On Perfectly Secure Two-Party Computation for Symmetric Functionalities with Correlated Randomness

Bar Alon[*] alonbar08@gmail.com

Olga Nissenbaum[*] olga@nissenbaum.ru

Eran Omri[*†] omrier@ariel.ac.il

Anat Paskin-Cherniavsky[‡] anatpc@ariel.ac.il

Arpita Patra[§] arpita@iisc.ac.in

Abstract

A multiparty computation protocol is *perfectly secure* for some function f if it perfectly emulates an ideal computation of f. Thus, perfect security is the strongest and most desirable notion of security, as it guarantees security in the face of any adversary and eliminates the dependency on any security parameter. Ben-Or et al. [STOC '88] and Chaum et al. [STOC '88] showed that any function can be computed with perfect security if strictly less than one-third of the parties can be corrupted. For two-party sender-receiver functionalities (where only one party receives an output), Ishai et al. [TCC '13] showed that any function can be computed with perfect security in the correlated randomness model. Unfortunately, they also showed that perfect security cannot be achieved in general for two-party functions that give outputs to both parties (even in the correlated randomness model).

We study the feasibility of obtaining perfect security for deterministic symmetric two-party functionalities (i.e., where both parties obtain the same output) in the face of malicious adversaries. We explore both the plain model as well as the correlated randomness model. We provide positive results in the plain model, and negative results in the correlated randomness model. As a corollary, we obtain the following results.

1. We provide a characterization of symmetric functionalities with (up to) four possible outputs that can be computed with perfect security. The characterization is further refined when restricted to three possible outputs and to Boolean functions. All characterizations are the same for both the plain model and the correlated randomness model.
2. We show that if a functionality contains an embedded XOR or an embedded AND, then it cannot be computed with perfect security (even in the correlated randomness model).

Keywords: perfect security; two-party computation; correlated randomness

[*]Department of Computer Science, Ariel University. Ariel Cyber Innovation Center (ACIC). Work was supported in part by grants from the Israel Science Foundation (no.152/17), and by the Ariel Cyber Innovation Center in conjunction with the Israel National Cyber directorate in the Prime Minister's Office.

[†]Work was supported in part by the Robert L. McDevitt, K.S.G., K.C.H.S. and Catherine H. McDevitt L.C.H.S. endowment at Georgetown University. Part of this work was done while hosted by Georgetown University.

[‡]Department of Computer Science, Ariel University. Ariel Cyber Innovation Center (ACIC). Work was supported in part by the Ariel Cyber Innovation Center in conjunction with the Israel National Cyber directorate in the Prime Minister's Office.

[§]Indian Institute of Science, Bangalore, India. Arpita Patra would like to acknowledge financial support from DST National Mission on Interdisciplinary Cyber-Physical Systems (NM-ICPS) 2020-2025 and SERB MATRICS (Theoretical Sciences) Grant 2020-2023.

Contents

1 Introduction

Secure Multiparty Computation (MPC) protocols allow a set of mutually distrusting parties to compute a joint function of their private inputs. The two main security properties that are desirable for protocols are correctness of the computation and privacy (i.e., the adversary should not learn anything about the inputs or outputs of the honest parties except what is leaked from the output of the function). There are two main types of adversaries that are considered. These are semi-honest (passive) adversaries and malicious (active) adversaries. A semi-honest adversary always follows the prescribed protocol, but may try to infer additional information from the joint view of the corrupted parties in the protocol. A malicious adversary may instruct the corrupted parties to deviate from the prescribed protocol in any manner it chooses.

A general paradigm for defining the desired security of protocols is known as the ideal vs real paradigm. This paradigm avoids the need to specify a list of desired properties. Rather, security is defined by describing an ideal functionality, where parties interact via a trusted party to compute the task at hand. A real-world protocol is then deemed secure, if no adversary can do more harm than an adversary in the ideal-world. In a bit more detail, the definition requires that the view of the adversary in a real-world execution, can be simulated by an adversary (corrupting the same parties) in the ideal-world. There are three types of measurements for the strength of security that may be considered. These are called computational, statistical, and perfect security. Computational security requires that the distribution of the view in the real-world is indistinguishable from the distribution of the view in the ideal-world to a computationally bounded machine. Statistical security requires these distributions to be statistically close (indistinguishable even for unbounded machines). Finally, perfect security means that the views in both worlds are identically distributed.

In this paper we consider perfect security for two-party computation (i.e., with no honest majority), in the face of malicious adversaries (when considering perfect security, we naturally assume the adversary to be computationally unbounded). Apart from being a natural research goal, perfect security provides important and useful security advantages over protocols that offer computational security and even over those that have a negligible probability of failure (i.e., offer statistical security). Because of the stringent requirement, perfectly secure constructions tend to have a simple structure. More importantly, perfect security completely eliminates the need for a security parameter, making protocols that are perfectly secure highly scalable.

Perfect security in the plain model

In the basic setting of secure computation, parties communicate with each other over some communication network. It is generally assumed that the channels are secure, but no other setup assumption is made. In this setting, Ben-Or et al. [2], Chaum et al. [5] showed the feasibility of computing any function with perfect security in the face of malicious adversaries that can corrupt strictly less than one-third of the parties.[1]

In the two party setting, Kushilevitz [10] characterized the set of functions that can be computed with perfect security in the face of semi-honest adversaries. Cleve [7] showed that full-security (where honest parties always receive an output) is impossible in general, even for computationally bounded malicious adversaries. For the setting of two-party plain-model protocols with perfect security in the face of malicious adversaries, very little was known prior to our work.

[1] For semi-honest adversaries, they showed that an honest majority is sufficient.

Perfect security with correlated randomness

It is natural to ask whether the impossibilities of obtaining prefect security can be circumvented by making some reasonable assumption. This brings to the table the correlated-randomness model that is both theoretically and practically motivated. In this model, parties are given strings sampled from some fixed joint distribution at the onset of the protocol. These strings are independent of their inputs, and are then used alongside the inputs of the parties to run a secure computation protocol. Interestingly, Cleve's impossibility result does not apply to this setting.

In the correlated randomness setting, Ishai et al. [9] showed that it is possible to construct perfectly and maliciously secure protocols in the sender-receiver model, i.e., where both parties have an input, but only one receives an output. On the negative side, [9] showed that, in general, perfect security is impossible to achieve for two-party functionalities that deliver outputs to both the parties. In particular, they show that it is impossible to compute the XOR function in this setting. In fact, the negative implication carries forward even to security with abort, where the adversary may itself get the output, but can deprive the honest parties from the output. Other than this result, very little was known prior to our work regarding perfect security in the correlated randomness setting where both parties receive an output.

In light of the above, the main question studied in this paper is.

Characterize the set of two-party functionalities that can be computed with malicious perfect security in the plain model and in the correlated randomness model.

We make substantial progress in this direction and leave open several challenging followup questions. We summarize our results below.

1.1 Our Contribution

In this work, we consider the model of two-party computation of deterministic symmetric functionalities (i.e., where both parties have the same output in the computation). We are interested in perfect security and consider computation both in the plain model and in the correlated randomness model. We provide both positive results in the plain model, and negative results in the correlated randomness model. In particular, our results form a full characterization for four-output functionalities, showing that there are only two families of functionalities that can be computed with perfect security.

Before giving the results in more details, let us first define the two families of functionalities mentioned above. In the following, for any symmetric deterministic functionality $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{Z}$, we associate with it a matrix $\mathbf{M}_f \in \mathcal{Z}^{|\mathcal{X}|\times|\mathcal{Y}|}$ defined as $\mathbf{M}_f(x, y) = f(x, y)$, for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.

Definition 1.1 (Spiral functionality, informal)**.** *A symmetric deterministic functionality $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{Z}$ is called* spiral*, if $\mathbf{M}_f$ is either constant, or, up to permuting the rows and columns, and transposing the matrix, is of the form $(\mathbf{M}||\mathbf{M}')$ where $\mathbf{M}$ is constant-column, $\mathbf{M}'$ is spiral, and where the set of entries in the two matrices are disjoint.*

As an example, consider the following spiral matrix.

$$\left(\begin{array}{ccc|c|c}
7 & 7 & \multicolumn{1}{c}{7} & \multicolumn{1}{c}{7} & 7 \\
6 & 6 & \multicolumn{1}{c}{6} & \multicolumn{1}{c}{6} & 6 \\
\hline
5 & 4 & 4 & 1 & 2 \\ \cline{4-4}
5 & 4 & 4 & 0 & 2 \\ \cline{4-5}
5 & 4 & 4 & \multicolumn{1}{c}{3} & 3
\end{array}\right)$$

Definition 1.2 (Transparent transfer functionality, informal). *A symmetric deterministic functionality $f : \mathcal{X} \times \mathcal{Y} \mapsto \{0, 1, 2, 3\}$ is called* transparent transfer *if, up to permuting and duplicating the rows and columns, and transposing the matrix, $\mathbf{M}_f$ is of the form*

$$\begin{pmatrix} a & c \\ a & d \\ b & c \\ b & d \end{pmatrix} \tag{1}$$

where $\{a, b, c, d\} = \{0, 1, 2, 3\}$.

We refer the reader to Remark 3.4 for the reasoning behind the name. We are now ready to state our main result, providing a full characterization for the four-output functionalities that can be computed with perfect security.

Theorem 1.3 (Characterization of four-output functionalities, informal). *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \{0, 1, 2, 3\}$ be a symmetric deterministic four-output two-party functionality. If f can be computed with perfect security in the correlated randomness model, then f is either spiral or transparent transfer. Conversely, any spiral and transparent transfer functionality can be computed with perfect security in the plain model.*

A few notes are in place. First, observe that, in particular, we obtain a characterization for symmetric ternary-output and Boolean functionalities. Specifically, since transparent transfer functionalities require four outputs, for the ternary-output case, it follows that the only functionalities that can be computed with perfect security are spiral. Thus, we have the following.

Corollary 1.4. *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \{0, 1, 2\}$ be a symmetric deterministic ternary-output two-party functionality. If f can be computed with perfect security in the correlated randomness model, then f is spiral. Conversely, any spiral functionality can be computed with perfect security in the plain model.*

As for the Boolean case, observe that a Boolean functionality is spiral if and only if it is independent of one of its inputs, which we refer to as trivial functionalities. Therefore, we obtain the following result.

Corollary 1.5. *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \{0, 1\}$ be a Boolean symmetric deterministic two-party functionality. If f can be computed with perfect security in the correlated randomness model, then f is trivial. Conversely, any trivial functionality can be computed with perfect security in the plain model.*

Second, although our main results consider only four-output functionalities, we stress that both our positive and negative results can be extended to the more general case. However, it is currently unknown if these results provide a characterization for even five-output functionalities.

Third, observe that Theorem 1.3 implies that for four-output functionalities, the plain model and the correlated randomness model are equivalent.

Finally, our techniques for the negative direction provide an impossibility result for a larger class of functionalities, including those with more than four outputs. An interesting corollary of this general result, is that if a functionality has an embedded XOR or an embedded AND,[2] then the functionality cannot be computed with perfect security.

functionality	**trivial**	**spiral**	**transparent transfer**
Boolean	$\begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}$	-	-
ternary	$\begin{pmatrix} 0 & 1 & 2 \\ 0 & 1 & 2 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 \\ 0 & 2 \end{pmatrix}$	-
four-output	$\begin{pmatrix} 0 & 1 & 2 & 3 \\ 0 & 1 & 2 & 3 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 2 \\ 0 & 1 & 3 \end{pmatrix}; \begin{pmatrix} 0 & 1 & 1 \\ 0 & 3 & 2 \end{pmatrix}; \begin{pmatrix} 0 & 0 & 0 \\ 1 & 2 & 3 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & 1 & 1 \\ 2 & 3 & 2 & 3 \end{pmatrix}$

Table 1: The table above shows the functionalities that can be computed with perfect security with correlated randomness (for presentation, we do not include constant functions). As stated in Theorem 1.3, up to transposing the matrix, re-encoding the output, and permuting and duplicating the rows and columns, these are the *only* functionalities that can be computed with perfect security.

1.2 Our Techniques

We now turn to describe our techniques. To warm-up for our techniques, we first briefly explain the impossibility result for the symmetric XOR functionality $\mathsf{XOR}(x, y) = x \oplus y$ due to Ishai et al. [9]. We then show where it falls short even for the AND functionality $\mathsf{AND}(x, y) = x \wedge y$. Then, we show how to overcome this shortcoming and prove a general impossibility result. Finally, we show how to compute spiral and transparent transfer functionalities with perfect security.

Impossibility of XOR. Let us start with recalling the proof that $\mathsf{XOR}(x, y) = x \oplus y$ cannot be computed with perfect security, even when the parties are given correlated randomness. Assume towards contradiction that there is a protocol Π for computing f with perfect security in the correlated randomness model.

