Security of Linear Secret-Sharing Schemes against Mass Surveillance

Irene Giacomelli[1], Ruxandra F. Olimid[2], and Samuel Ranellucci[1]

[1] Department of Computer Science, Aarhus University, Denmark
[2] Department of Computer Science, University of Bucharest, Romania and Applied Cryptography Group, Orange

Abstract Following the line of work presented recently by Bellare, Paterson and Rogaway, we formalize and investigate the resistance of linear secret-sharing schemes to mass surveillance. This primitive is widely used to design IT systems in the modern computer world, and often it is implemented by a proprietary code that the provider ("big brother") could manipulate to covertly violate the privacy of the users (by implementing Algorithm-Substitution Attacks or ASAs). First, we formalize the security notion that expresses the goal of big brother and prove that for any linear secret-sharing scheme there exists an undetectable subversion of it that efficiently allows surveillance. Second, we formalize the security notion that assures that a sharing scheme is secure against ASAs and construct the first sharing scheme that meets this notion.

Keywords: linear secret-sharing, algorithm-substitution attack, mass surveillance, kleptography.

1 Introduction

The paper considers the possibility of mass surveillance by *algorithm-substitution attacks* (ASAs) against secret sharing. Secret-sharing generally refers to a method for splitting a secret into pieces (called *shares* of the secret) so that the secret can be reconstructed when a qualified set of shares are combined together (reconstruction property); on the other hand, unqualified sets of shares reveal no information about the original secret (privacy property). An ASA replaces the real sharing algorithm by a subverted version that allows a privileged party (*big brother*) to break privacy and reconstruct the secret from an unqualified sets of shares. Since secret sharing is widely used as building block for distributed protocols and systems, its insecurity against this kind of attack could have significant consequences. For example, big brother could mount ASA against a key backup system based on secret sharing, recover the private keys and break confidentiality (in order to maintain the same terminology as in the existing literature [1], we refer to this kind of scenario as *mass surveillance*).

Motivation. Applications for access control, key backup and recovery or secure storage systems sometimes implement proprietary piece of code to perform secret sharing [2–6]. Often, the security of the entire system relies on the privacy property of the underlying secret sharing scheme (e.g. access control systems grant permission only if a set of qualified shares are available for reconstruction). Therefore, mounting ASAs against such systems might lead to serious consequences: big brother can ruin access control, disclose private keys or learn secret data.

To exemplify, we focus on the scenario of long-term secure storage systems that use secret sharing to assure data confidentiality and availability. A client-side application runs a sharing algorithm to split data in share that are privately sent to a set of independent storage nodes, which can be located across different geographical and network areas, benefit of distinct protection mechanisms and even belong to various storage providers. To later access the stored data, the client application requests a qualified set of shares from several storage nodes and reconstructs. The architecture introduces multiple points of trust: reconstruction is possible only if the adversary breaks into several storage nodes and obtains a qualified set of shares; the architecture assumes no trust on individual storage providers, as no one can access the data using its own shares only. Now, suppose an undetectable ASA replaces the client-side application code with a subverted version designed by big brother that allows reconstruction from an unqualified sets of shares; if big brother is a storage provider, then it can perform surveillance by breaking the privacy property using the shares stored on its own servers; if big brother is an outsider, it can perform surveillance by only breaking into a few storage nodes, independently of the access structure. On the other hand, the client would like a guarantee that no ASAs will succeed, under the minimal detectability conditions.

Related Work. Kleptography was introduced by Young and Yung in the 90s to consider undetectable modifications to cryptosystems that deliberately provide trapdoor capabilities [7,8], as an extension to the existing notions of subliminal and convert channels [9,10]. Since then, kleptographic attacks have been designed for a wide range of cryptographic primitives and protocols. Despite the amount of work that has been done on the field, only recently Bellare, Paterson and Rogaway formalize the security notions in the settings of modern cryptography [1]. They set the terminology for ASAs (Asymmetric Substitution Attacks) and use a game-based approach to model both negative and positive results, i.e. when an adversary (big brother) can, respectively cannot perform surveillance without being detected. Their work focuses on symmetric encryption and highlights its impact on real-world systems. We follow their line of work, formalize and investigate the resistance of *linear secret-sharing* to mass surveillance. The security in this framework of other fundamental primitives has already been studied: see the recent work of Ateniese, Magri and Venturi [11] for a formal treatment of subversion-resilient signature schemes.

Modeling and Results. We assume that big brother subverts the sharing scheme embedding in it a strategy $\mathcal{T}$ and an encryption key. Big brother aims for a strong form of subversion, that disallows users from detecting ASAs or gain his abilities to perform surveillance even in case of reverse engineering. So, we

Table 1. Strong Subversion and Resilience Modeling

	Strong subversion (big brother's goal)	Strong resilience (users' goal)
Detection algorithm	$\mathtt{PK}, \mathcal{T}$; choose the secret	$\emptyset$; access SECRET oracle
Subverted algorithm	$\mathtt{PK}, \mathcal{T}$	$\mathtt{PK}, \mathtt{SK}, \mathcal{T}$

consider asymmetric ASAs, where big brother embeds into the code a public key $\mathtt{PK}$ and keeps the corresponding secret key $\mathtt{SK}$ private. In this strong surveillance model, the subverted algorithm has access to the public key $\mathtt{PK}$ and the strategy $\mathcal{T}$ and it remains undetectable by the users even if both $\mathtt{PK}$ and $\mathcal{T}$ are given to the detection algorithm (run by the users). We give additional power to the detection algorithm and allow it to choose the secret to be shared. This models big brother's goal to keep subversion hidden for all possible secrets and hence make the ASA undetectable. Following the strategy $\mathcal{T}$, big brother corrupts a set of unqualified parties and uses their shares to gain information about the secret. This is the framework we formalize in Section 3, where we also show our negative result: for any linear secret-sharing scheme there exists an undetectable subverted version of it that efficiently allows surveillance.

On the other hand, users aim for a strong form of resilience against surveillance, that allows detectability even if they only have black-box access to the subverted sharing algorithm. In this strong resilience model, the subverted algorithm can also be given access to the private key $\mathtt{SK}$ and it is detectable by users even if the detection algorithm is given nothing (except the inputs and outputs of the black-box). Symmetric ASAs suffice, as $(\mathtt{PK}, \mathtt{SK})$ can be seen a single secret key $\mathtt{K}$ embedded into the code; however, we maintain the asymmetric notation for continuity. We now disallow the detection algorithm to choose the secret to be shared and give it access to a SECRET oracle, reflecting that users should detect surveillance for sampled inputs. We formalize this framework in Section 4, where we also give the first construction of a linear secret-sharing scheme that is resilient against any efficient subversion. To obtain this positive result, we require that all the users give input to the sharing algorithm.

In contrast to [1], we consider strong forms of subversion and resilience to model the goals of big brother, respectively users and give the detection and subverted algorithms distinct capabilities. Similar to [1] (where big brother is not allowed to select the encryption key), we do not allow big brother to select the secret. However, we discuss in Section 4 the settings that allow surveillance resilience when big brother is allowed to select the secret and show that our proposal remains secure under this settings.

2 Preliminaries

Let $\mathbb{F}$ be a finite field and $\boldsymbol{v} \in \mathbb{F}^n$ a vector of n components; we denote by $\boldsymbol{v}[i]$ its i-th component. We denote sampling uniformly at random a value x from a set X as $x \twoheadleftarrow X$ and assigning a value Y to a variable y as $y \leftarrow Y$.

