Secure Multi-party Computation: Information Flow of Outputs and Game Theory

Patrick Ah-Fat and Michael Huth(✉)

Department of Computing, Imperial College London, London SW7 2AZ, UK
{patrick.ah-fat14,m.huth}@imperial.ac.uk

Abstract. Secure multiparty computation enables protocol participants to compute the output of a public function of their private inputs whilst protecting the confidentiality of their inputs. But such an output, as a function of its inputs, inevitably leaks some information about input values regardless of the protocol used to compute it. We introduce foundations for quantifying and understanding how such leakage may influence input behaviour of deceitful protocol participants as well as that of participants they target. Our model captures the beliefs and knowledge that participants have about what input values other participants may choose. In this model, measures of information flow that may arise between protocol participants are introduced, formally investigated, and experimentally evaluated. These information-theoretic measures not only suggest advantageous input behaviour to deceitful participants for optimal updates of their beliefs about chosen inputs of targeted participants. They also allow targets to quantify the information-flow risk of their input choices. We show that this approach supports a game-theoretic formulation in which deceitful attackers wish to maximise the information that they gain on inputs of targets once the computation output is known, whereas the targets wish to protect the privacy of their inputs.

1 Introduction

In Secure Multiparty Computations (SMC), participants ought to provide their secret inputs and abide by a protocol in order to compute a public function in cooperation with the other parties. Such a protocol not only allows the participants to compute the correct output without having to rely on any other third party, but it also ensures that no information about the inputs leaks from the computation, other than that revealed by the output itself [5,10,14,29]. *Passive* adversaries are those parties who do abide by the protocol but try to infer as much information as possible on the other parties' inputs given all the information they get during the protocol. On the other hand, *active* adversaries will try to deviate from the protocol in order to learn more evidence on the other inputs. Private and robust protocols have been designed so as to deal with such kinds of adversaries, and to guarantee that the only information that leaks from the protocol is that based on the observation of the calculated output [1,4,7,10,15].

This information that is revealed about the inputs when the output is opened is called *acceptable leakage* and is commonly referred to in the literature as

M. Maffei and M. Ryan (Eds.): POST 2017, LNCS 10204, pp. 71–92, 2017.
DOI: 10.1007/978-3-662-54455-6_4

the only information that an SMC computation is allowed to leak about the inputs [10,19]. Current researches in SMC aim at building efficient protocols dealing with both types of adversaries [3,11,17,19,27] and take for granted that the function that is being calculated is secure, in that the knowledge of its output would not harm the privacy of any of the inputs [10,15,19]. However, it is not clear how one user, willing to take part into a secure computation, could assess the security of a function that is at stake. One of our aims is to build on information-theoretic principles to propose a measure of information flow that can occur in SMC and to quantify how secure the computation of a function is.

Moreover, private and robust protocols do not question the truthfulness of the input that each party actually provides. Indeed, we cannot prevent any party from giving an erroneous input as any input could be his actual choice. In particular, we cannot prevent a party from using a hazardous input which would question the privacy of the other inputs. This liberty that a party has to be able to influence the protocol, by deliberately choosing a particular input for his own benefit, is called *input substitution* [10]. This influence is part of, and is the only *allowed influence* that an SMC protocol tolerates. Again, our work introduces a measure of such an influence that an attacker can have on an SMC protocol and we derive a probabilistic analysis that both an attacker and an attacked party can use to quantify the information flows that could occur after computation.

Indeed, the choice of one party's input can make dramatic changes in the information he gets from the output. For instance, in the three-party computation of $a * b + c$ with positive integer inputs a, b and c, held by the respective parties A, B and C, player A is able to learn C's input by choosing 0 as input. We could further imagine that we know that c is bounded by an integer M and then player A could choose his input such that $a > M$, so that the knowledge of a and $a * b + c$ would let him learn both values of b and c, by Euclidean division.

In order to study the influence an attacker can gain from input substitution, it is helpful to define a new variant of attackers. In an SMC context, a passive adversary provides the input he was planning to use to the protocol, abides by the rules defined by the protocol, but is still *curious*, i.e. he will try to infer as much information as possible on the other parties given the information that he is sent during the protocol. On the other hand, the notion of active adversary reflects the fact that such a participant would try to convey inconsistent information during the protocol in order to maximise his information gain. We now introduce the notion of a *deceitful* adversary that reflects the will of a curious participant who abides by the protocol, to provide a judiciously chosen input that can be different from his honest and intended input, and which in particular can optimise the information that he seeks on the other inputs once he learns the output. We will also consider the coalition of several such deceitful adversaries, and we will allow them to attack any set of targets.

We can state the objectives set out for this paper as follows:

– To propose a model of attackers, targets and spectators that fits the context of SMC and enables us to reason about the information flows that may occur between the parties. This includes modelling the beliefs and knowledge of the participants as probability distributions.

- To define a mathematical measure of the information that a set of attackers can learn on a set of targets. This can guide attackers through input substitution and help them to choose a judicious input vector that would maximise their information gain on average.
- Conversely, to evaluate the risk that the targeted participants (referred to as 'targets' below) would run when entering a computation with given inputs.
- To show that these quantitative measures can also be used for preventive mechanisms taken to prevent a computation from happening that would otherwise seriously compromise the privacy of certain inputs.
- To extend our approach to a one-round game where a set of attackers A and a set of targets T both have to choose a vector of inputs, and where A tries to maximise the information he learns on T after computation whereas T tries to minimise this same amount of information.

Contributions of Our Paper: Our work explores one way of assessing the security of a function which is calculated with an SMC protocol, and we show that this notion of secure function can complement and thus harden the security that is ensured by the SMC protocols. We introduce the notion of deceitful adversaries who are willing to make use of input substitution to attack their targets. We implement a probabilistic approach based on information theory which enables us to quantify and predict the amount of information that such attackers would receive on their targets once a computation is executed. And we show that notions from game theory can be fruitfully applied to understand the strategic dynamics of input substitutions under the model of deceitful adversaries. In future work, we will study different use contexts of SMC – e.g. infrequent e-voting versus frequent accountancy computations. Different use contexts may then require different measures of information leakage, as discussed in Sect. 7 below, and different proactive or reactive behaviour.

Outline of Paper: We introduce some useful notions of SMC and information theory in Sect. 2. We give a formal definition of our model of attackers in Sect. 3. Then in Sect. 4 we present the probabilistic inferences that an attacker can make from the observation of the output of a secure computation. Section 5 shows how input substitution can be made practical thanks to this analysis and we present a game-theoretic setting that generalises this approach in Sect. 6. We compare our approach with recent works on information flow analysis in Sect. 7 and we conclude in Sect. 8.

2 Background

Secure Multiparty Computation: This domain of cryptography enables several players to compute the result of a public function of their private inputs, without having to rely on a third trusted party while ensuring that the inputs are kept secret during the computation. Protocols that achieve these tasks are split into two main categories. Yao's garbled circuit is the basis of protocols that are particularly adapted to secure 2-party computations of boolean functions [10,17,18,31]. Such protocols are based on oblivious transfer and security is achieved since only obfuscated values of intermediate results are shared between the parties. On the other hand, the Shamir secret sharing scheme [10,29,31] is designed to handle secure multiparty computation of arithmetic functions. It relies on Lagrange interpolation in finite fields and secrecy is ensured by realising the operations of the function algebraically on homomorphic shares of the inputs.

We study in this work arithmetic functions of more than 2 inputs. We thus assume that the functions we consider are securely computed for example by the CEAS protocol (Circuit Evaluation with Active Security) [10] based on the Shamir secret sharing scheme. Consequently, we considered only those operations that are allowed by CEAS, such as *addition*, *multiplication* and *multiplication by constant*. We also justify the use of *subtraction* in a similar manner as addition: using the notations introduced in CEAS, players holding two shared values $[\![a, f_a]\!]$ and $[\![b, f_b]\!]$ are able to share $[\![a-b, f_a - f_b]\!]$. With the recombination vector, they can then reconstruct the difference $d = a - b$. The only assumption that we have to make is that we work in the field $\mathbb{Z}/p\mathbb{Z}$ where the prime p is strictly greater than $M^+ + M^-$ which are respectively the absolute value of the greatest positive number and of the smallest negative number that the output (and inputs) of all the intermediate results can take. The output o of any computation is thus in $\mathbb{Z}/p\mathbb{Z}$ and non-negative; whenever $o > M^+$ is true, the actual result should be regarded as $o - p$, otherwise it would be o.

Information Theory: In order to measure the unpredictability of a random variable [30], or in other words, the amount of information we have on it, we recall the Shannon entropy $\mathrm{H}(X)$ defined for a random variable X defined on D_X by:

$$\mathrm{H}(X) = -\sum_{x \in D_X} p(x) \log p(x) \tag{1}$$

where log represents the binary logarithm also some times written $\log_2$. In this paper, we will use Shannon entropy as a measure of the amount of information that leaks from a computation. However, this choice is questionable and we will debate this decision in the discussion of Sect. 7.

Finally, for the sake of readability, we will abuse notation throughout the paper when there is no ambiguity. We will sum over a variable when the input domain is obvious: $\sum_{X_\mathbb{T}}$ will refer to $\sum_{\boldsymbol{x}_\mathbb{T} \in D_\mathbb{T}}$ and we will abbreviate the probability of an event as follows: $\sum_{X_\mathbb{T}} p(X_\mathbb{T})$ will refer to $\sum_{\boldsymbol{x}_\mathbb{T} \in D_\mathbb{T}} p(X_\mathbb{T} = \boldsymbol{x}_\mathbb{T})$.

3 Formal Setting

In this section, we develop a mathematical representation of an SMC computation that is suitable for information flow analysis.

