Synthesis with Privacy Against an Observer*

Orna Kupferman, Ofer Leshkowitz(✉), and Naama Shamash Halevy(✉)

School of Computer Science and Engineering, The Hebrew University, Jerusalem, Israel

ofer.leshkowitz@mail.huji.ac.il

Abstract. We study automatic *synthesis* of systems that interact with their environment and maintain *privacy* against an observer to the interaction. The system and the environment interact via sets I and O of input and output signals. The input to the synthesis problem contains, in addition to a specification, also a list of *secrets*, a function $\mathsf{cost} : I \cup O \to \mathbb{N}$, which maps each signal to the cost of hiding it, and a bound $b \in \mathbb{N}$ on the budget that the system may use for hiding of signals. The desired output is an (I/O)-transducer $\mathcal{T}$ and a set $\mathcal{H} \subseteq I \cup O$ of signals that respects the bound on the budget, thus $\sum_{s \in \mathcal{H}} \mathsf{cost}(s) \leq b$, such that for every possible interaction of $\mathcal{T}$, the generated computation satisfies the specification, yet an observer from which the signals in $\mathcal{H}$ are hidden, cannot evaluate the secrets.

We first show that the complexity of the problem is 2EXPTIME-complete for specifications and secrets in LTL, thus it is not harder than synthesis with no privacy requirements. We then analyze the complexity of the problem more carefully, isolating the two aspects that do not exist in traditional synthesis, namely the need to hide the value of the secrets and the need to choose the set $\mathcal{H}$. We do this by studying settings in which traditional synthesis can be solved in polynomial time – when the specification formalism is deterministic automata and when the system is closed, and show that each of the two aspects involves an exponential blow-up in the complexity. We continue and study *bounded synthesis with privacy*, where the input also includes a bound on the size of the synthesized transducer, as well as a variant of the problem in which the observer has *knowledge about the specification*, which can be helpful in evaluating the secrets. We study the effect of both variants on the different aspects of the problem and provide algorithms with a tight complexity.

1 Introduction

Synthesis is the automated construction of correct systems from their specifications [2]. While synthesized systems are correct, there is no guarantee about their *quality*. Since designers will be willing to give up manual design only after being convinced that the automatic process replacing it generates systems of comparable quality, it is extremely important to develop and study quality measures for automatically-synthesized systems. An important quality measure is *privacy*:

* This research is supported by the Israel Science Foundation, Grant 2357/19, and the European Research Council, Advanced Grant ADVANSYNT.

N. Kobayashi and J. Worrell (Eds.): FoSSaCS 2024, LNCS 14574, pp. 256–277, 2024.
https://doi.org/10.1007/978-3-031-57228-9_13

making sure that the system and its environment do not reveal information they prefer to keep private. Privacy is a vivid research area in Theoretical Computer Science. There, the notion of *differential privacy* is used for formalizing when an algorithm maintains privacy. Essentially, an algorithm is differentially private if by observing its output, one cannot tell if a particular individual's information is used in the computation [9,11]. Another related notion is *obfuscation* in system development, where we aim to develop systems whose internal operation is hidden [1,15]. Obfuscation has been mainly studied in the context of software, where it has exciting connections with cryptography [1,15].

In the setting of automated synthesis in formal methods, a very basic notion of privacy has been studied by means of synthesis with *incomplete information* [27,21,7]. There, the system should satisfy its specification eventhough it only has a partial view of the environment. Lifting differential privacy to formal methods, researchers have introduced the temporal logic *HyperLTL*, which extends LTL with explicit trace quantification [8]. Such a quantification can relate computations that differ only in non-observable elements, and can be used for specifying that computations with the same observable input have the same observable output. The synthesis problem of HyperLTL is undecidable, yet is decidable for the fragment with a single existential quantifier, which can specify interesting properties [13]. In [18], the authors suggested a general framework for automated synthesis of privacy-preserving reactive systems. In their framework, the input to the synthesis problem includes, in addition to the specification, also *secrets*. During its interaction with the environment, the system may keep private some of the assignments to the output signals, and it directs the environment which assignments to the input signals it should keep private. Consequently, the satisfaction value of the specification and secrets may become unknown. The goal is to synthesize a system that satisfies the specification yet keeps the value of the secrets unknown. Finally, lifting obfuscation to formal methods, researchers have studied the synthesis of obfuscation policies for temporal specifications. In [32], an obfuscation mechanism is based on edit functions that alter the output of the system, aiming to make it impossible for an observer to distinguish between secret and non-secret behaviors. In [10], the goal is to synthesize a control function that directs the user which actions to disable, so that the observed sequence of actions would not disclose a secret behavior.

In this paper we continue to study privacy-preserving reactive synthesis. As in [18], our setting is based on augmenting the specification with secrets whose satisfaction value should remain unknown. Unlike [18], the system and the environment have complete information about the assignments to the input and output signals, and the goal is to hide the secrets from a third party, and to do so by hiding the assignment to some of the signals throughout the interaction. As an example, consider a system that directs a robot patrolling a warehouse storage. Typical specifications for the system require it to direct the robot so that it eventually reaches the shelves of requested items, it never runs out of energy, etc. An observer to the interaction between the system and the robot may infer properties we may want to keep private, like dependencies between

customers and shelves visited, locations of battery docking stations, etc. If we want to prevent the observer from inferring these properties (a.k.a,. the secrets), we have to hide the interaction from it. Different effort should be made in order to hide different components of the interaction (alarm sound, content of shelves, etc.). Our framework synthesizes a system that realizes the specification without the secrets being revealed, subject to restrictions on hiding of signals. As another example, consider a scheduler that should grant access to a joint resource. The scheduler should maintain mutual exclusion (grants are not given to different users simultaneously) and non-starvation (all requests are granted), while hiding details like waiting time or priority to specific users. In Examples 1 and 2, we describe in detail the application of our framework for the synthesis of such a scheduler, as well as its application in the synthesis of a robot that paints parts of manufactured pieces. The robot should satisfy some requirements about the generated pattern of colors while hiding other features of the pattern.

Formally, we consider a reactive system that interacts with its environments via sets I and O of input and output signals. At each moment in time, the system reads a truth assignment, generated by the environment, to the signals in I, and it generates a truth assignment to the signals in O. The interaction between the system and its environment generates a *computation*. The system *realizes* a specification φ if all its computations satisfy φ [25]. We introduce and study the problem of *synthesis with privacy in the presence of an observer*. Given a specification φ, and secrets $\psi_1, \ldots, \psi_k$ over $I \cup O$, our goal is to return, in addition to a system that realizes the specification φ, also a set $\mathcal{H} \subseteq I \cup O$ of *hidden signals*, such that the satisfaction value of the secrets $\psi_1, \ldots, \psi_k$ is unknown to an observer that does not know the truth values of the signals in $\mathcal{H}$. Thus, secrets are evaluated according to a *three-valued semantics*. The use of secrets enables us to hide *behaviors*, rather than just signals. [1] Obviously, hiding all signals guarantees that the satisfaction value of every secret is unknown. Hiding of signals, however, is not always possible or involves some cost. We formalize this by adding to the setting a function $\mathsf{cost} : I \cup O \to \mathbb{N}$, which maps each signal to the cost of hiding its value, and a bound $b \in \mathbb{N}$ on the budget that the system may use for hiding of signals. The set $\mathcal{H}$ of hidden signals has to respect the bound, thus $\sum_{s \in \mathcal{H}} \mathsf{cost}(s) \leq b$.

In some cases, it is desirable to hide the truth value of a secret only when some condition holds. For example, we may require to hide the content of selves only in some sections of the warehouse. We extend our framework to *conditional secrets*: pairs of the form $\langle \theta, \psi \rangle$, where the satisfaction value of the secret ψ should be hidden from the observer only when the trigger θ holds. In particular, when $\theta = \psi$, we require to hide the secret only when it holds. For example, we may require to hide an unfair scheduling policy only when it is applied. Note that a conditional secret $\langle \theta, \psi \rangle$ is not equivalent to a secret $\theta \rightarrow \psi$ or $\theta \rightarrow \neg\psi$, and that the synthesized system may violate the trigger, circumventing the need

[1] Hiding of signals is a special case of our framework. Specifically, hiding of a signal p can be done with the secrets Fp and $F\neg p$.

to hide the secret. For example, by synthesizing a fair scheduler, the designer circumvents the need to hide an unfair policy.

We show that synthesis with privacy is 2EXPTIME-complete for specifications and secrets in LTL. Essentially, once the set $\mathcal{H}$ of hidden signals is determined, we can compose an automaton for the specification with automata that verify, for each secret, that the assignments to the signals in $(I \cup O) \setminus \mathcal{H}$ can be completed both in a way that satisfies the secret and in a way that does not satisfy it. A similar algorithm works for conditional secrets.

While the complexity of our algorithm is not higher than that of LTL synthesis with no privacy, it would be misleading to conclude that handling of privacy involves no increase in the complexity. The 2EXPTIME complexity follows from the need to translate LTL specifications to deterministic automata on infinite words. Such a translation involves a doubly-exponential blow-up [22,20], which possibly dominates other computational tasks of the algorithm. In particular, two aspects of synthesis with privacy that do not exist in usual synthesis are a need to go over all possible choices of signals to hide, and a need to go over all assignments to the hidden signals.

Our main technical contribution is a finer complexity analysis of the problem, which reveals that each of the two aspects above involves an exponential complexity: the first in the number of signals and the second in the size of the secret. We start with the need to go over all assignments of hidden signals and show that even when the specification is $\mathtt{T}$, the set $\mathcal{H}$ of hidden signals is given, and there is only one secret, given by a deterministic Büchi automaton, synthesis with privacy is EXPTIME-complete. This is exponentially higher than synthesis of deterministic Büchi automata, which can be solved in polynomial time. We continue to the need to go over all possible choices of $\mathcal{H}$. For that, we focus on the closed setting, namely when $I = \varnothing$, and the case the specification and secrets are given by deterministic automata. We show that while synthesis with privacy can be then solved in polynomial time for a given set $\mathcal{H}$, it is NP-complete when $\mathcal{H}$ is not given, even when the function cost is uniform.

We continue and study two variants of the problem: *bounded synthesis* and *knowledgeable observer*. One way for coping with the 2EXPTIME complexity of LTL synthesis, which is carried over to a doubly-exponential lower bound on the size of the generated system [28], is bounded synthesis. There, the input to the problem includes also a bound on the size of the system [30,12,19]. In a setting with no privacy, the bound reduces the complexity of LTL synthesis to PSPACE, as one can go over all candidate systems. We study bounded synthesis with privacy and show that privacy makes the problem much harder: it is EXPSPACE-complete when the specification and secrets are given by LTL formulas, and is PSPACE-complete when they are given by deterministic parity (or Büchi) automata.