2.1 Secret Sharing

Let n be the set of parties (e.g. the different storage nodes) $\mathcal{P} = \{P_1, \ldots, P_n\}$. A *secret sharing scheme* consists of two algorithms $\Pi = (\mathsf{Sh}, \mathsf{Rec})$ such that:

- the *sharing algorithm* Sh is a randomized algorithm that receives as input a secret $\boldsymbol{s}$ and outputs a vector of shares $\boldsymbol{S} = (\boldsymbol{S}[1], \ldots, \boldsymbol{S}[n])$; We call *dealer* the entity that runs the algorithm on input $\boldsymbol{s}$ and that receives the output $\boldsymbol{S}$. We assume that the sharing algorithm is connected by a bidirectional secure channel[3] with each players P_i, in such a way that the share $\boldsymbol{S}[i]$ is securely sent to the player P_i.
  For any subset of players $A \subset \{P_1, \ldots, P_n\}$, let $\boldsymbol{S}_A$ be the vector of shares held by players in A, i.e. $\boldsymbol{S}_A = (\boldsymbol{S}[i])_{P_i \in A}$. A set $A \subset \{P_1, \ldots, P_n\}$ is called *unqualified* if the distribution of $\boldsymbol{S}_A$ is independent from $\boldsymbol{s}$, while it is called *qualified* if the secret $\boldsymbol{s}$ is uniquely determined from $\boldsymbol{S}_A$.
- the *reconstruction algorithm* Rec is a deterministic algorithm that receives as input a subset of shares $\boldsymbol{S}_A$ and outputs the value $\boldsymbol{s}$ if the set of shares corresponds to a qualified set of players; otherwise it outputs the special symbol $\perp$. We ask that the entire set of players $\{P_1, \ldots, P_n\}$ is always qualified.

The access structure of Π, Γ, is defined as the set of all $A \subset \{P_1, \ldots, P_n\}$ that are qualified and Γ_{min} is the set of the minimal qualified subsets, i.e. $\Gamma_{min} = \{B \in \Gamma \mid \nexists B' \subset B, B' \in \Gamma\}$. Let γ be the cardinality of the largest set in Γ_{min}, i.e. $\gamma = \mathsf{max}\{|B| \mid B \in \Gamma_{min}\}$ and let ρ the reconstruction threshold, i.e. the smallest integer such that every $A \subset \{P_1, \ldots, P_n\}$ of cardinality ρ is qualified.

Remark 1. In general, γ differs from the reconstruction threshold ρ. For example, let $n = 4$ and $\Gamma_{min} = \{\{P_2, P_3\}, \{P_2, P_4\}, \{P_3, P_4\}\}$. Then $\gamma = 2$, but $\rho = 3$. The inequality $\gamma \leq \rho$ always holds.

2.2 Linear Secret Sharing

Informally, a secret sharing scheme is called *linear* if the secret and the shares are elements of some vector spaces and the shares are computed as a linear function of the secret.

More precisely, given $\boldsymbol{M}$ a $n \times m$ matrix $(m > l)$ with elements in $\mathbb{F}$, the Linear Secret-Sharing Scheme (LSSS) associated to $\boldsymbol{M}$, $\Pi_{\boldsymbol{M}} = (\mathsf{Sh}_{\boldsymbol{M}}, \mathsf{Rec}_{\boldsymbol{M}})$,

[3] By secure channel we mean an authenticated and private channel that is also subversion resilient, that is big bother can not implement surveillance over it. Using the results of [1] and [11] for encryption scheme and digital signature such a channel can be easily implemented.

```
$\mathsf{Sh}_{\boldsymbol{M}}(\boldsymbol{s})$
    $\boldsymbol{r} \twoheadleftarrow \mathbb{F}^d$
    $\boldsymbol{f}^T \leftarrow (\boldsymbol{s}, \boldsymbol{r})^T$
    $\boldsymbol{S} \leftarrow \boldsymbol{M} \cdot \boldsymbol{f}$
    return $\boldsymbol{S}$

$\mathsf{Rec}_{\boldsymbol{M}}(\boldsymbol{S}_B)$
    if $B$ is qualified then
        $\boldsymbol{s} \leftarrow \boldsymbol{N}_B \cdot \boldsymbol{S}_B$
    else
        $\boldsymbol{s} \leftarrow \perp$
    return $\boldsymbol{s}$
```

Construction 1: LSSS $\mathit{\Pi}_{\boldsymbol{M}} = (\mathsf{Sh}_{\boldsymbol{M}}, \mathsf{Rec}_{\boldsymbol{M}})$

is defined in Construction 1. To share a secret $\boldsymbol{s} = (\boldsymbol{s}[1], \ldots, \boldsymbol{s}[l]) \in \mathbb{F}^l$, the algorithm first forms a column vector $\boldsymbol{f} \in \mathbb{F}^m$ where $\boldsymbol{s}$ appears in the first l entries and with the last d entries chosen uniformly at random and then computes $\boldsymbol{S} = \boldsymbol{M} \cdot \boldsymbol{f}$. We will use π_l to denote the projection that outputs the first l coordinates of a vector, *i.e.* $\pi_l(\boldsymbol{f}) = \boldsymbol{s}$. Similarly, let $\pi^d(\boldsymbol{f})$ be the last d elements of $\boldsymbol{f}$; hence, $\pi^d(\boldsymbol{f}) = \boldsymbol{r}$, where $d = m - l$.

Let $\boldsymbol{m}_i$ be the row i of $\boldsymbol{M}$ and $\boldsymbol{m}^i$ be the column i of $\boldsymbol{M}$. If $B \subseteq \mathcal{P}$, then $\boldsymbol{M}_B = (\boldsymbol{m}_i)_{P_i \in B}$ denotes the matrix built from all rows $\boldsymbol{m}_i$ such that $P_i \in B$.

It easy to see that a player subset B is qualified if and only if there exists a $l \times |B|$ matrix $\boldsymbol{N}_B$ such that for any $\boldsymbol{f} \in \mathbb{F}^d$, $\boldsymbol{N}_B \cdot (\boldsymbol{M}_B \cdot \boldsymbol{f}) = \pi_l(\boldsymbol{f})$.

Remark 2. The inequality $\gamma > l$ always holds from the correctness of reconstruction and the usage of randomness ($d > 0$).

For the rest of the paper, we fix $\boldsymbol{M}$ and denote $\mathit{\Pi}_{\boldsymbol{M}} = (\mathsf{Sh}_{\boldsymbol{M}}, \mathsf{Rec}_{\boldsymbol{M}})$ by $\mathit{\Pi} = (\mathsf{Sh}, \mathsf{Rec})$ to simplify notation.

Example 1 (Additive secret-sharing scheme). To share a secret $\boldsymbol{s} \in \mathbb{F}$ among n players, the sharing algorithm chooses random values $\boldsymbol{S}[1], \ldots, \boldsymbol{S}[n]$ in $\mathbb{F}$ such that $\sum_{i=1}^{n} \boldsymbol{S}[i] = \boldsymbol{s}$ and sends the value $\boldsymbol{S}[i]$ to P_i. It is clear that the set of all the players can reconstruct the secret from the received values, while any set of at most $n - 1$ players has no information on the value $\boldsymbol{s}$ held by the dealer. Notice that in this case $\gamma = n$.

Example 2 (Packed Shamir's scheme [12]). Let $\{\alpha_1, \ldots, \alpha_n\}$ and $\{e_1, \ldots, e_l\}$ be two disjoint sets of distinct random elements of $\mathbb{F}$. To share the secret $\boldsymbol{s} \in \mathbb{F}^l$, the sharing algorithm samples a polynomial $f(x) \in \mathbb{F}[x]$ of degree at most $\tau + l - 1$ such that $f(e_b) = \boldsymbol{s}[b]$ and sends to player P_i the evaluation $f(\alpha_i)$. Using Lagrange's interpolation it can be proved that any set of τ shares gives no information about the secret $\boldsymbol{s}$, while any set of $\tau + l$ shares can reconstruct it. In this scheme we have $\gamma = \tau + l$.

3 Subverting Secret-Sharing

This section models big brother's $\mathcal{B}$ goal: to subvert the sharing algorithm Sh to an algorithm $\widetilde{\mathsf{Sh}}$ that allows him to perform surveillance, while it remains undetected under the strong subversion scenario (see Section 1).

