# Constant Size Secret Sharing: with General Thresholds, Towards Standard Assumptions, and Applications

Katarzyna Kapusta[1,2], Matthieu Rambaud[2], and Ferdinand Sibleyras[3]

[1] Thales ThereSIS
[2] Télécom Paris & Institut Polytechnique de Paris
[3] NTT Social Informatics Laboratories

**Abstract.** We consider threshold Computational Secret Sharing Schemes, i.e., such that the secret can be recovered from any $t+1$ out of $n$ shares, and such that no computationally bounded adversary can distinguish between $t$ shares of a chosen secret and a uniform string. We say that such a scheme has Constant Size (CSSS) if, in the asymptotic regime of many shares of small size the security parameter, then the total size of shares reaches the minimum, which is the size of an erasures-correction encoding of the secret with same threshold. But all CSSS so far have only maximum threshold, i.e., $t = n - 1$. They are known as All Or Nothing Transforms (AONT). On the other hand, for arbitrary thresholds $t < n - 1$, the shortest scheme known so far is [Kra93, Crypto], which has instead twice larger size in the previous regime, due to a size overhead of $n$ times the security parameter. The other limitation of known CSSS is that they require a number of calls to *idealized* primitives which grows linearly with the size of the secret.
Our first contribution is to show that the CSSS of [Des00, Crypto], which holds under the ideal cipher assumption, looses its privacy when instantiated with a plain pseudorandom permutation.
Our main contribution is a scheme which: is the first CSSS for any threshold $t$, and furthermore, whose security holds, for the first time, under any plain pseudorandom function, with the only idealized assumption being in the key-derivation function. It is based on the possibly new observation that the scheme of [Des00] can be seen as an additive secret-sharing of an encryption key, using the ciphertext itself as a source of randomness. A variation of our construction enables to improve upon known schemes, that we denote as *Encryption into Shares with Resilience against Key exposure* (ESKE), having the property that all ciphertext blocks are needed to obtain any information, even when the key is leaked. We obtain the first ESKE with arbitrary threshold $t$ and constant size, furthermore in one pass of encryption. Also, for the first time, the only idealized assumption is in the key-derivation.
Then, we demonstrate how to establish fast revocable storage on an untrusted server, *from any black box* ESKE. Instantiated with our ESKE, then encryption and decryption both require only 1 pass of symmetric primitives under standard assumptions (except the key-derivation), compared to at least 2 consecutive passes in [MS18, CT-RSA] and more in [Bac+16, CCS].

We finally bridge the gap between two conflicting specifications of AONT in the literature: one very similar to CSSS, which has indistinguishability, and one which has not.

## 1 Introduction

### 1.1 Constant Size Secret Sharing: Applications and Limitations

A much investigated technique in distributed storage [Gar+00] consists in fragmenting then dispersing data over multiple independent storage locations in order to block an attacker unable to compromise all of them. On the other hand, users seek for additional ways of confidentiality and availability reinforcements, in case of data leakage, e.g., of secret keys [AGH12; KR19; KSLC18; KRM20]. A secret sharing scheme with $n$ shares and threshold $t$ allows a dealer to transform its plaintext secret into $n$ bitstrings, denoted *Shares*, such that, if the dealer computed them correctly, then (i) the secret can be efficiently reconstructed from any $t+1$ shares (ii) no information can be obtained on the secret from any $t$ shares.

Krawczyk [Kra93] was the first to evidence that secret sharing schemes can achieve a total size of shares which is short, when basing security on computational assumptions. His scheme proceeds by the following steps: (i) generate a symmetric encryption of the secret using a secret key $K$ sampled at random, (ii) apply $t$-erasure resilient Reed-Solomon encoding to the resulting ciphertext, to form a codeword of $n$ coordinates, denoted as the "fragments" $(F_i)_{i\in[n]}$, (iii) then each share $i$ consists in the concatenation of: $F_i$, with a secret share $K_i$ of the key $K$ under a $t$ out of $n$ secret sharing scheme. [Krawczyk's scheme was then made robust in [BR07].] However there is a storage overhead in Krawczyk's scheme: in addition to the Reed-Solomon encoding of the ciphertext, which is unavoidable for recoverability from any $t+1$ shares, we see that $n$ shares of key need to be stored, all of them having same size as the key. This overhead is critical when transforming the secret into a large number of shares, each of them of small size, i.e., close to the key's. This impacts the many use cases recalled below, to which we add key exposure resilience and revocation (§1.2), since they require this range of parameters. [Notice that the previous overhead is amortized over multiple secrets in [Gar+00], at the cost that their reconstruction involves processing of a common secret key which, if leaked to the adversary, compromises the privacy of any secret of which it knows at least one share.]

In this paper, we define as Computational **Constant Size Secret Sharing (CSSS)** schemes that overcome the size issue described above. **All existing CSSS so-far are limited to the specific threshold $t = n - 1$ out of $n$.** In this regime, the closest definition to secret sharing is the one in [Des00, Def. 4]. It is denoted as "AONT", but this terminology is dangerous since employed for conflicting definitions, see §1.2.5. The first known CSSS is denoted as "OAEP", which is used in the PKCS ♯1 [Lab12], we refer to [Boy99] for a description. There, it is proven, under idealized assumption, that OAEP satisfies our definition of CSSS (and even more, since it can handle shares of size a few bits instead of one block). Notice that OAEP is given an (unpublished) proof of adaptive security in Lemma 1 of the long version [MS17] of [MS18].

The first use case of CSSS was introduced by Rivest [Riv97], as the main building block of his encryption scheme denoted "All Or Nothing", in order to slow down brute-force key search attacks performed on encrypted data. Notice that the CSSS considered in [Riv97], under the name "All-Or-Nothing package transform", can be viewed as a special case of "OAEP", as evidenced in [Boy99]. Later, [MPR96] introduced a public key encryption scheme consisting in preprocessing of a plaintext with a CSSS before encrypting *only one* of the obtained shares (of size of a block), which reduces the encryption cost in comparison to the Rivest's proposal. This idea was then generalized in [Boy99, §1] to any CSSS. [JSY99] stressed that this idea, of encrypting *only one block*, is also relevant when symmetric encryption is performed by a device containing the key, when one wants to minimize the communications and computations of this device in order to avoid leakage. Another application of CSSS, due to Rivest and proven in [BB00, §1.4], is a keyless compiler from: any ideal authentication mechanism (a "MAC channel"), into: a symmetric encryption scheme with ind-CPA. It consists in prepossessing the plaintext with a CSSS, then, applying the costly information theoretic encryption scheme on the first share only (of size one block), that takes every bit $b$ to the pair ($b$ correctly authenticated, $\bar{b}$ wrongly authenticated).

However, we observe that **all existing CSSS so far make a number of calls to idealized primitives which is linear in the size of the plaintext**:

- OAEP, as specified in [Boy99] applies an idealized hash function on a bitsize equal to the *whole* length of the secret; and an idealized random generator that outputs a bitlength equal to the *whole* length of the secret.
- Desai's CSSS [Des00], denoted as "CTRT" and described in details in Section 3, holds in the ideal blockcipher model.
- A CSSS, denoted as "AONT-RS", was proposed by Resch & Plank in [RP11]. The scheme is a combination of Rivest's first step of the OAEP processing with the Reed-Solomon coding. It was attacked, and then repaired in the Random Oracle model, by [CLM17].
- Authors of [Can+00] consider a variation of CSSS which needs a part of the output of the scheme - denoted as the *private* part, which is *necessary* for the reconstruction - to be *kept secret* and never leaked. Their resilience against leakage of "all but one shares" is thus further conditioned to *nonleakage* of this secret part. They consider constructions of this notion from black-box functions denoted as ERF. Some of the ERF introduced in their work are claimed to hold under standard assumptions. Fixing the problem of the secret part would lead to a proper secret sharing scheme under standard assumptions. Such fix is indeed provided by the authors , but at the cost of an increase of the output size, which does not fit anymore in our requirement of *Constant Size* of the shares.

## 1.2 Contributions

### 1.2.1 Privacy of the CSSS of [Des00] under Standard Assumptions.

Idealized assumptions seem indispensable in all of the schemes reviewed above. In §3 We illustrate its importance by taking the example of the CSSS of [Des00], for which we show that, when instantiated under the *standard* assumption of plain pseudorandom permutation (PRP), then an adversary can distinguish between two secrets.

### 1.2.2 Main Contribution: Constant Sized Secret Sharing for General Thresholds, and Requiring one Single Call to an Idealized Function.

In §4 we completely remove the first limitation by allowing arbitrary thresholds in CSSS, and address the second by making a significant step towards removing the use of idealized primitives in CSSS. We achieve both claims in one single construction. It is based on the simple but new observation that the scheme of [Des00] can be seen as an additive perfect secret sharing of a secret blockcipher key, with source of randomness equal to an encryption of the secret under this key. For this idea to work, a fortiori when generalized to arbitrary thresholds, the conditions of operation of perfect secret sharing must be guaranteed, in particular that the randomness be independent from the secret, i.e., the key. Whereas this was made possible in [Des00] by the use of encryption with an Ideal Cipher, we instead achieve a construction that operates with *any* plain pseudorandom function (including a PRP). We do still use an idealized transform, but called only once on a constant small number of bits, equal to the security parameter, for the purpose of key-derivation. We leave as an open problem to completely remove idealized assumptions or, on the contrary, prove that the single-call to an idealized primitive, that we do, is unavoidable.

### 1.2.3 Improved Encryption into Shares with Resilience against Key Exposure (ESKE).

The idea of Rivest's encryption [Riv97], denoted "All or Nothing", was specified by [Des00, Def. 2] as "Non-Separability of Keys". Roughly speaking, a symmetric encryption scheme based on a blockcipher has this property if and only if: let $n$ be the number of blocks of a ciphertext, then an adversary making at most $n-1$ oracle calls to the inverse of the blockcipher with the correct secret key, cannot infer any information about the secret key. This definition was given a more intuitive variant in [KSLC18], denoted "CAKE". There, the adversary, instead of making oracle calls, is: being leaked the secret key, and allowed to learn at most $n-2$ ciphertext blocks. On the other hand their recoverability is guaranteed only provided all $n$ blocks. The definition was then generalized in [KRM20] under the name "SAKE". In Definition 5 we generalize the definition into any threshold $t$, to which we give the possibly more intuitive name of "ESKE". A ESKE is a keyed randomized transform of a plaintext secret into $n$ shares, which is invertible from any $t+1$ out of $n$ shares, such that we have both resiliences: (1) IND-CPA in the classical sense, i.e., indistinguishability between two chosen plaintexts given all their shares, with respect to the secret key $K^{\mathrm{acc}}$; and (2) for any adversary being furthermore leaked the secret key $K^{\mathrm{acc}}$, then indistinguishability between two chosen secrets given any $t$ shares, as for secret sharing. Concretely, $K^{\mathrm{acc}}$ is the "access key" given to persons accredited to learn the secret, and the scenario (2) deals with "exposure" of $K^{\mathrm{acc}}$. [Notice also a rich line of research, e.g., [Dzi06; GWZ22], with orthogonal specifications: their adversary 2. has the capacity to download all ciphertexts, but can subsequently store much less than all-but-one shares, before being leaked the key.]