Finally, recall that a system keeps a secret ψ private if an observer cannot reveal the truth value of ψ: every observable computation can be completed both to a computation that satisfies ψ and to a computation that does not satisfy ψ. We study a setting in which the observer knows the specification φ of the system.

Consequently, the observer knows that only completions that satisfy φ should be taken into account. If, for example, $\varphi \rightarrow \psi$, then ψ cannot be kept private. We describe an algorithm for this variant of the problem and analyze the way knowledge of the specification influences the complexity. In particular, we show that the problem becomes EXPTIME-complete even when the specification is given by a deterministic Büchi automaton and the secrets are of a fixed size.

Due to the lack of space, some examples and proofs are omitted and can be found in the full version, in the authors' URLs.

2 Preliminaries

2.1 Synthesis

For a finite nonempty alphabet Σ, an infinite *word* $w = \sigma_0 \cdot \sigma_1 \cdot \ldots \in \Sigma^\omega$ is an infinite sequence of letters from Σ. A *language* $L \subseteq \Sigma^\omega$ is a set of infinite words.

Let I and O be disjoint finite sets of input and output signals, respectively. We consider the alphabet $2^{I \cup O}$ of truth assignments to the signals in $I \cup O$. Then, a languages $L \subseteq (2^{I \cup O})^\omega$ can be viewed as a *specification*, and the *truth value* of L in a computation $w \in (2^{I \cup O})^\omega$ is `T` if $w \in L$, and is `F` otherwise.

An (I/O)-*transducer* is a tuple $\mathcal{T} = \langle I, O, S, s_0, \eta, \tau \rangle$, where S is a finite set of states, $s_0 \in S$ is an initial state, $\eta : S \times 2^I \rightarrow S$ is a transition function, and $\tau : S \rightarrow 2^O$ is a labeling function. We extend the transition function η to words in $(2^I)^*$ in the expected way, thus $\eta^* : S \times (2^I)^* \rightarrow S$ is such that for all $s \in S$, $x_I \in (2^I)^*$, and $i \in 2^I$, we have that $\eta^*(s, \epsilon) = s$, and $\eta^*(s, x_I \cdot i) = \eta(\eta^*(s, x_I), i)$. For a word $w_I = i_0 \cdot i_1 \cdot i_2 \cdot \ldots \in (2^I)^\omega$, we define the *computation of* $\mathcal{T}$ *on* w_I to be the word $\mathcal{T}(w_I) = (i_0 \cup o_0) \cdot (i_1 \cup o_1) \cdot \ldots \in (2^{I \cup O})^\omega$, where for all $j \geq 0$, we have that $o_j = \tau(\eta^*(s_0, i_0 \cdots i_j))$. The *language* of $\mathcal{T}$, denoted $L(\mathcal{T})$, is the set of computations of $\mathcal{T}$, that is $L(\mathcal{T}) = \{\mathcal{T}(w_I) : w_I \in (2^I)^\omega\}$.

We say that $\mathcal{T}$ *realizes* a language $L \subseteq (2^{I \cup O})^\omega$ if $L(\mathcal{T}) \subseteq L$. We say that a language $L \subseteq (2^{I \cup O})^\omega$ is *realizable* if there is an (I/O)-transducer that realizes it. In the *synthesis* problem, we are given a specification language $L \subseteq (2^{I \cup O})^\omega$ and we have to return an (I/O)-transducer that realizes L or decide that L is not realizable. The language L is given by an automaton over the alphabet $2^{I \cup O}$ or a temporal logic formula over $I \cup O$ (see definitions in Section 2.4).

2.2 Synthesis with privacy

In the *synthesis with privacy* problem, we are given, in addition to the specification language $L_\varphi \subseteq (2^{I \cup O})^\omega$, also a *secret* $L_\psi \subseteq (2^{I \cup O})^\omega$, which defines a behavior that we want to hide from an observer[2]. Thus, we seek an (I/O)-transducer that realizes L_φ without revealing the truth value of L_ψ in the generated computations. Keeping the truth value of L_ψ secret is done by hiding the truth value of some signals in $I \cup O$. Before we define synthesis with privacy formally, we first need some notations.

[2] See Remark 2.3 for an extension of the setting to multiple and conditional secrets.

Consider a set $\mathcal{H} \subseteq I \cup O$ of *hidden signals*. Let $\mathcal{V} = (I \cup O) \setminus \mathcal{H}$ denote the set of *visible signals*. For an assignment $\sigma \in 2^{I \cup O}$, let $\mathsf{hide}_{\mathcal{H}}(\sigma) \in 2^{\mathcal{V}}$ be the restriction of σ to the visible signals. That is, $\mathsf{hide}_{\mathcal{H}}(\sigma)(v) = \sigma(v)$ for all $v \in \mathcal{V}$. Also, let $\mathsf{noise}_{\mathcal{H}}(\sigma) \subseteq 2^{I \cup O}$ be the set of assignments that differ from σ in assignments to the signals in $\mathcal{H}$. Thus, $\mathsf{noise}_{\mathcal{H}}(\sigma) = \{\sigma' \in 2^{I \cup O} : \sigma \cap \mathcal{V} = \sigma' \cap \mathcal{V}\}$. Then, for an infinite computation $w = \sigma_0 \cdot \sigma_1 \cdots \in (2^{I \cup O})^\omega$, we have that $\mathsf{hide}_{\mathcal{H}}(w) = \mathsf{hide}_{\mathcal{H}}(\sigma_0) \cdot \mathsf{hide}_{\mathcal{H}}(\sigma_1) \cdots \in (2^{\mathcal{V}})^\omega$ and $\mathsf{noise}_{\mathcal{H}}(w)$ is the set of all computations that differ from w in assignments to the signals in $\mathcal{H}$. Formally, $\sigma'_0 \cdot \sigma'_1 \cdots \in \mathsf{noise}_{\mathcal{H}}(w)$ iff $\sigma'_i \in \mathsf{noise}_{\mathcal{H}}(\sigma_i)$ for all $i \geq 0$. Note that for all $w, w' \in (2^{I \cup O})^\omega$, it holds that $w' \in \mathsf{noise}_{\mathcal{H}}(w)$ iff $w \in \mathsf{noise}_{\mathcal{H}}(w')$ iff $\mathsf{hide}_{\mathcal{H}}(w') = \mathsf{hide}_{\mathcal{H}}(w)$, and that $w \in \mathsf{noise}_{\mathcal{H}}(w)$ for all $w \in (2^{I \cup O})^\omega$ and $\mathcal{H} \subseteq I \cup O$. Intuitively, when the signals in $\mathcal{H}$ are hidden, then an observer of a computation $w \in (2^{I \cup O})^\omega$ only knows that the computation is in $\mathsf{noise}_{\mathcal{H}}(w)$.

Consider a specification $L_\varphi \subseteq (2^{I \cup O})^\omega$ and a secret $L_\psi \subseteq (2^{I \cup O})^\omega$. For a set $\mathcal{H} \subseteq I \cup O$ of hidden signals, we say that an (I/O)-transducer $\mathcal{T}$ *$\mathcal{H}$-hides* L_ψ if for all words $w_I \in (2^I)^\omega$, the truth value of the secret L_ψ in the computation $\mathcal{T}(w_I)$ cannot be deduced from $\mathsf{hide}_{\mathcal{H}}(\mathcal{T}(w_I))$. Formally, for every $w_I \in (2^I)^\omega$, there exist two computations $w^+, w^- \in \mathsf{noise}_{\mathcal{H}}(\mathcal{T}(w_I))$, such that $w^+ \in L_\psi$ and $w^- \notin L_\psi$. We say that $\mathcal{T}$ *realizes* $\langle L_\varphi, L_\psi, \mathcal{H} \rangle$ *with privacy* if $\mathcal{T}$ realizes L_φ and $\mathcal{H}$-hides L_ψ. We say that $\langle L_\varphi, L_\psi, \mathcal{H} \rangle$ is *realizable with privacy* if there exists an (I/O)-transducer that realizes $\langle L_\varphi, L_\psi, \mathcal{H} \rangle$ with privacy.

Clearly, hiding is monotone with respect to $\mathcal{H}$, in the sense that the larger $\mathcal{H}$ is, the more likely it is for an (I/O)-transducer $\mathcal{T}$ to $\mathcal{H}$-hide L_ψ. Indeed, if $\mathcal{T}$ $\mathcal{H}$-hides L_ψ, then $\mathcal{T}$ $\mathcal{H}'$-hides L_ψ for all $\mathcal{H}'$ with $\mathcal{H} \subseteq \mathcal{H}'$. In particular, taking $\mathcal{H} = I \cup O$, we can hide all non-trivial secrets. Hiding of signals, however, is not always possible, and may sometimes involve a cost. Formally, we consider a *hiding cost function* $\mathsf{cost} : I \cup O \to \mathbb{N}$, which maps each signal to the cost of hiding it, and a *hiding budget* $b \in \mathbb{N}$, which bounds the cost that the system may use for hiding of signals. The cost of hiding a set $\mathcal{H} \subseteq I \cup O$ of signals is then $\mathsf{cost}(\mathcal{H}) = \sum_{p \in \mathcal{H}} \mathsf{cost}(p)$, and we say that $\mathcal{H}$ *respects* b if $\mathsf{cost}(\mathcal{H}) \leq b$. Note that if $\mathsf{cost}(p) > b$, for $p \in I \cup O$, then p cannot be hidden. Also, when $\mathsf{cost}(p) = 1$ for all $p \in I \cup O$, we say that cost is *uniform*. Note that then, b bounds the number of signals we may hide.

Now, we say that $\langle L_\varphi, L_\psi, \mathsf{cost}, b \rangle$ is *realizable with privacy* if there exists a set $\mathcal{H} \subseteq I \cup O$ such that $\mathcal{H}$ respects b and $\langle L_\varphi, L_\psi, \mathcal{H} \rangle$ is realizable with privacy. Finally, in the *synthesis with privacy* problem, we are given L_φ, L_ψ, cost, and b, and we have to return a set $\mathcal{H} \subseteq I \cup O$ that $\mathcal{H}$ respects b and an (I/O)-transducer $\mathcal{T}$ that realizes $\langle L_\varphi, L_\psi, \mathcal{H} \rangle$ with privacy, or determine that $\langle L_\varphi, L_\psi, \mathsf{cost}, b \rangle$ is not realizable with privacy.