Surveillance means that $\mathcal{B}$ compromises privacy and learns the secret (or part of it) from corrupting an unqualified set of parties. To do so, $\mathcal{B}$ can embed in the code a key and a strategy. The embedded key is used to favor $\mathcal{B}$ over other entities, by leaking information in encrypted form. In real life, $\mathcal{B}$ aims to keep decryption capabilities to itself even in case of reverse engineering the algorithm, so our definitions consider asymmetric ASAs ($\mathcal{B}$ embeds a public key $\mathtt{PK}$ in the code and keeps the corresponding secret key $\mathtt{SK}$ private). The strategy $\mathcal{T}$ defines the unqualified set of parties $\mathcal{B}$ must corrupt to break the privacy of the scheme. We expect that $\mathcal{B}$ embeds in the code and hence follows a strategy $\mathcal{T}$ that maximizes its chances to win (e.g. minimum number of parties, if all parties are equally susceptible to corruption or easy to corrupt parties otherwise).

Undetectability means that no efficient detection algorithm $\mathcal{U}$ that is not given the decryption key $\mathtt{SK}$ can distinguish between the real and the subverted sharing algorithm. In the absence of the undetectability condition, subversion is always possible: $\widetilde{\mathsf{Sh}}$ simply distributes the secret (or parts of it) in shares in accordance to the strategy $\mathcal{T}$.

3.1 Definitions

Let $\Pi = (\mathsf{Sh}, \mathsf{Rec})$ be a secret-sharing scheme and let $\mathcal{K}$ be a probabilistic key generation algorithm that outputs a public-private key pair $(\mathtt{PK}, \mathtt{SK})$. A subversion of Π is a pair $\widetilde{\Pi} = (\widetilde{\mathsf{Sh}}, \widetilde{\mathsf{Rec}})$, with the following features: the subverted sharing algorithm $\widetilde{\mathsf{Sh}}$ is a randomized algorithm that maps $(\boldsymbol{s}, \mathtt{PK}, \mathtt{ID}, \mathcal{T})$ to a share vector $\mathbf{S}$. The input $\mathtt{ID}$ identifies the dealer that runs the sharing algorithm; this information is in general available in the system (e.g. the IP address or any authentication information of the client application for storage systems). $\mathcal{T}$ is a strategy that outputs the subset $T \subset \{P_1, \ldots, P_n\}$ used to leak information. The subverted reconstruction algorithm $\widetilde{\mathsf{Rec}}$ is an algorithm that maps $(\mathbf{S}_T, \mathtt{ID}, \mathtt{SK})$ to the shared secret $\boldsymbol{s}$, where $\mathbf{S}_T$ is the subset of shares that belongs to the unqualified set T.

We give next the definitions for detection and surveillance games. In contrast to the traditional unbounded adversarial power in secret sharing, our model is defined in the computational settings [13, 14]. In the following, we say that a function ϵ is *negligible* in N if for every polynomial function $p(N)$ there exists a constant c such that $\epsilon(N) < \frac{1}{p(N)}$ when $N > c$. With the notation $\mathcal{A}^{\mathtt{ALG}}(z)$ we mean that the entity $\mathcal{A}$ has oracle access to the algorithm $\mathtt{ALG}$ with knowledge of z.

DETECTION ADVANTAGE. Let $\Pi = (\mathsf{Sh}, \mathsf{Rec})$ be a secret sharing scheme and $\widetilde{\Pi} = (\widetilde{\mathsf{Sh}}, \widetilde{\mathsf{Rec}})$ a subversion of it. Let $\mathcal{U}$ be a detection algorithm that is not given $\mathtt{SK}$. The advantage of $\mathcal{U}$ to detect the ASA is defined as:

```
Game $\mathsf{DETECT}^{\mathcal{U}}_{\Pi,\widetilde{\Pi}}$
    $b \twoheadleftarrow \{0,1\}$
    $(\mathtt{PK}, \mathtt{SK}) \twoheadleftarrow \mathcal{K}$
    $b' \twoheadleftarrow \mathcal{U}^{\textsc{Share}}(\mathtt{PK}, \mathcal{T})$
    return $(b = b')$

$\textsc{Share}(\boldsymbol{s})$
    if $b$=1 then
        $\boldsymbol{S} \leftarrow \mathsf{Sh}(\boldsymbol{s})$
    else
        $\boldsymbol{S} \leftarrow \widetilde{\mathsf{Sh}}(\boldsymbol{s}, \mathtt{ID}, \mathtt{PK}, \mathcal{T})$
    return $\boldsymbol{S}$
```

Game 1: DETECT (Detection Game)

$$\mathsf{Adv}^{\mathsf{det}}_{\Pi,\widetilde{\Pi}}(\mathcal{U}) = 2\Pr[\mathsf{DETECT}^{\mathcal{U}}_{\Pi,\widetilde{\Pi}} \Rightarrow \mathtt{true}] - 1$$

A subversion $\widetilde{\Pi}$ is *undetectable* if $\mathsf{Adv}^{\mathsf{det}}_{\Pi,\widetilde{\Pi}}(\mathcal{U})$ is negligible for any efficient $\mathcal{U}$.

Detectability measures the ability of $\mathcal{U}$ to detect an ASA. In the DETECT game, $\mathcal{U}$ must detect if it receives shares produced by the real algorithm Sh or by its subversion $\widetilde{\mathsf{Sh}}$. To capture the case of reverse engineering, we allow $\mathcal{U}$ to use the encryption key PK and the strategy $\mathcal{T}$ that are embedded in the code; of course, the detection algorithm does not have access to the decryption key SK.

Clearly, $\mathcal{B}$ wants a subversion to be undetectable. By allowing $\mathcal{U}$ full control over the secret, the shares and the embedded PK, our definition captures the strongest form of detectability.

SURVEILLANCE ADVANTAGE. Let $\Pi = (\mathsf{Sh}, \mathsf{Rec})$ be a secret sharing scheme and $\widetilde{\Pi} = (\widetilde{\mathsf{Sh}}, \widetilde{\mathsf{Rec}})$ a subversion of it. Let $\mathcal{B}$ (big brother) be an adversary that knows SK. The advantage of $\mathcal{B}$ to detect the ASA is defined as:

$$\mathsf{Adv}^{\mathsf{srv}}_{\Pi,\widetilde{\Pi}}(\mathcal{B}) = 2Pr[\mathsf{SURV}^{\mathcal{B}}_{\Pi,\widetilde{\Pi}} \Rightarrow \mathtt{true}] - 1$$

A scheme Π is *secure against surveillance* if $\mathsf{Adv}^{\mathsf{srv}}_{\Pi,\widetilde{\Pi}}(\mathcal{B})$ is negligible for any efficient $\mathcal{B}$ and for any $\widetilde{\Pi}$.

Surveillance advantage measures the ability of a scheme to be secure against ASAs. Clearly, $\mathcal{B}$ wants to break privacy. Our definition models the stronger property that $\mathcal{B}$ cannot even distinguish between the real algorithm Sh and its subversion $\widetilde{\mathsf{Sh}}$; in particular, the subversion gives $\mathcal{B}$ no advantage to restore the secret by corrupting an unqualified set of parties. SURV game is similar to the DETECT game, except that the adversary $\mathcal{B}$ is given the secret key SK and cannot select the secret to be shared, but interrogates a Secret oracle to obtain it.

