# Incentive-driven attacker for corrupting two-party protocols

Soft Computing (2018) 22:7733–7740
https://doi.org/10.1007/s00500-018-3342-3

FOCUS

# Incentive-driven attacker for corrupting two-party protocols

Yilei Wang[1] · Roberto Metere[2] · Huiyu Zhou[3] · Guanghai Cui[1] · Tao Li[1]

Published online: 22 June 2018


**Abstract**
Adversaries in two-party computation may sabotage a protocol, leading to possible collapse of the information security management. In practice, attackers often breach security protocols with specific incentives. For example, attackers manage to reap additional rewards by sabotaging computing tasks between two clouds. Unfortunately, most of the existing research works neglect this aspect when discussing the security of protocols. Furthermore, the construction of corrupting two parties is also missing in two-party computation. In this paper, we propose an incentive-driven attacking model where the attacker leverages corruption costs, benefits and possible consequences. We here formalize the utilities used for two-party protocols and the attacker(s), taking into account both corruption costs and attack benefits. Our proposed model can be considered as the extension of the seminal work presented by Groce and Katz (Annual international conference on the theory and applications of cryptographic techniques, Springer, Berlin, pp 81–98, 2012), while making significant contribution in addressing the corruption of two parties in two-party protocols. To the best of our knowledge, this is the first time to model the corruption of both parties in two-party protocols.

**Keywords** Cost corruption · Incentive-driven adversary · Two-party computation

## 1 Introduction

Two-party protocols allow two distributed and distrustful parties to jointly consider a general function with their private inputs. Two-party protocols can be widely used in various fields such as cloud computing (Gao et al. 2018; Tian et al. 2018; Ibtihal et al. 2017; Zheng and Wang 2018), file encryption (Yang et al. 2018), verifiable computation (Chen et al. 2016), keyword searching (Li et al. 2015) and trust evaluation (Jiang et al. 2016, 2018; Jhaveri et al. 2018). Normally, the security of two-party protocols is discussed with a single external attacker, who is assumed to arbitrarily sabotage the protocols by corrupting at most one party. In most cases, however, a realistic attacker often sabotages the security protocols with certain incentives instead of arbitrary purposes (Haddi and Benchaïba 2015; Zhao et al. 2012; Wu et al. 2014). If the attacker can justify his (or her) attacks, he (or she) will be motivated to launch the corruption such that the protocols are changed toward the way to benefit him (or her). On the other hand, corruption is not free for any involved party. Otherwise, any party will modify the protocols without any liability (Li et al. 2018; Liu et al. 2018b; Gupta et al. 2016). The settings of corrupting two parties are often missing, and it is less important to discuss the security of a protocol with two corrupted parties in two-party protocols. However, it has a great potential to consider the case of corrupting both parties for an incentive-driven attacker, who has to leverage corruption costs and possible benefits. It is worth pointing out that semi-honest and malicious adversaries may never reveal authentic incentives (Li et al. 2017; Yu et al. 2018).

Communicated by B. B. Gupta.

✉ Yilei Wang
wang_yilei2000@126.com

Roberto Metere
r.metere2@ncl.ac.uk

Huiyu Zhou
hz143@le.ac.uk

Guanghai Cui
cuigh@ldu.edu.cn

Tao Li
litao_888@sina.com

[1] School of Information and Electrical Engineering, Ludong University, Yantai, China

[2] Department of Computing Science, Newcastle University, Newcastle, United Kingdom

[3] Department of Informatics, University of Leicester, Leicester, United Kingdom

In this paper, we revisit the problem of incentive-driven attackers and corruption costs by significantly extending the framework reported in Groce and Katz (2012). We propose the new incentive-driven attacking model based on the intuition that no party will participate in a protocol without any benefit. Existing works assume parties are arbitrarily malicious, which is not proper for reality attackers. Rational parties take part into protocols to maximize their payoffs. It is rarely considered to corrupt two parties at the same time. However, currently corrupting two parties is in line with reality. For example, one malicious attacker learns additional information with respect to a protocol and can maximize his profit when obtaining inputs of both parties. At the same time, two parties in the protocol do not learn the additional information and can only obtain their own specific payoff when following the protocol. The malicious attacker has incentives to bribe both parties with a price, which is higher than the party's specific payoff, but lower than the maximize profit. On the other hand, the corrupted parties would like to sell their inputs and receive the bribe money. Therefore, we formalize this malicious attacker into an incentive-driven attacker and discuss the conditions when the bribe is feasible. The basic idea of this work is more or less similar to social engineering (Hadnagy 2010; Abraham and Chengalur-Smith 2010). The main contributions of our work are summarized below.

1. We propose a new incentive-driven attacking model for costing the corruption of two-party protocols. Our proposed model describes the price of an attack when the attacker(s) sabotage the protocols. This cost computation influences the decision made for the launch of any attack.
2. We extensively discuss the required utilities under three corruption case studies. The significance of our work is the consideration of corrupting both parties, different from the other state-of-the-art techniques.
3. We instantiate the incentives of the attacker(s) by largely extending the framework reported in Groce and Katz (2012). We take into account the private inputs from the parties involved in the protocols, which may lead to different outcomes. We substantially investigate the preconditions for the attacker(s) to successfully sabotage the protocols' security.