2.3 Multiple and conditional secrets

In this section we discuss two natural extensions of our setting. First, often we need to hide form the observer more than one secret. We extend the definition of synthesis with privacy to a set of secrets $S = \{L_{\psi_1}, L_{\psi_2}, \ldots, L_{\psi_k}\}$ in the natural

way. Thus, an (I/O)-transducer $\mathcal{T}$ realizes $\langle \varphi, S, \mathcal{H} \rangle$ with privacy if it realizes φ and $\mathcal{H}$-hides L_{ψ_i}, for all $i \in [k]$. Note that

Then, a *conditional secret* is a pair $\langle L_\theta, L_\psi \rangle$, consisting of a *trigger* and a *secret*. The truth value of the secret should be unknown only in computations that satisfy the trigger. Formally, for a set $\mathcal{H} \subseteq I \cup O$ of hidden signals, we say that an (I/O)-transducer $\mathcal{T}$ $\mathcal{H}$-*hides* $\langle L_\theta, L_\psi \rangle$ if for all input sequences $w_I \in (2^I)^\omega$ such that $\mathsf{noise}_\mathcal{H}(\mathcal{T}(w_I)) \subseteq L_\theta$, the truth value of L_ψ in the computation $\mathcal{T}(w_I)$ cannot be deduced from $\mathsf{hide}_\mathcal{H}(\mathcal{T}(w_I))$, thus there exist two computations $w^+, w^- \in \mathsf{noise}_\mathcal{H}(\mathcal{T}(w_I))$, such that $w^+ \in L_\psi$ and $w^- \notin L_\psi$. A useful special case of conditional secrets is when the trigger and the secret coincide, and so we have to hide the truth value of the secret only if there are computations where the value of secret is $\mathtt{T}$. Formally, $\mathcal{T}$ $\mathcal{H}$-hides $\langle L_\psi, L_\psi \rangle$ if for all input sequences $w_I \in (2^I)^\omega$, there exists a computation $w^- \in \mathsf{noise}_\mathcal{H}(\mathcal{T}(w_I))$ such that $w^- \notin L_\psi$.

Note that unlike a collection of specifications, which can be conjuncted, hiding a set of secrets is not equivalent to hiding their conjunction. Likewise, hiding a conditional secret is not equivalent to hiding the implication of the secret by the trigger. Thus, the two variants require an extension of the solution for the case of a single or unconditional secret. In Remark 2, we describe such an extension.

2.4 Automata and LTL

An *automaton* on infinite words is $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, \alpha \rangle$, where Σ is an alphabet, Q is a finite set of *states*, $q_0 \in Q$ is an *initial state*, $\delta : Q \times \Sigma \to 2^Q$ is a *transition function*, and α is an *acceptance condition*, to be defined below. For states $q, s \in Q$ and a letter $\sigma \in \Sigma$, we say that s is a σ-successor of q if $s \in \delta(q, \sigma)$. Note that we do not require the transition function to be *total*. That is, we allow that $\delta(q, \sigma) = \varnothing$. If $|\delta(q, \sigma)| \leq 1$ for every state $q \in Q$ and letter $\sigma \in \Sigma$, then $\mathcal{A}$ is *deterministic*. For a deterministic automaton $\mathcal{A}$ we view δ as a function $\delta : Q \times \Sigma \to Q \cup \{\bot\}$, where $\bot$ is a distinguished symbol, and instead of writing $\delta(q, \sigma) = \{q\}$ and $\delta(q, \sigma) = \varnothing$, we write $\delta(q, \sigma) = q$ and $\delta(q, \sigma) = \bot$, respectively.

A *run* of $\mathcal{A}$ on $w = \sigma_0 \cdot \sigma_1 \cdots \in \Sigma^\omega$ is an infinite sequence of states $r = r_0 \cdot r_1 \cdot r_2 \cdot \ldots \in Q^\omega$, such that $r_0 = q_0$, and for all $i \geq 0$, we have that $r_{i+1} \in \delta(r_i, \sigma_i)$. The acceptance condition α determines which runs are "good". We consider here the *Büchi*, *co-Büchi*, *generalized Büchi* and *parity* acceptance conditions. All conditions refer to the set $inf(r) \subseteq Q$ of states that r traverses infinitely often. Formally, $inf(r) = \{q \in Q : q = r_i \text{ for infinitely many } i\text{'s}\}$. In generalized Büchi the acceptance condition is of the form $\alpha = \{\alpha_1, \alpha_2, \ldots, \alpha_k\}$, for $k \geq 1$ and sets $\alpha_i \subseteq Q$. In a generalized Büchi automaton, a run r is accepting if for all $1 \leq i \leq k$, we have that $inf(r) \cap \alpha_i \neq \varnothing$. Thus, r visits each of the sets in α infinitely often. Büchi automata is a special case of its generalized form with $k = 1$. That is, a run r is accepting with respect to the Büchi condition $\alpha \subseteq Q$, if $inf(r) \cap \alpha \neq \varnothing$. Dually, in co-Büchi automata, a run r is accepting if $inf(r) \cap \alpha = \varnothing$. Finally, in a parity automaton, the acceptance condition $\alpha : Q \to \{1, ..., k\}$, for some $k \geq 1$, maps states to ranks, and a run r is accepting if the maximal rank of a state in $inf(r)$ is even. Formally, $\max_{q \in inf(r)}\{\alpha(q)\}$ is even. A run that is not accepting is *rejecting*. We refer to the number k in α as

the *index* of the automaton. A word w is accepted by $\mathcal{A}$ if there is an accepting run of $\mathcal{A}$ on w. The language of $\mathcal{A}$, denoted $L(\mathcal{A})$, is the set of words that $\mathcal{A}$ accepts. Two automata are *equivalent* if their languages are equivalent.

We denote the different classes of automata by three-letter acronyms in $\{\text{D,N}\} \times \{\text{B,C,GB,P}\} \times \{\text{W}\}$. The first letter stands for the branching mode of the automaton (deterministic, nondeterministic); the second for the acceptance condition type (Büchi, co-Büchi, generalized Büchi, or parity); and the third indicates we consider automata on words. For example, NBWs are nondeterministic Büchi word automata.

LTL is a linear temporal logic used for specifying on-going behaviors of reactive systems [24]. Specifying the behavior of (I/O)-transducers, formulas of LTL are defined over the set $I \cup O$ of signals using the usual Boolean operators and the temporal operators G ("always") and F ("eventually"), X ("next time") and U ("until"). The semantics of LTL is defined with respect to infinite computations in $(2^{I \cup O})^\omega$. Thus, each LTL formula φ over $I \cup O$ induces a language $L_\varphi \subseteq (2^{I \cup O})^\omega$ of all computations that satisfy φ.

Recall that the input to the synthesis with privacy problem includes languages L_φ and L_ψ. We sometimes replace L_φ and L_ψ in the different notations with automata or LTL formulas that describe them, thus talk about realizability with privacy of $\langle \mathcal{A}_\varphi, \mathcal{A}_\psi, \mathcal{H} \rangle$ or $\langle \varphi, \psi, \mathcal{H} \rangle$, for automata $\mathcal{A}_\varphi$ and $\mathcal{A}_\psi$, or LTL formulas φ and ψ.

Example 1. Consider a scheduler that serves two users and grant them with access to a joint resource. The scheduler can be viewed as an open system with $I = \{\mathsf{req}_1, \mathsf{req}_2\}$, with req_i ($i \in \{1, 2\}$) standing for a request form User i, and $O = \{\mathsf{grant}_1, \mathsf{grant}_2\}$, with grant_i standing for a grant to User i. The system should satisfy mutual exclusion and non-starvation. Formally, the specification for the system is $\varphi_1 \wedge \varphi_2 \wedge \varphi_3$, for $\varphi_1 = G((\neg \mathsf{grant}_1) \vee (\neg \mathsf{grant}_2))$, $\varphi_2 = G(\mathsf{req}_1 \rightarrow F\mathsf{grant}_1)$, and $\varphi_3 = G(\mathsf{req}_2 \rightarrow F\mathsf{grant}_2)$.

We may want to hide from an observer of the interaction the exact policy scheduling of the system. For example,[3] the secret $\psi_1 = ((\neg \mathsf{grant}_1) W \mathsf{req}_1) \wedge G(\mathsf{grant}_1 \rightarrow X((\neg \mathsf{grant}_1) W \mathsf{req}_1))$ reveals whether the system gives User 1 grants only after requests that have not been granted yet. Indeed, ψ_1 specifies that once a grant to User 1 is given, no more grants are given to her, unless a new request from her arrives. A similar secret can be specified for User 2. Note that in order to hide ψ_1, it is sufficient to hide only one of the signals req_1 or grant_1. In fact, this is true even when the observer knows the specification for the system. Then, the secret $\psi_2 = G((\mathsf{req}_1 \rightarrow \mathsf{grant}_1 \vee X\mathsf{grant}_1) \wedge (\mathsf{req}_2 \rightarrow \mathsf{grant}_2 \vee X\mathsf{grant}_2))$ reveals whether delays in grants are limited to one cycle. Here, unlike with ψ_2, it is not sufficient hiding only a single request or even both. Indeed, some policies disclose the satisfaction value of ψ_2 even when requests are hidden. For example, a system that simply alternates between grants, thus outputs $\{\mathsf{grant}_1\}, \{\mathsf{grant}_2\}, \{\mathsf{grant}_1\}, \{\mathsf{grant}_2\}, \ldots$, satisfies the specification and clearly satisfies ψ_2 regardless of the users' requests.

[3] The LTL operator W is "weak Until", thus $p_1 W p_2 = (p_1 U p_2) \vee G p_1$.

Consider now the secret $\psi_3 = FG(\mathsf{req}_1 \rightarrow \mathsf{grant}_1)$, which asserts that eventually, the requests of User 1 are always granted immediately. A system that satisfies ψ_3 is unfair to User 2. Aiming to hide this unfair behavior, we can use the conditional secret $\langle \psi_3, \psi_3 \rangle$, which requires a system that satisfies ψ_3 to hide its satisfaction.

Some computations that satisfy ψ_3, however, may still be fair to User 2. For example, if ψ_3 is satisfied vacuously or if only finitely many requests are sent from User 2, then the behavior specified in ψ_3 is fair, and we need not hide it. Accordingly, we can strengthen the trigger ψ_3 and restrict further the computations in which the satisfaction value of ψ_3 should be hidden. Formally, we replace the trigger ψ_3 by a trigger $\psi_3 \wedge \theta$, for a behavior θ in which a scheduling policy that satisfies ψ_3 is not fair (and hence, need to be hidden).

