48th Annual IEEE Symposium on Foundations of Computer Science

Intrusion-Resilient Secret Sharing

Stefan Dziembowski*
Department of Computer Science,
University of Rome *La Sapienza*

Krzysztof Pietrzak
CWI Amsterdam

Abstract

We introduce a new primitive called Intrusion-Resilient Secret Sharing (IRSS)*, whose security proof exploits the fact that there exist functions which can be efficiently computed interactively using low communication complexity in k, but not in $k-1$ rounds.*

IRSS is a means of sharing a secret message amongst a set of players which comes with a very strong security guarantee. The shares in an IRSS are made artificially large so that it is hard to retrieve them completely, and the reconstruction procedure is interactive requiring the players to exchange k short messages. The adversaries considered can attack the scheme in rounds, where in each round the adversary chooses some player to corrupt and some function, and retrieves the output of that function applied to the share of the corrupted player. This model captures for example computers connected to a network which can occasionally be infected by malicious software like viruses, which can compute any function on the infected machine, but cannot sent out a huge amount of data.

Using methods from the Bounded-Retrieval Model*, we construct an IRSS scheme which is secure against any computationally unbounded adversary as long as the total amount of information retrieved by the adversary is somewhat less than the length of the shares, and the adversary makes at most $k-1$ corruption rounds (as described above, where k rounds are necessary for reconstruction). We extend our basic scheme in several ways in order to allow the shares sent by the dealer to be short (the players then blow them up locally) and to handle even stronger adversaries who can learn some of the shares completely.*

As mentioned, there is an obvious connection between IRSS schemes and the fact that there exist functions with an exponential gap in their communication complexity for k and $k-1$ rounds. Our scheme implies such a separation which is in several aspects stronger than the previously known ones.

*Supported by the EU Marie-Curie Fellowship MEIF-CT-2006-024300-*Cryptosensors*.

1 Introduction

Cryptography can be seen as the art of using the intractability of a certain tasks in order to prove security properties of some schemes. Shannon proved one-time pad encryption to be secure based on an information-theoretic impossibility result. Practical symmetric encryption was later based on the hardness of inverting functions [18], and asymmetric encryption was proven possible based on the (conjectured) hardness of number theoretical problems [9]. Other intractability assumptions that were used include the Heisenberg uncertainty principle [25], or the impossibility of errorless measurement of some physical phenomena [20], to mention just a few. Therefore what is bad news in other areas (i.e. the hardness of some problems) can be good news for cryptographers. In this paper we exploit the intractability of a type of problems which for now have not been used in cryptography, namely the fact that there exist multiparty functions whose *communication complexity* has an exponential gap for k versus $k-1$ round protocols. We do so by introducing a new primitive that we call *Intrusion-Resilient Secret Sharing (IRSS)*, and which can be described as a "secret-sharing scheme that is secure in the *Bounded-Retrieval Model (BRM)*", which is a model that was recently introduced in [7, 13].

Bounded-Retrieval Model The main idea of BRM is to make cryptographic protocols resilient against attacks of malicious software like viruses by assuming that the secrets stored on the infected machines are too large to be retrieved completely. The motivation for this model is as follows: traditional cryptographic protocols are designed with the assumption that (at least some of) the machines on which they are executed are beyond the control of the adversary. In the BRM one makes a more pessimistic (and more realistic) assumption that the machines of the users may fall under the control of the adversary (since he may install malicious software on them). Of course as long as the adversary is actually controlling the machine there is not much hope for security. Therefore the first assumption that one makes in this model is that the adversary controls the machine only

0272-5428/07 $25.00 © 2007 IEEE
DOI 10.1109/FOCS.2007.63

occasionally. In other words, there exists periods when the machine is virus-free.

The key assumption is the second one. It comes from an observation that if the secrets stored on the infected machine $\mathcal{M}$ are short, then the adversary can easily retrieve them, create his own copy of $\mathcal{M}$, and run this copy even when he lost access to the original $\mathcal{M}$. To prevent this type of an attack one makes the secrets stored on users' machines so large (10 GB, say), that it is infeasible to download them entirely (the virus can download up to 5 GB, say). We will use the strongest variant [13] of the BRM where it is assumed that the virus can perform any computation on the victim's machine before deciding what to transfer (a weaker notion [8, 7] only allows the adversary to access some limited number of bits of the secrets stored on the machine).

It was shown in [13] how to construct intrusion-resilient protocols for session-key agreement and entity authentication in this model (the construction of [13] was later improved in [5]). Schemes for secret data storage in this model were considered in [14].[1]

2 Informal description of IRSS

Notation For a finite set $\mathcal{A}$ let $\mathcal{A}^*$ denote the set of all finite sequences whose entries belong to $\mathcal{A}$, i.e. $\mathcal{A}^* = \bigcup_{i=0}^{\infty} \mathcal{A}^i$. Let "$||$" denote concatenation of finite sequences. We will overload this symbol and, for sets $\alpha_0, \alpha_1 \subseteq \mathcal{A}^*$, write $\alpha_0||\alpha_1$ to denote the set $\{a_0||a_1 : a_0 \in \alpha_0 \text{ and } a_1 \in \alpha_1\}$. Sequence $(a_1, \ldots, a_n) \in \mathcal{A}^*$ is a *subsequence* of $B \in \mathcal{A}^*$ if there exists sequences $B_1, \ldots, B_{n+1}$ such that $B = B_1||(a_1)||\cdots||B_n||(a_n)||B_{n+1}$. For $\ell \in \mathbf{N}$ let $(a_1, \ldots, a_n)^\ell$ denote the sequence of length $n\ell$ consisting of $(a_1, \ldots, a_n)$ repeated ℓ times. For a sequence $A \in \{0,1\}^m$, we denote with $A[1, \ldots, n]$ (where $n \leq m$) the n-bit prefix of A.

2.1 The two-party case

We start with the two-party case, as it is simpler but already explains the main idea. Then, we describe the general case which we are going to work with in this paper. A formal definition is given in Sect. 6. Recall the definition of a standard 2-*out-of*-2 *secret sharing (SS)* scheme: we have two players *Alice* and *Bob*, and a dealer. The dealer holds a secret message $M \in \{0,1\}^n$ and wants to share it among Alice and Bob in such a way that (1) Alice and Bob together can reconstruct M by computing some efficient function $reconstruct$ on their respective shares T_A and T_B, and (2) each of the players separately has no information about M. A trivial example of such a scheme is the one in which the dealer chooses T_A randomly from $\{0,1\}^n$ and sets $T_B = M \oplus T_A$.

The motivation for our work comes from the following observation. Suppose the dealer uses any standard SS scheme (like e.g. the one described above) to share a short secret M, and the players store their secrets T_A and T_B on their machines. Then, an adversary that got temporary access to the machine of Alice and then to the machine of Bob, can reconstruct M (assuming that he downloaded T_A and T_B). Our idea is to make the task of such an adversary significantly harder, by using the methods of the Bounded-Retrieval Model, i.e. the shares T_A and T_B will be so large that it is infeasible for the adversary to retrieve them completely. Even though the shares are large, our scheme will have a quite efficient reconstruction phase, where Alice and Bob will exchange 2ℓ short messages (i.e. they make ℓ "loops", where a loop is a message sent from Alice to Bob followed by a message sent from Bob to Alice. Here ℓ is a parameter that can initially be chosen by the dealer). The messages will be rather short (linear in the length of the shared secret) and the computational cost for Alice and Bob will be reasonable, in particular both will only have to access a small fraction of their shares.

Roughly speaking, we construct a scheme as above with the security property that the only way to reconstruct the secret M is to actually make ℓ loops as in the reconstruct procedure. The adversary we consider is computationally unbounded and can corrupt Alice and Bob, but never both at the same time. So he can "hop" between Alice and Bob, and in each round retrieve some information about the share of the corrupted player. We put two restrictions on the power of the adversary: (1) a bound on the amount of information the adversary can retrieve, and (2) a bound on the number of "hops" the adversary can make.

