On the Structure of the Privacy Hierarchy*

Benny Chor† Mihály Geréb-Graus‡ Eyal Kushilevitz§

Abstract

An N argument function $f(x_1, \ldots, x_N)$ is called t – private if there exists a protocol for computing f so that no coalition of at most t parties can infer any additional information from the execution, other than the value of the function. The motivation of this work is to understand what levels of privacy are attainable. So far, only two levels of privacy were known for N argument functions which are defined over finite domains: Functions that are N – private, and functions that are $\lfloor (N-1)/2 \rfloor$ – private but not $\lceil N/2 \rceil$ – private.

In this work, we show that the *privacy hierarchy* for N-argument functions which are defined over finite domains, has exactly $\lceil (N+1)/2 \rceil$ levels. We prove this by constructing, for any $\lceil N/2 \rceil \leq t \leq N-2$, an N-argument function which is t – private but not $t+1$ – private.

Keywords: private functions, privacy hierarchy, distributed computing.

1 Introduction

An N-argument function $f(x_1, \ldots, x_N)$ is called t – *private* if there exists a protocol for distributively computing f, so that no coalition of at most t parties can infer any *additional information* from the execution of the protocol. By "additional information" we mean any information, in the information-theoretic sense, on inputs of non-coalition members which does not follow from inputs of coalition members and the value of the function $f(x_1, \ldots, x_N)$. Ben-Or, Goldwasser and Wigderson [1] and Chaum, Crepeau and Damgard [5] have shown that over *finite* domains, every function can be computed $\lfloor (N-1)/2 \rfloor$ – privately. Some functions, like modular addition [3], are even N – private, while others, like Boolean OR, are $\lfloor (N-1)/2 \rfloor$ – private but not $\lceil N/2 \rceil$ – private [1].

These two levels of privacy raise the question whether functions which are t – private but not $t+1$ – private, for $\lceil N/2 \rceil \leq t \leq N-2$, exist. For certain infinite families of functions

*Research supported by US-Israel Binational Science Foundation grant 88-00282.

†e-mail: benny@cs.technion.ac.il . Department of Computer Science, Technion, Haifa 32000, Israel.

‡e-mail: gereb@cs.tufts.edu . Department of Computer Science, Tufts University, Medford, MA 02155.

§e-mail: eyalk@techunix.technion.ac.il . Department of Computer Science, Technion, Haifa 32000, Israel. Current address: Aiken Computation Lab., Harvard University, Cambridge, MA 02138, USA

the answer for this question is negative. Chor and Kushilevitz [7] proved that every *Boolean* function which is $\lceil N/2 \rceil$ – private is also N-private. Chor and Shani [8] proved a similar result for a class of symmetric functions. No function which is t – private but not $t+1$ – private, for $\lceil N/2 \rceil \le t \le N-2$, was known to date. In this paper, we show that this "gap" between $\lfloor (N-1)/2 \rfloor$ – privacy and N – privacy is a property of specific families of functions, and is not true in general. Specifically, we show that for every $\lceil N/2 \rceil \le t \le N-2$ there exists a function that is t – private but not $t+1$ – private. This proves the existence of a "dense" privacy hierarchy, with no gaps in it.

2 Definitions and Background

In this section we describe the model of communication, give the formal definition of privacy, and state two known lemmata which are used in the sequel. The system we consider is a distributed network of N synchronous, computationally unbounded parties $P_1, P_2, \ldots, P_N$. Each pair of parties is connected by a secure (no eavesdropping) and reliable communication channel.

At the beginning of an execution, each party P_i has an input x_i. In addition, each party can flip unbiased and independent random bits. (As usual, more general sources of randomness could also be used without seriously effecting the capabilities of the model.)

We denote by r_i the string of random bits flipped by P_i (sometimes we refer to the string r_i as the *random input* of P_i). The parties wish to compute the value of a function $f(x_1, x_2, \ldots, x_N)$. To this end, they exchange messages as prescribed by a protocol $\mathcal{F}$. Messages are sent in rounds, where in each round every processor can send a message to every other processor. Each message a party sends in the k-th round is determined by its input, its random input, the messages it received during the first $k-1$ rounds, and the identity of the receiver. We say that the protocol $\mathcal{F}$ computes the function f if last message in the protocol is an identical message sent by P_1 to all other parties, and consists of the value $f(x_1, x_2, \ldots, x_N)$.