Let us consider possible behaviors θ for the conditional secret $\langle \psi_3 \wedge \theta, \psi_3 \rangle$. As discussed above, behaviors that make ψ_3 unfair are $GF\mathsf{req}_1$, implying that ψ_3 is not satisfied vacuously, and $GF\mathsf{req}_2$, implying that the immediate grants to User 1 are not due to no requests from User 2. Taking $\theta = (GF\mathsf{req}_1) \wedge (GF\mathsf{req}_2)$ results in a more precise conditional secret.

The trigger θ can be made more precise: taking $\theta = GF(\mathsf{req}_1 \wedge \mathsf{req}_2)$ still guarantees no vacuous satisfaction and also asserts that immediate grants to User 1 are given even when the requests of User 1 arrive together with those of User 2. In fact, $\theta = GF(\mathsf{req}_2 \wedge (\neg \mathsf{grant}_2) U \mathsf{req}_1))$ is even more precise, as it excludes the possibility that the requests of User 1 arrive before those of User 2. Note that the secret can be made less restrictive too, for example with $\psi_3' = FG(\mathsf{req}_1 \rightarrow ((\neg \mathsf{grant}_2) U \mathsf{grant}_1))$, which specifies that eventually, grants to User 1 are always given before grants to User 2. □

Example 2. As a different example, consider a paint robot that paints parts of manufactured pieces. The set O includes 3 signals c_0, c_1, c_2 that encode 8 colors. The encoding corresponds to the composition of the color from paint in three different containers. For example, color 101 stands for the robot mixing paints from containers 0 and 2. The observer does not see the generated pattern, but, unless we hide it, may see the arm of the robot when it reaches a container. Accordingly, hiding of signals in O involve different costs.

The user instructs the robot whether to stay with the current color or change it, thus $I = \{\mathsf{change}\}$. We seek a system that directs the robot which color to chose, in a way that satisfies requirements about the generated pattern. For example, in addition to the requirement to respect the changing instructions ($\xi_{respect}$), the specification φ may require the pattern to start with color 000, and if there are infinitely many changes, then all colors are used (ξ_{all}), yet color 000 repeats between each two colors (ξ_{repeat}). Formally,

- $\xi_{respect} = G((X\neg\mathsf{change}) \leftrightarrow ((c_0 \leftrightarrow Xc_0) \wedge (c_1 \leftrightarrow Xc_1) \wedge (c_2 \leftrightarrow Xc_2)))$,
- $\xi_{all} = GF(\bar{c_0} \wedge \bar{c_1} \wedge \bar{c_2}) \wedge GF(\bar{c_0} \wedge \bar{c_1} \wedge c_2) \wedge \cdots \wedge GF(c_0 \wedge c_1 \wedge c_2)$,
- $\xi_{repeat} = G((c_0 \vee c_1 \vee c_2) \rightarrow X(\mathsf{change} \rightarrow (\bar{c_0} \wedge \bar{c_1} \wedge \bar{c_2})))$, and
- $\varphi = (\bar{c_0} \wedge \bar{c_1} \wedge \bar{c_2}) \wedge \xi_{respect} \wedge ((GF\mathsf{change}) \rightarrow (\xi_{all} \wedge \xi_{repeat}))$.

We may want to hide from an observer certain patterns that the robot may produce. For example, the fact color 111 is used only after color 110 (with color

000 between them), the fact there are colors other than 000 that repeat without a color different from 000 between them, and more. Note that not all the signals in O need to be hidden, and that the choice of signals to hide depends on the secrets as well as the cost of hiding. □

3 Solving Synthesis with Privacy

In this section we describe a solution to the problem of synthesis with privacy for LTL specifications and show that it is 2EXPTIME-complete, thus not harder than LTL synthesis. The solution is based on replacing the specification by one that guarantees the hiding of the secret. For this, we need the following two constructions.

Lemma 1. *Consider a nondeterministic automaton $\mathcal{A} = \langle 2^{I \cup O}, Q, q_0, \delta, \alpha \rangle$. Given a set $\mathcal{H} \subseteq I \cup O$, there is a transition function $\delta^{\mathcal{H}} : Q \times 2^{I \cup O} \to 2^Q$ such that the nondeterministic automaton $\mathcal{A}^{\mathcal{H}} = \langle 2^{I \cup O}, Q, q_0, \delta^{\mathcal{H}}, \alpha \rangle$ is such that $L(\mathcal{A}^{\mathcal{H}}) = \mathsf{noise}_{\mathcal{H}}(L(\mathcal{A}))$.*

Proof. Intuitively, the transition function $\delta^{\mathcal{H}}$ increases the nondeterminism of δ by guessing an assignment to the signals in $\mathcal{H}$. Formally, for $q \in Q$ and $\sigma \in 2^{I \cup O}$, we define $\delta^{\mathcal{H}}(q, \sigma) = \bigcup_{\sigma' \in \mathsf{noise}_{\mathcal{H}}(\sigma)} \delta(q, \sigma')$. It is easy to see that a word w' is accepted by $\mathcal{A}^{\mathcal{H}}$ iff there is a word w accepted by $\mathcal{A}$ such that $w' \in \mathsf{noise}_{\mathcal{H}}(w)$. □

Note that while $\mathcal{A}^{\mathcal{H}}$ maintains the state space and acceptance condition of $\mathcal{A}$, it does not preserve determinism. Indeed, unless $\mathcal{H} = \varnothing$, we have that $\mathcal{A}^{\mathcal{H}}$ is nondeterministic even when $\mathcal{A}$ is deterministic. Next, in Lemma 2 we construct automata that accept computations that satisfy the specification and hide the secret when a given set of signals is hidden.

Lemma 2. *Consider two disjoint finite sets I and O, a subset $\mathcal{H} \subseteq I \cup O$, and ω-regular languages L_φ and L_ψ over the alphabet $2^{I \cup O}$. There exists a DPW $\mathcal{D}^{\mathcal{H}}_{\varphi,\psi}$ with alphabet $2^{I \cup O}$ that accepts a computation $w \in (2^{I \cup O})^\omega$ iff $w \in L_\varphi$ and there exist $w^+, w^- \in \mathsf{noise}_{\mathcal{H}}(w)$ such that $w^+ \in L_\psi$ and $w^- \notin L_\psi$.*

1. *If L_φ and L_ψ are given by LTL formulas φ and ψ, then $\mathcal{D}^{\mathcal{H}}_{\varphi,\psi}$ has $2^{2^{O(|\varphi|+|\psi|)}}$ states and index $2^{O(|\varphi|+|\psi|)}$.*
2. *If L_φ and L_ψ are given by DPWs $\mathcal{D}_\varphi$ and $\mathcal{D}_\psi$ with n_φ and n_ψ states, and of indices k_φ and k_ψ, then $\mathcal{D}^{\mathcal{H}}_{\varphi,\psi}$ has $2^{O(n_\varphi \cdot k_\varphi \cdot (n_\psi \cdot k_\psi)^2 \log(n_\varphi \cdot k_\varphi \cdot n_\psi \cdot k_\psi))}$ states and index $O(n_\varphi \cdot k_\varphi \cdot (n_\psi \cdot k_\psi)^2)$.*

Proof. We start with the case L_φ and L_ψ are given by LTL formulas φ and ψ. Let $\mathcal{A}_\varphi$, $\mathcal{A}_\psi$ and $\mathcal{A}_{\neg\psi}$ be NGBWs for L_φ, L_ψ, and $L_{\neg\psi}$. By [31], such NGBWs exist, and are of size exponential in the corresponding LTL formulas. Let $\mathcal{A}^{\mathcal{H}}_\psi$ and $\mathcal{A}^{\mathcal{H}}_{\neg\psi}$ be the NGBWs for $\mathsf{noise}_{\mathcal{H}}(L(\mathcal{A}_\psi))$ and $\mathsf{noise}_{\mathcal{H}}(L(\mathcal{A}\neg\psi))$, respectively, constructed as in Lemma 1.

Now, let $\mathcal{N}^{\mathcal{H}}_{\varphi,\psi}$ be an NGBW for the intersection of the three automata $\mathcal{A}_\varphi$, $\mathcal{A}^{\mathcal{H}}_{\psi}$, and $\mathcal{A}^{\mathcal{H}}_{\neg\psi}$. The NGBW $\mathcal{N}$ can be easily defined on top of the product of the three automata, and hence is of size $2^{O(|\varphi|+|\psi|)}$ and index $O(|\varphi|+|\psi|)$. Observe that indeed, a word $w \in (2^{I\cup O})^\omega$ is accepted by $\mathcal{N}^{\mathcal{H}}_{\varphi,\psi}$ iff $w \models \varphi$ and there exist $w^+, w^- \in \mathsf{noise}_{\mathcal{H}}(w)$ such that $w^+ \models \psi$ and $w^- \not\models \psi$. By [29,23], determinizing $\mathcal{N}^{\mathcal{H}}_{\varphi,\psi}$ results in a DPW $\mathcal{D}^{\mathcal{H}}_{\varphi,\psi}$ with $2^{2^{O(|\varphi|+|\psi|)}}$ states and index $2^{O(|\varphi|+|\psi|)}$, and we are done.

We continue with the case L_φ and L_ψ are given by DPWs $\mathcal{D}_\varphi$ and $\mathcal{D}_\psi$. We first obtain from $\mathcal{D}_\psi$ two NBWs, $\mathcal{A}_\psi$ and $\mathcal{A}_{\neg\psi}$ for L_ψ and $L_{\neg\psi} = (2^{I\cup O})^\omega \setminus L_\psi$ respectively, and also we translate $\mathcal{D}_\varphi$ into an equivalent NBW $\mathcal{A}_\varphi$. Note that the NBWs $\mathcal{A}_\psi$ and $\mathcal{A}_{\neg\psi}$ can be defined with $O(n_\psi \cdot k_\psi)$ states, and that $\mathcal{A}_\varphi$ can be defined with $O(n_\varphi \cdot k_\varphi)$ states. We then obtain the NBWs $\mathcal{A}^{\mathcal{H}}_{\psi}$ and $\mathcal{A}^{\mathcal{H}}_{\neg\psi}$ by applying the construction in Lemma 1 on $\mathcal{A}_\psi$ and $\mathcal{A}_{\neg\psi}$, respectively. We then define an NBW of size $O(n_\varphi \cdot k_\varphi \cdot (n_\psi \cdot k_\psi)^2)$ for the intersection of the three NBWs $\mathcal{A}_\varphi$, $\mathcal{A}^{\mathcal{H}}_{\psi}$, and $\mathcal{A}^{\mathcal{H}}_{\neg\psi}$, and finally determinize it into a DPW $\mathcal{D}^{\mathcal{H}}_{\varphi,\psi}$ with $2^{O(n_\varphi \cdot k_\varphi \cdot (n_\psi k_\psi)^2 \log(n_\varphi \cdot k_\varphi \cdot n_\psi k_\psi))}$ states and index $O(n_\varphi \cdot k_\varphi \cdot (n_\psi k_\psi)^2)$. □

Remark 1. **[The size of $\mathcal{D}^{\mathcal{H}}_{\varphi,\psi}$ for specifications and secrets given by DBWs]** The exponential dependency of $\mathcal{D}^{\mathcal{H}}_{\varphi,\psi}$ in the DPW $\mathcal{D}_\varphi$ in the construction in Lemma 2 follows from the exponential blow up in DPW intersection [4]. When L_φ is given by a DBW $\mathcal{D}_\varphi$, we can first construct a DPW for the intersection of $\mathcal{A}^{\mathcal{H}}_{\psi}$ and $\mathcal{A}^{\mathcal{H}}_{\neg\psi}$, and only then take its intersection with $\mathcal{D}_\varphi$. This results in a DPW $\mathcal{D}^{\mathcal{H}}_{\varphi,\psi}$ of size exponential in $\mathcal{D}_\psi$, but only polynomial in $\mathcal{D}_\varphi$. □

We can now solve synthesis with privacy for LTL formulas.

