}}$, $\mathsf{EG}^{\ell}_{\mathrm{ell2}}$	**eegToKE1**	p	sCDH-PSA	t_{EllGen}	$3, \ell + 1$	1	$6k$
$\mathsf{EG}^{\ell}_{\mathrm{ell1\text{-}rs}}$, $\mathsf{EG}^{\ell}_{\mathrm{ell2\text{-}rs}}$	**eegToKE1**	—	sCDH-PSA	t_{EllGen}	$5,\ \ell{+}1$	1	$5k$

Table 1. Our elliptic curve based CPR-PSA+WDC-PSA KEMs. p denotes the modulus of the field. Efficiency of $\mathsf{KE.G}$ is dominated by the sampling time of the curves. Efficiency of $\mathsf{KE.E}$ (average, worst case) and $\mathsf{KE.D}$ (worst case) is given as the number of exponentiations on the curves. The key size is measured in bits, $k = \lceil |\mathbb{F}_p| \rceil$ being the bit length of the used modulus. For the rejection sampling based constructions, ℓ denotes the cut-off bound. For transform **eegToKE2** and the constructions based on Elligator curves (last two rows) see [4].

---

instantiations of eeg families from elliptic curves with security based on different assumptions. An overview of the resulting KEMs is given in Table 1. We discuss our results in greater detail below.

Efficiently-embeddable group families. As described above, having users utilize different groups typically enables linking ciphertexts to the intended receiver and hence violating CPR-PSA. However, certain families of groups allow to efficiently map group elements to a space, which is independent of the particular group of the family. Building on these types of group families it is possible to achieve CPR-PSA secure encryption while still allowing each user to choose his own group.

We formalize the required properties via *efficiently embeddable group families*, a novel abstraction that we believe is of independent interest. An eeg family EG specifies a parameter generation algorithm $\mathsf{EG.P}$ sampling parameters to be used by the other algorithms, and a group generation algorithm $\mathsf{EG.G}$ sampling a group from the family. Embedding algorithm $\mathsf{EG.E}$ embeds elements of the group into some embedding space $\mathsf{EG.ES}$. The group element can be recovered using inversion algorithm $\mathsf{EG.I}$. An important property is that the embedding space only depends on the parameters and in particular not on the used group. Looking ahead, the KEM's public key will contain a group sampled with $\mathsf{EG.S}$ and ciphertexts will be embeddings. We require two security properties for EG in order to achieve CPR-PSA+WDC-PSA KEMs. Both assume parameter subversion attacks and are defined with respect to a sampling algorithm $\mathsf{EG.S}$, which samples (not necessarily uniformly distributed) group elements. The first, embedding pseudorandomness (EPR-PSA), is that embeddings of group elements sampled with $\mathsf{EG.S}$ are indistinguishable from uniform. Further we give a definition the strong computational Diffie-Hellman assumption (sCDH-PSA) with respect to

EG— an adaption of the interactive assumption introduced in [2] to our setting. It differs from the usual strong computational Diffie-Hellman assumption in two points. The group used for the challenge is sampled using $\mathsf{EG.G}$ on a parameter of the adversary's choice and additionally one of the exponents used in the challenge is sampled with sampling algorithm $\mathsf{EG.S}$.

Key ecapsulation mechanisms from eeg families. We provide a transform **eegToKE1** of eeg families to secure KEMs. If the eeg family is both EPR-PSA and sCDH-PSA the resulting KEM is CPR-PSA and WDC-PSA.

Key encapsulation from weaker assumptions. In the full version of this paper [4] we give a second transform **eegToKE2** from eeg families to secure KEMs. It is applicable to eeg families consisting of groups, which order has no small prime factors. Its security is based on the weaker computational Diffie-Hellman assumption (CDH-PSA), i.e. it achieves a CPR-PSA and WDC-PSA KEM under the weaker assumption that EG is both EPR-PSA and CDH-PSA. However, this comes at the cost of larger key size and slower encryption and decryption.

Instantiations from elliptic curves. We propose several instantiations of eeg families from elliptic curves. It is well known that elliptic curves are not all equal in security. We target elliptic-curve groups over the field $\mathbb{F}_p$ for a large odd prime p since they are less vulnerable to discrete-log-finding attacks than groups over fields of characteristic two [28, 40]. While the usage of standardized primes allows for more efficient implementations, several cryptanalysts further suggest that p should be as random as possible for maximal security, see for example Brainpool's RFC on ECC [36]. These constraints make building eeg families more challenging. We offer solutions for both cases. We first identify an eeg family implicitly given in prior work [34, 37]. The family consists of curve-twist pairs of elliptic curves. Its embedding space depends on the modulus p of the underlying field, which serves as parameter of the construction.

Building on eeg family $\mathsf{EG}_{\text{twist}}$ we also provide alternatives, which no longer rely on a fixed modulus. The constructions have empty parameters and p is sampled at random in the group generation algorithm. The technical challenge is to still achieve pseudorandom embeddings in an embedding space independent of the group. Our solution $\mathsf{EG}^{\ell}_{\text{twist-rs}}$ achieves this by using rejection sampling with cut-off parameter ℓ. Its embedding space consists of bit strings of length only dependent on the security parameter. The sampling algorithm has a worst-case running time of ℓ exponentiations, but the average cost is two exponentiations independently of ℓ. Eeg family $\mathsf{EG}_{\text{twist-re}}$ uses a range expansion technique from [33] and improves on $\mathsf{EG}^{\ell}_{\text{twist-rs}}$ both in terms of efficiency and security. As in the other construction embeddings are bit strings, but sampling only requires a single exponentiation.

Security of the instantiations. We now discuss the security properties of our instantiations in greater detail. An overview is given in Table 2. All of our constructions achieve EPR-PSA statistically. Embeddings in eeg families $\mathsf{EG}_{\text{twist}}$, and $\mathsf{EG}_{\text{twist-re}}$ are perfectly random, i.e. any (unbounded) adversary has advantage

eeg family	Curve type	Parameter	$\Delta_{\text{EPR-PSA}}$	See
$\mathsf{EG}_{\text{twist}}$	twist	p	0	§ 5.2
$\mathsf{EG}^{\ell}_{\text{twist-rs}}$	twist	—	$(1/2)^{\ell}$	§ 5.3
$\mathsf{EG}_{\text{twist-re}}$	twist	—	0	§ 5.4
$\mathsf{EG}^{\ell}_{\text{ell1}}$, $\mathsf{EG}^{\ell}_{\text{ell2}}$	Elligator	p	$(2/3)^{\ell}$	[4]
$\mathsf{EG}^{\ell}_{\text{ell1-rs}}$, $\mathsf{EG}^{\ell}_{\text{ell2-rs}}$	Elligator	—	$(4/5)^{\ell}$	[4]

Table 2. Security of our eeg families. The modulus of the used field is denoted by p. $\Delta_{\text{EPR-PSA}}$ denotes the maximal advantage of an (unbounded) adversary in breaking EPR-PSA. ℓ denotes the cut-off bound used in the construction based on rejection sampling.

0 in breaking EPR-PSA. For family $\mathsf{EG}^{\ell}_{\text{twist-rs}}$ the advantage decays exponentially in the cut-off bound ℓ.

Diffie-Hellman problem sCDH-PSA is non standard. It is defined with respect to the eeg family's sampling algorithm and assumes parameter subversion attacks. However, for all of our proposed instantiations we are able to show that sCDH-PSA can be reduced to assumptions, which no longer depend on the sampling algorithms, but use uniformly sampled exponents instead. Considering the parameters of our constructions, they belong to one of two classes. Eeg familiy $\mathsf{EG}_{\text{twist}}$ uses the modulus p as parameter, which might be subject to subversion. Accordingly sCDH-PSA in this case corresponds to the assumption that the adversary's possibility to choose p does not improve its capacities in solving Diffie-Hellman instances on either the curve or its twist for a curve-twist pair sampled from the family. Eeg families $\mathsf{EG}^{\ell}_{\text{twist-rs}}$ and $\mathsf{EG}_{\text{twist-re}}$ serve as more conservative alternatives. They are parameter-free and each user choses his own modulus at random, resulting in the weaker assumption that solving Diffie-Hellman instances over curves sampled with respect to a randomly chosen modulus is hard.

Instantiations from Elligator curves In the full version of this paper [4] we provide alternatives to our curve-twist pair based constructions. Eeg families $\mathsf{EG}^{\ell}_{\text{ell1}}$, $\mathsf{EG}^{\ell}_{\text{ell2}}$, $\mathsf{EG}^{\ell}_{\text{ell1-rs}}$ and $\mathsf{EG}^{\ell}_{\text{ell2-rs}}$ make use of the Elligator1 and Elligator2 curves of [17]. $\mathsf{EG}^{\ell}_{\text{ell1}}$ and $\mathsf{EG}^{\ell}_{\text{ell2}}$ were implicitly given in [17] and use the modulus of the underlying field as parameter. Constructions $\mathsf{EG}^{\ell}_{\text{ell1-rs}}$ and $\mathsf{EG}^{\ell}_{\text{ell2-rs}}$ serve as parameter-free alternatives.

Related work. One might consider generating parameters via a multi-party computation protocol so that no particular party controls the outcome. It is unclear however what parties would perform this task and why one might trust any of them. PKE resistant to parameter subversion provides greater security.

Parameter subversion as we consider it allows the adversary full control of the parameters. This was first considered for NIZKs [9] and (under the term backdoored) for PRGs [26, 25]. Various prior works, in various contexts, considered relaxing the assumptions on parameters in some way [20, 32, 30, 35], but these do not allow the adversary full control of the parameters and thus do not provide security against what we call parameter subversion.

Algorithm-substitution attacks, studied in [12, 10, 24, 11, 3], are another form of subversion, going back to the broader framework of kleptography [43, 44]. The cliptography framework of RTYZ [41] aims to capture many forms of subversion. In [42] the same authors consider PKE that retains security in the face of substitution of any of its algorithms, but do not consider parameter subversion.

2 Preliminaries

Notation. We let ε denote the empty string. If X is a finite set, we let $x \leftarrow\!\!\$\; X$ denote picking an element of X uniformly at random and assigning it to x. All our algorithms are randomized and polynomial time (PT) unless stated otherwise. An adversary is an algorithm. Running time is worst case. If A is an algorithm, we let $y \leftarrow A(x_1, \ldots ; r)$ denote running A with random coins r on inputs $x_1, \ldots$ and assigning the output to y. We let $y \leftarrow\!\!\$\; A(x_1, \ldots)$ be the result of picking r at random and letting $y \leftarrow A(x_1, \ldots ; r)$. We let $[A(x_1, \ldots)]$ denote the set of all possible outputs of A when invoked with inputs $x_1, \ldots$. We use the code based game playing framework of [14]. (See Figure 3 for an example.) By Pr[G] we denote the probability that the execution of game G results in the game returning true. We also adopt the convention that the running time of an adversary refers to the worst case execution time of the game with the adversary. This means that the time taken for oracles to compute replies to queries is included. The random oracle model [13] is captured by a game procedure RO that implements a variable output length random oracle. It takes a string x and an integer m and returns a random m-bit string. We denote by $\mathcal{P}_k$ the set of primes of bit length k and by $[d]$ the set $\{0, \ldots, d-1\}$. Furthermore, the uniform distribution on M is denoted by U_M. If two random variables X and Y are equal in distribution we write $X \sim Y$. The statistical distance between X and Y is denoted by $\mathit{\Delta}(X;Y)$. If $\mathit{\Delta}(X;Y) \leq \delta$ we say X is δ-close to Y.

3 Public-Key Encryption Resistant to Parameter Subversion

In this section we recall public-key encryption schemes and key encapsulation mechanisms. For both primitives we define the strong security notion of pseudorandomness of ciphertexts in the setting of parameter subversion and show that it implies both indistinguishability of encryptions and public-key hiding. We further define the security notion of well-distributedness of ciphertexts for key encapsulation mechanisms. Finally, we recall symmetric encryption schemes and revisit the hybrid encryption paradigm in the setting of ciphertext pseudorandomness under parameter subversion attacks.

3.1 Public-Key Encryption Schemes

Below we give a syntax for public-key encryption schemes. It follows [23], but uses slightly different notation and includes an additional algorithm setting up global parameters to be utilized by all users. We then formalize a novel security requirement of pseudorandomness of ciphertexts under parameter subversion attacks (CPR-PSA), which says that even if the parameters of the scheme are controlled by the adversary, ciphertexts obtained under any public key are indistinguishable from random elements of the ciphertext space, which depends only on the security parameter, the message length and the global parameters. We then recall two existing requirements of public-key encryption schemes adapting them to the setting of parameter subversion attacks. The first is the well-known notion of indistinguishability of encryptions [31], the second, from [6, 1], is that ciphertexts under different public keys are indistinguishable, which they called anonymity or key hiding and we call public-key hiding. In Proposition 1 we show that the first requirement implies the other two, allowing us to focus on it subsequently. We model the possibility of subverted parameters by having the adversary provide the parameters, which are used in the security games.