Consider an execution of Π on inputs $(x, y) \leftarrow \{0, 1\}^2$ chosen uniformly at random. Since the protocol is perfectly correct, there exists a round where the output of party A is fixed (e.g., the last round). That is, regardless of the correlated randomness generated for the parties, any continuation of the protocol results in A outputting $x \oplus y$. Let i be the first such round. Similarly, let j be the first round, where the output of B is fixed to $x \oplus y$. Since the parties send message one after the

[2]A functionality f is said to have an embedded XOR if there exists $x_1, x_2 \in \mathcal{X}$ and $y_1, y_2 \in \mathcal{Y}$ such that $f(x_1, y_1) = f(x_2, y_2) \neq f(x_1, y_2) = f(x_2, y_1)$. The functionality is said to have an embedded AND if $f(x_2, y_2) \neq f(x_1, y_1) = f(x_1, y_2) = f(x_2, y_1)$.

other, it holds that $i \neq j$. Assume without loss of generality that $i < j$. Then at round i, party A "knows" the output, while party B does not. In more details, there exists correlated randomness (r_1, r_2) for which at round i, there exists messages that A can send causing party B to output $1 \oplus x \oplus y$.

Consider the following adversary $\mathcal{A}$ corrupting A, that aims to "bias" the output of B towards 0. It instructs A to behave honestly until round i. At this point, $\mathcal{A}$ can locally compute the output $z = x \oplus y$. If $z = 0$, then it instructs A to continue honestly until the termination of the protocol. Otherwise, it sends random messages sampled independently and uniformly random.

Observe that the probability the adversary sees $z = 0$ is $1/2$, where the probability is taken over the sampling of the inputs and the correlated randomness. In this case, by the definition of $\mathcal{A}$, the honest party will output 0. On the other hand, if $z = 1$, then as the output of B is not fixed, there is a non-zero probability that both the correlated randomness is (r_1, r_2), and A sends the "correct messages" to B, causing it to output 0. Overall, it follows that the probability that B outputs 0 is *strictly greater* than $1/2$. On the other hand, in the ideal world, the output of B is 0 with probability exactly $1/2$ regardless of the input of corrupted A to the trusted party, since B's input y is chosen uniformly at random.

Impossibility of AND. Before generalizing the impossibility result of [9] let us first explain where their argument fails even for the AND functionality $\mathsf{AND}(x, y) = x \wedge y$. Consider the adversary $\mathcal{A}$ defined previously, that aims to bias the output of B towards 0. Note that if $y = 1$, then a simulator can simulate the attack by sending $x = 0$ with the "correct" probability (i.e., the probability that the correlated randomness and the messages that $\mathcal{A}$ sends cause B to output 0). On the other hand, if $y = 0$, then regardless of what $\mathcal{A}$ does in the real world, B already "knows" that the output is 0, thus $\mathcal{A}$ cannot introduce any bias. A similar argument shows that biasing towards 1 might also be simulatable.

To overcome this issue, instead of just biasing the output of the honest party towards a certain value, we let the adversary also guess uniformly at random the input of the honest party. To see why it works, let us first analyze the probability that $\mathcal{A}$ biases the output of B towards 0 and guesses its input correctly. Let Succ be the event where the adversary succeeds. First, consider the case where $x = 0$. Here, $\mathcal{A}$ will guess y correctly with probability $1/2$, and always cause B to output 0. Therefore, $\Pr[\mathsf{Succ} \mid x = 0] = 1/2$. Next, consider the case where $x = 1$. In this case, $\mathcal{A}$ always learns y from the output. Additionally, if $y = 0$ then B will always output 0. If $y = 1$, then $\mathcal{A}$ will send random messages starting at round i, hence with non-zero probability, B will output 0. Therefore,

$$\begin{aligned}
\Pr[\mathsf{Succ} \mid x = 1] &= \Pr[y = 0] \cdot \Pr[\mathsf{Succ} \mid x = 1 \wedge y = 0] + \Pr[y = 1] \cdot \Pr[\mathsf{Succ} \mid x = 1 \wedge y = 1] \\
&= \frac{1}{2} + \frac{1}{2} \cdot \Pr[\mathsf{Succ} \mid x = 1 \wedge y = 1] \\
&> \frac{1}{2}.
\end{aligned}$$

Overall, we conclude that the adversary succeeds with probability $\Pr[\mathsf{Succ}] > 1/2$.

To see why no simulator exists for $\mathcal{A}$, observe that if a simulator sends $x = 0$ to the trusted party, then it does not obtain any information on y, and if it sends $x = 1$, then B will output 0 only if $y = 0$, which occurs with probability $1/2$. Overall, the simulator can succeed with probability at most $1/2$, hence no simulator can perfectly simulate $\mathcal{A}$.

Generalizing the impossibility result. We now explain how to generalize the above argument to a more general, possibly non-Boolean, class of functionalities. Our argument applies for a class of functionalities that are not captured by Theorem 1.3. We next describe this set of functionalities, and claim they cannot be computed with perfect security with correlated randomness.

Lemma 1.6 (Informal)**.** *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{Z}$ be symmetric deterministic two-party functionality. Suppose there exists $\mathcal{X}' \subseteq \mathcal{X}$, $\mathcal{Y}' \subseteq \mathcal{Y}$, and $\mathcal{Z}' \subset \mathcal{Z}$ such that the submatrix $\mathbf{M}'$ of matrix $\mathbf{M}_f$ induced by $\mathcal{X}'$ and $\mathcal{Y}'$ satisfies the following.*

1. *$\mathbf{M}'$ contains an element from $\mathcal{Z} \setminus \mathcal{Z}'$.*
2. *There is a natural $h \geq 1$, such that every row in $\mathbf{M}'$ contains exactly h* distinct *elements from $\mathcal{Z}'$, and every other row in the matrix $\mathbf{M}_f$ associated with f contains at most h distinct elements from $\mathcal{Z}'$, within the columns of $\mathcal{Y}'$.*
3. *There is a natural $h' \geq 1$, such that every column in $\mathbf{M}'$ contains exactly h'* distinct *elements from $\mathcal{Z}'$, and every other column in the matrix $\mathbf{M}_f$ contains at most h' distinct elements from $\mathcal{Z}'$, within the rows of $\mathcal{X}'$.*

Then f cannot be computed with perfect security in the correlated randomness model.

The negative direction of Theorem 1.3 follows from Lemma 1.6 via a combinatorial argument, showing that if such a submatrix does not exist, then f is either spiral or transparent transfer. The proof is somewhat technical and is therefore omitted from the introduction. We refer the reader to Section 5 for the proof.

Let us first describe the attacker. Roughly speaking, the attack follows similar ideas to the attacker for AND, however, instead of biasing towards a specific value, $\mathcal{A}$ will bias the output of the honest party towards the set $\mathcal{Z}' \subset \mathcal{Z}$. In more details, if $\mathcal{A}$ sees that the output z is inside $\mathcal{Z}'$ then it will continue honestly. Otherwise, it will send random messages. Additionally, $\mathcal{A}$ outputs a guess for y that is consistent with the output it saw, i.e., it outputs a uniform y^* conditioned on $f(x, y^*) = z$.

We next show that $\mathcal{A}$ cannot be simulated for $x \leftarrow \mathcal{X}'$ and $y \leftarrow \mathcal{Y}'$. We first analyze the success probability of the adversary in the real world. Let Succ denote the event that $\mathcal{A}$ both guesses y correctly, and causes B to output a value from $\mathcal{Z}'$. We denote by z_{B} the output of B. First, observe that for any fixed $x \in \mathcal{X}'$ it holds that

$$\begin{aligned}
\Pr\left[z_{\mathsf{B}} \in \mathcal{Z}' \wedge y^* = y\right] &= \sum_{z \in \mathcal{Z}'} \Pr\left[z_{\mathsf{B}} = z\right] \cdot \Pr\left[y^* = y \mid z_{\mathsf{B}} = z\right] \\
&= \sum_{z \in \mathcal{Z}'} \frac{|\{y' \in \mathcal{Y}' : f(x, y') = z\}|}{|\mathcal{Y}'|} \cdot \frac{1}{|\{y' \in \mathcal{Y}' : f(x, y') = z\}|} \\
&= \frac{h}{|\mathcal{Y}'|},
\end{aligned}$$

where the last equality follows from Item 2, asserting there are exactly h distinct element from $\mathcal{Z}'$ in the x^{th} row of $\mathbf{M}'$. Therefore

$$\Pr\left[\mathsf{Succ}\right] = \frac{h}{|\mathcal{Y}'|} + \Pr\left[z \notin \mathcal{Z}'\right] \cdot \Pr\left[\mathsf{Succ} \mid z \notin \mathcal{Z}'\right]$$

for every fixed $x \in \mathcal{X}'$. Now, since we assume that $\mathbf{M}'$ contains an element outside of $\mathcal{Z}'$, it follows that there exists a choice of x, for which $\Pr[z \notin \mathcal{Z}'] > 0$. Furthermore, since the output of B is not fixed, there is a non-zero chance that the random messages that $\mathcal{A}$ sends to it will cause it to output a value from $\mathcal{Z}'$. Therefore $\Pr[\mathsf{Succ} \mid z \notin \mathcal{Z}'] > 0$. We conclude that $\Pr[\mathsf{Succ}] > h/|\mathcal{Y}'|$.

To show that $\mathcal{A}$ cannot be simulated, we prove that any simulator can both guess y correctly and cause B to output a value from $\mathcal{Z}'$, with probability at most $h/|\mathcal{Y}'|$. We show that this is true for any input x the simulator sends to the trusted party. Indeed, the probability that B outputs a fixed value $z \in \mathcal{Z}'$ is exactly $\frac{|\{y' \in \mathcal{Y}' : f(x,y')=z\}|}{|\mathcal{Y}'|}$. Given this output z, the simulator can guess y with probability $\frac{1}{|\{y' \in \mathcal{Y}' : f(x,y')=z\}|}$. However, among all the appearances of values from $\mathcal{Z}'$, at most h of them are distinct. Thus, the simulator successfully guesses y correctly and force B to output a value from $\mathcal{Z}'$, with probability at most $h/|\mathcal{Y}'|$.

Impossibility of embedded XOR or embedded AND. To show the usefulness of Lemma 1.6, we next show that if f contains an embedded XOR or an embedded AND, then f cannot be computed with perfect security in the correlated randomness model. In fact, we show that if there exists inputs $x_1, x_2 \in \mathcal{X}$ and $y_1, y_2 \in \mathcal{Y}$, and there exists $a \neq b \in \mathcal{Z}$ such that the 2×2 submatrix $\mathbf{M}$ induced by those inputs is of the form

$$\begin{pmatrix} a & b \\ b & * \end{pmatrix} \quad \text{or} \quad \begin{pmatrix} b & a \\ * & b \end{pmatrix},$$

where $*$ is any element from $\mathcal{Z}$, then f cannot be computed with perfect security in the correlated randomness model. We show that the constraints from Lemma 1.6 hold for $\mathcal{X}' = \{x_1, x_2\}$, $\mathcal{Y}' = \{y_1, y_2\}$, and $\mathcal{Z}' = \{b\}$. Indeed, $\mathbf{M}$ contains the element $a \notin \mathcal{Z}'$, and every row and column in $\mathbf{M}$ contains exactly one (distinct) element from $\mathcal{Z}'$. Finally, any other row or column in the matrix $\mathbf{M}_f$ associated with f, will contain at most one (distinct) element from $\mathcal{Z}'$.

The positive direction. We now turn to prove our positive results. Let us start with describing a protocol for (non-constant) spiral functionalities. Recall that f is said to be spiral, if its associated matrix $\mathbf{M}_f$ or its transpose is, up to permuting the rows and columns, of the form $(\mathbf{M}||\mathbf{M}')$ where the entries of $\mathbf{M}$ and $\mathbf{M}'$ are disjoint, $\mathbf{M}$ is constant, and $\mathbf{M}'$ is spiral. Assume without loss of generality that $\mathbf{M}_f$ is of the form $(\mathbf{M}||\mathbf{M}')$. The idea is to let party B (which is associated with the columns) to send to A the output in case the input y belongs to the columns of $\mathbf{M}$. Otherwise, it sends $\perp$ and the parties inductively compute $\mathbf{M}'$. The security of the protocol stems from the fact that the entries of $\mathbf{M}$ and $\mathbf{M}'$ are disjoint. Thus, the output reveals to A whether y belongs to the columns of $\mathbf{M}$.

We next show that any transparent transfer functionality f can be computed with perfect security. We assume without loss of generality that the associated matrix is

$$\mathbf{M}_f = \begin{pmatrix} 0 & 2 \\ 0 & 3 \\ 1 & 2 \\ 1 & 3 \end{pmatrix}.$$

Consider the protocol, where B sends its input y to A, and then A sends $f(x,y)$ back to B. Clearly the protocol is correct and secure against any corrupt B. We argue that the protocol is secure against any adversary $\mathcal{A}$ corrupting A as well. First, as we are concerned with perfect security if there is no simulator for $\mathcal{A}$, then there exists a fixed choice of the randomness of $\mathcal{A}$ for which no simulator exists. Therefore, we may assume without loss of generality that $\mathcal{A}$ is deterministic.

Let $\mathcal{Y} = \{y_1, y_2\}$ be the domain of B. The idea is to let the simulator query $\mathcal{A}$ on both possible inputs y_1 and y_2, rewinding it each time. This provides the simulator with two outputs $z_1 \in \{0,1\}$ and $z_2 \in \{2,3\}$. Since $\mathbf{M}_f$ contains all possible rows from $\{0,1\} \times \{2,3\}$, the simulator can find an input x^* whose corresponding row is (z_1, z_2). Finally, the simulator sends x^* to the trusted party, and outputs as the view y_1 if the output it received is from $\{0,1\}$, and outputs y_2 otherwise.