More precisely, we assume that time is divided into rounds and in the ith round the adversary can issue a "corruption" request. As a result of such a request the adversary gets access to either T_A (in which case we say that he *corrupted Alice*) or to T_B (we say then that he *corrupted Bob*). The adversary cannot retrieve the share T of the corrupted player completely, but he can choose any function $h_i : \{0,1\}^t \to \{0,1\}^{s_i}$ and retrieve $H_i = h_i(T)$. We say that an adversary is *s-bounded* if the total length of the H_i's retrieved from T_A is at most s, and the same holds for the H_i's retrieved from T_B.

Our secret sharing scheme guarantees that unless an s-bounded (where s is huge) adversary "hops" at least ℓ times Alice to Bob and back, he does not learn any significant information about M. We call an adversary which never hops ℓ times back and forth an ℓ-admissible adversary.

[1] In [14] this model was called a *Limited Communication Model*.

2.2 The multi-party case

We now extend the idea from the previous section to the *multi-party* case. Let $\mathcal{P} = \{P_0, \ldots, P_{a-1}\}$ be the set of players. We define an a-out-of-a-secret sharing scheme, in which a dealer (holding a secret $M \in \{0,1\}^n$) gives to each P_i a share $T_i \in \{0,1\}^t$ (where t is a large number). The reconstruction of the secret M requires players to make ℓ "loops", where a loop is a sequence of a short messages, the first sent from player P_0 to P_1, the second message (which is computed as a function of the message received from P_0 and the share of P_1) from player P_1 to P_2, and so on until the ath message which is sent from player P_{a-1} back to P_0. The adversary can adaptively issue corruption requests: in round i the adversary chooses a player P_{c_i} and a function $h_i : \{0,1\}^t \to \{0,1\}^{s_i}$ in order to learn $H_i = h_i(T_{c_i})$. We now must extend the notion of "ℓ-admissible" and "s-bounded" adversaries to the case of more than two players.

Let $\sigma := (P_{c_1}, \ldots, P_{c_w})$ denote the sequence of players the adversary chooses to corrupt in consecutive rounds and let $h_1, \ldots, h_w$ be the retrieval functions. We say that the adversary is s-bounded, if for every player P_j, the total length of the H_i's where $P_{c_i} = P_j$ is at most s. We say that the adversary is ℓ-admissible, if he never chooses a corruption sequence σ such that $(P_0, \ldots, P_{a-1})^\ell || (P_0)$ is a subsequence of σ. Observe that any adversary which is not ℓ-admissible can easily learn the secret by evaluating the reconstruction procedure (thus retrieving only very few bits). An example of a possible corruption sequence produced by a 3-admissible (but not by a 2-admissible) adversary is depicted below (what the last row in the table means will be explained later):

$$\begin{array}{rccccccccccc} i = & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\ P_{c_i} = & P_1 & \underline{P_0} & P_2 & \underline{P_1} & \underline{P_2} & P_2 & \underline{P_0} & \underline{P_1} & P_0 & \underline{P_2} & \underline{P_0} \\ cpl_i = & 0 & 1 & 1 & 2 & 3 & 3 & 4 & 5 & 5 & 6 & 7 \end{array} \quad (1)$$

2.3 Extensions

Our "basic" construction of an IRSS scheme is given in Sect.7. We also propose some variations of the basic scheme which are more efficient or achieve security against even stronger adversaries.

Local Share Expansion The size of the shares $T_0, \ldots, T_{a-1}$ of an IRSS must be large as it directly implies the retrieval-bound of the adversaries we want to tolerate. Having to store such large shares (say, 10GB) on a local computer is, given the extremely low price of storage, not a big issue (see Sect. 2.4), but getting such large shares from the dealer to the players might be a problem. In Sect. 8 we show how to construct schemes in which the dealer sends to the players quite short messages $\tau_0, \ldots, \tau_{a-1}$. Then, each player P_i after receiving τ_i computes his share T_i using a long locally generated random string ρ_i. Of course for this to work we need to assume that τ_i can be reliably erased and that during the distribution phase (more precisely: until the τ_i's get erased) the adversary does not corrupt any of the players.

Complete Leaks Our basic IRSS scheme is based on a so called BSM-secure function (BSM for Bounded Storage Model), which is a special kind of extractor. We cannot prove this scheme secure against adversaries which can retrieve some of the shares completely, but in Sect.9 we observe that if the BSM-secure function has the special property of being a permutation, then the scheme can be proven secure even against such adversaries. We give a generic construction (a two round Feistel network) to construct a BSM-secure *permutation* from any BSM-secure *function* (in fact, this construction works for any kind of extractors).

Computational power of the adversary In general the schemes considered in this paper are information-theoretically secure. The only exception is Sect. 11, where we introduce a computationally-secure variant of IRSS. The advantage of the scheme constructed in Sect. 11 is that it is significantly more efficient (in terms of communication complexity) than the other schemes in the paper, especially if the shared secret M is large.

2.4 Practical aspects

In this section we describe possible scenarios in which our schemes may be used in practice. Although our initial motivation was presented in the context of PCs connected to the Internet, our schemes can also be used in other environments, where highly-sensitive data needs to be shared among a group of users. Clearly, feasibility of our assumptions depends on the relation between the costs of *storing* and *downloading* data. Let us note, that currently the price of storing 10GB is extremely low (a 500GB hard disc costs around 100€), while downloading several GBs in an unnoticeable way can be considered infeasible in many settings.

Distribution phase We imagine two ways in which the shares can be distributed. The first is to use the basic scheme and have the players obtain their large shares directly from the dealer on some physical device like a DVD disc. The other possibility is to use the "local-share-expansion" trick, where the players only abtain small amounts of data from the dealer. These short messages can for example be transmitted by trusted links over the Internet. The drawback of this solution is that we must assume that the players reliably erase the τ_i's (in particular, in the sharing phase all the players need to be trusted). One also must be careful about how the channels that transmit τ_i's are secured. It is not enough

to use any standard encryption method, since if the adversary eavesdrops the ciphertext and later corrupts P_i then he may simply retrieve the key and decrypt τ_i. Hence, we need to use a forward-secure encryption scheme (see e.g. [10]). Or, if we also want to be secure against corruptions that occurred before the τ_i's were sent, we can use the intrusion-resilient scheme of [13, 5].

Reconstruction phase Also the reconstruction can be done in two different ways. In the first one the players bring their shares to a trusted center that reconstructs the secret. The trusted center can be some machine with a freshly installed operating system, which was never connected to the Internet. This method has a high communication complexity (since the entire shares need to be sent).

The other method is to let the players reconstruct the secret interactively, say at a given time they all switch their machines on, connect them to the Internet, and run the reconstruction procedure. The players have to exchange $a\ell$ messages, but the communication complexity is low.

3 Related Work

Secret-sharing schemes were introduced in [23, 3]. We have already discussed other work done in the Bounded Retrieval Model in Sect. 1. In this section we discuss the relation to communication complexity and proactive security.

Proactive security An alternative way to protect against the attacks of viruses is to construct protocols that are *proactively secure* (for an overview of this area see [4]). These methods require that the secrets stored by the machines of the users are periodically refreshed and the adversary must not corrupt too many machines between two such refresh phases. We do not require such interactive refreshments, but instead a bound on the amount of data retrieved by the adversary.

Rounds in communication complexity The (two-party) communication complexity of a function $f(.,.)$ is the number of bits that two players must exchange in order to compute $f(X,Y)$, where X and Y are held by the players respectively [26]. Papadimitriou and Sipser [22] asked how the communication complexity is affected if the players are restricted to exchange at most k messages. They conjectured that there is an exponential gap between the k and $k-1$ round communication complexity for a problem called pointer-jumping. This conjecture has been proven by Duris, Galil and Schnitger [12], with a subsequent tight bound by Nisan and Wigderson [21].

Our Intrusion-Resilient Secret-Sharing scheme directly implies an exponential gap for k and $k-1$ round communication complexity, and this separation is stronger than the separations achieved for pointer-jumping or any other problem we are aware of, as (1) we get an exponential gap for k and $k-1$ round protocol even if we allow the communication complexity in the $k-1$ round case to be a *constant fraction of the size of the inputs* (in previous separations the communication always was $o(n)$, where n is the input size); and (2) not only can the value $f(X,Y)$ not be computed by the $k-1$ round protocol with some constant probability, but in fact $f(X,Y)$ will be *statistically close to uniform given the view after* $k-1$ *rounds* (assuming that the inputs X and Y are chosen uniformly at random).