We write $[\![1, n]\!]$ for $\{1, 2, \ldots, n\}$ below for $n \geq 1$. For n parties $P_1, \cdots, P_n$, we denote by $\mathbb{P}$ this set of parties and will partition $\mathbb{P}$ into 3 groups: Let $\mathbb{A} \subseteq \mathbb{P}$

be a set of attackers. Let us define $\mathbb{T} \subseteq (\mathbb{P} \setminus \mathbb{A})$ as a set of targets. Finally, the remaining parties $\mathbb{S} = \mathbb{P} \setminus (\mathbb{A} \cup \mathbb{T})$ are called the spectators.

In our model, each party $P_i \in \mathbb{P}$ will control one input $x_i \in D_i$ with its associated domain D_i, that we assume to be a finite subset of $\mathbb{Z}$. For the purpose of our analyses, we will often gather the inputs with respect to the 3 groups of participants $\mathbb{A}$, $\mathbb{T}$ and $\mathbb{S}$. By abuse of notation, we will write $\boldsymbol{x}_{\mathbb{A}} = (x_i)_{i \in \mathbb{A}}$ for the vector containing the attackers' inputs. Similarly, we will write $\boldsymbol{x}_{\mathbb{T}} = (x_i)_{i \in \mathbb{T}}$ and $\boldsymbol{x}_{\mathbb{S}} = (x_i)_{i \in \mathbb{S}}$ to refer to the targets' and the spectators' inputs, respectively. The same rule applies for the domain of those vector variables and we define the attackers' input domain $D_{\mathbb{A}} = \times_{a \in \mathbb{A}} D_a$, the targets' input domain $D_{\mathbb{T}} = \times_{t \in \mathbb{T}} D_t$ and the spectators' input domain $D_{\mathbb{S}} = \times_{s \in \mathbb{S}} D_s$.

Assumptions: We assume that every participant has a prior belief on each of the 3 vector inputs $\boldsymbol{x}_{\mathbb{A}}$, $\boldsymbol{x}_{\mathbb{T}}$ and $\boldsymbol{x}_{\mathbb{S}}$. We further assume that the attackers $\mathbb{A}$ and the targets $\mathbb{T}$ come to an agreement within their own group so that they share the same belief on those variables. Furthermore, we assume that these beliefs are public and that every group is aware of what the others think of their potential input. This would be a plausible assumption when all the groups believe that the inputs are uniformly distributed. We will aim in future works to weaken that assumption, e.g. to make such knowledge about distributions probabilistic itself. We represent these beliefs as probability distributions and we write $\pi_{\mathbb{A}} : D_{\mathbb{A}} \longrightarrow [0,1]$ for the distribution of the attackers' inputs from the targets' point of view. Similarly, we write $\pi_{\mathbb{T}} : D_{\mathbb{T}} \longrightarrow [0,1]$ for the distribution of the targets' inputs from the attackers' point of view. We also assume that attackers and targets have a shared prior belief $\pi_{\mathbb{S}} : D_{\mathbb{S}} \longrightarrow [0,1]$ on the spectators' inputs.

From the point of view of the attackers, the values of the input $\boldsymbol{x}_{\mathbb{T}}$ and $\boldsymbol{x}_{\mathbb{S}}$ appear as random variables that we call $X_{\mathbb{T}}$ and $X_{\mathbb{S}}$ respectively. Conversely, from the targets' point of view, the random variable corresponding to the attackers' inputs will be written $X_{\mathbb{A}}$. Note that a lack of prior knowledge on one group of parties may be represented as a uniform distribution over their input domain. As every party enters their input without knowing the input value of the other parties, we may assume that the variables $X_{\mathbb{A}}$, $X_{\mathbb{T}}$ and $X_{\mathbb{S}}$ are independent.

Finally, let us consider an n-ary function f with domain $\times_{i \in \mathbb{P}} D_i \longrightarrow D_O$. As a function of discrete random variables, the output of f will also be considered as a random variable, written O.

We want to observe how the attackers can update their beliefs on the targeted parties once the output of the function f is revealed. We thus define $\pi_{\mathbb{T},o}^{\mathbb{A},\boldsymbol{x}_{\mathbb{A}}} : D_{\mathbb{T}} \longrightarrow [0,1]$ which returns the posterior joint probability distribution of a set of targeted values $\boldsymbol{x}_{\mathbb{T}} \in D_{\mathbb{T}}$, given an observed output $o \in D_O$ and a set of attackers' inputs $\boldsymbol{x}_{\mathbb{A}} \in D_{\mathbb{A}}$. We will detail the calculation of this function next.

4 Information Flow from One Observed Public Output

We next present how the set of attackers $\mathbb{A}$ can use probabilistic inference to update their belief on the inputs of the targets in $\mathbb{T}$ once the values of $x_1, \cdots, x_n$ have been input and the output of the public function has been calculated.

First, based on the attackers' beliefs on the targets' and spectators' distributions $\pi_{\mathbb{T}}$ and $\pi_{\mathbb{S}}$, the attackers calculate the prior distribution of the output O, that they will use to update their belief on $X_{\mathbb{T}}$. From the attackers' point of view and for a given set of values $\boldsymbol{x}_{\mathbb{A}} = (x_a)_{a \in \mathbb{A}}$ in $D_{\mathbb{A}}$, the probability for the output O to be a given value o takes into account all the combinations of $\boldsymbol{x}_{\mathbb{T}} = (x_t)_{t \in \mathbb{T}}$ and $\boldsymbol{x}_{\mathbb{S}} = (x_s)_{s \in \mathbb{S}}$ that satisfy $f(x_1, \cdots, x_n) = o$. For every o in the output domain D_O, we have the following forward propagation:

$$p(O = o \mid X_{\mathbb{A}} = \boldsymbol{x}_{\mathbb{A}}) = \sum_{\substack{X_{\mathbb{T}} \\ X_{\mathbb{S}} \\ f(x_1,\cdots,x_n)=o}} p(X_{\mathbb{T}}, X_{\mathbb{S}}) \tag{2}$$

We now consider the actual public output o as hard evidence in order to update the attackers' beliefs on the targets' inputs probability distribution. By virtue of Bayes' theorem with conditional probabilities and noticing that $p(X_{\mathbb{T}} = \boldsymbol{x}_{\mathbb{T}} \mid X_{\mathbb{A}} = \boldsymbol{x}_{\mathbb{A}}) = p(X_{\mathbb{T}} = \boldsymbol{x}_{\mathbb{T}})$ since $X_{\mathbb{A}}$ and $X_{\mathbb{T}}$ are independent, the backward propagation can be written as follows, for every $\boldsymbol{x}_{\mathbb{T}}$ in $D_{\mathbb{T}}$:

$$p(X_{\mathbb{T}} = \boldsymbol{x}_{\mathbb{T}} \mid O = o, X_{\mathbb{A}} = \boldsymbol{x}_{\mathbb{A}}) = \frac{p(O = o \mid X_{\mathbb{T}} = \boldsymbol{x}_{\mathbb{T}}) \cdot p(X_{\mathbb{T}} = \boldsymbol{x}_{\mathbb{T}})}{p(O = o \mid X_{\mathbb{A}} = \boldsymbol{x}_{\mathbb{A}})} \tag{3}$$

The attackers' belief on the probability distribution of $X_{\mathbb{T}}$ gives us $p(X_t = x_t)$ for each $\boldsymbol{x}_{\mathbb{T}} \in D_{\mathbb{T}}$. We have also just calculated the prior probability for the output $p(O = o)$ in (2). Finally, the conditional probability $p(O = o \mid X_{\mathbb{T}} = \boldsymbol{x}_{\mathbb{T}}, X_{\mathbb{A}} = \boldsymbol{x}_{\mathbb{A}})$ of O given $\boldsymbol{x}_{\mathbb{T}}$ can be calculated in a similar manner as in (2) for the prior probability distribution of O. Indeed, we take into account all the combinations of $\boldsymbol{x}_{\mathbb{S}}$ in $D_{\mathbb{S}}$ that satisfy $f(x_1, \cdots, x_n) = o$ and we have:

$$p(O = o \mid X_{\mathbb{T}} = \boldsymbol{x}_{\mathbb{T}}, X_{\mathbb{A}} = \boldsymbol{x}_{\mathbb{A}}) = \sum_{\substack{X_{\mathbb{S}} \\ f(x_1,\cdots,x_n)=o}} p(X_{\mathbb{S}}) \tag{4}$$

This gives the attackers a way of recovering the joint probability distribution of their targets $\pi_{\mathbb{T},o}^{\mathbb{A},\boldsymbol{x_a}} : D_{\mathbb{T}} \longrightarrow [0,1]$, defined for any vector $\boldsymbol{x}_{\mathbb{T}}$ in $D_{\mathbb{T}}$, $\boldsymbol{x}_{\mathbb{A}}$ in $D_{\mathbb{A}}$ and o in D_O by:

$$\pi_{\mathbb{T},o}^{\mathbb{A},\boldsymbol{x}_{\mathbb{A}}}(\boldsymbol{x}_{\mathbb{T}}) = p(X_{\mathbb{T}} = \boldsymbol{x}_{\mathbb{T}} \mid O = o, X_{\mathbb{A}} = \boldsymbol{x}_{\mathbb{A}}) \tag{5}$$

Based on this posterior distribution, we can measure the amount of information we obtain by calculating $\mathrm{H}(X_{\mathbb{T}} \mid X_{\mathbb{A}} = \boldsymbol{x}_{\mathbb{A}}, O = o)$, the specific, conditional entropy of $X_{\mathbb{T}}$ given the values $\boldsymbol{x}_{\mathbb{A}}$ and o that we formalise next.