Definition 1*:* *Let $\mathcal{F}$ be an N party protocol, as described above. The* communication $S(\vec{x}, \vec{r})$ *sent in an execution of $\mathcal{F}$ is the concatenation of all messages sent in the execution, parsed according to sender, receiver, and round number.*

Definition 2*:* *Given a protocol $\mathcal{F}$, a* communication string *S is a string parsed according to sender, receiver, and round number, which equals $S(\vec{x}, \vec{r})$ for some input $\vec{x}$ and random input $\vec{r}$. Let S be a communication string, and let $T \subseteq \{1, 2, \ldots, N\}$. The* projected communication string*, S_T, is the communication string S after the deletion of messages sent between parties in $\bar{T}$.*

Intuitively, S_T is the view of the members of T of the communication string S.

Definition 3*:* *Let $\mathcal{F}$ be an N party protocol which computes a function f, and let T be a coalition of parties, $T \subseteq \{1, 2, \ldots, N\}$. We say that the coalition T* does not learn any additional information *from the execution of $\mathcal{F}$ if the following holds: For every two input vectors $\vec{x}$ and*

$\vec{y}$ that agree in their T entries (i.e. $\forall i \in T : x_i = y_i$) and for which f has the same value $f(\vec{x}) = f(\vec{y})$, for every choice of random inputs $\{r_i\}_{i\in T}$, and for every projected communication string S_T

$$Pr_{\{r_i\}_{i\in \bar{T}}}(S_T|\vec{x}, \{r_i\}_{i\in T}) = Pr_{\{r_i\}_{i\in \bar{T}}}(S_T|\vec{y}, \{r_i\}_{i\in T}) \ .$$

(The probability space is over the random inputs of all parties in $\bar{T}$.)

This definition implies that for all inputs which "look the same" from the coalition's point of view (and for which, in particular, f has the same value), the communication exchanged between T and $\bar{T}$ also "look the same" (it is identically distributed). Therefore, by executing $\mathcal{F}$, the coalition T cannot infer any information on the inputs of $\bar{T}$ (other than what follows from the inputs of T and the value of the function).

Definition 4*: A protocol $\mathcal{F}$ for computing f is* t-private *if any coalition T of at most t parties does not learn any additional information from the execution of the protocol. A function f is* t-private *if there exists a t-private protocol that computes it.*

In the proofs that follow, we will use two known lemmata of Chor and Kushilevitz [7, 9]. The first lemma states a necessary condition for t – privacy ($t \geq \lceil N/2 \rceil$) of f, in terms of 1 – privacy of a related two-argument function. The second lemma states a necessary condition for 1 – privacy of two-argument functions.

The Partition Lemma [7]: Let $A_1, A_2, \ldots, A_N$ and B be non-empty sets, $t \geq \lceil N/2 \rceil$, and $f : A_1 \times A_2 \times \ldots \times A_N \rightarrow B$ be t – private. Let $S \subseteq \{1, 2, \ldots, N\}$ be any subset of size t. Denote by D (resp. E) the Cartesian product of the A_i with $i \in S$ (resp. $i \in \bar{S}$). Let f' be the function obtained by viewing f as a two argument function $f' : D \times E \rightarrow B$. That is, f' satisfies $f(x_1, x_2, \ldots, x_N) = f'(\{x_i\}_{i\in D}, \{x_i\}_{i\in E})$. In this setting, if f is t – private, then f' is 1–private.

The Corners Lemma [7, 9]: Let D, E and B be non-empty sets, and $f : D \times E \rightarrow B$ be 1 – private. For every $d_1, d_2 \in D$, $e_1, e_2 \in E$ and $b \in B$, if $f(d_1, e_1) = f(d_1, e_2) = f(d_2, e_1) = b$ then $f(d_2, e_2) = b$.

3 The Hierarchy

Theorem 1*: Let t be an integer in the interval $\lceil N/2 \rceil \leq t \leq N-2$. There exists an N-argument function f_t which is t – private but not $t+1$ – private.*

Proof: For every t ($\lceil N/2 \rceil \leq t \leq N-2$), let $f_t : \{0,1\}^N \rightarrow \{0,1\}^{t+2} \cup \{0,1\}$ be defined by

$$f_t(x_1, x_2, \ldots, x_N) \triangleq \begin{cases} 0 & \text{if } x_i = 0 \text{ for all } 1 \leq i \leq t+1 \\ x_{t+2} & \text{if } x_i = 1 \text{ for all } 1 \leq i \leq t+1 \\ (x_1, \ldots, x_{t+2}) & \text{otherwise} \end{cases}$$

Note that the function f_t depends only on its first $t+2$ arguments.