Public-Key Encryption. A *public-key encryption scheme* (PKE) PE specifies the following. Parameter generation algorithm $\mathsf{PE.P}$ takes input 1^k, where $k \in \mathbb{N}$ is the security parameter, and returns global parameters π. Key-generation algorithm $\mathsf{PE.G}$ takes input $1^k, \pi$ and returns a tuple (pk, sk) consisting of the public (encryption) key pk and matching secret (decryption) key sk. $\mathsf{PE.CS}$ associates to k, π and message length $m \in \mathbb{N}$ a finite set $\mathsf{PE.CS}(k, \pi, m)$ that is the *ciphertext space* of PE. Encryption algorithm $\mathsf{PE.E}$ takes $1^k, \pi, pk$ and a message $M \in \{0,1\}^*$ and returns a ciphertext $c \in \mathsf{PE.CS}(k, \pi, |M|)$. Deterministic decryption algorithm $\mathsf{PE.D}$ takes $1^k, \pi, sk$ and a ciphertext c and returns either a message $M \in \{0,1\}^*$ or the special symbol $\perp$ indicating failure. The correctness condition requires that for all $k \in \mathbb{N}$, all $\pi \in [\mathsf{PE.P}(1^k)]$, all $(pk, sk) \in [\mathsf{PE.G}(1^k, \pi)]$ and all $M \in \{0,1\}^*$ we have $\Pr\big[\mathsf{PE.D}(1^k, \pi, sk, c) = M\big] \geq 1 - \mathsf{PE.de}(k)$, where the probability is over $c \leftarrow\!\!\$\ \mathsf{PE.E}(1^k, \pi, pk, M)$ and $\mathsf{PE.de} : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is the *decryption error* of PE. Our PKEs will be in the ROM [13], which means the encryption and decryption algorithms have access to a random oracle specified in the security games. Correctness must then hold for all choices of the random oracle. We say a PKE is *parameter-free* if $\mathsf{PE.P}$ returns ε on every input 1^k.

Ciphertext pseudorandomness. Consider game $\mathbf{G}^{\text{cpr-psa}}_{\mathsf{PE},\mathcal{A}}(k)$ of Figure 2 associated to PKE PE, adversary $\mathcal{A}$ and security parameter k, and let

$$\mathbf{Adv}^{\text{cpr-psa}}_{\mathsf{PE},\mathcal{A}}(k) = 2\Pr[\mathbf{G}^{\text{cpr-psa}}_{\mathsf{PE},\mathcal{A}}(k)] - 1 .$$

We say that PE has pseudorandom ciphertexts under parameter subversion attacks (also called CPR-PSA) if the function $\mathbf{Adv}^{\text{cpr-psa}}_{\mathsf{PE},\mathcal{A}}(\cdot)$ is negligible for every $\mathcal{A}$. In the game, b is a challenge bit. When $b = 1$, the challenge ciphertext c^* is an encryption of a message of the adversary's choice, but if $b = 0$ it is chosen at random from the ciphertext space. Given the public key and challenge ciphertext, the adversary outputs a guess b' and wins if b' equals b, the game returning true in this case and false otherwise. The adversary has access to an oracle Init,

```
Games $\mathbf{G}^{\text{cpr-psa}}_{\mathsf{PE},\mathcal{A}}(k), \mathbf{G}^{\text{ind-psa}}_{\mathsf{PE},\mathcal{A}}(k), \mathbf{G}^{\text{pkh-psa}}_{\mathsf{PE},\mathcal{A}}(k)$
$c^* \leftarrow \bot$
$b \leftarrow\$ \{0,1\}$
$b' \leftarrow\$ \mathcal{A}^{\textsc{Init},\textsc{Enc},\textsc{Dec},\mathrm{RO}}(1^k)$
Return $(b = b')$

$\mathrm{RO}(x, m)$ // $\mathbf{G}^{\text{cpr-psa}}_{\mathsf{PE},\mathcal{A}}, \mathbf{G}^{\text{ind-psa}}_{\mathsf{PE},\mathcal{A}}, \mathbf{G}^{\text{pkh-psa}}_{\mathsf{PE},\mathcal{A}}$
If $(T[x, m] = \bot)$
   then $T[x, m] \leftarrow\$ \{0,1\}^m$
Return $T[x, m]$

$\textsc{Enc}(M)$ // $\mathbf{G}^{\text{cpr-psa}}_{\mathsf{PE},\mathcal{A}}$
If $(pk = \bot)$ then return $\bot$
If $(b = 0)$ then $c^* \leftarrow\$ \mathsf{PE.CS}(k, \pi, |M|)$
Else $c^* \leftarrow\$ \mathsf{PE.E}^{\mathrm{RO}}(1^k, \pi, pk, M)$
Return $c^*$

$\textsc{Enc}(M_0, M_1)$ // $\mathbf{G}^{\text{ind-psa}}_{\mathsf{PE},\mathcal{A}}$
If $(pk = \bot)$ then return $\bot$
If $(|M_0| \neq |M_1|)$ then return $\bot$
$c^* \leftarrow\$ \mathsf{PE.E}^{\mathrm{RO}}(1^k, \pi, pk, M_b)$
Return $c^*$

$\textsc{Enc}(M)$ // $\mathbf{G}^{\text{pkh-psa}}_{\mathsf{PE},\mathcal{A}}$
If $(pk_0 = \bot \vee pk_1 = \bot)$
   return $\bot$
$c^* \leftarrow\$ \mathsf{PE.E}^{\mathrm{RO}}(1^k, \pi, pk_b, M)$
Return $c^*$

$\textsc{Init}(\pi)$ // $\mathbf{G}^{\text{cpr-psa}}_{\mathsf{PE},\mathcal{A}}, \mathbf{G}^{\text{ind-psa}}_{\mathsf{PE},\mathcal{A}}$
$(pk, sk) \leftarrow\$ \mathsf{PE.G}(1^k, \pi)$
Return $pk$

$\textsc{Init}(\pi)$ // $\mathbf{G}^{\text{pkh-psa}}_{\mathsf{PE},\mathcal{A}}$
$(pk_0, sk_0) \leftarrow\$ \mathsf{PE.G}(1^k, \pi)$
$(pk_1, sk_1) \leftarrow\$ \mathsf{PE.G}(1^k, \pi)$
If $(pk_0 = \bot \vee pk_1 = \bot)$
   return $\bot$
Return $(pk_0, pk_1)$

$\textsc{Dec}(c)$ // $\mathbf{G}^{\text{cpr-psa}}_{\mathsf{PE},\mathcal{A}}, \mathbf{G}^{\text{ind-psa}}_{\mathsf{PE},\mathcal{A}}$
If $(c = c^*)$ then return $\bot$
Else return $\mathsf{PE.D}^{\mathrm{RO}}(1^k, \pi, sk, c)$

$\textsc{Dec}(c)$ // $\mathbf{G}^{\text{pkh-psa}}_{\mathsf{PE},\mathcal{A}}$
If $(c = c^*)$ then return $\bot$
$M_0 \leftarrow \mathsf{PE.D}^{\mathrm{RO}}(1^k, \pi, sk_0, c)$
$M_1 \leftarrow \mathsf{PE.D}^{\mathrm{RO}}(1^k, \pi, sk_1, c)$
Return $(M_0, M_1)$
```

Fig. 2. Games defining security of PKEs. In each game the adversary is given access to oracles. The game, to which an oracle belongs, is indicated behind the oracle's name. In each game oracles Init and Enc may be queried only once. Further Init has to be queried before using any of the other oracles.

which sets up the public key using parameters of the adversary's choice, and an oracle Enc to generate the challenge ciphertext. Furthermore it has access to the random oracle and a decryption oracle crippled to not work on the challenge ciphertext. We require that the adversary queries the oracles Init and Enc only once. Furthermore Init has to be queried before using any of the other oracles.

Indistinguishability of encryptions. Consider game $\mathbf{G}^{\text{ind-psa}}_{\mathsf{PE},\mathcal{A}}(k)$ of Figure 2 associated to PKE PE, adversary $\mathcal{A}$ and security parameter k, and let

$$\mathbf{Adv}^{\text{ind-psa}}_{\mathsf{PE},\mathcal{A}}(k) = 2\Pr[\mathbf{G}^{\text{ind-psa}}_{\mathsf{PE},\mathcal{A}}(k)] - 1 .$$

We say that PE has indistinguishable encryptions under parameter subversion attacks (also called IND-PSA) if the function $\mathbf{Adv}^{\text{ind-psa}}_{\mathsf{PE},\mathcal{A}}(\cdot)$ is negligible for every $\mathcal{A}$. In the game, b is a challenge bit. The adversary has access to an oracle Init, which sets up the public key using parameters of the adversary's choice, and an oracle Enc, which receives as input two messages M_0, M_1 of the same length and outputs the challenge ciphertext c^*. When $b = 0$, the challenge ciphertext is an encryption of M_0, if $b = 1$ an encryption of M_1. Given the public key and challenge ciphertext, the adversary outputs a guess b' and wins if b' equals b, the game returning true in this case and false otherwise. Again, the adversary has access to the random oracle and a decryption oracle crippled to not work on the challenge ciphertext. We require that the adversary queries the oracles Init and Enc only once. Furthermore Init has to be queried before using any of the other oracles.

Public-key hiding. Consider game $\mathbf{G}^{\text{pkh-psa}}_{\mathsf{PE},\mathcal{A}}(k)$ of Figure 2 associated to PKE PE, adversary $\mathcal{A}$ and security parameter k, and let

$$\mathbf{Adv}^{\text{pkh-psa}}_{\mathsf{PE},\mathcal{A}}(k) = 2\Pr[\mathbf{G}^{\text{pkh-psa}}_{\mathsf{PE},\mathcal{A}}(k)] - 1 .$$

We say that PE is public-key hiding under parameter subversion attacks (also called PKH-PSA) if the function $\mathbf{Adv}^{\text{pkh-psa}}_{\mathsf{PE},\mathcal{A}}(\cdot)$ is negligible for every $\mathcal{A}$. In the game, b is a challenge bit. Unlike the prior games, two key pairs are generated, not one. The challenge ciphertext c^* is an encryption of a message of the adversary's choice under pk_b. Given the public keys and the challenge ciphertext, the adversary outputs a guess b' and wins if b' equals b. This time the crippled decryption oracle returns decryptions under both secret keys. The adversary sets up the public keys with its call to oracle Init, and an uses oracle Enc to generate the challenge ciphertext. Again we require that the adversary queries the oracles Init and Enc only once. Furthermore Init has to be queried before using any of the other oracles.

Relations. The following says that pseudorandomness of ciphertexts implies both indistinguishable encryptions and anonymity. We give both asymptotic and concrete statements of the results.

Proposition 1. *Let* PE *be a PKE that has pseudorandom ciphertexts under parameter subversion attacks. Then:*

1. PE *is* IND-PSA. *Concretely, given an adversary* $\mathcal{A}$ *the proof specifies an adversary* $\mathcal{B}_0$ *such that* $\mathbf{Adv}^{\text{ind-psa}}_{\mathsf{PE},\mathcal{A}}(k) \leq 2 \cdot \mathbf{Adv}^{\text{cpr-psa}}_{\mathsf{PE},\mathcal{B}_0}(k)$ *for every* $k \in \mathbb{N}$, *and* $\mathcal{B}_0$ *has the same running time and query counts as* $\mathcal{A}$.
2. PE *is* PKH-PSA. *Concretely, given an adversary* $\mathcal{A}$ *the proof specifies an adversary* $\mathcal{B}_1$ *such that* $\mathbf{Adv}^{\text{pkh-psa}}_{\mathsf{PE},\mathcal{A}}(k) \leq 2 \cdot \mathbf{Adv}^{\text{cpr-psa}}_{\mathsf{PE},\mathcal{B}_1}(k)$ *for every* $k \in \mathbb{N}$, *and* $\mathcal{B}_0$ *has the same running time and query counts as* $\mathcal{A}$.

The proof of the proposition can be found in the full version of this paper [4].