Definition 1. *We define the entropy of the inputs of a set of targets $\mathbb{T}$ attacked by a set of attackers $\mathbb{A}$ as the function $ent_{\mathbb{T}}^{\mathbb{A}} : D_{\mathbb{A}} \times D_O \longrightarrow \mathbb{R}_0^+$ defined for a given observed output o and a vector of attackers' inputs $\boldsymbol{x}_{\mathbb{A}} \in D_{\mathbb{A}}$ as:*

$$ent_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}, o) = \sum_{X_{\mathbb{T}}} g(p(X_{\mathbb{T}} \mid X_{\mathbb{A}} = \boldsymbol{x}_{\mathbb{A}}, O = o)) \tag{6}$$

where we define $g(x) = -x \log x$.

Note 1. The entropy of the posterior distribution $\pi_{\mathbb{T},o}^{\mathbb{A};\boldsymbol{x}_{\mathbb{A}}}$ gives us an idea of the amount of information that a set of attackers having input $\boldsymbol{x}_{\mathbb{A}}$ gets on $X_{\mathbb{T}}$ once and *after* they learn the public output o.

5 Anticipated Information Flow from Expected Outputs

The latter notion of entropy informs the attackers of how much information they have on $X_{\mathbb{T}}$ once the computation has been realised and the output o has been revealed to everyone. However, the attackers and the targets may want to measure the average amount of information that would leak from the result o of a function before its computation occurs — and thus before we learn the value of o. This would enable the attackers to estimate the amount of information they would gain, whereas the targets would be able to evaluate the risk that they run before entering a computation. We thus calculate the average entropy of variable $X_{\mathbb{T}}$ over all possible outputs weighted by their likelihood given that $\boldsymbol{x}_{\mathbb{A}}$ and $\boldsymbol{x}_{\mathbb{T}}$ are known. We formally define an indicator that reflects this.

Definition 2. *The* joint weighted average entropy *of variable $X_{\mathbb{T}}$ attacked by parties $\mathbb{A}$ is the function* $\mathrm{jwae}_{\mathbb{T}}^{\mathbb{A}} : D_{\mathbb{A}} \times D_{\mathbb{T}} \longrightarrow \mathbb{R}^{+}$ *defined for all $\boldsymbol{x}_{\mathbb{A}} \in D_{\mathbb{A}}$ and $\boldsymbol{x}_{\mathbb{T}} \in D_{\mathbb{T}}$ by:*

$$\mathrm{jwae}_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}, \boldsymbol{x}_{\mathbb{T}}) = \sum_{o \in D_O} p(O = o \mid X_{\mathbb{A}} = \boldsymbol{x}_{\mathbb{A}}, X_{\mathbb{T}} = \boldsymbol{x}_{\mathbb{T}}) \cdot ent_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}, o) \tag{7}$$

The measure defined in (7) informs us about the information that the attackers would learn on average about the targets if $\boldsymbol{x}_{\mathbb{A}}$ and $\boldsymbol{x}_{\mathbb{T}}$ were chosen. But as a group of attackers, parties $\mathbb{A}$ would be interested in measuring how informative one of their inputs $\boldsymbol{x}_{\mathbb{A}}$ is, regardless of what the targets choose. We thus define the average of the joint weighted average entropy over all possible values of $\boldsymbol{x}_{\mathbb{T}}$ weighted by their prior probabilities.

Definition 3. *The attackers' weighted average entropy of variable $X_{\mathbb{T}}$ attacked by parties $\mathbb{A}$ is the function* $\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}} : D_{\mathbb{A}} \longrightarrow \mathbb{R}^{+}$ *defined for all $\boldsymbol{x}_{\mathbb{A}} \in D_{\mathbb{A}}$ by:*

$$\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}) = \sum_{\boldsymbol{x}_{\mathbb{T}} \in D_{\mathbb{T}}} p(X_{\mathbb{T}} = \boldsymbol{x}_{\mathbb{T}}) \cdot \mathrm{jwae}_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}, \boldsymbol{x}_{\mathbb{T}}) \tag{8}$$

We also define the targets' weighted average entropy as the function $\mathrm{twae}_{\mathbb{T}}^{\mathbb{A}} : D_{\mathbb{T}} \longrightarrow \mathbb{R}^{+}$ *defined for all $\boldsymbol{x}_{\mathbb{T}} \in D_{\mathbb{T}}$ by:*

$$\mathrm{twae}_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{T}}) = \sum_{\boldsymbol{x}_{\mathbb{A}} \in D_{\mathbb{A}}} p(X_{\mathbb{A}} = \boldsymbol{x}_{\mathbb{A}}) \cdot \mathrm{jwae}_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}, \boldsymbol{x}_{\mathbb{T}}) \tag{9}$$

Note 2. We notice that the attackers' weighted average entropy can be written:

$$\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}) = \sum_{o \in D_O} ent_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}, o) \left(\sum_{X_{\mathbb{T}}} p(X_{\mathbb{T}}) \cdot p(O = o \mid X_{\mathbb{A}} = \boldsymbol{x}_{\mathbb{A}}, X_{\mathbb{T}}) \right) \tag{10}$$

But we know that $p(X_{\mathbb{T}}) = p(X_{\mathbb{T}} \mid X_{\mathbb{A}} = \boldsymbol{x}_{\mathbb{A}})$ since those two random variables are deemed to be independent. The law of total probabilities with conditional probabilities thus gives us:

$$\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}) = \sum_{o \in D_O} p(O = o \mid X_{\mathbb{A}} = \boldsymbol{x}_{\mathbb{A}}) \cdot ent_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}, o) \tag{11}$$

that we can identify to $\mathrm{H}(X_{\mathbb{T}} \mid X_{\mathbb{A}} = \boldsymbol{x}_{\mathbb{A}}, O)$, the posterior conditional Shannon entropy of $X_{\mathbb{T}}$ given $\boldsymbol{x}_{\mathbb{A}}$ and O.

This function informs the attackers of how much information they are likely to learn on $X_{\mathbb{T}}$ depending on the input vector $\boldsymbol{x}_{\mathbb{A}}$ that they choose. The lower measure $\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}})$ is, the more information $\boldsymbol{x}_{\mathbb{A}}$ reveals on $X_{\mathbb{T}}$. Let us illustrate those definitions through simple examples and show that they can be used by the parties to measure their expected entropy once the output is computed.

Example 1. Let us consider 3 parties X, Y and Z holding inputs corresponding to the respective lower case letters. Let us consider the function f defined by $f(x, y, z) = 3xy - 2yz$. We further imagine that attacker $\mathbb{A} = \{X\}$ is attacking target $\mathbb{T} = \{Y\}$, with a spectator $\mathbb{S} = \{Z\}$. Let us first study the target's weighted average entropy and draw the values of $\mathrm{twae}_{\mathbb{T}}^{\mathbb{A}}(y)$ in Fig. 1.

This graph informs the target party $\mathbb{T}$ of how much information the attacker will learn on average once the output is computed. We can see that those results are not easily predictable and that the computation of $\mathrm{twae}_{\mathbb{T}}^{\mathbb{A}}(y)$ can indeed be useful for party $\mathbb{T}$. For example, we have $\mathrm{twae}_{\mathbb{T}}^{\mathbb{A}}(6) \simeq 1.57$ and $\mathrm{twae}_{\mathbb{T}}^{\mathbb{A}}(29) \simeq 0.40$, which means that on average, input $y = 29$ would be more easily guessed by the attackers than input $y = 6$.

Conversely, we can compute the attacker's weighted average entropy for his possible inputs. We draw in Fig. 2 the values of $\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}}(x)$ for all values of x.

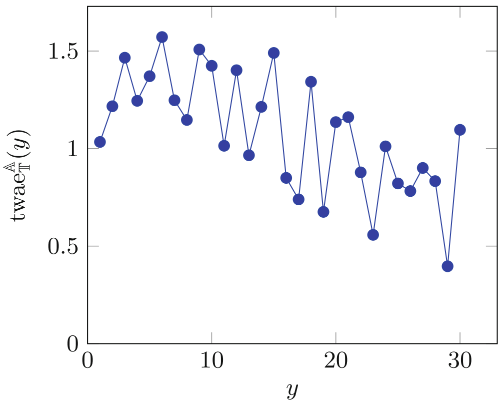

Fig. 1. Behaviour of $\mathrm{twae}_{\mathbb{T}}^{\mathbb{A}}(y)$ for the function $f(x, y, z) = 3xy - 2yz$ with inputs ranged and uniformly distributed in $[\![1, 30]\!]$, where $\mathbb{A} = \{X\}$, $\mathbb{T} = \{Y\}$ and $\mathbb{S} = \{Z\}$.

Here again, we can first observe that the results we obtain are not straightforward. Indeed, some inputs x are more informative than others. In particular, we can speculate that on average, the attacker X would learn more information about the target's input if he chooses an odd input x.

In this simple example, we can have an intuitive idea that would explain this conjecture. Indeed, the expression of f can be written as follows: $f(x, y, z) = y(3x - 2z)$. Consequently, if x is odd, $3x$ will be odd and necessarily $3x - 2z$ will be odd as well. Thus, the parity of variable y is determined by the parity of the output. More precisely, we know that y is odd if and only if the output o is odd, which would rule out half of the possibilities for y, and which could not be possible if x was even. However, in more complex cases, a mathematical explanation that would predict the amount of information that an attacker could learn on a target is not available in general. The computation and inspection of $\text{awae}_{\mathbb{T}}^{\mathbb{A}}$ could give us an estimate of such information flow.

This example also shows us another interesting feature that an attacker could use. Let us imagine that the attacker $\mathbb{A}$ has as honest, intended input the value $x = 14$. We can imagine that this attacker would like to learn as much information as possible on the input of target $\mathbb{T}$, but that he would also be mindful of the accuracy of the output of the SMC computation. Then, the graph in Fig. 2 would again allow him to substitute his input for another one, say $x = 15$, that is as close as possible as his honest input, but that will also give him much more information on y. On the other hand, we could also imagine that the parties are intelligent enough to detect if one of the inputs has deliberately been corrupted. In particular, the targets could also run probabilistic analyses on the attackers' inputs. Then, choosing a corrupted input that is close to his honest input might enable an attacker to overcome the risk of being accused of cheating.

