Games for Security under Adaptive Adversaries

Timos Antonopoulos
Yale University
New Haven, Connecticut, 06511, USA
Email: timos.antonopoulos@yale.edu

Tachio Terauchi
Waseda University
Shinjuku-ku, Tokyo, 169-8555, Japan
Email: terauchi@waseda.jp

***Abstract*—This work explores methods for proving and disproving security of systems under *adaptive* adversaries. Adaptive adversaries are ones which make their next move based on the previous observations. Our first contribution is a new *game* based characterization of security. We show that the game accurately captures security of deterministic and probabilistic systems against adaptive (probabilistic) adversaries. In addition, we build on top of the game characterization and present techniques that expedite proving the existence of attacker and defender strategies, and consequently proving security or vulnerability of systems. The first is what we call attack (and defense) *slopes* which give simple sufficient criteria for existence of winning strategies (for attacker and defender). The second is *reductions* of one game to another achieved by mapping a strategy of one to that of the other. We show that such reductions can prove or disprove security by reducing from a game of a secure system or reducing to that of a non-secure system.**

I. Introduction

This work is concerned with analyzing systems that operate on secret information, such as users' passwords. *Information flow analysis* [6], [16], [22], [24] is a well-established approach to detecting or proving the absence of attacks that steal the secrets by observing the execution behavior. For instance, *non-interference*, which says that there is no information leakage from observable channels, has been used to check security against timing channel attacks [1], [5]. Further, *quantitative information flow* is an extension of non-interference that maps observable channels to the quantity of information leak, where information leak is formalized by information-theoretic notions such as Shannon entropy [4], [8], [19], [25]. It has been applied to measure the degree of security of systems against various classes of attacks [7], [9], [11], [27].

Program `Leaky Login`

```
def LL(password, guess):
  for (int i = 0; i < guess.length; ++i):
    if (password[i] != guess[i]):
      return false
  return true
```

Fig. 1: Leaky login program

This work concerns analyzing the security of systems under *adaptive* attacks in which the adversaries can make their next move based on the previous observations. By itself, information flow analysis is often insufficient for accurately detecting the possibility of adaptive attacks or proving the absence thereof. For instance, consider the code snippet shown in Fig. 1, adopted from [5], [12]. The function `LL` is a login program that checks if the input `guess` matches the secret `password`. Note that the running time of the program is proportional to the length of the common prefix of `password` and `guess`. Therefore, the program is insecure against an adaptive attacker who can observe the running times as he will be able to reveal the elements of `password` iteratively by crafting appropriate inputs to be used as `guess` each time.[1]

More concretely, suppose that the alphabet of the password is the binary $\{0,1\}$ for simplicity. Then, in the setting of information flow, the situation is formalized as the program $P_{\mathsf{ins}}(s,v)$ that takes secret input s and public input v where $s,v \in \{0,1\}^\ell$, ℓ is the length of the password, and $P_{\mathsf{ins}}(s,v)$ outputs the length of the common prefix of s and v, along with a boolean that expresses whether s is equal to v. P_{ins} is not non-interferent [6], [16], [22], [24] because there exist inputs v, s_1, s_2 such that $P_{\mathsf{ins}}(s_1,v) \neq P_{\mathsf{ins}}(s_2,v)$, for example, $v = 0000$, $s_1 = 0100$, and $s_2 = 0010$. (Formally, P is said to be *non-interferent* if and only if $P(s_1,v) = P(s_2,v)$ for all public inputs v and secret inputs s_1 and s_2.) In this way, the non-interference based method of [5] is able to successfully detect the vulnerability. However, consider a *correct* login program that only reveals whether the guess matches the password or not. Such a program can be described as $P_{\mathsf{sec}}(s,v)$ where $P_{\mathsf{sec}}(s,v) = \mathsf{true}$ if $s = v$ and $P_{\mathsf{sec}}(s,v) = \mathsf{false}$ otherwise. Non-interference is insufficient to show the security of P_{sec}. Indeed, $P_{\mathsf{sec}}(s_1,v) \neq P_{\mathsf{sec}}(s_2,v)$ for any $v = s_1 \neq s_2$.

Quantitative information flow (QIF) has been proposed as a formalism to cope with situations such as the above [4], [8], [19], [25].[2] QIF maps programs to real numbers denoting the quantity of information leak so that the larger the leak quantity the less secure the program. For example, the *capacity* QIF measure defines information leak to be the number of different outputs obtained by varying the secret input, maximum over all public inputs. (Incidentally, capacity is equivalent to the maximum Shannon entropy over the distributions of secrets, and to min entropy for uniformly distributed secrets [19].[3]) Note that the capacity is 1 for non-interferent programs, 2 for

[1] Often, the output of the program (*regular channel*) is treated separately from observations related to the running time (*side channel*), but in this work both are treated the same, bundled as an *observation*.

[2] Another, orthogonal, approach is *declassification* which permits information marked as "declassified" to leak for free [13], [17], [26] (also implicitly used in [5]). However, the security would now depend on the correctness of the declassification policy.

[3] Technically, capacity is defined to be the log of the number of outputs.