1.3 Additional Related Work

In the semi-honest setting, [2] showed that AND is impossible to compute with statistical security, let alone perfect security in the dishonest-majority setting. The work of [6] characterizes the Boolean functionalities that can be computed with dishonest majority.

One of the commonly-known correlated randomness is that of oblivious transfer (OT) which is a pair-wise correlation. In this, the first party gets a pair of inputs (x_0, x_1) and the second party gets (b, x_b). Brassard et al. [3] showed that given sufficiently many invocations of the above OT correlation, the 1-out-of-n string OT functionality can be computed with perfect security against malicious adversaries. Wolf and Wullschleger [11] showed how to compute 1-out-of-2 bit TO perfectly, which is the same as OT where the roles of the parties are reversed. Finally, [1] showed that given access to sufficiently many parallel ideal computations of OT, most sender-receiver functionalities, where the sender's domain size is strictly larger than the receiver's domain size, can be computed with perfect security.

1.4 Organization

The preliminaries and definition of the model of computation appear in Section 2. The statements of our main results are provided in Section 3. The negative direction is proved in Sections 4 and 5. Specifically, in Section 4 we prove the more general impossibility result, and in Section 5 we deduce the result for four-output functionalities. Finally, we prove the positive direction in Section 6.

2 Preliminaries

2.1 Notations

For $n \in \mathbb{N}$ we let $[n] = \{1, 2 \ldots n\}$. For a set $\mathcal{S}$ we write $s \leftarrow \mathcal{S}$ to indicate that s is selected uniformly at random from $\mathcal{S}$. Given a random variable (or a distribution) X, we write $x \leftarrow X$ to indicate that x is selected according to X.

Given a matrix $\mathbf{M}$ whose rows and columns are indexed by $\mathcal{X}$ and $\mathcal{Y}$, respectively, we let $\mathbf{M}(x, \cdot) = (\mathbf{M}(x,y))_{y \in \mathcal{Y}}$ be the x^{th} row, where $x \in \mathcal{X}$. Similarly, we let $\mathbf{M}(\cdot, y) = (\mathbf{M}(x,y))_{x \in \mathcal{X}}$ be the y^{th} column, where $y \in \mathcal{Y}$. We call a matrix $\mathbf{M}$ *constant-row* if for all $x \in \mathcal{X}$ it holds that $\mathbf{M}(x, \cdot)$ is a constant vector. Similarly, we call $\mathbf{M}$ *constant-column* if $\mathbf{M}(\cdot, y)$ is constant for all $y \in \mathcal{Y}$. Given two matrices $\mathbf{M}_1$ and $\mathbf{M}_2$ with the same number of rows, we let $(\mathbf{M}_1 || \mathbf{M}_2)$ denote the matrix obtained from concatenating $\mathbf{M}_1$ and $\mathbf{M}_2$.

The following notion captures when two matrices are the same up to permuting the rows and columns, and transposing either of the matrices.

Definition 2.1. *Let $\mathbf{M}_1 \in \mathcal{Z}^{n_1 \times m_1}$ and $\mathbf{M}_2 \in \mathcal{Z}^{n_2 \times m_2}$ be two matrices. We say that $\mathbf{M}_1 \sim \mathbf{M}_2$ if one of the following holds.*

- *$n_1 = n_2$, $m_1 = m_2$, and there exists a permutation π over the rows of $\mathbf{M}_1$ and a permutation σ over the columns of $\mathbf{M}_1$, such that*

$$\mathbf{M}_1(\pi(x), \sigma(y)) = \mathbf{M}_2(x, y)$$

  for all x and y.

- *$n_1 = m_2$, $m_1 = n_2$, and there exists a permutation π over the rows of $\mathbf{M}_1$ and a permutation σ of $\mathbf{M}_1$ over the columns, such that*

$$\mathbf{M}_1(\pi(x), \sigma(y)) = \mathbf{M}_2^T(y, x)$$

  for all x and y.

We next define the reduced form of matrix, which removes all duplicated rows and columns.

Definition 2.2 (Reduced form of a matrix)**.** *For a matrix $\mathbf{M}$, its reduced form, denoted* $\mathsf{red}(\mathbf{M})$, *is the matrix obtained by* repeatedly *removing all duplicated rows and columns from $\mathbf{M}$ (note that this is well-defined).*

The next definition associates a matrix with any 2-ary function f.

Definition 2.3 (The matrix associated with a function)**.** *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{Z}$ be a 2-ary function. The matrix associated with f, denoted $\mathbf{M}_f \in \mathcal{Z}^{|\mathcal{X}| \times |\mathcal{Y}|}$, is defined as $\mathbf{M}_f(x, y) = f(x, y)$ for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.*

We next define a combinatorial rectangle.

Definition 2.4 (Combinatorial rectangles)**.** *Given two sets $\mathcal{X}$ and $\mathcal{Y}$, a* combinatorial rectangle *(in short,* a rectangle*) over $\mathcal{X} \times \mathcal{Y}$, is a subset $\mathbf{R} = \mathcal{X}_\mathbf{R} \times \mathcal{Y}_\mathbf{R}$, where $\mathcal{X}_\mathbf{R} \subseteq \mathcal{X}$ and $\mathcal{Y}_\mathbf{R} \subseteq \mathcal{Y}$.*

Given a matrix and combinatorial rectangle over its rows and columns, we can define the submatrix induced by the rectangle.

Definition 2.5 (The submatrix induced by a rectangle)**.** *Let $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{Z}$ be three sets, and let $\mathbf{M} \in \mathcal{Z}^{|\mathcal{X}| \times |\mathcal{Y}|}$ be a matrix, whose rows and columns are indexed with elements from $\mathcal{X}$ and $\mathcal{Y}$, respectively. For a combinatorial rectangle $\mathbf{R} = \mathcal{X}_\mathbf{R} \times \mathcal{Y}_\mathbf{R}$ over $\mathcal{X} \times \mathcal{Y}$, we denote by $\mathbf{M}^\mathbf{R} \in \mathcal{Z}^{|\mathcal{X}_\mathbf{R}| \times |\mathcal{Y}_\mathbf{R}|}$ the submatrix of $\mathbf{M}$ induced by $\mathbf{R}$, i.e., $\mathbf{M}^\mathbf{R}(x, y) = \mathbf{M}(x, y)$ for all $x \in \mathcal{X}_\mathbf{R}$ and $y \in \mathcal{Y}_\mathbf{R}$.*

2.2 Security Model

We provide the basic definitions for secure multiparty computation according to the real/ideal paradigm, for further details see [8]. Intuitively, a protocol is considered secure if whatever an adversary can do in the real execution of the protocol, can be done also in an ideal computation, in which an uncorrupted trusted party assists the computation. For concreteness, we present the model and the security definition of perfect two-party computation with an adversary corrupting a single party, as this is the main focus of this work. We refer to [8] for the general definition.

In this paper we focus on deterministic symmetric two-party functionalities $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{Z}$, i.e., both parties receive the same output.[3]

[3]The typical convention in secure computation is to let $f : \{0,1\}^* \times \{0,1\}^* \mapsto \{0,1\}^*$. However, we consider only functionalities with a constant domain, which is why we introduce this notation.

The Real Model

A two-party protocol Π is defined by a set of two interactive Turing machines A and B. Each Turing machine (party) holds at the beginning of the execution a private input, and random coins. The *adversary* $\mathcal{A}$ is an interactive Turing machine describing the behavior of a corrupted party $\mathsf{P} \in \{\mathsf{A}, \mathsf{B}\}$. It starts the execution with input that contains the identity of the corrupted party and its input. We assume the protocol proceeds in round, where every odd round party A sends a message, and every even round party B sends a message.

Throughout the execution of the protocol, all the honest parties follow the instructions of the prescribed protocol, whereas the corrupted party receive its instructions from the adversary. The adversary is considered to be *malicious*, meaning that it can instruct the corrupted party to deviate from the protocol in any arbitrary way. Additionally, the adversary has full-access to the view of the corrupted party, which consists of its input, its random coins, and the messages it sees throughout this execution. At the conclusion of the execution, the honest parties output their prescribed output from the protocol, the corrupted party outputs nothing, and the adversary outputs a function of its view (containing the views of the corrupted party).

We denote by $\text{REAL}_{\Pi,\mathcal{A}}(x, y)$ the joint output of the adversary $\mathcal{A}$ (that may corrupt one of the parties) and of the honest parties in a random execution of Π, on input $x \in \mathcal{X}$ for A and input $y \in \mathcal{Y}$ for B.

Remark 2.6 (On the absence of a security parameter)**.** *Typically, the parties are also given a security parameter 1^κ, which is also used to bound the computational complexity of the parties. However, we are concerned with perfect security and functionalities of constant domain, thus having a security parameter is redundant.*

Additionally, the adversary is usually said to be non-uniform, and holds an auxiliary input. However, as there is no security parameter in our setting, the auxiliary input does not provide $\mathcal{A}$ any additional power.

The Correlated Randomness Hybrid Model

For some of our result, we consider an augmentation of the real world where the parties are provided with a trusted setup for generating correlated randomness. Formally, we let CR denote the randomized functionality that receives no input, and outputs random values r_1 and r_2 to A and B, respectively. Here, $(r_1, r_2) \leftarrow D$ where D is a fixed distribution known in advance. At the start of the protocol (before the parties receive their inputs), the parties call the functionality CR exactly once to obtain r_1 and r_2. The parties then continue in a real execution as described previously. We call this model the CR-hybrid world.

We denote by $\text{REAL}^{\mathsf{CR}}_{\Pi,\mathcal{A}}(x, y)$ the joint output of the adversary $\mathcal{A}$ (that may corrupt one of the parties) and of the honest parties in a random execution of Π in the CR-hybrid world, on input $x \in \mathcal{X}$ for A and input $y \in \mathcal{Y}$ for B.

The Ideal Model

We consider an ideal computation with *guaranteed output delivery* (also referred to as *full security*), where a trusted party performs the computation on behalf of the parties, and the ideal-world adversary *cannot* abort the computation. An ideal computation of a deterministic symmetric two-party functionality $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$, on inputs $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, with an ideal-world adversary $\mathcal{A}$ corrupting a single party $\mathsf{P} \in \{\mathsf{A}, \mathsf{B}\}$ proceeds as follows:

Parties send inputs to the trusted party: Each honest party sends its input to the trusted party. The adversary $\mathcal{A}$ sends a value w from the corrupted party's domain as the input for the corrupted party. Let (x', y') denote the inputs received by the trusted party.

The trusted party performs computation: The trusted party computes $z = f(x', y')$ and sends z to both A and B.

Outputs: Each honest party outputs whatever output it received from the trusted party, and the corrupted party outputs nothing. The adversary $\mathcal{A}$ outputs some function of its view (i.e., the input and output of the corrupted party).

We denote by $\text{IDEAL}_{f,\mathcal{A}}(x, y)$ the joint output of the adversary $\mathcal{A}$ (that may corrupt one of the parties) and the honest parties in a random execution of the ideal-world computation of f on input x for A and input y for B.

The Security Definition

Having defined the real and ideal models, we can now define security of protocols according to the real/ideal paradigm.

Definition 2.7 (Security)**.** *Let $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$ be a deterministic symmetric two-party functionality, and let Π be a two-party protocol. We say that Π computes f with* perfect security*, if for every adversary $\mathcal{A}$, controlling at most one party in the real world, there exists an adversary* Sim*, controlling the same party (if there is any) in the ideal world such that for every $x \in \mathcal{X}$ and every $y \in \mathcal{Y}$ it holds that*

$$\text{IDEAL}_{f,\mathsf{Sim}}(x, y) \equiv \text{REAL}_{\Pi,\mathcal{A}}(x, y).$$

To remove possible confusion, we will explicitly write that Π computes f with perfect security in the plain model.

We say that Π computes f with perfect security in the CR*-hybrid model if*

$$\text{IDEAL}_{f,\mathsf{Sim}}(x, y) \equiv \text{REAL}^{\mathsf{CR}}_{\Pi,\mathcal{A}}(x, y)$$

for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.

The Hybrid Model

The *hybrid model* is a model that extends the real model with a trusted party that provides ideal computation for specific functionalities. The parties communicate with this trusted party in exactly the same way as in the ideal model described above.

Let f be a functionality. Then, an execution of a protocol Π computing a functionality g in the f-hybrid model involves the parties sending normal messages to each other (as in the real model) and, in addition, having access to a trusted party computing f. It is essential that the invocations of f are done sequentially, meaning that before an invocation of f begins, the preceding invocation of f must finish. In particular, there is at most a single call to f per round, and no other messages are sent during any round in which f is called. Note that the CR-hybrid is a special case, where the parties call CR once at the onset of the protocol.

Let $\mathcal{A}$ be an adversary controlling a single party $\mathsf{P} \in \{\mathsf{A}, \mathsf{B}\}$. We denote by $\textsc{hybrid}^{f}_{\Pi,\mathcal{A}}(x, y)$ the random variable consisting of the output of the adversary and the output of the honest parties, following an execution of Π with ideal calls to a trusted party computing f, on input x given to A and input y given to B.