3.2 Key Encapsulation Mechanisms

Below we first give a syntax for key encapsulation mechanisms. It follows [23] but with notation a bit different and including an additional algorithm setting up global parameters to be utilized by all users. As for public-key encryption schemes we formalize the security requirement of pseudorandomness of ciphertexts under

```
Game $\mathbf{G}^{\text{cpr-psa}}_{\mathsf{KE},\mathcal{A}}(k)$
$b \leftarrow\$ \{0,1\}$
$b' \leftarrow\$ \mathcal{A}^{\textsc{Init},\textsc{Dec},\text{RO}}(1^k)$
Return $(b = b')$

Game $\mathbf{G}^{\text{ind-psa}}_{\mathsf{KE},\mathcal{A}}(k)$
$b \leftarrow\$ \{0,1\}$
$b' \leftarrow\$ \mathcal{A}^{\textsc{Init},\textsc{Dec},\text{RO}}(1^k)$
Return $(b = b')$

Game $\mathbf{G}^{\text{pkh-psa}}_{\mathsf{KE},\mathcal{A}}(k)$
$b \leftarrow\$ \{0,1\}$
$b' \leftarrow\$ \mathcal{A}^{\textsc{Init},\textsc{Dec},\text{RO}}(1^k)$
Return $(b = b')$

$\textsc{Dec}(c)$ // $\mathbf{G}^{\text{cpr-psa}}_{\mathsf{KE},\mathcal{A}}, \mathbf{G}^{\text{ind-psa}}_{\mathsf{KE},\mathcal{A}}$
If $(c = c^*)$ then return $\perp$
$K \leftarrow \mathsf{KE.D}^{\text{RO}}(1^k, \pi, sk, c)$
Return $K$

$\textsc{Dec}(c)$ // $\mathbf{G}^{\text{pkh-psa}}_{\mathsf{KE},\mathcal{A}}$
If $(c = c^*)$ then return $\perp$
$K_0 \leftarrow \mathsf{KE.D}^{\text{RO}}(1^k, \pi, sk_0, c)$
$K_1 \leftarrow \mathsf{KE.D}^{\text{RO}}(1^k, \pi, sk_1, c)$
Return $(K_0, K_1)$

$\text{RO}(x, m)$ // $\mathbf{G}^{\text{cpr-psa}}_{\mathsf{KE},\mathcal{A}}, \mathbf{G}^{\text{ind-psa}}_{\mathsf{KE},\mathcal{A}}, \mathbf{G}^{\text{pkh-psa}}_{\mathsf{KE},\mathcal{A}}$
If $(T[x, m] = \perp)$
    then $T[x, m] \leftarrow\$ \{0,1\}^m$
Return $T[x, m]$

$\textsc{Init}(\pi)$ // $\mathbf{G}^{\text{cpr-psa}}_{\mathsf{KE},\mathcal{A}}$
$(pk, sk) \leftarrow\$ \mathsf{KE.G}(1^k, \pi)$
If $(pk = \perp)$ then return $\perp$
If $(b = 1)$ then $(K^*, c^*) \leftarrow\$ \mathsf{KE.E}^{\text{RO}}(1^k, \pi, pk)$
Else $K^* \leftarrow\$ \mathsf{KE.KS}(k)$
$c^* \leftarrow\$ \mathsf{KE.CS}(k, \pi)$
Return $(pk, K^*, c^*)$

$\textsc{Init}(\pi)$ // $\mathbf{G}^{\text{ind-psa}}_{\mathsf{KE},\mathcal{A}}$
$(pk, sk) \leftarrow\$ \mathsf{KE.G}(1^k, \pi)$
If $(pk = \perp)$ then return $\perp$
$(K^*, c^*) \leftarrow\$ \mathsf{KE.E}^{\text{RO}}(1^k, \pi, pk)$
If $(b = 0)$ then $K^* \leftarrow\$ \mathsf{KE.KS}(k)$
Return $(pk, K^*, c^*)$

$\textsc{Init}(\pi)$ // $\mathbf{G}^{\text{pkh-psa}}_{\mathsf{KE},\mathcal{A}}$
$(pk_0, sk_0) \leftarrow\$ \mathsf{KE.G}(1^k, \pi)$
$(pk_1, sk_1) \leftarrow\$ \mathsf{KE.G}(1^k, \pi)$
If $(pk_0 = \perp \vee pk_1 = \perp)$ then return $\perp$
$(K^*, c^*) \leftarrow\$ \mathsf{KE.E}^{\text{RO}}(1^k, \pi, pk_b)$
Return $(pk_0, pk_1, K^*, c^*)$
```

Fig. 3. Games defining security of key encapsulation mechanism KE. In each game the adversary is given access to oracles. The game, to which an oracle belongs, is indicated behind the oracle's name. In each game oracle Init must be queried only once, which has to be done before using any of the other oracles.

parameter subversion attacks (CPR-PSA). We then adapt the two existing KEM requirements of indistinguishability of encryptions [23] and public-key hiding [6, 1] to the setting of parameter subversion attacks. In Proposition 2 we show that —as in the case of public-key encryption— the first requirement implies the other two. We furthermore define a new security requirement called well-distributedness of ciphertexts, which is necessary to achieve CPR-PSA in the hybrid PKE construction. It states that key-ciphertext pairs generated using the KEM's encapsulation algorithm are indistinguishable from choosing a ciphertext at random and then computing its decapsulation.

KEMs. A *key encapsulation mechanism* (KEM) KE specifies the following. Parameter generation algorithm KE.P takes input 1^k, where $k \in \mathbb{N}$ is the security parameter, and returns global parameters π. Key-generation algorithm KE.G takes input $1^k, \pi$ and returns a tuple (pk, sk) consisting of the public (encryption) key pk and matching secret (decryption) key sk. KE.KS associates to k a finite set $\mathsf{KE.KS}(k)$ only depending on the security parameter that is the *key space* of KE. $\mathsf{KE.CS}$ associates to k and parameters π a finite set $\mathsf{KE.CS}(k,\pi)$ that is the *ciphertext space* of KE. Encapsulation algorithm $\mathsf{KE.E}$ takes $1^k, \pi, pk$ and returns (K, c) where $K \in \mathsf{KE.KS}(k)$ is the *encapsulated key* and $c \in \mathsf{KE.CS}(k,\pi)$ is a ciphertext encapsulating K. Deterministic decapsulation algorithm $\mathsf{KE.D}$ takes $1^k, \pi, sk$ and a ciphertext c and returns either a key $K \in \mathsf{KE.KS}(k)$ or the special symbol $\bot$ indicating failure. The correctness condition requires that for all $k \in \mathbb{N}$, all $\pi \in [\mathsf{KE.P}(1^k)]$ and all $(pk, sk) \in [\mathsf{KE.G}(1^k, \pi)]$ we have $\Pr\big[\mathsf{KE.D}(1^k, \pi, sk, c) = K\big] \geq 1 - \mathsf{KE.de}(k)$, where the probability is over $(K, c) \leftarrow_{\$} \mathsf{KE.E}(1^k, \pi, pk)$ and $\mathsf{KE.de} : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is the *decryption error* of KE. Our KEMs will be in the ROM [13], which means the encapsulation and decapsulation algorithms have access to a random oracle specified in the security games. Correctness must then hold for all choices of the random oracle. We say a KEM is *parameter-free* if $\mathsf{KE.P}$ returns ε on every input 1^k.

Ciphertext pseudorandomness. Consider game $\mathbf{G}^{\text{cpr-psa}}_{\mathsf{KE},\mathcal{A}}(k)$ of Figure 3 associated to KEM KE, adversary $\mathcal{A}$ and security parameter k, and let

$$\mathbf{Adv}^{\text{cpr-psa}}_{\mathsf{KE},\mathcal{A}}(k) = 2\Pr[\mathbf{G}^{\text{cpr-psa}}_{\mathsf{KE},\mathcal{A}}(k)] - 1 .$$

We say that KE has pseudorandom ciphertexts under parameter subversion attacks (also called CPR-PSA) if the function $\mathbf{Adv}^{\text{cpr-psa}}_{\mathsf{KE},\mathcal{A}}(\cdot)$ is negligible for every $\mathcal{A}$. In the game, b is a challenge bit. When $b = 1$, the challenge key K^* and ciphertext c^* are generated via the encapsulation algorithm, but if $b = 0$ they are chosen at random, from the key space and ciphertext space, respectively. Given the public key, challenge key and challenge ciphertext, the adversary outputs a guess b' and wins if b' equals b, the game returning true in this case and false otherwise. The adversary has access to an oracle Init, which sets up the challenge. We require that the adversary queries Init before using any of the other oracles and that it queries Init only once. Further the adversary has access to an oracle for decapsulation under sk, crippled to not work when invoked on the challenge ciphertext. It, and the encapsulation and decapsulation algorithms, have access to the random oracle RO. The parameters used in the game are provided by the adversary via its call to Init.

Indistinguishability of encapsulated keys from random. Consider game $\mathbf{G}^{\text{ind-psa}}_{\mathsf{KE},\mathcal{A}}(k)$ of Figure 3 associated to KEM KE, adversary $\mathcal{A}$ and security parameter k, and let

$$\mathbf{Adv}^{\text{ind-psa}}_{\mathsf{KE},\mathcal{A}}(k) = 2\Pr[\mathbf{G}^{\text{ind-psa}}_{\mathsf{KE},\mathcal{A}}(k)] - 1 .$$

We say that KE has encapsulated keys indistinguishable from random under parameter subversion attacks (also called IND-PSA) if the function $\mathbf{Adv}^{\text{ind-psa}}_{\mathsf{KE},\mathcal{A}}(\cdot)$ is negligible for every $\mathcal{A}$. In the game, b is a challenge bit. When $b = 1$, the challenge key K^* and ciphertext c^* are generated via the encapsulation algorithm, while if $b = 0$ the key is switched to one drawn randomly from the key space, the ciphertext remaining real. Given the public key, challenge key and challenge ciphertext, the adversary outputs a guess b' and wins if b' equals b. Again the adversary has access to a crippled decapsulation oracle and the random oracle and provides the parameters used in the game via his call to the oracle Init, which has to be queried before using any of the other oracles.

Public-key hiding. Consider game $\mathbf{G}^{\text{pkh-psa}}_{\mathsf{KE},\mathcal{A}}(k)$ of Figure 3 associated to KEM KE, adversary $\mathcal{A}$ and security parameter k, and let

$$\mathbf{Adv}^{\text{pkh-psa}}_{\mathsf{KE},\mathcal{A}}(k) = 2\Pr[\mathbf{G}^{\text{pkh-psa}}_{\mathsf{KE},\mathcal{A}}(k)] - 1 \ .$$

We say that KE is public-key hiding under parameter subversion attacks (also called PKH-PSA) if the function $\mathbf{Adv}^{\text{pkh-psa}}_{\mathsf{KE},\mathcal{A}}(\cdot)$ is negligible for every $\mathcal{A}$. In the game, b is a challenge bit. Unlike the prior games, two key pairs are generated, not one. The challenge key K^* and ciphertext c^* are generated via the encapsulation algorithm under pk_b. Given the public keys, challenge key and challenge ciphertext, the adversary outputs a guess b' and wins if b' equals b. This time the crippled decapsulation oracle returns decapsulations under both secret keys. Again the adversary provides the parameters to be used in the game via his single call to the oracle Init, which has to be queried before using any of the other oracles.

Relations. The following says that in the parameter subversion setting CPR-PSA implies both IND-PSA and PKH-PSA. We give both the asymptotic and concrete statements of the results.

Proposition 2. *Let* KE *be a KEM that has pseudorandom ciphertexts under parameter subversion attacks. Then:*

1. KE *is* IND-PSA. *Concretely, given an adversary* $\mathcal{A}$ *the proof specifies an adversary* $\mathcal{B}$ *such that* $\mathbf{Adv}^{\text{ind-psa}}_{\mathsf{KE},\mathcal{A}}(k) \le 2 \cdot \mathbf{Adv}^{\text{cpr-psa}}_{\mathsf{KE},\mathcal{B}}(k)$ *for every* $k \in \mathbb{N}$*, and* $\mathcal{B}$ *has the same running time and query counts as* $\mathcal{A}$*.*
2. KE *is* PKH-PSA. *Concretely, given an adversary* $\mathcal{A}$ *the proof specifies an adversary* $\mathcal{B}$ *such that* $\mathbf{Adv}^{\text{pkh-psa}}_{\mathsf{KE},\mathcal{A}}(k) \le 2 \cdot \mathbf{Adv}^{\text{cpr-psa}}_{\mathsf{KE},\mathcal{B}}(k)$ *for every* $k \in \mathbb{N}$*, and* $\mathcal{B}$ *has the same running time and query counts as* $\mathcal{A}$*.*

The proof of the proposition can be found in the full version of this paper [4].

Well-distributed ciphertexts. Consider game $\mathbf{G}^{\text{wdc-psa}}_{\mathsf{KE},\mathcal{A}}(k)$ of Figure 4 associated to KEM KE, adversary $\mathcal{A}$ and security parameter k, and let

$$\mathbf{Adv}^{\text{wdc-psa}}_{\mathsf{KE},\mathcal{A}}(k) = 2\Pr[\mathbf{G}^{\text{wdc-psa}}_{\mathsf{KE},\mathcal{A}}(k)] - 1.$$

We say KE has well distributed ciphertexts under parameter subversion attacks (also called WDC-PSA), if the function $\mathbf{Adv}^{\text{wdc-psa}}_{\mathsf{KE},\mathcal{A}}(\cdot)$ is negligible for every adversary $\mathcal{A}$. In the game b is a challenge bit. If b equals 1 the adversary as response to querying the initialization procedure, which may be done at most once, receives a key-ciphertext pair generated using $\mathsf{KE.E}$. If b equals 0 it receives a pair (c^*, K^*) generated by choosing c^* at random and then setting K^* to be the decapsulation of c^*. The adversary has access to a decryption oracle. We require that the adversary queries Init before querying any of the other oracles. Looking ahead, all of our instantiations achieve this notion statistically.

```
Game $\mathbf{G}^{\text{wdc-psa}}_{\mathsf{KE},\mathcal{A}}(k)$
$b \leftarrow\!\$ \{0,1\}$
$b' \leftarrow\!\$ \mathcal{A}^{\textsc{Init},\textsc{Dec},\text{RO}}(1^k)$
Return $(b = b')$

$\textsc{Init}(\pi)$
$(pk, sk) \leftarrow\!\$ \mathsf{KE.G}(1^k, \pi)$
If $(pk = \bot)$ then return $\bot$
If $(b = 1)$ then $(K^*, c^*) \leftarrow\!\$ \mathsf{KE.E}^{\text{RO}}(1^k, \pi, pk)$
Else $c^* \leftarrow\!\$ \mathsf{KE.CS}(k, \pi)$
$K^* \leftarrow \mathsf{KE.D}^{\text{RO}}(1^k, \pi, sk, c^*)$
Return $(pk, K^*, c^*)$

$\text{RO}(x, m)$
If $(T[x, m] = \bot)$
    then $T[x, m] \leftarrow\!\$ \{0,1\}^m$
Return $T[x, m]$

$\textsc{Dec}(c)$
If $(c = c^*)$ then return $\bot$
$K \leftarrow \mathsf{KE.D}^{\text{RO}}(1^k, \pi, sk, c)$
Return $K$
```

Fig. 4. Game defining well-distributedness of ciphertexts of KEs.