Theorem 1. [Synthesis with privacy, LTL] *Given two disjoint finite sets I and O, LTL formulas φ and ψ over $I \cup O$, a cost function $\mathsf{cost} : I \cup O \to \mathbb{N}$, and a budget $b \in \mathbb{N}$, deciding whether $\langle \varphi, \psi, \mathsf{cost}, b\rangle$ is realizable with privacy is 2EXPTIME-complete.*

Proof. We start with the upper bound. Given φ, ψ, cost, and b, we go over all $\mathcal{H} \subseteq I \cup O$ such that $\mathsf{cost}(\mathcal{H}) \leq b$, construct the DPW $\mathcal{D}^{\mathcal{H}}_{\varphi,\psi}$ defined in Lemma 2, and check whether $L(\mathcal{D}^{\mathcal{H}}_{\varphi,\psi})$ is realizable. Since realizability of a DPW with n states and index k can be solved in time at most $O(n^k)$ [5], the 2EXPTIME upper bound follows from $\mathcal{D}^{\mathcal{H}}_{\varphi,\psi}$ having $2^{2^{O(|\varphi|+|\psi|)}}$ states and index $2^{O(|\varphi|+|\psi|)}$.

For the lower bound, we describe a reduction from LTL synthesis with no privacy. Note that adding to a specification φ a secret `T` or `F` does not work, as an observer knows its satisfaction value. It is easy, however, to add a secret that is independent of the specification. Specifically, given a specification φ over $I \cup O$, let $O' = O \cup \{p\}$, where p is a fresh signal not in $I \cup O$. Consider the secret $\psi = p$ and a cost function with $\mathsf{cost}(p) = 0$. Clearly, an (I/O)-transducer $\mathcal{T}$ realizes φ iff the (I/O')-transducer $\mathcal{T}'$ that agrees with $\mathcal{T}$ and always assigns `F` to p, realizes φ and $\{p\}$-hides ψ. Conversely, for an I/O'-transducer $\mathcal{T}'$, let $\mathcal{T}$ be the (I/O)-transducer obtained from $\mathcal{T}$ by ignoring the assignments to p.

Clearly, $\mathcal{T}'$ $\{p\}$-hides ψ. In addition, as φ does not refer to p, we have that $\mathcal{T}'$ realizes φ iff $\mathcal{T}$ realizes φ. Thus, φ is realizable iff $\langle \varphi, \psi, \mathsf{cost}, 0 \rangle$ is realizable with privacy. □

Remark 2. **[Solving privacy with multiple and conditional secrets]** Recall that for a set of secrets $S = \{\psi_1, \psi_2, \ldots, \psi_k\}$, an (I/O)-transducer $\mathcal{T}$ realizes $\langle \varphi, S, \mathcal{H} \rangle$ with privacy if it realizes φ and $\mathcal{H}$-hides ψ_i, for all $i \in [k]$. It is easy to extend Theorem 1 to the setting of multiple secrets by replacing the DPW $\mathcal{D}^{\mathcal{H}}_{\varphi,\psi}$ by a DPW obtained by determinizing the product of $\mathcal{A}_\varphi$ with automata $\mathcal{A}^{\mathcal{H}}_{\psi_i}$ and $\mathcal{A}^{\mathcal{H}}_{\neg\psi_i}$, for all $1 \leq i \leq k$.

As for conditional secrets, recall that a computation should satisfy the specification, and from the point of view of an observer, either the trigger is not triggered, thus $\pi \in L(\mathcal{A}^{\mathcal{H}}_{\neg\theta})$, or the secret is hidden, thus $\pi \in L(\mathcal{A}^{\mathcal{H}}_{\psi}) \cap L(\mathcal{A}^{\mathcal{H}}_{\neg\psi})$. Accordingly, we need to construct a deterministic automaton for $L(\mathcal{A}_\varphi) \cap (L(\mathcal{A}^{\mathcal{H}}_{\neg\theta}) \cup L(\mathcal{A}^{\mathcal{H}}_{\psi}) \cup L(\mathcal{A}^{\mathcal{H}}_{\neg\psi}))$. This can done by determinizing an NBW that is defined on top of the product of $\mathcal{A}_\varphi, \mathcal{A}^{\mathcal{H}}_{\neg\theta}, \mathcal{A}^{\mathcal{H}}_{\psi}$, and $\mathcal{A}^{\mathcal{H}}_{\neg\psi}$. □

While the complexity of our algorithm is not higher than that of LTL synthesis with no privacy, it would be misleading to state that handling of privacy involves no increase in the complexity. Indeed, the algorithm involved two components whose complexity may have been dominated by the doubly exponential translation of the LTL formulas to deterministic automata:

1. A need to go over all candidate sets $\mathcal{H} \subseteq I \cup O$.
2. A need to check that the generated transducer $\mathcal{H}$-hides the secret.

In the next two sections, we isolate these two components of synthesis with privacy and show that each of them involves an exponential complexity: the first in the number of signals and the second in the size of the secret.

3.1 Hiding secrets is hard

The synthesis problem for DBWs can be solved in polynomial time. Indeed, the problem can be reduced to solving a Büchi game played on top of the specification automaton. In this section we show that synthesis with privacy is EXPTIME hard even for a given set $\mathcal{H}$ of hidden signals (in fact, even a singleton set $\mathcal{H} \subseteq I$), a trivial specification, and a secret given by a DBW.

We start by showing that $\mathcal{H}$-hiding is hard even for secrets given by DBWs.

Theorem 2. *Given two disjoint finite sets I and O, a DBW $\mathcal{D}_\psi$ over $2^{I \cup O}$, and a set $\mathcal{H} \subseteq I \cup O$ of hidden signals, deciding whether there exists an (I/O)-transducer that $\mathcal{H}$-hides $\mathcal{D}_\psi$ is EXPTIME-hard. The problem is EXPTIME-hard already when $\mathcal{H} \subseteq I$.*

Proof. We describe a polynomial-time reduction from NBW realizability, which is EXPTIME-hard [26,17]. Given an NBW $\mathcal{A}$ over $2^{I \cup O}$, we define a set of signals $\mathcal{H}$ and a DBW $\mathcal{D}_\psi$ over $2^{I \cup O \cup \mathcal{H}}$, such that $L(\mathcal{A})$ is realizable iff there exists an $((I \cup \mathcal{H})/O)$-transducer that $\mathcal{H}$-hides $L(\mathcal{D}_\psi)$.

Let $\mathcal{A} = \langle 2^{I \cup O}, Q, q_0, \delta, \alpha \rangle$. W.l.o.g, we assume that $\mathcal{A}$ has a single initial state and that every word in $(2^{I \cup O})^\omega$ has at least one rejecting run in $\mathcal{A}$. The latter can be achieved, for example, by adding a nondeterministic transition from the initial state to a rejecting sink upon any assignment $i \cup o \in 2^{I \cup O}$. Let $\mathcal{H}$ be a set of signals that encode Q. Thus, each assignment $s \in 2^{\mathcal{H}}$ is associated with a single state in Q. We refer to a letter in $2^{I \cup O \cup \mathcal{H}}$ as a pair $\langle \sigma, q \rangle \in 2^{I \cup O} \times Q$, and we view a word in $(2^{I \cup O \cup \mathcal{H}})^\omega$ as the combination $w \oplus r$, of a word $w \in (2^{I \cup O})^\omega$ with a word $r \in Q^\omega$. Formally, for $w = \sigma_0 \cdot \sigma_1 \cdots \in (2^{I \cup O})^\omega$ and $r = r_1 \cdot r_2 \cdots \in Q^\omega$, let $w \oplus r = \langle \sigma_0, r_1 \rangle \cdot \langle \sigma_1, r_2 \rangle \cdots \in (2^{I \cup O \cup \mathcal{H}})^\omega$. Then, we define $\mathcal{D}_\psi$ so that $L(\mathcal{D}_\psi) = \{w \oplus r \in (2^{I \cup O \cup \mathcal{H}})^\omega : \text{the sequence } q_0 \cdot r \text{ is an accepting run of } \mathcal{A} \text{ on } w\}$. Note that since every word in $(2^{I \cup O})^\omega$ has at least one rejecting run in $\mathcal{A}$, then every word $w \in L(\mathcal{A})$ has at least one word $r^+ \in Q^\omega$ such that $w \oplus r^+ \in L(\mathcal{D}_\psi)$ and at least one word $r^- \in Q^\omega$ such that $w \oplus r^- \notin L(\mathcal{D}_\psi)$.