### 1.1 Comparison against the state of the arts

Halpern and Teague (2004) introduced rational parties in secret sharing schemes and secure multi-party computation protocols, where rational parties have incentives to maximize their utilities. Asharov et al. (2011) presented formal definitions for the security of rational protocols with both positive and negative outputs on two-party protocols. Groce and Katz (2012) redefined utilities for rational parties. They proved that negative outcomes described in Asharov et al. (2011) can be avoided when rational parties were given proper incentives. However, their work has not considered the effects of costs on the corruption. Garay et al. introduced an external attacker and transferred a protocol to a two-party game (Garay et al. 2013). They also discussed the relationship between the corruption cost and the utilities when the attacker successfully broke privacy. However, they failed to present specific utility definitions with respect to the attacker's incentives. Furthermore, they only considered the corruption of partial parties instead of all the involved parties. Recently, bitcoin and blockchain (Meng et al. 2018; Lin et al. 2018; Liu et al. 2018a) are introduced as incentives in multi/two-party computation for correctness (Kumaresan and Bentov 2014) and fairness (Bentov and Kumaresan 2014; Andrychowicz et al. 2014). Andrychowicz et al. (2014) simulated fairness in two-party computation, where the party which does not learn the output may be awarded financial compensation. Unfortunately, their work does not cover the setting of corruption costs. Wang et al. (2018, 2016) propose rational secure two-party computation to describe the attacking model toward the view of game theory. Adat et al. (2018) propose an economic incentive-based risk transfer mechanism, which can prevent denial of service attack Adat et al. (2018) in Internet of Things. Their economic incentive-based agreement can provide additional security for Internet of Things if needed. Other attacking models consist of covert channels (Zhang et al. 2018) and correlated fading channels (Fan et al. 2017), which are out of the scope of this paper.

Our attacking model does not break the security of protocols per se. The attacker corrupts parties by stealing or bribing, which is the biggest difference from the work reported in Groce and Katz (2012). For the protocols, the attacker learns the output through cost estimation. On the other hand, honest or corrupted parties either obtain pecuniary compensation or certain beneficial outputs. We compare our work against the other and show the comparisons in Table 1.

### 1.2 Outlines

Section 2 presents the proposed environmental model and explains the used notions such as outcomes and expected utility. Section 3 shows the attacking model and defines the corresponding utility functions for different corruption cases. At the end of Sect. 3, an instance based on the protocol of Groce and Katz (2012) is presented to show the possibility for an attacker to corrupt one party or both parties. Section 4 concludes this paper and presents future work.

**Table 1** Comparations with other works

| | Garay et al. (2013) | Groce and Katz (2012) | Asharov et al. (2011) | Andrychowicz et al. (2014) | Ours |
|---|---|---|---|---|---|
| Corruption cost | ✓ | × | × | × | ✓ |
| Corrupt both parties | × | × | × | × | ✓ |
| Incentive-driven attacker | ✓ | × | × | ✓ | ✓ |
| Private types | × | × | ✓ | × | ✓ |

## 2 Environmental model

In this paper, we consider a general two-party computation protocol $\pi$. Two distinct parties, who execute $\pi$, belong to set $\mathcal{P} = \{A, B, C, D, \ldots\}$. Note that parties may be entities such as hosts or machines. Let $\pi_{P_1 P_2}$ denote an instance of $\pi$, where $P_1, P_2 \in \mathcal{P}$ execute $\pi$ with their inputs. Each party has his (or her) own identities such that they recognize each other in the specific protocol. Two parties can learn an output jointly with their inputs. Let $\mathcal{A}$ denote an attacker, either internal or external. The internal attacker is one of the parties inside the protocol and an external attacker outside the protocol. Note that the internal attacker belongs to $\mathcal{P}$, who attacks the other parties by providing proper incentives. Recall that only external attackers are discussed in traditional two-party computation. In this paper, we consider the possibility of the internal attackers for simplicity. We will discuss this setting in the following sections.

The main target of the attacker is to learn the output of the protocol by attacking two parties in the protocol. The attacker is assumed to have the following abilities: (1) he (or she) controls the communication channel (like eavesdrop) between parties $A$ and $B$; (2) the corruption is not free; (3) he (or she) may corrupt the parties to retrieve the identities (and the private inputs) to play protocol $\pi$. Here, corrupting one party has two styles. One is to steal inputs and identities with necessary costs without paying any cost for the corrupted parties. Another one is to bribe parties with bribery funds, and the corrupted parties reward the attacker afterward.

Let $p_A$ and $p_B$ denote the probabilities when the attacker bribes parties $A$ and $B$, respectively. The attacker may learn the output by replacing the corrupted parties in protocol $\pi$ with the stolen or rewarded inputs and identities from the corrupted parties. The difference between these two corruption styles is whether the corrupted parties are paid when they lose their inputs and identities. Both corruption styles are not free, and the costs are identical in both styles. Our model can apply to secure two-party computation protocols, where parties' identities are authenticated. The attacker may steal or bribe parties for their inputs and identities like session keys.