3.3 Symmetric Encryption

Below, we recall symmetric encryption. Our definition follows [23] but uses different notation. We further define the security notion of ciphertext pseudorandomness for symmetric key encryption.

One-Time symmetric-Key Encryption. A symmetric-key encryption scheme (SKE) specifies the following. $\mathsf{SE.KS}$ associates to security parameter k key space $\mathsf{SE.KS}(k)$. $\mathsf{SE.CS}$ associates to security parameter k and message length $m \in \mathbb{N}$ the ciphertext space $\mathsf{SE.CS}(k, m)$. Deterministic encryption algorithm $\mathsf{SE.E}$ takes as input 1^k, key $K \in \mathsf{SE.KS}(k)$ and a message $M \in \{0,1\}^*$ and returns ciphertext $c \in \mathsf{SE.CS}(k, |M|)$. Deterministic decryption algorithm $\mathsf{SE.D}$ on input $1^k, K \in \mathsf{SE.KS}(k), c \in \mathsf{SE.CS}(k, m)$ returns either a message $M \in \{0,1\}^m$ or the special symbol $\bot$ indicating failure. For correctness we require that $M = \mathsf{SE.D}(1^k, K, c)$ for all k, all $K \in \mathsf{SE.KS}(k)$ and all $M \in \{0,1\}^*$, where $c \leftarrow \mathsf{SE.E}(1^k, K, M)$.

One-time security Consider game $\mathbf{G}^{\text{cpr}}_{\mathsf{SE},\mathcal{A}}(k)$ of Figure 5 associated to SKE SE, adversary $\mathcal{A}$ and security parameter k, and let

$$\mathbf{Adv}^{\text{cpr}}_{\mathsf{SE},\mathcal{A}}(k) = 2\Pr[\mathbf{G}^{\text{cpr}}_{\mathsf{SE},\mathcal{A}}(k)] - 1\ .$$

We say that SE has pseudorandom ciphertexts (also called CPR) if the function $\mathbf{Adv}^{\text{cpr}}_{\mathsf{SE},\mathcal{A}}(\cdot)$ is negligible for every $\mathcal{A}$. We require that $\textsc{Enc}$ is queried at most once.

3.4 PKE from Key Encapsulation and Symmetric-Key Encryption

Below, we analyze hybrid encryption in the setting of parameter subversion. Formally we give a transform **KEMToPE** that associates to KEM KE and symmetric-key encryption scheme SE a public-key encryption scheme PE. The construction essentially is the hybrid encryption scheme of [23] including an additional parameter generation algorithm. The scheme's parameter generation, key

```
Game $\mathbf{G}^{\mathrm{cpr}}_{\mathsf{SE},\mathcal{A}}(k)$
$b \leftarrow\!\!\$ \{0,1\}$
$K \leftarrow\!\!\$ \mathsf{SE.KS}(k)$
$b' \leftarrow \mathcal{A}^{\mathrm{Enc},\mathrm{Dec}}(1^k)$
Return $(b = b')$

$\mathrm{Enc}(M)$
If $(b = 0)$ then $c^* \leftarrow\!\!\$ \mathsf{SE.CS}(k, |M|)$
Else $c^* \leftarrow \mathsf{SE.E}(1^k, K, M)$
Return $c^*$

$\mathrm{Dec}(c)$
If $(c = c^*)$ then return $\perp$
Else return $\mathsf{SE.D}(1^k, K, c)$
```

Fig. 5. Game defining one-time security notions of SKEs.

```
$\mathsf{PE.P}(1^k)$
$\pi \leftarrow\!\!\$ \mathsf{KE.P}(1^k)$
Return $\pi$

$\mathsf{PE.G}(1^k, \pi)$
$(pk, sk) \leftarrow\!\!\$ \mathsf{KE.G}(1^k, \pi)$
Return $(pk, sk)$

$\mathsf{PE.E}(1^k, \pi, pk, M)$
$(K, c_1) \leftarrow\!\!\$ \mathsf{KE.E}^{\mathrm{RO}}(1^k, \pi, pk)$
$c_2 \leftarrow \mathsf{SE.E}(1^k, K, M)$
Return $(c_1, c_2)$

$\mathsf{PE.D}(1^k, \pi, sk, c)$
$(c_1, c_2) \leftarrow c$
$K \leftarrow \mathsf{KE.D}^{\mathrm{RO}}(1^k, \pi, sk, c_1)$
$M \leftarrow \mathsf{SE.D}(1^k, K, c_2)$
Return $M$
```

Fig. 6. PKE **KEMToPE**[KE, SE] associated to KEM KE and SE SE.

generation encryption and decryption algorithms are in Figure 6. PE's ciphertext space is given by $\mathsf{PE.CS}(k, \pi, m) = \mathsf{KE.CS}(k, \pi) \times \mathsf{SE.CS}(k, m)$. It is easy to verify that PE has decryption error $\mathsf{PE.de}(k) = \mathsf{KE.de}(k)$. The following essentially states that hybrid encryption also works in setting of ciphertext pseudorandomness under parameter subversion attacks, i.e., combining a KEM that is both CPR-PSA and WDC-PSA with a SKE that is CPR yields a CPR-PSA PKE, where the well-distributedness of the KEM's ciphertext is necessary to correctly simulate the decryption oracle in the CPR-PSA game with respect to PE.

Proposition 3. *Let* KE *a KEM and* SE *a SE such that* $\mathsf{KE.KS}(k) = \mathsf{SE.KS}(k)$ *for all* $k \in \mathbb{N}$. *Let* $\mathsf{PE} = \mathbf{KEMToPE}[\mathsf{KE}, \mathsf{SE}]$ *be the PKE associated to* KE *and* SE. *If* KE *is* CPR-PSA *and* WDC-PSA *and if* SE *is* CPR *then* PE *is* CPR-PSA *Concretely, given adversary* $\mathcal{A}$ *against* $\mathbf{G}^{\mathrm{cpr\text{-}psa}}_{\mathsf{PE},\mathcal{A}}(k)$, *there exist adversaries* $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$ *having the same running time and query count as* $\mathcal{A}$, *which satisfy*

$$\mathbf{Adv}^{\mathrm{cpr\text{-}psa}}_{\mathsf{PE},\mathcal{A}}(k) \leq 2\,\mathbf{Adv}^{\mathrm{cpr\text{-}psa}}_{\mathsf{KE},\mathcal{B}_1}(k) + \mathbf{Adv}^{\mathrm{wdc\text{-}psa}}_{\mathsf{KE},\mathcal{B}_2}(k) + \mathbf{Adv}^{\mathrm{cpr}}_{\mathsf{SE},\mathcal{B}_3}(k) + \mathsf{KE.de}(k)\ .$$

The proof of the proposition can be found in the full version of this paper [4].

```
Game $\mathbf{G}^{\text{epr-psa}}_{\mathsf{EG},\mathcal{A}}(k)$
$b \leftarrow\$ \{0,1\}$
$b' \leftarrow\$ \mathcal{A}^{\text{INIT}}(1^k)$
Return $(b = b')$

INIT$(\pi)$
$G \leftarrow\$ \mathsf{EG.G}(1^k, \pi)$
If $(G = \bot)$ then return $\bot$
$(\langle \mathbb{G} \rangle, n, g) \leftarrow G$
If $(b = 1)$ then
    $y \leftarrow\$ \mathsf{EG.S}(1^k, \pi, G)$
    $c \leftarrow\$ \mathsf{EG.E}(1^k, \pi, G, g^y)$
Else $c \leftarrow\$ \mathsf{EG.ES}(k, \pi)$
Return $(G, c)$
```

Fig. 7. Game defining embedding pseudorandomness of eeg family EG.

4 KEMs from Efficiently Embeddable Group Families

In this section we define efficiently embeddable group families (eeg). We define the security notion of pseudorandom embeddings under parameter subversion attacks (EPR-PSA) and adapt the strong computational Diffie-Hellman problem (sCDH-PSA) to the setting of efficiently embeddable group families and parameter subversion. Further we give a generic constructions of key encapsulation mechanisms from eeg families. It achieves security assuming the eeg family is sCDH-PSA and EPR-PSA.

4.1 Efficiently Embeddable Group families

EFFICIENTLY EMBEDDABLE GROUP FAMILIES. An embeddable group family EG specifies the following. Parameter generation algorithm $\mathsf{EG.P}$ takes as input 1^k, where $k \in \mathbb{N}$ is the security parameter, and returns parameters π. Group generation algorithm $\mathsf{EG.G}$ on input $1^k, \pi$ returns a tuple $G = (\langle \mathbb{G} \rangle, n, g)$, where $\langle \mathbb{G} \rangle$ is a description of a cyclic group $\mathbb{G}$ of order n, and g is a generator of $\mathbb{G}$. $\mathsf{EG.ES}$ associates to k a finite set $\mathsf{EG.ES}(k, \pi)$ called the embedding space that is only dependent on k and π. Sampling algorithm $\mathsf{EG.S}$ on input of $1^k, \pi$ and $G \in [\mathsf{EG.G}(1^k, \pi)]$ outputs $y \in \mathbb{Z}_n$. (Not necessarily uniformly distributed.) Embedding algorithm $\mathsf{EG.E}$ receives as input 1^k, π, $G \in [\mathsf{EG.G}(1^k, \pi)]$ and $h \in \mathbb{G}$ and returns an element $c \in \mathsf{EG.ES}(k, \pi)$. Deterministic inversion algorithm $\mathsf{EG.I}$ on input of 1^k, π, $G \in [\mathsf{EG.G}(1^k, \pi)]$ and $c \in \mathsf{EG.ES}(k, \pi)$ returns an element of $\mathbb{G}$. The correctness condition requires that for all $k \in \mathbb{N}$, all $\pi \in \mathsf{EG.P}(1^k)$ and all $G \in [\mathsf{EG.G}(1^k, \pi)]$ we have $\Pr\left[\mathsf{EG.I}(1^k, \pi, G, h) = g^y\right] \geq 1 - \mathsf{EG.ie}(k)$, where the probability is over $y \leftarrow\$ \mathsf{EG.S}(1^k, \pi, G)$ and $h \leftarrow\$ \mathsf{EG.E}(1^k, \pi, G, g^y)$, and $\mathsf{EG.ie} : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is the *inversion error* of EG. If $\mathsf{EG.P}$ returns ε on every input 1^k, i. e. if no parameters are used, we say that EG is *parameter-free.*

EMBEDDING PSEUDORANDOMNESS. Consider game $\mathbf{G}^{\text{epr-psa}}_{\mathsf{EG},\mathcal{A}}(k)$ of Figure 7 associated to eeg family EG, adversary $\mathcal{A}$ and security parameter k. Let

$$\mathbf{Adv}^{\text{epr-psa}}_{\mathsf{EG},\mathcal{A}}(k) = 2\Pr[\mathbf{G}^{\text{epr-psa}}_{\mathsf{EG},\mathcal{A}}(k)] - 1.$$

```
Game $\mathbf{G}^{\text{scdh-psa}}_{\mathsf{EG},\mathcal{A}}(k)$
    $Z \leftarrow\!\!\$\ \mathcal{A}^{\textsc{Init},\textsc{ddh}}(1^k)$
    Return $(Z = g^{xy} \wedge G \neq \bot)$

$\textsc{ddh}(\tilde{Y}, \tilde{Z})$
    Return $(\tilde{Y}^x = \tilde{Z})$

$\textsc{Init}(\pi)$
    $G \leftarrow\!\!\$\ \mathsf{EG.G}(1^k, \pi)$
    If $(G = \bot)$ then return $\bot$
    $(\langle\mathbb{G}\rangle, n, g) \leftarrow G$
    $x \leftarrow\!\!\$\ \mathbb{Z}_n$
    $y \leftarrow\!\!\$\ \mathsf{EG.S}(1^k, \pi, G)$
    Return $(G, g^x, g^y)$
```

Fig. 8. Experimentfor the strong computational Diffie-Hellman problem with respect to eeg family EG. Oracle Init may be queried only once and has to be queried before using oracle ddh.

We say that EG has pseudorandom embeddings under parameter subversion attacks (also called EPR-PSA) if the function $\mathbf{Adv}^{\text{epr-psa}}_{\mathsf{EG},\mathcal{A},\cdot}$ is negligible for every $\mathcal{A}$. In the game, b is a challenge bit. When $b = 1$, the challenge embedding c^* is generated by sampling an exponent using $\mathsf{EG.S}$ and embedding the group generator raised to the exponent with $\mathsf{EG.E}$. If $b = 0$ the adversary is given an embedding sampled uniformly from the embedding space. Given the group and the embedding, the adversary outputs a guess b' and wins if b' equals b. The parameters used in the game are provided by the adversary making a single call to the oracle Init. All of our instantiations sample exponents such that the resulting embeddings are statistically close to uniform on $\mathsf{EG.ES}(k, \pi)$, and hence achieve this notion statistically.

Diffie-Hellman problem with respect to EG. The computational Diffie-Hellman problem for a cyclic group $\mathbb{G}$ of order n, which is generated by g, asks to compute g^{xy} given g^x and g^y, where $x, y \leftarrow\!\!\$\ \mathbb{Z}_n$. In the strong computational Diffie-Hellman problem introduced by Abdalla et al. in [2] the adversary additionally has access to an oracle, which may be used to check whether $Y^x = Z$ for group elements $Y, Z \in \mathbb{G}$. We provide a definition for the strong computational Diffie-Hellman problem with respect to eeg families EG, which allows parameter subversion. An additional difference is that y is not chosen uniformly from $\mathbb{Z}_n$ but instead sampled using $\mathsf{EG.S}$.