P_{sec}, and ℓ for P_{ins}. Hence, based on the capacity measure, we may conclude that P_{sec} is reasonably secure, as it only leaks a constant amount of information regardless of the secret size. However, it is easy to construct a situation in which the adversary is able to efficiently decipher the secret despite the program only having a constant capacity QIF. For instance, let $P_{\mathsf{lq}}(s, v)$ be such that $s \in \{0,1\}^\ell$, $v \in \{1, ..., \ell\}$, and $P_{\mathsf{lq}}(s, v) = s[v-1]$. That is, $P_{\mathsf{lq}}(s, v)$ leaks the v-th element of s (see Fig. 2 in Section V). The capacity of P_{lq} is 2 as there are only two possible outputs, but P_{lq} is efficiently attackable by simply querying the secret one element at a time.

To address the issue, researchers have suggested to extend the information flow analysis approach to compute QIF over multiple attacker queries [15], and more generally over adaptive attack strategies [7], [10], [11]. The latter works introduced the notion of *attack tree* which is an important conceptual device for reasoning about security against adaptive attacks.

In this paper, we build on the previous works to explore new methods for accurately analyzing the security of systems under adaptive adversaries. Our first contribution is a new *game*-based characterization of security, in which there are two players *attacker* and *defender* such that the existence of a winning strategy for the attacker implies that the system is insecure and that for the defender implies that it is secure. The game is inspired by the notion of attack trees mentioned above. However, whereas the previous works have used the idea to compute QIF for a particular attack strategy or maximum QIF over all attack strategies of some bounded lengths, we show that our game is able to accurately capture the existence or the impossibility of efficient attacks, where efficiency is measured by bounds on the number of attack steps and the probability of the attack success. Importantly, our approach is able to derive bounds that are *parametric* to size of the secrets.

Further, building on the game-based characterization, we propose approximation techniques, called *attack slopes* and *defense slopes*, that can be used to show the existence of attacker and defender winning strategies. Currently, the slope technique is limited to deterministic systems and uniformly-distributed secrets, but we expect it to be extendable to more general settings. Finally, we continue capitalizing on the notion of game strategies to propose a form of *reduction* from one game to another, that aims to reuse the work done on proving a system secure or vulnerable, on the analysis of new systems.

The main contributions of the paper are summarized below.

(1) A new game-based characterization of security under adaptive adversaries (Section III).
(2) Approximation techniques called attack slopes and defense slopes (Section V).
(3) Reductions between games that can be used to show that one game (and hence the corresponding system) is less or more secure than the other (Section VI).

The rest of the paper is organized as follows. Section II provides some preliminary definitions. In Section III, we define the games and prove that they accurately characterize the security of systems against adaptive adversaries. Section IV illustrates the games on deterministic systems with uniformly-distributed secrets, and Section V presents the slope-based approximations for such systems. Section VI presents the game reductions. We discuss related work in Section VII and conclude the paper in Section VIII.

II. Preliminaries

We write $\mathbb{N}$ for the set of natural numbers $\{0, 1, \ldots\}$, and $\mathbb{N}^+$ for the set of positive natural numbers $\mathbb{N} \setminus \{0\}$. $\mathbb{R}$ is the set of real numbers, and we denote by $[r_1, r_2]$ the set of real numbers $\{r \in \mathbb{R} \mid r_1 \leq r \leq r_2\}$. For a set D, we write D^* for the set of finite sequences of elements from D, and denote elements of D^* with the vector notation $\vec{d}$. We denote the length of such a sequence with $|\vec{d}|$. If $\vec{d} \in D^*$ and $b \in D$, we denote with $\vec{d} \cdot b$ the sequence obtained by appending b to the sequence $\vec{d}$, resulting in a sequence of length $|\vec{d}| + 1$. We denote with $\underline{\varepsilon}$ the empty sequence.

As standard, we assume *security parameter* that describes the size of secrets. A *system* (or *program*) P is an indexed family of functions (indexed by the security parameter) that represent the input-output relation of the system. $\mathcal{S}$ is a security-parameter-indexed family of sets of *secrets*, $\mathcal{I}$ is a security-parameter-indexed family of sets of *attacker-controlled inputs*, and $\mathcal{O}$ is a security-parameter indexed family of *observations*. A security parameter is a natural number that represents the size of secrets, and we write $\mathcal{S}_\ell$ for the set of secrets of size ℓ, and $\mathcal{I}_\ell$ and $\mathcal{O}_\ell$ for the corresponding attacker inputs and observations. Each indexed function P_ℓ is a function from $\mathcal{S}_\ell \times \mathcal{I}_\ell$ to probability distributions over $\mathcal{O}_\ell$. That is, for each $(s, v) \in \mathcal{S}_\ell \times \mathcal{I}_\ell$, $P_\ell(s, v) = \mu$ for some $\mu : \mathcal{O}_\ell \to [0, 1]$ such that $\sum_{o \in \mathcal{O}_\ell} \mu(o) = 1$. We omit the security parameter when it is clear from the context and write $\mathcal{S}$, $\mathcal{I}$, $\mathcal{O}$, and P for $\mathcal{S}_\ell$, $\mathcal{I}_\ell$, $\mathcal{O}_\ell$, and P_ℓ, respectively.