Suppose two parties $A$ and $B$ execute protocol $\pi$. In fact, any two parties belonging to $\mathcal{P}$ may execute the protocol. Let vectors $\mathbf{o}_A = (o_A^1, o_A^2, o_A^3, \ldots, o_A^n)$ and $\mathbf{o}_B = \{o_B^1, o_B^2, o_B^3, \ldots, o_B^n\}$ denote all the possible outcomes for $A$ and $B$, respectively, where $n$ denotes the number of the possible outcomes. Let vectors $\boldsymbol{\alpha} = \{\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_n\}$ and $\boldsymbol{\beta} = \{\beta_1, \beta_2, \beta_3, \ldots, \beta_n\}$ denote the distribution of the corresponding outcomes. Let vectors $\mathbf{u}_A = \{u_A^1, u_A^2, u_A^3, \ldots, u_A^n\}$ and $\mathbf{u}_B = \{u_B^1, u_B^2, u_B^3, \ldots, u_B^n\}$ denote the corresponding utilities for $A$ and $B$, respectively. Therefore, parties $A$ and $B$ have the expected utilities $U^A = \mathbf{u}_A \boldsymbol{\alpha}^T$, $U^B = \mathbf{u}_B \boldsymbol{\beta}^T$, respectively.

The approach proposed in this paper is distinct from the established frameworks due to the incentives held by the attacker. The attacker $\mathcal{A}$ may not arbitrarily attack a protocol. Instead, he (or she) sabotages a protocol with specific incentives by paying necessary costs. We formulate the incentives through utilities. That is, the attacker has incentives to sabotage a protocol if it brings positive utilities. Therefore, the attacker should find additional criteria in the computation for costs or losses. For example, the attacker has higher probability $\boldsymbol{\gamma} = \{\gamma_1, \gamma_2, \gamma_3, \ldots, \gamma_n\}$ and $\boldsymbol{\delta} = \{\delta_1, \delta_2, \delta_3, \ldots, \delta_n\}$ on the outcomes of parties $A$ and $B$. In this case, the corresponding outcome and utility of $\mathcal{A}$ are denoted as $\mathbf{o}_\mathcal{A} = \{o_\mathcal{A}^1, o_\mathcal{A}^2, o_\mathcal{A}^3, \ldots, o_\mathcal{A}^n\}$ and $\mathbf{u}_\mathcal{A} = \{u_\mathcal{A}^1, u_\mathcal{A}^2, u_\mathcal{A}^3, \ldots, u_\mathcal{A}^n\}$, respectively. Note that $\mathbf{u}_\mathcal{A}$ may be different under the cases of corruptions $A$ and $B$. For simplicity, in this paper, we only use $\mathbf{u}_\mathcal{A}$ to denote the corresponding utility of $\mathcal{A}$ when one party is corrupted. It will not exclude the case where both A and B are corrupted, which is often ignored in two-party computation protocols.

## 3 Attacking model and the corresponding utility functions

The attacking model on two-party protocols consists of three corruption cases: no one is corrupted, only one is corrupted and both are corrupted. In this section, we only present definitions of the utility functions with respect to these cases. In the following section, we instantiate the utilities by using a concrete protocol.

**No one is corrupted**

$A$ and $B$ execute protocol $\pi$. $\mathcal{A}$ corrupts no one. The expected utilities are shown in Eq. (1).

$$
\begin{aligned}
U_{AB}^{A} &= \mathbf{u}_A \boldsymbol{\alpha}^T \\
U_{AB}^{B} &= \mathbf{u}_B \boldsymbol{\beta}^T \\
U_{AB}^{\mathcal{A}} &= f(m_1, m_2, \ldots, m_n).
\end{aligned} \tag{1}
$$

$U_{AB}^{A}$, $U_{AB}^{B}$ and $U_{AB}^{\mathcal{A}}$ denote the expected utilities of $A$, $B$ and $\mathcal{A}$, respectively, in the protocol $\pi_{AB}$. The utility of $\mathcal{A}$ is the function of the intermediate messages $m_1, m_2, \ldots, m_n$ within protocol $\pi$. It is hard to measure the attacker's utility when he/she does not participate in the protocol. Therefore, we assume a function $f(\cdot)$ with respect to the intermediate messages to denote his utility.