4 Further Work and Open Problems

Simultaneous Corruptions In this paper we focus on adversaries which can corrupt only one player at the time: in round i, the adversary chooses a function h_i and a player $P_{c_i} \in \mathcal{P}$ to retrieve $H_i = h_i(T_{c_i})$. A natural and much stronger adversarial model is to allow the adversary to corrupt subsets of players: the adversary chooses h_i and $\mathcal{P}_i \subset \mathcal{P}$, and retrieves $H_i = h_i(T_{\mathcal{P}_i})$, where $T_{\mathcal{P}_i}$ denotes all the shares hold by players in $\mathcal{P}_i$.

We can generalize the definition of ℓ-admissible adversaries (as sketched in Section 2.2) to this stronger model: let us call an adversary (as just described) ℓ-*set* admissible, if he never chooses a corruption sequence $\mathcal{P}_1, \ldots, \mathcal{P}_w$, such that $(P_0, \ldots, P_{a-1})^\ell \| P_0$ is a subsequence of $\mathcal{P}_1^* || \ldots || \mathcal{P}_w^*$. We leave it as an open problem to prove our (or another) protocol secure against such adversaries.

When making an additional assumption on either the corruption sequence or on the communication complexity of the h_i's, we can prove our protocol secure against ℓ-set-admissible adversaries as we will sketch now:

- The version of our protocol which is secure against complete leaks, is also secure against ℓ-set-admissible adversaries if for any corrupted sets of players $\mathcal{P}_i, \mathcal{P}_j$ we either have $\mathcal{P}_i = \mathcal{P}_j$ or $\mathcal{P}_i \cap \mathcal{P}_j = \emptyset$. The reason is that for any set $\mathcal{P}_i$ of corrupted players, we can give to the adversary the shares of all but one (say the lexicographically first) player for free. The condition on the $\mathcal{P}_i$'s just described then guarantees that the corruption sequence on the remaining players does not make ℓ loops.

- A (normal) ℓ-admissible adversary $\mathcal{A}'$ can simulate a ℓ-set-admissible adversary $\mathcal{A}$: if $\mathcal{A}$ requests $h_i(\mathcal{P}_{c_i})$, $\mathcal{A}'$ corrupts the players in $\mathcal{P}_{c_i}$ one at a time (possibly several times) until it learns $h_i(\mathcal{P}_{c_i})$. $\mathcal{A}'$ can do so by retrieving a total of $s_i' := s_i + cc(h_i)$ bits, where s_i is the output length of h_i and $cc(h_i)$ is the communication complexity of h_i, i.e. an upper bound on the total length of the messages the players in $\mathcal{P}_i$ must exchange

in order to jointly compute $h_i(T_{\mathcal{P}_i})$ (recall that $T_{\mathcal{P}_i}$ denotes the shares hold by the players in $\mathcal{P}_i$). Thus an IRSS which is secure against s-bounded ℓ-admissible adversaries, is also secure against ℓ-set-admissible adversaries, as long as the total communication complexity plus the output length of the h_i's chosen by the adversary is at most s.

Very Short Messages The length of the messages exchanged during reconstruction is linear in the length of the shared secret. It would be nice to have a protocol where the length of (most of) the messages is independent of the length of the shared secret (i.e. only depends on some security parameter). In the full version of the paper we propose a scheme where the length of all but the last a messages (a being the number of players) is very small. Unfortunately, we do not yet have a security proof for that scheme.

General Access Structures In this paper we only consider a-out-of-a secret sharing schemes, i.e. all a players must cooperate in order to reconstruct the secret. It is straightforward to generalize the scheme to work for any access structure $\Pi = \{\mathcal{S}_1, \mathcal{S}_2, \ldots\}$, each $\mathcal{S}_i \subseteq \mathcal{P}$ (the meaning being that any set of players $\mathcal{S} \in \Pi$ can reconstruct the secret), at the cost of increasing one of the shares by the size of $|\Pi| \cdot n$, where n is the size of the message shared. Of course the scheme is then only secure against adversaries which are ℓ-admissible for every $\mathcal{S} \in \Pi$. It is an open question if we can do better than that for simple access structures like threshold secret sharing (where for some threshold b, each player sets $\mathcal{S}, |\mathcal{S}| \geq b$ is able to reconstruct).

5 Tools and Notation

We will use the concept of Markov chains and Shannon entropy whose description can be found e.g. in [6]. Moreover, additionally to the notation introduced at the beginning of Sect. 2 we need the following. Let random variables X_0, X_1 be distributed over some set $\mathcal{X}$, and let Y be a random variable distributed over $\mathcal{Y}$. Define the *statistical distance between* X_0 *and* X_1 as $\Delta(X_0\ ;\ X_1) = \frac{1}{2}\sum_{x\in\mathcal{X}} \mid P_{X_0}(x) - P_{X_1}(x)|$. Moreover let $\Delta(X_0\ ;\ X_1 \mid Y) := \Delta(X_0, Y\ ;\ X_1, Y)$ be the *statistical distance between* X_0 *and* X_1 *conditioned on* Y. If X_1 has uniform distribution over $\mathcal{X}$ and is independent from Y then define $d(X_0) := \Delta(X_0\ ;\ X_1)$ and $d(X_0 \mid Y) := \Delta(X_0\ ;\ X_1 \mid Y)$ as the (conditional) distance of X_0 from uniform.[2] It is a straightforward calculation that $\Delta(X_0\ ;\ X_1 \mid Y)$ is equal to the following expected value $\sum_{y\in\mathcal{Y}} \mathsf{P}(y = Y) \cdot \Delta(P_{X_0|y=Y}\ ;\ P_{X_1|y=Y})$, and similarly $d(X_0 \mid Y) = \sum_{y\in\mathcal{Y}} \mathsf{P}(y = Y) \cdot d(P_{X_0|y=Y})$. It is also straightforward to verify that the following *triangle inequality* holds for Δ: for any X_0,X_1, and X_2 we have $\Delta(X_0\ ;\ X_1) \leq \Delta(X_0\ ;\ X_1) + \Delta(X_1\ ;\ X_2)$, and the same holds when the *conditional* statistical distance is used. The proofs of the following lemmas about statistical distance appear in [16].

[2] We will overload the symbols Δ and d and sometimes apply them to the probability distributions instead of the random variables.

Lemma 1 *Let $K, \widetilde{K}, R, T$ be random variables such that K is uniformly random, and let ϕ be any function. Then $d(\phi(\widetilde{K}, R) \mid \widetilde{K}, T) \leq d(\phi(K, R) \mid K, T) + d(\widetilde{K} \mid T)$.*

Lemma 2 *Let T, E, F be random variables where $T \rightarrow E \rightarrow F$ is a Markov chain (i.e. $P_{F|ET} = P_{F|E}$), then $d(F \mid E, T) = d(F \mid E)$.*

Lemma 3 *Let A, B be random variables where $A \in \mathcal{A}$. Then $\mathsf{P}(B = A) \leq d(A \mid B) + 1/|\mathcal{A}|$.*

Lemma 4 *Let A, B be independent random variables and consider a sequence $V_1, \ldots, V_i$ of random variables, where for some function ϕ, $V_i = \phi(V_1, \ldots, V_{i-1}, C_i)$, where each C_i is either A or B, then $B \rightarrow (V_1, \ldots, V_i) \rightarrow A$ is a Markov chain.*

Bounded-Storage Model One of our tools is a method for secure *key-expansion* in the Bounded-Storage Model (BSM), a model introduced by Maurer [19]. Because of the lack of space we do not discuss this model in detail here (the reader may consult e.g. [1, 15, 24]). We say that a function $f : \{0,1\}^m \times \{0,1\}^t \rightarrow \{0,1\}^n$ is (ϵ, s)-*BSM secure* if for every $h : \{0,1\}^t \rightarrow \{0,1\}^s$ we have $d(f(K, R) \mid h(R), K) \leq \epsilon$, where K and R are independent and distributed uniformly over $\{0,1\}^m$ and $\{0,1\}^t$ respectively (R will often be called a *randomizer*) . In this paper we do not use the Bounded-Storage Model itself, but we just apply some of the theorems proved in this area, which state that for large t and small m and n there exist (ϵ, s)-BSM secure functions where ϵ is negligible, s is a constant fraction of t and which (on any input) access only a small part of R (see Sect. 10 for details)

6 Definition of IRSS

The definitions in this section were already discussed informally in Sect. 2.2. Let $\mathcal{P} = \{P_0, \ldots, P_{a-1}\}$ be a set of players. We start with the functional definition (the security definition is given in Sect. 6.1) of Intrusion-Resilient Secret-Sharing (the meaning of the frames in the definition below will be explained later).