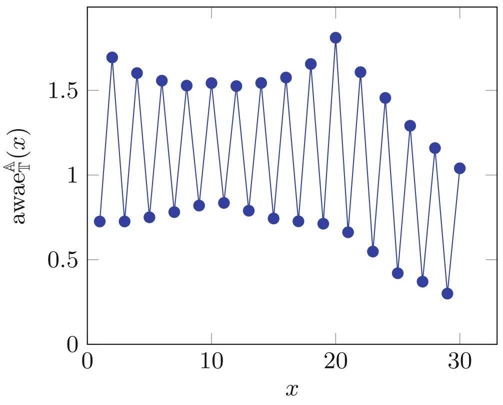

Fig. 2. Behaviour of $\text{awae}_{\mathbb{T}}^{\mathbb{A}}(x)$ for the function $f(x, y, z) = 3xy - 2yz$ with inputs uniformly distributed over $[\![1, 30]\!]$, where $\mathbb{A} = \{X\}$, $\mathbb{T} = \{Y\}$ and $\mathbb{S} = \{Z\}$.

Example 2. We would like to explore if measures $\text{awae}_{\mathbb{T}}^{\mathbb{A}}$ and $\text{twae}_{\mathbb{T}}^{\mathbb{A}}$ effectively helps us to predict the average amount of information flow that occurs after a computation reveals its output. In order to assess the information that these functions provide, we measure the average number of tries that the attacker $\mathbb{A}$ needs in order to guess the targeted inputs $\boldsymbol{x}_{\mathbb{T}}$ once the computation is executed, and given the values of $\text{awae}_{\mathbb{T}}^{\mathbb{A}}$ or $\text{twae}_{\mathbb{T}}^{\mathbb{A}}$ respectively. Theoretically, we would expect this guessing entropy to grow at least exponentially with the Shannon entropy [23]. We consider the same function $f(x, y, z) = 3xy - 2yz$ as in Example 1 with the same groups $\mathbb{A} = \{X\}$, $\mathbb{T} = \{Y\}$ and $\mathbb{S} = \{Z\}$ having uniforms prior beliefs on the other inputs over $[\![1, 30]\!]$, and we perform the following test.

For each value of $x \in [\![1, 30]\!]$, we randomly pick some value for y and z and calculate the output $o = f(x, y, z)$. We then let $\mathbb{A} = \{X\}$ guess the value of $\boldsymbol{x}_{\mathbb{T}} = \{y\}$ given its posterior distribution and the knowledge of o. In particular, $\mathbb{A}$ is more likely to try a value of $\boldsymbol{x}_{\mathbb{T}}$ that has a high posterior probability. We record the number of tries that were needed for $\mathbb{A}$ to find the correct value of $\boldsymbol{x}_{\mathbb{T}}$ and repeat the iteration 500 times. We report in Fig. 3 the average number of tries n_{guess} as a function of $\text{awae}_{\mathbb{T}}^{\mathbb{A}}(x)$ for the 30 values of x that we tested. We also performed the same regression test by fixing the initial value of y and choosing a random value for x in order to reflect the point of view of $\mathbb{T}$, and we also plotted the average number of tries n_{guess} that $\mathbb{A}$ needs to guess $\boldsymbol{x}_{\mathbb{T}}$ as a function of $\text{twae}_{\mathbb{T}}^{\mathbb{A}}(y)$ for all values of $y \in [\![1, 30]\!]$. The results displayed in Fig. 3 show that the higher the value of $\text{awae}_{\mathbb{T}}^{\mathbb{A}}(x)$ is, the more tries an attacker would need to guess $\boldsymbol{x}_{\mathbb{T}}$. Conversely, the higher the value of $\text{twae}_{\mathbb{T}}^{\mathbb{A}}(y)$ is, the more tries $\mathbb{A}$ needs to guess $\boldsymbol{x}_{\mathbb{T}}$, which confirms the intuitive meaning of those functions.

The graph of Fig. 3 also suggests that measure $\text{twae}_{\mathbb{T}}^{\mathbb{A}}$ predicts occurring information flow less precisely than $\text{awae}_{\mathbb{T}}^{\mathbb{A}}$ does. This can be explained by the fact that $\text{twae}_{\mathbb{T}}^{\mathbb{A}}$ not only takes into account the beliefs that $\mathbb{A}$ has on $\boldsymbol{x}_{\mathbb{T}}$, but it also considers the belief that $\mathbb{T}$ has on $\boldsymbol{x}_{\mathbb{A}}$, which $\text{awae}_{\mathbb{T}}^{\mathbb{A}}$ does not need.

We now present some mathematical properties of $\text{awae}_{\mathbb{T}}^{\mathbb{A}}$. A set of attackers $\mathbb{A}$ can manage to learn no less information on a set of targets $\mathbb{T}'$ than they can learn on a set of target $\mathbb{T}$ if $\mathbb{T}' \subseteq \mathbb{T}$. We formalise the idea in the theorem below.

Theorem 1. *Let $\mathbb{P}$ be a set of parties partitioned into a set of attackers $\mathbb{A}$, targets $\mathbb{T}$, and spectators $\mathbb{S}$. We also consider a subset of targets $\mathbb{T}' \subseteq \mathbb{T}$. For all attackers' input $\boldsymbol{x}_{\mathbb{A}} \in D_{\mathbb{A}}$, we then have:*

$$\text{awae}_{\mathbb{T}'}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}) \leq \text{awae}_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}) \tag{12}$$

Proof. For the sake of readability, we will use the following abuse of notation in the proofs. For a given vector $\boldsymbol{x}_{\mathbb{A}}$, we will abbreviate the probability $p(X_{\mathbb{A}} = \boldsymbol{x}_{\mathbb{A}})$ as $p(\boldsymbol{x}_{\mathbb{A}})$. Similarly, for a given event X, we will write the conditional probability $p(X \mid X_{\mathbb{A}} = \boldsymbol{x}_{\mathbb{A}})$ as $p(X \mid \boldsymbol{x}_{\mathbb{A}})$. The expression of the $\text{awae}_{\mathbb{T}}^{\mathbb{A}}$ for the set of targets $\mathbb{T} = \mathbb{T}' \cup \mathbb{T}''$, where $\mathbb{T}'' = \mathbb{T} \setminus \mathbb{T}'$, and for an input $\boldsymbol{x}_{\mathbb{A}} \in D_{\mathbb{A}}$ is defined as:

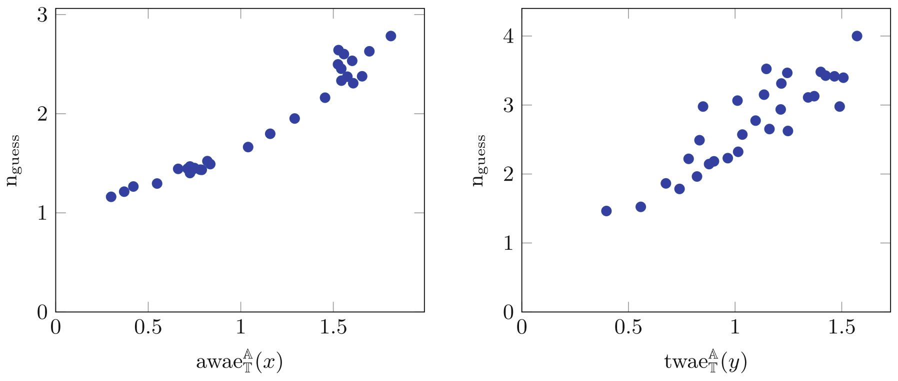

Fig. 3. Correlation between the average number of guesses n_{guess} (that $\mathbb{A}$ needs on average to guess $\boldsymbol{x}_{\mathbb{T}}$) and the values of $\text{awae}_{\mathbb{T}}^{\mathbb{A}}(x)$ and $\text{twae}_{\mathbb{T}}^{\mathbb{A}}(y)$ for the function $f(x,y,z) = 3xy - 2yz$ with inputs ranged in $[\![1,30]\!]$, where $\mathbb{A} = \{X\}$, $\mathbb{T} = \{Y\}$ and $\mathbb{S} = \{Z\}$. We ran 30 tests for each values of x (left) and 30 tests for each values of y (right) that we detail in Example 2.

$$\begin{aligned}\text{awae}_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}) &= \sum_{O} p(O \mid \boldsymbol{x}_{\mathbb{A}}) \cdot \left[\sum_{X_{\mathbb{T}}} g(p(X_{\mathbb{T}} \mid O, \boldsymbol{x}_{\mathbb{A}}))\right] \\ &= \sum_{O} p(O \mid \boldsymbol{x}_{\mathbb{A}}) \cdot \left[\sum_{X_{\mathbb{T}'}} \sum_{X_{\mathbb{T}''}} g(p(X_{\mathbb{T}'}, X_{\mathbb{T}''} \mid O, \boldsymbol{x}_{\mathbb{A}}))\right] \qquad (13)\end{aligned}$$

where we recall $g(x) = -x \log x$.