**One party is corrupted**

We only present the expected utility of the case when $A$ is corrupted. The case when $B$ is corrupted is analogously defined. $\mathcal{A}$ corrupts $A$ with cost $c_A$. The corruption cost $c_A$ should be at least $U_{AB}^{A}$, otherwise $A$ would not accept the corruption. $\mathcal{A}$ participates in the protocol $\pi$ with $B$ instead of $A$. The expected utilities are demonstrated in Eq. (2).

$$
\begin{aligned}
U_{\mathcal{A}B}^{\mathcal{A}} &= \mathbf{u}_{\mathcal{A}} \boldsymbol{\gamma}^T - c_A \\
U_{\mathcal{A}B}^{B} &= \mathbf{u}_B \boldsymbol{\beta}^T \\
U_{\mathcal{A}B}^{A} &= p_A c_A + \Delta_A.
\end{aligned} \tag{2}
$$

$U_{\mathcal{A}B}^{A}$, $U_{\mathcal{A}B}^{B}$ and $U_{CB}^{\mathcal{A}}$ denote the expected utilities of $A$, $B$ and $\mathcal{A}$, respectively, in the protocol $\pi$. $\Delta_A$ denotes the utility if $A$ does not learn the output.

**Both is corrupted**

Generally, corruption of two parties is often neglected since it is less important to discuss security property in two-party computation (Goldreich 2009). In this paper, we consider the attacking model on parties instead of the protocol itself. That is, the protocol per se is secure. On the contrary, we allow the attacker to corrupt the two parties with double costs. The balance between the utility gained by the corruption and the costs may be much higher than 0, which refers to the incentives for the attacker. In the example of two millionaires (Yao 1982), we assume that the inputs of two millionaires are $x$ and $y$, respectively. Let Greater$(f(x, y)_{x>y}) =$ (yes, no) if $x > y$, otherwise Greater$(f(x, y)_{x>y}) =$ (no, yes), where the first value is returned to the first millionaire and the second value to the second millionaire. It is identical for the attacker to corrupt one party and two parties if $x > y$ or $x < y$. An extreme case is $x = y$. If the attacker corrupts one party, he (or she) can learn which is of larger cost and the output of one party is either yes or no. However, if the attacker corrupts two parties, he (or she) may have the output (no, no), which means two millionaires have the same amount of money. If the attacker is another millionaire, the corruption of the two parties may infer additional information. In this paper, we include the case of the corruption of two parties and reason the practical scenario. It is possible $\mathcal{A}$ can learn the output by itself. The expected utilities are depicted in Eq. 3.

$$
\begin{aligned}
U_{\mathcal{A}\mathcal{A}}^{\mathcal{A}} &= \Delta_{\mathcal{A}} - c_A - c_B \\
U_{\mathcal{A}\mathcal{A}}^{A} &= c_A + \Delta'_A \\
U_{\mathcal{A}\mathcal{A}}^{B} &= c_B + \Delta'_B.
\end{aligned} \tag{3}
$$

$U_{\mathcal{A}\mathcal{A}}^{A}$, $U_{\mathcal{A}\mathcal{A}}^{B}$ and $U_{\mathcal{A}\mathcal{A}}^{\mathcal{A}}$ denote the utilities for parties $A$, $B$ and $\mathcal{A}$, respectively. $\Delta_{\mathcal{A}}$ denotes the utility when $\mathcal{A}$ learns the output. $\Delta'_A$ is the utility when $A$ does not learn the output. $\Delta'_B$ denotes the utility when $B$ does not learn the output.

We only list the utilities for general cases, and the concrete utilities depend on specific implements. An instance of utilities are presented in the following section.

## 4 An instance for the proposed attacking model

### 4.1 The basic framework of the hybrid protocol

We apply our proposed attacking model to the protocols (Moran et al. 2009; Katz 2007; Gordon et al. 2008; Gordon and Katz 2012; Groce and Katz 2012), which include a "pre-processing" stage and a "share-exchanging" stage. Recall that in the first stage, there exists a trusted party. We restate the framework of Groce and Katz (2012) for completeness.

**The first stage of our proposed framework**

1. Two parties $A$ and $B$ present their private inputs $x_A$ and $x_B$ to the trusted party, which correctly computes $f(x_A, x_B)$.
2. The trusted party selects $i^*$ ($i^* \in \{1, 2, \ldots, n\}$) according to a geometric distribution $p$.
3. Random values $r_i^A$ and $r_i^B$ are chosen:
   (a) $r_i^A$ and $r_i^B$ are randomly chosen in the domain of $f(\cdot)$ when $i < i^*$ ($i \in \{1, 2, \ldots, n\}$).
   (b) $r_i^A$ and $r_i^B$ are set to be $f(x_A, x_B)$ when $i \geq i^*$.
4. Randomly values $s_i^A, s_i^B$ and $t_i^A, t_i^B$ (shares of $r_i^A$ and $r_i^B$) are chosen such that $s_i^A \oplus t_i^A = r_i^A$ and $s_i^B \oplus t_i^B = r_i^B$. Message authentication codes (Black 2000) on values $s_i^A$, $s_i^B$ and $t_i^A, t_i^B$ are also generated to guarantee the validity of the shares.
5. $s_i^A$, $s_i^B$ and $t_i^A$, $t_i^B$ with their corresponding message authentication codes are presented to $A$ and $B$, respectively.

There are altogether $n$ rounds in the second stage. $A$ and $B$ exchange their shares in each round.