Thus, consider game $\mathbf{G}^{\text{scdh-psa}}_{\mathsf{EG},\mathcal{A}}(k)$ of Figure 8. The game is associated to eeg family EG, adversary $\mathcal{A}$ and security parameter k. The adversary has access to an oracle Init setting up a problem instance according to the parameters it is provided. Let

$$\mathbf{Adv}^{\text{scdh-psa}}_{\mathsf{EG},\mathcal{A}}(k) := \Pr\Big[\mathbf{G}^{\text{scdh-psa}}_{\mathsf{EG},\mathcal{A}}(k)\Big].$$

We say that the strong computational Diffie-Hellman problem under parameter subversion (also called sCDH-PSA) is hard with respect to EG if $\mathbf{Adv}^{\text{scdh-psa}}_{\mathsf{EG},\mathcal{A}}(\cdot)$ is negligible for every adversary $\mathcal{A}$.

```
$\mathsf{KE.G}_1(1^k, \pi)$
    $G \leftarrow_{\$} \mathsf{EG.G}(1^k, \pi)$
    If $(G = \bot)$ return $\bot$
    $(\langle \mathbb{G} \rangle, n, g) \leftarrow G$
    $x \leftarrow_{\$} \mathbb{Z}_n$; $X \leftarrow g^x$
    $pk \leftarrow (G, X)$
    $sk \leftarrow (G, x, pk)$
    Return $(pk, sk)$

$\mathsf{KE.E}_1^{\mathrm{RO}}(1^k, \pi, pk)$
    $(G, X) \leftarrow pk$
    $y \leftarrow_{\$} \mathsf{EG.S}(1^k, \pi, G)$
    $Y \leftarrow g^y$
    $c \leftarrow_{\$} \mathsf{EG.E}(1^k, \pi, G, Y)$
    $K \leftarrow \mathrm{RO}((pk, c, X^y), m(k))$
    Return $(K, c)$

$\mathsf{KE.D}_1^{\mathrm{RO}}(1^k, \pi, sk, c)$
    $(G, x, pk) \leftarrow sk$
    $Y \leftarrow \mathsf{EG.I}(1^k, \pi, G, c)$
    $K \leftarrow \mathrm{RO}((pk, c, Y^x), m(k))$
    Return $K$

$\mathsf{KE.P}_1(1^k)$
    $\pi \leftarrow_{\$} \mathsf{EG.P}(1^k)$
    Return $\pi$
```

Fig. 9. KEM $\mathsf{KE}_1 = \mathbf{eegToKE1}[\mathsf{EG}, m]$ built from eeg family EG and polynomial m via our transform. The KE has key space $\mathsf{KE.KS}(k) = \{0,1\}^{m(k)}$ and ciphertext space $\mathsf{KE.CS}(k, \pi) = \mathsf{EG.ES}(k, \pi)$.

4.2 Key Encapsulation from Efficiently Embeddable Group Families

In this section we give a generic construction of a key encapsulation mechanism from an eeg family EG. Its security is based on the strong Diffie-Hellman problem, i.e. if sCDH-PSA is hard with respect to EG, the KEM is IND-PSA. If additionally EG has pseudorandom embeddings, the KEM has pseudorandom and well-distributed ciphertexts. The construction is similar to the standard El Gamal based key encapsulation mechanism as for example used in [2, 23]. As an intermediate step in the proof that the construction is CPR-PSA we obtain that it is IND-PSA. The proof of this property follows the outlines of the proofs given in [2, 23]. Afterwards we use the pseudorandomness of the eeg family's embeddings to show, that our construction achieves pseudorandom and well-distributed ciphertexts.

Formally, we define a transform **eegToKE1** that associates to an eeg family EG and a polynomial $m : \mathbb{N} \to \mathbb{N}$ a KEM $\mathsf{KE} = \mathbf{eegToKE1}[\mathsf{EG}, m]$. The parameter generation, key generation, encryption and decryption algorithms of KE are in Figure 9. The construction is in the ROM, so that encryption and decryption invoke the RO oracle. The key space is $\mathsf{KE.KS}(k) = \{0,1\}^{m(k)}$. The ciphertext space $\mathsf{KE.CS}(k, \pi) = \mathsf{EG.ES}(k, \pi)$ is the embedding space of EG. It is easy to verify that $\mathsf{KE.de} = \mathsf{EG.ie}$, meaning the decryption error of the KEM equals the inversion error of the eeg family.

Security of the construction. The following says that if sCDH-PSA is hard with respect to eeg family EG then $\mathbf{eegToKE1}[\mathsf{EG}, m]$ has desirable security properties.

Theorem 4. *Let* $\mathsf{KE} = \mathbf{eegToKE1}[\mathsf{EG}, m]$ *be the KEM associated to eeg family* EG *and polynomial* $m : \mathbb{N} \to \mathbb{N}$ *as defined in Figure 9. Assume that* EG *is* EPR-PSA *and that* sCDH-PSA *is hard with respect to* EG*. Then*

(i) KE *has pseudorandom ciphertexts under parameter subversion attacks.*
(ii) KE *has well-distributed ciphertexts under parameter subversion attacks.*

Moreover, if EG *is parameter-free so is* KE*. Concretely, given an adversary* $\mathcal{A}$ *making at most* $q(k)$ *queries to* RO *the proof specifies adversaries* $\mathcal{B}_1$ *and* $\mathcal{B}_2$ *having the same running time as* $\mathcal{A}$ *satisfying*

$$\mathbf{Adv}^{\text{cpr-psa}}_{\mathsf{KE}}(\mathcal{A})(k) \leq \mathbf{Adv}^{\text{scdh-psa}}_{\mathsf{EG},\mathcal{B}_1}(k) + \mathbf{Adv}^{\text{epr-psa}}_{\mathsf{EG},\mathcal{B}_2}(k) \ ,$$

where $\mathcal{B}_2$ *makes at most* $q(k)$ *queries to* DDH*. Furthermore given an adversary* $\mathcal{A}'$ *the proof specifies an adversary* $\mathcal{B}'$ *having the same running time as* $\mathcal{A}'$ *such that,*

$$\mathbf{Adv}^{\text{wdc-psa}}_{\mathsf{KE},\mathcal{A}'}(k) \leq \mathbf{Adv}^{\text{epr-psa}}_{\mathsf{EG},\mathcal{B}'}(k) + \mathsf{EG.ie}(k) \ .$$

The proof of the theorem can be found in the full version of this paper [4]. In the full version of this paper [4] we also provide a transform **eegToKE2**, which achieves security under the weaker CDH-PSA assumption with respect to EG.

5 Efficiently Embeddable Group Families from Curve-Twist Pairs

In this section we give instantiations of eeg families based on elliptic curves. The main tool of the constructions is a bijection of [34] mapping points of an elliptic curve and its quadratic twist to an interval of integers. We first give a construction using parameters, the parameter being a prime p of length k serving as the modulus of the prime field the curves are defined over. The construction has embedding space $[2p+1]$. Since we assume, that the parameter shared by all users might be subject to subversion, security of this construction corresponds to the assumption that there exist no inherently bad choices for p, i.e. that for *any* sufficiently large prime p it is possible to find elliptic curves defined over $\mathbb{F}_p$ on which the strong computational Diffie-Hellman assumption holds.

As an alternative we also give parameter-free eeg-families whose security is based on the weaker assumption that for *random* k-bit prime p it is possible to find elliptic curves defined over $\mathbb{F}_p$, such that the strong computational Diffie-Hellman assumption holds. Since in this construction the modulus p is sampled along with the curve, it is no longer possible to use $[2p+1]$ as the embedding space of the eeg family. We propose two solutions to overcome this, one using rejection sampling to restrict the embedding space to the set $[2^k]$, the other one is based on a technique from [33] and expands the embedding space to $[2^{k+1}]$.

5.1 Elliptic Curves

Let $p \geq 5$ be prime and $\mathbb{F}_p$ a field of order p. An elliptic curve over $\mathbb{F}_p$ can be expressed in short Weierstrass form, that is as the set of projective solutions of an equation of the form

$$YZ^2 = X^3 + aXZ^2 + bZ^3,$$

where $a, b \in \mathbb{F}_p$ with $4a^3 + 27b^2 \neq 0$. We denote the elliptic curve generated by p, a, b by $E(p, a, b)$. $E(p, a, b)$ possesses exactly one point with Z-coordinate 0, the so called point at infinity $\mathcal{O} = (0 : 1 : 0)$. After normalizing by $Z = 1$ the curve's other points can be interpreted as the solutions $(x, y) \in \mathbb{F}_p^2$ of the affine equation $y^2 = x^3 + ax + b$. It is possible to establish an efficiently computable group law on $E(p, a, b)$ with $\mathcal{O}$ serving as the neutral element of the group. We use multiplicative notation for the group law to be consistent with the rest of the paper.

Twists of Elliptic Curves. In [34, section 4] Kaliski establishes the following one-to-one correspondence between two elliptic curves defined over $\mathbb{F}_p$ which are related by twisting and a set of integers.

Lemma 5. *Let $p \in \mathbb{N}_{\geq 5}$ be prime. Let $u \in \mathbb{Z}_p$ be a quadratic nonresidue modulo p and $a, b \in \mathbb{Z}_p$ such that $4a^3+27b^2 \neq 0$. Consider the elliptic curves $E_0 := E(p, a, b)$ and $E_1 := E(p, au^2, bu^3)$. Then $|E_0| + |E_1| = 2p + 2$. Furthermore, the functions $l_0 : E_0 \longrightarrow [2p + 2]$ and $l_1 : E_1 \longrightarrow [2p + 2]$ defined as*

$$l_0\left(P\right) = \begin{cases} p & \textit{if } P = \mathcal{O}_0 \\ (ux \bmod p) & \textit{if } (P = (x, y)) \wedge (0 \leq y \leq (p-1)/2) \\ (ux \bmod p) + p + 1 & \textit{if } (P = (x, y)) \wedge ((p-1)/2 < y) \end{cases},$$

$$l_1\left(P\right) = \begin{cases} 2p + 1 & \textit{if } P = \mathcal{O}_1 \\ x & \textit{if } (P = (x, y)) \wedge (0 < y \leq (p-1)/2) \\ x + p + 1 & \textit{if } (P = (x, y)) \wedge ((y = 0) \vee ((p-1)/2 < y)) \end{cases}$$

are injective with nonintersecting ranges, where $\mathcal{O}_0$ and $\mathcal{O}_1$ denote the neutral elements of E_0 and E_1 respectively.

Lemma 6. *The functions l_0 and l_1 can be efficiently inverted. That is, given $z \in [2p + 1]$, one can efficiently compute the unique $(P, \delta) \in E_0 \cup E_1 \times \{0, 1\}$ such that $l_\delta(P) = z$.*

The proof of the lemma can be found in the full version of this paper [4].

Definition 7. *A curve-twist generator* TGen *on input of security parameter 1^k and a k-bit prime p returns (G_0, G_1), where $G_0 = (\langle E_0 \rangle, n_0, g_0)$ and $G_1 = (\langle E_1 \rangle, n_1, g_1)$ are secure cyclic elliptic curves defined over the field $\mathbb{F}_p$. More precisely we require $E_0 := E(p, a, b)$ and $E_1 := E(p, au^2, bu^3)$ for $a, b \in \mathbb{F}_p$ such that $(4a^3 + 27b^2) \neq 0$ and quadratic nonresidue u. Furthermore we require that g_0 generates E_0 and g_1 generates E_1 as well as $|E_0| = n_0$, $|E_1| = n_1$ and* $\gcd(n_0, n_1) = 1$.

Generation of secure Twisted Elliptic Curves. There exist several proposals for properties an elliptic curve over a prime field $\mathbb{F}_p$ should have to be considered secure (e.g., [18, 27]). Firstly, the elliptic curve's order is required to be either the product of a big prime and a small cofactor — or preferably prime. Secondly, several conditions preventing the transfer of discrete logarithm problems on the curve to groups, where faster algorithms to compute discrete logarithms may be applied, should be fulfilled. Finally, for our applications we need both the elliptic curve and its quadratic twist to be secure, a property usually called twist security. For concreteness, we suggest to implement $\mathsf{TGen}(1^k, p)$ by sampling the necessary parameters a, b, u with rejection sampling such that the resulting curve $E(p, a, b)$ fulfills the three security requirement mentioned above. This way, TGen can be implemented quite efficiently[3] and furthermore, with overwhelming probability, the resulting curve fulfills all relevant security requirements from [18, 27] that are not covered by the three security properties explicitly mentioned above.