Of course we have the generic solution to achieve ESKE which is: encrypt the secret, then secret-share the ciphertext with a computational scheme. However the whole requires at least two consecutive passes of symmetric primitives on the length of the secret, thus in Table 1 we compare to a non-trivial line of research that applies only one pass of primitives. Our contribution is that, using our methods of §3, we build a ESKE using *one pass* of primitives, which: for the first time, achieves general threshold $t < n$ instead of only $t = n - 1$ as previously, and for the first time achieves total constant size from a single call to an idealized KDF of size the security parameter.

| Scheme | Privacy Resilience [#shares] | Erasures Tolerance [#shares] | Total size of shares | Assumptions |
|---|---|---|---|---|
| [KSLC18] Bastion | $n-2$ | 0 | $\frac{n}{n-1}\lvert\boldsymbol{S}\rvert$ | Ideal Cipher |
| [KRM20] SSAKE | $n-1$ | 0 | $2\lvert\boldsymbol{S}\rvert$ | Ideal Cipher |
| [KRM20] ROSSAKE | $n-1$ | 0 | $2\lvert\boldsymbol{S}\rvert$ | Random Oracle |
| **§5, for any $\boldsymbol{t < n}$** | $t$ | $n-(t+1)$ | $\frac{n}{t}\lvert\boldsymbol{S}\rvert$ | PRP + ideal KDF |

**Table 1.** Comparison between our $(n, t)$ ESKE for any $t$, and the schemes of [KSLC18; KRM20], which have fixed privacy ($n-1$ or $n-2$) and reconstruction ($n$) thresholds. $\lvert\boldsymbol{S}\rvert$ denotes the bitsize of the secret, and for simplicity the sizes are expressed in the asymptotic regime of "large secret divided in many small shares", more specifically $\lvert\boldsymbol{S}\rvert \cong t\kappa$ for large $t$, where $\kappa$ is the security parameter, which is typically the case in the application of ESKE §1.2.4 for $t = n - 1$. See below Def. 3 for other asymptotics (both SSAKE and ROSSAKE have same size as [Kra93]).

#### 1.2.4 Faster Revocable Storage on an Untrusted Server.

We consider a secret owner Alice who has access to a storage place, denoted "server", typically a cloud provider. The server can only be trusted to correctly save or erase the data, but not to perform access control on it. It can be seen as a public storage box to which everyone can access. Every data stored on the server is assumed read by what we denote as an "inner adversary" $\mathcal{A}^{in}$, typically the cloud provider. The first requirement is that $\mathcal{A}^{in}$ should not learn any plaintext secret. While giving access to the secret without leaking it to $\mathcal{A}^{in}$ is simple, i.e., encrypting it and then distributing the key to authorized persons, efficiently revoking access to a person is a challenge.

Let us outline the idea of solution introduced in [Che+13] and popularized by [Bac+16; MS18]. To store a secret, Alice transforms into a ciphertext, which comes as a concatenation of logical units of small size, which we denote here as "shares", such that all but one shares give no information even if given the encryption key. After uploading the transformed data to the server, Alice privately gives the symmetric key, denoted "access key $K^{\mathrm{acc}}$", to the persons accredited to learn the secret. To subsequently revoke a person, Alice: (i) chooses at random a small number $n_{miss}$ of shares (ii) re-encrypts them with a fresh symmmetric key, (iii) gives the fresh key only to the non-revoked persons, along with the indices of the re-encrypted shares. This operation prevents further access to the secret, with a high rate of success, for any previously accredited person who: does not have the new key, and who did not store *all* the re-encrypted shares locally in advance, see also [Wan+17]. For instance, as discussed in [Bac+16, §4], if the total number of stored shares is $n$ and this person stored locally only $n_{loc}$ shares, then she is likely to be unable to learn the value of all re-encrypted shares, except with probability lower than $(n_{loc}/n)^{n_{miss}}$. This probability is small, since it is supposed that a person will not store the whole data, as it is an impractical solution. [Let us furthermore make the observation that, after $i$ consecutive revocations, then this probability is raized to the power $i$. Thus, chances of success of previously revoked persons decrease exponentially with the number of subsequent revocations. ] In conclusion, this idea saves much bandwith and computation costs, since each revocation operates only on shares of small constant size, by comparison with the naive solution which would have consisted in re-encrypting the whole secret at every revocation.

Then, [Bac+16] proposed an optimisation, using mechanisms known as "key-regression", that enables accredited persons to manage only the most recent fresh key, from which all previous keys can be derived. The state of the art instantiation of the idea is [MS18, §3], which reduce the encryption and decryption cost of [Bac+16] down to 2 consecutive passes of symmetric primitives on the whole secret (see the discussion in their §3). The symmetric encryption scheme of [MS18] consists in (a) performing symmetric encryption on the secret (in detail: using the scheme "$Enc(k,m)$" of [Eve+17, Figure 4]) (b) followed by a CSSS (which they denote "AONT", although with relaxed security of all-but-"$\ell$" shares).

Our contribution is the following very simple but apparently new scheme, that relies only the specification of a black box $(n, n-1)$-ESKE. Alice generates an ESKE secret key $K^{\mathrm{acc}}$, which she gives to the accredited persons. Then, to give access to any subsequent secret to these accredited persons, she generates a ESKE ciphertext of it with $K^{\mathrm{acc}}$, then stores it on the server. Revocation of a person from one or several files goes as in (i)(ii)(iii) (with the same fresh key for all files). Although intuitive, in Theorem 7 we formalize and prove that the previous new scheme solves the problem with the same aforementioned probability of success, as [Bac+16; MS18].

Besides its generality, of using any black box ESKE, our scheme also clarifies that only a single key $K^{\mathrm{acc}}$ needs to be managed by the accredited persons for potentially multiple secrets, whereas [Bac+16; MS18] studied revocation for only one secret under one key. The main benefits of our scheme then appear when *instantiated* with our new ESKE of §3, with shares of size one block (typically

128 bits). First, it is the first proven scheme that requires only one call to an idealized primitive, of small input and output, i.e., the size of a key. By contrast, [MS18] required an AONT as sub-routine. But, before our main contribution §1.2.2, all AONTs applied idealized primitives on the whole plaintext. Last but not least, the new scheme requires only 1 pass of symmetric primitives, for both initial encryption and subsequent decryptions. By constrast, access to a plaintext in [MS18] costs 2 consecutive passes of symmetric primitives, thus not in a parallelisable way, which thus doubles the latency.

#### 1.2.5 Minor Contribution: *a Posteriori* Constant Size ESKE.

We could imagine the following more constrained use-case, denoted *a posteriori ESKE*. Namely, consider that we are given an *existing* standard ciphertext, of which we do not know the key, but we would like to upgrade it into an ESKE ciphertext. A generic solution to would be to apply a CSSS on this ciphertext, but we aim at cheaper transforms. Our new ESKE of §5, itself derived from §4, is however not applicable here, since the secret-sharing is performed in a way that requires knowledge of the encryption key. In Theorem 9 we make a move towards the question. We show that CTR encryption with an ideal cipher, followed by one linear transform denoted "AONT" by [Sti01], yields a ESKE. The construction owes much to [KSLC18] and upgrades their privacy threshold into $n-1$ shares. But it is not included in table 1, since it is strictly improved by our new ESKE of §5 (which is however not *a posteriori*). The most important part in Theorem 9 is actually the converse, which shows that any linear transform having this property is an "AONT".

Surprisingly, what is denoted "AONT" above, following the terminologies of [Riv97] and [Sti01], does *not* guarantee indistinguishability of two chosen secrets, for an adversary which is given on all but one shares of their transforms. So this strongly conflicts with the ones of [Boy99; Des00], which are generalized by the definition of secret sharing, as discussed below our Def. 2. [This conflict of definitions was stressed by [Boy99, §1.3], we give some concrete examples under Def. 8.]

## 2 Notations and Definitions

We make use of the following data structures that all can be represented as strings *string* of $|string|$ bits.

**$\kappa$ is our security parameter**, typically of the size of a symmetric key.

**Sampling $K \xleftarrow{\$} \{0,1\}^\kappa$** uniformly at random.

**$[M] := \{1,\ldots,M\}$** for any integer $M$.

**Sum $s + pad$** is the sum of two binary vectors $s$ and $pad$ of same length, their bitwise XOR. Sometimes we will denote $c - pad$ to stress that we remove the one-time pad.

**Output of an Adversary $\mathbb{P}(\mathcal{A}^{\mathcal{O}} \rightarrow b)$** For a (possibly idealized) function $\mathcal{O}$, a bit $b \in \{0,1\}$ and a machine $\mathcal{A}$, denoted *Adversary*, we denote $\mathcal{A}^{\mathcal{O}} \rightarrow b$ the event

"$\mathcal{A}$ interacts with oracle $\mathcal{O}$ and then outputs 0", then with a $\mathbb{P}$ the probability of this event.

**A Short Random Oracle (RO) $\mathbf{f} : \{0,1\}^\kappa \longrightarrow \{0,1\}^\kappa$** is an idealized function, that returns the same output when queried on the same input, and on every input not queried before, returns a string of size $\kappa$ sampled uniformly at random. Alternatively, it can be seen as a function uniformly drawn at random among the $(2^\kappa)^{(2^\kappa)}$ possibilities

**Pseudorandom Advantage of a Keyed Function (PRF).** Let $\mathrm{F}_\bullet : \{0,1\}^\kappa \times \{0,1\}^\kappa \longrightarrow \{0,1\}^\kappa$ be a (keyed) function. For any key $K \in \{0,1\}^\kappa$ and any machine $\mathcal{A}$ interacting with either a Short Random Oracle f or with $\mathrm{F}_K$, we denote $\mathcal{A}^{BC_K}$ and $\mathcal{A}^{\mathrm{f}}$ the output bit of $\mathcal{A}$ when interacting with the one or the other. We define the *PRF advantage* of $\mathrm{F}_\bullet$, parametrized by any integers $\sigma$ and $\tau$, as: given a secret key $K \xleftarrow{\$} \{0,1\}^\kappa$ sampled uniformly at random, the maximum distinguishing advantage over adversaries $\mathcal{A}$ making at most $\sigma$ oracle accesses to $\mathrm{F}_K$ and running in time $\tau$:

$$\mathrm{Adv}_{\mathrm{PRF}}^{\mathrm{F}_\bullet}(\sigma) := \max_{\mathcal{A}} \mathbb{P}(\mathcal{A}^{\mathrm{F}_K} \to 0) - \mathbb{P}(\mathcal{A}^{\mathrm{f}} \to 0) \tag{1}$$

If the PRF advantage is negligible for any polynomial $\tau$, then we say that F is a *Pseudorandom Function* (PRF). The pseudorandomness model is weaker than the random oracle one but more realistic as it allows the inputs to be related to the outputs. See §A.2 for an example of PRF: Pseudo-Random Permutations (PRP), of which Ideal Ciphers.