**Table 2** Utility of Groce and Katz (2012)

| | Correct | Incorrect |
|---|---|---|
| Correct | $(a_1, a_2)$ | $(b_1, c_2)$ |
| Incorrect | $(c_1, b_2)$ | $(d_1, d_1)$ |

**The second stage of our proposed framework**

1. In the $i$th round,
   (a) $A$ firstly passes $t_i^B$ to $B$. $B$ ensures the validity of the shares using the corresponding message authentication codes. $B$ computes $r_i^B = t_i^B \oplus s_i^B$ and the protocol moves to the second step.
   (b) $B$ passes $s_i^A$ to $A$. $A$ verifies the validity of the shares using the corresponding message authentication codes. $A$ computes $r_i^A = t_i^A \oplus s_i^A$.
2. Each party considers its latest reconstructed value as its final output.
3. If both parties do not abort in the $i$th round, the protocol will move into the $i + 1$th round.

The utility matrix of Groce and Katz (2012) can be presented in a matrix (ref. Table 2).

### 4.2 Analysis of the attacking model

We apply our attacking model to the protocol of Groce and Katz (2012), where both parties $A$ and $B$ are rational without the participation of an external/internal attacker $\mathcal{A}$. There are two stages in Groce and Katz (2012): $A$ and $B$ receive shares of the result at the end of the first stage and exchange the shares in the second stage to reconstruct the result. In this paper, we consider practical settings and an attacking model for the second stage.

1. Suppose $A$ and $B$ have private types: *honest* or *dishonest* with incomplete information. Here, *honest* means that the parties honestly pass shares in each round and *dishonest* means that the parties abort in a certain round. Note that we do not consider the case where parties send fake shares since they will be detected due to the validity of message authentication codes (Black 2000).
2. $A$ and $B$ have a prior probability on the private types. $B$ treats $A$ as *honest* with probability $\mu$. $A$ regards $B$ as *honest* with probability $\nu$. Both parties hold the expected utilities at the end of the protocol.
3. The practical attacker $\mathcal{A}$ owns some additional information on the private types of $A$ and $B$. $\mathcal{A}$ regards $A$ as *honest* with probability $\mu$ and $B$ as *honest* with probability $\nu$. We assume that $\eta > \mu$ and $\theta > \nu$. Otherwise, $\mathcal{A}$ has no incentives to attack this protocol. Note that $A$ and $B$ learn nothing about $\eta$ and $\theta$. Furthermore, they may not even know that $\mathcal{A}$ own the additional information.
4. $\mathcal{A}$ may seek some advantage attacking one party or both of them, which depends on the utility functions. Here, when stating $\mathcal{A}$ corrupts the parties, we mean that $\mathcal{A}$ bribes one party or both the parties with required costs and participate the protocol with the replacement of the party or both of them.

Table 2 shows that $b_1 > a_1 \geq d_1 \geq c_1$ and $b_2 > a_2 \geq d_2 \geq c_2$. Correct means that the party learns the correct output, and Incorrect means that the party learns an incorrect output. In Groce and Katz (2012), the utilities of $A$ and $B$ are defined according to Table 2.

In this section, we list the expected utility of $\mathcal{A}$ and analyze the conditions for $\mathcal{A}$ to corrupt only one party. Consequently, the conditions for the cases, no one or both are corrupted, can be drawn based on the conditions mentioned above. That is, $\mathcal{A}$ has incentives to corrupt one party if he gets advantages for him compared with the case when no one is corrupted. Therefore, we should first get the expected utilities $U^A$ and $U^B$ when no one is corrupted, which are detailed below. Here, $U_X^H$ and $U_X^D$ denote the utility of $X$ when $\mathcal{A}$ treats his (or her) opponent $Y$ as an honest and dishonest ones, respectively, where $X, Y \in \{A, B, \mathcal{A}\}$ and $X \neq Y$ [ref. Eq. (4)].

$$\begin{aligned} U^A &= \nu U_A^H + (1-\nu)U_A^D \\ &= \nu a_1 + (1-\nu)[\varphi d_1 + pc_1 + (1-p-\varphi)a_1] \\ U^B &= \mu U_B^H + (1-\mu)U_B^D \\ &= \mu[\varphi d_2 + pb_2 + (1-p-\varphi)a_2] \\ &\quad + (1-\mu)[\varphi d_2 + pd_2 + (1-p-\varphi)a_2]. \end{aligned} \tag{4}$$

Let $c_A = U^A$, $c_B = U^B$ be the maximum costs for $\mathcal{A}$ to corrupt $A$ and $B$, respectively. Let $U_{\mathcal{A}B}^B = U_{AB}^B$ and $U_{A\mathcal{A}}^A = U_{AB}^A$ [ref. Eq. (5)].