Definition 1 (IRSS, functional definition) *An Intrusion-Resilient Secret Sharing (IRSS) scheme $\Xi_{a,\ell}$ is a protocol between a dealer and a players $\mathcal{P} = \{P_0, \ldots, P_{a-1}\}$. It*

consists of the following two algorithms, indexed by the number of players $a \in \mathbf{N}$ and a parameter $\ell \in \mathbf{N}$.

- *$share_{a,\ell}$ is a randomized algorithm that takes as input a message $M \in \{0,1\}^n$. It returns a sequence $T_0, \ldots, T_{a-1}$ of bit-strings, where each T_i is of length t.[3] The algorithm is executed by the dealer that later sends each T_i to a player P_i.*

- *$reconstruct_{a,\ell}$ is a deterministic algorithm that takes as input a sequence $(T_0, \ldots, T_{a-1})$ (as produced by the share algorithm) We require that always $reconstruct_{a,\ell}(share_{a,\ell}(M)) = M$. The output of $reconstruct_{a,\ell}(T_0, \ldots, T_{a-1})$ can be computed by a protocol between players $P_0, \ldots, P_{a-1}$ (player P_i holding T_i) in $a\ell$ rounds, where in round j ($0 \le j \le a\ell - 1$) a short message K_j is sent from from player $P_{j \bmod a}$ to $P_{j+1 \bmod a}$. Finally P_0 outputs the shared value.*

It will be convenient to define a (weak) variant of IRSS, which we call Intrusion-Resilient *Random*-Secret Sharing (IR**R**SS). Here the *share* algorithm does not take any input, but the shared message (i.e. the message output by *reconstruct*) will be random. Def. 1 is also a definition for IR**R**SS when ignoring the text in frames.

6.1 Adversarial Model

Let $\Xi_{a,\ell} = (share_{a,\ell}, reconstruct_{a,\ell})$ be an IRSS scheme as in Def. 1. Consider an adversary $\mathcal{A}$ that plays the following game against an oracle Ω. At the beginning, the adversary sends to the oracle a pair (M_0, M_1). The oracle chooses a random bit b and runs $share_{a,\ell}(M_b)$ to obtain the values $T_0, \ldots, T_{a-1}$. Now, the adversary can issue an (adaptively chosen) sequence $corrupt_1, \ldots, corrupt_e$ of corruption requests. Each request $corrupt_i$ is a pair (P_{c_i}, h_i), where $P_{c_i} \in \mathcal{P}$, and $h_i : \{0,1\}^t \to \{0,1\}^{s_i}$ is an arbitrary function. On input $corrupt_i$ the oracle replies with $H_i := h(T_{c_i})$. We will say that *the adversary chooses a corruption sequence* $\mathcal{C} = (P_{c_1}, \ldots, P_{c_e})$. Finally $\mathcal{A}$ outputs a bit $guess_{\mathcal{A}} \in \{0,1\}$, we say that *$\mathcal{A}$ breaks the scheme with an advantage* ϵ if $\epsilon = 2 \cdot \left| \mathsf{P}(guess_{\mathcal{A}} = b) - \frac{1}{2} \right|$.

Definition 2 (Corruption Path Length/Loops) *Let $\mathcal{C} = (P_{c_1}, \ldots, P_{c_w}) \in \mathcal{P}^*$ be a corruption sequence. The* corruption path length of *$\mathcal{C}$, denoted $cpl(\mathcal{C})$ is defined as the length of the maximal prefix of $(P_0, \ldots, P_{a-1})^*$, that is a subsequence of $\mathcal{C}$. We say that $\mathcal{C}$ makes ℓ loops, if $cpl(\mathcal{C})/a > \ell$. An example of a corruption sequence that makes 2 loops is given in (1), the underlined P_i's denoting the subsequence (of length 7) which is the longest prefix of $(P_0, P_1, P_2)^*$.*

Definition 3 (ℓ-admissible adversary) *An adversary $\mathcal{A}$ is ℓ-admissible, if any corruption sequence $\mathcal{C}$ chosen by $\mathcal{A}$ makes less than ℓ loops, i.e. $cpl(\mathcal{C}) \le a \cdot \ell$.*

Definition 4 (s-bounded adversary) *An adversary $\mathcal{A}$ is s-bounded, if the corruption sequence $\mathcal{C} = (P_{c_1}, \ldots, P_{c_w})$ chosen by $\mathcal{A}$ satisfies the following: for every $P_j \in \mathcal{P}$ we have $\sum s_i \le s$, where the summation is over all i such that $P_{c_i} = P_j$, and $s_i := |H_i|$ is the length of the output of h_i.*

To simplify the statements and the proofs of our results we assume that for any prefix $C^i = (P_{c_1}, \ldots, P_{c_i})$ of the corruption sequence chosen by $\mathcal{A}$ the message $K_{cpl(C^i)-1}$ (see Def. 1) is contained in $(H_1, \ldots, H_i)$. In other words, the adversary always computes all K_i's that he can trivially calculate by simulating the $reconstruct_{a,\ell}$ procedure.[4]

Definition 5 (Security of IRSS) *An IRSS scheme $\Xi_{a,\ell}$* is (ϵ, s)-secure *if every ℓ-admissible s-bounded adversary $\mathcal{A}$ breaks $\Xi_{a,\ell}$ with an advantage at most ϵ.*

To define security for Intrusion-Resilient *Random* Secret Sharing (IR**R**SS), we consider the same adversary model as for IRSS, except that now the adversary does not initially send to the oracle a pair (M_0, M_1) of messages. Moreover the final output is not just a bit, but can be an arbitrary string $out_{\widetilde{\mathcal{A}}}$. Let $\widetilde{M}$ denote the random string shared by the sharing algorithm, then we say that *$\widetilde{\mathcal{A}}$ breaks the IR**R**SS scheme with an advantage ϵ* if given the output of $\widetilde{\mathcal{A}}$, the distribution of $\widetilde{M}$ is ϵ far from uniform, i.e. $\epsilon = d(\widetilde{M} \mid out_{\widetilde{\mathcal{A}}})$.

Definition 6 (Security of IRRSS) *an IR**R**SS scheme $\widetilde{\Xi}_{a,\ell}$* is (ϵ, s)-secure *if every ℓ-admissible s-bounded adversary $\widetilde{\mathcal{A}}$ breaks $\widetilde{\Xi}_{a,\ell}$ with an advantage at most ϵ.*

IR**R**SS is a much weaker primitive than IRSS, and seems much easier to construct. By the following Lemma we can turn an IR**R**SS into an IRSS by using the shared random secret $\widetilde{M}$ as a one time pad to encrypt M. The security loss in this reduction is exponential in the length of the shared message.

Lemma 5 *Let $\widetilde{\Xi}_{a,\ell} = (share'_{a,\ell}, reconstruct'_{a,\ell})$ be an (ϵ, s)-secure IR**R**SS. Consider an IRSS $\Xi_{a,\ell} = (share_{a,\ell}, reconstruct_{a,\ell})$ constructed from $\widetilde{\Xi}_{a,\ell}$ as follows: algorithm $share_{a,\ell}(M)$ simply executes $share'_{a,\ell}$ (let $\mathcal{T}$ denote the resulting shares); P_0 additionally gets $C =$*

[3] In fact, in our schemes the T_i's may slightly differ in length: the share T_0 will be a little bit longer than the other shares. For simplicity we assume that all the lengths are equal, since the shorter shares can always be artificially "padded" to have length equal to t.

[4] The K_i's will be short compared with s, hence the adversary needs very little memory to retrieve them, and therefore we essentially do not loose generality by making this assumption.