Formally, $\mathcal{D}_\psi = \langle 2^{I \cup O \cup \mathcal{H}}, Q, q_0, \delta', \alpha \rangle$ has the same state space and acceptance condition as $\mathcal{A}$, and it uses the Q-component of each letter in order to resolve the nondeterministic choices in $\mathcal{A}$. Thus, the transitions function $\delta' : Q \times 2^{I \cup O \cup \mathcal{H}} \to Q$ is defined as follows. For every state $q \in Q$ and letter $\langle \sigma, s \rangle \in 2^{I \cup O \cup \mathcal{H}}$, we have that $\delta'(q, \langle \sigma, s \rangle) = s$ if $s \in \delta(q, \sigma)$, and otherwise $\delta'(q, \langle \sigma, s \rangle) = \bot$. We prove that indeed $L(\mathcal{D}_\psi)$ accepts exactly all words $w \oplus r$ such that $q_0 \cdot r$ is an accepting run of $\mathcal{A}$ on w. By definition of δ', it holds that r' is a run of $\mathcal{D}_\psi$ over $w \oplus r$ iff $r' = q_0 \cdot r$, and $q_0 \cdot r$ is a run of $\mathcal{A}$ over w. Hence, a run $r' = q_0 \cdot r$ of $\mathcal{D}_\psi$ over $w \oplus r$ is accepting, iff $inf(r') \cap \alpha \neq \varnothing$, iff $r' = q_0 \cdot r$ is an accepting run of $\mathcal{A}$ over w, and we are done. □

Note that in the proof of Theorem 2, we could have defined $\mathcal{H}$ so that it resolves the nondeterminism in $\mathcal{A}$ in a more concise way. In particular, if we assume that the nondeterminism degree in $\mathcal{A}$ is at most 2, then a set $\mathcal{H}$ of size 1 can resolve the nondeterminizm of δ. Hence, as NBW synthesis is EXPTIME-hard already for NBWs with branching degree 2 (this follows from the fact that a bigger branching degree can be decomposed along several transitions), EXPTIME hardness holds already when hiding a single input signal.

Theorem 3. [Synthesis with privacy, DPWs] *Given two disjoint finite sets I and O, DPWs $\mathcal{D}_\varphi$ and $\mathcal{D}_\psi$ over $2^{I \cup O}$, and a set $\mathcal{H} \subseteq I \cup O$ of hidden signals, deciding whether $\langle \mathcal{D}_\varphi, \mathcal{D}_\psi, \mathcal{H} \rangle$ is realizable with privacy is EXPTIME-complete. Moreover, hardness holds already when the specification is trivial and the secret is given by a DBW.*

Proof. For the upper bound, we solve the synthesis problem for the DPW $\mathcal{D}^{\mathcal{H}}_{\varphi,\psi}$ defined in Lemma 2. As specified there, the size of $\mathcal{D}^{\mathcal{H}}_{\varphi,\psi}$ is exponential in the size and index of both $\mathcal{D}_\varphi$ and $\mathcal{D}_\psi$, and its index is polynomial in the size and index of $\mathcal{D}_\varphi$ and $\mathcal{D}_\psi$. Membership in EXPTIME then follows from the complexity of the synthesis problem for DPWs [2].

For the lower bound, fix a DBW $\mathcal{D}_{\mathrm{T}}$ such that $L(\mathcal{D}_{\mathrm{T}}) = (2^{I \cup O})^\omega$. Then, it is easy to see that $\langle \mathcal{D}_{\mathrm{T}}, \mathcal{D}_\psi, \mathcal{H} \rangle$ is realizable with privacy iff there is an (I/O)-transducer that $\mathcal{H}$-hides $\mathcal{D}_\psi$. Thus, hardness in EXPTIME follows from Theorem 2. □

3.2 Searching for a set of signals to hide is hard

Another component in the algorithm that is dominated by the doubly-exponential translation of LTL to DPWs is the need to go over all subsets of $I \cup O$ in a search for the set $\mathcal{H}$ of signals to hide. Trying to isolate the influence of this search, it is not enough to consider specifications and secrets that are given by DBWs, as the synthesis with privacy problem is EXPTIME-hard already for a given set $\mathcal{H}$, and so again, the complexity of the search is dominated by the complexity of the synthesis problem. Fixing the size of the secret, which is the source of the exponential complexity, does not not work either, as it also fixes the number of signals that we may need to hide. We address this challenge by moving to an even simpler setting for the problem, namely synthesis with privacy of a *closed* system. We are going to show that in this setting, the search for $\mathcal{H}$ is the only non-polynomial component in the algorithm.

In the closed setting, all signals are controlled by the system, namely $I = \varnothing$. Consequently, each transducer has a single computation, and realizability coincides with satisfiability. In particular, for $I = \varnothing$, we have that $\langle L_\varphi, L_\psi, \mathcal{H}\rangle$ is realizable with privacy iff there exists a word $w \in L_\varphi$, for which there exist two words $w^+, w^- \in \mathsf{noise}_\mathcal{H}(w)$ such that $w^+ \in L_\psi$ and $w^- \notin L_\psi$. We show that while synthesis with privacy in the closed setting can be solved in polynomial time for a given set $\mathcal{H}$ of hidden signals, it is NP-complete when $\mathcal{H}$ is not given, even when the function *cost* is uniform.

We start with the case $\mathcal{H}$ is given.

Theorem 4. *Given a finite set O of output signals, a set $\mathcal{H} \subseteq O$ of hidden signals, and DPWs $\mathcal{D}_\varphi$ and $\mathcal{D}_\psi$ over 2^O, deciding whether $\langle \mathcal{D}_\varphi, \mathcal{D}_\psi, \mathcal{H}\rangle$ is realizable with privacy can be done in polynomial time.*

Proof. First, we complement $\mathcal{D}_\psi$, which results in a DPW $\mathcal{D}_{\neg\psi}$ of the same size, and of index $k+1$, where k is the index of $\mathcal{D}_\psi$. Then, we translate $\mathcal{D}_\varphi$, $\mathcal{D}_\psi$ and $\mathcal{D}_{\neg\psi}$ into equivalent NBWs $\mathcal{A}_\varphi$, $\mathcal{A}_\psi$ and $\mathcal{A}_{\neg\psi}$, respectively. All three NBWs can be defined in size that is polynomial in their deterministic DPW counterpart. Let $\mathcal{A}_\psi^\mathcal{H}$ and $\mathcal{A}_{\neg\psi}^\mathcal{H}$ be NBWs obtained by applying the construction in Lemma 1 on $\mathcal{A}_\psi$ and $\mathcal{A}_{\neg\psi}$, respectively. By Lemma 1, the NBWs $\mathcal{A}_\psi^\mathcal{H}$ and $\mathcal{A}_{\neg\psi}^\mathcal{H}$ have the same number of states as $\mathcal{A}_\psi$ and $\mathcal{A}_{\neg\psi}$ respectively. Let $\mathcal{N}$ be an NBW for the intersection $L(\mathcal{A}_\varphi) \cap L(\mathcal{A}_\psi^\mathcal{H}) \cap L(\mathcal{A}_{\neg\psi}^\mathcal{H})$. Note that $\mathcal{N}$ can be defined with size that is polynomial in $\mathcal{A}_\varphi$, $\mathcal{A}_\psi^\mathcal{H}$ and $\mathcal{A}_{\neg\psi}^\mathcal{H}$. Moreover, $\mathcal{N}$ accepts a word w iff $w \in L(\mathcal{A}_\varphi)$, and there exist two words $w^+, w^- \in \mathsf{noise}_\mathcal{H}(w)$ such that $w^+ \in L(\mathcal{A}_\psi)$ and $w^- \notin L(\mathcal{A}_\psi)$. Thus, realizability with privacy of $\langle \mathcal{A}_\varphi, \mathcal{A}_\psi, \mathcal{H}\rangle$ can be reduced to the nonemptiness of $\mathcal{N}$, which can be decided in polynomial time. □

We continue to the case $\mathcal{H}$ should be searched.

Theorem 5. *Given a finite set of output signals O, DPWs $\mathcal{A}_\varphi$ and $\mathcal{A}_\psi$ over 2^O, a hiding cost function $\mathsf{cost} : O \to \mathbb{N}$, and a budget $b \in \mathbb{N}$, deciding whether $\langle \mathcal{A}_\varphi, \mathcal{A}_\psi, \mathsf{cost}, b\rangle$ is realizable with privacy is NP-complete. Moreover, hardness holds already when the specification and secret are given by DBWs.*

Proof. For the upper bound, a nondeterministic Turing machine can guess a set $\mathcal{H} \subseteq O$, check whether $\mathsf{cost}(\mathcal{H}) \leq b$, and, by Theorem 4, check in polynomial time whether $\langle \mathcal{A}_\varphi, \mathcal{A}_\psi, \mathcal{H} \rangle$ is realizable with privacy.

For the lower bound, we describe a polynomial-time reduction from the *vertex-cover* problem. In this problem, we are given an undirected graph $G = \langle V, E \rangle$ and $k \geq 1$, and have to decide whether there is a set $S \subseteq V$ such that $|S| \leq k$ and for every edge $\{v, u\} \in E$, we have that $E \cap S \neq \varnothing$. Given an undirected graph $G = \langle V, E \rangle$, with $E = \{e_1, e_2, \ldots, e_m\}$, we consider a closed setting with $O = V$ and construct DBWs $\mathcal{A}_\varphi$ and $\mathcal{A}_\psi$ over the alphabet 2^V such that for all $\mathcal{H} \subseteq V$, it holds that $\langle \mathcal{A}_\varphi, \mathcal{A}_\psi, \mathcal{H} \rangle$ is realizable with privacy iff $\mathcal{H}$ is a vertex cover of G. Accordingly, there is a vertex cover of size k in G iff $\langle \mathcal{A}_\varphi, \mathcal{A}_\psi, \mathsf{cost}, k \rangle$ is realizable with privacy for the uniform cost function that assigns 1 to all signals in O.

We define $\mathcal{A}_\varphi$ and $\mathcal{A}_\psi$ over the alphabet 2^V as follows. The DBW $\mathcal{A}_\varphi$ is a 2-state DBW that accepts the single word $\varnothing^\omega$. The DBW $\mathcal{A}_\psi = \langle 2^V, Q, q_1, \delta, \alpha \rangle$ for the secret is defined as follows. The set of states is $Q = \{q_1, q_2, \ldots, q_{m+1}\}$, the set of accepting states is $\alpha = \{q_{m+1}\}$ and the transition function δ is defined for all $S \subseteq O$ and $i \leq m$ by, $\delta(q_i, S) = q_{i+1}$ if $S \cap e_i \neq \varnothing$, and $\delta(q_i, S) = \bot$ otherwise. Finally, $\delta(q_{m+1}, S) = q_{m+1}$ for all $S \subseteq V$. That is, words in $L(\mathcal{A}_\psi)$ encode vertex covers of G. Indeed, if $w = S_1 \cdot S_2 \cdot S_2 \cdot \ldots \in L(\mathcal{A}_\psi)$, then for all $i \leq m$ we have that $S_i \cap e_i \neq \varnothing$. Thus, if for all $i \leq m$ we set $v_i \in V$ to be some vertex in $S_i \cap e_i$, then we get that $\{v_i, \ldots, v_m\}$ is a vertex cover of G.