Similarly to Definition 2.7, we say that Π computes g with perfect security in the f-hybrid model if for any adversary $\mathcal{A}$ there exists a simulator Sim such that $\textsc{hybrid}^{f}_{\pi,\mathcal{A}}(x, y)$ and $\textsc{ideal}_{g,\mathsf{Sim}}(x, y)$ are identically distributed.

The sequential composition theorem of Canetti [4] states the following. Let ρ be a protocol that computes f with perfect security. Then, if a protocol Π computes g in the f-hybrid model, then the protocol Π^{ρ}, that is obtained from Π by replacing all ideal calls to the trusted party computing f with the protocol ρ, computes g in the real model with perfect security.

Theorem 2.8 ([4]). *Let f be a two-party functionality, let ρ be a protocol that computes f with perfect security, and let Π be a protocol that computes g with perfect security in the f-hybrid model. Then, protocol Π^{ρ} computes g with perfect security in the real model.*

3 Analyzing Symmetric Functionalities

In this section, we state our results. Our main results is a characterization of the symmetric deterministic two-party functionalities with four-outputs that can be computed with perfect security. Furthermore, the impossibility result can be extended to functionalities with more than four outputs. Interestingly, although the impossibility result holds in the CR-hybrid world, for any choice of CR, the positive results are stated in the *plain model*, where the parties do not receive correlated randomness.

Before stating our results, we first describe three families of symmetric deterministic two-party functionalities. We then assert that among the four-output functionalities, these are the *only* ones that can be computed with perfect security in the CR-hybrid world.

We first define trivial functionalities, for which the output depends on only one of the inputs.

Definition 3.1 (Trivial functionalities). *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{Z}$ be a deterministic symmetric two-party functionality. We say that f is* trivial *if it is independent of one of its inputs, i.e., either $f(x, y) = g(x)$ or $f(x, y) = g(y)$ for some function g.*

Note that for the matrix $\mathbf{M}_f$ of a trivial functionality f, either all rows are constant or all columns are constant.

We next define the family of spiral functionalities, which is an extension of the family of trivial functionalities. The definition is recursive. Roughly, a functionality f is spiral, if it's trivial or if by removing constant columns or constant rows (containing a single value α) from the associated matrix $\mathbf{M}_f$, results in a matrix associated with a spiral functionality, and contains no α values.

Definition 3.2 (The spiral functionality and matrix). *We call a matrix $\mathbf{M}$ a* spiral matrix *if one of the following holds.*

- $\mathbf{M}$ *is a constant matrix.*

- *There exist a constant-column matrix* $\mathbf{M}_1 \in \mathcal{Z}_1^{n_1 \times m_1}$*, and there a spiral matrix* $\mathbf{M}_2 \in \mathcal{Z}_2^{n_2 \times m_2}$*, where* $\mathcal{Z}_1 \cap \mathcal{Z}_2 = \emptyset$*, such that* $\mathbf{M} \sim (\mathbf{M}_1 || \mathbf{M}_2)$*, i.e., equality holds up to permutation of the rows and columns and transposing the matrix.*

We call a deterministic symmetric two-party functionality f a spiral *functionality, if its associated matrix* $\mathbf{M}_f$ *is a spiral matrix.*

Definition 3.3. *Let* $f : \mathcal{X} \times \mathcal{Y} \mapsto \{0, 1, 2, 3\}$ *be a deterministic symmetric two-party functionality. We call it* a transparent transfer *if the reduced form of its associated matrix satisfies*

$$\mathsf{red}\,(\mathbf{M}_f) \sim \begin{pmatrix} a & c \\ a & d \\ b & c \\ b & d \end{pmatrix}$$

where $\{a, b, c, d\} = \{0, 1, 2, 3\}$.

Remark 3.4 (On the naming of the function)**.** *Let us provide the reasoning behind the naming of transparent transfer functions. Consider the symmetric functionality* $f' : \{0,1\}^2 \times \{0,1\} \mapsto \{0,1\}^2$ *defined as* $f'((x_0, x_1), i) = (x_i, i)$*. Observe that, up to the encoding of the output, it is the same as the transparent transfer functionality defined in Definition 3.3. Indeed, mapping the output* $(x_i, i) \mapsto x_i + 2i$ *results in a matrix of the form*

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

Since the mapping is bijective, we conclude the functions to be the equivalent.

3.1 Characterization of Four-Output Functionalities

We are now ready to state our main result, providing a characterization of the symmetric deterministic four-output two-party functionalities that can be computed with perfect security in the CR-hybrid model.

Theorem 3.5 (Characterization of four-output functionalities)**.** *Let* $f : \mathcal{X} \times \mathcal{Y} \mapsto \{0, 1, 2, 3\}$ *be a deterministic symmetric two-party four-output functionality. Then,* f *can be computed with perfect security in the* CR*-hybrid model if and only if it is either a spiral function or a transparent transfer function.*

The proof of Theorem 3.5 follows from the combination of the following two lemmas. For the negative direction, we prove the following.

Lemma 3.6 (Lower bound for four-output functionalities)**.** *Let* $f : \mathcal{X} \times \mathcal{Y} \mapsto \{0, 1, 2, 3\}$ *be a deterministic symmetric two-party four-output functionality. Assume that* f *can be computed with perfect security in the* CR*-hybrid model. Then,* f *is either a spiral function or a transparent transfer function.*

Lemma 3.6 is proved in Section 5. Towards proving it, in Section 4, we prove a more general impossibility result, Lemma 4.1, which holds for functionalities that are not necessarily four-output. When restricting the discussion to four-output functionalities, our general impossibility result yields the lower bound for four-output functionalities, see Section 5 for the full details.

For the positive direction of Theorem 3.5, we prove the following lemma stating that every spiral functionality, and that the transparent transfer functionality can be computed with perfect security. Furthermore, this can be done using deterministic protocols in the plain model, and it holds regardless of the number of outputs.

Lemma 3.7 (Upper bound for four-output functionalities)**.** *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{Z}$ be a deterministic symmetric two-party functionality. If f is a spiral or a transparent transfer functionality, then f can be computed with perfect security in the plain model.*

Lemma 3.7 is proved in Section 6.

3.2 Characterization of Boolean and Ternary-Output Functionalities

When restricting the discussion to functions with range of size two and of size three, Theorem 3.5 yields more refined characterizations. First, note that Definition 3.3 requires at least four distinct output values. It hence follows that a ternary-output functionality can be computed with perfect security if and only if the functionality is a spiral.

Corollary 3.8 (Characterization of ternary-output functionalities)**.** *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \{0, 1, 2\}$ be a deterministic symmetric two-party ternary functionality. Then f can be computed with perfect security in the CR-hybrid model if and only if it is spiral.*

Second, observe that any spiral Boolean functionality must be trivial. Thus, we obtain the following characterization for Boolean functionalities.

Corollary 3.9 (Characterization of Boolean functionalities)**.** *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \{0, 1\}$ be a deterministic symmetric two-party Boolean functionality. Then, f can be computed with perfect security in the CR-hybrid model if and only if it is trivial.*

3.3 Impossibility of Embedded XOR and Embedded AND

In this section we show that any functionality that contains an embedded XOR or an embedded AND cannot be computed with perfect security in the CR-hybrid model. Recall that a functionality f is said to have an embedded XOR, if there exists $x_1, x_2 \in \mathcal{X}$ and $y_1, y_2 \in \mathcal{Y}$ such that $f(x_1, y_1) = f(x_2, y_2) \neq f(x_1, y_2) = f(x_2, y_1)$. The functionality is said to have an embedded AND if $f(x_2, y_2) \neq f(x_1, y_1) = f(x_1, y_2) = f(x_2, y_1)$. In fact, we are able to prove a stronger result. To formalize this, we first define the notion of a forbidden submatrix.

Definition 3.10 (Forbidden 2×2 submatrices and rectangles)**.** *Let $\mathbf{M}$ be a matrix with entries from some set $\mathcal{Z}$. We call a 2×2 rectangle $\mathbf{R}$* forbidden *if its induced submatrix $\mathbf{M}^{\mathbf{R}}$ satisfies*

$$\mathbf{M}^{\mathbf{R}} \sim \begin{pmatrix} a & b \\ b & * \end{pmatrix} \tag{2}$$

where a and b denote distinct elements of $\mathcal{Z}$, and $*$ denotes an arbitrary element of $\mathcal{Z}$. We also say that $\mathbf{M}$ is forbidden if it contains a forbidden combinatorial rectangle.

Theorem 3.11. *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{Z}$ be a deterministic symmetric two-party functionality. Assume there exists a 2×2 rectangle $\mathbf{R}$ such that its corresponding induced submatrix is forbidden. Then f cannot be computed with perfect security in the* CR*-hybrid model.*

The proof is given in Section 5 and is derived from the general impossibility result proven in Section 4. We get the following corollary.

Corollary 3.12. *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{Z}$ be a deterministic symmetric two-party functionality. Assume that $\mathbf{M}_f$ contains an embedded* XOR *or an embedded* AND*. Then f cannot be computed with perfect security in the* CR*-hybrid model.*

4 A General Impossibility Result for Perfect Security

In this section, we prove a general impossibility result for perfectly secure two-party protocols for a large class of functionalities. Roughly speaking, we identify several properties that cannot coincide for any sub-matrix, and show that if the matrix associated with the functionality f contains a sub-matrix that has all these properties, then f cannot be computed with perfect security in the CR-hybrid model.

Lemma 4.1. *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{Z}$ be a deterministic symmetric two-party functionality. Assume there exists a combinatorial rectangle $\mathbf{R} = \mathcal{X}_{\mathbf{R}} \times \mathcal{Y}_{\mathbf{R}}$, where $\mathcal{X}_{\mathbf{R}} \subseteq \mathcal{X}$ and where $\mathcal{Y}_{\mathbf{R}} \subseteq \mathcal{Y}$, and assume there exists a strict subset of the outputs $\mathcal{Z}_{\mathbf{R}} \subset \mathcal{Z}$ such that the following hold.*

1. *At least one entry of $\mathbf{M}_f^{\mathbf{R}}$ (recall that $\mathbf{M}_f^{\mathbf{R}}$ is the sub-matrix induced by $\mathbf{R}$) contains an element from $\mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}$.*
2. *There exists $h \in \mathbb{N}^+$ such that for all $x \in \mathcal{X}_{\mathbf{R}}$ it holds that*
$$\left|\left\{\mathbf{M}_f^{\mathbf{R}}(x, y) : y \in \mathcal{Y}_{\mathbf{R}}\right\} \cap \mathcal{Z}_{\mathbf{R}}\right| = h.$$
In other words, every row in $\mathbf{M}_f^{\mathbf{R}}$ contains exactly h distinct *elements from $\mathcal{Z}_{\mathbf{R}}$. Additionally, for all $x \in \mathcal{X} \setminus \mathcal{X}_{\mathbf{R}}$ it holds that*
$$|\{\mathbf{M}_f(x, y) : y \in \mathcal{Y}_{\mathbf{R}}\} \cap \mathcal{Z}_{\mathbf{R}}| \leq h,$$
namely, every row $x \notin \mathcal{X}_{\mathbf{R}}$ of $\mathbf{M}_f$ contains at most h elements from $\mathcal{Z}_{\mathbf{R}}$, within the columns of $\mathcal{Y}_{\mathbf{R}}$.
3. *There exists $h' \in \mathbb{N}^+$ such that for all $y \in \mathcal{Y}_{\mathbf{R}}$ it holds*
$$\left|\left\{\mathbf{M}_f^{\mathbf{R}}(x, y) : x \in \mathcal{X}_{\mathbf{R}}\right\} \cap \mathcal{Z}_{\mathbf{R}}\right| = h'.$$
Additionally, for all $y \in \mathcal{Y} \setminus \mathcal{Y}_{\mathbf{R}}$ it holds that
$$|\{\mathbf{M}_f(x, y) : x \in \mathcal{X}_{\mathbf{R}}\} \cap \mathcal{Z}_{\mathbf{R}}| \leq h'.$$

Then f cannot be computed in CR*-hybrid model with perfect security.*

Example 4.2. *To illustrate the requirements of Lemma 4.1, consider the ternary-output functionality f whose associated matrix is defined as*

$$\mathbf{M}_f = \begin{pmatrix} 0 & 1 & 2 & 2 \\ 2 & 1 & 0 & 2 \\ 0 & 1 & 2 & 1 \end{pmatrix}.$$

For $\mathbf{R} = \{x_1, x_2\} \times \{y_1, y_2, y_3\}$, $\mathcal{Z}_{\mathbf{R}} = \{0, 1\}$ the condition is satisfied with $h = 2, h' = 1$. Thus, the precondition of Lemma 4.1 is satisfied, hence f cannot be computed with perfect security in the CR*-hybrid model. It also, for example, satisfies the precondition with $\mathbf{R} = \{x_1, x_2\} \times \{y_1, y_3\}$, $\mathcal{Z}_{\mathbf{R}} = \{0\}$, and $h = h' = 1$ (indeed, there is no uniqueness requirement on $\mathbf{R}$).*

Before formally proving Lemma 4.1, let us provide some intuition. First, similarly to the impossibility of XOR due to Ishai et al. [9], we use the fact that any protocol for computing f has a first round i in which (in any honest execution of the protocol) one of the parties, say A, "fully knows" the output, while the other does not. That is, any continuation from round i would result in A outputting the correct output. Conversely, there exists a continuation (and a choice of correlated randomness) forcing B to output a different value.