The *support* of a probability distribution $\mu : \mathcal{O} \to [0, 1]$ is defined as standard: $\mathsf{supp}(\mu) = \{o \in \mathcal{O} \mid \mu(o) > 0\}$. For a program P, $v \in \mathcal{I}$, and $o \in \mathcal{O}$, we write $P^{-1}(v, o)$ for the set $\{s \in \mathcal{S} \mid o \in \mathsf{supp}(P(s, v))\}$. Roughly, $P^{-1}(v, o)$ is the set of secrets that have a non-zero probability of making P output o on input v. We also extend the notation to a sequence of inputs and observations so that for $\vec{v} = v_1, \ldots, v_n$ and $\vec{o} = o_1, \ldots, o_n$, $P^{-1}(\vec{v}, \vec{o})$ is the set of secrets that have a non-zero probability of making P output o_i on input v_i for each $i \in \{1, \ldots, n\}$. Formally, $P^{-1}(\vec{v}, \vec{o})$ is defined inductively by: $P^{-1}(\underline{\varepsilon}, \underline{\varepsilon}) = \mathcal{S}$ and $P^{-1}(\vec{v} \cdot v, \vec{o} \cdot o) = P^{-1}(\vec{v}, \vec{o}) \cap P^{-1}(v, o)$.

We say that P_ℓ is *deterministic* if for all $(s, v) \in \mathcal{S}_\ell \times \mathcal{I}_\ell$, $P_\ell(s, v)$ is a point mass (*i.e.* $P_\ell(s, v)(o) = 1$ for some $o \in \mathcal{O}_\ell$). We say that P is deterministic when P_ℓ is deterministic for each ℓ. In literature, deterministic systems are sometimes also called *noiseless* to contrast with the case where the attacker's observation is subject to noise and cannot be made exact. For example, often in timing attacks, the attacker is not able to know exactly how many loop iterations have taken place. Such noisy observations may be modeled by the more general probabilistic systems.

For any two functions $f : \mathbb{N}^+ \to \mathbb{R}$ and $g : \mathbb{N}^+ \to \mathbb{R}$, we denote with $(f \mathbin{\mathsf{op}} g)$, for $\mathsf{op} \in \{+, -, \times, /\}$, the function of

type $\mathbb{N}^+ \to \mathbb{R}$ that maps any $n \in \mathbb{N}^+$ to $f(n)$ op $g(n)$. For two functions $f : \mathbb{N} \to \mathbb{R}$ and $g : \mathbb{N} \to \mathbb{R}$, we say that $f > g$, if for all $n \in \mathbb{N}$, $f(n) > g(n)$. By abuse of notation, we often implicitly treat an expression e on the security parameter ℓ as the function $\lambda \ell \in \mathbb{N}^+.e$.

An *attacker* is a randomized algorithm $\mathcal{A}$ that attempts to discover the secret by making some number of queries to the system. As standard, we assume that $\mathcal{A}$ has the full knowledge of the system. Formally, $\mathcal{A}$ takes as inputs a function $\eta : \mathcal{I}_\ell \to \mathcal{O}_\ell$ and a natural number $n \in \mathbb{N}$ and outputs an element of $\mathcal{S}_\ell$. We restrict $\mathcal{A}(\eta, n)$ to call η at most n many times. As standard, $\mathcal{A}$ is only allowed to use η "extensionally" as an oracle, and it is not allowed to see the internals of η. Intuitively, $\mathcal{A}$ is not allowed to see the internals of the system once a secret has been fixed. Namely, η will be of the form $\lambda v \in \mathcal{I}_\ell.P_\ell(s, v)$ for some $s \in \mathcal{S}_\ell$ that the attacker aims to recover. We impose no restriction on how the attacker chooses the inputs to the system. Importantly, he may choose the inputs based on the outputs of previous oracle queries. Such an attacker is said to be *adaptive* [11].

The set of secrets $\mathcal{S}_\ell$ are equipped with probability distribution $\mu_\ell : \mathcal{S}_\ell \mapsto [0, 1]$. For a distribution of secrets μ, we write $\Pr_{s \leftarrow \mu}[A]$ for the probability that the event A happens when the secret s is chosen randomly according to the distribution μ. Therefore, for instance, $\Pr_{s \leftarrow \mu}[\mathcal{A}(\lambda v \in \mathcal{I}_\ell.P_\ell(s, v), n) = s]$ is the probability that the attacker $\mathcal{A}$ recovers the secret s in n many queries to the system, when s is chosen randomly according to the distribution μ. We formalize the notion of security against adaptive attackers as follows.

Definition II.1 (Security)**.** Let $\epsilon : \mathbb{N} \to [0, 1]$, $f : \mathbb{N} \to \mathbb{N}$ be two mappings. We say that P is (f, ϵ)-*secure* if for any $\ell \in \mathbb{N}^+$ and any attacker $\mathcal{A}$, we have

$$\Pr_{s \leftarrow \mu_\ell}[\mathcal{A}(\lambda v.P(s, v), f(\ell)) = s] < \epsilon(\ell)$$

Otherwise, the program P is (f, ϵ)-*insecure*.