In the full version, we prove that $\langle \mathcal{A}_\varphi, \mathcal{A}_\psi, \mathcal{H} \rangle$ is realizable with privacy iff $\mathcal{H}$ is a vertex cover of G. □

4 Bounded Synthesis with Privacy

In the general synthesis problem, there is no bound on the size of the generated system. It is not hard to see that if a system that realizes the specification exists, then there is also one whose size is bounded by the size of a deterministic automaton for the specification. For the case of LTL specifications, this gives a doubly-exponential bound on the size of the generated transducer, which is known to be tight [28]. In [30], the authors suggested to study *bounded synthesis*, where the input to the problem includes also a bound on the size of the system. The bound not only guarantees the generation of a small system, if it exists, but also reduces the complexity of the synthesis problem and gives rise to a symbolic implementation and further extensions [12,19]. In particular, for LTL, it is easy to see that bounded synthesis can be solved in PSPACE, as one can go over and model-check all candidate systems. For specifications in DPW, the bound actually increases the complexity, as going over all candidates results in an algorithm in NP.

In this section we study bounded synthesis with privacy. As in traditional synthesis, the hope is to both reduce the complexity of the problem and to end up with smaller systems. In addition to a specification L_φ, a secret L_ψ, and a set $\mathcal{H} \subseteq I \cup O$ of hidden signals, we are given a bound $n \in \mathbb{N}$, represented in

unary, and we are asked to construct an (I/O)-transducer with at most n states that realizes $\langle L_\varphi, L_\psi, \mathcal{H}\rangle$ with privacy, or to determine that no such transducer exists. As in the unbounded case, we can define the problem also with respect to a hiding cost function and a budget.

4.1 Hiding secrets by a bounded system is hard

We first show that hiding secrets in a bounded setting is hard. In fact, the complexity of hiding goes beyond the complexity of bounded synthesis with no privacy already in the case the specification and secrets are given by LTL formulas. In the case of DBWs and DPWs, hiding is also more complex than bounded synthesis without privacy, but the difference is not significant.

The key idea in both results is similar to the one in the proof of Theorem 2. There, we reduce realizability of NBWs to hiding of secrets given by DBWs. Essentially, we use the hidden signals to imitate nondeterminism. Here, with a bound on the size of the system, we cannot reduce from realizability, as the problem has the flavor of model checking many candidates. Accordingly, we reduce from universality, either in the form of LTL formulas with universally-quantified atomic propositions, or in the form of language-universality for NBWs.

Theorem 6. *Given two disjoint finite sets I and O, an LTL formula ψ over $2^{I\cup O}$, a set $\mathcal{H} \subseteq I \cup O$ of hidden signals, and a bound $n \geq 1$, given in unary, deciding whether there exists an (I/O)-transducer of size at most n that $\mathcal{H}$-hides ψ is EXPSPACE-hard. The problem is EXPSPACE-hard already when $n = 1$.*

Theorem 7. *Given two disjoint finite sets I and O, a DBW $\mathcal{D}_\psi$ over $2^{I\cup O}$, a set $\mathcal{H} \subseteq I \cup O$ of hidden signals, and a bound $n \geq 1$, represented in unary, deciding whether there exists an (I/O)-transducer of size at most n that $\mathcal{H}$-hides $\mathcal{D}_\psi$ is PSPACE-hard. The problem is PSPACE-hard already when $n = 1$.*

4.2 Solving bounded synthesis with privacy

We can now present the tight complexity for bounded synthesis with privacy for both types of specification formalisms. For the upper bounds, we construct an NGBW $\mathcal{N}^{\mathcal{H}}_{\varphi,\psi}$ that accepts exactly all words that satisfy φ and hide ψ, and search for an (I/O)-transducer of size n whose language is contained in that of $\mathcal{N}^{\mathcal{H}}_{\varphi,\psi}$.

Theorem 8. [Bounded synthesis with privacy] *Given two disjoint finite sets I and O, specification L_φ and secret L_ψ over $I\cup O$, a set $\mathcal{H} \subseteq I\cup O$ of hidden signals, and a bound $n \in \mathbb{N}$, represented in unary, deciding whether there is an (I/O)-transducer with at most n states that realizes $\langle L_\varphi, L_\psi, \mathcal{H}\rangle$ with privacy is PSPACE-complete for L_φ and L_ψ given by DPWs, and is EXPSPACE-complete for L_φ and L_ψ given by LTL formulas. Hardness in PSPACE holds already for DBWs.*

5 When the Observer Knows the Specification

In this section we study a setting in which the observer knows the specification φ of the system. Technically, it means that when the observer tries to evaluate the secret, she knows that only computations that satisfy φ should be taken into account. If, for example, $\varphi \rightarrow \psi$, then ψ cannot be kept private in a setting in which the observer knows φ. Indeed, the fact φ is realized by the system reveals that ψ is satisfied. Formally, we say that $\mathcal{T}$ *realizes* $\langle \varphi, \psi, \mathcal{H} \rangle$ *with privacy under the knowledge of the specification* if $\mathcal{T}$ realizes φ, and for every $w_I \in (2^I)^\omega$, there exist $w^+, w^- \in \mathsf{noise}_{\mathcal{H}}(\mathcal{T}(w_I)) \cap L_\varphi$ such that $w^+ \models \psi$ and $w^- \not\models \psi$. Thus, the satisfaction of the secret ψ in a computation $\mathcal{T}(w_I)$ cannot be deduced from the observable computation $\mathsf{hide}_{\mathcal{H}}(\mathcal{T}(w_I))$ even when the observer knows that φ is satisfied in $\mathcal{T}(w_I)$. The adjustment for the definition of the problem with respect to a hiding cost function and a budget is similar.

We start by showing the analogue of Lemma 2 for the setting in which the observer knows the specification. The construction is similar to that of Lemma 2, except that now, the construction of the DPW $\mathcal{D}^{\mathcal{H}}_{\psi|\varphi}$ involves an existential projection on $\mathcal{H}$ also in the automaton for the specification. Accordingly, the size of the DPW is exponential in both the specification and the secret even in the case they are given by DBWs.

Lemma 3. *Consider two disjoint finite sets I and O, a subset $\mathcal{H} \subseteq I \cup O$, and regular languages L_φ and L_ψ over the alphabet $2^{I \cup O}$. There exists a DPW $\mathcal{D}^{\mathcal{H}}_{\psi|\varphi}$ with alphabet $2^{I \cup O}$ that accepts a computation $w \in (2^{I \cup O})^\omega$ iff $w \in L_\varphi$ and there exist $w^+, w^- \in \mathsf{noise}_{\mathcal{H}}(w)$ such that $w^+ \in L_\varphi \cap L_\psi$ and $w^- \in L_\varphi \setminus L_\psi$.*

1. *If L_φ and L_ψ are given by LTL formulas φ and ψ, then $\mathcal{D}^{\mathcal{H}}_{\psi|\varphi}$ has $2^{2^{O(|\varphi|+|\psi|)}}$ states and index $2^{O(|\varphi|+|\psi|)}$.*
2. *If L_φ and L_ψ are given by DPWs $\mathcal{D}_\varphi$ and $\mathcal{D}_\psi$ with n_φ and n_ψ states, and of indices k_φ and k_ψ, then $\mathcal{D}^{\mathcal{H}}_{\psi|\varphi}$ has $2^{O((n_\varphi \cdot k_\varphi)^3 \cdot (n_\psi \cdot k_\psi)^2 \log(n_\varphi \cdot k_\varphi \cdot n_\psi \cdot k_\psi))}$ states and index $O((n_\varphi \cdot k_\varphi)^3 \cdot (n_\psi \cdot k_\psi)^2)$.*

Lemma 3 implies that all the asymptotic upper bounds described in Section 3 are valid also in a setting with an observer that knows the specification. Also, as the lower bounds in Theorems 1 and 3 involve secrets that are independent of the specification, they are valid for this setting too. Two issues require a consideration:

1. The need to search for $\mathcal{H}$: the NP-hardness proof in Theorem 5 is no longer valid, as there, $\varphi \rightarrow \neg\psi$, and so the satisfaction value of the secret is revealed in a setting with an observer that knows the specification.
2. The construction in Lemma 3 results in an algorithm that is exponential also in the specification, even when given by a DBW. On the other hand, the EXPTIME-hardness proof in Theorem 2 does not imply an exponential lower bound in the specification.

Below we address the two issues, providing lower bounds for a setting in which the observer knows the specification. Matching upper bounds follow the same considerations in Theorems 3 and 5, where $\mathcal{D}^{\mathcal{H}}_{\psi|\varphi}$ replaces $\mathcal{D}^{\mathcal{H}}_{\varphi,\psi}$. We start with a variant of Theorem 5, showing NP-hardness also in the setting of a knowledgeable observer. As mentioned above, the lower bound in the proof of Theorem 5 does not work when the observer knows the specification, yet, can easily be modified to work for the case of a knowledgeable observer.

Theorem 9. *Given a set O of output signals, a cost function $\mathsf{cost} : O \to \mathbb{N}$, a hiding budget $b \in \mathbb{N}$, and DBWs $\mathcal{A}_\varphi$ and $\mathcal{A}_\psi$ over 2^O, deciding whether there is $\mathcal{H} \subseteq O$, with $\mathsf{cost}(\mathcal{H}) \leq b$, such that $\langle \mathcal{A}_\varphi, \mathcal{A}_\psi, \mathcal{H}\rangle$ is realizable with privacy under knowledge of the specification is NP-hard. Moreover, hardness holds already when cost is uniform.*

We continue to the second issue, proving that synthesis with privacy under knowledge of the specification is EXPTIME-hard even for specifications in DBWs and secrets of a fixed size. Note that synthesis with privacy (without knowledge of the specification) can be solved in PTIME in this case (see Remark 1). The proof is similar to that of Theorem 2, except that here the lower bound needs the secret to be a of a fixed size, making the specification more complex. It follows that the exponential blow-up in $\mathcal{D}_\varphi$, which exists in Lemma 3 cannot be avoided even when it is a DBW and $\mathcal{D}_\psi$ is of a fixed size.

Theorem 10. *Given two disjoint finite sets I and O, DBWs $\mathcal{D}_\varphi$ and $\mathcal{D}_\psi$ over $2^{I\cup O}$, and a set $\mathcal{H} \subseteq I \cup O$ of hidden signals, deciding whether $\langle \mathcal{D}_\varphi, \mathcal{D}_\psi, \mathcal{H}\rangle$ is realizable with privacy under knowledge of the specification is EXPTIME-hard already when $\mathcal{D}_\psi$ is of fixed size.*

Remark 3. Recall that an observer that knows the specification can restrict the search for computations on which she evaluates the secret to ones that satisfy the specification. In fact, the observer can do better, and restricts the search to computations that are generated by an (I/O)-transducer that realizes the specification.