As the function g is concave, we have:

$$\sum_{X_{\mathbb{T}''}} g\left(p(X_{\mathbb{T}'}, X_{\mathbb{T}''} \mid O, \boldsymbol{x}_{\mathbb{A}})\right) \geq g\Big(\sum_{X_{\mathbb{T}''}} p(X_{\mathbb{T}'}, X_{\mathbb{T}''} \mid O, \boldsymbol{x}_{\mathbb{A}})\Big)$$

So Eq. (13) becomes:

$$\text{awae}_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}) \geq \sum_{O} p(O \mid \boldsymbol{x}_{\mathbb{A}}) \cdot \left[\sum_{X_{\mathbb{T}'}} g\Big(\sum_{X_{\mathbb{T}''}} p(X_{\mathbb{T}'}, X_{\mathbb{T}''} \mid O, \boldsymbol{x}_{\mathbb{A}})\Big)\right]$$

But we know that $\sum_{X_{\mathbb{T}''}} p(X_{\mathbb{T}'}, X_{\mathbb{T}''} \mid O, \boldsymbol{x}_{\mathbb{A}}) = p(X_{\mathbb{T}'} \mid O, \boldsymbol{x}_{\mathbb{A}})$, so we get:

$$\text{awae}_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}) \geq \sum_{O} p(O \mid \boldsymbol{x}_{\mathbb{A}}) \cdot \left[\sum_{X_{\mathbb{T}'}} g(p(X_{\mathbb{T}'} \mid O, \boldsymbol{x}_{\mathbb{A}}))\right]$$

which is equivalent to the expected result:

$$\text{awae}_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}) \geq \text{awae}_{\mathbb{T}'}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}) \qquad \blacksquare$$

We have seen that the smaller the set of targets, the more information the attackers can learn. We show now a similar result: intuitively, the fewer the attackers are, the less information they can infer.

Theorem 2. *Let $\mathbb{P}$ be a set of parties partitioned into a set of attackers $\mathbb{A}$, targets $\mathbb{T}$ and spectators $\mathbb{S}$. We also define an additional set of parties $\mathbb{A}'' \subseteq \mathbb{A}$ that will turn into spectators by setting $\mathbb{A}' = \mathbb{A} \setminus \mathbb{A}''$ and $\mathbb{S}' = \mathbb{S} \cup \mathbb{A}''$. We have:*

$$\forall \boldsymbol{x}_{\mathbb{A}'} \in D_{\mathbb{A}'}. \min_{\boldsymbol{x}_{\mathbb{A}''} \in D_{\mathbb{A}''}} \mathrm{awae}_{\mathbb{T}}^{\mathbb{A}} \begin{pmatrix} \boldsymbol{x}_{\mathbb{A}'} \\ \boldsymbol{x}_{\mathbb{A}''} \end{pmatrix} \leq \mathrm{awae}_{\mathbb{T}}^{\mathbb{A}'}(\boldsymbol{x}_{\mathbb{A}'}) \tag{14}$$

where the notation $\begin{pmatrix} \boldsymbol{x}_{\mathbb{A}'} \\ \boldsymbol{x}_{\mathbb{A}''} \end{pmatrix}$ represents the vector in $D_{\mathbb{A}} \simeq D_{\mathbb{A}'} \times D_{\mathbb{A}''}$ that concatenates $\boldsymbol{x}_{\mathbb{A}'}$ and $\boldsymbol{x}_{\mathbb{A}''}$.

Theorem 2 means that if a set of attackers can gain a certain amount of information on a set of targets, they cannot increase that amount of information gain if some attackers leave the group.

In other words, a set of attackers $\mathbb{A}$ that contains a smaller set $\mathbb{A}'$ can always manage to learn at least as much information as the latter.

Proof. Let $\boldsymbol{x}_{\mathbb{A}'} \in D_{\mathbb{A}'}$ be a vector of attackers' input. Let us show that there exists a $\boldsymbol{x}_{\mathbb{A}''} \in D_{\mathbb{A}''}$ that satisfies $\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}} \begin{pmatrix} \boldsymbol{x}_{\mathbb{A}'} \\ \boldsymbol{x}_{\mathbb{A}''} \end{pmatrix} \leq \mathrm{awae}_{\mathbb{T}}^{\mathbb{A}'}(\boldsymbol{x}_{\mathbb{A}'})$. The weighted average entropy for attackers $\mathbb{A}'$ and target set $\mathbb{T}$ can be written by definition as:

$$\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}'}(\boldsymbol{x}_{\mathbb{A}'}) = \sum_{O} p(O \mid \boldsymbol{x}_{\mathbb{A}'}) \cdot \left[\sum_{X_{\mathbb{T}}} g(p(X_{\mathbb{T}} \mid O, \boldsymbol{x}_{\mathbb{A}'})) \right]$$

where we recall the definition of $g(x) = -x \log x$.

The law of total probability with conditional probabilities gives us:

$$p(X_{\mathbb{T}} \mid O, \boldsymbol{x}_{\mathbb{A}'}) = \sum_{X_{\mathbb{A}''}} p(X_{\mathbb{A}''} \mid O, \boldsymbol{x}_{\mathbb{A}'}) \cdot p(X_{\mathbb{T}} \mid O, \begin{pmatrix} \boldsymbol{x}_{\mathbb{A}'} \\ X_{\mathbb{A}''} \end{pmatrix})$$

As $\sum_{X_{\mathbb{A}''}} p(X_{\mathbb{A}''} \mid O, \boldsymbol{x}_{\mathbb{A}'}) = 1$ and g is concave, we have:

$$g(\sum_{X_{\mathbb{A}''}} p(X_{\mathbb{A}''} \mid O, \boldsymbol{x}_{\mathbb{A}'}) \cdot p(X_{\mathbb{T}} \mid O, \begin{pmatrix} \boldsymbol{x}_{\mathbb{A}'} \\ X_{\mathbb{A}''} \end{pmatrix})) \geq \sum_{X_{\mathbb{A}''}} p(X_{\mathbb{A}''} \mid O, \boldsymbol{x}_{\mathbb{A}'}) \cdot g(p(X_{\mathbb{T}} \mid O, \begin{pmatrix} \boldsymbol{x}_{\mathbb{A}'} \\ X_{\mathbb{A}''} \end{pmatrix}))$$

and thus:

$$\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}'}(\boldsymbol{x}_{\mathbb{A}'}) \geq \sum_{O} p(O \mid \boldsymbol{x}_{\mathbb{A}'}) \cdot \left[\sum_{X_{\mathbb{T}}} \sum_{X_{\mathbb{A}''}} p(X_{\mathbb{A}''} \mid O, \boldsymbol{x}_{\mathbb{A}'}) \cdot g(p(X_{\mathbb{T}} \mid O, \begin{pmatrix} \boldsymbol{x}_{\mathbb{A}'} \\ X_{\mathbb{A}''} \end{pmatrix})) \right] \tag{15}$$

However, by virtue of Bayes' theorem with conditional probabilities, we have:

$$p(O \mid \boldsymbol{x}_{\mathbb{A}'}) \cdot p(X_{\mathbb{A}''} \mid O, \boldsymbol{x}_{\mathbb{A}'}) = p(X_{\mathbb{A}''} \mid \boldsymbol{x}_{\mathbb{A}'}) \cdot p(O \mid \begin{pmatrix} \boldsymbol{x}_{\mathbb{A}'} \\ X_{\mathbb{A}''} \end{pmatrix})$$

So Eq. (15) becomes:

$$\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}'}(\boldsymbol{x}_{\mathbb{A}'}) \geq \sum_{O}\sum_{X_{\mathbb{T}}}\sum_{X_{\mathbb{A}''}} p(X_{\mathbb{A}''} \mid \boldsymbol{x}_{\mathbb{A}'}) \cdot p(O \mid \begin{pmatrix}\boldsymbol{x}_{\mathbb{A}'}\\ X_{\mathbb{A}''}\end{pmatrix}) \cdot g(p(X_{\mathbb{T}} \mid O, \begin{pmatrix}\boldsymbol{x}_{\mathbb{A}'}\\ X_{\mathbb{A}''}\end{pmatrix}))$$

and by rearranging the sums, we have:

$$\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}'}(\boldsymbol{x}_{\mathbb{A}'}) \geq \sum_{X_{\mathbb{A}''}} p(X_{\mathbb{A}''} \mid \boldsymbol{x}_{\mathbb{A}'})\cdot \left[\sum_{O} p(O \mid \begin{pmatrix}\boldsymbol{x}_{\mathbb{A}'}\\ X_{\mathbb{A}''}\end{pmatrix}) \cdot \sum_{X_{\mathbb{T}}} g(p(X_{\mathbb{T}} \mid O, \begin{pmatrix}\boldsymbol{x}_{\mathbb{A}'}\\ X_{\mathbb{A}''}\end{pmatrix}))\right]$$

which is equivalent to:

$$\begin{aligned}\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}'}(\boldsymbol{x}_{\mathbb{A}'}) &\geq \sum_{X_{\mathbb{A}''}} p(X_{\mathbb{A}''} \mid \boldsymbol{x}_{\mathbb{A}'}) \cdot \mathrm{awae}_{\mathbb{T}}^{\mathbb{A}} \begin{pmatrix}\boldsymbol{x}_{\mathbb{A}'}\\ X_{\mathbb{A}''}\end{pmatrix}\\ &\geq \sum_{X_{\mathbb{A}''}} p(X_{\mathbb{A}''}) \cdot \mathrm{awae}_{\mathbb{T}}^{\mathbb{A}} \begin{pmatrix}\boldsymbol{x}_{\mathbb{A}'}\\ X_{\mathbb{A}''}\end{pmatrix}\end{aligned}$$

since $X_{\mathbb{A}'}$ (which belongs to the attackers) and $X_{\mathbb{A}''}$ (that is held by the spectators) are independent.