Recall that the attack of [9] used the existence of such a round to present an attacker that "biases" the output of the honest party. We extend this attack strategy to one, where the adversary, corrupting A, tries to both bias the output of the honest B towards the subset of outputs $\mathcal{Z}_{\mathbf{R}}$, and at the same time, guess the input of the honest party. Additionally, we are more lenient with the knowledge of the adversary, requiring it only to "wait" until it knows whether the output is in $\mathcal{Z}_{\mathbf{R}}$ or not. We show that if the inputs of the parties are chosen independently and uniformly at random from the rectangle $\mathbf{R}$, then the attacker can both guess the input of the honest party correctly, and force it to output a value from $\mathcal{Z}_{\mathbf{R}}$, with probability higher than what any simulator can do in the ideal world. We now provide the formal argument.

Proof of Lemma 4.1. Assume towards contradiction that there exists a protocol Π in the CR-hybrid model computing f with perfect security. Consider an honest execution of Π, where the inputs of A and B are $\tilde{x} \leftarrow \mathcal{X}_{\mathbf{R}}$ and $\tilde{y} \leftarrow \mathcal{Y}_{\mathbf{R}}$, respectively, and are sampled independently. The next claim asserts the existence of a round, in which one of the parties always "knows" if the output is in $\mathcal{Z}_{\mathbf{R}}$ or not, regardless of the choice of the correlated randomness, while the other party does not necessarily "know" this.

Claim 4.3. *There exists a round $i > 0$, and a party* $\mathsf{P} \in \{\mathsf{A}, \mathsf{B}\}$*, such that the following hold.*

1. *For all inputs $x \in \mathcal{X}_{\mathbf{R}}$ and $y \in \mathcal{Y}_{\mathbf{R}}$, and for every possible correlated randomness $(r_1, r_2) \in \operatorname{Supp}(D)$, there exists a set $\mathcal{Z}' \in \{\mathcal{Z}_{\mathbf{R}}, \mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}\}$, such that the following holds. In any execution of Π, where up to (and including) round i, party* A *acts honestly (according to x, r_1) and party* B *acts honestly (according to y, r_2), the output of an honest* P *must be a value from $\mathcal{Z}'$, regardless of the messages it receives in the following rounds (i.e., regardless of the behavior of the other party).*

2. *There exist inputs $x \in \mathcal{X}_{\mathbf{R}}$ and $y \in \mathcal{Y}_{\mathbf{R}}$, with $f(x, y) \in \mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}$, and there exists correlated randomness $(r_1, r_2) \in \operatorname{Supp}(D)$, such that the following holds. Consider an execution of Π, where up to (and including) round i, party* A *acts honestly (according to x, r_1) and party* B *acts honestly (according to y, r_2). Then, there exists a continuation of Π, in which the*

remaining party $\mathsf{P}' \neq \mathsf{P}$ *continues to behave honestly, such that, the output of* P' *is a value from* $\mathcal{Z}_{\mathbf{R}}$*. Specifically, there exists a sequence of messages that* P *can send in the following rounds to cause this effect.*

The proof of the claim is given below. We first use it to conclude the proof of Lemma 4.1. We fix the round i as given by Claim 4.3, and assume without loss of generality that $\mathsf{P} = \mathsf{A}$. The case where $\mathsf{P} = \mathsf{B}$ is handled analogously. We next construct an attacker corrupting A and show that it cannot be simulated in the ideal world.

Define the adversary $\mathcal{A}$ that corrupts A as follows.

1. Given the input $\tilde{x}$ of party A and the randomness r_1 it obtains from CR, the adversary $\mathcal{A}$ emulates A honestly up to and including round i.

2. Consider the lexicographically first honest continuation of the protocol, and let z' denote the resulting output of A in such an execution.

3. If $z' \in \mathcal{Z}_{\mathbf{R}}$, then $\mathcal{A}$ continues to emulate A honestly until the termination of the protocol.

4. Otherwise, if $z' \notin \mathcal{Z}_{\mathbf{R}}$, then in the remaining rounds $\mathcal{A}$ sends (on behalf of A) random messages chosen independently and uniformly at random.

5. The adversary outputs a guess for the input of B (one that is consistent with the output). That is, $\mathcal{A}$ samples
$$y^* \leftarrow \{y \in \mathcal{Y}_{\mathbf{R}} : f(\tilde{x}, y) = z'\},$$
and outputs y^*.

We next prove that $\mathcal{A}$ cannot be simulated, and hence the protocol is not secure. It follows that f cannot be realized with perfect security. We analyze the probability that $\mathcal{A}$ successfully, both guesses the input $\tilde{y}$ of B, and causes B to output a value from $\mathcal{Z}_{\mathbf{R}}$. We then compare this to an arbitrary simulator in the ideal world, showing that no simulator can do the same with exactly the same probability. Formally, we prove the following two claims. In the following, let $\mathsf{Succ}_{\mathrm{REAL}}$ denote the event in the real-world that the output of the adversary is $y^* = \tilde{y}$ and B outputs an element from $\mathcal{Z}_{\mathbf{R}}$. Similarly, let $\mathsf{Succ}_{\mathrm{IDEAL}}$ denote the event in the ideal-world that the output of the simulator implies $y^* = \tilde{y}$ and B outputs an element from $\mathcal{Z}_{\mathbf{R}}$.

The proof (of Lemma 4.1) is concluded from the following two claims (Claims 4.4 and 4.5) that show that $\Pr[\mathsf{Succ}_{\mathrm{REAL}}] > \Pr[\mathsf{Succ}_{\mathrm{IDEAL}}]$, and hence, that $\mathcal{A}$ cannot be simulated for random $(\tilde{x}, \tilde{y}) \leftarrow \mathbf{R}$. Thus, there exists inputs $x \in \mathcal{X}_{\mathbf{R}}$ and $y \in \mathcal{Y}_{\mathbf{R}}$ for which $\mathcal{A}$ cannot be simulated. Therefore, f cannot be computed with perfect security in the CR-hybrid model.

We introduce some notations that will be useful for the following two claims. For every $x \in \mathcal{X}$ and for every $z \in \mathcal{Z}_{\mathbf{R}}$ let
$$w_x(z) := |\{y \in \mathcal{Y}_{\mathbf{R}} : f(x, y) = z\}|$$
denote the number of appearances of z in the x^{th} row of $\mathbf{M}_f$ and the columns corresponding to $\mathcal{Y}_{\mathbf{R}}$. Finally, let
$$W_x := \sum_{z \in \mathcal{Z}_{\mathbf{R}}} w_x(z) = |\{y \in \mathcal{Y}_{\mathbf{R}} : f(x, y) \in \mathcal{Z}_{\mathbf{R}}\}|$$
denote the number of entries from $\mathcal{Z}_{\mathbf{R}}$ in the x^{th} row of $\mathbf{M}_f$ and the columns corresponding to $\mathcal{Y}_{\mathbf{R}}$.

Claim 4.4. *In the real world, it holds that*

$$\Pr\left[\mathsf{Succ}_{\mathrm{REAL}}\right] > \frac{h}{|\mathcal{Y}_{\mathbf{R}}|}.$$

Proof. We next analyze the probability that $\mathsf{Succ}_{\mathrm{REAL}}$ occurs in the real world. Recall that $\tilde{x}$ denotes the input given to A, that z' denotes the prescribed output before the attack, and that y^* is the adversary's guess for the input $\tilde{y}$ held by B. Observe that

$$\begin{aligned}
\Pr\left[\mathsf{Succ}_{\mathrm{REAL}}\right] &= \sum_{x \in \mathcal{X}_{\mathbf{R}}} \Pr\left[\tilde{x} = x\right] \cdot \Pr\left[\mathsf{Succ}_{\mathrm{REAL}} \mid \tilde{x} = x\right] \\
&= \sum_{x \in \mathcal{X}_{\mathbf{R}}} \Pr\left[\tilde{x} = x\right] \cdot \Pr\left[z' \in \mathcal{Z}_{\mathbf{R}} \mid \tilde{x} = x\right] \cdot \Pr\left[\mathsf{Succ}_{\mathrm{REAL}} \mid z' \in \mathcal{Z}_{\mathbf{R}} \wedge \tilde{x} = x\right] \\
&+ \sum_{\substack{x \in \mathcal{X}_{\mathbf{R}}: \\ W_x < |\mathcal{Y}_{\mathbf{R}}|}} \Pr\left[\tilde{x} = x\right] \cdot \Pr\left[z' \notin \mathcal{Z}_{\mathbf{R}} \mid \tilde{x} = x\right] \cdot \Pr\left[\mathsf{Succ}_{\mathrm{REAL}} \mid z' \notin \mathcal{Z}_{\mathbf{R}} \wedge \tilde{x} = x\right] \\
&= \frac{1}{|\mathcal{X}_{\mathbf{R}}|} \cdot \sum_{x \in \mathcal{X}_{\mathbf{R}}} \Pr\left[z' \in \mathcal{Z}_{\mathbf{R}} \mid \tilde{x} = x\right] \cdot \Pr\left[y^* = y \mid z' \in \mathcal{Z}_{\mathbf{R}} \wedge \tilde{x} = x\right] \\
&+ \frac{1}{|\mathcal{X}_{\mathbf{R}}|} \cdot \sum_{\substack{x \in \mathcal{X}_{\mathbf{R}}: \\ W_x < |\mathcal{Y}_{\mathbf{R}}|}} \Pr\left[z' \notin \mathcal{Z}_{\mathbf{R}} \mid \tilde{x} = x\right] \cdot \Pr\left[\mathsf{Succ}_{\mathrm{REAL}} \mid z' \notin \mathcal{Z}_{\mathbf{R}} \wedge \tilde{x} = x\right], \qquad (3)
\end{aligned}$$

where the probabilities are taken over the sampling of the inputs $\tilde{x}$ and $\tilde{y}$, the sampling of the correlated randomness, and the sampling of y^*. The second equality follows from fact that if $W_x < |\mathcal{Y}_{\mathbf{R}}|$, then there exists $y \in \mathcal{Y}_{\mathbf{R}}$ such that $f(x, y) \notin \mathcal{Z}_{\mathbf{R}}$. We now analyze each term in the summation. Observe that for every $x \in \mathcal{X}_{\mathbf{R}}$ it holds that

$$\Pr\left[z' \in \mathcal{Z}_{\mathbf{R}} \mid \tilde{x} = x\right] = \Pr\left[f(\tilde{x}, \tilde{y}) \in \mathcal{Z}_{\mathbf{R}} \mid \tilde{x} = x\right] = \frac{W_x}{|\mathcal{Y}_{\mathbf{R}}|}.$$

Additionally, for every $x \in \mathcal{X}_{\mathbf{R}}$ it holds that

$$\begin{aligned}
\Pr\left[y^* = \tilde{y} | z' \in \mathcal{Z}_{\mathbf{R}} \wedge \tilde{x} = x\right] &= \sum_{\substack{z \in \mathcal{Z}_{\mathbf{R}}: \\ w_x(z) > 0}} \Pr\left[z' = z \mid z' \in \mathcal{Z}_{\mathbf{R}} \wedge \tilde{x} = x\right] \cdot \Pr\left[y^* = \tilde{y} | z' = z \wedge \tilde{x} = x\right] \\
&= \sum_{\substack{z \in \mathcal{Z}_{\mathbf{R}}: \\ w_x(z) > 0}} \frac{w_x(z)}{W_x} \cdot \frac{1}{w_x(z)} \\
&= \frac{h}{W_x}.
\end{aligned}$$

Substituting this into the first summation in Equation (3) for $\frac{W_x}{|\mathcal{Y}_{\mathbf{R}}|} \cdot \frac{h}{W_x}$, we obtain

$$\Pr\left[\mathsf{Succ}_{\mathrm{REAL}}\right] = \frac{h}{|\mathcal{Y}_{\mathbf{R}}|} + \frac{1}{|\mathcal{X}_{\mathbf{R}}|} \cdot \sum_{\substack{x \in \mathcal{X}_{\mathbf{R}}: \\ W_x < |\mathcal{Y}_{\mathbf{R}}|}} \Pr\left[z' \notin \mathcal{Z}_{\mathbf{R}} \mid \tilde{x} = x\right] \cdot \Pr\left[\mathsf{Succ}_{\mathrm{REAL}} \mid z' \notin \mathcal{Z}_{\mathbf{R}} \wedge \tilde{x} = x\right]. \qquad (4)$$

To conclude the proof, it suffices to show that there exists $x \in \mathcal{X}_{\mathbf{R}}$ where $W_x < |\mathcal{Y}_{\mathbf{R}}|$, such that

$$\Pr\left[z' \notin \mathcal{Z}_{\mathbf{R}} \mid \tilde{x} = x\right] \cdot \Pr\left[\mathsf{Succ}_{\mathrm{REAL}} \mid z' \notin \mathcal{Z}_{\mathbf{R}} \wedge \tilde{x} = x\right] \neq 0.$$