We can now model a *negative result*: a scheme Π is susceptible to ASAs if there exists an undetectable subversion $\widetilde{\Pi}$ of Π that allows an efficient adversary $\mathcal{B}$ to have a non-negligible surveillance advantage (e.g. to break privacy). We call $\widetilde{\Pi}$ a *successful subversion* of Π. We show that this is the case for any LSSS in Section 3.3.

```
Game $\mathsf{SURV}^{\mathcal{B}}_{\Pi,\widetilde{\Pi}}$
  $b \twoheadleftarrow \{0,1\}$
  $(\mathtt{PK},\mathtt{SK}) \twoheadleftarrow \mathcal{K}$
  $b' \twoheadleftarrow \mathcal{B}^{\textsc{Share}}(\mathtt{PK},\mathtt{SK},\mathcal{T})$
  return $(b = b')$

$\textsc{Secret}()$
  $\boldsymbol{s} \twoheadleftarrow \mathbb{F}^{l}$
  return $\boldsymbol{s}$

$\textsc{Share}()$
  $\boldsymbol{s} \leftarrow \textsc{Secret}()$
  if $b$=1 then
    $\boldsymbol{S} \leftarrow \mathsf{Sh}(\boldsymbol{s})$
  else
    $\boldsymbol{S} \leftarrow \widetilde{\mathsf{Sh}}(\boldsymbol{s},\mathtt{ID},\mathtt{PK},\mathcal{T})$
  return $\boldsymbol{s},\boldsymbol{S}$
```

Game 2: SURV (Surveillance Game)

3.2 Share-Fixing

Inspired by the existing work on bit-fixing [15, 16], we introduce share-fixing notions that we will later use to construct undetectable subversion of LSSS.

Let $\Pi = (\mathsf{Sh}, \mathsf{Rec})$ be a secret sharing scheme and $T \subset \{P_1, \ldots, P_n\}$. $\boldsymbol{S}_T$ is called a *share-fixing vector* for a secret $\boldsymbol{s}$ if there exists $\boldsymbol{S}$ a valid sharing of $\boldsymbol{s}$ such that $\boldsymbol{S}[i] = \boldsymbol{S}_T[i]$, for all $P_i \in T$. Intuitively, a share-fixing vector is a subset of ordered shares that can be expanded to a complete set of valid shares. A randomized algorithm $\mathcal{F}_\Pi$ that generates $\boldsymbol{S}_T$ for a given T and any secret $\boldsymbol{s}$ is called a *share-fixing source*. We will use $\mathcal{F}_\Pi(\boldsymbol{s}, T)$ to denote that $\mathcal{F}$ runs on input $(\boldsymbol{s}, T)$. Note that it is always possible to construct a share-fixing source by simply running $\mathsf{Sh}(\boldsymbol{s})$ and restrict its output to T.

For a share-fixing source $\mathcal{F}_\Pi$ and any secret $\boldsymbol{s}$, a randomized algorithm $\widehat{\mathsf{Sh}}$ that maps $(\boldsymbol{s}, \mathcal{F}_\Pi(\boldsymbol{s}, T))$ to a valid set of shares $\boldsymbol{S}$ such that $\boldsymbol{S}[i] = \boldsymbol{S}_T[i]$, for all $P_i \in T$ is called a *share-fixing extractor*. Intuitively, a share-fixing extractor expands the output $\boldsymbol{S}_T$ of the share-fixing source to a complete set of valid shares $\boldsymbol{S}$. Note that it is always possible to construct a share-fixing extractor by simply running $\mathsf{Sh}(\boldsymbol{s})$ repeatedly until $\boldsymbol{S}$ expands $\boldsymbol{S}_T$ (obviously, the construction is inefficient).

EXTRACTOR DETECTION ADVANTAGE. Let $\Pi = (\mathsf{Sh}, \mathsf{Rec})$ be a secret sharing scheme and $T \subseteq \{P_1, \ldots, P_n\}$. Let $\mathcal{F}_\Pi$ be a share-fixing source for (Π, T) and $\widehat{\mathsf{Sh}}$ a share-fixing extractor for $(\Pi, \mathcal{F}_\Pi)$. Let $\widehat{\Pi} = (\widehat{\mathsf{Sh}}, \mathsf{Rec})$ be the secret sharing scheme obtained from Π by replacing the sharing algorithm Sh with the share-fixing extractor $\widehat{\mathsf{Sh}}$. The advantage of an algorithm $\mathcal{U}$ to detect the share-fixing extractor is defined as:

$$\mathsf{Adv}^{\mathsf{e\text{-}det}}_{\Pi,\widehat{\Pi}}(\mathcal{U}) = 2Pr[\mathsf{E\text{-}DETECT}^{\mathcal{U}}_{\Pi,\widehat{\Pi}} \Rightarrow \mathtt{true}] - 1$$

```
Game $\mathsf{E\text{-}DETECT}^{\mathcal{U}}_{\Pi,\widehat{\Pi}}$
    $b \twoheadleftarrow \{0,1\}$
    $b' \twoheadleftarrow \mathcal{U}^{\textsc{Share}}$
    return $b = b'$

$\textsc{Share}(\boldsymbol{s}, \mathcal{F}_{\Pi}, T)$
    if $b$=1 then
        $\boldsymbol{S} \leftarrow \mathsf{Sh}(\boldsymbol{s})$
    else
        $\boldsymbol{S}_T \twoheadleftarrow \mathcal{F}_{\Pi}(\boldsymbol{s}, T)$
        $\boldsymbol{S} \leftarrow \widehat{\mathsf{Sh}}(\boldsymbol{s}, \boldsymbol{S}_T)$
    return $\boldsymbol{S}$
```

Game 3: $\mathsf{E\text{-}DETECT}$ (Extraction Detection Game)

A share-fixing extractor $\widehat{\mathsf{Sh}}$ is *undetectable* if $\mathsf{Adv}^{\mathsf{e\text{-}det}}_{\Pi,\widehat{\Pi}}(\mathcal{U})$ is negligible for any efficient $\mathcal{U}$.

Extraction detectability measures the ability of $\mathcal{U}$ to distinguish a share-fixing extractor $\widehat{\mathsf{Sh}}$ from the real Sh. In the $\mathsf{E\text{-}DETECT}$ game, $\mathcal{U}$ must detect if it receives shares produced by the real algorithm Sh or by a share-fixing extractor $\widehat{\mathsf{Sh}}$, given a share-fixing source $\mathcal{F}_{\Pi}$. Clearly, undetectability is impossible if the share-fixing source $\mathcal{F}_{\Pi}$ samples $\boldsymbol{S}_T$ from a distribution which can be efficiently distinguished from the distribution of the shares produced by the original sharing algorithm. But that is not always the case: in the proof of Theorem 1 we show that for any LSSS it is always possible to find a nonempty set T such that the distribution of the shares held by players in T is easy to simulate (i.e. it is the uniform one).

Theorem 1. *Let $\Pi = (\mathsf{Sh}, \mathsf{Rec})$ be a LSSS. Then, there exists a nonempty unqualified set of players T of cardinality t such that if $\mathcal{F}_{\Pi}$ is an algorithm that maps $\boldsymbol{s} \in \mathbb{F}^l$ to a uniformly random $\boldsymbol{S}_T \in \mathbb{F}^t$, it holds that $\mathcal{F}_{\Pi}$ is a share-fixing source for (Π, T).*

Proof. Let $B \in \Gamma_{min}$ with $|B| = b$. By definition, we have that $rank(\boldsymbol{M}_B) = b$ and $rank(\pi^d(\boldsymbol{M}_B)) \geq b - l > 0$ with $\pi^d(\boldsymbol{M}_B)$ denoting the last d columns of $\boldsymbol{M}_B$. Let $t = rank(\pi^d(\boldsymbol{M}_B))$, then there exists $T \subset B$ of cardinality t such that $rank(\pi^d(\boldsymbol{M}_T)) = t$ (take as T a set of players that corresponds to nonempty proper subset of the indices of the rows that are linear independent in $\pi^d(\boldsymbol{M}_B)$). Notice that T is trivially unqualified. The proof reduces to the existence of $\boldsymbol{r}$ such that $\pi^d(\boldsymbol{f}) = \boldsymbol{r}$ and $\boldsymbol{M}_T \cdot \boldsymbol{f} = \boldsymbol{S}_T$, where both $\boldsymbol{S}_T$ and $\pi_l(\boldsymbol{f}) = \boldsymbol{s}$ are fixed. Let $\boldsymbol{M}_T = (\pi_l(\boldsymbol{M}_T) \mid \pi^d(\boldsymbol{M}_T))$. Under this notation, $\boldsymbol{M}_T \cdot \boldsymbol{f} = \boldsymbol{S}_T$ becomes $\pi_l(\boldsymbol{M}_T) \cdot \boldsymbol{s} + \pi^d(\boldsymbol{M}_T) \cdot \boldsymbol{r} = \boldsymbol{S}_T$ or equivalently $\pi^d(\boldsymbol{M}_T) \cdot \boldsymbol{r} = \boldsymbol{S}_T - \pi_l(\boldsymbol{M}_T) \cdot \boldsymbol{s}$, which always has a solution because the matrix $\pi^d(\boldsymbol{M}_T)$ has full row-rank by construction.