$M \oplus \widetilde{M}$ *where* $\widetilde{M} \leftarrow reconstruct_{a,\ell}(\mathcal{T})$. *The procedure* $reconstruct_{a,\ell}(\mathcal{T}, C)$ *runs* $\widetilde{M} \leftarrow reconstruct'_{a,\ell}(\mathcal{T})$ *and output* $M = C \oplus \widetilde{M}$. *Then* $\Xi_{a,\ell}$ *is an* $(\epsilon \cdot 2^n, s)$ *secure IRSS.*

Proof Consider an s-bounded ℓ-admissible adversary $\mathcal{A}$ that breaks $\Xi_{a,\ell}$ with an advantage ζ. We construct an s-bounded ℓ-admissible adversary $\widetilde{\mathcal{A}}$ that attacks $\widetilde{\Xi}_{a,\ell}$, by simulating $\mathcal{A}$ in a black-box manner. Initially $\widetilde{\mathcal{A}}$ stores the messages M_0 and M_1 output by $\mathcal{A}$. Then $\widetilde{\mathcal{A}}$ simply lets $\mathcal{A}$ attack $\widetilde{\Xi}_{a,\ell}$ by forwarding his corruption requests to the oracle. The only nontrivial problem is how to handle the corruptions of P_0. This is because unlike in $\Xi_{a,\ell}$, in $\widetilde{\Xi}_{a,\ell}$ the player P_0 does not hold a value $C = M_b \oplus \widetilde{M}$. We let $\widetilde{\mathcal{A}}$ simply set C to be some random value $Z \in \{0,1\}^n$ (meaning that $\widetilde{\mathcal{A}}$ replaces h_i he gets from $\mathcal{A}$ with an h'_i which does exactly the same thing as h_i, but uses Z instead of C). Finally, $\mathcal{A}$ outputs his guess $guess_{\mathcal{A}}$, and we let $\widetilde{\mathcal{A}}$ output $out_{\widetilde{\mathcal{A}}} := Z \oplus M_{guess_{\mathcal{A}}}$. By Lemma 3 we get that $\mathsf{P}(out_{\widetilde{\mathcal{A}}} = \widetilde{M})$ is at most

$$2^{-n} + d(\widetilde{M} \mid out_{\widetilde{\mathcal{A}}}) \leq 2^{-n} + \epsilon, \qquad (2)$$

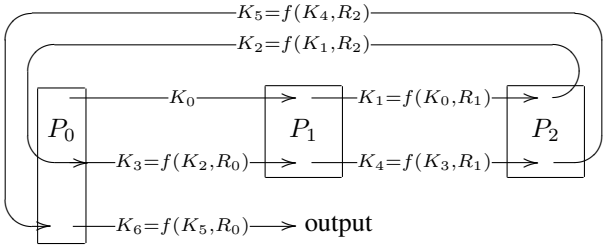
where (2) comes from (ϵ, s)-security of $\widetilde{\Xi}$. Now, suppose that for some $d \in \{0,1\}$ the following event $\mathcal{E}_d$ occurred: $Z = M_d \oplus \widetilde{M}$. In this case $\widetilde{\mathcal{A}}$ simply simulated the execution of $\mathcal{A}$ against an oracle Ω with $b = d$. Since Z is chosen uniformly thus $\mathsf{P}(\mathcal{E}_0) = \mathsf{P}(\mathcal{E}_1) = 2^{-n}$. Therefore conditioned on the event $\mathcal{E}_0 \cup \mathcal{E}_1$ the probability that $guess_{\mathcal{A}} = d$ is at least $\frac{1}{2} + \frac{1}{2} \cdot \zeta$. Thus we have that $\mathsf{P}(out_{\widetilde{\mathcal{A}}} = \widetilde{M})$ is at least equal to $\mathsf{P}(out_{\widetilde{\mathcal{A}}} = \widetilde{M} \mid \mathcal{E}_0 \cup \mathcal{E}_1) \cdot \mathsf{P}(\mathcal{E}_0 \cup \mathcal{E}_1)$, which is at least $\left(\frac{1}{2} + \frac{1}{2} \cdot \zeta\right) \cdot 2^{-n+1} = (1+\zeta) \cdot 2^{-n}$. Combining it with (2) we get $2^{-n} + \epsilon \geq (1+\zeta) \cdot 2^{-n}$, which implies that $\zeta \leq \epsilon \cdot 2^n$ as claimed. □

The following simple observation (whose proof appears in [16]) will also be useful.

Observation 1 *In the definition of IR**R**SS we can restrict ourselves to adversaries that are (1) deterministic, and (2) output* $out_{\widetilde{\mathcal{A}}} = (H_1, \ldots, H_w)$.

7 The Basic IRRSS/IRSS Schemes $\widetilde{\Xi}^f_{a,\ell}/\Xi^f_{a,\ell}$

In this section we construct a $(2^n \cdot a\ell\epsilon, s)$-secure Intrusion-Resilient Secret Sharing scheme from any (ϵ, s)-BSM secure function $f : \{0,1\}^m \times \{0,1\}^t \rightarrow \{0,1\}^m$ for a set $\mathcal{P} = \{P_0, \ldots, P_{a-1}\}$ of players. We follow the approach outlined in Lemma 5, i.e. we first construct an IR**R**SS, from which we then get an IRSS by using the shared random secret as a one-time pad to encrypt the message to be shared. The schemes are extremely simple, in round i of the reconstruction procedure, player $P_{i \bmod a}$ gets a (short) message K_i and uses it as a key to extract the next message K_{i+1} from his (large) randomizer $R_{i \bmod a}$ with the BSM-secure function f. The shared random message in the IR**R**SS is simply $K_{a\ell}$, the figure below illustrates the reconstruction procedure for $a = 3$ and $\ell = 2$.



The pair $(share_{a,\ell}, reconstruct_{a,\ell})$ below is an IR**R**SS scheme when ignoring the text in frames, we will refer to this IR**R**SS scheme as $\widetilde{\Xi}^f_{a,\ell}$. By adding the text in frames we get an IRSS (as outlined in Lemma 5), we will refer to this IRSS scheme as $\Xi^f_{a,\ell}$.

$share_{a,\ell}(M)$: Choose $K_0 \in \{0,1\}^m$ and $R_0, \ldots, R_{a-1} \in \{0,1\}^t$ uniformly at random. The share of each player $P_i \in \mathcal{P}$ is R_i. Player P_0 additionally gets K_0 and C which is computed as:

1. For $i = 1, \ldots, a\ell$ let $K_i := f(K_{i-1}, R_{i \bmod a})$
2. Set $C := M \oplus K_{a\ell}[1, \ldots, n]$

$reconstruct_{a,\ell}(K_0, R_0, \ldots, R_{a-1}, C)$: The players execute the following procedure

1. Player P_0 sends K_0 to P_1.
2. For $i = 1, \ldots, a\ell - 1$ player $P_{i \bmod a}$ sends $K_i = f(K_{i-1}, R_{i \bmod a})$ to player $P_{i+1 \bmod a}$.
3. P_0 computes $K_{a\ell} = f(K_{a\ell-1}, R_0)$ and outputs $K_{a\ell}[1, \ldots, n] \oplus C$.

7.1 Security

Theorem 1 *The IRSS scheme* $\Xi^f_{a,\ell}$ *for messages of length* n*, based on a* (ϵ, s)*-BSM secure function* $f : \{0,1\}^m \times \{0,1\}^t \rightarrow \{0,1\}^m$*, is* $(2^n \cdot a\ell\epsilon, s)$*-secure.*

Proof The security follows from the $(a\ell\epsilon, s)$-security of the IR**R**SS $\widetilde{\Xi}_{a\ell}$ (as proven in Lemma 7 below) by applying the IR**R**SS-to-IRSS reduction from Lemma 5. □

Before stating and proving Lemma 7 we present the following intuitive lemma (whose proof appears in [16]), which essentially states that an s-bounded adversary learns no more than s bits of each share as required by the definition of a BSM secure function.