Now, for every $x \in \mathcal{X}_{\mathbf{R}}$ where $W_x < |\mathcal{Y}_{\mathbf{R}}|$, let

$$\varepsilon_x := \Pr\left[\mathsf{Succ}_{\mathrm{REAL}} \mid z' \notin \mathcal{Z}_{\mathbf{R}} \wedge \tilde{x} = x\right],$$

where the probability is taken over the sampling of the input $\tilde{y}$, the guess y^* of the adversary, and the sampling of the correlated randomness. Then

$$\Pr\left[z' \notin \mathcal{Z}_{\mathbf{R}} \mid \tilde{x} = x\right] \cdot \Pr\left[\mathsf{Succ}_{\mathrm{REAL}} \mid z' \notin \mathcal{Z}_{\mathbf{R}} \wedge \tilde{x} = x\right] = \left(1 - \frac{W_x}{|\mathcal{Y}_{\mathbf{R}}|}\right) \cdot \varepsilon_x.$$

We now show that there exists $x \in \mathcal{X}_{\mathbf{R}}$ such that $\varepsilon_x > 0$ and that $W_x < |\mathcal{Y}_{\mathbf{R}}|$. By Claim 4.3, there exists inputs $x \in \mathcal{X}_{\mathbf{R}}$ and $y \in \mathcal{Y}_{\mathbf{R}}$, and correlated randomness $(r_1, r_2) \in \mathrm{Supp}(D)$, such that $f(x, y) \in \mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}$, and for which there is a continuation of Π after round i causing B to output a value from $\mathcal{Z}_{\mathbf{R}}$. Observe that since $f(x, y) \in \mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}$, the adversary $\mathcal{A}$ sends messages sampled uniformly at random and independently starting from round i. Therefore, conditioned on $\tilde{x} = x$, the probability it causes B to output a value from $\mathcal{Z}_{\mathbf{R}}$, is the probability that $\tilde{y} = y$, the correlated randomness is (r_1, r_2), and $\mathcal{A}$ sampled the correct messages and guessed $y^* = \tilde{y}$ correctly. By Claim 4.3, this event occurs with non-zero probability, i.e., $\varepsilon_x > 0$. □

Claim 4.5. *For any simulator $\mathsf{Sim}_{\mathcal{A}}$ in the ideal world, it holds that*

$$\Pr\left[\mathsf{Succ}_{\mathrm{IDEAL}}\right] \leq \frac{h}{|\mathcal{Y}_{\mathbf{R}}|}.$$

Proof. We analyze the probability that $\mathsf{Succ}_{\mathrm{IDEAL}}$ occurs in the ideal world, and show that no simulator can both guess $\tilde{y}$ and force B to output $z \in \mathcal{Z}_{\mathbf{R}}$ with probability greater than $h/|\mathcal{Y}_{\mathbf{R}}|$. Fix an ideal world simulator $\mathsf{Sim}_{\mathcal{A}}$. It sends to the trusted party T a random $x^* \in \mathcal{X}$ (not necessarily from only $\mathcal{X}_{\mathbf{R}}$), chosen according to some fixed distribution that depends only on the input $\tilde{x}$. It then receives $f(x^*, \tilde{y})$, where $\tilde{y} \leftarrow \mathcal{Y}_{\mathbf{R}}$ is the input of B. We first show that for any *fixed* $x^* \in \mathcal{X}$, the probability that $\mathsf{Succ}_{\mathrm{IDEAL}}$ occurs is at most $h/|\mathcal{Y}_{\mathbf{R}}|$. Indeed, let y^* denote the output of $\mathsf{Sim}_{\mathcal{A}}$, being its guess of the input $\tilde{y}$ of B. Then for any $x \in \mathcal{X}$ it holds that

$$\begin{aligned}
\Pr\left[\mathsf{Succ}_{\mathrm{IDEAL}} \mid x^* = x\right] &= \Pr\left[f(x, \tilde{y}) \in \mathcal{Z}_{\mathbf{R}} \wedge y^* = \tilde{y} \mid x^* = x\right] \\
&= \Pr\left[f(x, \tilde{y}) \in \mathcal{Z}_{\mathbf{R}} \mid x^* = x\right] \cdot \Pr\left[y^* = \tilde{y} | f(x, \tilde{y}) \in \mathcal{Z}_{\mathbf{R}} \wedge x^* = x\right] \\
&\leq \frac{W_x}{|\mathcal{Y}_{\mathbf{R}}|} \cdot \frac{h}{W_x} \\
&= \frac{h}{|\mathcal{Y}_{\mathbf{R}}|},
\end{aligned}$$

where the inequality is due to the "additionally" party of Item 2.

Thus, for any distribution over x^*, it holds that

$$\Pr\left[\mathsf{Succ}_{\mathrm{IDEAL}}\right] = \sum_{x \in \mathcal{X}} \Pr\left[x^* = x\right] \cdot \Pr\left[\mathsf{Succ}_{\mathrm{IDEAL}} \mid x^* = x\right] \leq \frac{h}{|\mathcal{Y}_{\mathbf{R}}|}.$$

□

It is left to prove Claim 4.3, roughly asserting that there exists a round where the output of one party is fixed, while the output of the other is not.

Proof of Claim 4.3. For any certain inputs $(x, y) \in \mathbf{R}$ and correlated randomness $(r_1, r_2) \in \operatorname{Supp}(D)$, denote as $i_{\mathsf{A}}(x, y, r_1, r_2)$ the first round for which given an honest execution of Π up to and including round $i_{\mathsf{A}}(x, y, r_1, r_2)$, the following holds. There exists a set $\mathcal{Z}' \in \{\mathcal{Z}_{\mathbf{R}}, \mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}\}$ such that any continuation of Π when A continues to behave honestly results in A outputting a value from $\mathcal{Z}'$ (regardless of the behavior of B). Note that such a round exists, since perfect correctness implies that at the end of the protocol, if both parties behave honestly, then for any fixed (x, y, r_1, r_2) party A outputs $f(x, y)$.

Similarly, denote by $i_{\mathsf{B}}(x, y, r_1, r_2)$ the first round for which given an honest execution of Π up to and including round $i_{\mathsf{B}}(x, y, r_1, r_2)$, any continuation of Π when B continues to behave honestly results in it outputting a value from $\mathcal{Z}'' \in \{\mathcal{Z}_{\mathbf{R}}, \mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}\}$ (regardless of the behavior of A).

Next, we let i_{A} be the first round, for which for all inputs $(x, y) \in \mathbf{R}$ and for all possible correlated randomness $(r_1, r_2) \in \operatorname{Supp}(D)$, the output of A is defined to be either in $\mathcal{Z}_{\mathbf{R}}$ or in $\mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}$. In the same way, we define i_{B} for B. Formally,

$$i_{\mathsf{A}} := \max_{\substack{(x,y)\in\mathbf{R} \\ (r_1,r_2)\in\operatorname{Supp}(D)}} i_{\mathsf{A}}(x, y, r_1, r_2), \qquad i_{\mathsf{B}} := \max_{\substack{(x,y)\in\mathbf{R} \\ (r_1,r_2)\in\operatorname{Supp}(D)}} i_{\mathsf{B}}(x, y, r_1, r_2).$$

Observe that there exists a row $x \in \mathcal{X}_{\mathbf{R}}$ and a column $y \in \mathcal{Y}_{\mathbf{R}}$, such that $\mathbf{M}_f^{\mathbf{R}}(x, \cdot)$ and $\mathbf{M}_f^{\mathbf{R}}(\cdot, y)$ contain both values from $\mathcal{Z}_{\mathbf{R}}$ and from $\mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}$. Indeed, by Item 1 (in the statement of Lemma 4.1), there exists a cell $(x, y) \in \mathbf{R}$ where $\mathbf{M}_f^{\mathbf{R}}(x, y) \in \mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}$. By Item 2 and Item 3 it follows that $\mathbf{M}_f^{\mathbf{R}}(x, \cdot)$ and $\mathbf{M}_f^{\mathbf{R}}(\cdot, y)$ contain at least one element from $\mathcal{Z}_{\mathbf{R}}$ each. This implies that $i_{\mathsf{A}}, i_{\mathsf{B}} > 0$, because there exist $(x, y) \in \mathbf{R}$ where the possible output for both parties before the first round can be from $\mathcal{Z}_{\mathbf{R}}$ and from $\mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}$. Furthermore, as the parties send messages one after another, it follows that $i_{\mathsf{A}} \neq i_{\mathsf{B}}$. If $i_{\mathsf{A}} < i_{\mathsf{B}}$ then assign $\mathsf{P} := \mathsf{A}$ and $i := i_{\mathsf{A}}$, else we set $\mathsf{P} := \mathsf{B}$ and $i := i_{\mathsf{B}}$.

The claim follows from the definition of i_{A} and i_{B}. Indeed, assume that $i_{\mathsf{A}} < i_{\mathsf{B}}$, then for all inputs $(x, y) \in \mathbf{R}$ and correlated randomness $(r_1, r_2) \in \operatorname{Supp}(D)$, given an honest execution of Π up to and including round i_{A}, there exists a set $\mathcal{Z}' \in \{\mathcal{Z}_{\mathbf{R}}, \mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}\}$ such that any continuation of i_{A} when A behaves honestly, results in A outputting a value from $\mathcal{Z}'$. On the other hand, as $i_{\mathsf{B}} > i_{\mathsf{A}}$, by the definition of i_{B}, there exist inputs $(x, y) \in \mathbf{R}$ and correlated randomness $(r_1, r_2) \in \operatorname{Supp}(D)$ such that $i_{\mathsf{B}}(x, y, r_1, r_2) > i_{\mathsf{A}}$. Hence, for such (x, y, r_1, r_2) there are possible continuations of Π resulting in honest B outputting a value from $\mathcal{Z}_{\mathbf{R}}$, and continuations resulting in honest B outputting a value from $\mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}$. Furthermore, there exists such $(x, y) \in \mathbf{R}$ satisfying $f(x, y) \in \mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}$ as well. Indeed, observe that since there exists a malicious continuation making B output a value from $\mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}$ (regardless of the value of the real output $f(x, y)$), then there must exist an input $x' \in \mathcal{X}_{\mathbf{R}}$ and randomness r_1' for A that are consistent with the transcript up to round i_{A}, and that cause B to output a value from $\mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}$. Notice that it must be the case that $f(x', y) \in \mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}$ (in addition to $i_{\mathsf{B}}(x', y, r_1', r_2) > i_{\mathsf{A}}$), thus $(x', y) \in \mathbf{R}$ is the desired pair of inputs. The case where $i_{\mathsf{A}} > i_{\mathsf{B}}$ follows an analogous argument. □

□

Remark 4.6 (On proving impossibility of security-with-abort)**.** *Similarly to [9], we can prove that the class of functionality captured by Lemma 4.1 cannot be computed with perfect security-with-abort. To see this, observe that the real world adversary still has a non-zero chance of increasing* *the probability that the honest party outputs a value from $\mathcal{Z}_{\mathbf{R}}$, while in the ideal world, giving the simulator the ability to cause the honest party to output $\perp$ will not increase its success probability.*

5 An Impossibility Result for Perfect Security for Four-Output Functionalities

In this section, we prove Lemma 3.6. Our starting point is the general impossibility result stated in Lemma 4.1, which appears in Section 4. Let us first restate the lemma.

Lemma 5.1 (Restatement of Lemma 3.6)**.** *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \{0, 1, 2, 3\}$ be a deterministic symmetric two-party four-output functionality. Assume that f can be computed with perfect security in the* CR*-hybrid model. Then, f is either a spiral function or a transparent transfer function.*

Towards proving the lemma, we first derive Theorem 3.11 as a corollary from Lemma 4.1.

Corollary 5.2 (Restatement of Theorem 3.11)**.** *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{Z}$ be a deterministic symmetric two-party functionality. Assume there exists a 2×2 rectangle $\mathbf{R}$ such that its corresponding submatrix $\mathbf{M}_f^{\mathbf{R}}$ is forbidden (see Definition 3.10). Then f cannot be computed with perfect security in the* CR*-hybrid model.*

Proof. We show that for $\mathcal{Z}_{\mathbf{R}} = \{b\}$ and $h = h' = 1$, the constraints from Lemma 4.1 hold. Indeed, since $\mathbf{M}_f^{\mathbf{R}}$ contains the element $a \neq b$, Item 1 holds. As for Items 2 and 3, note that $|\mathcal{Z}_{\mathbf{R}}| = h = h' = 1$, and each row and column in $\mathbf{M}_f$ cannot contain more than one distinct elements from $\mathcal{Z}_{\mathbf{R}}$. □

As a corollary, any 2×2 rectangle of a functionality that can be computed with perfect security in the CR-hybrid model, must be one of the remaining forms. That is, we have the following result.

Corollary 5.3. *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{Z}$ be a deterministic symmetric two-party functionality. Suppose that f can be computed with perfect security in the* CR*-hybrid model. Then any 2×2 rectangle $\mathbf{R} \subseteq \mathcal{X} \times \mathcal{Y}$ induces one of the following submatrices:*

$$\mathbf{M}_f^{\mathbf{R}} \sim \begin{pmatrix} a & a \\ a & a \end{pmatrix}; \quad \mathbf{M}_f^{\mathbf{R}} \sim \begin{pmatrix} a & a \\ b & b \end{pmatrix}; \quad \mathbf{M}_f^{\mathbf{R}} \sim \begin{pmatrix} a & a \\ b & c \end{pmatrix}; \quad \mathbf{M}_f^{\mathbf{R}} \sim \begin{pmatrix} a & b \\ c & d \end{pmatrix}, \tag{5}$$

where a, b, c and d denote distinct elements of $\mathcal{Z}$.