Then, it follows that for any LSSS there exists a share-fixing extractor. More precisely:

```
$\widehat{\mathsf{Sh}}(\boldsymbol{s}, \mathcal{F}_{\Pi}, T)$
  $\pi_l(\boldsymbol{f}) \leftarrow \boldsymbol{s}$
  $\boldsymbol{S}_T \leftarrow \mathcal{F}_{\Pi}(\boldsymbol{s}, T)$    ($T$ and $\mathcal{F}_{\Pi}$ as in Theorem 1)
  solve $\pi^d(\boldsymbol{M}_T) \cdot \boldsymbol{r} = \boldsymbol{S}_T - \pi_l(\boldsymbol{M}_T) \cdot \boldsymbol{s}$ for $\boldsymbol{r}$, where $\pi^d(\boldsymbol{M}_T)$ and $\pi_l(\boldsymbol{M}_T)$
  denote the last $d$ columns, respectively the first $l$ columns of $\boldsymbol{M}_T$
  (if $t < d$, fix $\boldsymbol{r}$ uniformly at random from the set of possible solutions)
  $\boldsymbol{f} \leftarrow (\boldsymbol{s}, \boldsymbol{r})^T$
  $\boldsymbol{S} \leftarrow \boldsymbol{M} \cdot \boldsymbol{f}$
  return $\boldsymbol{S}$
```

Construction 2: Share-fixing extractor $\widehat{\mathsf{Sh}}$ for $(\Pi, \mathcal{F}_{\Pi})$

Theorem 2. *Let $\Pi = (\mathsf{Sh}, \mathsf{Rec})$ be a LSSS and $\mathcal{F}_{\Pi}$ be a sharing-fixing source as defined in Theorem 1. Then, the algorithm $\widehat{\mathsf{Sh}}$ in Construction 2 is an undetectable share-fixing extractor $\widehat{\mathsf{Sh}}$ for $(\Pi, \mathcal{F}_{\Pi})$.*

Proof. Let $\widehat{\mathsf{Sh}}$ be defined as in Construction 2, where T is as in Theorem 1. $\widehat{\mathsf{Sh}}$ computes $\boldsymbol{r}$ as a solution of $\pi^d(\boldsymbol{M}_T) \cdot \boldsymbol{r} = \boldsymbol{S}_T - \pi_l(\boldsymbol{M}_T) \cdot \boldsymbol{s}$ (see Theorem 1). From the hypothesis, $\mathcal{F}_{\Pi}$ outputs $\boldsymbol{S}_T$ uniformly at random and hence $\boldsymbol{S}_T - \pi_l(\boldsymbol{M}_T) \cdot \boldsymbol{s}$ is uniformly at random. Since $\pi^d(\boldsymbol{M}_T)$ has full rank t, $\boldsymbol{r}$ is uniformly random in $\mathbb{F}^d$. Note that from the definition of LSSS, Sh also chooses $\boldsymbol{r}$ uniformly at random in $\mathbb{F}^d$. Once $\boldsymbol{r}$ is fixed, $\widehat{\mathsf{Sh}}$ follows Sh exactly: forms the column vector $\boldsymbol{f}$ and computes $\boldsymbol{S} = \boldsymbol{M} \cdot \boldsymbol{f}$. To conclude, the output distribution of $\widehat{\mathsf{Sh}}$ equals the output distribution of Sh and the share-fixing extractor $\widehat{\mathsf{Sh}}$ is undetectable with $\mathsf{Adv}^{\mathsf{e\text{-}det}}_{\Pi,\widehat{\Pi}}(\mathcal{U}) = 0$.

Example 3 (Additive secret-sharing scheme). $\mathcal{F}_{\Pi}$ from Theorem 1 can fix up to $n-1$ shares $\boldsymbol{S}[i_j] = \boldsymbol{S}_T[i_j]$, $j = 1 \ldots, n-1$. The share fixing extractor $\widehat{Sh}$ computes $\boldsymbol{S}[i_n] = \boldsymbol{s} - \sum_{j=1}^{n-1} \boldsymbol{S}[i_j]$.

Example 4 (Packed Shamir's scheme). $\mathcal{F}_{\Pi}$ from Theorem 1 can fix up to τ shares $f(\alpha_j) = \boldsymbol{S}_T[i_j]$. The share fixing extractor $\widehat{Sh}$ interpolates f of degree at most $\tau + l - 1$ such that $f(e_b) = \boldsymbol{s}[b]$, $b = 1, \ldots, l$ and $f(\alpha_j) = \boldsymbol{S}_T[i_j]$, $j = 1, \ldots, \tau$.

3.3 Shares Replacement Attack

We show that for any LSSS there exists an undetectable subverted version that efficiently allows surveillance. Let $\Pi = (\mathsf{Sh}, \mathsf{Rec})$ be a LSSS. Then, we construct a successful subversion $\widetilde{\Pi} = (\widetilde{\mathsf{Sh}}, \widetilde{\mathsf{Rec}})$ of Π such that an efficient adversary $\mathcal{B}$ learns the secret $\boldsymbol{s}$ or parts of it with probability 1.

Let $T = \{P_{i_1}, \ldots, P_{i_t}\}$, as defined in Theorem 1. The subverted sharing algorithm $\widetilde{\mathsf{Sh}}$ implements a share fixing source $\mathcal{F}_{\Pi}$ to generate a subset of shares

```
$\widetilde{\mathsf{Sh}}(\boldsymbol{s}, \mathtt{ID}, \mathtt{PK}, \mathcal{T})$
    $T \leftarrow \mathcal{T}$
    $\boldsymbol{S}_T \twoheadleftarrow \mathcal{F}_{\Pi}(\boldsymbol{s}, T)$
    $\boldsymbol{S} \leftarrow \widehat{\mathsf{Sh}}(\boldsymbol{s}, \boldsymbol{S}_T)$
    return $\boldsymbol{S}$

$\mathcal{F}_{\Pi}(\boldsymbol{s}, T)$
    $x \twoheadleftarrow \mathbb{F}$
    $\boldsymbol{S}_T[i_1] \leftarrow \mathcal{E}(\mathtt{PK}, x)$
    $\boldsymbol{S}' \leftarrow \mathsf{PRG}(x)$
    for $j = 2 \dots t$ do
        $\boldsymbol{S}_T[i_j] \leftarrow \boldsymbol{s}[j-1] + \boldsymbol{S}'[j-1]$
    return $\boldsymbol{S}_T$

$\widetilde{\mathsf{Rec}}(\boldsymbol{S}_T, \mathtt{ID}, \mathtt{SK})$
    $x \leftarrow \mathcal{D}(\mathtt{SK}, \boldsymbol{S}[i_1])$
    $\boldsymbol{S}' \leftarrow \mathsf{PRG}(x)$
    for $j = 2 \dots t$ do
        $\boldsymbol{s}[j-1] \leftarrow \boldsymbol{S}_T[i_j] - \boldsymbol{S}'[j-1]$
    return $(\boldsymbol{s}[1], \dots, \boldsymbol{s}[t-1])$
```

Construction 3: Subverted scheme $\widetilde{\Pi} = (\widetilde{\mathsf{Sh}}, \widetilde{\mathsf{Rec}})$ $(t \geq 2)$

$\boldsymbol{S}_T$ that allows $\mathcal{B}$ to compute the secret $\boldsymbol{s}$ (or a part of it), then expands $\boldsymbol{S}_T$ to a full set of shares $\boldsymbol{S}$ using the share-fixing extractor $\widehat{\mathsf{Sh}}$ from Theorem 2. To hide information about $\boldsymbol{s}$ into $\boldsymbol{S}_T$, $\widetilde{\mathsf{Sh}}$ uses a deterministic public key encryption scheme $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ such that if m is sampled uniformly at random from $\mathbb{F}$ then $\mathcal{E}(m)$ is uniformly distributed in $\mathbb{F}$ and a pseudo-random generator PRG that maps a seed in $\mathbb{F}$ to an element in $\mathbb{F}^t$. It is natural to assume such constructions exist [17–22][4].