## 2.1 Perfect Secret Sharing with Uniformity and Full Reconstruction.

We recall threshold perfect secret sharing, and give names to two additional properties of the Shamir scheme, which are: uniformity of adversarial shares, and reconstructibility of the randomness. The latter will be crucial to reach thresholds $t < n-1$ in §4 and §5. The former will enable our implementation of CSS, in §4, to match the uniformity requirement of Definition 2.

***Definition 1 (PSSUF).*** Let $0 \le t < n$, and $u$ and $d$, be some positive integers. A *Perfect Secret Sharing Scheme with Uniformity and Full reconstruction* for secret space $\{0,1\}^\kappa$, randomness space $\{0,1\}^{u\kappa}$, shares space $\{0,1\}^{d\kappa}$, threshold $t$ and $n$ shares, is a deterministic transformation:

$$\begin{array}{llll} \phi : & \{0,1\}^\kappa \times \{0,1\}^{u\kappa} & \longrightarrow & \left(\{0,1\}^{d\kappa}\right)^n \\ & K \quad , \quad \boldsymbol{R} & \longrightarrow & sh_1, \ldots, sh_n \text{ such that:} \end{array} \tag{2}$$

- for every fixed secret $K \in \{0,1\}^\kappa$, *if* $\boldsymbol{R} \in \{0,1\}^{u\kappa}$ varies uniformly at random, then any fixed subset $\mathcal{I} \subset [n]$ of $t$ shares $(sh_i)_{i\in\mathcal{I}}$ varies uniformly at random in $\left(\{0,1\}^{d\kappa}\right)^t$;
- we have a map that reconstructs both $K$ and $\boldsymbol{R}$ from any $t+1$ shares, i.e.,

  $$\mathsf{FullReco} : \left\{(t+1)\text{-subsets } \mathcal{H} \subset [n]\right\} \times \left(\{0,1\}^{d\kappa}\right)^{t+1} \longrightarrow \{0,1\}^\kappa \times \{0,1\}^{u\kappa} ,$$

  such that for any $K$, $\boldsymbol{R}$ and $\mathcal{H}$, $\mathsf{FullReco}^{\mathcal{H}}\left(\left(\phi(K,\boldsymbol{R})_i\right)_{i\in\mathcal{H}}\right) = (K, \boldsymbol{R})$.

Uniformity of any $t$ shares implies all other alternative privacy requirements, such as: [BGK20] any $t$ shares have the same distribution for any secret, or [CDN15, §11.9.2] do not diminish entropy of the secret.

The first example of PSSUF that we will use is the ***additive*** $\boldsymbol{(n, n-1)}$***-scheme***, where $u = n-1$ and $d = 1$. On input $(K, \boldsymbol{R} = (R_1, \ldots, R_{n-1})$, output $\left(R_1, \ldots, R_{n-1}\, , \, K - \sum_{j=1}^{n-1} R_j\right)$.

The second example of PSSUF that we will use is the ***Shamir*** $\boldsymbol{(n, t)}$***-scheme***, where $d = \max(\lceil \lceil \log_2(n) \rceil / \kappa \rceil, 1)$ and $u = t$. Embed the secret space $\{0,1\}^\kappa$ in some finite field $\mathbf{F}_q$ containing at least $n$ elements, e.g., $q = 2^\kappa$ if $n \leq 2^\kappa$. Consider any $n$ fixed public distinct elements $(\alpha_i)_{i \in [n]} \in \mathbf{F}_q$ denoted as "evaluation points". $\phi$ is defined as the linear map that, on input $(K, \boldsymbol{R} = (R_1, \ldots, R_t) \in \mathbf{F}_q \times \mathbf{F}_q^t$, outputs the shares $\left(K + \sum_{j=1}^{t} R_j\, \alpha_i^j\right)_{i \in [n]}$. Notice that shares can be seen as evaluations at the $(\alpha_i)_{i \in [n]}$ of the polynomial $P(X) = K + \sum_{j=1}^{t} R_j\, X^j$. For any $(t+1)$-sized $\mathcal{H} \subset [n]$, $\mathsf{FullReco}^{\mathcal{H}}$ recovers the polynomial $P$, and thus its coefficients $(K, \boldsymbol{R})$, as a linear combination of the shares with the Lagrange coefficients corresponding to $\mathcal{H}$.

[For simplicity in the description above we considered that we had $n+1$ evaluation points, since we implicitely considered that $\alpha_i \neq 0\ \forall i$, since we encoded the secret $K$ as $P(0)$. This assumption can be removed by "evaluation at infinity", i.e., by encoding $K$ instead as the coefficient of degree $t$ of $P$.]

Finally, one can optimize the computational cost when $n << 2^\kappa$ by choosing instead a smaller $q = 2^{\lceil \log_2(n) \rceil}$: divide the secret in $\lceil \kappa / \lceil \log_2(n) \rceil \rceil$ pieces, each embedded in $\mathbf{F}_q$, then perform the PSSUF of Shamir on each of them in parallel.

**2.2 (Computational) Secret Sharing Schemes** We introduce the definition of secret sharing that we will use, then compare it to the literature.

***Definition 2 (CSS).*** Let $0 \leq t < n$, and $d$, be some positive integers. A *Computational Threshold Secret Sharing Scheme (with Uniformity)* (CSS) with $n$ shares and threshold $t$ is a pair an efficiently computable randomized function $\boldsymbol{\Sigma}$ and a deterministic function $\mathsf{SReco}$ as follows. $\boldsymbol{\Sigma}$ takes as input any non empty sequence of bits $\boldsymbol{S} \in \{0,1\}^*$, denoted the secret, and returns $n$ bit-strings $(Sh_i := \boldsymbol{\Sigma}(\boldsymbol{S})_i)_{i \in [n]}$, denoted as the *Shares*, whose sizes are only determined by $|\boldsymbol{S}|$. $\mathsf{SReco}$ takes any $(t+1)$-sized subset $\mathcal{H} \subset [n]$ of indices, any $(t+1)$ bitstrings. There satisfy the two following guarantees:

- *Recoverability against* $n-(t+1)$ *erasure adversaries:* for any $\boldsymbol{S}$ and $(t+1)$-sized $\mathcal{H} \subset [n]$, we have $\mathsf{SReco}^{\mathcal{H}}\left[\left(\boldsymbol{\Sigma}(\boldsymbol{S})_i\right)_{i \in \mathcal{H}}\right] = \boldsymbol{S}$.

- *Computational Privacy with uniformity against* $t$*-adversaries*: for a given bitsize $|\boldsymbol{S}|$, let us consider the two oracles $\mathcal{O}\boldsymbol{\Sigma}$ and \$. $\mathcal{O}\boldsymbol{\Sigma}$, when queried with: $\boldsymbol{S}$ of size $|\boldsymbol{S}|$, and a $t$-subset $\mathcal{I} \subset [n]$ of indices, computes $\boldsymbol{\Sigma}(\boldsymbol{S})$ then returns the shares $\left(\boldsymbol{\Sigma}(\boldsymbol{S})_i\right)_{i \in \mathcal{I}}$. The dummy oracle \$ has the same interface, but on every query $\boldsymbol{S}$, it samples then returns a bitstring uniformly at random, of same length as $\left(\boldsymbol{\Sigma}(\boldsymbol{S})_i\right)_{i \in \mathcal{I}}$, irrespectively of if $\boldsymbol{S}$ was already queried or not. Consider the maximum distinguishing advantage over all adversaries $\mathcal{A}$ running in time $\tau$ and making at most $M$ oracle queries:

$$\max_{\mathcal{A}} \Pr(\mathcal{A}^{\mathcal{O}\boldsymbol{\Sigma}} \to 0) - \Pr(\mathcal{A}^{\$} \to 0) \tag{3}$$

Then the above maximum advantage is required to be negligible, for any (polynomial) $|\boldsymbol{S}|$, over the adversaries with polynomial time $\tau$.

Notice that PSSUF (Def. 1) implies CSS (Def. 2).

Let us compare to the definitions of [BR07], which are, among others, formalizations of [Kra93]. *Recoverability against erasure adversaries* is the same as their "PR0" p6, for the class of adversaries, denoted "threshold" [BR07, p7], which erase at most $n-(t+1)$ shares. Notice that Recoverability against arbitrary adversaries is not under the scope of our Def. 2. It could be implemented by having the dealer sign every share, but more efficient techniques for this purpose can be found [CLM17; BR07] and [CDN15, §11.8].

*Computational Privacy with Uniformity*: implies the privacy guarantee denoted "CSS" [BR07, p6], i.e., indistinguishability between two chosen plaintext secrets. Precisely, we consider the class of adversaries, denoted "threshold" [BR07, p7], which learn at most $t$ shares. Requiring uniformity is motivated by [Des00], below its Def. 4, for the use-case of slowing down key-search attacks.

Let us compare to [Des00]. In the special case where: $t=n-1$, $|\boldsymbol{S}| = t\kappa = (n-1)\kappa$, then Definition 2 is the [Des00, Def. 4] denoted "AONT", with the only difference that we allowed $M$ oracle queries instead of 1. Notice that [Des00, Def. 4] makes $n$ depend on $|\boldsymbol{S}|$ without precision, but his implementation, which we recall in §3, considers only the particular case $|\boldsymbol{S}| = (n-1)\kappa$.

***Definition 3 (CSSS).*** For any $0 \le t < n$, assume furthermore that $n < 2^{\kappa}$, then we say that a CSS $\boldsymbol{\Sigma}$ is of *Constant Size* if, on input a secret of bitsize $|\boldsymbol{S}| = (t+e)\kappa$, where $e$ is a positive integer, which stands for "extra-length", then the total size of shares is

$$\sum_{i=1}^{n} |Sh_i| \le n\kappa + \frac{n}{t+1} e\kappa \; . \tag{4}$$

The condition $n \le 2^{\kappa}$ is essentially unavoidable, by the information-theoretic lower bound of [BGK20] in $\log_2(t)$ on the size of shares. However it becomes significant for a number of shares which is exponential in the security parameter, whereas the definition of CSS is meaningful only for polynomial adversaries. By "essentially", we exclude the two cases: $t \ll n$, and $t = n-1$, to which the bound of [BGK20] does not apply, and thus for which the limitation $n \le 2^{\kappa}$ could potentially be overcome. But as stressed above, these questions become meaningful in a regime where our computational definitions are meaningless.