Lemma 6 *Consider any player P_x holding the share R_x and let $(H_1, \ldots, H_i)$ be the information retrieved by an s-bounded adversary in the first i rounds. If f is an (ϵ, s)-secure BSM function then $d(f(K, R_x) \mid H_1, \ldots, H_i, K) \leq \epsilon$, where K is uniformly random (and independent of the other variables).*

Lemma 7 *The IRRSS scheme $\widetilde{\Xi}^f_{a,\ell}$ is $(a\ell\epsilon, s)$-secure.*

Proof Consider any adversary $\widetilde{\mathcal{A}}$ which attacks $\widetilde{\Xi}^f_{a,\ell}$. Let $P_{c_1}, P_{c_2}, \ldots$ denote the corruption sequence, and let $cpl_i = cpl(P_{c_1}, \ldots, P_{c_i})$. Note that $cpl_i = cpl_{i-1} + 1$ iff $cpl_{i-1} \bmod a = c_i$, which implies that always $cpl_i \bmod a \neq c_i$. Recall that $H_i = h_i(R_{c_i})$ is the information that $\widetilde{\mathcal{A}}$ learns in round i. To save on notation let $H^i = H_1, \ldots, H_i$, and let H^0 denote the empty sequence. Recall that in Def. 4 we assumed that $K_{cpl_i - 1}$ is known after the ith round, thus (using the convention that K_{-1} is empty)[5]

$$\mathsf{H}(K_{cpl_i - 1} \mid H^i) = 0. \tag{3}$$

We will prove by induction over i that K_{cpl_i} is close to uniform after the first i corruptions:

$$d(K_{cpl_i} \mid H^i) = d(K_{cpl_i} \mid H^i, K_{cpl_i - 1}) \leq cpl_i \cdot \epsilon. \tag{4}$$

After showing this we will be done since clearly $d(K_{cpl_i}[1, \ldots, n] \mid H^i) \leq d(K_{cpl_i} \mid H^i)$. For the induction basis, note that (4) is true for $i = 0$: we have $d(K_0 \mid H^0) = d(K_0) = 0$. We now prove that (4) holds for any $i > 0$ assuming it holds for $i - 1$. We make a case distinction, and first prove it for the rounds i where the cpl increases, i.e. for i's where $cpl_{i-1} + 1 = cpl_i$. The steps used in the following calculation are explained in detail below (the variable K in (8) is uniformly random).

$$\begin{aligned}
& d(K_{cpl_i} \mid H^i) \\
= \; & d(\overbrace{K_{cpl_i}}^{F} \mid \overbrace{H^{i-1}, K_{cpl_i - 1}}^{E}, \overbrace{H_i}^{T}) && (5)\\
= \; & d(K_{cpl_i} \mid H^{i-1}, K_{cpl_i - 1}) && (6)\\
= \; & d(f(K_{cpl_i - 1}, R_{cpl_i}) \mid H^{i-1}, K_{cpl_i - 1}) && (7)\\
\leq \; & d(f(K, R_{cpl_i}) | H^{i-1}, K) + d(K_{cpl_i - 1} | H^{i-1}) && (8)\\
\leq \; & \epsilon + d(K_{cpl_i - 1} \mid H^{i-1}) && (9)\\
= \; & \epsilon + d(K_{cpl_{i-1}} \mid H^{i-1}) && (10)\\
\leq \; & \epsilon + cpl_{i-1} \cdot \epsilon \leq cpl_i \cdot \epsilon && (11)
\end{aligned}$$

Step (5) just uses the definition of our scheme and (3). The next step (6) uses Lemma 2. To apply this Lemma one must show that $T \to E \to F$ is a Markov chain. Basically, this follows from Lemma 4 by identifying A from the lemma with R_{c_i}, further B with all the other R_j's, and $V_i = \phi(V_1, \ldots, V_{i-1}, A)$ with $H_i = h_i(R_{c_i})$, noting that the (adaptive) adversary computes h_i as a function of $H_1, \ldots, H_{i-1}$ (for space reasons, a detailed proof is only given in the full version [16]). Step (7) follows by definition and step (8) follows from Lemma 1. Step (9) follows from Lemma 6. Step (10) follows by the assumption that $cpl_{i-1} + 1 = cpl_i$. The last step (11) follow from the induction hypothesis (4) for $i - 1$. It remains to prove (4) for i's such that $cpl_{i-1} = cpl_i$:

$$\begin{aligned}
d(K_{cpl_i} | H^i) &= d(\overbrace{K_{cpl_i}}^{F} | \overbrace{H^{i-1}, K_{cpl_i - 1}}^{E}, \overbrace{H_i}^{T}) && (12)\\
&= d(K_{cpl_i} \mid H^{i-1}, K_{cpl_i - 1}) && (13)\\
&= d(K_{cpl_{i-1}} \mid H^{i-1}, K_{cpl_{i-1} - 1}) && (14)\\
&= d(K_{cpl_{i-1}} \mid H^{i-1}) && (15)\\
&= cpl_{i-1} \cdot \epsilon = cpl_i \cdot \epsilon && (16)
\end{aligned}$$

Step (13) uses Lemma 2 (again, to apply this lemma one must show that $T \to E \to F$ is a Markov chain, which follows from Lemma 4 as explained in detail in [16]). In step (14) and the very last step we use the assumption that $cpl_{i-1} = cpl_i$. Step (15) uses (3). Step (16) uses the induction hypothesis (4) for $i - 1$. □

8 Local Share Expansion

In this section we show a modified IRSS scheme where the shares each player receives from the dealer are very small, and where each player, after receiving the share from the dealer, blows up it up locally (this was informally discussed in Sect. 2). After the players have deleted their short shares that they have received from the dealer, we are basically in the same situation as in our original scheme $\Xi^f_{a,\ell}$ (except that now the players additionally have some Y_i values which they have to store), and we have exactly the same security guarantee (cf. Thm. 1) as our original scheme. The scheme is defined below and its security is proven in [16]. Besides the usual *share* and *reconstruct* procedure, now for $i = 0, \ldots, a-1$ there also is a procedure $expand^i$ which is run locally by player P_i after receiving the share.

$share_{a,\ell}(M)$: Choose $K_0, \ldots, K_{a \cdot \ell} \in \{0,1\}^m$ uniformly at random (below we use the convention that K_i is empty for $i \notin \{0, \ldots, a\ell\}$). Set $C := M \oplus K_{\ell \cdot a}[1, \ldots, n]$. For $i = -1, \ldots, a \cdot \ell$, send (K_i, K_{i+1}) to player $P_{i+1 \bmod a}$. Send C to player P_0.

$expand^i_{a,\ell}(\{(K_{i+ad-1}, K_{i+ad})\}_{d=0}^{\ell})$ Choose a randomizer $R_i \in \{0,1\}^t$ uniformly at random, and save it. For each (K_{j-1}, K_j) in the input (where both K_{j-1} and K_j are nonempty, i.e. $j \in \{1, \ldots, a\ell\}$) compute and save $Y_j = f(K_{j-1}, R_i) \oplus K_j$. If $i = 0$, save K_0. Delete all other K_i's.

[5] Recall that the symbol H denotes Shannon entropy.

$reconstruct_{a,\ell}(K_0, R_0, \ldots, R_{a-1}, C)$: The players execute the following procedure

1. Player P_0 sends K_0 to P_1
2. For $i = 1, \ldots, a\ell - 1$ player $P_{i \bmod a}$ sends $K_i = Y_i \oplus f(K_{i-1}, R_{i \bmod a})$ to player $P_{i+1 \bmod a}$.
3. P_0 computes $K_{a\ell} = Y_{a\ell} \oplus f(K_{a\ell-1}, R_0)$ and outputs $M = C \oplus (K_{a \cdot \ell}[1, \ldots, n])$.