	$\overbrace{0, 0, \ldots, 0}^{N-t-1}$	$\overbrace{1, 1, \ldots, 1}^{N-t-1}$
$\underbrace{0, 0, \ldots, 0}_{t+1}$	0	0
$\underbrace{1, 1, \ldots, 1}_{t+1}$	0	1

Figure 1: g_t does not satisfy the Corners Lemma

First we show that f_t is not $t+1$-private. By the Partition Lemma it is enough to demonstrate a partition $S, \bar{S}$ of $\{1, \ldots, N\}$ such that S is of size $t+1$, and the induced two-argument function is not 1-private. We choose $S = \{1, 2, \ldots, t+1\}$, so that $\bar{S} = \{t+2, \ldots, N\}$. ($t$ should satisfy $t \leq N-2$ for $\bar{S}$ to be non empty, and $t \geq \lceil N/2 \rceil$ for the Partition Lemma to be applicable.) In Figure 1 we show four points, where the rows correspond to $x_1, \ldots, x_{t+1}$, and the columns to $x_{t+2}, \ldots, x_N$ (see Figure 1). It is clear that the induced two-argument function, g_t, does not satisfy the Corners Lemma. Therefore g_t is not 1 – private, and thus f_t is not $t+1$ – private.

Now we show that f_t is t-private. We present an appropriate protocol, $\mathcal{F}_t$ and prove that it is a t – private protocol.

1. Party P_{t+2} chooses at random $t+1$ bits $m_1, \ldots, m_{t+1}$ such that $\sum_{i=1}^{t+1} m_i = x_{t+2} \bmod 2$ (each such $t+1$ tuple is chosen with the same probability). P_{t+2} sends m_i to P_i ($1 \leq i \leq t+1$). (The effect of this step is that the party P_{t+2} shares its input, x_{t+2}, among the parties $P_1, P_2, \ldots, P_{t+1}$ using a $t+1$ out of $t+1$ *secret-sharing scheme* [10, 4].) This ensures that $P_1, P_2, \ldots, P_{t+1}$ together can reconstruct x_{t+2}, while any subset of them does not have any information about x_{t+2}.

2. Each party among $P_1, P_2, \ldots, P_{t+1}$ sends its input to all other parties in this list.

3. If $x_1 = x_2 = \ldots = x_{t+1} = 0$, then the parties $P_1, P_2, \ldots, P_{t+1}$ announce that the output is 0 (*i.e.* $f_t(\vec{x}) = 0$), and the protocol terminates.

4. If $x_1 = x_2 = \ldots = x_{t+1} = 1$, then the parties $P_1, P_2, \ldots, P_{t+1}$ reconstruct x_{t+2} by computing $x_{t+2} = \sum_{i=1}^{t+1} m_i \bmod 2$. Party P_1 announces that the output is x_{t+2}, and the protocol terminates.

5. Otherwise, $P_1, P_2, \ldots, P_{t+1}$ reconstruct x_{t+2} by computing $x_{t+2} = \sum_{i=1}^{t+1} m_i \bmod 2$. P_1 announces that the output is $(x_1, \ldots, x_{t+2})$, and the protocol terminates.

We now prove that the protocol $\mathcal{F}_t$ is indeed t – private. The parties $P_{t+3}, \ldots, P_N$ are not active in the protocol. The following claim says that we can ignore these passive parties while proving the t – privacy and consider only coalitions which are subsets of $P_1, \ldots, P_{t+2}$.

Claim 1*:* *Let $T_1 \subseteq \{1, \ldots, t+2\}$ and $T_2 \subseteq \{t+3, \ldots, N\}$. If the coalition T_1 does not learn any additional information from the execution of the protocol $\mathcal{F}_t$, then neither does the coalition $T_1 \cup T_2$.*

Proof: Observe that in every execution, the parties in T_2 receive only the final message from P_1 containing the output of the protocol (and send no messages). This implies that for every communication string S, the projected communication string with respect to $T_1 \cup T_2$, $S_{T_1 \cup T_2}$, equals the projected communication string with respect to T_1, S_{T_1}, together with those messages containing the output. Therefore, the claim follows from Definition 3. ♣

The next claim says that the the arguments of $P_{t+3}, \ldots, P_N$ have no influence on the communication.