Let us compare to the scheme of [Kra93, §3], which is also a CSS with recoverability against erasures, also in the regime $n \le 2^{\kappa}$. It has total shares size equal to $\frac{n}{t+1}(t+e)\kappa + n\kappa$. There, the left summand corresponds to the Reed-Solomon encoding of the ciphertext, denoted as "Rabin's Information Dispersal Algorithm (IDA)". Whereas the right summand corresponds to the $n$ Shamir shares of the key. Thus, in the regime where $n > t \gg e$, then Definition 3 of CSSS requires size $\frac{n}{t}\left|\boldsymbol{S}\right|$, whereas the size of [Kra93, §3] is $\frac{n}{t+1}\left|\boldsymbol{S}\right| + \frac{n}{t}\left|\boldsymbol{S}\right|$, which is roughly twice larger. The terminology, of Constant Size, comes from the fact that $\frac{n}{t} \cong \frac{n}{t+1}$ is the best expectable expansion rate, since equal to the one of an $(n-(t+1))$-erasure-correcting Reed Solomon encoding of the plaintext secret. On the other hand, in the regime where $e \gg n > t$, then Def. 3 does not require a total size smaller than [Kra93].

## 3 Privacy of Desai's CSSS under Standard Assumptions

Let us recall in our notations the scheme denoted as "CTRT" in [Des00, Theorem 2], which is a CSSS with threshold $t := n-1$ out of $n$ shares. It requires $bc_{\bullet}$ an Ideal Cipher, the definition of which is recalled in §A.2.3. For simplicity of the description we consider it of blocksize $\beta = \kappa$. CTRT is described in [Des00] for a secret of size $(n-1)\kappa$, i.e., $\boldsymbol{S} = (S_1, \ldots, S_{n-1})$ where $\forall i$, $\left|S_i\right| = \kappa$. Sample $K \xleftarrow{\$} \{0,1\}^{\kappa}$ then output:

$$(5) \qquad Sh_i := bc_K(i) + S_i \ \text{ for } i \in [n-1], \ \text{ and } \ Sh_n := K + \sum_{i=1}^{n-1} Sh_i$$

For secrets of longer size $(n-1+e)\kappa$ for some integer $e > 0$, anticipating on our generalization of §4, one may generalize CTRT as: compute $n$ shares $(Sh_i^{sk})_{i\in[n]}$ from $(S_1, \ldots, S_{n-1})$ as previously, which are denoted as the *Shares of the Key*, following the observation made in §1.2.2. Then compute the vector $\left\{S_j + bc_K(j)\,,\, j \in [n, \ldots, e]\right\}$, divide it arbitrarily in $n$ strings $(Sh'_i)_{i\in[n]}$ denoted *Extra-Shares*, then append them to the shares of the key.

Let us show that, when we replace the Ideal Cipher $bc_{\bullet}$ by a specific plain PRF, then CTRT enables an adversary to efficiently *distinguish between two chosen secrets, when given all but one share*. By definition this then violates the privacy guarantee denoted "CSS" in [BR07] (and thus, a fortiori, our slightly stronger "Computational Privacy" guarantee of Def. 2). The PRF that we consider is known as the Even-Mansour cipher, denoted $BC_{\bullet}$, which is built from a public Ideal Permutation $\pi$ and a key $K$, and defined by $BC_K(i) := \pi(K+i)+K$. [Let us recall that $BC_{\bullet}$ is proven to be a Pseudorandom permutation, the definition of which is recalled in §A.2.2, secure up to the birthday bound. Precisely, given a $\kappa$-bit random key $K$, it requires at least $2^{\kappa/2}$ encryptions or offline computations of the underlying permutation to distinguish $BC_K$ from an ideal permutation. Hence $BC_{\bullet}$ is indeed a PRF, by the PRP/PRF switching Lemma, recalled in §A.2.2.] The attack is as follows. Consider an adversary $\mathcal{A}$ which chooses two arbitrary secrets $\boldsymbol{S}^{(b)} \in \{0,1\}^{(n-1)\kappa}$ for $b \in \{L, R\}$ (for "left" and "right"), such that: $n \geq 3$, $S_1^L = S_1^R = 0$, and for at least one index $j \in \{2, \ldots, n-1\}$ we have $S_j^L \neq S_j^R$. Consider a secret bit $b \in \{L, R\}$ sampled away from $\mathcal{A}$. $\mathcal{A}$ is returned all shares of $CTRT(\boldsymbol{S}^{(b)})$ except $Sh_1^{(b)}$. Notice we could have chosen any other missing index in $[n-1]$, provided a change of indices in the strategy above to choose $\boldsymbol{S}^L$ and $\boldsymbol{S}^R$. The shares are such that

$$(6) \qquad Sh_i^{(b)} := \pi(K+i) + K \text{ for } i \in [n-1], \text{ and } Sh_n^{(b)} := K + \sum_{i=1}^{n-1} Sh_i^{(b)} .$$

The adversary is able to compute $Sh_n^{(b)} - \sum_{i=2}^{n-1} Sh_i = K + Sh_1^{(b)} = K + (\pi(K+1)+K) = \pi(K+1)$ and thus can simply compute the inverse of $\pi$ on this value to recover $K$. Then it can decrypt $Sh_j^{(b)}$ into $S_j^{(b)}$ and thus determine $b$. This attack shows that Desai's CSSS necessarily relies on the cipher $bc_{\bullet}$ being ideal, which is a stronger requirement than pseudorandomness.

## 4 Threshold CSSS, with a Single Idealized Primitive Call

We consider $\mathrm{f} : \{0,1\}^\kappa \longrightarrow \{0,1\}^\kappa$ a Short Random Oracle. It is our idealized model of a "key derivation" function. Notice that our result holds, with essentially the same proof (adapt Game 4 below), when f is alternatively an ideal permutation oracle. We now consider any Pseudorandom Function $\mathrm{F}_{\bullet}$, e.g., a plain PRP, see §A.2. Denote $\phi$ a $(n,t)$-PSSUF (Def. 1). For sake of size we specify it to be the additive sharing if $t = n-1$; and otherwise the Shamir secret sharing if $t < n-1$. Denote $RS : (\{0,1\}^*)^e \longrightarrow (\{0,1\}^*)^n$ the Reed-Solomon encoding into $n$ coordinates, with resilience against the erasure of any $n-(t+1)$ coordinates. Notice that it is the identity function if $t+1 = n$. On input a secret $\boldsymbol{S} = (S_i)_{i\in[t+e]} \in \{0,1\}^{(t+e)\kappa}$ for some integer $e \geq 0$:

- sample the *Secret-Shared key*: $K^{\mathrm{ss}} \xleftarrow{\$} \{0,1\}^\kappa$;
- derive the *PRF Key*: $K^{\mathrm{prf}} := \mathrm{f}(K^{\mathrm{ss}})$;
- generate the *Randomness*: $\boldsymbol{R} := \big(R_j := S_j + \mathrm{F}_{K_m^{\mathrm{prf}}}(j)\big)_{j\in[t]} \in \{0,1\}^{t\kappa}$;
- compute the *Shares of the Key*: $(Sh_i^{\mathrm{sk}})_{i\in[n]} := \phi(K^{\mathrm{ss}}, \boldsymbol{R}) \in \{0,1\}^{n\kappa}$;
- compute the *Extra Shares*: $(Sh'_i)_{i\in[n]} := RS\Big(\big\{S_j + \mathrm{F}_{K^{\mathrm{prf}}}(j)\big\}_{j\in[t+1,\dots,t+e]}\Big)$;
- output the *Shares*, which are the concatenations $Sh_i := Sh_i^{\mathrm{sk}} \| Sh'_i$ for $i \in [n]$.

For reconstruction from any $(t+1)$-subset $\mathcal{H} \subset [n]$ of shares: decode the $(Sh'_i)_{i\in\mathcal{H}}$ into the vector $\big\{S_j + \mathrm{F}_{K^{\mathrm{prf}}}(j)\big\}_{j\in[t+1,\dots,e]}$; reconstruct $(K^{\mathrm{ss}}, \boldsymbol{R})$ from the $(Sh_i)_{i\in\mathcal{H}}$, where recovery of $\boldsymbol{R}$ is enabled by Full Reconstruction; deduce $K^{\mathrm{prf}} = \mathrm{f}(K^{\mathrm{ss}})$; decrypt all $S_j + \mathrm{F}_{K^{\mathrm{prf}}}(j)$ into $S_j$ for $j \in [t+e]$.

Notice that the CTRT scheme of [Des00], described in §3, would be recovered by setting: $K^{\mathrm{prf}} := K^{\mathrm{ss}}$, instead of $K^{\mathrm{prf}} := \mathrm{f}(K^{\mathrm{ss}})$; $\mathrm{F}_{\bullet}$ instantiated with an ideal cipher $bc_{\bullet}$, and $\boldsymbol{S} \in \{0,1\}^{(n-1)\kappa}$.

Our construction now applies an idealized construction only once, just on the key, and not, as in CTRT, on a long (counter) sequence of same length as the whole secret. Our construction thus repairs the problem evidenced in §3, in that the key $K^{\mathrm{prf}}$ is sampled independently of $K^{\mathrm{ss}}$, thanks to the idealized key derivation function f. Thus, nonwithstanding $K^{\mathrm{ss}}$ is the shared secret, we can nevertheless apply the PRF property of $\mathrm{F}_{\bullet}$ with respect to $K^{\mathrm{prf}}$, to conclude that $\boldsymbol{R}$ is an eligible source of randomness to secret-share $K^{\mathrm{ss}}$. In the following formal statement and proof, this last argument will be the transition to Game 2, while the set of events where it fails will be the transition to Game 3.

**Theorem 4.** *The previous scheme, denoted $\boldsymbol{\Sigma}$, is a CSSS. More precisely, denote* $\mathrm{Adv}^{\mathrm{F}}_{PRF}$ *the PRF advantage of* $\mathrm{F}_{\bullet}$*, then the computational privacy advantage, of Definition 2, for any adversary $\mathcal{A}$ running in time $t$, making $M$ queries to its oracle and $M'$ queries to* $\mathrm{f}$*, is at most:*

$$M \mathrm{Adv}^{\mathrm{F}_{\bullet}}_{PRF}(t+e) + (M^2 + M.M')/2^{\kappa} \ . \tag{7}$$

*Proof.* For $m \in [M]$, denote $\mathcal{I}^{(m)} \subset [n]$ the $t$ indices of the shares that the adversary asks to see in response to its $m$-th query. Notice that the queries of the Adversary to the public keyed function $\mathrm{F}_{\bullet}$ are incorporated in its time budget $\tau$. We proceed by a series of games starting from **Game 0**, where the adversary is facing the dummy oracle \$ of length $(\{0,1\}^{\kappa})^{t}$, up to $\mathcal{O}\boldsymbol{\Sigma}$, and we bound the distinguishing advantage of each step. Let $\mathcal{A}^{i}$ be the adversary interacting with the oracle of Game $i$.