$$\begin{aligned} U_{\mathcal{A}B}^{\mathcal{A}} &= \theta U_{\mathcal{A}}^H + (1-\theta)U_{\mathcal{A}}^D - c_A \\ &= \theta a_1 + (1-\theta)[\varphi d_1 + pc_1 + (1-p-\varphi)a_1] - c_A \\ &= (\theta - \nu)[p(a_1 - c_1) + \varphi(a_1 - d_1)] \\ U_{A\mathcal{A}}^{\mathcal{A}} &= \eta U_{\mathcal{A}}^H + (1-\eta)U_{\mathcal{A}}^D - c_B \\ &= \eta[\varphi d_2 + pb_2 + (1-p-\varphi)a_2] \\ &\quad + (1-\eta)[\varphi d_2 + pd_2 + (1-p-\varphi)a_2] - c_B \\ &= (\eta - \mu)p(b_2 - d_2). \end{aligned} \tag{5}$$

Here, $p$ denotes the probability of $i = i^*$, after which round both parties reconstruct the output. However, $B$ may reconstruct the output, but $A$ cannot achieve the same when $B$ receives the share and aborts in the $i^*$th round. Let $\varphi$ denote

the probability of $i < i^*$, where both parties reconstruct random values. Recall that we assume $\theta > \nu$, $\eta > \mu$, $b_1 > a_1 \geq d_1 \geq c_1$ and $b_2 > a_2 \geq d_2 \geq c_2$. It satisfies that $U^{\mathcal{A}}_{\mathcal{A}B} > 0$, $U^{\mathcal{A}}_{A\mathcal{A}} > 0$. That is, given necessary information on the types of $A$ and $B$, $\mathcal{A}$ has incentives to corrupt one party or both of them since the advantages are positive.

**Definition 1** The advantage is defined as the additional income for an attacker, which is the attacker's utility minus the corruption cost.

The advantage is used for describing the attacker's incentives to corrupt parties. Recall that existing works malicious attackers sabotage protocols without reason, whom are simply assumed to break the security of the protocols.

**Proposition 1** *The attacker has incentives to corrupt parties if his advantage is positive.*

In this paper, we utilize the notion of advantage to measure the incentives for the attacker to corrupt the parties. For example, the attacker has strong incentives to corrupt parties if advantage is large enough.

**Theorem 1** *Given $\theta > \nu$, $\eta > \mu$, $b_1 > a_1 \geq d_1 \geq c_1$ and $b_2 > a_2 \geq d_2 \geq c_2$, it is possible for $\mathcal{A}$ to corrupt one or two parties.*

***Proof*** (Brief:) The inequations $\theta > \nu$, $\eta > \mu$ mean that the attacker has additional information with respect to the private types of the parties. Each party learns little about the private type of his opponent. Therefore, they are cautious when they participate into the protocol. On the contrary, the attacker masters more information and he may take adventurous actions when he participate into the protocol. However, he should first corrupt one or two parties. Equation (5) lists the attacker's advantages when he corrupts $A$ and $B$, respectively.

Given $\theta > \nu$, $\eta > \mu$, $b_1 > a_1 \geq d_1 \geq c_1$ and $b_2 > a_2 \geq d_2 \geq c_2$, it satisfies that $U^{\mathcal{A}}_{\mathcal{A}B} > 0$, $U^{\mathcal{A}}_{A\mathcal{A}} > 0$. Thus, the attacker has incentives to corrupt one or two parties according to Proposition 1. □

As we have mentioned above, the attacker may be an internal party. For example, $B$ may bribe $A$ in the protocol. That is, $B$ bribes $A$ with cost $c_A$, has the input of $A$ and then learns the output. We assume the utility is still calculated referring to Table 2. It can be derived that $B$ has incentives to bribe $A$ if $U^A - b_2 < c_A < c_1 - U^B$.

### 4.3 A random solution for the proposed attacking model

The assumption on the adversary $\mathcal{A}$ is strong enough to corrupt both the parties, where the adversary's income is no less than $c_A + c_B$. However, we may have a sensible attacking model in practice if the adversary dominates the entire system. Therefore, measures must be taken to prevent such attack. The intuition is that two parties may resort to some cryptographic primitives in order to identify the membership of them. More specifically, two parties may request an accumulator, a one-way function (Derler et al. 2015), with probability $\psi > 0$ so as to enforce cooperation before they exchange their shares. The function of the accumulator is to proving a membership without leaking information with respect to any individual members. Note that $\psi$ is not necessary to be 1 since the introduction of the accumulator through the hybrid protocol may increase the computational complexity. Therefore, we only choose a proper $\psi$ to deter the adversary and prevent attacking.

In the scenario where the adversary corrupts two parties, one party (say $A$) has utility $c_A$ when he (or she) is corrupted (bribed). Suppose $A$ calls an accumulator and cooperates with $B$ with probability $\psi$, the expected utility is: $\psi a_1 + (1-\psi)c_A$. It should satisfy $\psi a_1 + (1-\psi)c_A > c_A$ such that the attack is prevented. It satisfies $a_2 > c_B$ for the same reason. Recall that in Eq. (2), $\mathbf{u}_{\mathcal{A}}\boldsymbol{\gamma}^T$ is at least $a_1$ and $\mathbf{u}_{\mathcal{A}}\boldsymbol{\delta}^T$ is at least $a_2$. In Eq. (3), $\Delta_{\mathcal{A}}$ is at least $a_1 + a_2$, otherwise $\mathcal{A}$ has no incentives to corrupt one party or two parties. Therefore, the conditions for the two parties to resist the attack, where $a_1 > c_A$ or (and) $a_2 > c_B$, are satisfied. That is, the adversary $\mathcal{A}$ cannot conduct the attack mentioned in Sect. 4.2 when $\psi > 0$.