Claim 2*:* *For any two input vectors $\vec{x}$ and $\vec{y}$ that agree on the first $t+2$ arguments, the communication in the protocol $\mathcal{F}_t$ is distributed in the same way.*

Proof: As the parties $P_{t+3}, \ldots, P_N$ do not send or receive any messages (except receiving the final output), then for every choice of random inputs for all parties, $\vec{r}$, we have $S(\vec{x}, \vec{r}) = S(\vec{y}, \vec{r})$. ♣

Pairs of inputs $\vec{x}$ and $\vec{y}$ for which $f_t(\vec{x}) \neq f_t(\vec{y})$ can always be distinguished by any coalition. Indeed, there are privacy requirements only with respect to pairs of inputs $\vec{x}$ and $\vec{y}$ satisfying $f_t(\vec{x}) = f_t(\vec{y})$. Therefore, it is convenient to break the proof of the t – privacy to cases, by the output values. Any output different from $\{0, 1\}$ completely determines the input values of the active parties $P_1, \ldots, P_{t+2}$. Therefore, by the definition, the privacy requirements are always met on these inputs. This leaves us with inputs $\vec{x}$ for which $f_t(\vec{x}) = 0$ or $f_t(\vec{x}) = 1$. If the output is 1, the input vector is of the form

$$\vec{x} = (\underbrace{1, 1, \ldots, 1}_{t+2}, x_{t+3}, \ldots, x_N).$$

All such $\vec{x}$'s agree on the first $t+2$ arguments, which are all 1's. By Claim 2, for each of the possible input vectors of this form we have the same distribution of communications. In particular, for any coalition $T \subseteq \{1, \ldots, t+2\}$, S_T is identically distributed for all these inputs. Therefore the privacy requirements are satisfied for inputs with $f(\vec{x}) = 1$. The remaining case is where the output is 0. This corresponds to inputs $\vec{x}$ of the two forms

$$\vec{x} = (\underbrace{0, 0, \ldots, 0}_{t+1}, x_{t+2}, x_{t+3}, \ldots, x_N) \quad \text{and} \quad \vec{x} = (\underbrace{1, 1, \ldots, 1}_{t+1}, 0, x_{t+3}, \ldots, x_N).$$

There are three possibilities for coalitions of size at most t which contain only active parties (subsets of $P_1, \ldots, P_{t+2}$).

- Coalitions of size at most t which are (non-empty) subsets of $\{P_1, P_2, \ldots, P_{t+1}\}$.
- The coalition $\{P_{t+2}\}$.

- Coalitions that consist of P_{t+2} and at least one of $P_1, \ldots, P_{t+1}$.

1. Coalitions T of size at most t which are (non-empty) subsets of $\{P_1, P_2, \ldots, P_{t+1}\}$. Such a coalition does not contain P_{t+2}, and should be unable to distinguish between any pair of input vectors of the form
$$\vec{x} = (\underbrace{0, 0, \ldots, 0}_{t+1}, 0, x_{t+3}, \ldots, x_N)$$
and
$$\vec{y} = (\underbrace{0, 0, \ldots, 0}_{t+1}, 1, x_{t+3}, \ldots, x_N).$$
By the way that $m_1, \ldots, m_{t+1}$ are chosen, every proper subset of $P_1, \ldots, P_{t+1}$, and in particular T, sees the same distribution of messages, in Step (1) of the protocol, for $\vec{x}$ and $\vec{y}$. Steps (2) and (3) of the protocol are identical for $\vec{x}$ and $\vec{y}$, and in this case the protocol terminates in step (3). (The parties $P_1, \ldots, P_{t+1}$ do not reconstruct the input x_{t+2}.) Therefore, the distribution of the projected communication string with respect to T, S_T, is identical for $\vec{x}$ and $\vec{y}$.

2. The coalition $\{P_{t+2}\}$. This coalition should not be able to distinguish between input vectors of the form
$$\vec{x} = (\underbrace{0, 0, \ldots, 0}_{t+1}, 0, x_{t+3}, \ldots, x_N)$$
and
$$\vec{y} = (\underbrace{1, 1, \ldots, 1}_{t+1}, 0, x_{t+3}, \ldots, x_N).$$
In this case, the party P_{t+2}, does not receive any message during the execution of the protocol (except the output of the function). The coalition contains no active party among the first $t+1$ parties $P_1, \ldots, P_{t+1}$, and so the distribution of the projected communication string $S_{\{P_{t+2}\}}$ that P_{t+2} receives and sends on $\vec{x}$ and $\vec{y}$ is identical.