This last equation means that the average of $\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}} \begin{pmatrix}\boldsymbol{x}_{\mathbb{A}'}\\ X_{\mathbb{A}''}\end{pmatrix}$ over the values of $X_{\mathbb{A}''}$ is smaller than $\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}'}(\boldsymbol{x}_{\mathbb{A}'})$. In particular, we can choose an appropriate vector $\boldsymbol{x}_{\mathbb{A}''} = \arg\min_{X_{\mathbb{A}''}} \mathrm{awae}_{\mathbb{T}}^{\mathbb{A}} \begin{pmatrix}\boldsymbol{x}_{\mathbb{A}'}\\ X_{\mathbb{A}''}\end{pmatrix}$ that will realise the desired property $\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}} \begin{pmatrix}\boldsymbol{x}_{\mathbb{A}'}\\ \boldsymbol{x}_{\mathbb{A}''}\end{pmatrix} \leq \mathrm{awae}_{\mathbb{T}}^{\mathbb{A}'}(\boldsymbol{x}_{\mathbb{A}'})$. ■

Note 3. If we consider an empty set of attackers $\mathbb{A}'$, Theorem 2 states that a set of attackers $\mathbb{A}$ that attack targets $\mathbb{T}$ always have a combination of inputs $\boldsymbol{x}_{\mathbb{A}}$ that can optimise their goal. Not only does this imply that input substitution is effective, but it also demonstrates that this measure of information flow adapts easily to any possible target set $\mathbb{T}$, whereas a semantic approach would be less scalable. Let us now illustrate these theorems with a simple example.

Example 3. Let us consider 5 parties A, B, C, D and E holding an input represented as the corresponding lower case letter. We assume that each of these inputs range over the domain $D = [\![1, 5]\!]$, and that the beliefs of all the parties on the other inputs are uniform over this domain. Let us consider the function f defined by $f(a, b, c, d, e) = ae + (b-2)(b-3)(c+d) + 2(b-2)c + 3d$. We study the values of $\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}}$ that a set of attackers $\mathbb{A}$ learn on their targets $\mathbb{T}$ and compare different situations. We first study the case where $\mathbb{A} = \{A, B\}$ wish to attack $\mathbb{T} = \{C, D\}$ and compare it to the case where $\mathbb{A}$ attack a smaller set of targets $\mathbb{T}' = \{C\}$. We then compare the first case to the situation where a restricted set of attackers $\mathbb{A}' = \{A\}$ attacks $\mathbb{T}$.

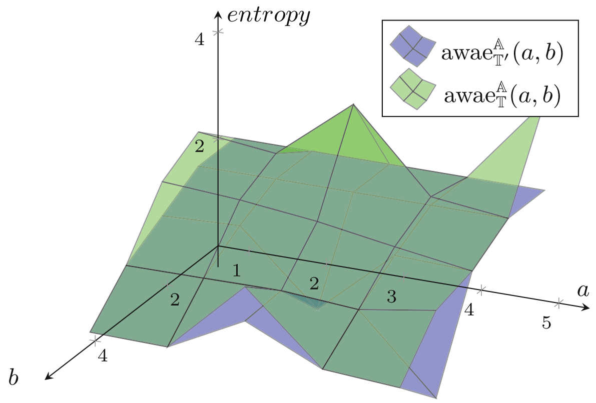

Fig. 4. Comparison of information flow on $\mathbb{T}$ and a smaller $\mathbb{T}'$.

Case 1: Let us compare and draw in Fig. 4 the values of $\text{awae}^{\mathbb{A}}_{\mathbb{T}}\begin{pmatrix}a\\b\end{pmatrix}$ and $\text{awae}^{\mathbb{A}}_{\mathbb{T}'}\begin{pmatrix}a\\b\end{pmatrix}$ for all values of attackers' input $\begin{pmatrix}a\\b\end{pmatrix}$. We notice that for all input values $\begin{pmatrix}a\\b\end{pmatrix}$, we have $\text{awae}^{\mathbb{A}}_{\mathbb{T}'}\begin{pmatrix}a\\b\end{pmatrix} \leq \text{awae}^{\mathbb{A}}_{\mathbb{T}}\begin{pmatrix}a\\b\end{pmatrix}$ as claimed in Theorem 1.

We can also notice the particular point $\text{awae}^{\mathbb{A}}_{\mathbb{T}'}(5,4) = 0$ which means that for the input $(a,b) = (5,4)$, the attackers $\mathbb{A}$ will learn the exact value of their target input c. Indeed, in this case, the attackers know that the output value can be written $o = 6c + 5(e+d)$. For inputs ranged in $D = [\![1,5]\!]$, we can prove that the value of the target input c is then determined by the value of the output.

Case 2: We now want to observe the influence that a set of attackers can gain when they collude. We draw in Fig. 5 the values of $\text{awae}^{\mathbb{A}'}_{\mathbb{T}}(a)$ and $\text{awae}^{\mathbb{A}}_{\mathbb{T}}\begin{pmatrix}a\\b\end{pmatrix}$ for all values of a and b. We can notice that for all a, there exists a b such that $\text{awae}^{\mathbb{A}}_{\mathbb{T}}\begin{pmatrix}a\\b\end{pmatrix} \leq \text{awae}^{\mathbb{A}'}_{\mathbb{T}}(a)$, as claimed in Theorem 2.

However, we can also notice that the attackers have to choose their inputs cautiously. For example, $\text{awae}^{\mathbb{A}}_{\mathbb{T}}(5,1) \simeq 3.97$ whereas $\text{awae}^{\mathbb{A}'}_{\mathbb{T}}(5) \simeq 2.92$, which means that even though the attackers $\mathbb{A}$ know the values of more inputs than $\mathbb{A}'$, some combinations of inputs might hinder their information retrieval.

Moreover, we can add that the values of $\text{awae}^{\mathbb{A}}_{\mathbb{T}}$ for the different values of $\boldsymbol{x}_{\mathbb{A}} \in D_{\mathbb{A}}$ can guide the attackers towards a choice of informative inputs with respect to the set of targets that they wish to attack. Conversely, the targets can take advantage of the values provided by $\text{twae}^{\mathbb{A}}_{\mathbb{T}}$ in order to measure the risk that they would run if they entered a particular input $\boldsymbol{x}_{\mathbb{T}} \in D_{\mathbb{T}}$ given a potential set of attackers $\mathbb{A}$. However, in our model, targets are deemed to be honest parties who shall neither collude nor share any information on their inputs. This kind

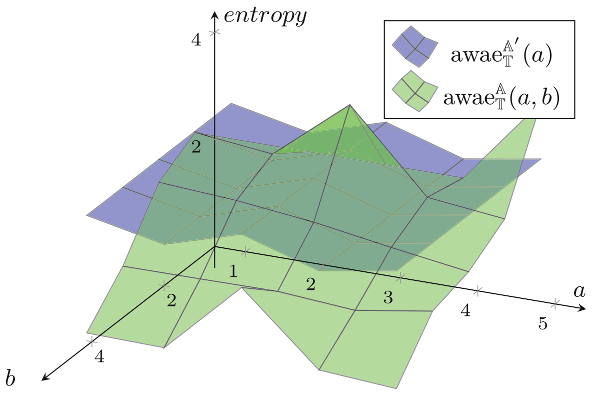

Fig. 5. Influence that $\mathbb{A}$ has compared to a smaller $\mathbb{A}'$.

of inference would thus only be drawn by single targets $\mathbb{T} = \{P_t\}$. Furthermore, to the extent that the output of the computation does not matter, the targets would also have an incentive to substitute their inputs in order to protect their privacy. However, as the functions $\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}}$ and $\mathrm{twae}_{\mathbb{T}}^{\mathbb{A}}$ can be calculated by every party, we could further imagine that the attackers and the targets would not choose the inputs that would directly minimise (or respectively maximise) the entropy of $X_{\mathbb{T}}$ after computation, but would have a strategy over their possible inputs that would be the best response to its opponent's expected choice. In the following section, we propose a game-theoretic extension of this SMC context where the payoff of each group of parties is given by the posterior entropy of $X_{\mathbb{T}}$.

6 Game Theoretic MPC

In this section, we define a two-player game based on the usual context of SMC that could model the strategies that the participants of the protocol would follow in order to control the information flows that may arise from this computation.

We consider the same formal setting as for an SMC protocol and we identify the two players of the game as the set of attackers $\mathbb{A}$ and the set of targets $\mathbb{T}$. In order to play a game, the players have to choose an input $\boldsymbol{x}_{\mathbb{A}} \in D_{\mathbb{A}}$ and $\boldsymbol{x}_{\mathbb{T}} \in D_{\mathbb{T}}$ respectively. A third set of inputs $\boldsymbol{x}_{\mathbb{S}}$ will be picked at random in $D_{\mathbb{S}}$ by a third party representing the spectators, and those three inputs will be used to feed the secure computation of $f(x_1, \cdots, x_n) = o$ whose result o will be publicly revealed. Once the result is broadcast, the participants in $\mathbb{A}$, referred to as 'player' $\mathbb{A}$ subsequently, will learn a certain amount of information on the input $\boldsymbol{x}_{\mathbb{T}}$ of player $\mathbb{T}$, that can be measured by the entropy of $\pi_{\mathbb{T},o}^{\mathbb{A},\boldsymbol{x}_{\mathbb{A}}}$, the posterior distribution of $X_{\mathbb{T}}$, as calculated in the previous section. The aim for player $\mathbb{A}$ is to maximise this information that he gains on $X_{\mathbb{T}}$ whereas the aim for 'player' $\mathbb{T}$ is to minimise this information flow.

$$\begin{array}{cc} & \boldsymbol{x}_{\mathbb{A}} \\ & \cdots \\ \boldsymbol{x}_{\mathbb{T}} & \left(\cdots \quad \mathrm{jwae}_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}, \boldsymbol{x}_{\mathbb{T}}) \quad \right) \end{array}$$

Fig. 6. Payoff matrix for row player $\mathbb{T}$. Columns indicate attackers' strategies.

$$\begin{pmatrix} 3/4 & 3/4 & 1/2 & 1/2 \\ 1/2 & 3/4 & 1/4 & 1/2 \\ 3/4 & 1/2 & 1/4 & 1/2 \\ 1/2 & 1 & 0 & 0 \end{pmatrix}$$

Fig. 7. Row player $\mathbb{T}$'s payoff matrix from Example 4.