***Game 1*** Now for each new query $m \in [M]$, the oracle: samples uniformly at random a pair of keys in its head $\left(K^{\mathrm{ss}}_{(m)}, K^{\mathrm{prf}}_{(m)}\right)$, each in $\{0,1\}^{\kappa}$; generates the shares $Sh_i^{(m)}$ as in the scheme, except that in the formulas, all variables $\mathrm{F}_{K^{\mathrm{prf}}_{(m)}}(j) \in \{0,1\}^{\kappa}$ for $j \in [t+e]$ for $m \in [M]$ are replaced by random uniform samplings, we denote them $r^{(m)}(j) \in \{0,1\}^{\kappa}$. In particular the keys $K^{\mathrm{prf}}_{(m)}$ are not used at all. Thus in detail, the shares $(Sh_i^{(m)})_{i\in\mathcal{I}^{(m)}}$ received by the adversary are the concatenation:

– of the $(Sh_i'^{(m)})_{i\in\mathcal{I}^{(m)}}$, which are $t$ codewords of a Reed-Solomon encoding of $\{$the one-time pad of part of part of the chosen secret vector $(S_j^{(m)})_{j\in[t+1,\dots,e]}$ with uniform randomness $\left(r^{(m)}(j)\right)_{j\in[t+1,\dots,e]}$ $\}$. Since any $t$ (and even $t+1$) evaluations at distinct points of a uniform random polynomial of degree $t$ vary uniformly at random, they vary uniformly at random;
– and of the $(Sh_i^{\mathrm{sk},(m)})_{i\in\mathcal{I}^{(m)}}$, which are $t$ secret shares of $K^{\mathrm{ss}}_{(m)}$ generated with $\phi$, using a source of randomness which is $\left(r^{(m)}(j)\right)_{j\in[t]}$. In particular, this source of randomness varies uniformly independently of $K^{\mathrm{ss}}_{(m)}$ and of the rest of the view of $\mathcal{A}$, i.e., of the $(Sh_i^{(m)})_{i\in\mathcal{I}^{(m)}}$. Thus, by Definition of a PSSU, the $(Sh_i^{\mathrm{sk},(m)})_{i\in\mathcal{I}^{(m)}}$ vary uniformly at random in $\{0,1\}^{t\kappa}$, independently from the $(Sh_i'^{(m)})_{i\in\mathcal{I}^{(m)}}$.

In conclusion, the statistical distribution of the view of the adversary is the same as in Game 0, thus

$$\Pr(\mathcal{A}^1 \to 0) - \Pr(\mathcal{A}^0 \to 0) = 0$$

***Game 2*** differs from Game 1 as the $\left(r^{(m)}(j)\right)_{j\in[t+e]}$ are computed as $\left(\mathrm{F}_{K^{\mathrm{prf}}_{(m)}}(j)\right)_{j\in[t+e]}$ as in the scheme. To bound the distinguishing advantage, we proceed by the following cascade of intermediary games $m \in \{0, 1 \ldots, M\}$ where $m = 0$ corresponds to Game 1 and, for $m \geq 1$, in the $m$-th game, all queries up to $m$ are replaced as previously.

For each $m \in \{0, \ldots, M-1\}$, we construct as follows an adversary $\mathcal{A}^{(m)}_{PRF}$ against the PRF game for F, whose distinguishing advantage is thus upper-bounded by $\mathrm{Adv}^{\mathrm{F}}_{PRF}$. Our goal is to show that its advantage is at least as large as the maximum one between intermediary games $m$ and $m{+}1$, as follows. $\mathcal{A}^{(m)}_{PRF}$ runs a copy of an adversary $\mathcal{A}^{(m)}$ against the $m$-th intermediary game. For all $m' \in [M]$, $\mathcal{A}^{(m)}_{PRF}$ samples in its head a key pair $(K^{\mathrm{ss}}_{(m')}, K^{\mathrm{prf}}_{(m')})$.

- If $m' < m$, $\mathcal{A}^{(m)}_{PRF}$ answers the query as in Game 3, i.e., using $\big(\mathrm{F}_{K^{\mathrm{prf}}_{(m')}}(j)\big)_{j \in [t+e]}$.
- If $m' = m$, $\mathcal{A}^{(m)}_{PRF}$ answers the query using $\big(chall(j)\big)_{j \in [t+e]}$ where $chall(j)$ is obtained by querying on $j$ its PRF oracle.
- If $m' > m$, $\mathcal{A}^{(m)}_{PRF}$ answers the query as in Game 2, i.e., using $\big(r^{(m)}(j)\big)_{j \in [t+e]}$ sampled uniformly at random.

***Game 3*** differs from Game 2 by two sorts of events, denoted $\mathcal{G}$ as "guess": *Either*, for each new query $m \in [M]$, if $K^{\mathrm{ss}}_{(m)}$ is equal to one of the previously sampled keys $K^{\mathrm{ss}}_{(m')}$, then: replace $K^{\mathrm{prf}}_{(m)}$ by $K^{\mathrm{prf}}_{(m')}$, and return the same output as the one of the $m'$-th query. By the birthday paradox for the keys $(K^{\mathrm{ss}}_{(m)})_{m \in [M]}$, the total probability of these events is upper-bounded by $M^2/2^{\kappa}$.

*Or*, when f is queried on some previously sampled key $K^{\mathrm{ss}}_{m}$ then $f$ outputs $K^{\mathrm{prf}}_{m}$ instead of uniformly at random in $\{0,1\}^{\kappa}$; and conversely, when some $K^{\mathrm{ss}}_{(m)}$ is sampled equal a value previously requested to f, then $K^{\mathrm{prf}}_{(m)}$ is set to the previous output of $f$ on this value, instead of uniformly at random in $\{0,1\}^{\kappa}$. By the (generalized) birthday paradox, the total probability of both sub-sorts of events is bounded by $M.M'/2^{\kappa}$.

We thus have the actual scheme oracle $\mathcal{O}\boldsymbol{\Sigma}$.

## 5 Shorter ESKE, from Standard assumptions

***Definition 5.*** Let $t < n$ be fixed positive integers. A $(n,t)$-ESKE scheme is an efficiently computable randomized transformation $\boldsymbol{\mathcal{E}}$ that takes as input a secret key $K^{\mathrm{acc}}$ and a plaintext $\boldsymbol{S}$. It returns $n$ strings $(Sh_i)_{i \in [n]}$, denoted "shares", of bitsizes $|Sh_i|$ depending only on $\big|\boldsymbol{S}\big|$, such that $\boldsymbol{S}$ is efficiently reconstructible given any $t{+}1$ shares and $K^{\mathrm{acc}}$. Privacy should be insured in two separate cases:

(1) We require IND-CPA as in [KL14, Def 3.22], for multiple challenges. Namely, we consider the following two oracles $\mathcal{O}\boldsymbol{\mathcal{E}}^L$ and $\mathcal{O}\boldsymbol{\mathcal{E}}^R$, which stand for "left" and "right". They both sample a secret $K^{\mathrm{acc}} \xleftarrow{\$} \{0,1\}^{\kappa}$, then answer up to $M$ queries, each query consisting in a pair of plaintexts $(\boldsymbol{S}^L_{(m)}, \boldsymbol{S}^R_{(m)})$ of same lengths. On every query, $\mathcal{O}\boldsymbol{\mathcal{E}}^L$ returns $\boldsymbol{\mathcal{E}}(\boldsymbol{S}^L)$ while $\mathcal{O}\boldsymbol{\mathcal{E}}^R$ returns $\boldsymbol{\mathcal{E}}(\boldsymbol{S}^R)$. We require that for any (polynomial) length $\big|\boldsymbol{S}\big|$, the maximum distinguishing

advantage, defined by:

$$\max_{\mathcal{A}} \Pr(\mathcal{A}^{\mathcal{O}\boldsymbol{\mathcal{E}}^L} \to 0) - \Pr(\mathcal{A}^{\boldsymbol{\mathcal{E}}^R} \to 0) \tag{8}$$

over all adversaries $\mathcal{A}$ running in time $\tau$ and making at most $M$ oracle queries, be negligible.

(2) For any $\mathcal{A}$ that knows the secret key $K^{\text{acc}}$, the scheme should be secure as a $(n,t)$-CSS, Definition 2.

**Theorem 6.** *Consider the scheme of §4, with the two following modifications: initially sample $K^{\text{acc}} \xleftarrow{\$} \{0,1\}^\kappa$ then, on every input $\boldsymbol{S}$, modify the second step [derive] as: $K^{\text{prf}} := \mathrm{f}(K^{\text{acc}} + K^{\text{ss}})$. Then it is an ESKE, with same distinguishing advantage as given by equation* (7)*, in both cases (1) and (2).*

*Proof.* (2) The proof of privacy for the case where $\mathcal{A}$ knows $K^{\text{acc}}$ is the same as for Theorem 4 where we define the public random oracle as $\mathrm{f}'(x) = \mathrm{f}(K^{\text{acc}} + x)$. Indeed if f is a public random oracle then also is f$'$ even when $K^{\text{acc}}$ is known.

(1) To prove privacy when $\mathcal{A}$ gets all the shares but does not know $K^{\text{acc}}$, let us prove the following stronger indistinguishability between: all shares of a chosen plaintext, and a sample in a fixed distribution. Namely, we bound the maximum distinguishing advantage between the two following oracles. The actual scheme oracle $\mathcal{O}\boldsymbol{\mathcal{E}}$ returns the actual $\boldsymbol{\mathcal{E}}(\boldsymbol{S})$ on every query $\boldsymbol{S}$. The $\phi$-dummy oracle $\mathcal{O}\phi^\$$ has the same interface but on every query, returns a sample in the following distribution: $\mathcal{D}^\phi := \{\phi(K^{\text{ss}}, \boldsymbol{R}) \text{ for } (K^{\text{ss}}, \boldsymbol{R}) \xleftarrow{\$} \{0,1\}^{(t+1)\kappa}\}$, independently of the query $\boldsymbol{S}$ nor if it was already queried or not. The adversary makes $M$ queries to its respective game oracle, i.e., $\mathcal{O}\boldsymbol{\mathcal{E}}$ or $\mathcal{O}\phi^\$$, and has also access to the oracle f to which he makes $M'$ queries. [Finally, we give strictly more information to the adversary, in that, against the oracle $\mathcal{O}\boldsymbol{\mathcal{E}}$, then *after* he has done *all* oracle interactions, both to $\mathcal{O}\boldsymbol{\mathcal{E}}$ *and* to f, then we give to it $K^{\text{acc}}$ before it decides on an output. Symmetrically, when facing the dummy oracle $\mathcal{O}\phi^\$$, we give to $\mathcal{A}$ a randomly sampled $K^{\text{acc}}$ after all interactions, although this provides no added information, the goal is to keep the same interface as with $\mathcal{O}\boldsymbol{\mathcal{E}}$.]