If $t \geq 2$, a random seed x is encrypted under the public key $\mathtt{PK}$ of $\mathcal{B}$ to obtain $\boldsymbol{S}_T[i_1]$, the first share in $\boldsymbol{S}_T$. Then, $\widetilde{\mathsf{Sh}}$ simply hides in the remaining components of $\boldsymbol{S}_T$ some of the components of $\boldsymbol{s}$ by adding them (using the addition operation from $\mathbb{F}$) to the pseudo-random values given by the output of the pseudo-random generator.

The subverted scheme is correct. Since $\boldsymbol{S}$ is a valid vector of shares, reconstruction and privacy hold by construction.

Theorem 3. *Let $\Pi = (\mathsf{Sh}, \mathsf{Rec})$ be a LSSS with $\gamma - l \geq 2$ (this assures $t \geq 2$). Then, its subversion $\widetilde{\Pi} = (\widetilde{\mathsf{Sh}}, \mathsf{Rec})$ defined in Construction 3 is successful and $\mathcal{B}$ learns the first $t-1$ components of $\boldsymbol{s}$ with probability* 1.

Proof. In the subversion game, $\mathcal{B}$ extracts $\boldsymbol{S}_T$ from $\boldsymbol{S}$ accordingly to the embedded strategy $\mathcal{T}$ and then runs $\widetilde{\mathsf{Rec}}(\boldsymbol{S}'_T, \mathtt{ID}, \mathtt{SK})$ to get $(\boldsymbol{s}'[1], \dots, \boldsymbol{s}'[t-1])$. If $\boldsymbol{s}'[i] = \boldsymbol{s}[i]$ for all $i = 1, \dots, t-1$, then $\mathcal{B}$ outputs 0, otherwise $\mathcal{B}$ outputs 1. The surveillance advantage $\mathsf{Adv}^{\mathsf{srv}}_{\Pi, \widetilde{\Pi}}(\mathcal{B}) = 2|1 - 1/|\mathbb{F}|^t| - 1$ is clearly non-negligible.

In the detection game, $\boldsymbol{S}_T$ is indistinguishable from random in $\mathbb{F}^t$ by exploiting encryption and PRG security. Thus, by Theorem 2, $\mathcal{F}_{\Pi}$ is a share-fixing

[4] For [17] see Section X. *Avoiding Reblocking when Encrypting a Signed Message*

```
$\widetilde{\mathsf{Sh}}(\boldsymbol{s}, \mathtt{ID}, \mathtt{PK}, \mathcal{T})$
    $T \leftarrow \mathcal{T}$
    $\boldsymbol{S}_T \twoheadleftarrow \mathcal{F}_{\Pi}(\boldsymbol{s}, T)$
    $\boldsymbol{S} \leftarrow \widehat{\mathsf{Sh}}(\boldsymbol{s}, \boldsymbol{S}_T)$
    return $\boldsymbol{S}$

$\mathcal{F}_{\Pi}(\boldsymbol{s}, T)$
    $x \twoheadleftarrow \mathbb{F}$ such that $lsb(x) = lsb(\boldsymbol{s}[1])$
    $\boldsymbol{S}_T[i_1] \leftarrow \mathcal{E}(\mathtt{PK}, x)$
    return $\boldsymbol{S}_T$

$\widetilde{\mathsf{Rec}}(\boldsymbol{S}_T, \mathtt{ID}, \mathtt{SK})$
    $x \leftarrow \mathcal{D}(\mathtt{SK}, \boldsymbol{S}[i_1])$
    $lsb(\boldsymbol{s}[1]) = lsb(x))$
    return $lsb(\boldsymbol{s}[1])$
```

Construction 4: Subverted scheme $\widetilde{\Pi} = (\widetilde{\mathsf{Sh}}, \widetilde{\mathsf{Rec}})$ $(t = 1)$

source and $\widehat{Sh}$ is undetectable with $\mathsf{Adv}^{\text{e-det}}_{\Pi,\widehat{\Pi}}(\mathcal{U}) = 0$. Then, the detection advantage is $\mathsf{Adv}^{\mathsf{det}}_{\Pi,\widetilde{\Pi}}(\mathcal{U}) \leq \mathsf{Adv}_{\mathcal{E}}(\mathcal{U}) + \mathsf{Adv}_{\mathsf{PRG}}(\mathcal{U})$, which is negligible because of the security of the PRG and the assumption on the encryption scheme. We can therefore conclude that $\widetilde{\Pi}$ is a successful subversion.

The condition $\gamma - l \geq 2$ is satisfied by many commonly used sharing schemes. For example, it is satisfied by the additive scheme with more than 2 players (Example 1 with $n > 2$) and by Shamir's scheme with at least 2 privacy (Example 2 with $\tau \geq 2$).

We give in Construction 4 an undetectable subversion for $t = 1$, which reveals one bit of the secret. Naturally, the construction works for any $t \geq 1$, but it gives big brother significantly less information about $\boldsymbol{s}$ (which might be less desirable in real life). Constructions for $t = 1$ and $t \geq 2$ can easily be combined into a single one, but we keep them separated for clearness of exposure.

Let $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a deterministic public key encryption scheme as defined before, which will be used to securely encrypt the *lsb* (least significant bit) of $\boldsymbol{s}[1]$.[5]

Theorem 4. *Let $\Pi = (\mathsf{Sh}, \mathsf{Rec})$ be a LSSS. Then, its subversion $\widetilde{\Pi} = (\widetilde{\mathsf{Sh}}, \mathsf{Rec})$ defined in Construction 4 is successful and $\mathcal{B}$ learns the least significant bit of $\boldsymbol{s}[1]$ with probability* 1.

Proof. In the subversion game, $\mathcal{B}$ extracts $\boldsymbol{S}_T$ from $\boldsymbol{S}$ accordingly to the embedded strategy $\mathcal{T}$ and then runs $\widetilde{\mathsf{Rec}}(\boldsymbol{S}'_T, \mathtt{ID}, \mathtt{SK})$ to get a bit b'. If $b' = lsb(\boldsymbol{s})$, then $\mathcal{B}$ outputs 0, otherwise $\mathcal{B}$ outputs 1. $\mathcal{B}$ wins with probability 1 when $b' \neq lsb(\boldsymbol{s})$ and with probability 1/2 when $b' = lsb(\boldsymbol{s})$. Hence, the surveillance advantage $\mathsf{Adv}^{\mathsf{srv}}_{\Pi,\widetilde{\Pi}}(\mathcal{B}) = 2|1/2 \cdot 1 + 1/2 \cdot 1/2| - 1 = 1/2$ is clearly non-negligible.

[5] Again, such encryption systems exists, for example padded RSA where encryption is repeated until the ciphertext lies in $\mathbb{F}$.