3. Coalitions T that consist of P_{t+2}, and at least one of $P_1, \ldots, P_{t+1}$, a party we will denote by P_i $(1 \leq i \leq t+1)$. It is enough to show that for any two inputs $\vec{x}$ and $\vec{y}$ satisfying $f_t(\vec{x}) = f_t(\vec{y}) = 0$, $x_{t+2} = y_{t+2}$, and $x_i = y_i$, the projected communication string S_T is identically distributed for $\vec{x}$ and $\vec{y}$. By the definition of f_t, if $f_t(\vec{x}) = 0$, then $x_1 = x_2 = \ldots = x_{t+1}$. So in our case, since $x_i = y_i$, we have $x_1 = x_2 = \ldots = x_{t+1} = y_1 = y_2 = \ldots = y_{t+1}$, and in addition $x_{t+2} = y_{t+2}$. In other words, the inputs $\vec{x}$ and $\vec{y}$ agree on their first $t+2$ arguments. By Claim 2, S_T is identically distributed for $\vec{x}$ and $\vec{y}$.

This completes the proof of the theorem. $\square$

Combining Theorem 1 with the $\lfloor (N-1)/2 \rfloor$ – private protocols of [1, 5] we get

Corollary 2: *The privacy hierarchy of functions defined over finite domains consists of exactly* $\lceil (N+1)/2 \rceil$ *(non-empty) levels, which correspond to* $\lfloor (N-1)/2 \rfloor, \lfloor (N-1)/2 \rfloor + 1, \ldots, N-2$, *and* N *privacy.*

We remark that by the definition of privacy, an $N-1$ – private function is also N – private, so there is no additional level in the privacy hierarchy.

4 Concluding Remarks

In proving that f_t is not $t+1$ – private, we used a partition argument (the Partition Lemma). We demonstrated a partition of $\{1, 2, \ldots, N\}$ into sets $S, \bar{S}$ with $|S| = t+1$, such that the induced two argument function is not 1 – private (by the Corners Lemma). All known proofs of non t – privacy for functions with finite domain and t in the range $\lceil N/2 \rceil \leq t \leq N-1$ are based on a similar partition argument, together with either the Corners Lemma or the two party characterization of [9, 2]. It is an open problem whether such an argument always suffices; that is, whether non t – privacy can always be proved by a partition argument.

It would be interesting to know the situation with respect to functions defined over *infinite* domains. Clearly, the privacy hierarchy for the infinite case contains at least as many levels as the privacy hierarchy in the finite case. However, in the infinite case the hierarchy contains at least one more level: The authors, in [6], proved that there are functions (over countable domains) which are not even 1 – private. The existence of functions, over infinite countable domains, which are t – private but not $t+1$ – private, for $1 \leq t < \lfloor (N-1)/2 \rfloor$, remains an open problem.

Acknowledgments

We wish to thank the anonymous referee for speed and clarity.

References

[1] Ben-or M., S. Goldwasser, and A. Wigderson, "Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation" *Proc. of 20th STOC*, 1988, pp. 1-10.

[2] Beaver, D., "Perfect Privacy for Two Party Protocols", Technical Report TR-11-89, Harvard University, 1989.

[3] Benaloh (Cohen), J.D., "Secret Sharing Homomorphisms: Keeping Shares of a Secret Secret", *Advances in Cryptography - Crypto86 (proceedings)*, A.M. Odlyzko (ed.), Springer-Verlag, Lecture Notes in Computer Science, Vol. 263, pp. 251-260, 1987.

[4] Blakley, G.R., "Safeguarding Cryptographic Keys", *Proc. NCC AFIPS 1979* , pp. 313-317, 1979.

[5] Chaum, D., C. Crepeau, and I. Damgard, "Multiparty Unconditionally Secure Protocols" *Proc. of 20th STOC*, 1988, pp. 11-19.

[6] Chor, B., M. Geréb-Graus, and E. Kushilevitz, "Private Computations Over the Integers", *31th IEEE Conference on the Foundations of Computer Science*, October 1990, pp. 335-344.

[7] Chor, B., and E. Kushilevitz, "A Zero-One Law for Boolean Privacy", *SIAM J. Discrete Math.*, Vol 4, No 1, 1991, pp. 36-47. Early version in *Proc. of 21th STOC*, 1989, pp. 62-72.

[8] Chor, B., and N. Shani, "Privacy of Dense Symmetric Functions", *Sequences II, Proceedings of the 2nd International Workshop on Sequences*, R. Capocelli, A. De Santis, and U. Vaccaro, eds., June 1991, pp. 345–359.

[9] Kushilevitz, E., "Privacy and Communication Complexity", *SIAM Jour. on Disc. Math.* Vol. 5, No. 2, May 1992, pp. 273-284. Early version in *30th IEEE Conference on the Foundations of Computer Science*, 1989, pp. 416–421.

[10] Shamir, A., "How to Share a Secret", *Comm. ACM* , Vol. 22, 1979, pp. 612-613.