Computational problems associated to TGen. Let TGen a curve-twist generator. We give two versions of the strong computational Diffie-Hellman assumption with respect to TGen. In the first version the prime p on which TGen is invoked is chosen by the adversary, while in the second version p is sampled uniformly at random from all k-bit primes. For $d \in \{0, 1\}$ consider games $\mathbf{G}^{\mathrm{twist}_d\text{-cp-scdh}}_{\mathsf{TGen},\mathcal{A}}(\cdot)$ and $\mathbf{G}^{\mathrm{twist}_d\text{-up-scdh}}_{\mathsf{TGen},\mathcal{A}}(\cdot)$ of Figure 10. We define advantage functions

$$\mathbf{Adv}^{\mathrm{twist}_d\text{-cp-scdh}}_{\mathsf{TGen},\mathcal{A}}(k) = \Pr\left[\mathbf{G}^{\mathrm{twist}_d\text{-cp-scdh}}_{\mathsf{TGen},\mathcal{A}}(k)\right] ,$$
$$\mathbf{Adv}^{\mathrm{twist}_d\text{-up-scdh}}_{\mathsf{TGen},\mathcal{A}}(k) = \Pr\left[\mathbf{G}^{\mathrm{twist}_d\text{-up-scdh}}_{\mathsf{TGen},\mathcal{A}}(k)\right] .$$

Definition 8. *Let TGen be a curve-twist generator. We say the strong computational Diffie-Hellman assumption for chosen (uniform) primes holds with respect to curve-twist generator TGen, if both $\mathbf{Adv}^{\mathrm{twist}_0\text{-cp-scdh}}_{\mathsf{TGen},\mathcal{A}}(\cdot)$ and $\mathbf{Adv}^{\mathrm{twist}_1\text{-cp-scdh}}_{\mathsf{TGen},\mathcal{A}}(\cdot)$ (or $\mathbf{Adv}^{\mathrm{twist}_0\text{-up-scdh}}_{\mathsf{TGen},(P_k)_k,\mathcal{A}}(\cdot)$ and $\mathbf{Adv}^{\mathrm{twist}_1\text{-up-scdh}}_{\mathsf{TGen},(P_k)_k,\mathcal{A}}(\cdot)$ respectively) are negligible for all adversaries $\mathcal{A}$.*

5.2 An Eeg Family from Elliptic Curves

In [34] Kaliski implicitly gives an eeg family based on elliptic curves. The family is parameter-using, the parameter being a prime p serving as the modulus of the field the elliptic curves are defined over. The definition of eeg family $\mathsf{EG}_{\mathrm{twist}}$ may be found in Figure 11. Parameter generation algorithm $\mathsf{EG}_{\mathrm{twist}}.\mathsf{P}$ on input of security parameter 1^k returns a randomly sampled k-bit prime[4] p. Group generation algorithm $\mathsf{EG}_{\mathrm{twist}}.\mathsf{G}$ on input of parameter $\pi = p$ checks, whether p is indeed a prime of appropriate length, and —if so— runs a curve-twist

[3] In [29] Galbraith and McKee consider elliptic curves E chosen uniformly from the set of elliptic curves over a fixed prime field $\mathbb{F}_p$. They give a conjecture (together with some experimental evidence) for a lower bound on the probability of $|E|$ being prime. Using a similar technique [27] argue, that the probability of a uniformly chosen elliptic curve over a fixed prime field $\mathbb{F}_p$ to be both secure and twist secure is bounded from below by $0.5/\log^2(p)$. Since their definition of security of an elliptic curve includes primality of the curve order and since due to Lemma 5 the orders of curve and twist sum up to $2p + 2$, this in particular implies that the curve and its twist are cyclic and have coprime group order.

[4] In practice one would preferably instantiate $\mathsf{EG}_{\mathrm{twist}}$ with a standardized prime.

```
Game $\mathbf{G}^{\text{twist}_d\text{-cp-scdh}}_{\mathsf{TGen},\mathcal{A}}(k)$
    $Z \leftarrow_{\$} \mathcal{A}^{\text{Init},\text{ddh}}(1^k)$
    Return $(Z = g_d^{xy})$

ddh$(\tilde{Y}_d, \tilde{Z}_d)$
    If $\tilde{Y}_d \notin E_d \lor \tilde{Z}_d \notin E_d$
        return $\perp$
    Return $(\tilde{Y}_d{}^x = \tilde{Z}_d)$

Init$(\pi)$
    $p \leftarrow \pi$
    If $(p \notin \mathcal{P}_k)$ then return $\perp$
    $(G_0, G_1) \leftarrow_{\$} \mathsf{TGen}(1^k, p)$
    $(\langle E_d \rangle, n_d, g_d) \leftarrow G_d$
    $x \leftarrow_{\$} \mathbb{Z}_{n_d}$; $y \leftarrow_{\$} \mathbb{Z}_{n_d}$
    $X \leftarrow g_d^x$, $Y \leftarrow g_d^y$
    Return $(G_0, G_1, X, Y)$
```

```
Game $\mathbf{G}^{\text{twist}_d\text{-up-scdh}}_{\mathsf{TGen},\mathcal{A}}(k)$
    $Z \leftarrow_{\$} \mathcal{A}^{\text{Init},\text{ddh}}(1^k)$
    Return $(Z = g_d^{xy})$

ddh$(\tilde{Y}_d, \tilde{Z}_d)$
    If $(\tilde{Y}_d \notin E_d \lor \tilde{Z}_d \notin E_d)$
        return $\perp$
    Return $(\tilde{Y}_d{}^x = \tilde{Z}_d)$

Init
    $p \leftarrow_{\$} \mathcal{P}_k$
    $(G_0, G_1) \leftarrow_{\$} \mathsf{TGen}(1^k, p)$
    $(\langle E_d \rangle, n_d, g_d) \leftarrow G_d$
    $x \leftarrow_{\$} \mathbb{Z}_{n_d}$; $y \leftarrow_{\$} \mathbb{Z}_{n_d}$
    $X \leftarrow g_d^x$, $Y \leftarrow g_d^y$
    Return $(G_0, G_1, p, X, Y)$
```

Fig. 10. Experiments for the sCDH problem for chosen (uniform) primes with respect to $d \in \{0, 1\}$, adversary $\mathcal{A}$ and curve-twist generator TGen.

generator $\mathsf{TGen}(1^k, \pi)$ to obtain the description of two cyclic secure cyclic elliptic curves $G_0 = (\langle E_0 \rangle, n_0, g_0)$ and $G_1 = (\langle E_1 \rangle, n_1, g_1)$. Its output is $(\langle \mathbb{G} \rangle, n, g)$, where $\mathbb{G} \leftarrow E_0 \times E_1$ is the direct product of the two elliptic curves, $n \leftarrow n_0 \cdot n_1$ and $g \leftarrow (g_0, g_1)$. Here we assume that the description $\langle \mathbb{G} \rangle$ of $\mathbb{G}$ includes the values n_0 and n_1, which are used by $\mathsf{EG}_{\text{twist}}$'s other algorithms. Note that $|\mathbb{G}| = n$ and since n_0 and n_1 are coprime, g generates $\mathbb{G}$. Furthermore, if we regard E_0 and E_1 as subgroups of $\mathbb{G} = E_0 \times E_1$ in the natural way, we may rewrite the set $E_0 \cup E_1 \subseteq \mathbb{G}$ as

$$\begin{aligned} E_0 \cup E_1 &= \{(h_0, \mathcal{O}_1) \mid h_0 \in E_0\} \cup \{(\mathcal{O}_0, h_1) \mid h_1 \in E_1\} \\ &= \{(g_0, g_1)^y \mid y \in \mathbb{Z}_n \colon y \equiv 0 \mod n_0 \text{ or } y \equiv 0 \mod n_1\} \end{aligned}$$

Algorithm $\mathsf{EG}_{\text{twist}}.\mathsf{S}$ uses this property to efficiently sample $y \in \mathbb{Z}_n$ such that $g^y \sim U_{E_0 \cup E_1}$. It first samples $z \leftarrow_{\$} \mathbb{Z}_{2p+1}$. If $z < n_0$ it returns $\varphi_{\text{crt}}(z, 0)$. Else it returns $\varphi_{\text{crt}}(0, z - n_0 - 1)$. Here φ_{crt} denotes the canonical isomorphism $\varphi_{\text{crt}} : \mathbb{Z}_{n_0} \times \mathbb{Z}_{n_1} \to \mathbb{Z}_n$. As a result $y \leftarrow_{\$} \mathsf{EG}_{\text{twist}}.\mathsf{S}(1^k, G)$ satisfies $y \sim U_M$, where $M := \{y \in \mathbb{Z}_n \mid y \equiv 0 \mod n_0 \text{ or } y \equiv 0 \mod n_1\}$. Embedding algorithm $\mathsf{EG}_{\text{twist}}.\mathsf{E}$ receives as input 1^k, π, G and $h = (h_0, h_1) \in \mathbb{G}$. It first checks, whether h lies outside of the support $[\mathsf{EG}_{\text{twist}}.\mathsf{S}(1^k, \pi, G)]$ of the sampling algorithm, i.e. whether both $h_0 \neq \mathcal{O}_0$ and $h_1 \neq \mathcal{O}_1$. In this case the element is mapped to 0. If h is an element of $[\mathsf{EG}_{\text{twist}}.\mathsf{S}(1^k, \pi, G)]$, algorithm $\mathsf{EG}_{\text{twist}}.\mathsf{E}$ returns $l_0(h_0)$ if $h_1 = \mathcal{O}_1$, and $l_1(h_1)$ if $h_1 \neq \mathcal{O}_1$. Here $l_0 \colon E_0 \to [2p + 2]$ and $l_1 \colon E_1 \to [2p + 2]$ denote the maps of Lemma 5. By Lemma 5 the map $\mathsf{EG}_{\text{twist}}.\mathsf{E}(1^k, G, \cdot)|_{E_0 \cup E_1}$ is a

```
$\underline{\mathsf{EG}_{\mathrm{twist}}.\mathsf{P}(1^k)}$
$p \leftarrow\$ \mathcal{P}_k$
$\pi \leftarrow p$
Return $\pi$

$\underline{\mathsf{EG}_{\mathrm{twist}}.\mathsf{G}(1^k, \pi)}$
$p \leftarrow \pi$
If $(p \notin \mathcal{P}_k)$ return $\bot$
$(G_0, G_1) \leftarrow\$ \mathsf{TGen}(1^k, p)$
$(\langle E_0 \rangle, g_0, n_0) \leftarrow G_0$; $(\langle E_1 \rangle, g_1, n_1) \leftarrow G_1$
$\mathbb{G} \leftarrow E_0 \times E_1$; $g \leftarrow (g_0, g_1)$; $n \leftarrow n_0 \cdot n_1$
$G \leftarrow (\langle \mathbb{G} \rangle, n, g)$
Return $G$

$\underline{\mathsf{EG}_{\mathrm{twist}}.\mathsf{S}(1^k, \pi, G)}$
$p \leftarrow \pi$
$z \leftarrow\$ \mathbb{Z}_{2p+1}$
If $(z < n_0)$ return $\varphi_{\mathrm{crt}}(z, 0)$
Else return $\varphi_{\mathrm{crt}}(0, z - n_0 - 1)$

$\underline{\mathsf{EG}_{\mathrm{twist}}.\mathsf{E}(1^k, \pi, G, (h_0, h_1))}$
If $(h_0 \neq \mathcal{O}_0 \wedge h_1 \neq \mathcal{O}_1)$ return 0
Elseif $h_1 = \mathcal{O}_1$ return $l_0(h_0)$
Else return $l_1(h_1)$

$\underline{\mathsf{EG}_{\mathrm{twist}}.\mathsf{I}(1^k, \pi, G, z)}$
If $(z \in \mathrm{im}(l_0))$ return $l_0^{-1}(z)$
Else return $l_1^{-1}(z)$
```

Fig. 11. Definition of eeg family $\mathsf{EG}_{\mathrm{twist}}$ with embedding space $\mathsf{EG}_{\mathrm{twist}}.\mathsf{ES}(k, \pi) = [2p + 1]$. l_0 and l_1 denote the maps from Lemma 5, φ_{crt} the canonical isomorphism $\mathbb{Z}_{n_0} \times \mathbb{Z}_{n_1} \to \mathbb{Z}_n$.

bijection between $E_0 \cup E_1$ and $[2p+1]$ and we obtain $\mathsf{EG}_{\mathrm{twist}}.\mathsf{E}(1^k, G, g^y) \sim U_{[2p+1]}$ for y sampled with $\mathsf{EG}_{\mathrm{twist}}.\mathsf{S}(1^k, G)$. We obtain the following

Lemma 9. *EG_{twist} as defined in Figure 11 is an eeg family with embedding space $\mathsf{EG}_{twist}.\mathsf{ES}(k, G) = [2p + 1]$ and inversion error $\mathsf{EG}_{twist}.\mathsf{ie}(k) = 0$. Furthermore EG_{twist} has pseudorandom embeddings. More precisely, for every (potentially unbounded) adversary $\mathcal{A}$ we have*

$$\mathbf{Adv}^{\mathrm{epr\text{-}psa}}_{\mathsf{EG}_{twist}, \mathcal{A}}(k) = 0 \ .$$

A proof of the lemma can be found in the full version of the paper [4]. Concerning the hardness of sCDH-PSA with respect to $\mathsf{EG}_{\mathrm{twist}}$ we obtain the following.

Lemma 10. *Let EG_{twist} be the embeddable group generator constructed with respect to twisted elliptic curve generator TGen as described above. If the strong Diffie-Hellman assumption for chosen primes holds with respect to TGen, then the strong Diffie-Hellman assumption holds with respect to EG_{twist}.*

Concretely for every adversary $\mathcal{A}$ against game $\mathbf{G}^{\mathrm{scdh\text{-}psa}}_{\mathsf{EG}_{twist}, \mathcal{A}}(\cdot)$, which makes at most Q queries to its DDH*-oracle, there exist adversaries $\mathcal{B}_0$, $\mathcal{B}_1$ against games $\mathbf{G}^{\mathrm{twist}_0\text{-cp-scdh}}_{\mathsf{TGen}, \mathcal{B}_0}(\cdot)$ or $\mathbf{G}^{\mathrm{twist}_1\text{-cp-scdh}}_{\mathsf{TGen}, \mathcal{B}_1}(\cdot)$ respectively making at most Q queries to their* DDH*-oracles, satisfying*

$$\mathbf{Adv}^{\mathrm{scdh\text{-}psa}}_{\mathsf{EG}_{twist}, \mathcal{A}}(k) \leq \mathbf{Adv}^{\mathrm{twist}_0\text{-cp-scdh}}_{\mathsf{TGen}, \mathcal{B}_0}(k) + \mathbf{Adv}^{\mathrm{twist}_1\text{-cp-scdh}}_{\mathsf{TGen}, \mathcal{B}_1}(k).$$

The proof of the lemma can be found in the full version of this paper [4].