The payoff matrix for player $\mathbb{T}$ representing this two-player zero sum game can be expressed as $(\mathrm{jwae}_{\mathbb{T}}^{\mathbb{A}}(\boldsymbol{x}_{\mathbb{A}}, \boldsymbol{x}_{\mathbb{T}}))_{\substack{\boldsymbol{x}_{\mathbb{A}} \in D_{\mathbb{A}} \\ \boldsymbol{x}_{\mathbb{T}} \in D_{\mathbb{T}}}}$ and can be written as in Fig. 6.

We can notice that this matrix is indeed the payoff matrix for row player $\mathbb{T}$ in that the higher the value of $\mathrm{jwae}_{\mathbb{T}}^{\mathbb{A}}$, the better off player $\mathbb{T}$ is. Let us illustrate such a game in a simple situation.

Example 4. Let us consider 3 sets of parties $\mathbb{A} = \{X\}$, $\mathbb{T} = \{Y\}$ and $\mathbb{S} = \{Z\}$ controlling the respective inputs x, y and z ranged in $[\![1, 4]\!]$. Let us consider a function f to be securely computed $f(x, y, z) = 2xy - xz - yz$. Based on Eq. (7) of Definition 2, we draw player $\mathbb{T}$'s payoff matrix in Fig. 7.

Strategy $y = 2$ for player $\mathbb{T}$ is dominated by strategy $y = 1$. Indeed, for all x in $[\![1, 4]\!]$ we have $\mathrm{jwae}_{\mathbb{T}}^{\mathbb{A}}(x, 2) \leq \mathrm{jwae}_{\mathbb{T}}^{\mathbb{A}}(x, 1)$. Thus, player Y has no incentive to play strategy $y = 2$ and will never play it. Similarly, strategy $x = 4$ is dominated by strategy $x = 3$ for the player X since for all y in $[\![1, 4]\!]$ we have $\mathrm{jwae}_{\mathbb{T}}^{\mathbb{A}}(3, y) \leq \mathrm{jwae}_{\mathbb{T}}^{\mathbb{A}}(4, y)$, and thus the attacker will never play strategy $x = 4$.

We can calculate the average weighted entropy for player X and player Y:

n	$\mathrm{twae}_{\mathbb{T}}^{\mathbb{A}}(n)$	$\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}}(n)$
1	0.625	0.625
2	0.5	0.75
3	0.5	0.25
4	0.375	0.375

Based on the values of player Y's weighted average entropy, we could think that player Y has an incentive to play $y = 1$ in order to maximise his posterior entropy. Conversely, based on the values of $\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}}$, we would say that attacker X has an incentive to choose input $x = 3$ in order to minimise the expected entropy of Y after computation. But if we look at the payoff matrix for target Y displayed in Eq. (7), we can notice that these strategies $(x = 1, y = 3)$ form a Nash equilibrium in pure strategies [2]. This means that in this situation, not only can the attacker learn as much information as possible on the target, but this is also the best strategy he could play given player Y's strategy.

This formulation can help an attacker in a normal SMC context to rule out some dominated strategies. Indeed, a deceitful attacker would never have an incentive to choose an input that would always produce a higher entropy for $X_{\mathbb{T}}$.

Similarly, the payoff matrix could also emphasise some inputs $\boldsymbol{x}_{\mathbb{T}}$ that would be compromising for player $\mathbb{T}$. Again, in a pure SMC context, a set of targets $\mathbb{T}$ could precompute the risk that their input could be guessed and evaluate the entropy that the observed output would leak about their input $\boldsymbol{x}_{\mathbb{T}}$. They could then accept or refuse to take part in an SMC protocol.

However, the targets would only be able to assess an estimate of the risk that they run on average by supplying a certain input. The exact entropy that a set of attackers would gain on $\boldsymbol{x}_{\mathbb{T}}$ can only be calculated once the output is revealed, that is to say once $\boldsymbol{x}_{\mathbb{A}}$, $\boldsymbol{x}_{\mathbb{T}}$ but also $\boldsymbol{x}_{\mathbb{S}}$ have been submitted to the protocol. We could thus ideally imagine an SMC protocol that not only realises perfect security with robustness against malicious adversaries, but which would also be robust against deceitful adversaries. Such a protocol would only allow those computations that guarantee that any information flow does not exceed a certain threshold. Concretely, we could imagine an internal mechanism that would lead the protocol to fail if the information flow associated with the given combination of inputs is too high. An intuitive way of evaluating the information flow *IFlow* associated to a given set of inputs would be to consider *IFlow* as a function of the inputs, and to calculate it via an SMC protocol. We would then have to consider the information flow that leaks from this new computation and possibly iterate this process to attain some form of limit or fixed point.

Indeed, a failure of such an algorithm would of course prevent a risky output to be calculated in a hazardous situation, but it would still reveal some information about the combination of the three inputs $\boldsymbol{x}_{\mathbb{A}}$, $\boldsymbol{x}_{\mathbb{T}}$ and $\boldsymbol{x}_{\mathbb{S}}$ [22]. In particular, the attackers would learn that they are in a situation that makes the entropy of $X_{\mathbb{T}}$ lower than a given threshold. In order to prevent this kind of inference, we could imagine a probabilistic algorithm that would fail with a certain probability. In this case, if the algorithm terminates and returns an output, the latter is guaranteed to preserve the privacy of the targets' input up to a certain threshold. On the other hand, if the protocol fails, it would not be obvious for the attackers whether it is a probabilistic failure or one that gives some information about $\boldsymbol{x}_{\mathbb{T}}$. In such a probabilistic protocol, the privacy of $\boldsymbol{x}_{\mathbb{T}}$ would be enhanced but the chances for the output to be calculated would decrease.

7 Discussion and Related Work

Quantitative Information Flow: In our paper, we used the notion of Shannon entropy in order to measure the amount of information that an attacker learns on some private inputs. However, this choice is debatable and it would be worth studying the application of other entropy measures to our scenario. Indeed, different measures of entropy are more appropriate for assessing different security concerns [6,35]. For example, Shannon entropy is unable to estimate or to give an upper bound on the expected number of guesses [21,23], also known as the guessing entropy or guesswork. Moreover, other security concerns cannot be addressed by Shannon entropy, such as the probability of a secret to be guessed in one try, also known as Bayes vulnerability [33]. Instead, the min-entropy, another instance of Rényi entropy, directly reflects this threat. Also, information flow analysis can be shaped in order to measure a certain security requirement thanks to the more general notion of g-leakage [25] where the desired security property in question can be specified using a gain function. Our approach is parametric in the choice of such an information-theoretic measure and so could, in principle, support such alternative measures. This is subject to future work.

Input Substitution: We introduced in this paper the notion of deceitful adversaries who take advantage of information flow analysis in order to manipulate the input that they provide to an SMC in such a way that they learn more information on some targeted inputs. This deceitful behaviour is questionable and might not be realistic in some applications of SMC. For example, in the case of e-voting [26], the consequences of falsifying one's input could be dramatic, and therefore the notion of deceitful adversaries would not apply in this context.

However, SMC can also be used in other domains where the exactitude of a single party's input is less decisive than in e-voting. In particular, in data mining [19], SMC can be applied to compute a statistical function averaged over a population and whose output would not be affected by a slightly noisy input. A deceitful adversary who cares about learning the correct output of the function would still have an incentive to substitute his input, in that he will still get a reasonable approximation of the output. Moreover, we can think of cases where a negligible perturbation in the attackers' inputs could trigger a significant information flow. We could thus imagine that a deceitful adversary could benefit from a larger information gain without affecting too much the integrity of the result of the SMC. Example 1 features a situation where a perturbation of size 1 in the attacker's input suffices to produce a high information leak.

We have seen that the correctness of the output should also be taken into account by a deceitful adversary, reflecting what is at stake in a computation. An adversary may also want to explore a trade-off between the information that he gets from the output and the relevance of the output itself. SMC has been used for example to implement an auction between Danish farmers [5] while protecting the confidentiality of their bids and therefore of their individual economic position. In such a situation, we could imagine using or extending the notion of g-leakage in order to reflect the interest of a farmer to slightly deviate from his legitimate bid with the aim of maximising his benefits while protecting his personal details, or attacking someone else's economic position.

Information Flow in Programs: Information flow analysis in imperative programs is a field that has been explored with many different approaches. One of the fundamental concepts of this domain of study is the consideration of security classes introduced by Denning [12], which enables us to classify the variables of a program with respect to their level of privacy in order to form a lattice of information. Based on this classification, type systems [36] and semantic approaches [16] have been implemented in order to define the security of instructions involving such variables. The most basic model considers only two security classes L and H separating the variables with a low and high level of privacy respectively [12]. The security of a program is then expressed with the notion of non-interference between both classes [13,32,36]. However, as programs in practice may contain some interference, other quantitative approaches [8,9,20,28,34,37] have been proposed in order to measure the information flow that can arise between variables from different security classes. The computation of such quantitative information flows also includes the use of probabilistic instructions [16,24,32] that can randomise the algorithms and make programs non-deterministic and thus in some cases protect the privacy of variables in H.

In our approach, we measure the information flow that can leak from the observation of the output of an SMC computation and this echoes some of those works on information flow analysis in programs. In comparison, we could identify the target's inputs of our setting to the higher security level class H whereas both the attackers' inputs $\boldsymbol{x}_{\mathbb{A}}$ and the output o would be added to the lower security class L. The non-interference property would then be satisfied if the posterior entropy of $X_{\mathbb{T}}$ is equal to its prior entropy. We can also notice that the role of the spectators in our work could be compared to those probabilistic instructions in sequential programs. Indeed, the attackers only have a prior belief on the spectators' inputs, which hides the targets' inputs and spreads their posterior probability distribution once the output is revealed. Finally, our work differs from the concepts of program analysis in that we do not measure the information that leaks from every intermediate instruction or every step of a loop in an imperative program. We only study the information flow that leaks from the output of a public function. We may consider the latter as a functional program whose computation is known to be secure, and is realised in practice by an SMC protocol.