9 Complete Leaks

Although security against adversaries as considered in Def. 5 is already quite strong, we did not consider so far the case when the adversary is able to learn one or more shares completely. In this section we show that for some particular class of BSM-secure functions (that we construct) the IRSS scheme from Sect. 7 is secure against such complete leaks. We don't actually know if already our basic IRSS scheme $\Xi^f_{a,\ell}$ is secure against full leaks if *any* BSM-secure function f is used. Our security proof of the IRSS $\Xi^f_{a,\ell}$, relying on the security of the IR**R**SS $\widetilde{\Xi}^f_{a,\ell}$, breaks down completely in this case as $\widetilde{\Xi}^f_{a,\ell}$ is not a secure IR**R**SS when the adversary is allowed to learn the complete share R_0 of player P_0. This is because $d(K_{a\cdot\ell} \mid R_0)$ can be large (so the shared secret $K_{a\ell}$ is far from uniform given R_0). To see this recall that $K_{a\ell} = f(K_{a\ell-1}, R_0)$, and let us assume for the moment that $f(\cdot, R_0) : \{0,1\}^m \to \{0,1\}^m$ behaves like a *uniformly random function*. It is a simple calculation that in this case the size of the range of $f(\cdot, R_0)$ is roughly $2^m(1 - 1/e)$, where $e = 2.71\ldots$ is Euler's number. But then, for any distribution of $K_{a\ell-1}$, the value of $f(K_{a\ell-1}, R_0)$ is far from uniformly random as it will avoid a $1/e$ fraction of all possible outputs (and an adversary who knows R_0 *completely*, will know what this $1/e$ fraction is). Our idea therefore is to limit ourselves to a special class of the BSM-secure functions that makes the above problem disappear, namely to functions $f(\cdot, R) : \{0,1\}^m \to \{0,1\}^m$ whose range is equal to $\{0,1\}^m$, or, equivalently, functions $f(\cdot, R)$ which are *permutation* (we will simply call such *f BSM permutations*). To formalize what it means to "leak a share completely", we define the following adversaries.

Definition 7 ((adaptive) strong ℓ-admissible adversary) *A* strong *ℓ-admissible adversary can initially choose a subset of* completely corrupted players $\mathcal{P}' \subset \mathcal{P}$*, and then gets the shares of all those players. Then he can attack according to any corruption sequence which does not make ℓ loops on the remaining players $\mathcal{P} \setminus \mathcal{P}' = \{P_{r_1}, \ldots, P_{r_x}\}$, i.e. if $P_{c_1}, \ldots, P_{c_w}$ is the corruption sequence then $(r_1, \ldots, r_x)^\ell$ must not be a subsequence of $c_1, \ldots, c_w$. An* adaptive *strong ℓ-admissible adversary is defined similarly, but he can choose the shares he wants to learn completely adaptively during the attack.*

We will say that an IRSS (IR**R**SS, resp.) scheme is $(\epsilon, s)^*$-*secure*, if it is (ϵ, s)-secure as in Def. 5 (Def. 6, resp.), with the only difference that in the definition "ℓ-admissible adversary" is replaced with "*strong* ℓ-admissible adversary". Analogously, we say that an IRSS scheme is $(\epsilon, s)^{**}$-secure if we instead consider "*adaptive* strong ℓ-admissible adversaries".

Observation 2 *Every IRSS or IR**R**SS scheme that is $(\epsilon, s)^*$-secure is also $(\epsilon 2^a, s)^{**}$-secure. This is because a non-adaptive adversary can always simulate the adaptive one, by guessing the set $\mathcal{P}'$ of players whose shares the adaptive adversary will choose to learn. This guess will be correct with probability 2^{-a}, hence the 2^a loss in security.*

Lemma 5 extends easily to the "$*$" and "$**$" security notions. Therefore it is enough to show how to construct an $(\epsilon, s)^*$-secure IR**R**SS. Such a scheme can be constructed by replacing an *arbitrary* BSM secure function f with a BSM permutation. Observe that if we base our IR**R**SS scheme $\Xi^\pi_{a,\ell}$ on a BSM secure *permutation* π, then we can write $K_i = \Pi_{i \bmod a}(K_{i-1})$ where Π_i is the permutation $\pi(., R_i)$. The effect of leaking a complete share R_i is thus the same as giving Π_i to the adversary. Consider an adversary which attacks $\Xi^\pi_{a,\ell}$, and let $\mathcal{P}'$ be the set of the completely corrupted players. From the viewpoint of the adversary the scheme now looks almost like the original one with the players being $\mathcal{P} \setminus \mathcal{P}'$, where occasionally the intermediate keys K_i get permuted. Say if the adversary knows Π_i, then K_{j-1} (where $i = j \bmod a$) is mapped to $K_j = \Pi_i(K_{j-1})$, but as the Π_i's are bijections, there is no entropy loss due to this mappings. Using this fact we can adapt (the proof of) Lemma 7 to the "$*$" notion, i.e.

Lemma 8 *The IR**R**SS $\widetilde{\Xi}^\pi_{a,\ell}$ scheme from Sect. 7, if based on a (ϵ, s)-BSM secure* permutation*, is $(a\ell\epsilon, s)^*$-secure and thus also $(2^a a\ell\epsilon, s)^{**}$-secure.*

And further by the reduction from Lemma 5 we get:

Theorem 2 *The IRSS scheme $\Xi^\pi_{a,\ell}$ for messages of length n, based on a (ϵ, s)-BSM secure* permutation *$\pi : \{0,1\}^m \times \{0,1\}^t \to \{0,1\}^m$, is $(2^n \cdot a\ell\epsilon, s)^*$-secure (and thus also $(2^{a+n} \cdot a\ell\epsilon, s)^{**}$-secure)*

Of course the above theorem is only interesting if we can come up with a BSM secure permutation. Below we show how to get such an object from any "normal" BSM secure function (the proof appears in [16]).

Theorem 3 *Let $f : \{0,1\}^m \times \{0,1\}^t \to \{0,1\}^m$ be an (ϵ, s)-BSM secure function, then $F : \{0,1\}^{2m} \times \{0,1\}^t \to \{0,1\}^{2m}$ defined as a two-round Feistel-network with f as round functions, i.e. $F(K_\ell \| K_r, R) := K_r \oplus f(K_\ell, R) \| K_\ell \oplus f(K_r \oplus f(K_\ell, R), R)$ is $(2\epsilon, s - m)$-BSM secure . Moreover, as the Feistel-network is a permutation, $F(\cdot, R)$ is a permutation on $\{0,1\}^{2m}$ for any $R \in \{0,1\}^t$.*

10 Concrete schemes

All the schemes that we constructed in this paper used as building block a BSM-secure function f. In this section we give examples of two concrete IRSS schemes based on the BSM functions of [24, 15]. The proofs of the corollaries below appear in [16].

Corollary 1 (Scheme based on the function of [24]) *For every $\beta \in [0,1)$, every n and t and every $\epsilon > 2^{n-t/2^{O(\log^* t)}}$ there exists an $(a\ell\epsilon, \beta t)$-secure IRSS scheme $\Xi^f_{a,\ell}$ for sharing messages of length $n \leq t(1-\beta)/2 - O(\log(1/\epsilon))$, such that (1) the size of each share is t (except the share of P_0 that has a size $t + O(\log t + \log(1/\epsilon) + n)$); (2) in the ith round the player $P_{i \bmod a}$ needs to access at most $O(\log t + \log(1/\epsilon) + n)$ bits of his share (except of P_0 that in the final round needs to access additionally n bits of his share); (3) the length of each message K_i communicated in the ith round of the reconstruction procedure is $O(\log t + \log(1/\epsilon) + n)$; (4) each K_i is computable in time polynomial in $\log t + \log(1/\epsilon) + n$.*

Corollary 2 (Scheme based on the function of [15]) *For every n, v and $L > 100$ there exists an $(a\ell\epsilon, s)$-secure IRSS scheme $\Xi^f_{a,\ell}$ for sharing messages of length n, such that (1) the size of each share is $t = v(L + n - 1)$ (except the share of P_0 that has a size $t + \lceil v \log_2 L \rceil + n$); (2) in the ith round the player $P_{i \bmod a}$ needs to access $v \lceil v \log_2 L \rceil$ bits of his share (except of P_0 that in the final round needs additionally to access n bits of his share); (3) the length of each message K_i (communicated in the ith round of the reconstruction procedure) is $v \lceil v \log_2 L \rceil$; (4) each K_i is computable in time linear in the number of accessed bits; (5) $\epsilon = \lceil v \log_2 L \rceil \cdot 2^{-v/2+n}$; (6) $s := 0.08t - 1.5v(\lceil v \log_2 L \rceil + 1)$.*

11 Computationally-secure IRSS

The scheme constructed in Sect. 7 has an obvious drawback that the messages K_i communicated in the reconstruction phase are longer than the shared secret M. We leave it as an open problem to show information-theoretically secure IRSS schemes with a smaller communication complexity.[6] One can make the messages very short (and independent of the length of M) at the cost of trading information-theoretic for computational security in a straightforward way: instead of sharing the message M directly, one shares a key k for a symmetric encryption (Enc, Dec) and gives player P_0 additionally the ciphertext $C = Enc(k, M)$ of M. Reconstruction is straightforward, P_0 now outputs $M = Dec(k, C)$ instead of the shared k. Denote this new computationally-secure scheme as $\Xi^{c1}_{a\ell}$.