5.3 A Parameter-Free Eeg Family Using Rejection Sampling

Eeg family $\mathsf{EG}_{\mathrm{twist}}$ of Section 5.2 is parameter-using, the parameter being the size p of the field $\mathbb{F}_p$. Correspondingly, hardness of sCDH-PSA with respect to

```
$\underline{\mathsf{EG}^{\ell}_{\text{twist-rs}}.\mathsf{P}(1^k)}$
Return $\varepsilon$

$\underline{\mathsf{EG}^{\ell}_{\text{twist-rs}}.\mathsf{G}(1^k,\pi)}$
$p \leftarrow\!\$\ \mathcal{P}_k$
$G' \leftarrow\!\$\ \mathsf{EG}_{\text{twist}}.\mathsf{G}(1^k,p)$
$G \leftarrow (G',p)$
Return $G$
```

```
$\underline{\mathsf{EG}^{\ell}_{\text{twist-rs}}.\mathsf{S}(1^k,\pi,G)}$
$(G',p) \leftarrow G$
For $\ell^* = 1$ to $\ell$
    Do $y \leftarrow \mathsf{EG}_{\text{twist}}.\mathsf{S}(1^k,p,G')$
    If $(\mathsf{EG}_{\text{twist}}.\mathsf{E}(1^k,p,G,g^y) < 2^k)$
        return $y$
Return $\perp$
```

```
$\underline{\mathsf{EG}^{\ell}_{\text{twist-rs}}.\mathsf{E}(1^k,\pi,G,h)}$
$(G',p) \leftarrow G'$
$z \leftarrow\!\$\ \mathsf{EG}_{\text{twist}}.\mathsf{E}(1^k,p,G',h)$
Return $z$

$\underline{\mathsf{EG}^{\ell}_{\text{twist-rs}}.\mathsf{I}(1^k,\pi,G,z)}$
$(G',p) \leftarrow G'$
$h \leftarrow \mathsf{EG}_{\text{twist}}.\mathsf{I}(1^k,p,G',z)$
Return $h$
```

Fig. 12. Parameter-free eeg family $\mathsf{EG}^{\ell}_{\text{twist-rs}}$.

$\mathsf{EG}_{\text{twist}}$ follows from the assumption, that the elliptic curves output by curve-twist generator TGen are secure, independently of the prime p the curve-twist generator TGen is instantiated with. In this section we show how $\mathsf{EG}_{\text{twist}}$ can be used to construct an eeg family $\mathsf{EG}^{\ell}_{\text{twist-rs}}$ for which hardness of sCDH-PSA follows from the weaker assumption that TGen instantiated with a *randomly* chosen prime is able to sample secure elliptic curves. The construction is parameter-free and has embedding space $[2^k]$. The size p of the field over which the elliptic curves are defined is now sampled as part of the group generation. The embedding algorithm uses rejection sampling to ensure that embeddings of group elements g^y for y sampled with $\mathsf{EG}^{\ell}_{\text{twist-rs}}.\mathsf{S}$ are elements of $[2^k]$. The specification of $\mathsf{EG}^{\ell}_{\text{twist-rs}}$'s algorithms may be found in Figure 12.

Theorem 11. *Let $\ell : \mathbb{N} \to \mathbb{N}$ be a polynomial. $\mathsf{EG}^{\ell}_{\textit{twist-rs}}$ as described above is an eeg family with embedding space $\mathsf{EG}^{\ell}_{\textit{twist-rs}}.\mathsf{ES}(k,\pi) = [2^k]$ and inversion error $\mathsf{EG}^{\ell}_{\textit{twist-rs}}.\mathsf{ie}(k) \leq 2^{-\ell(k)}$. Furthermore $\mathsf{EG}^{\ell}_{\textit{twist-rs}}$ has pseudorandom embeddings. More precisely, for every (potentially unbounded) adversary $\mathcal{A}$ we have*

$$\mathbf{Adv}^{\text{epr-psa}}_{\mathsf{EG}^{\ell}_{\textit{twist-rs}},\mathcal{A}}(k) \leq 2^{-\ell(k)} \ .$$

The proof of the theorem can be found in the full version of this paper [4]. As discussed above, we obtain that —assuming that TGen invoked on randomly sampled prime p returns a secure curve-twist pair— the sCDH-PSA-problem with respect to eeg family $\mathsf{EG}^{\ell}_{\text{twist-rs}}$ is hard.

Lemma 12. *Let $\ell : \mathbb{N} \to \mathbb{N}$ be a polynomial and $\mathsf{EG}^{\ell}_{\textit{twist-rs}}$ the eeg family with underlying curve-twist generator TGen as described above. If the sCDH assumption for uniform primes holds with respect to TGen, then* sCDH-PSA *is hard with respect to $\mathsf{EG}^{\ell}_{\textit{twist-rs}}$. Concretely, for every adversary $\mathcal{A}$ against game $\mathbf{G}^{\text{scdh-psa}}_{\mathsf{EG}^{\ell}_{\textit{twist-rs}},\mathcal{A}}(\cdot)$ making at most Q queries to its* DDH*-oracle there exist adversaries $\mathcal{B}_0$, $\mathcal{B}_1$ against $\mathbf{G}^{\text{twist}_0\text{-up-scdh}}_{\mathsf{TGen},\mathcal{B}_0}(\cdot)$ or $\mathbf{G}^{\text{twist}_1\text{-up-scdh}}_{\mathsf{TGen},\mathcal{B}_1}(\cdot)$ respectively, making at most Q queries to their* DDH*-oracles and running in the same time as $\mathcal{A}$, which satisfy*

$$\mathbf{Adv}^{\text{scdh-psa}}_{\mathsf{EG}^{\ell}_{\textit{twist-rs}},\mathcal{A}}(k) \leq 3\left(\mathbf{Adv}^{\text{twist}_0\text{-up-scdh}}_{\mathsf{TGen},\mathcal{B}_0}(k) + \mathbf{Adv}^{\text{twist}_1\text{-up-scdh}}_{\mathsf{TGen},\mathcal{B}_1}(k)\right) + 2^{-\ell(k)}$$

for all $k \in \mathbb{N}_{\geq 6}$.

```
$\underline{\mathsf{EG}_{\text{twist-re}}.\mathsf{G}(1^k, \pi)}$
$p \leftarrow\$ \mathcal{P}_k$
$G' \leftarrow\$ \mathsf{EG}_{\text{twist}}.\mathsf{G}(1^k, p)$; $G \leftarrow (G', p)$
Return $G$

$\underline{\mathsf{EG}_{\text{twist-re}}.\mathsf{S}(1^k, \pi, G)}$
$(G', p) \leftarrow G$
$z \leftarrow\$ [2^{k+1}]$
If $(z \leq 2p)$
    $y \leftarrow \psi_G(z)$
    If $(\mathsf{EG}_{\text{twist}}.\mathsf{E}(1^k, p, G', g^y) < 2^{k+1} - (2p+1))$
        return $y$
    Else $z \leftarrow \mathsf{EG}_{\text{twist}}.\mathsf{E}(1^k, p, G', g^y)$
Else $z \leftarrow z - (2p+1)$
$y \leftarrow \psi_G(z)$
Return $y$

$\underline{\mathsf{EG}_{\text{twist-re}}.\mathsf{P}(1^k)}$
Return $\varepsilon$

$\underline{\mathsf{EG}_{\text{twist-re}}.\mathsf{E}(1^k, \pi, G, h)}$
$(G', p) \leftarrow G$; $b \leftarrow\$ \{0, 1\}$
$z \leftarrow \mathsf{EG}_{\text{twist}}.\mathsf{E}(1^k, p, G', h)$
If $z < 2^{k+1} - (2p+1)$
    $z \leftarrow z + b(2p+1)$
Return $z$

$\underline{\mathsf{EG}_{\text{twist-re}}.\mathsf{I}(1^k, \pi, G, z)}$
$(G', p) \leftarrow G$
If $(z \geq 2p+1)$
    $z \leftarrow z - (2p+1)$
$h \leftarrow \mathsf{EG}_{\text{twist}}.\mathsf{I}(1^k, p, G', z)$
Return $h$
```

Fig. 13. Definition of eeg family $\mathsf{EG}_{\text{twist-re}}$ with embedding space $\mathsf{EG}_{\text{twist-re}}.\mathsf{ES}(k, \pi) := [2^{k+1}]$. ψ_G denotes the bijection $[2p+1] \rightarrow [\mathsf{EG}_{\text{twist}}.\mathsf{S}(1^k, p, G')]$ defined in Section 5.4.

The proof of the lemma can be found in the full version of this paper [4].

5.4 A Parameter-Free Family Using Range Expansion

In this section we modify the algorithms of $\mathsf{EG}_{\text{twist}}$ to obtain an embeddable group family $\mathsf{EG}_{\text{twist-re}}$ with embedding space $\mathsf{EG}_{\text{twist-re}}.\mathsf{ES}(k, \pi) = [2^{k+1}]$. The eeg family has inversion error $\mathsf{EG}_{\text{twist-re}}.\mathsf{ie}(k) = 0$ and achieves uniformly distributed embeddings. The construction is building on a technique introduced by Hayashi et al. [33], where it is used to expand the range of one way permutations. As in Section 5.3, the hardness sCDH-PSA with respect to $\mathsf{EG}_{\text{twist-re}}$ is based on the hardness of the sCDH problem for uniform primes with respect to TGen. The sampling algorithm — in contrast to the construction based on rejection sampling — needs access to only one uniformly random sampled integer, performs at most one exponentiation in the group and uses at most one evaluation of $\mathsf{EG}_{\text{twist}}.\mathsf{E}$ to output y with the correct distribution. Furthermore, exponents sampled by $\mathsf{EG}_{\text{twist-re}}.\mathsf{S}$ are distributed such that the eeg family achieves $\mathsf{EG}_{\text{twist-re}}.\mathsf{ie}(k) = 0$ and for every (potentially unbounded) adversary $\mathcal{A}$ we additionally have $\mathbf{Adv}^{\text{epr-psa}}_{\mathsf{EG}_{\text{twist-re}},\mathcal{A}}(k) = 0$.

The description of $\mathsf{EG}_{\text{twist-re}}$ may be found in Figure 13. We now discuss the construction in greater detail. Let $(G', p) = G \in [\mathsf{EG}_{\text{twist-re}}.\mathsf{G}(k, \pi)]$, where $G' = (\langle \mathbb{G} \rangle, n, g)$. The idea of the construction is to partition $[\mathsf{EG}_{\text{twist}}.\mathsf{S}(1^k, p, G')]$ into two sets M_1, M_2 with $M_1 \cup M_2 = [\mathsf{EG}_{\text{twist}}.\mathsf{S}(1^k, p, G')]$, $\{\mathsf{EG}_{\text{twist}}.\mathsf{E}(1^k, p, G', g^y) \mid y \in M_1\} = \{2^{k+1} - (2p+1), \cdots, 2p\}$ and $\{\mathsf{EG}_{\text{twist}}.\mathsf{E}(1^k, p, G', g^y) \mid y \in M_2\} = \{0, \cdots, 2^{k+1} - (2p+2)\}$. The sampling algorithm $\mathsf{EG}_{\text{twist-re}}.\mathsf{S}$ is constructed such that for y sampled by $\mathsf{EG}_{\text{twist-re}}.\mathsf{S}(1^k, \pi, G)$, the probability $\Pr[y = y']$ equals 2^{-k} for all $y' \in M_2$ and $2^{-(k+1)}$ for all $y' \in M_1$. Embedding algorithm $\mathsf{EG}_{\text{twist-re}}.\mathsf{E}$

on input $(1^k, \pi, G, h)$ first computes $c \leftarrow \mathsf{EG}_{\text{twist}}.\mathsf{E}(1^k, p, G', h)$. If $c \in \{2^{k+1} - (2p+1), \cdots, 2p\}$ its output remains unchanged. Otherwise it is shifted to $\{2p+1, \cdots, 2^{k+1}-1\}$ with probability 1/2. In this way we achieve embeddings , which are uniformly distributed on $\mathsf{EG}_{\text{twist-re}}.\mathsf{ES}(k, \pi) = [2^{k+1}]$.

Our construction relies on the existence of a bijection $\psi_G : [2p+1] \rightarrow [\mathsf{EG}_{\text{twist}}.\mathsf{S}(1^k, p, G')]$ for all $(G', p) = G \in [\mathsf{EG}_{\text{twist-re}}.\mathsf{G}(1^k, \pi)]$. We use the bijection, which was implicitly given in the definition of $\mathsf{EG}_{\text{twist}}.\mathsf{S}$. That is, for $z \in [2p+1]$ we define

$$\psi_G(z) := \begin{cases} \varphi_{\text{crt}}(z, 0) & \text{if } z < n_0 \\ \varphi_{\text{crt}}(0, z - n_0 - 1) & \text{else,} \end{cases}$$

where φ_{crt} denotes the canonical isomorphism $\mathbb{Z}_{n_0} \times \mathbb{Z}_{n_1} \rightarrow \mathbb{Z}_n$.