Notice that for any program P, and functions $\epsilon : \mathbb{N} \to [0, 1]$ and $f : \mathbb{N} \to \mathbb{N}$ the following hold:

- If P is (f, ϵ)-secure, then it is also (f, ϵ')-secure for any $\epsilon' : \mathbb{N} \to [0, 1]$ such that $\epsilon < \epsilon'$.
- If P is (f, ϵ)-insecure, then it is also (f', ϵ)-insecure for any $f' : \mathbb{N} \to \mathbb{N}$ such that $f < f'$.

It should be noted that this definition of security is used in our earlier work [23] and it also closely corresponds to the definition used in the DARPA STAC program [20].

Given an attacker algorithm $\mathcal{A}$, we denote with $\mu_\mathcal{A}$ the probability distribution with which the attacker selects a public input $w \in \mathcal{I}$ to feed into the program and observe the output. More formally, for any $\vec{v} \in \mathcal{I}^*$, $\vec{o} \in \mathcal{O}^*$ with $|\vec{v}| = |\vec{o}|$ and $w \in \mathcal{I}$, we denote with $\mu_\mathcal{A}[\vec{v}, \vec{o}](w)$ the probability that $\mathcal{A}$ will choose $w \in \mathcal{I}$ for the next input, after having observed the sequence $\vec{o}$ when having fed to the program the sequence of inputs $\vec{v}$ in the previous $|\vec{v}|$ queries. By defining the attacker algorithm in such a way, we allow both deterministic and randomized attacker algorithms.

III. Games

Definitions: For a program P, $s \in \mathcal{S}_\ell$, $v \in \mathcal{I}$, and $o \in \mathcal{O}$ we denote with $\mu(o \mid s, v)$ the probability that $P(s, v)$ will output o. That is, $\mu(o \mid s, v) = P(s, v)(o)$. We use $\mu(s \mid \vec{v}, \vec{o})$ to denote the probability that the secret is s conditionally on having observed the sequence $\vec{o}$ when given inputs $\vec{v}$, where $|\vec{v}| = |\vec{o}|$. Using Bayes' theorem, it can be shown that for any $s \in \mathcal{S}_\ell$, $\vec{v} \in \mathcal{I}^*$, $w \in \mathcal{I}$, $\vec{o} \in \mathcal{O}^*$, and $q \in \mathcal{O}$,

$$\mu(s \mid \vec{v} \cdot w, \vec{o} \cdot q) = \frac{\mu(q \mid s, w) \cdot \mu(s \mid \vec{v}, \vec{o})}{\sum_{s' \in \mathcal{S}_\ell} \mu(q \mid s', w) \cdot \mu(s' \mid \vec{v}, \vec{o})}. \quad (1)$$

Here, $\mu(s \mid \underline{\varepsilon}, \underline{\varepsilon}) = \mu_\ell(s)$ where μ_ℓ is the prior probability distribution associated with $\mathcal{S}_\ell$ (cf. Section II).

For $\vec{v} \in \mathcal{I}^*$, $w \in \mathcal{I}$, $\vec{o} \in \mathcal{I}^*$, and $q \in \mathcal{O}$, where $|\vec{v}| = |\vec{o}|$, we denote with $\mu(q \mid w; \vec{v}, \vec{o})$ the probability of observing the output q, by feeding the program the input w, under the condition that we have already observed the sequence of observations $\vec{o}$ given inputs $\vec{v}$. More formally, the definition of $\mu(q \mid w; \vec{v}, \vec{o})$ is

$$\mu(q \mid w; \vec{v}, \vec{o}) = \sum_{s \in \mathcal{S}_\ell} \mu(s \mid \vec{v}, \vec{o}) \cdot \mu(q \mid s, w). \quad (2)$$

Thus, from (1) and (2), for any $s \in \mathcal{S}_\ell$,

$$\mu(s \mid \vec{v} \cdot w, \vec{o} \cdot q) = \frac{\mu(q \mid s, w) \cdot \mu(s \mid \vec{v}, \vec{o})}{\mu(q \mid w; \vec{v}, \vec{o})}, \quad (3)$$

and therefore, for any $s \in \mathcal{S}_\ell$, where $\mu(s \mid \vec{v} \cdot w, \vec{o} \cdot q) \neq 0$,

$$\mu(q \mid w; \vec{v}, \vec{o}) = \frac{\mu(q \mid s, w) \cdot \mu(s \mid \vec{v}, \vec{o})}{\mu(s \mid \vec{v} \cdot w, \vec{o} \cdot q)}.$$

Note that for any $s \in \mathcal{S}_\ell$, if $\mu(s \mid \vec{v} \cdot w, \vec{o} \cdot q) \neq 0$ then $\mu(s \mid \vec{v}, \vec{o}) \neq 0$ and $\mu(q \mid s, w) \neq 0$.