This method is very similar to the one used to construct a computationally-secure Forward-Secure Storage scheme in [14]. We omit the formal proof that this construction is computationally secure (as long as the scheme (Enc, Dec) is semantically secure). This proof is similar to the proof of Lemma 3 in [14]. The security definition in this case is identical to the one in Sect. 6, except that now we must additionally require that $\mathcal{A}$ is a probabilistic polynomial-time machine and we must assume that the functions h_i (that are parts of the $corrupt_i$ requests, cf. Sec. 6.1) are efficiently computable (e.g. representable by polynomial size circuits).

Reducing the communication complexity even further – a connection with the theory of [17] We note that in general one has to be careful when switching to computational security in IRSS. The motivating question is as follows: can we reduce the length of the K_i's even further, assuming that the adversary is computationally bounded? First, let us look at the exact length of the K_i's. Clearly, the length depends on the scheme we use. Since in the scheme from Cor. 1 the exact values are hidden behind the O-notation let us concentrate on the scheme from Cor. 2. Here we have $|K_i| = v \log_2 L$, where v is such that $\epsilon = v \log_2 L \cdot 2^{-v/2+n}$ is negligible (and hence v has to be larger than $2n$), and L is a parameter that in a practical scheme would be relatively large: $L \approx 2^{20}$, say. Thus, $|K_i|$ is at least $2 \log_2 L \approx 40$ times larger than n. Therefore if we use the computationally secure IRSS presented above, then each K_i is at least 40 times longer than the key for symmetric encryption (Enc, Dec).

A natural way to reduce the communication complexity of $\Xi^{c1}_{a,\ell}$ even further, is to construct a new scheme $\Xi^{c2}_{a,\ell}$ as follows. Let $G : \{0,1\}^n \to \{0,1\}^m$ (for $n < m$) be a cryptographic pseudorandom generator. The scheme $\Xi^{c2}_{a,\ell}$ is defined as $\Xi^{c1}_{a,\ell}$ with the following difference. First, instead of taking $f : \{0,1\}^m \times \{0,1\}^t \to \{0,1\}^m$, take some BSM-secure function $f' : \{0,1\}^m \times \{0,1\}^t \to \{0,1\}^n$. In the *share* procedure (see Sect. 7) replace "$K_i := f(K_{i-1}, R_{i \bmod a})$" with "$K_i := G(K'_i)$, where $K'_i = f'(K_{i-1}, R_{i \bmod a})$". In the *reconstruct* procedure it is enough that (in Step 2) each $P_{i \bmod a}$ sends to $P_{i+1 \bmod a}$ the value K'_i, and then $P_{i+1 \bmod a}$ computes $K_i = G(K'_i)$ himself. In other words, instead of sending K_i we send its "compressed" version: K'_i. Observe that now the message communicated in each round is just a seed of G, and hence we can assume that it is equal to the length n of the key in the encryption scheme (Enc, Dec). Thus, it is significantly shorter than the size of the message communicated in each

[6] Observe that using the techniques from Sect. 7 it is impossible to make the K_i's shorter than M, since we have to pay the price of the $2^{|M|}$ factor in the advantage of the adversary, and hence to have $|K_i| < |M|$ we would need to construct an (ϵ, s)-BSM secure scheme with $\epsilon < 2^{|K|}$, which is impossible as the adversary can always guess K with probability $2^{|K|}$.

We also note that in every IR**R**SS scheme at least the last message $K_{a\ell-1}$ cannot be shorter than M, since the security of IRSS implies that the function defined as $E(K_{a\ell-1}, M) := (f(K_{a\ell-1}, R) \oplus M, R)$, (where R is random) is an information-theoretically secure (randomized) encryption scheme, and thus by Shannon's theorem $|M| \geq |K_{a\ell-1}|$.

round in scheme Ξ^{c1}. At first sight it may seem that this construction is secure (and it is of course secure if we model G as a *random oracle* [2]), however, a naive proof the security fails for the following reason: one needs to show that a (computationally bounded) adversary that sees R cannot *compress it* to shorter value $U = h(R)$, such that when he later learns K', he can distinguish $f(G(K'), R)$ from random. It is not clear how to show just from the assumption that G is a PRG. This problem is very similar to the problem of showing that the Φ_{c2} scheme of [14] (Sect. 6.3) is computationally secure, and is closely related to the theory of *compressibility of NP-instances* [17] (see also [11]).

References

[1] Y. Aumann, Y. Z. Ding, and M. O. Rabin. Everlasting security in the bounded storage model. *IEEE Transactions on Information Theory*, 48(6):1668–1680, 2002.

[2] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[3] G. R. Blakley. Safeguarding cryptographic keys. In *Proc. AFIPS 1979 National Computer Conference*, pages 313–317, 1979.

[4] R. Canetti, R. Gennaro, A. Herzberg, and D. Naor. Proactive security: Long-term protection against break-ins. *RSA CryptoBytes*, 3(1):1–8, 1997.

[5] D. Cash, Y. Z. Ding, Y. Dodis, W. Lee, R. J. Lipton, and S. Walfish. Intrusion-resilient key exchange in the bounded retrieval model. In *TCC'07*, volume 4392 of *LNCS*, pages 479–498, 2007.

[6] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley and Sons, Inc., 1991.

[7] G. Di Crescenzo, R. J. Lipton, and S. Walfish. Perfectly secure password protocols in the bounded retrieval model. In *TCC'06*, volume 3876 of *LNCS*, pages 225–244, 2006.

[8] D. Dagon, W. Lee, and R. J. Lipton. Protecting secret data from insider attacks. In *Financial Cryptography and Data Security*, pages 16–30, 2005.

[9] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[10] W. Diffie, P. C. van Oorschot, and M. J. Wiener. Authentication and authenticated key exchanges. *Designs, Codes and Cryptography*, 2(2):107–125, 1992.

[11] B. Dubrov and Y. Ishai. On the randomness complexity of efficient sampling. In *ACM Symposium on Theory of Computing*, pages 711–720, 2006.

[12] P. Duris, Z. Galil, and G. Schnitger. Lower bounds on communication complexity. *Inf. Comput.*, 73(1):1–22, 1987.

[13] S. Dziembowski. Intrusion-Resilience Via the Bounded-Storage Model. In *TCC'06*, volume 3876 of *LNCS*, pages 207–224. Springer, 2006.

[14] S. Dziembowski. On Forward-Secure Storage. In *CRYPTO'06*, volume 4117 of *LNCS*, pages 251–270, 2006.

[15] S. Dziembowski and U. Maurer. Optimal randomizer efficiency in the bounded-storage model. *Journal of Cryptology*, 17(1):5–26, January 2004.

[16] S. Dziembowski and K. Pietrzak. Intrusion-Resilient Secret Sharing. Cryptology ePrint Archive, 2007. full version of this paper.

[17] D. Harnik and M. Naor. On the compressibility of np instances and cryptographic applications. In *FOCS '06*, pages 719–728. IEEE, 2006.

[18] J. Hastad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[19] U. Maurer. Conditionally-perfect secrecy and a provably-secure randomized cipher. *Journal of Cryptology*, 5(1):53–66, 1992.

[20] U. Maurer. Secret key agreement by public discussion. *IEEE Transactions on Information Theory*, 39(3):733–742, 1993.

[21] N. Nisan and A. Widgerson. Rounds in communication complexity revisited. In *STOC '91*, pages 419–429. ACM, 1991.

[22] C. H. Papadimitriou and M. Sipser. Communication complexity. In *STOC*, pages 196–200. ACM, 1982.

[23] A. Shamir. How to share a secret. *Communications of the ACM*, 22:612–613, November 1979.

[24] S. P. Vadhan. Constructing locally computable extractors and cryptosystems in the bounded-storage model. *Journal of Cryptology*, 17(1):43–77, January 2004.

[25] S. Wiesner. Conjugate coding. *SIGACT News*, 15(1):78–88, 1983.

[26] A. Chi-Chih Yao. Some complexity questions related to distributive computing (preliminary report). In *STOC '79*, pages 209–213. ACM, 1979.