Theorem 13. *$\mathsf{EG}_{\textit{twist-re}}$ as specified in Figure 13 is an embeddable group family with embedding space $\mathsf{EG}_{\textit{twist-re}}.\mathsf{ES}(k, \pi) = [2^{k+1}]$ and inverson error $\mathsf{EG}_{\textit{twist-re}}.\mathsf{ie}(k) = 0$. Furthermore $\mathsf{EG}_{\textit{twist-re}}$ has pseudorandom embeddings. More precisely, for every (potentially unbounded) adversary $\mathcal{A}$ we have*

$$\mathbf{Adv}^{\text{epr-psa}}_{\mathsf{EG}_{\textit{twist-re}}, \mathcal{A}}(k) = 0 \ .$$

The proof of the theorem can be found in the full version of this paper [4]. As in the case of $\mathsf{EG}^{\ell}_{\text{twist-rs}}$, we obtain that —assuming that TGen invoked on randomly sampled prime p returns a secure curve-twist pair— sCDH-PSA with respect to eeg family $\mathsf{EG}_{\text{twist-re}}$ is hard.

Lemma 14. *Let $\mathsf{EG}_{\textit{twist-re}}$ be the eeg family defined above with underlying curve-twist generator TGen. If the sCDH assumption holds with respect to TGen, then* sCDH-PSA *is hard with respect to $\mathsf{EG}_{\textit{twist-re}}$. Concretely, for every adversary $\mathcal{A}$ against $\mathbf{G}^{\text{scdh-psa}}_{\mathsf{EG}_{\textit{twist-re}}, \mathcal{A}}(\cdot)$ making at most Q queries to its* DDH*-oracle there exist adversaries $\mathcal{B}_0$, $\mathcal{B}_1$ against $\mathbf{G}^{\text{twist}_0\text{-up-scdh}}_{\mathsf{TGen}, \mathcal{B}_0}(\cdot)$ or $\mathbf{G}^{\text{twist}_1\text{-up-scdh}}_{\mathsf{TGen}, \mathcal{B}_1}(\cdot)$ respectively running in the same time as $\mathcal{A}$ and making at most Q queries to their* DDH-*oracles, which satisfy*

$$\mathbf{Adv}^{\text{scdh-psa}}_{\mathsf{EG}_{\textit{twist-re}}, \mathcal{A}}(k) \leq 2\left(\mathbf{Adv}^{\text{twist}_0\text{-up-scdh}}_{\mathsf{TGen}, \mathcal{B}_0}(k) + \mathbf{Adv}^{\text{twist}_1\text{-up-scdh}}_{\mathsf{TGen}, \mathcal{B}_1}(k)\right) .$$

The proof of the lemma can be found in the full version of this paper [4].

Acknowledgments

Benedikt Auerbach was supported by the NRW Research Training Group SecHuman. Mihir Bellare was supported in part by NSF grants CNS-1526801 and CNS-1717640, ERC Project ERCC FP7/615074 and a gift from Microsoft corporation. Eike Kiltz was supported in part by ERC Project ERCC FP7/615074 and by DFG SPP 1736 Big Data.

References

1. M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi. Searchable encryption revisited: Consistency

properties, relation to anonymous IBE, and extensions. In V. Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 205–222. Springer, Heidelberg, Aug. 2005.

2. M. Abdalla, M. Bellare, and P. Rogaway. The oracle Diffie-Hellman assumptions and an analysis of DHIES. In D. Naccache, editor, *Topics in Cryptology – CT-RSA 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 143–158. Springer, Heidelberg, Apr. 2001.
3. G. Ateniese, B. Magri, and D. Venturi. Subversion-resilient signature schemes. In I. Ray, N. Li, and C. Kruegel:, editors, *ACM CCS 15: 22nd Conference on Computer and Communications Security*, pages 364–375. ACM Press, Oct. 2015.
4. B. Auerbach, M. Bellare, and E. Kiltz. Public-key encryption resistant to parameter subversion and its realization from efficiently-embeddable groups. Cryptology ePrint Archive, Report 2018/023, 2018. http://eprint.iacr.org/2018/023.
5. T. Baignères, C. Delerablée, M. Finiasz, L. Goubin, T. Lepoint, and M. Rivain. Trap me if you can – million dollar curve. Cryptology ePrint Archive, Report 2015/1249, 2015. http://eprint.iacr.org/2015/1249.
6. M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In C. Boyd, editor, *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 566–582. Springer, Heidelberg, Dec. 2001.
7. M. Bellare, A. Boldyreva, and S. Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In B. Preneel, editor, *Advances in Cryptology – EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 259–274. Springer, Heidelberg, May 2000.
8. M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among notions of security for public-key encryption schemes. In H. Krawczyk, editor, *Advances in Cryptology – CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 26–45. Springer, Heidelberg, Aug. 1998.
9. M. Bellare, G. Fuchsbauer, and A. Scafuro. NIZKs with an untrusted CRS: Security in the face of parameter subversion. In J. H. Cheon and T. Takagi, editors, *Advances in Cryptology – ASIACRYPT 2016, Part II*, volume 10032 of *Lecture Notes in Computer Science*, pages 777–804. Springer, Heidelberg, Dec. 2016.
10. M. Bellare and V. T. Hoang. Resisting randomness subversion: Fast deterministic and hedged public-key encryption in the standard model. In E. Oswald and M. Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part II*, volume 9057 of *Lecture Notes in Computer Science*, pages 627–656. Springer, Heidelberg, Apr. 2015.
11. M. Bellare, J. Jaeger, and D. Kane. Mass-surveillance without the state: Strongly undetectable algorithm-substitution attacks. In I. Ray, N. Li, and C. Kruegel:, editors, *ACM CCS 15: 22nd Conference on Computer and Communications Security*, pages 1431–1440. ACM Press, Oct. 2015.
12. M. Bellare, K. G. Paterson, and P. Rogaway. Security of symmetric encryption against mass surveillance. In J. A. Garay and R. Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 1–19. Springer, Heidelberg, Aug. 2014.
13. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93: 1st Conference on Computer and Communications Security*, pages 62–73. ACM Press, Nov. 1993.
14. M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In S. Vaudenay, editor, *Advances in Cryptology*

– *EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 409–426. Springer, Heidelberg, May / June 2006.

15. D. J. Bernstein, T. Chou, C. Chuengsatiansup, A. Hülsing, T. Lange, R. Niederhagen, and C. van Vredendaal. How to manipulate curve standards: a white paper for the black hat. Cryptology ePrint Archive, Report 2014/571, 2014. http://eprint.iacr.org/2014/571.
16. D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B.-Y. Yang. High-speed high-security signatures. In B. Preneel and T. Takagi, editors, *Cryptographic Hardware and Embedded Systems – CHES 2011*, volume 6917 of *Lecture Notes in Computer Science*, pages 124–142. Springer, Heidelberg, Sept. / Oct. 2011.
17. D. J. Bernstein, M. Hamburg, A. Krasnova, and T. Lange. Elligator: elliptic-curve points indistinguishable from uniform random strings. In A.-R. Sadeghi, V. D. Gligor, and M. Yung, editors, *ACM CCS 13: 20th Conference on Computer and Communications Security*, pages 967–980. ACM Press, Nov. 2013.
18. D. J. Bernstein and T. Lange. Safecurves: choosing safe curves for elliptic-curve cryptography. https://safecurves.cr.yp.to. Accessed: 18 May 2016.
19. D. J. Bernstein, T. Lange, and R. Niederhagen. Dual EC: A standardized back door. Cryptology ePrint Archive, Report 2015/767, 2015. http://eprint.iacr.org/2015/767.
20. R. Canetti, R. Pass, and a. shelat. Cryptography from sunspots: How to use an imperfect reference string. In *48th Annual Symposium on Foundations of Computer Science*, pages 249–259. IEEE Computer Society Press, Oct. 2007.
21. S. Checkoway, S. Cohney, C. Garman, M. Green, N. Heninger, J. Maskiewicz, E. Rescorla, H. Shacham, and R.-P. Weinmann. A systematic analysis of the juniper dual ec incident. In *Proceedings of the 23rd ACM conference on Computer and communications security*. ACM, 2016.
22. R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In H. Krawczyk, editor, *Advances in Cryptology – CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25. Springer, Heidelberg, Aug. 1998.
23. R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003.
24. J. P. Degabriele, P. Farshim, and B. Poettering. A more cautious approach to security against mass surveillance. In G. Leander, editor, *Fast Software Encryption – FSE 2015*, volume 9054 of *Lecture Notes in Computer Science*, pages 579–598. Springer, Heidelberg, Mar. 2015.
25. J. P. Degabriele, K. G. Paterson, J. C. N. Schuldt, and J. Woodage. Backdoors in pseudorandom number generators: Possibility and impossibility results. In M. Robshaw and J. Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 403–432. Springer, Heidelberg, Aug. 2016.
26. Y. Dodis, C. Ganesh, A. Golovnev, A. Juels, and T. Ristenpart. A formal treatment of backdoored pseudorandom generators. In E. Oswald and M. Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 101–126. Springer, Heidelberg, Apr. 2015.
27. J.-P. Flori, J. Plût, J.-R. Reinhard, and M. Ekerå. Diversity and transparency for ecc. Cryptology ePrint Archive, Report 2015/659, 2015. http://eprint.iacr.org/.
28. G. Frey. How to disguise an elliptic curve (weil descent). Talk given at ECC 1998, 1998.

29. S. D. Galbraith and J. McKee. The probability that the number of points on an elliptic curve over a finite field is prime. *Journal of the London Mathematical Society*, 62(3):671–684, 2000.
30. S. Garg, V. Goyal, A. Jain, and A. Sahai. Bringing people of different beliefs together to do UC. In Y. Ishai, editor, *TCC 2011: 8th Theory of Cryptography Conference*, volume 6597 of *Lecture Notes in Computer Science*, pages 311–328. Springer, Heidelberg, Mar. 2011.
31. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
32. J. Groth and R. Ostrovsky. Cryptography in the multi-string model. In A. Menezes, editor, *Advances in Cryptology – CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 323–341. Springer, Heidelberg, Aug. 2007.
33. R. Hayashi, T. Okamoto, and K. Tanaka. An RSA family of trap-door permutations with a common domain and its applications. In F. Bao, R. Deng, and J. Zhou, editors, *PKC 2004: 7th International Workshop on Theory and Practice in Public Key Cryptography*, volume 2947 of *Lecture Notes in Computer Science*, pages 291–304. Springer, Heidelberg, Mar. 2004.
34. B. S. Kaliski Jr. One-way permutations on elliptic curves. *Journal of Cryptology*, 3(3):187–199, 1991.
35. J. Katz, A. Kiayias, H.-S. Zhou, and V. Zikas. Distributing the setup in universally composable multi-party computation. In M. M. Halldórsson and S. Dolev, editors, *33rd ACM Symposium Annual on Principles of Distributed Computing*, pages 20–29. Association for Computing Machinery, July 2014.
36. M. Lochter and J. Mekle. *RFC 5639: ECC Brainpool Standard Curves & Curve Generation*. Internet Engineering Task Force, Mar. 2010.
37. B. Möller. A public-key encryption scheme with pseudo-random ciphertexts. In P. Samarati, P. Y. A. Ryan, D. Gollmann, and R. Molva, editors, *ESORICS 2004: 9th European Symposium on Research in Computer Security*, volume 3193 of *Lecture Notes in Computer Science*, pages 335–351. Springer, Heidelberg, Sept. 2004.
38. NIST. Digital signature standard (DSS), 2013. FIPS PUB 186-4.
39. H. Orman. The oakley key determination protocol, 1998.
40. C. Petit and J.-J. Quisquater. On polynomial systems arising from a Weil descent. In X. Wang and K. Sako, editors, *Advances in Cryptology – ASIACRYPT 2012*, volume 7658 of *Lecture Notes in Computer Science*, pages 451–466. Springer, Heidelberg, Dec. 2012.
41. A. Russell, Q. Tang, M. Yung, and H.-S. Zhou. Cliptography: Clipping the power of kleptographic attacks. In J. H. Cheon and T. Takagi, editors, *Advances in Cryptology – ASIACRYPT 2016, Part II*, volume 10032 of *Lecture Notes in Computer Science*, pages 34–64. Springer, Heidelberg, Dec. 2016.
42. A. Russell, Q. Tang, M. Yung, and H.-S. Zhou. Generic semantic security against a kleptographic adversary. In B. M. Thuraisingham, D. Evans, T. Malkin, and D. Xu, editors, *ACM CCS 17: 24th Conference on Computer and Communications Security*, pages 907–922. ACM Press, Oct. / Nov. 2017.
43. A. Young and M. Yung. The dark side of "black-box" cryptography, or: Should we trust capstone? In N. Koblitz, editor, *Advances in Cryptology – CRYPTO'96*, volume 1109 of *Lecture Notes in Computer Science*, pages 89–103. Springer, Heidelberg, Aug. 1996.
44. A. Young and M. Yung. Kleptography: Using cryptography against cryptography. In W. Fumy, editor, *Advances in Cryptology – EUROCRYPT'97*, volume 1233 of *Lecture Notes in Computer Science*, pages 62–74. Springer, Heidelberg, May 1997.