We are now ready to prove Lemma 3.6, which gives necessary conditions for a two-party four-output symmetric deterministic functionality $f : \mathcal{X} \times \mathcal{Y} \mapsto \{a, b, c, d\}$, to be computable with perfect security in the CR-hybrid model. Recall that Lemma 3.6 asserts that for such functionalities to be computable with perfect security, they must be one of two types: either a spiral or a transparent transfer functionality. The proof follows from the following two claims, that give the conditions for when a four-output functionalities is a spiral, and when it is a transparent transfer.

Claim 5.4. *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \{0, 1, 2, 3\}$ be a deterministic symmetric four-output two-party functionality. Assume that f can be computed with perfect security in the* CR*-hybrid model. Further, assume that $\mathbf{M}_f$ contains a 2×2 submatrix of the form*

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix},$$

where $\{a, b, c, d\} = \{0, 1, 2, 3\}$. Then f is a transparent transfer functionality.

Claim 5.5. *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \{0, 1, 2, 3\}$ be a deterministic symmetric four-output two-party functionality. Assume that in $\mathbf{M}_f$ there is no forbidden 2×2 submatrix and* no *2×2 submatrix of the form*

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix},$$

where $\{a, b, c, d\} = \{0, 1, 2, 3\}$. Then f is a spiral functionality.

The claims are proven below. We first use them to prove Lemma 3.6.

Proof of Lemma 3.6. Fix a symmetric deterministic functionality $f : \mathcal{X} \times \mathcal{Y} \mapsto \{0, 1, 2, 3\}$, and assume it can be computed with perfect security in the CR-hybrid model. By Corollary 5.3, every 2×2 submatrix of $\mathbf{M}_f$ is of one of the following forms (up to permuting the rows and columns, and transposing the matrix).

$$\begin{pmatrix} a & a \\ a & a \end{pmatrix}; \quad \begin{pmatrix} a & a \\ b & b \end{pmatrix}; \quad \begin{pmatrix} a & a \\ b & c \end{pmatrix}; \quad \text{or} \quad \begin{pmatrix} a & b \\ c & d \end{pmatrix}.$$

If $\mathbf{M}_f$ contains the last submatrix, then by Claim 5.4 the functionality f is the transparent transfer functionality. Otherwise, by Claim 5.5 it is spiral.

□

It is left to prove Claims 5.4 and 5.5.

Proof of Claim 5.4. Suppose there exists a 2×2 rectangle $\mathbf{R} = \{x_1, x_2\} \times \{y_1, y_2\}$ such that

$$\mathbf{M}_f^{\mathbf{R}} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}.$$

Using the notation of Lemma 4.1, consider taking $\mathcal{Z}_{\mathbf{R}} = \{a, d\}$. As each row and each column in $\mathbf{M}_f^{\mathbf{R}}$ contains exactly 1 element from $\mathcal{Z}_{\mathbf{R}}$, and the submatrix contains an element from $\mathcal{Z} \setminus \mathcal{Z}_{\mathbf{R}}$, Item 1 from Lemma 4.1 holds. As we assume that f can be computed with perfect security in the CR-hybrid model, by Lemma 4.1, at least one of the following must hold.

- There exists a row $x_3 \in \mathcal{X} \setminus \{x_1, x_2\}$ such that

  $$\{\mathbf{M}_f(x_3, y_1), \mathbf{M}_f(x_3, y_2)\} = \{a, d\}.$$

  Observe that if $\mathbf{M}_f(x_3, y_1) = d$ and $\mathbf{M}_f(x_3, y_2) = a$, then the submatrix induced by $\{x_1, x_3\} \times \{y_1, y_2\}$ is

  $$\begin{pmatrix} a & b \\ d & a \end{pmatrix}$$

  As this is a forbidden submatrix, by Corollary 5.2 this contradicts the assumption that f can be computed with perfect security in the CR-hybrid model. Thus,

  $$(\mathbf{M}_f(x_3, y_1), \mathbf{M}_f(x_3, y_2)) = (a, d).$$

- There exists a column $y_3 \in \mathcal{Y} \setminus \{y_1, y_2\}$ such that
$$\{\mathbf{M}_f(x_1, y_3), \mathbf{M}_f(x_2, y_3)\} = \{a, d\}\,.$$
Similarly to the previous case, it must be the case where
$$(\mathbf{M}_f(x_1, y_3), \mathbf{M}_f(x_2, y_3)) = (a, d)\,.$$

Taking $\mathcal{Z}_{\mathbf{R}} = \{b, c\}$ and using an analogous argument, it follows that at least one of the following holds.

- There exists a row $x_4 \in \mathcal{X} \setminus \{x_1, x_2\}$ such that
$$(\mathbf{M}_f(x_4, y_1), \mathbf{M}_f(x_4, y_2)) = (c, b)\,.$$
- There exists a column $y_3 \in \mathcal{Y} \setminus \{y_1, y_2\}$ such that
$$(\mathbf{M}_f(x_1, y_4), \mathbf{M}_f(x_2, y_4)) = (b, c)\,.$$

We conclude that one of the following must be a submatrix of $\mathbf{M}_f$.

$$\left(\begin{array}{cc} a & b \\ c & d \\ \hline a & d \\ c & b \end{array}\right); \quad \left(\begin{array}{cc|cc} a & b & a & b \\ c & d & d & c \end{array}\right); \quad \left(\begin{array}{cc|c} a & b & b \\ c & d & c \\ \hline a & d & * \end{array}\right); \quad \left(\begin{array}{cc|c} a & b & a \\ c & d & d \\ \hline c & b & * \end{array}\right), \tag{6}$$

where $*$ is an arbitrary element of $\{a, b, c, d\}$. Next, observe that the latter two cases are impossible. This is true since for any assignment for value of $*$ (out of the four possible values), yields a 2×2 forbidden submatrix (in particular, an embedded AND). Hence, by Corollary 5.2 these two submatrices are forbidden. We assume without loss of generality that the first submatrix from Equation (6) appears in $\mathbf{M}_f$. It is left to show that any other row and column in $\mathbf{M}_f$ is a duplication.

Consider $x_5 \in \mathcal{X} \setminus \{x_1, x_2, x_3, x_4\}$ (assuming such exists). Observe that $\mathbf{M}_f(x_5, y_1) \neq b$ as otherwise, the submatrix induced by $\{x_1, x_5\} \times \{y_1, y_2\}$ is the forbidden submatrix
$$\begin{pmatrix} a & b \\ b & * \end{pmatrix},$$
where $*$ is an arbitrary value. By Corollary 5.2, this contradicts the assumption that f can be computed with perfect security in the CR-hybrid model. Similarly, it holds that $\mathbf{M}_f(x_5, y_1) \neq d$ and $\mathbf{M}_f(x_5, y_2) \notin \{a, c\}$, as otherwise this induces a forbidden submatrix. Therefore, $\mathbf{M}_f(x_5, y_1) \in \{a, c\}$ and $\mathbf{M}_f(x_5, y_2) \in \{b, d\}$. All possible rows satisfying those conditions are $\mathbf{M}_f(x_1, \cdot)$, $\mathbf{M}_f(x_2, \cdot)$, $\mathbf{M}_f(x_3, \cdot)$, and $\mathbf{M}_f(x_4, \cdot)$. Therefore, any possible row $\mathbf{M}_f(x_5, \cdot)$ must be a duplication.

Next, consider a column $y_3 \in \mathcal{X} \setminus \{y_1, y_2\}$. Observe that $\mathbf{M}_f(x_1, y_3) \neq c$, as otherwise the submatrix induced by the rectangle $\{x_1, x_2\} \times \{y_1, y_3\}$ is the forbidden submatrix
$$\begin{pmatrix} a & c \\ c & * \end{pmatrix}.$$

Similarly, it holds that $\mathbf{M}_f(x_1, y_3) \neq d$. We next consider two cases.

We assume that $\mathbf{M}_f(x_1, y_3) = a$, as the case where $\mathbf{M}_f(x_1, y_3) = b$ can be handled using an analogous argument. Then $\mathbf{M}_f(x_3, y_3) = a$, as otherwise the submatrix induced by a rectangle $\{x_1, x_3\} \times \{y_1, y_3\}$ is forbidden. Similarly, note that if $\mathbf{M}_f(x_2, y_3) \neq c$ then $\mathbf{M}_f$ contains an induced forbidden submatrix. Finally, $\mathbf{M}_f(x_4, y_3) = c$, as otherwise the submatrix induced by a rectangle $\{x_2, x_4\} \times \{y_1, y_3\}$ is forbidden. Thus,

$$(\mathbf{M}_f(x_1, y_3), \mathbf{M}_f(x_2, y_3), \mathbf{M}_f(x_3, y_3), \mathbf{M}_f(x_4, y_3)) = (a, c, a, c),$$

which is a duplication of the first column.

□

Proof of Claim 5.5. We prove this by induction on $|\operatorname{Im}(f)| \leq 4$, i.e., the number of possible outputs. We first show that it suffices to prove that $\mathbf{M}_f$ contains a constant row or a constant column. Indeed, assume without loss of generality that $\mathbf{M}_f$ contains the constant row of the value a. Let

$$\mathcal{X}' = \{x \in \mathcal{X} : \forall y \in \mathcal{Y} \text{ such that } \mathbf{M}_f(x, y) = a\}$$

be the set of all rows such that $\mathbf{M}_f(x, \cdot)$ is the constant a-row. Observe that $\mathbf{M}_f(x, y) \neq a$ for all $x \in \mathcal{X} \setminus \mathcal{X}'$ and all $y \in \mathcal{Y}$. Indeed, otherwise for some $x \in \mathcal{X} \setminus \mathcal{X}'$ and $y \in \mathcal{Y}$ it holds that $\mathbf{M}_f(x, y) = a$. Since the row x is not constant, there exists $y' \in \mathcal{Y}$ such that $\mathbf{M}_f(x, y) \neq a$. This, however, is a contradiction as this results in a forbidden submatrix. Thus, we may apply the induction hypothesis to the matrix induced by the rectangle $(\mathcal{X} \setminus \mathcal{X}') \times \mathcal{Y}$, concluding the proof.

It is left to show that either $\mathbf{M}_f$ contains a constant row, or it contains a constant column. Assume towards contradiction there is no constant row nor a constant column. Fix some row $x_1 \in \mathcal{X}$. As it is non-constant there exists $y_1, y_2 \in \mathcal{Y}$ such that $\mathbf{M}_f(x_1, y_1) \neq \mathbf{M}_f(x_1, y_2)$. Now, since y_1 is non-constant as well, there exists $x_2 \in \mathcal{X}$ satisfying $\mathbf{M}_f(x_2, y_1) \neq \mathbf{M}_f(x_1, y_1)$. Observe that $\mathbf{M}_f(x_2, y_1) \neq \mathbf{M}_f(x_1, y_2)$, as otherwise this results in a forbidden matrix. Similarly, either $\mathbf{M}_f(x_2, y_2) = \mathbf{M}_f(x_2, y_1)$ or $\mathbf{M}_f(x_2, y_2) = \mathbf{M}_f(x_1, y_2)$, as otherwise there is a forbidden matrix. It follows the submatrix induced by the rectangle $\mathbf{R} = \{x_1, x_2\} \times \{y_1, y_2\}$ is of the form

$$\begin{pmatrix} a & b \\ c & c \end{pmatrix} \quad \text{or} \quad \begin{pmatrix} a & b \\ c & b \end{pmatrix}$$

where a, b, and c are distinct values from $\{0, 1, 2, 3\}$. Assume without loss of generality that $\mathbf{M}_f^{\mathbf{R}}$ equals the first matrix. Since we assume no row is constant, there exists $y \in \mathcal{Y} \setminus \{y_1, y_2\}$ such that $\mathbf{M}_f(x_2, y) \neq c$. Furthermore, $\mathbf{M}_f(x_2, y) \notin \{a, b\}$ as otherwise this results in a forbidden submatrix. Therefore, it must be the case where $\mathbf{M}_f(x_2, y) = d$.

We now show that $\mathbf{M}_f(x, y) = d$ for all $x \in \mathcal{X}$, contradicting the assumption there is no constant column. First, observe that if $\mathbf{M}_f(x, y) = c$ for some $x \in \mathcal{X} \setminus \{x_2\}$, then $\mathbf{M}_f$ contains a forbidden submatrix. Next, consider $x = x_1$. In this case, if $\mathbf{M}_f(x_1, y) = a$, then the submatrix induced by $\{x_1, x_2\} \times \{y_2, y\}$ equals to

$$\begin{pmatrix} b & a \\ c & d \end{pmatrix},$$

contradicting the assumption that no such submatrix exists. Thus, $\mathbf{M}_f(x_1, y) \neq a$. Similarly, it holds that $\mathbf{M}_f(x_1, y) \neq b$. Therefore, it must be the case where $\mathbf{M}_f(x_1, y) = d$. Finally, if for some $x \in \mathcal{X} \setminus \{x_1, x_2\}$ it holds that $\mathbf{M}_f(x, y) \neq d$, then $\mathbf{M}_f$ contains a forbidden submatrix. Thus, $\mathbf{M}_f(x, y) = d$ for all $x \in \mathcal{X}$. □

6 Positive Results for Perfect Security

In this section, we prove Lemma 3.7, serving as the positive direction of Theorem 3.5. Specifically, we prove that every spiral functionality and the transparent transfer functionality can be computed with perfect security. In fact, we show that these functionalities can be computed by deterministic protocols in the plain model, i.e., where the parties do not receive correlated randomness. We first restate the lemma.