Game: We define the n-round r-confidence game, for $n \in \mathbb{N}$ and $r \in [0, 1]$ as follows. The game is played by two players, *Attacker* and *Defender*.[4] At any round $i \in \mathbb{N}$ of the game, we denote the position by $\langle \vec{v}, \vec{o}, r_i \rangle$, where $|\vec{v}| = |\vec{o}| = i$ and $r_i \in [0, 1]$. At round 0, the position is $\langle \underline{\varepsilon}, \underline{\varepsilon}, r_0 \rangle$, where r_0 is the confidence parameter r. At each round, from position $\langle \vec{v}, \vec{o}, r_i \rangle$, Attacker makes the first move, and chooses $w \in \mathcal{I}$ and a function $r_w : \mathcal{O} \to [0, 1]$, such that $r_i = \sum_{q \in \mathcal{O}} \mu(q \mid w; \vec{v}, \vec{o}) \cdot r_w(q)$. Defender then replies with an observation $q \in \mathcal{O}$. The new position is then $\langle \vec{v} \cdot w, \vec{o} \cdot q, r_w(q) \rangle$. In other words, the input w is appended to the vector of inputs, the observation q is appended to the vector of observations, and the confidence parameter r_{i+1} is set to $r_w(q)$. Defender wins the game at round n, at position $\langle \vec{v}, \vec{o}, r_n \rangle$ exactly when $\max_{s \in \mathcal{S}_\ell} \mu(s \mid \vec{v}, \vec{o}) < r_n$. Otherwise, Attacker wins at round n.

Remark III.1. Defender, when it's time for her move, can choose any $q \in \mathcal{O}$, without any condition on the probability with which q may be observed at that point. In fact, the probability of q being observed on public input $w \in \mathcal{I}$, having already observed $\vec{o}$ on inputs $\vec{v}$, could just as well be 0. In other

[4]For convenience, we choose Defender to be female and Attacker to be male.

words $\mu(q \mid w; \vec{v}, \vec{o})$ could be 0. Notice however, that for all such $q \in \mathcal{O}$, Attacker can set $r_w(q)$ to 0, without affecting the sum $\sum_{q \in \mathcal{O}} \mu(q \mid w; \vec{v}, \vec{o}) \cdot r_w(q)$, and as such without affecting the condition for constructing r_w. If Defender chooses such a q, then she is bound to lose the game because after any subsequent moves, $\max_{s \in \mathcal{S}_\ell} \mu(s \mid \vec{v}, \vec{o})$ will not be strictly less than $r_w(q) = 0$ at the end.

As a result, at each round, after a move $w \in \mathcal{I}$ by Attacker, we can think of Defender as internally choosing a secret $s \in \mathcal{S}$, that is probabilistically still possible, and then choosing an observation q that is possible after executing the underlying program P on inputs w and s, *without* disclosing what s is. Under this interpretation, notice that Defender is allowed to keep changing her secret s as the game progresses. ▲

A strategy for a player is a set of rules that describes how the player moves given the history of moves played already. We say that Defender wins the n-round r-confidence game, if she has a strategy to do so for any Attacker moves, starting at position $\langle \underline{\varepsilon}, \underline{\varepsilon}, r \rangle$. Similarly, we say Attacker wins the n-round r-confidence game if he has a strategy to do so, for any Defender moves.

In Section IV, we go through an example for a simplified version of this game, corresponding to deterministic systems and uniformly-distributed secrets. As stated in the theorem below, the game completely characterizes (f, ϵ)-security.

Theorem III.2. *A program is (f, ϵ)-secure if and only if, for all $\ell \in \mathbb{N}^+$, Defender wins the $f(\ell)$-round $\epsilon(\ell)$-confidence game.*

Proof: Given $\vec{v} \in \mathcal{I}^*$ and $\vec{o} \in \mathcal{O}^*$, we denote with $\mu[\vec{v}, \vec{o}]$ the probability distribution over $\mathcal{S}_\ell$, conditionally upon having observed $\vec{o}$ when given inputs $\vec{v}$. Formally, for any $s \in \mathcal{S}_\ell$, $\mu[\vec{v}, \vec{o}](s) = \mu(s \mid \vec{v}, \vec{o})$. For what follows, we fix arbitrary $\ell \in \mathbb{N}^+$.

We will show by induction on n, that for all $n \in \mathbb{N}^+$, $r \in [0, 1]$ and for all $\vec{v} \in \mathcal{I}^*$ and $\vec{o} \in \mathcal{O}^*$, it holds that $\Pr_{s \leftarrow \mu[\vec{v}, \vec{o}]}[\mathcal{A}(\lambda v.P(s, v), n) = s] < r$ for any attacker $\mathcal{A}$, if and only if Defender wins the n-round r-confidence game from position $\langle \vec{v}, \vec{o}, r \rangle$ on P.

For the base case, we have that $n = 0$. Then it holds that $\Pr_{s \leftarrow \mu[\vec{v}, \vec{o}]}[\mathcal{A}(\lambda v.P(s, v), n) = s] < r$ for any attacker $\mathcal{A}$ if and only if $\max_{s \in \mathcal{S}} \mu(s \mid \vec{v}, \vec{o}) < r$. That is because in the worst case Attacker will choose the most likely secret s from $P^{-1}(\vec{v}, \vec{o})$, and succeed with probability $\mu(s \mid \vec{v}, \vec{o})$. We proceed to the inductive case. For $s \in \mathcal{S}$, let $\eta_s = \lambda v.P(s, v)$. Furthermore, for any attacker $\mathcal{A}$, and any $w \in \mathcal{I}, q \in \mathcal{O}$, we denote with $\mathcal{A}_{(w,q)}$ the attacker that corresponds to how the attacker $\mathcal{A}$ operates after observing $q \in \mathcal{O}$ on their query to η_s with $w \in \mathcal{I}$. Then, notice that