## 5 Conclusion

Secure two-party computation protocols may avoid attacks in the communication between two distributed parties in the presence of adversaries. In general, these protocols should satisfy specific security requirements like privacy, correctness and fairness. However, the abilities for the adversary are over-estimated and attacks to the system cannot be avoided in the real world. Therefore, the discussion under such settings can be made for impractical protocols. In this paper, we firstly considered a security problem in a realistic scenario, which cannot be directly solved by the existing two-party protocols. We proposed a new attacking model, where the attacker has asymmetrical information compared with the case of two parties. The attacker has incentives to corrupt one party or both of them if he (or she) is awarded enough by corrupting the protocols. We have derived the probability of corrupting both parties by giving an instance based on the protocol reported in Groce and Katz (2012). Finally, we compared our work against the other techniques in six aspects and concluded that our work has more better achievements than the others.

The future work includes the introduction of more intelligent settings like stealing shares instead of bribing parties. Furthermore, we will also consider the collusion between two

parties or between the attacker and one party and demonstrate its impact on the communication security.

## Compliance with ethical standards

**Funding** This study was funded by National Natural Science Foundation of China (Grant Number: 61502218, 61771231), Natural Science Foundation of Shandong Province (Grant Number: ZR2017MF010, ZR2017MF062), H. Zhou was funded by EU H2020 DOMINOES Project (Grant Number: 771066).

**Conflict of Interest** The authors declare that they have no conflict of interest.

**Ethical approval** This article does not contain any studies with human participants or animals performed by any of the authors.



## References

Abraham S, Chengalur-Smith IS (2010) An overview of social engineering malware: trends, tactics, and implications. Technol Soc 32(3):183–196

Adat V, Dahiya A, Gupta BB (2018) Economic incentive based solution against distributed denial of service attacks for IoT customers. In: IEEE international conference on consumer electronics, ICCE 2018, Las Vegas, NV, USA, January 12–14, 2018, pp 1–5. https://doi.org/10.1109/ICCE.2018.8326280

Andrychowicz M, Dziembowski S, Malinowski D, Mazurek Ł (2014) Fair two-party computations via bitcoin deposits. In: International conference on financial cryptography and data security. Springer, Berlin, pp 105–121

Asharov G, Canetti R, Hazay C (2011) Towards a game theoretic view of secure computation. EUROCRYPT 6632:426–445

Bentov I, Kumaresan R (2014) How to use bitcoin to design fair protocols. In: International cryptology conference. Springer, Berlin, pp 421–439

Black JR (2000) Message authentication codes. University of California, Davis

Chen X, Li J, Weng J, Ma J, Lou W (2016) Verifiable computation over large database with incremental updates. IEEE Trans Comput 65(10):3184–3195

Derler D, Hanser C, Slamanig D (2015) Revisiting cryptographic accumulators, additional properties and relations to other primitives. In: Cryptographers∘∅ track at the RSA conference. Springer, San Francisco, pp 127–144

Fan L, Lei X, Yang N, Duong TQ, Karagiannidis GK (2017) Secrecy cooperative networks with outdated relay selection over correlated fading channels. IEEE Trans Veh Technol 66(8):7599–7603

Gao C, Cheng Q, He P, Susilo W, Li J (2018) Privacy-preserving Naive Bayes classifiers secure against the substitution-then-comparison attack. Inf Sci 444:72–88

Garay J, Katz J, Maurer U, Tackmann B, Zikas V (2013) Rational protocol design: cryptography against incentive-driven adversaries. In: 2013 IEEE 54th annual symposium on foundations of computer science (FOCS). IEEE, pp 648–657

Goldreich O (2009) Foundations of cryptography: volume 2, basic applications. Cambridge University Press, Cambridge

Gordon SD, Katz J (2012) Partial fairness in secure two-party computation. J Cryptol 25(1):14–40

Gordon SD, Hazay C, Lindell Y, Katz J (2008) Complete fairness in secure two-party computation. In 40th annual ACM symposium on theory of computing (STOC). Citeseer

Groce A, Katz J (2012) Fair computation with rational players. In: Annual international conference on the theory and applications of cryptographic techniques. Springer, Berlin, pp 81–98

Gupta BB, Agrawal DP, Yamaguchi S (2016) Handbook of research on modern cryptographic solutions for computer and cyber security. IGI Global, Hershey

Haddi FL, Benchaïba M (2015) A survey of incentive mechanisms in static and mobile p2p systems. J Netw Comput Appl 58:108–118

Hadnagy C (2010) Social engineering: the art of human hacking. J Quant Spectrosc Radiat Trans 130(11):51–61