In order to see the difference between the two definitions, consider the case where $I = \mathcal{H} = \{p_1, p_2\}$, $O = \{q\}$, $\varphi = (q \leftrightarrow p_1) \vee G p_2$, and $\psi = p_1$. An observer that knows φ does not know which of its two disjuncts is satisfied, and thus, even though she observes q, the value of p_1 stays secret. Formally, a transducer that realizes $q \leftrightarrow p_1$ $\mathcal{H}$-hides ψ from the observer, even if the observer knows that φ is satisfied. Indeed, for every observable computation $\kappa \in 2^{\{q\}}$, there is a computation $w^+ \in \mathsf{noise}_{\mathcal{H}}(\kappa)$ that satisfies $p_1 \wedge G p_2$ and a computation $w^- \in \mathsf{noise}_{\mathcal{H}}(\kappa)$ that satisfies $(\neg p_1) \wedge G p_2$. Hence, $\langle \varphi, \psi, \mathcal{H}\rangle$ is realizable with privacy even when the observer knows the specification.

On the other hand, a clever observer, especially one that has read [16,14], knows that a transducer $\mathcal{T}$ realizes φ iff $\mathcal{T}$ realizes $q \leftrightarrow p_1$. Indeed, if $\mathcal{T}$ does not satisfy $q \leftrightarrow p_1$, then φ is not satisfied in computations that do not satisfy $G p_2$, which is the case for almost all the computations of $\mathcal{T}$. Accordingly, a clever observer that knows φ can learn the secret p_1 by observing the value of q.

Using the terminology of [16,14], the specification φ and $q \leftrightarrow p_1$ are *open equivalent*: for every transducer $\mathcal{T}$, we have that $\mathcal{T}$ realizes φ iff $\mathcal{T}$ realizes $q \leftrightarrow p_1$. Note that open equivalence is weaker than equivalence. Once we can simplify a specification to an open-equivalent specification that does not include *inherent vacuity*, the two definitions coincide. Such a simplification, however, requires further study. Also, as an unrealizable specification is open-equivalent to `F`, such a simplification is at least as complex as the realizability problem (which is also good news, as it means that an observer needs to solve a 2EXPTIME problem in order to benefit from the difference between the definitions). □

6 Directions for Future Research

We suggested a framework for the synthesis of systems that satisfy their specifications while keeping some behaviors secret. Behaviors are kept secret from an observer by hiding the truth value of some input and output signals, subject to budget restrictions: each signal has a hiding cost, and there is a bound on the total hiding cost. Our framework captures settings in which the choice and cost of hiding are fixed throughout the computation. For example, settings with signals that cannot be hidden (e.g., alarm sound, or the temperature outside), signals that can be hidden throughout the computation with some effort (e.g., hand movement of a robot), or signals that are anyway hidden (e.g., values of internal control variables). Our main technical contribution are lower bounds for the complexity of the different aspects of privacy: the need to choose the hidden signals, and the need to hide the secret behaviors. We show that both aspects involve an exponential blow up in the complexity of synthesis without privacy.

The exponential lower bounds apply already in the relatively simple cost mechanism we study. Below we discuss possible extensions of this mechanism. In settings with a *dynamic hiding of signals*, we do not fix a set $\mathcal{H} \subseteq I \cup O$ of hidden signals. Instead, the output of the synthesis algorithm contains a transducer that describes not only the assignments to the output signals but also the choice of input and output signals that are hidden in the next cycle of the interaction. Thus, signals may be hidden only in segments of the interaction – segments that depend on the history of the interaction so far. For example, we may hide information about a string that is being typed only after a request for a password. In addition, the cost function need not be fixed and may depend on the history of the interaction too. For example, hiding the location of a robot may be cheap in certain sections of the warehouse and expensive in others. Solving synthesis with privacy in a setting with such dynamic hiding and pricing of signals involves automata over the alphabet $3^{I \cup O}$, reflecting the ability of signals to get an "unknown" truth value in parts of the computation. Moreover, as the cost is not known in advance (even when the cost of hiding signals is fixed), several mechanisms for bounding the budget are possible (energy, mean-payoff, etc. [3,6]).

References

1. B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S.P. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. *J. ACM*, 59(2):6:1–6:48, 2012.
2. R. Bloem, K. Chatterjee, and B. Jobstmann. Graph games and reactive synthesis. In *Handbook of Model Checking.*, pages 921–962. Springer, 2018.
3. A. Bohy, V. Bruyère, E. Filiot, and J-F. Raskin. Synthesis from LTL specifications with mean-payoff objectives. In *Proc. 19th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems*, volume 7795 of *Lecture Notes in Computer Science*, pages 169–184. Springer, 2013.
4. U. Boker. Why these automata types? In *Proc. 22nd Int. Conf. on Logic for Programming Artificial Intelligence and Reasoning*, volume 57 of *EPiC Series in Computing*, pages 143–163, 2018.
5. C.S. Calude, S. Jain, B. Khoussainov, W. Li, and F. Stephan. Deciding parity games in quasipolynomial time. In *Proc. 49th ACM Symp. on Theory of Computing*, pages 252–263, 2017.
6. K. Chatterjee and L. Doyen. Energy parity games. In *Proc. 37th Int. Colloq. on Automata, Languages, and Programming*, pages 599–610, 2010.
7. K. Chatterjee, L. Doyen, T. A. Henzinger, and J-F. Raskin. Algorithms for ω-regular games with imperfect information. In *Proc. 15th Annual Conf. of the European Association for Computer Science Logic*, volume 4207 of *Lecture Notes in Computer Science*, pages 287–302, 2006.
8. M.R. Clarkson, B. Finkbeiner, M. Koleini, K.K. Micinski, M.N. Rabe, and C. Sánchez. Temporal logics for hyperproperties. In *3rd International Conference on Principles of Security and Trust*, volume 8414 of *Lecture Notes in Computer Science*, pages 265–284. Springer, 2014.
9. I. Dinur and K. Nissim. Revealing information while preserving privacy. In *Proceedings of the 22nd ACM Symposium on Principles of Database Systems*, pages 202–210. ACM, 2003.
10. J. Dubreil, Ph. Darondeau, and H. Marchand. Supervisory control for opacity. *IEEE Transactions on Automatic Control*, 55(5):1089–1100, 2010.
11. C. Dwork, F. McSherry, K. Nissim, and A.D. Smith. Calibrating noise to sensitivity in private data analysis. *J. Priv. Confidentiality*, 7(3):17–51, 2016.
12. R. Ehlers. Symbolic bounded synthesis. In *Proc. 22nd Int. Conf. on Computer Aided Verification*, volume 6174 of *Lecture Notes in Computer Science*, pages 365–379. Springer, 2010.
13. B. Finkbeiner, C. Hahn, P. Lukert, M. Stenger, and L. Tentrup. Synthesis from hyperproperties. *Acta Informatica*, 57(1-2):137–163, 2020.
14. D. Fisman, O. Kupferman, S. Sheinvald, and M.Y. Vardi. A framework for inherent vacuity. In *4th International Haifa Verification Conference*, volume 5394 of *Lecture Notes in Computer Science*, pages 7–22. Springer, 2008.
15. S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *SIAM J. Comput.*, 45(3):882–929, 2016.
16. K. Greimel, R. Bloem, B. Jobstmann, and M. Vardi. Open implication. In *Proc. 35th Int. Colloq. on Automata, Languages, and Programming*, volume 5126 of *Lecture Notes in Computer Science*, pages 361–372. Springer, 2008.
17. T.A. Henzinger, S.C. Krishnan, O. Kupferman, and F.Y.C. Mang. Synthesis of uninitialized systems. In *Proc. 29th Int. Colloq. on Automata, Languages, and*

Programming, volume 2380 of *Lecture Notes in Computer Science*, pages 644–656. Springer, 2002.
18. O. Kupferman and O. Leshkowitz. Synthesis of privacy-preserving systems. In *Proc. 42nd Conf. on Foundations of Software Technology and Theoretical Computer Science*, volume 250 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 42:1–42:23, 2022.
19. O. Kupferman, Y. Lustig, M.Y. Vardi, and M. Yannakakis. Temporal synthesis for bounded systems and environments. In *Proc. 28th Symp. on Theoretical Aspects of Computer Science*, pages 615–626, 2011.
20. O. Kupferman and A. Rosenberg. The blow-up in translating LTL to deterministic automata. In *Proc. 6th Workshop on Model Checking and Artificial Intelligence*, volume 6572 of *Lecture Notes in Artificial Intelligence*, pages 85–94. Springer, 2010.
21. O. Kupferman and M.Y. Vardi. Synthesis with incomplete information. In *Advances in Temporal Logic*, pages 109–127. Kluwer Academic Publishers, 2000.
22. O. Kupferman and M.Y. Vardi. From linear time to branching time. *ACM Transactions on Computational Logic*, 6(2):273–294, 2005.
23. N. Piterman. From nondeterministic Büchi and Streett automata to deterministic parity automata. In *Proc. 21st IEEE Symp. on Logic in Computer Science*, pages 255–264. IEEE press, 2006.
24. A. Pnueli. The temporal semantics of concurrent programs. *Theoretical Computer Science*, 13:45–60, 1981.
25. A. Pnueli and R. Rosner. On the synthesis of a reactive module. In *Proc. 16th ACM Symp. on Principles of Programming Languages*, pages 179–190, 1989.
26. M.O. Rabin. Automata on infinite objects and Church's problem. *Amer. Mathematical Society*, 1972.
27. J.H. Reif. The complexity of two-player games of incomplete information. *Journal of Computer and Systems Science*, 29:274–301, 1984.
28. R. Rosner. *Modular Synthesis of Reactive Systems*. PhD thesis, Weizmann Institute of Science, 1992.
29. S. Safra. On the complexity of ω-automata. In *Proc. 29th IEEE Symp. on Foundations of Computer Science*, pages 319–327, 1988.
30. S. Schewe and B. Finkbeiner. Bounded synthesis. In *5th Int. Symp. on Automated Technology for Verification and Analysis*, volume 4762 of *Lecture Notes in Computer Science*, pages 474–488. Springer, 2007.
31. M.Y. Vardi and P. Wolper. Reasoning about infinite computations. *Information and Computation*, 115(1):1–37, 1994.
32. Y. Wu, V. Raman, B.C. Rawlings, S. Lafortune, and S.A. Seshia. Synthesis of obfuscation policies to ensure privacy and utility. *Journal of Automated Reasoning*, 60(1):107–131, 2018.

□