***Game 0*** The adversary is facing $\mathcal{O}\phi^\$$. Namely: on every query, $\mathcal{O}\phi^\$$ outputs a sample in $\mathcal{D}^\phi$; then after all oracle interactions, the game randomly samples $K^{\text{acc}} \xleftarrow{\$} \{0,1\}^\kappa$ and gives it to the adversary before it makes its decision.

***Game 1*** Now, the oracle $\mathcal{O}_1$ replies to each query $m \in [M]$ as in the actual scheme, excepted that the $\left(\mathrm{F}_{K^{\text{prf}}_{(m)}}(j)\right)_{j\in[t+e]}$ are computed with a $K^{\text{prf}}_{(m)} \xleftarrow{\$} \{0,1\}^\kappa$ sampled uniformly at random for each $m$. The distinguishing advantage with Game 0 is upper-bounded by $M\mathrm{Adv}^{\mathrm{F}_\bullet}_{PRF}(t+e)$, following the same cascade of games as for Game 2 in the proof of Thm 4, excepted that all $n$ shares are given to $\mathcal{A}^{(m)}$ instead of only those in $\mathcal{I}^{(m)}$.

***Game 2*** is the real scheme $\mathcal{O}\boldsymbol{\Sigma}$ where, after oracle interactions, the game outputs $K^{\text{acc}}$. We *Claim* that the advantage of distinguishing Game 1 and Game 2 is bounded by total probability of the following two sorts of events, denoted $\mathcal{G}$:

*Either*, when drawing $K^{\mathrm{ss}}_{(m)}$ for some $m \in [M]$, we have that it collides with a previous value, that is $\exists m' < m : K^{\mathrm{ss}}_{(m)} = K^{\mathrm{ss}}_{(m')}$.

*Or*, when drawing $K^{\mathrm{acc}}$, we have that the adversary has made a previous query $k$ to f such that $\exists m \in [M] : K^{\mathrm{ss}}_{(m)} + K^{\mathrm{acc}} = k$. We bound the probability of the latter with the randomness of $K^{\mathrm{acc}}$. Overall, we obtain $\Pr(\mathcal{G}) \leq (M^2 + M' \cdot M)/2^{\kappa}$.

Finally, the *Claim* is because, if $\mathcal{G}$ does not occur, then the random sampling of $K^{\mathrm{prf}}$ acts the same way as the random oracle f since $K^{\mathrm{acc}} + K^{\mathrm{ss}}_{(m)}$ is a fresh input. Hence when $\mathcal{G}$ does not occur, Game 1 and Game 2 produce identically distributed views.

## 6 Access Revocation from any Black Box ESKE

We formalize the new access granting and revocation scheme, presented in §1.2.4, along with its properties, as security games against two adversaries. The first one is denoted "inner adversary" $\mathcal{A}^{in}$ and models the cloud provider: it has unlimited dowload and storage capacity but is not given access to any secret. The security game against $\mathcal{A}^{in}$ is the one of case (1.) of Definition 5 of an ESKE.

The second one is denoted the "revocation" adversary $\mathcal{A}^{rev}$ and models a person to which access to multiple secrets was initially granted, and then is revoked as presented in §1.2.4. $\mathcal{A}^{rev}$ must distinguish between the following two oracles $\mathcal{O}^L$ and $\mathcal{O}^R$, in the following game. We denote $\boldsymbol{\mathcal{E}}$ any $(n, n-1)$-ESKE scheme. We denote $\mathrm{Adv}_{\mathcal{E}}(M)$ an upper bound on the distinguishing advantage of any adversary in the case (2.) of definition 5. We denote by $E$ any symmetric encryption scheme, of which we denote $Adv^E(M)$ an upper bound in the game consisting in distinguishing actual encryptions of a sequence of $M$ chosen plaintexts of size one block, with $M$ random strings of same lengths.

- The game oracle samples $K^{\mathrm{acc}} \xleftarrow{\$} \{0,1\}^{\kappa}$ once for all, then gives it to $\mathcal{A}^{rev}$.
- $\mathcal{A}^{rev}$ gives to the game oracle a sequence of $M$ pairs of plaintexts $(\boldsymbol{S}^L_{(m)}, \boldsymbol{S}^R_{(m)})_{i\in[M]}$, of same sizes. For each $m \in [M]$, the $\mathcal{O}^L$ oracle returns $\mathcal{E}(K^{\mathrm{acc}}, \boldsymbol{S}^L)$, while the $\mathcal{O}^R$ oracle returns $\mathcal{E}(K^{\mathrm{acc}}, \boldsymbol{S}^R)$.
- For each index $m \in [M]$, $\mathcal{A}^{rev}$ can adaptively query the game oracle to reveal up to $n_{loc} \leq n$ shares.
- The oracle then: samples a key $K^{\mathrm{miss}}$ for $E$, selects at random $n_{miss}$ share indices at random in $[n]$, independently of the previous queries, and replaces these shares by encryptions of them under $K^{\mathrm{miss}}$, to which it appends a tag informing of their re-encryption.
- $\mathcal{A}^{rev}$ can then query the oracle to obtain all shares, of which the $n_{miss}$ re-encrypted ones instead of the original $n_{miss}$ ones. $\mathcal{A}$ outputs a bit $b'$, and wins if $b = b'$.

Notice that the previous game implicitely assumes that $\mathcal{A}^{rev}$ does not communicate with $\mathcal{A}^{in}$, i.e., can not retrieve the missing blocks as they were before revocation, nor does it communicate with persons still accredited after its revocation, i.e., is not given the new key $K^{miss}$.

**Theorem 7.** *The advantage of any adversary in the previous game is lower than* $M(n_{loc}/n)^{n_{miss}} + 2\mathrm{Adv}^{E}(M) + \mathrm{Adv}^{\boldsymbol{\mathcal{E}}}(M)$.

*Proof.* Let us fix an index $m \in [M]$. Consider the event $\mathcal{G}_m$ such that *all* the $n_{miss}$ shares indices which were re-encrypted, are included in the $n_{loc}$ shares indices. Since the choice of blocks re-encrypted is independant from the queries, the probability of this event is lower than $(n_{loc}/n)^{n_{miss}}$. Then, the probability of the union of these events for all indices $m \in [M]$ is lower than $M(n_{loc}/n)^{n_{miss}}$. Outside of these events, we have that, for each $m$, the view of the adversary is all shares, of which at least one re-encrypted under $K^{miss}$, of which it was not given the value before re-encryption. Let us give more power to the adversary: we let it choose one missing share index $j_m$ for each $m$, while giving to it all other shares (even the ones not in the $n_{loc}$ previously queried ones). Let us now consider the hybrid game where, for each $m$, instead of an actual re-encryption of the $j_m$-th share, the adversary is given a random string of same length. Then, for both values of $b$, the output of the adversary in the hybrid game differs by at most by $Adv^{E}(M)$ from the previous game. Finally, the distinguishing advantage of the adversary $\mathcal{A}'$ in this hybrid game is upper bounded by $Adv^{\mathcal{E}}(M)$. This follows from the trivial reduction where an adversary $\mathcal{A}$ in the case (2) of Definition 5 would play the role of the oracle towards $\mathcal{A}'$, by sampling the missing share uniformly at random.

## 7 Bridging the gap between two conflicting AONTs

We start with Stinson's [Sti01] definition of an AONT, where an input vector of information is multiplied by a random matrix, which has for goal to prevent an adversary to recover a single element of the vector even if he has all but one blocks of the multiplication result. Let $\mathbf{F}_q$ denote the finite field with $q$ elements.

***Definition 8.*** A Stinson's AONT of size $\ell + 1$ is an invertible linear transform $\phi : \mathbf{F}_q^{\ell+1} \longrightarrow \mathbf{F}_q^{\ell+1}$ such that the following holds. For any index $i_0$, and for any *image* vector: $\overrightarrow{y} := (y_1, \ldots, y_{i_0-1}, Y, y_{i_0+1}, \ldots, y_{\ell+1})$ where the $y_i$ for $i \neq i_0$ are *fixed*, what have that *if* $Y$ varies uniformly at random, the input coordinate $x_j$ of the *pre-image* vector $\overrightarrow{x} := \phi^{-1}(\overrightarrow{y})$ varies uniformly at random for any *single* index $j$.

Note that this definition does not prevent *two* input coordinates $x_u$ and $x_v$ from being correlated, which then *gives some information about the whole input vector*. Another example is that, taking $\phi$ equal to the matrix of [Sti01, Cor 2.3], then an adversary learning the first coordinates of the output veector, thus learns a fixed public linear combination on the blocks of the input vector. Thus it can distinguish between two chosen inputs.

A Stinson's AONT can be computed over $\mathbf{F}_4 = \mathbf{F}_2(\alpha)$ for any size $\ell$: the matrix with ones, except $\alpha$ on the diagonal. We make the straightforward abuse of notation consisting in applying a linear transformation $\phi \in \mathbf{F}_q^{(\ell+1)\times(\ell+1)}$ to vectors of $(\ell+1)$ *blocks* to output a vector of $(\ell+1)$ *blocks.* The actual transformation that this means is: interpret the input sequence of bits, as a sequence of elements in $\mathbf{F}_q$ ($\lceil \log_2(q) \rceil$ bits encode one element of $\mathbf{F}_q$). Possibly pad the input to have an exact number of elements of $\mathbf{F}_q$. Then apply $\phi$ to every subsequence of $(\ell+1)$ elements of $\mathbf{F}_q$.