In the detection game, $\boldsymbol{S}_T$ is indistinguishable from random in $\mathbb{F}^t$ by exploiting encryption security. Thus, by Theorem 2, $\mathcal{F}_{\Pi}$ is a share-fixing source and $\widehat{\mathsf{Sh}}$ is undetectable with $\mathsf{Adv}^{\text{e-det}}_{\Pi,\widehat{\Pi}}(\mathcal{U}) = 0$. Then, the detection advantage is $\mathsf{Adv}^{\text{det}}_{\Pi,\widetilde{\Pi}}(\mathcal{U}) \leq \mathsf{Adv}_{\mathcal{E}}(\mathcal{U})$, hence negligible. We can therefore conclude that $\widetilde{\Pi}$ is a successful subversion.

4 Subversion Resilient Secret Sharing

4.1 Multi-Input Secret Sharing

We aim to define (linear) secret-sharing schemes that stands against ASAs. To achieve this, we allow the parties to give input to the sharing algorithm: each player in $\mathcal{P}$ inputs a random element $\boldsymbol{u}[i]$ to Sh, while the dealer inputs, as always, the secret $\boldsymbol{s}$.

Let $\Pi = (\mathsf{Sh}, \mathsf{Rec})$ be a multi-input secret sharing scheme that consists of two algorithms such that:

- the *sharing algorithm* Sh receives as input from the dealer a secret $\mathbf{s}$ and as input from $\mathcal{P}$ a vector $\mathbf{u} = (\boldsymbol{u}[1], \ldots, \boldsymbol{u}[n])$, where $\boldsymbol{u}[i]$ is given by P_i and outputs a set of shares $\mathbf{S} = (\boldsymbol{S}[1], \ldots, \boldsymbol{S}[n])$; note that since we assume the existence of authenticated, private and subversion resilient channels between the sharing algorithm and the players, $\boldsymbol{u}[i]$ remains unknown to all parties, except P_i;
- the reconstruction algorithm Rec remains unchanged; it receives as input a set of shares $\mathbf{S}$ and outputs the secret $\mathbf{s}$ if the set of shares corresponds to a qualified set.

4.2 Definitions

Similar to Section 3, we introduce the definitions for detection and surveillance advantages. Notice that this section models the users' goal, so what we want is strong resilience: $\mathcal{B}$ can embed in the code the secret key $\mathtt{SK}$, while $\mathcal{U}$ is not given access to the strategy and the public key. Even more, we disallow $\mathcal{U}$ to select the secret or the inputs of the players and give it access to a SECRET oracle, reflecting that $\mathcal{U}$ should detect surveillance for any input. To differentiate the games from the ones in Section 3 defined for strong subversion, we prefix them by R (which stands for *resilience*).

DETECTION ADVANTAGE. Let $\Pi = (\mathsf{Sh}, \mathsf{Rec})$ be a (multi-input) secret sharing scheme and $\widetilde{\Pi} = (\widetilde{\mathsf{Sh}}, \widetilde{\mathsf{Rec}})$ a subversion of it. Let $\mathcal{U}$ be a detection algorithm that is not given $\mathtt{PK}$ and $\mathcal{T}$. The advantage of $\mathcal{U}$ to detect an ASA is defined as:

$$\mathsf{Adv}^{\text{r-det}}_{\Pi,\widetilde{\Pi}}(\mathcal{U}) = 2\Pr[\text{R-DETECT}^{\mathcal{U}}_{\Pi,\widetilde{\Pi}} \Rightarrow \mathtt{true}] - 1$$

A subversion $\widetilde{\Pi}$ is *undetectable* if $\mathsf{Adv}^{\text{r}-\text{det}}_{\Pi,\widetilde{\Pi}}(\mathcal{U})$ is negligible for any efficient $\mathcal{U}$.

```
Game R-DETECT$^{\mathcal{U}}_{\Pi,\widetilde{\Pi}}$
  $b \twoheadleftarrow \{0,1\}$
  $(\mathtt{PK},\mathtt{SK}) \twoheadleftarrow \mathcal{K}$
  $b' \twoheadleftarrow \mathcal{U}^{\textsc{Share}}$
  return $(b = b')$

SECRET()
  $\boldsymbol{s} \twoheadleftarrow \mathbb{F}^l$
  $\boldsymbol{u} \twoheadleftarrow \mathbb{F}^n$
  return $\boldsymbol{s}, \boldsymbol{u}$

SHARE()
  $\boldsymbol{s}, \boldsymbol{u} \leftarrow$ SECRET()
  if $b$=1 then
    $\boldsymbol{S} \leftarrow \mathsf{Sh}(\boldsymbol{s}, \mathbf{u})$
  else
      $\boldsymbol{S} \leftarrow \widetilde{\mathsf{Sh}}(\boldsymbol{s}, \mathbf{u}, \mathtt{ID}, \mathtt{PK}, \mathtt{SK}, \mathcal{T})$
  return $\boldsymbol{s}, \boldsymbol{u}, \boldsymbol{S}$
```

Game 4: R-DETECT (Detection Game)

Clearly, honest players want all subversions to be easily detectable (even when they cannot perform reverse engineering). By restricting $\mathcal{U}$ from accessing anything except the interface of the sharing algorithm and allowing $\mathcal{B}$ to embed in the code the secret key SK, our definition captures a strong notion of detectability.

SURVEILLANCE ADVANTAGE. Let $\Pi = (\mathsf{Sh}, \mathsf{Rec})$ be a (multi-input) secret sharing scheme and $\widetilde{\Pi} = (\widetilde{\mathsf{Sh}}, \widetilde{\mathsf{Rec}})$ a subversion of it. Let $\mathcal{B}$ (big brother) be an adversary that knows SK. The advantage of $\mathcal{B}$ to detect an ASA is defined as:

$$\mathsf{Adv}^{\mathsf{r\text{-}srv}}_{\Pi,\widetilde{\Pi}}(\mathcal{B}) = 2\Pr[\mathsf{R\text{-}SURV}^{\mathcal{B}}_{\Pi,\widetilde{\Pi}} \Rightarrow \mathtt{true}] - 1$$

A scheme Π is *secure against surveillance* if $\mathsf{Adv}^{\mathsf{r-srv}}_{\Pi,\widetilde{\Pi}}(\mathcal{B})$ is negligible for any efficient $\mathcal{B}$ and for any $\widetilde{\Pi}$.

SURV game is similar to the DETECT game, except that the adversary $\mathcal{B}$ is given the keys PK, SK and the strategy $\mathcal{T}$.

We can now model a *positive result*: a scheme Π is resilient to ASAs if all possible subversions $\widetilde{\Pi}$ of Π are detectable. We call Π *subversion resilient*. We give a secure construction in this sense in Section 4.3.

4.3 Subversion Resilient Multi-Input LSSS

Let $\Pi = (\mathsf{Sh}, \mathsf{Rec})$ be a LSSS. We construct $\Pi^* = (\mathsf{Sh}^*, \mathsf{Rec}^*)$ multi-input LSSS that cannot be subverted without violating detectability. Let PRG be a pseudorandom generator that maps a seed in $\mathbb{F}$ to an element in $\mathbb{F}^d$.