Lemma 6.1 (Restatement of Lemma 3.7)**.** *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{Z}$ be a deterministic symmetric two-party functionality. If f is a spiral or a transparent transfer functionality, then f can be computed with perfect security in the plain model.*

We prove that spiral functionalities can be computed with perfect security in Section 6.1. We handle transparent transfer functionalities in Section 6.2.

6.1 Computing Spiral Functionalities

In this section, we prove that any spiral functionality can be computed with perfect security. This result follows from the following two propositions, asserting the status of a functionality that is obtained from another by adding certain new rows or columns to the associated matrix.

The first proposition states an intuitive observation that for a symmetric functionality f, duplicating rows and columns in $\mathbf{M}_f$ does not affect the existence of a perfectly secure protocol.

Proposition 6.2. *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{Z}$ be a deterministic symmetric two-party functionality, and let $f' : \mathcal{X}' \times \mathcal{Y} \mapsto \mathcal{Z}$ be such that $\mathsf{red}(\mathbf{M}_f) \sim \mathsf{red}(\mathbf{M}_{f'})$. Then f' can be computed with perfect security in the f-hybrid model (and vice versa). Similarly, for $f'' : \mathcal{X} \times \mathcal{Y}'' \mapsto \mathcal{Z}$ such that $\mathsf{red}(\mathbf{M}_f) \sim \mathsf{red}(\mathbf{M}_{f''})$, it holds that f'' can be computed with perfect security in the f-hybrid model (and vice versa).*

Proposition 6.2 is proven below. We next state the second proposition, which asserts that given a functionality f that can be computed with perfect security, adding a constant row or column to $\mathbf{M}_f$ with new values, results in a functionality that can still be computed with perfect security.

Proposition 6.3. *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{Z}$ be a deterministic symmetric two-party functionality, and let $x_+ \notin \mathcal{X}$ and $z_+ \notin \mathcal{Z}$. Consider the functionality $f_+ : (\mathcal{X} \cup \{x_+\}) \times \mathcal{Y} \mapsto \mathcal{Z} \cup \{z_+\}$ defined as*

$$f_+(x, y) = \begin{cases} f(x, y) & \text{if } x \in \mathcal{X} \\ z_+ & \text{otherwise} \end{cases}$$

Then f_+ can be computed with perfect security in the f-hybrid model.

The proof of Proposition 6.3 is given below as well. We first observe that, combined with the composition theorem and the fact that any constant functionality can be computed with perfect security in the plain model, it follows that any spiral functionality can also be computed with perfect security in the plain model.

Corollary 6.4. *Let $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{Z}$ be a deterministic symmetric two-party functionality. Assume that f is spiral. Then f can be computed with perfect security in the plain model.*

It is left to prove Propositions 6.2 and 6.3.

Proof of Proposition 6.2. We prove the claim only for f', as the proof for f'' is analogous. The protocol Π' for computing f' proceeds as follows.

Protocol 6.5.
Inputs: *Party* A *holds input* $x' \in \mathcal{X}'$ *and party* B *holds input* $y \in \mathcal{Y}$.

1. *The parties call* f, *where the input of* A *is the lexicographically first* $x \in \mathcal{X}$ *satisfying* $\mathbf{M}_f(x,\cdot) = \mathbf{M}_{f'}(x',\cdot)$, *and where the input of* B *is* y.

2. *The parties output whatever they receive from* f.

Observe that by the assumption that $\mathsf{red}(\mathbf{M}_f) \sim \mathsf{red}(\mathbf{M}_{f'})$, for every $x' \in \mathcal{X}'$ there exists $x \in \mathcal{X}$ such that $\mathbf{M}_f(x,\cdot) = \mathbf{M}_{f'}(x',\cdot)$. Therefore, the protocol is both well-defined and admits perfect correctness.

We next prove that Π' is secure. We start with proving security against an adversary $\mathcal{A}$ corrupting A. We define the simulator $\mathsf{Sim}_{\mathcal{A}}$ as follows.

1. Query $\mathcal{A}$ for its input x to f.

2. Send to the trusted party T an input $x' \in \mathcal{X}'$ satisfying $\mathbf{M}_f(x,\cdot) = \mathbf{M}_{f'}(x',\cdot)$, and receive an output z.

3. Send z to $\mathcal{A}$, output whatever $\mathcal{A}$ outputs, and halt.

Since such an x' always exists, the output that $\mathcal{A}$ and B receive in both worlds is exactly the same, regardless of the input of B.

We next consider an adversary $\mathcal{B}$ corrupting B. Its simulator $\mathsf{Sim}_{\mathcal{B}}$ simply queries $\mathcal{B}$ for its input to f, send it to T, send the output received from T to $\mathcal{B}$, output whatever it outputs and halt. Clearly, the real and ideal world are exactly the same. Finally, when no party is corrupted, we indeed get an output of $f(x', y)$ for both, due to the choice of x. □

Proof of Proposition 6.3. We next describe a protocol Π_+ in the f-hybrid world for computing f_+ with perfect security.

Protocol 6.6.
Inputs: *Party* A *holds input* $x \in \mathcal{X} \cup \{x_+\}$ *and party* B *holds input* $y \in \mathcal{Y}$.

1. *If* $x = x_+$, *then* A *sends to* B *the value* z_+, *both parties then output* z_+ *and terminate.*

2. *Otherwise, if* $x \neq x_+$ *then* A *sends to* B *the message* $\perp$, *and the parties call* f *with their inputs, output whatever they obtain from* f, *and terminate.*

The protocol is clearly correct. We next prove that Π_+ computes f_+ with perfect security. We start with showing that any adversary $\mathcal{A}$ corrupting A can be simulated. We construct $\mathsf{Sim}_{\mathcal{A}}$ as follows.

1. Query $\mathcal{A}$ for the message z it sends to B.

2. If $z = z_+$, then send x_+ to the trusted party T.

3. Otherwise, query $\mathcal{A}$ for its input x^* used in the call to f, and send it to the trusted party T.

4. Send to $\mathcal{A}$ the output obtained from T, output whatever $\mathcal{A}$ outputs, and halt.

Since $\mathcal{A}$ does not receive any messages in the protocol, its suffices to consider only the output of B. If $\mathcal{A}$ sends $z = z_+$, then the output of B is z_+ in both worlds. Otherwise, it outputs $f(x^*, y)$, where x^* is the input used by $\mathcal{A}$ in the call to f.

Next, we fix an adversary $\mathcal{B}$ corrupting B. We define its simulator $\mathsf{Sim}_{\mathcal{B}}$ as follows.

1. Send $\perp$ to B

2. Query $\mathcal{B}$ for its input y^* it uses in the call to f, and send y^* to the trusted party T.

3. Obtain an output z. If $z \neq z_+$, then output whatever $\mathcal{A}$ outputs and halt.

4. Otherwise, rewind $\mathcal{B}$ and send z_+ to $\mathcal{B}$ as the message in the first round of Π_+, output whatever $\mathcal{B}$ outputs, and halt.

If $x \in \mathcal{X}$, then in both worlds the joint view of the adversary together with the output of A is $(\perp, y^*, f(x, y^*))$. For any $y \in \mathcal{Y}$, and for $x = x_+$, the output of A and the view of the adversary always equals to z_+ both in real and ideal worlds. Thus, the joint view of the adversary together with the output of A in the ideal world is the same as in the real world. □

6.2 Computing Transparent Transfer Functionalities

In this section we show that the transparent transfer functionality defined in Definition 3.3 can be computed with perfect security (in the plain model). In fact, we show a family of functionalities, extending the transparent transfer functionality and show that they can be computed with perfect security. Let us first define this family.

Definition 6.7 (Generalized transparent transfer functionality)**.** *Let $k, n \in \mathbb{N}$ and let $\Sigma = \{0, \ldots, k-1\}$. We define the symmetric* (k, n)-transparent transfer *functionality* $\mathsf{TT}_{k,n} : \Sigma^n \times [n] \mapsto \Sigma \times [n]$ *as*

$$\mathsf{TT}_{k,n}\left((x_1, \ldots, x_n), i\right) = (x_i, i)\,.$$

Note that $\mathsf{TT}_{2,2}$ is equivalent to the transparent transfer functionality from Definition 3.3 (i.e., by applying the mapping $(x_i, i) \mapsto (x_i + (i-1) \cdot k)$ to the output).

We next show that any (k, n)-transparent transfer functionality can be computed with perfect security in the plain model.

Claim 6.8. *For every $k, n \in \mathbb{N}$ the (k, n)-transparent transfer functionality $\mathsf{TT}_{k,n}$ can be computed with perfect security in the plain model.*

Proof. We define a protocol Π for $\mathsf{TT}_{k,n}$ as follows:

Protocol 6.9.
Inputs: *Party A holds input $(x_1, \ldots, x_n) \in \Sigma^n$ and party B holds input $i \in [n]$.*

1. B *sends its input* i *to* A*.*

2. A *sends* x_i *to* B*, and outputs* (x_i, i)*.*

3. If B *received a value* $x_i \notin \Sigma$*, then it outputs* $(0, i)$*.*

Clearly, the protocol admits perfect correctness. We next show that it is secure. Consider an adversary $\mathcal{A}$ corrupting A. We may assume without loss of generality that A is deterministic (by an averaging argument). We define its simulator $\mathsf{Sim}_{\mathcal{A}}$ as follows.

1. Query $\mathcal{A}$ on every possible $j \in [n]$ (rewinding each time). Let $(x_1^*, \ldots, x_n^*)$ be the corresponding messages sent by $\mathcal{A}$ to B for every such j. For any $j \in [n]$, if $x_j^* \notin \Sigma$ then change it to 0. Let $(x_1', \ldots, x_n')$ be the resulting vector.

2. Send $(x_1', \ldots, x_n')$ to the trusted party T, and let (x_i', i) denote the output it sends.

3. Rewind $\mathcal{A}$ to the beginning, send it i, output whatever it outputs, and halt.

We now analyze the simulator. Since $\mathcal{A}$ is deterministic, the input x_i^* it sends upon receiving i, is the same after rewinding. Thus, the simulator will send the same input to the trusted party (changing to 0 in case $x_i^* \notin \Sigma$).

We now consider the case where B is corrupted by an adversary $\mathcal{B}$. Its simulator $\mathsf{Sim}_{\mathcal{B}}$ proceeds as follows.

1. Query $\mathcal{B}$ to obtain the message i it sends to A in the first round.

2. Send i to the trusted party T, and obtain a value x.

3. Send x to $\mathcal{B}$, output whatever $\mathcal{B}$ outputs, and halt.

Clearly, the output of A is i in both worlds, and the view of $\mathcal{B}$ is x_i. Therefore, the real and ideal worlds are identical. □

References

[1] B. Alon and A. Paskin-Cherniavsky. On perfectly secure 2PC in the OT-hybrid model. In *Theory of Cryptography Conference*, pages 561–595. Springer, 2019. doi: 10.1016/j.tcs.2021.08.035.

[2] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computing. In *Proc. of the 20th STOC*, pages 1–10, 1988. doi: 10.1145/3335741.3335756.

[3] G. Brassard, C. Crépeau, and M. Santha. Oblivious transfers and intersecting codes. *IACR Cryptology ePrint Archive*, 1996:10, 1996. URL http://eprint.iacr.org/1996/010.

[4] R. Canetti. Security and composition of multiparty cryptographic protocols. *J. Cryptol.*, 13(1):143–202, 2000. doi: 10.1007/s001459910006. URL https://doi.org/10.1007/s001459910006.

[5] D. Chaum, C. Crépeau, and I. Damgard. Multiparty unconditionally secure protocols. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 11–19, 1988. doi: 10.1145/62212.62214.

[6] B. Chor and E. Kushilevitz. A zero-one law for boolean privacy. *SIAM Journal on Discrete Mathematics*, 4(1):36–47, 1991.

[7] R. Cleve. Limits on the security of coin flips when half the processors are faulty. In *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 364–369, 1986.

[8] O. Goldreich. *Foundations of Cryptography – VOLUME 2: Basic Applications.* Cambridge University Press, 2004.

[9] Y. Ishai, E. Kushilevitz, S. Meldgaard, C. Orlandi, and A. Paskin-Cherniavsky. On the power of correlated randomness in secure computation. In *Theory of Cryptography Conference*, pages 600–620. Springer, 2013. doi: 10.1007/978-3-642-36594-2_34.

[10] E. Kushilevitz. Privacy and communication complexity. *SIAM Journal on Discrete Mathematics*, 5(2):273–284, 1992.

[11] S. Wolf and J. Wullschleger. Oblivious transfer is symmetric. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 222–232. Springer, 2006.