$$
\begin{aligned}
& \Pr_{s \leftarrow \mu[\vec{v}, \vec{o}]}[\mathcal{A}(\eta_s, n) = s] \\
= \; & \textstyle\sum_{s \in \mathcal{S}} \mu(s \mid \vec{v}, \vec{o}) \cdot \Pr[\mathcal{A}(\eta_s, n) = s] \\
= \; & \textstyle\sum_{s \in \mathcal{S}} \mu(s \mid \vec{v}, \vec{o}) \cdot \sum_{w \in \mathcal{I}} \mu_{\mathcal{A}}[\vec{v}, \vec{o}](w) \cdot \\
& \qquad \textstyle\sum_{q \in \mathcal{O}} \mu(q \mid s, w) \cdot \Pr[\mathcal{A}_{(w,q)}(\eta_s, n-1) = s] \\
= \; & \textstyle\sum_{s \in \mathcal{S}} \sum_{w \in \mathcal{I}} \sum_{q \in \mathcal{O}} \mu(s \mid \vec{v}, \vec{o}) \cdot \mu_{\mathcal{A}}[\vec{v}, \vec{o}](w) \cdot \\
& \qquad \mu(q \mid s, w) \cdot \Pr[\mathcal{A}_{(w,q)}(\eta_s, n-1) = s].
\end{aligned}
$$

Then, since $\mu(s \mid \vec{v} \cdot w, \vec{o} \cdot q) = \frac{\mu(q \mid s, w) \cdot \mu(s \mid \vec{v}, \vec{o})}{\mu(q \mid w; \vec{v}, \vec{o})}$ by Equation (3), we have that the above is equal to

$$
\begin{aligned}
& \textstyle\sum_{s \in \mathcal{S}} \sum_{w \in \mathcal{I}} \sum_{q \in \mathcal{O}} \mu_{\mathcal{A}}[\vec{v}, \vec{o}](w) \cdot \mu(q \mid w; \vec{v}, \vec{o}) \cdot \\
& \qquad \mu(s \mid \vec{v} \cdot w, \vec{o} \cdot q) \Pr[\mathcal{A}_{(w,q)}(\eta_s, n-1) = s] \\
= \; & \textstyle\sum_{w \in \mathcal{I}} \sum_{q \in \mathcal{O}} \mu_{\mathcal{A}}[\vec{v}, \vec{o}](w) \cdot \mu(q \mid w; \vec{v}, \vec{o}) \cdot \\
& \qquad \textstyle\sum_{s \in \mathcal{S}} \mu(s \mid \vec{v} \cdot w, \vec{o} \cdot q) \Pr[\mathcal{A}_{(w,q)}(\eta_s, n-1) = s] \\
= \; & \textstyle\sum_{w \in \mathcal{I}} \sum_{q \in \mathcal{O}} \mu_{\mathcal{A}}[\vec{v}, \vec{o}](w) \cdot \mu(q \mid w; \vec{v}, \vec{o}) \cdot \\
& \qquad \Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}_{(w,q)}(\eta_s, n-1) = s] \\
= \; & \textstyle\sum_{w \in \mathcal{I}} \mu_{\mathcal{A}}[\vec{v}, \vec{o}](w) \cdot \sum_{q \in \mathcal{O}} \mu(q \mid w; \vec{v}, \vec{o}) \cdot \\
& \qquad \Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}_{(w,q)}(\eta_s, n-1) = s].
\end{aligned}
$$

For any $w \in \mathcal{I}$, let $r_w : \mathcal{O} \to [0, 1]$ be any function such that $r = \sum_{q \in \mathcal{O}} \mu(q \mid w; \vec{v}, \vec{o}) \cdot r_w(q)$.
Claim: The value of

$$
\begin{aligned}
& \textstyle\sum_{w \in \mathcal{I}} \mu_{\mathcal{A}}[\vec{v}, \vec{o}](w) \cdot \sum_{q \in \mathcal{O}} \mu(q \mid w; \vec{v}, \vec{o}) \cdot \\
& \qquad \Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}_{(w,q)}(\eta_s, n-1) = s]
\end{aligned}
$$

is less than r for any attacker $\mathcal{A}$, if and only if for any $w \in \mathcal{I}$,

$$
\sum_{q \in \mathcal{O}} \mu(q \mid w; \vec{v}, \vec{o}) \cdot \Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}_{(w,q)}(\eta_s, n-1) = s] < r,
$$

for any attacker $\mathcal{A}$.
Proof of Claim: Suppose that the sum above is less than r for any attacker $\mathcal{A}$. Suppose for contradiction that there exist $w \in \mathcal{I}$, and attacker $\mathcal{A}$, such that

$$
\sum_{q \in \mathcal{O}} \mu(q \mid w; \vec{v}, \vec{o}) \cdot \Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}_{(w,q)}(\eta_s, n-1) = s] \geq r.
$$