**Theorem 9.** *Let us consider an invertible linear map $\phi$ acting on vectors of size $\ell+1$. We consider an Ideal Blockcipher $bc_{\bullet}$ and any key $K \in \{0,1\}^{\kappa}$. Consider the following scheme, consisting in counter-mode encryption followed by a Stinson's linear AONT:*

- *On every input $\boldsymbol{S}$ of $\ell$ blocks: $\boldsymbol{S}_1, \ldots, \boldsymbol{S}_\ell$, each of size $\beta = \kappa$;*
- *sample* $\mathsf{iv} \xleftarrow{\$} \{0,1\}^{|K|}$ *(denoted as "initialization vector")*
- *compute the blocks $F_0 := \mathsf{iv}$ and $F_i := bc_K(\mathsf{iv}+i) + \boldsymbol{S}_i$ for $i = 1 \ldots \ell$*
- *output $\boldsymbol{\Sigma}(\boldsymbol{S}) := Sh_1, \ldots, Sh_{\ell+1}$ the $\ell+1$ output blocks of $\phi(F_0, \ldots, F_\ell)$.*

*Consider a model where this key $K$ is* given *to the adversary. Namely, $\left(bc_K, bc_K^{-1}\right)$ is now a* publicly accessible *Ideal Permutation Oracle, denoted from now $\left(\pi, \pi^{-1}\right)$. Then, in this model, we have:*

$$\Big\{ \boldsymbol{\Sigma} \textit{ defines a CSSS} \Big\} \text{ if and only if } \Big\{ \phi \textit{ is a Stinson's linear AONT} \Big\}.$$

We now prove the Theorem. We denote $M_\phi$ the matrix of $\phi$, acting on the right on line vectors. By construction we have:

$$\boldsymbol{\Sigma}(\boldsymbol{S}) := (Sh_1, \ldots, Sh_{\ell+1}) = (F_0, \ldots, F_\ell) \cdot M_\phi \quad \text{thus:} \tag{9}$$

$$(F_0, \ldots, F_\ell) = \boldsymbol{\Sigma}(\boldsymbol{S}) \cdot M_\phi^{-1} \tag{10}$$

### 7.1 Necessary Condition on $\phi$

The result follows from the next Lemma, since [Sti01, Theorem 2.1] then implies that $\phi$ is a Stinson's linear AONT.

***Lemma 10.*** If, *in this model where the Adversary is given access to* $\left(\pi, \pi^{-1}\right)$, *we have that $\boldsymbol{\Sigma}$ defines a CSSS,* then, *all entries of $M_\phi^{-1}$ are nonzero.*

*Proof.* Suppose by contradiction that there exists an entry $M_\phi^{-1}$ containing a zero. Denote $(row, col)$ its row and column indices. The key point is that this then implies, from Equation (10), that the block $F_{col}$ is a linear combination of $\ell+1-1 = \ell$ shares, namely, of the $\{Sh_i\}_{i \neq row}$. Both $(row, col)$ and the coefficients of the combination are fixed and public since $M_\phi^{-1}$ is public. Then $\mathcal{A}$ has the following winning strategy in the game of Definition 2. He chooses $\boldsymbol{S} := 0$ equal to a bit string with 0's only. Then, asks to be challenged with share indices $j \in \{1, \ldots \ell+1\} \backslash \{row\}$. Denote $Sh_j$ the shares that he obtains. He has to guess if they are bogus, or actual outputs from $\boldsymbol{\Sigma}(0)$. He first reconstructs a speculative

$F'_{col}$ using the shares $\{Sh'_i\}_{i \neq row}$ he received. Then he computes a speculative initialization vector $\mathsf{iv}' := \pi^{-1}(F'_{col})$. Finally, using this speculative $\mathsf{iv}'$ he can compute all the speculative ciphertext blocks $F'_0 := \mathsf{iv}$ and $F'_j := \pi(\mathsf{iv}' + j)$ for all $j \in \{1, \ldots, \ell\}$. And finally deduce speculative shares $Sh'_i$ by applying $\phi$. If they are equal to the ones $Sh_i$ received, then he outputs $b = 0$: that he received the real ones. Otherwise he outputs $b = 1$: that he received bogus shares.

## 7.2 Sufficient Condition on $\phi$

Let $\phi$ be a Stinson's linear AONT map acting on vectors of size $\ell + 1$. Consider any set $\mathcal{I_A}$ of $\ell$ columns in $M_\phi$. The matrix being invertible, these columns are of rank $\ell$. By elementary operations on columns, they can be put in column-wise strong echelon form $Ech_{\mathcal{I_A}}$, such that there are at least $\ell$ distinct pivot indices. Thus, there exists $\ell$ lines containing exactly a single 1 entry and 0 everywhere else, and such that these 1 are all on distinct columns:

$$Ech_{\mathcal{I_A}} = \left(\begin{array}{cccccc} 0 & 1 & 0 & 0 & . & 0 \\ 1 & 0 & 0 & 0 & . & 0 \\ 0 & 0 & 1 & 0 & . & 0 \\ \hline * & * & * & * & . & * \\ \hline 0 & 0 & 0 & 0 & . & 1 \\ . & . & . & . & . & . \\ 0 & 0 & 0 & 1 & . & 0 \end{array}\right) \tag{11}$$

The line with the $*$ is the only one not to be a pivot line, let $i_*$ be its index.

***Lemma 11.*** *If $\phi$ be a Stinson's linear AONT then the $i_*$ line contains no zero entry.*

*Proof.* By contradiction, suppose that there is a zero entry, that is there exists a column of index $j_0$ such that $Ech_{i_*,j_0} = 0$. By definition of columnwise echelon form, this implies that the $j_0$ column contains *exactly* a single 1 (and 0 otherwise): denote $i_0$ this row index.

Denote $\overrightarrow{y} = (y_1, y_2, \ldots, y_{\ell+1})$ an image vector in $\mathbf{F}_q^{\ell+1}$. Notice that the elementary columns operations that transform $M_{\mathcal{I_A}}$ into $Ech_{\mathcal{I_A}}$ is the linear combination of the image values (that are in the $\ell$ indices in $\mathcal{I_A}$) needed to build the pre-image vector $\overrightarrow{x} := \phi^{-1}(\overrightarrow{y})$ which will further depend on the value whose index is not in $\mathcal{I_A}$ if and only if the $i_*$ entry is not zero. In particular, we have that the $i_0$ coordinate of pre-image, $x_{i_0}$, is independent from the image value whose index is not in $\mathcal{I_A}$. Thus this is incompatible with the requirement of Definition 8.

Let us deduce from Lemma 11 that $\phi$ being a linear AONT is a sufficient condition for $\boldsymbol{\Sigma}$ to be a $CSSS$. Consider the elementary invertible matrix of column operations $N_{\mathcal{I_A}}$, of size $\ell \times \ell$, that puts $M_\phi$ in echelon form: $Ech_{\mathcal{I_A}} = M_{\mathcal{I_A}} \cdot N_{\mathcal{I_A}}$ , where the subscript $\mathcal{I_A}$ means that we restrict to the column indices in $\mathcal{I_A}$. Then in the game of Definition 2, consider the situation where the adversary receives actual shares $Sh_i(\boldsymbol{S})$ of his chosen secret $\ell$, with $\mathcal{I_A}$ the set of $\ell$ indices that he chose to see. The adversary chooses $\boldsymbol{S}$ and an index $j$ and receives the key $k$

and $\ell$ shares. We note the block cipher under a known key as $\pi(\cdot)$ as it's akin to a public permutation. Let us notice that the initial vector $(F_0,\ldots,F_\ell)$ can be decomposed in a fixed known/chosen part, plus a variable part:

$$(F_0,\ldots,F_\ell) := \Big[0^\kappa, \boldsymbol{S}\Big] + \Big[\mathsf{iv}, \pi(\mathsf{iv}+1),\ldots,\pi(\mathsf{iv}+\ell)\Big] \tag{12}$$

Let us multiply the vector of shares received by the adversary by the *invertible* matrix $N_{\mathcal{I}_\mathcal{A}}$ on the right. We obtain the coordinates in $\mathcal{I}_\mathcal{A}$ of the vector $(F_0,\ldots,F_\ell)M_\phi N_\phi$ whose variable part is:

$$\overrightarrow{V}(\mathsf{iv})_\mathcal{A} := \Big[\mathsf{iv}, \pi(\mathsf{iv}+1),\ldots,\pi(\mathsf{iv}+\ell)\Big]_{\mathcal{I}_\mathcal{A}} \underbrace{M_{\mathcal{I}_\mathcal{A}} N_{\mathcal{I}_\mathcal{A}}}_{Ech_{\mathcal{I}_\mathcal{A}}} \tag{13}$$

Let us express it explicitly: let $e_0,\ldots,e_\ell$ (same $\ell$ indices as $\mathcal{I}_\mathcal{A}$) be the entries of the special line $i_*$ of $Ech_{\mathcal{I}_\mathcal{A}}$, proven to be all nonzero in Lemma 11. Let $i_0,\ldots,i_\ell$ (same indices as $\mathcal{I}_\mathcal{A}$) be the row indices of the successive pivots (the 1 entries) in the columns of the matrix $Ech_{\mathcal{I}_\mathcal{A}}$. They are by definition *different* from $i_*$. Then we have, abusing of notation "$\pi(\mathsf{iv}+0)$":= $\mathsf{iv}$:

$$\overrightarrow{V}(\mathsf{iv})_\mathcal{A} = \Big(e_1\pi(\mathsf{iv}+i_*) + \pi(\mathsf{iv}+i_1)\,, \\ e_2\pi(\mathsf{iv}+i_*) + \pi(\mathsf{iv}+i_2),\ldots, e_\ell\pi(\mathsf{iv}+i_*) + \pi(\mathsf{iv}+i_\ell)\Big)\,. \tag{14}$$

We can conclude by:

***Proposition 12.*** *Under the Ideal Permutation assumption for* $\left(\pi,\pi^{-1}\right)$, *the* $e_{i\in\mathcal{I}_\mathcal{A}}$ *being all nonzero, then the function* $\mathsf{iv} \longrightarrow \overrightarrow{V}(\mathsf{iv})_\mathcal{A}$ *is a PRG (§A.1).*

Informally, considering polynomial adversaries and only forward queries, the permutation $\pi$ behaves like a short random oracle. Thus, to distinguish $\overrightarrow{V}(\mathsf{iv})_\mathcal{A}$ from a PRG an adversary has to perform inverse queries, but relevant inverse queries are hard to find when all $e_i$ are nonzero, as there are only given sums of two seemingly random outputs, as formally proven in [KRM20, Prop. 1].

## References

[AGH12] Adi Akavia, Shafi Goldwasser, and Carmit Hazay. "Distributed Public Key Schemes Secure against Continual Leakage". In: *PODC*. 2012.

[Bac+16] Enrico Bacis et al. "Mix&Slice: Efficient Access Revocation in the Cloud". In: *CCS*. 2016.

[BB00] Mihir Bellare and Alexandra Boldyreva. "The Security of Chaffing and Winnowing". In: *Advances in Cryptology — ASIACRYPT 2000*. Springer Berlin Heidelberg, 2000.

[BGK20] Andrej Bogdanov, Siyao Guo, and Ilan Komargodski. "Threshold Secret Sharing Requires a Linear-Size Alphabet". In: *Theory Comput.* (2020).