Halpern J, Teague V (2004) Rational secret sharing and multiparty computation: extended abstract. In: STOC 2004: proceedings of the 36th annum ACM symposium on theory of computing, New York, USA: ACM, pp 623–632

Ibtihal M, Driss EO, Hassan N (2017) Homomorphic encryption as a service for outsourced images in mobile cloud computing environment. Int J Cloud Appl Comput 7(2):27–40

Jhaveri RH, Patel NM, Zhong Y, Sangaiah AK (2018) Sensitivity analysis of an attack-pattern discovery based trusted routing scheme for mobile ad-hoc networks in industrial IoT. IEEE Access. https://doi.org/10.1109/ACCESS.2018.2822945

Jiang W, Guojun Wang M, Bhuiyan ZA, Jie W (2016) Understanding graph-based trust evaluation in online social networks: methodologies and challenges. ACM Comput Surv (CSUR) 49(1):10

Jiang L, Cheng Y, Yang L, Li J, Yan H, Wang X (2018) A trust-based collaborative filtering algorithm for e-commerce recommendation system. J Ambient Intell Humaniz Comput. https://doi.org/10.1007/s12652-018-0887-z

Katz J (2007) On achieving the best of both worlds in secure multiparty computation. In: Proceedings of the thirty-ninth annual ACM symposium on theory of computing. ACM, pp 11–20

Kumaresan R, Bentov I (2014) How to use bitcoin to incentivize correct computations. In: Proceedings of the 2014 ACM SIGSAC conference on computer and communications security. ACM, pp 30–41

Li J, Chen X, Xhafa F, Barolli L (2015) Secure deduplication storage systems supporting keyword search. J Comput Syst Sci 81(8):1532–1541

Li P, Li J, Huang Z, Li T, Gao C-Z, Yiu S-M, Chen K (2017) Multi-key privacy-preserving deep learning in cloud computing. Future Gener Comput Syst 74:76–85

Li T, Li J, Liu Z, Li P, Jia C (2018) Differentially private naive bayes learning over multiple data sources. Inf Sci 444:89–104

Lin Q, Yan H, Huang Z, Chen W, Shen J, Tang Y (2018) An id-based linearly homomorphic signature scheme and its application in blockchain. IEEE Access IEEE. https://doi.org/10.1109/ACCESS.2018.2809426

Liu Z, Wu Z, Li T, Li J, Shen C (2018a) GMM and CNN hybrid method for short utterance speaker recognition. IEEE Trans Ind Inform IEEE. https://doi.org/10.1109/TII.2018.2799928

Liu Z, Huang Y, Li J, Cheng X, Shen C (2018b) Divoram: towards a practical oblivious ram with variable block size. Inf Sci 447:1–11

Meng W, Tischhauser E, Wang Q, Wang Y, Han J (2018) When intrusion detection meets blockchain technology: a review. IEEE Access. https://doi.org/10.1109/ACCESS.2018.2799854

Moran T, Naor M, Segev G (2009) An optimally fair coin toss. In: Reingold O (ed) Theory of cryptography. Springer, Berlin, pp 1–18

Tian H, Chen Z, Chang C-C, Huang Y, Wang T, Huang Z, Cai Y, Chen Y (2018) Public audit for operation behavior logs with error locating

in cloud storage. Soft Comput. https://doi.org/10.1007/s00500-018-3038-8
Wang Y, Li T, Chen L, Li P, Leung H, Liu Z, Qiuliang X (2016) Rational computing protocol based on fuzzy theory. Soft Comput 20(2):429–438
Wang Y, Zhang S, Tang Y, Su Q, Chen B (2018) Rational adversary with flexible utility in secure two-party computation. J Ambient Intell Human Comput, pp 1–15
Wu T-Y, Lee W-T, Guizani N, Wang T-M (2014) Incentive mechanism for p2p file sharing based on social network and game theory. J Netw Comput Appl 41:47–55
Yang L, Han Z, Huang Z, Ma J (2018) A remotely keyed file encryption scheme under mobile cloud computing. J Netw Comput Appl 106:90–99
Yao AC (1982) Protocols for secure computations. In: 23rd annual symposium on foundations of computer science, 1982. SFCS'08. IEEE, pp 160–164
Yu C, Li J, Li X, Ren X, Gupta BB (2018) Four-image encryption scheme based on quaternion fresnel transform, chaos and computer generated hologram. Multimed Tools Appl 77(4):4585–4608
Zhang X, Tan Y-A, Liang C, Li Y, Li J (2018) A covert channel over volte via adjusting silence periods. IEEE Access 6:9292–9302
Zhao H, Yang X, Li X (2012) An incentive mechanism to reinforce truthful reports in reputation systems. J Netw Comput Appl 35(3):951–961
Zheng Q, Wang X, Muhammad KK, Wenfang Z, Gupta BB, Wei G (2018) A lightweight authenticated encryption scheme based on chaotic scml for railway cloud service. IEEE Access 6(99):711–722