Then define the attacker $\mathcal{A}$ to be such that $\mu_{\mathcal{A}}[\vec{v}, \vec{o}](w) = 1$ and $\mu_{\mathcal{A}}[\vec{v}, \vec{o}](w') = 0$ for all $w' \neq w$. Clearly then the value of the expression above is greater than or equal to r for some attacker $\mathcal{A}$, which is a contradiction. For the other direction, if for all $w \in \mathcal{I}$ it holds that

$$
\sum_{q \in \mathcal{O}} \mu(q \mid w; \vec{v}, \vec{o}) \cdot \Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}_{(w,q)}(\eta_s, n-1) = s] < r,
$$

for any attacker $\mathcal{A}$, then the expression above is less than r. ■ (Claim)
Claim: It holds that for all w in $\mathcal{I}$ and all attackers $\mathcal{A}$,

$$
\sum_{q \in \mathcal{O}} \mu(q \mid w; \vec{v}, \vec{o}) \cdot \Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}_{(w,q)}(\eta_s, n-1) = s] < r,
$$

if and only if, for all $w \in \mathcal{I}$ with respective r_w, there is a $q \in \mathcal{O}$ with $\mu(q \mid w; \vec{v}, \vec{o}) > 0$ such that for any attacker $\mathcal{A}$

$$
\Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}(\eta_s, n-1) = s] < r_w(q).
$$

Proof of Claim: By definition, for any r_w it holds that $r = \sum_{q \in \mathcal{O}} \mu(q \mid w; \vec{v}, \vec{o}) \cdot r_w(q)$. For the *only if* direction, suppose that for all $w \in \mathcal{I}$ and all attackers $\mathcal{A}$,

$$
\sum_{q \in \mathcal{O}} \mu(q \mid w; \vec{v}, \vec{o}) \cdot \Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}_{(w,q)}(\eta_s, n-1) = s] < r.
$$

Then for all $w \in \mathcal{I}$, any r_w that satisfies the game condition and any attacker $\mathcal{A}$,

$$\sum_{q \in \mathcal{O}} \mu(q \mid w; \vec{v}, \vec{o}) \cdot \Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}_{(w,q)}(\eta_s, n-1) = s] < \sum_{q \in \mathcal{O}} \mu(q \mid w; \vec{v}, \vec{o}) \cdot r_w(q).$$

Therefore, for all $w \in \mathcal{I}$, $r_w : \mathcal{O} \to [0,1]$ and attacker $\mathcal{A}$, there is a $q \in \mathcal{O}$ such that

$$\Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}_{(w,q)}(\eta_s, n-1) = s] < r_w(q).$$

For each $q \in \mathcal{O}$, let $\mathcal{A}_q^+$ be the attacker that maximises their probability of success for the remaining $n-1$ moves under the probability distribution of secrets $\mu[\vec{v} \cdot w, \vec{o} \cdot q]$. Formally, let A be the collection of all attackers, and for any $q \in \mathcal{O}$ let $\mathcal{A}_q^+$ be the attacker such that

$$\Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}_q^+(\eta_s, n-1) = s] = \max_{\mathcal{A} \in \mathsf{A}} \Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}(\eta_s, n-1) = s]. \quad (4)$$

Let then $\mathcal{A}^+$ denote the attacker that behaves like $\mathcal{A}_q^+$ once the observation q is fixed, for every $q \in \mathcal{O}$. We know that for all $w \in \mathcal{I}$, $r_w : \mathcal{O} \to [0,1]$ and any attacker $\mathcal{A}$, there is a $q \in \mathcal{O}$ such that

$$\Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}_{(w,q)}(\eta_s, n-1) = s] < r_w(q).$$

For any $w \in \mathcal{I}$ and $r_w : \mathcal{O} \to [0,1]$, let $q \in \mathcal{O}$ be the observation that works for the attacker $\mathcal{A}^+$. Then by definition of $\mathcal{A}^+$ (Eq. 4), it follows that for any $w \in \mathcal{I}$ and $r_w : \mathcal{O} \to [0,1]$, $q \in \mathcal{O}$ is such that

$$\Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}(\eta_s, n-1) = s] < r_w(q).$$

for any attacker $\mathcal{A}$.

For the *if* direction, suppose that for some w in $\mathcal{I}$ and some attacker $\mathcal{A}$,

$$\sum_{q \in \mathcal{O}} \mu(q \mid w; \vec{v}, \vec{o}) \cdot \Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}_{(w,q)}(\eta_s, n-1) = s] \geq r.$$

Change $\mathcal{A}$ to $\mathcal{A}^-$ so that the inequality above is actually an equality. In other words, for some w in $\mathcal{I}$ and the attacker $\mathcal{A}^-$,

$$\sum_{q \in \mathcal{O}} \mu(q \mid w; \vec{v}, \vec{o}) \cdot \Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}^-_{(w,q)}(\eta_s, n-1) = s] = r.$$

Then define r_w to be such that for each $q \in \mathcal{O}$ with $\mu(q \mid w; \vec{v}, \vec{o}) > 0$,

$$r_w(q) = \Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}^-_{(w,q)}(\eta_s, n-1) = s].$$