[Boy99] Victor Boyko. "On the Security Properties of OAEP As an All-or-Nothing Transform". In: *CRYPTO*. 1999.

[BR06] Mihir Bellare and Phillip Rogaway. "The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs". In: *EUROCRYPT*. 2006.

[BR07] Mihir Bellare and Phillip Rogaway. "Robust Computational Secret Sharing and a Unified Account of Classical Secret-sharing Goals". In: *CCS*. 2007.

[Can+00] Ran Canetti et al. "Exposure-resilient Functions and All-or-nothing Transforms". In: *EUROCRYPT*. 2000.

[CDN15] Ronald Cramer, Ivan Bjerre Damgård, and Jesper Buus Nielsen. *Secure Multiparty Computation and Secret Sharing*. Cambridge University Press, 2015.

[Che+13] Y. Cheng et al. "Efficient revocation in ciphertext-policy attribute-based encryption based cryptographic cloud storage". In: *Journal of Zhejiang University SCIENCE C* (2013).

[CLM17] Liqun Chen, Thalia M. Laing, and Keith M. Martin. "Revisiting and Extending the AONT-RS Scheme: A Robust Computationally Secure Secret Sharing". In: *AFRICACRYPT*. 2017.

[Des00] Anand Desai. "The Security of All-or-Nothing Encryption: Protecting Against Exhaustive Key Search". In: *CRYPTO*. 2000.

[Dzi06] Stefan Dziembowski. "On Forward-Secure Storage". In: *CRYPTO*. 2006.

[Eve+17] Adam Everspaugh et al. "Key Rotation for Authenticated Encryption". In: *CRYPTO*. 2017.

[Gar+00] Juan A. Garay et al. "Secure Distributed Storage and Retrieval". In: *Theor. Comput. Sci.* 243.1-2 (July 2000), pp. 363–389.

[GWZ22] Jiaxin Guan and Daniel Wichs and Mark Zhandry. "Incompressible Cryptography". In: *EUROCRYPT*. 2022.

[HKT11] Thomas Holenstein, Robin Künzler, and Stefano Tessaro. "The Equivalence of the Random Oracle Model and the Ideal Cipher Model, Revisited". In: *STOC*. 2011.

[JSY99] Markus Jakobsson, Julien P. Stern, and Moti Yung. "Scramble All, Encrypt Small". In: *FSE*. 1999.

[KL14] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, second edition*. Chapman & Hall/CRC, 2014.

[KR19] Yael Tauman Kalai and Leonid Reyzin. "A Survey of Leakage-Resilient Cryptography". In: *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*. ACM, 2019.

[Kra93] Hugo Krawczyk. "Secret Sharing Made Short". In: *CRYPTO*. 1993.

[KRM20] Katarzyna Kapusta, Matthieu Rambaud, and Gerard Memmi. "Revisiting Shared Data Protection Against Key Exposure". In: *AsiaCCS*. 2020.

[KSLC18] G. O. Karame et al. "Securing Cloud Data under Key Exposure". In: *IEEE Transactions on Cloud Computing* (2018).

[Lab12] RSA Laboratories. *PKCS ♯1 v2.2: RSA cryptography standard*. Tech. rep. EMC Corporation, 2012.

[MPR96] S Matyas, Mohammad Peyravian, and Allen Roginsky. *Encryption of long blocks using a short-block encryption procedure*. Tech. rep. IBM. 1996.

[MS17] Steven Myers and Adam Shull. *Efficient Hybrid Proxy Re-Encryption for Practical Revocation and Key Rotation*. Cryptology ePrint Archive, Report 2017/833. `https://eprint.iacr.org/2017/833`. 2017.

[MS18] Steven Myers and Adam Shull. "Practical Revocation and Key Rotation". In: *CT-RSA*. 2018.

[Riv97] Ronald Rivest. "All-or-Nothing Encryption and the Package Transform". In: *FSE*. 1997.

[RP11] J. Resch and J. Plank. “AONT-RS: Blending Security and Performance in Dispersed Storage Systems”. In: *Proceedings of the 9th USENIX Conference on File and Stroage Technologies.* FAST’11. San Jose, California: USENIX Association, 2011. ISBN: 978-1-931971-82-9.

[Sho04] Victor Shoup. “Sequences of games: a tool for taming complexity in security proofs”. In: *IACR Cryptol. ePrint Arch.* 2004 (2004), p. 332.

[Sti01] Douglas R. Stinson. “Something About All or Nothing”. In: *Des. Codes Cryptogr.* (2001).

[Wan+17] C. Wang et al. “Insecurity of Cheng et al.’s Efficient Revocation in Ciphertext-Policy Attribute-Based Encryption Based Cryptographic Cloud Storage”. In: *ISPA & IUCC.* 2017.

## A PRG and Examples of PRF: Ideal Cipher and Pseudo-Random Permutations

### A.1 Pseudorandom Generator (PRG)

$PRG : X \in \{0,1\}^{\kappa} \longrightarrow \{0,1\}^{*}$, in the sense of [KL14, Def 3.15] can be defined as a deterministic function such that, when its input $X$ is sampled at random and hidden from the Adversary, then he cannot distinguish the output from a bit string sampled uniformly at random. in this paper it shows up with Long Output size: this setting is also known as “Streamcipher” and may be instantiated (against a polynomial adversary) with a keyed blockcipher running in counter mode. This notion will also show up when $X$ is the second input of a “pseudorandom function”, as defined just below.

### A.2 Examples of PRF: Pseudorandom Permutation and Ideal Ciphers.

**A Block $\in \{0,1\}^{\beta}$** is a string of bits of fixed size $\beta$, which for simplicity we consider to be a submultiple of $\kappa$, e.g., $\kappa = 256$ and $\beta = 128$.

**A.2.1 Ideal Permutation** is an oracle such that, when initialized, then it draws a permutation $\pi : \{0,1\}^{\beta} \longrightarrow \{0,1\}^{\beta}$ uniformly at random. That is, among all the $(2^{\beta})!$ possible permutations. Then the oracle anwser any evaluation queries to $\pi$ and to its inverse $\pi^{-1}$.

An Ideal Permutation can equivalently be defined as a random oracle with two interfaces $(\pi, \pi^{-1})$ as follows. On every input $x \in \{0,1\}^{\beta}$ to $\pi$ such that: $x$ was not not queried before to $\pi$ nor returned by a previous request to $\pi^{-1}$, then: sample uniformly at random an output in $\big\{$the values of $\{0,1\}^{\beta}$ which: were not output before, and were not queried before to $\pi^{-1}$ $\big\}$.

It has the symmetric behavior on every query $y \in \{0,1\}^{\beta}$ to $\pi^{-1}$ which was not queried before nor returned by $\pi$.

When $\pi$ or $\pi^{-1}$ is queried again on the same input, then it returns the same output.

When $\pi$ is queried on a previously output image $x$ of $\pi^{-1}$, then returns the input of $\pi^{-1}$ for which it previously returned $x$. The same holds symmetrically for $\pi^{-1}$ when queried on a previously output image $y$ of $\pi$.

**A.2.2 Pseudorandom Advantage of a Keyed Permutation (PRP), dubbed as "plain Blockcipher"** ([KL14, Definition 3.28] or [Sho04, §5.2]) Consider a function $BC_{\bullet} : \{0,1\}^{\kappa} \times \{0,1\}^{\beta} \longrightarrow \{0,1\}^{\beta}$ such that $BC_K := BC(K, \bullet)$ defines a permutation for every $K$. We denote it keyed permutation, dubbed as "blockcipher". Consider an adversary with black box forward access either to $BC_K$, where $K \overset{\$}{\leftarrow} \{0,1\}^{\kappa}$, or to a permutation $p$ drawn at random among the $(2^{\beta})!$ possibilities. *Forward access* means that the adversary has not access to the inverses, i.e., $BC_K^{-1}$ and $p^{-1}$. The PRP security of $BC$ is the following maximum over all $\mathcal{A}$ running in time $\tau$ and making $\sigma$ queries to its oracle:

$$\mathrm{PRP}^{BC}(\sigma) = \max_{\mathcal{A}} \mathbb{P}(\mathcal{A}^{BC_K} \to 0) - \mathbb{P}(\mathcal{A}^{p} \to 0)$$

If the PRP advantage is negligible, then we say that $BC$ is a *Pseudorandom Permutation* (PRP).
**"PRP/PRF" switching Lemma** ([BR06, §2]): a PRP is a PRF up to $\mathcal{O}(2^{\beta/2})$ oracle calls. More concretely, consider a publicly accessible PRP $BC_{\bullet}$, then

$$\mathrm{Adv}_{PRF}^{BC}(\sigma) \leq \mathrm{PRP}^{BC}(\sigma) + \sigma(\sigma - 1)/2^{\beta+1} \tag{15}$$

where $\mathrm{PRP}^{BC}(\sigma)$ is the PRP security of $BC$ against adversaries making $\sigma$ blockcipher calls and running in time $\tau$.

**A.2.3 An Ideal Cipher** in the sense of Shannon's, can be seen as a random oracle such that, when initialized, draws a function $bc_{\bullet} : \{0,1\}^{\kappa} \times \{0,1\}^{\beta} \longrightarrow \{0,1\}^{\beta}$ at random in the space of functions satisfying that, for all $K \in \{0,1\}^{\kappa}$, $bc_K := bc(K, \bullet)$ is a permutation. (This space has thus cardinality $\kappa^{(2^{\beta})!}$). The only difference between $bc_K$ and the "keyed-" Short Random Oracle $\mathrm{f}_K$ of length $\beta$, is that $bc_K$ is a permutation for every $K$. Equivalence of the Ideal Cipher and Random Oracle models, is proven in [HKT11].

A Ideal Cipher can equivalently be defined as a random oracle $\mathcal{O}_{bc_{\bullet}, bc_{\bullet}^{-1}}$ that answers two type of queries: evaluation of $bc_{\bullet}$ on some input $(K, m) \in \{0,1\}^{\kappa} \times \{0,1\}^{\beta}$; and evaluation of $bc_{\bullet}^{-1}$ on some input $(K, m) \in \{0,1\}^{\kappa} \times \{0,1\}^{\beta}$. When queried $(K, m)$ with a left argument $K$ that was not queried before, then it privately initializes a fresh Ideal Permutation Oracle $\mathcal{O}_{bc_K, bc_K^{-1}}$ as defined in A.2.1. Subsequently, $\mathcal{O}_{bc_{\bullet}, bc_{\bullet}^{-1}}$ passes to $\mathcal{O}_{bc_K, bc_K^{-1}}$ every request with left argument $K$, and forwards the result. By this latter equivalent definition, we thus have $\mathrm{PRP}^{bc_{\bullet}} = 0$.