Theorem 5. *The multi-input LSSS $\Pi^* = (\mathsf{Sh}^*, \mathsf{Rec}^*)$ defined in Construction 5 is subversion resilient.*

```
Game $\mathsf{R\text{-}SURV}^{\mathcal{B}}_{\Pi,\widetilde{\Pi}}$
    $b \twoheadleftarrow \{0,1\}$
    $(\mathtt{PK},\mathtt{SK}) \twoheadleftarrow \mathcal{K}$
    $b' \twoheadleftarrow \mathcal{B}^{\textsc{Share}}(\mathtt{PK},\mathtt{SK},\mathcal{T})$
    return $(b = b')$

$\textsc{Secret}()$
    $\boldsymbol{s} \twoheadleftarrow \mathbb{F}^l$
    $\boldsymbol{u} \twoheadleftarrow \mathbb{F}^n$
    return $\boldsymbol{s},\boldsymbol{u}$

$\textsc{Share}()$
    $\boldsymbol{s},\boldsymbol{u} \leftarrow \textsc{Secret}()$
    if $b$=1 then
        $\boldsymbol{S} \leftarrow \mathsf{Sh}(\boldsymbol{s},\mathbf{u})$
    else
          $\boldsymbol{S} \leftarrow \widetilde{\mathsf{Sh}}(\boldsymbol{s},\mathbf{u},\mathtt{ID},\mathtt{PK},\mathtt{SK},\mathcal{T})$
    return $\boldsymbol{S}$
```

Game 5: R-SURV (Surveillance Game)

```
$\mathsf{Sh}(\boldsymbol{s},\boldsymbol{u})$
    $\boldsymbol{r} \leftarrow \mathsf{PRG}(\boldsymbol{u}[1] \oplus \cdots \oplus \boldsymbol{u}[n])$
    $\mathbf{f}^T \leftarrow (\boldsymbol{s},\boldsymbol{r})^T$
    $\mathbf{S} \leftarrow \boldsymbol{M} \cdot \boldsymbol{f}$
    return $\boldsymbol{S}$

$\mathsf{Rec}(\boldsymbol{S}_B)$
    if $B$ is qualified then
        $\boldsymbol{s} \leftarrow \boldsymbol{N}_B \cdot \boldsymbol{S}_B$
    else
        $\boldsymbol{s} \leftarrow \perp$
    return $\boldsymbol{s}$
```

Construction 5: Subversion Resilient Multi-Input LSSS $\Pi^* = (\mathsf{Sh}^*, \mathsf{Rec}^*)$

Proof. First, we note that the shares by Sh^* are a deterministic function of $\boldsymbol{u}$ and $\boldsymbol{s}$. The detection algorithm simply takes the values $\boldsymbol{u}[i]$ produced by each player and verifies that the shares sent are the ones that would be produced by Sh^*. Any subversion with advantage δ must produce a different set of shares with probability greater or equal to δ (if at least one player is honest, $\boldsymbol{u}[1] \oplus \ldots \oplus \boldsymbol{u}[n]$ is uniformly random and hence $\boldsymbol{r}$ is uniformly random from the security of PRG). We can therefore conclude that $\mathsf{Adv}^{\mathsf{r\text{-}det}}_{\Pi^*,\widetilde{\Pi^*}}(\mathcal{U}) \geq \delta$ for any possible subversion $\widetilde{\Pi^*}$.

Discussion. Our modeling does not allow big brother to select the secret. Otherwise, if detection and surveillance games run independently, it is trivial for big brother to generate an undetectable subversion. Namely, it subverts the algorithm as follows: if the secret queried is a fixed element (e.g. an element deterministically computed from the key), then the subverted algorithm outputs specific shares, otherwise it generates proper shares. Note that this subversion is undetectable since the key is randomly sampled. This reflects the fact that in practice big brother can always embed hidden pattern which will allow surveillance when this pattern is matched by a secret. This could be used to notice unauthorized storage of sensitive documents by embedding a secret pattern within the documents and then subverting the algorithm to misbehave under this hidden pattern. The best that a user can therefore hope to do is to be able to detect whether or not the sharing could have allowed surveillance. Hence, we could allow big brother to input the secret in the surveillance game, but require that detection is continuously performed at runtime. In terms of games, this can be easily modeled by giving the subverted algorithm permission to select the secret, while detection algorithm runs on all this secrets and the corresponding outputs. It is immediate that our construction remains secure under this settings, since any subversion would require different shares than the ones that would have been produced by Sh with very high probability.

Acknowledgements. Samuel Ranellucci and Irene Giacomelli acknowledge support from the Danish National Research Foundation and The National Science Foundation of China (under the grant 61361136003) for the Sino-Danish Center for the Theory of Interactive Computation and from the Center for Research in Foundations of Electronic Markets (CFEM), supported by the Danish Strategic Research Council within which part of this work was performed. Partially supported by Danish Council for Independent Research via DFF Starting Grant 10-081612. Partially supported by the European Research Commission Starting Grant 279447.

Ruxandra F. Olimid was supported by the strategic grant POSDRU/159/1.5/S/137750, "Project Doctoral and Postdoctoral programs support for increased competitiveness in Exact Sciences research" cofinanced by the European Social Found within the Sectorial Operational Program Human Resources Development 2007-2013.

References

1. Bellare, M., Paterson, K.G., Rogaway, P.: Security of symmetric encryption against mass surveillance. In: Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I. (2014) 1–19
2. Subbiah, A., Blough, D.M.: An approach for fault tolerant and secure data storage in collaborative work environments. In: StorageSS. (2005) 84–93
3. Storer, M.W., Greenan, K.M., Miller, E.L., Voruganti, K.: Potshards - a secure, recoverable, long-term archival storage system. TOS **5**(2) (2009)
4. Wylie, J.J., Bigrigg, M.W., Strunk, J.D., Ganger, G.R., Kiliççöte, H., Khosla, P.K.: Survivable information storage systems. Computer **33**(8) (2000) 61–68
5. Cleversafe. `http://www.cleversafe.com/` Last accessed: September 2015.
6. Dyadic. `https://www.dyadicsec.com/` Last accessed: September 2015.
7. Young, A.L., Yung, M.: The dark side of "black-box" cryptography, or: Should we trust capstone? In: Advances in Cryptology - CRYPTO '96, 16th Annual

International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings. (1996) 89–103

8. Young, A.L., Yung, M.: Kleptography: Using cryptography against cryptography. In: Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11-15, 1997, Proceeding. (1997) 62–74
9. Lampson, B.W.: A note on the confinement problem. Commun. ACM **16**(10) (1973) 613–615
10. Simmons, G.J.: The prisoners' problem and the subliminal channel. In: Advances in Cryptology, Proceedings of CRYPTO '83, Santa Barbara, California, USA, August 21-24, 1983. (1983) 51–67
11. Ateniese, G., Magri, B., Venturi, D.: Subversion-resilient signature schemes. Cryptology ePrint Archive, Report 2015/517 (2015) `http://eprint.iacr.org/`. To apper in Proceedings of the 2015 ACM SIGSAC Conference on Computer and Communications Security.
12. Franklin, M.K., Yung, M.: Communication complexity of secure computation (extended abstract). In: STOC. (1992) 699–710
13. Beimel, A.: Secret-sharing schemes: A survey. In: Coding and Cryptology - Third International Workshop, IWCC 2011, Qingdao, China, May 30-June 3, 2011. Proceedings. (2011) 11–46
14. Rogaway, P., Bellare, M.: Robust computational secret sharing and a unified account of classical secret-sharing goals. In: Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, October 28-31, 2007. (2007) 172–184
15. Gabizon, A., Raz, R., Shaltiel, R.: Deterministic extractors for bit-fixing sources by obtaining an independent seed. SIAM J. Comput. **36**(4) (2006) 1072–1094
16. Kamp, J., Zuckerman, D.: Deterministic extractors for bit-fixing sources and exposure-resilient cryptography. SIAM J. Comput. **36**(5) (2007) 1231–1247
17. Rivest, R.L., Shamir, A., Adleman, L.M.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM **21**(2) (1978) 120–126
18. Naccache, D., Stern, J.: A new public-key cryptosystem. In: Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11-15, 1997, Proceeding. (1997) 27–36
19. Chevallier-Mames, B., Naccache, D., Stern, J.: Linear bandwidth Naccache-Stern encryption. In: Security and Cryptography for Networks, 6th International Conference, SCN 2008, Amalfi, Italy, September 10-12, 2008. Proceedings. (2008) 327–339
20. Bogdanov, A., Viola, E.: Pseudorandom bits for polynomials. SIAM J. Comput. **39**(6) (2010) 2464–2486
21. Viola, E.: The sum of D small-bias generators fools polynomials of degree D. Computational Complexity **18**(2) (2009) 209–217
22. Wang, L., Hu, Z.: New sequences of period p^n and p^{n+1} via projective linear groups. In: Information Security and Cryptology - 8th International Conference, Inscrypt 2012, Beijing, China, November 28-30, 2012, Revised Selected Papers. (2012) 311–330