8 Conclusions

The notion of security of a protocol in secure multiparty computation ensures that for all function f that needs to be computed, no information about the inputs will leak, except from that which is leaked by the observation of the public output. With respect to this notion of security, those protocols can securely compute any function f that is composed of the supported operations of addition and multiplication. This definition of security is thus independent of the function that is being calculated. However, in practice, the participants of the secure computation of f would not only like to make sure that no information leaks from the protocol itself, but they would also like to know and minimise the risks of information flow from public outputs of function f – regardless of the protocol used to implement that computation. In this work, we analysed such risks.

We introduced the notion of deceitful adversary as well as a model of attackers and targets that fits the setting of SMC. We modelled the agents' beliefs by probability distributions and we used Shannon entropy to measure the unpredictability of the targets' inputs under different circumstances. Based on this modelling choice, we defined some measures $\mathrm{awae}_{\mathbb{T}}^{\mathbb{A}}$ and $\mathrm{twae}_{\mathbb{T}}^{\mathbb{A}}$ that can estimate the information flow that would arise for a given pair of attackers' and targets' inputs. We further explored the sensitivity of $\text{awae}_{\mathbb{T}}^{\mathbb{A}}$ with respect to the set of attackers $\mathbb{A}$ and targets $\mathbb{T}$. In particular we showed that several deceitful adversaries would generally have an incentive to collude and substitute their input appropriately in order to optimise their gain.

We experimentally tested those measures on sample functions and demonstrated that even for simple arithmetic functions, the values of $\text{awae}_{\mathbb{T}}^{\mathbb{A}}$ and $\text{twae}_{\mathbb{T}}^{\mathbb{A}}$ helps us to exhibit some non-trivial behaviours of the information flow that we could not have predicted with logical inference or a semantic approach.

Finally, we showed through simple examples that those measures can guide the attackers and targets through a choice of strategic, risk-aware inputs. This naturally led us to consider a more general game-theoretic setting that could model the risk management of participants of a secure multiparty computation.

It would be of interest to be able to develop similar results as Theorems 1 and 2 for the measure of $\text{twae}_{\mathbb{T}}^{\mathbb{A}}$. This would enable us to better understand how a target set could protect itself or better estimate the risks it faces. But, our approach assumes that the beliefs of the participants on their opponents are public. We would like to extend this simple setting to a multi-agent system where those beliefs would be private to each group of participants, and could be updated by the probabilistic analysis provided by those measures.

The notion of secure information flow of a function we studied can harden the security of SMC protocols and it may be beneficial to include such probabilistic analyses of information flow, based on the measures we defined, into existing SMC protocols. Such modified protocols would ideally detect if a sensitive combination of inputs is being examined, and would prevent such computations to return. Those combined protocols would not only preserve the privacy of the inputs inside the protocol, but would also contain inevitable information flow within a reasonable range reflecting risk appetite or risk budget.

Acknowledgements. This work was supported by the UK EPSRC with Fees Award and grants EP/N023242/1 and EP/N020030/1. We thank Geoffrey Smith and anonymous reviewers for their constructive comments and suggestions.

References

1. Asharov, G., Lindell, Y.: A full proof of the BGW protocol for perfectly secure multiparty computation. Cryptology ePrint Archive, Report 2011/136 (2011)
2. Avis, D., Rosenberg, G.D., Savani, R., Von Stengel, B.: Enumeration of Nash equilibria for two-player games. Econ. Theor. **42**(1), 9–37 (2010)
3. Baum, C., Damgård, I., Toft, T., Zakarias, R.: Better preprocessing for secure multiparty computation. In: Manulis, M., Sadeghi, A.-R., Schneider, S. (eds.) ACNS 2016. LNCS, vol. 9696, pp. 327–345. Springer, Cham (2016). doi:10.1007/978-3-319-39555-5_18
4. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: Proceedings of STOC, pp. 1–10. ACM (1988)

5. Bogetoft, P., et al.: Secure multiparty computation goes live. In: Dingledine, R., Golle, P. (eds.) Financial Cryptography and Data Security. LNCS, vol. 5628. Springer, Heidelberg (2009)
6. Cachin, C.: Entropy measures and unconditional security in cryptography. Ph. D thesis, Diss. Techn. Wiss. ETH Zürich, Nr. 12187 (1997). Ref.: Maurer, U., Korref, Massey, J.L. (1997)
7. Chaum, D., Crépeau, C., Damgard, I.: Multiparty unconditionally secure protocols. In: Proceedings of STOC, pp. 11–19. ACM (1988)
8. Clark, D., Hunt, S., Malacaria, P.: A static analysis for quantifying information flow in a simple imperative language. J. Comput. Secur. **15**(3), 321–371 (2007)
9. Michael, M.R., Myers, A.C., Schneider, F.B.: Quantifying information flow with beliefs. J. Comput. Secur. **17**(5), 655–701 (2009)
10. Cramer, R., Damgård, I., Nielsen, J.B.: Secure Multiparty Computation and Secret Sharing. Cambridge University Press, Cambridge (2015)
11. Damgård, I., Pastro, V., Smart, N., Zakarias, S.: Multiparty computation from somewhat homomorphic encryption. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 643–662. Springer, Heidelberg (2012). doi:10.1007/978-3-642-32009-5_38
12. Dorothy, D.E.: A lattice model of secure information flow. Commun. ACM **19**(5), 236–243 (1976)
13. Dima, C., Enea, C., Gramatovici, R.: Nondeterministic noninterference and deducible information flow. Technical report, Citeseer (2006)
14. Wenliang, D., Atallah, M.J.: Secure multi-party computation problems, their applications: a review and open problems. In: Proceedings of the Workshop on New Security Paradigms, pp. 13–22. ACM (2001)
15. Goldreich, O., Micali, S., Wigderson, A.: How to play ANY mental game. In: Proceedings of STOC 1987, pp. 218–229. ACM (1987)
16. Joshi, R., Leino, K.R.M.: A semantic approach to secure information flow. Sci. Comput. Program. **37**(1), 113–138 (2000)
17. Kolesnikov, V., Schneider, T.: Improved garbled circuit: free XOR gates and applications. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008. LNCS, vol. 5126, pp. 486–498. Springer, Heidelberg (2008). doi:10.1007/978-3-540-70583-3_40
18. Lindell, Y., Pinkas, B.: A proof of security of yao's protocol for two-party computation. J. Cryptology **22**(2), 161–188 (2009)
19. Lindell, Y., Pinkas, B.: Secure multiparty computation for privacy-preserving data mining. J. Priv. Confidentiality **1**(1), 5 (2009)
20. Malacaria, P.: Algebraic foundations for quantitative information flow. Math. Struct. Comput. Sci. **25**(02), 404–428 (2015)
21. Malone, D., Sullivan, W.: Guesswork is not a substitute for entropy. Slides (2005)
22. Mardziel, P., Magill, S., Hicks, M., Srivatsa, M.: Dynamic enforcement of knowledge-based security policies. In: IEEE 24th Computer Security Foundations Symposium, pp. 114–128. IEEE (2011)
23. Massey, J.L.: Guessing and entropy. In: Proceedings of the IEEE International Symposium on Information Theory, p. 204. IEEE (1994)
24. McIver, A., Morgan, C.: A probabilistic approach to information hiding. Programming Methodology. Monographs in Computer Science, pp. 441–460. Springer, Heidelberg (2003)
25. Alvim, M.S., Chatzikokolakis, K., Palamidessi, C., Smith, G.: Measuring information leakage using generalized gain functions. In: IEEE 25th Computer Security Foundations Symposium, pp. 265–279. IEEE (2012)

26. Nair, D.G., Binu, V.P., Kumar, G.S.: An improved e-voting scheme using secret sharing based secure multi-party computation. arXiv preprint arXiv:1502.07469 (2015)
27. Nielsen, J.B., Nordholt, P.S., Orlandi, C., Burra, S.S.: A new approach to practical active-secure two-party computation. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 681–700. Springer, Heidelberg (2012). doi:10.1007/978-3-642-32009-5_40
28. Phan, Q-S., Malacaria, P., Păsăreanu, C.S., d'Amorim, M.: Quantifying information leaks using reliability analysis. In: Proceedings of the International SPIN Symposium on Model Checking of Software, pp. 105–108. ACM (2014)
29. Shamir, A.: How to share a secret. CACM **22**(11), 612–613 (1979)
30. Shannon, C.E., Weaver, W.: The Mathematical Theory of Communication. University of Illinois Press, Urbana (1949)
31. Smart, N.P.: Cryptography Made Simple. Springer, Heidelberg (2016)
32. Smith, G.: Principles of secure information flow analysis. In: Christodorescu, M., Jha, S., Maughan, D., Song, D., Wang, C. (eds.) Malware Detection. Advances in Information Security, vol. 27, pp. 291–307. Springer, Heidelberg (2007)
33. Smith, G.: On the foundations of quantitative information flow. In: Alfaro, L. (ed.) FoSSaCS 2009. LNCS, vol. 5504, pp. 288–302. Springer, Heidelberg (2009). doi:10.1007/978-3-642-00596-1_21
34. Smith, G.: Quantifying information flow using min-entropy. In: Eighth International Conference on Quantitative Evaluation of Systems, pp. 159–167. IEEE (2011)
35. Smith, G.: Recent developments in quantitative information flow (invited tutorial). In: Proceedings of the 30th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), pp. 23–31. IEEE Computer Society (2015)
36. Volpano, D., Irvine, C., Smith, G.: A sound type system for secure flow analysis. J. Comput. Secur. **4**(2–3), 167–187 (1996)
37. Yasuoka, H., Terauchi, T.: Quantitative information flow as safety and liveness hyperproperties. Theor. Comput. Sci. **538**, 167–182 (2014)