Notice that since the sum above is equal to r, this mapping r_w satisfies the conditions of the game. But then notice that the following immediately follows. There is some $w \in \mathcal{I}$, some $r_w : \mathcal{O} \to [0,1]$ and the attacker $\mathcal{A}^-$ is such that for all $q \in \mathcal{O}$, if $\mu(q \mid w; \vec{v}, \vec{o}) > 0$ then

$$\Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}^-_{(w,q)}(\eta_s, n-1) = s] \geq r_w(q).$$

But then it also follows that there is some $w \in \mathcal{I}$, some $r_w : \mathcal{O} \to [0,1]$ such that for all $q \in \mathcal{O}$, there is an attacker $\mathcal{A}$, namely $\mathcal{A}^-_{(w,q)}$, such that if $\mu(q \mid w; \vec{v}, \vec{o}) > 0$ then

$$\Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}(\eta_s, n-1) = s] \geq r_w(q)$$

as required. ■ (Claim)

By the induction hypothesis, it holds that

$$\Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}(\eta_s, n-1) = s] < r_w(q),$$

for any attacker $\mathcal{A}$, if and only if Defender has a winning strategy for the $(n-1)$-round $r_w(q)$-confidence game from the position $\langle \vec{v} \cdot w, \vec{o} \cdot q, r_w(q) \rangle$. We showed that for any attacker $\mathcal{A}$, $\Pr_{s \leftarrow \mu[\vec{v}, \vec{o}]}[\mathcal{A}(\eta_s, n) = s] < r$ if and only if for all $w \in \mathcal{I}$ with respective r_w, there is a $q \in \mathcal{O}$ with $\mu(q \mid w; \vec{v}, \vec{o}) > 0$ such that for any attacker $\mathcal{A}$

$$\Pr_{s \leftarrow \mu[\vec{v} \cdot w, \vec{o} \cdot q]}[\mathcal{A}(\eta_s, n-1) = s] < r_w(q).$$

By the induction hypothesis it follows that the latter holds if and only if for all $w \in \mathcal{I}$ with respective r_w, there is a $q \in \mathcal{O}$ with $\mu(q \mid w; \vec{v}, \vec{o}) > 0$, such that Defender has a winning strategy for the $(n-1)$-round $r_w(q)$-confidence game from the position $\langle \vec{v} \cdot w, \vec{o} \cdot q, r_w(q) \rangle$. Finally, this holds if and only if Defender has a winning strategy for the n-round r-confidence game from the position $\langle \vec{v}, \vec{o}, r \rangle$, as required. ■

The intuition behind Attacker choosing a function r_w is as follows. When Attacker chooses a public input $w \in \mathcal{I}$, he does not know what $q \in \mathcal{O}$ Defender may reply with, so he assigns different success goals for each $q \in \mathcal{O}$ using the mapping $r_w : \mathcal{O} \to [0,1]$, and lets Defender know of how he has weighed all the different scenarios. Defender then, consults these targets, and replies with a $q \in \mathcal{O}$ trying to maximize her chances of winning. We remind the reader that Defender can reply with any $q \in \mathcal{O}$ irrespectively of the probability that such a q could be output by the program at input w. This is because these probabilities have been encoded in the constraints of how Attacker can construct the mapping r_w.

Remark III.3. It is worth noting that for the special case where $r = 1$, at each position $\langle \vec{v}, \vec{o}, r \rangle$, and for any move $w \in \mathcal{I}$, the only function r_w that satisfies the condition for Attacker's moves, is the one where $r_w(q) = 1$ for all $q \in \mathcal{O}$, for which $\mu(q \mid w; \vec{v}, \vec{o}) > 0$ holds. At the same time, as pointed out in Remark III.1, Defender should only consider responses q that could be observed on input w and some secret s that is still possible, or in other words q should be in $\mathsf{supp}(P(s, w))$ for some $s \in P^{-1}(\vec{v}, \vec{o})$. Therefore the game reduces to the simpler version where Attacker chooses $w \in \mathcal{I}$ and Defender chooses $s \in \mathcal{S}_\ell$ that is still in play, and replies with any observation $q \in \mathcal{O}$ possible by executing $P(s, w)$. ▲

Remark III.4. Attacker in the game does not need to consider probabilistic moves. As it can be seen by the proof of Theorem III.2, and in particular the second claim in it, there is a deterministic attack stategy that is optimal for Attacker.

IV. Deterministic Systems and Uniform Distributions

In this section, we consider a simplified version of a game in which programs are deterministic and the distribution of secrets is uniform. Also, for simplicity we assume that for each ℓ, the set of secrets $\mathcal{S}_\ell$ comprises binary strings of length ℓ, that is, $\mathcal{S}_\ell = \{0,1\}^\ell$ and $\mu_\ell(s) = 1/2^\ell$ for all $s \in \mathcal{S}_\ell$. Note that, in this setting, for any $s \in \mathcal{S}_\ell$ and $w \in \mathcal{I}$, $\mu(q \mid s, w)$ is equal to 1 for exactly one $q \in \mathcal{O}$, and 0 for all others. Similarly, it can be shown by induction, that for all $s \in \mathcal{S}_\ell$, $\vec{v} \in \mathcal{I}^*$ and $\vec{o} \in \mathcal{O}$, $\mu(s \mid \vec{v}, \vec{o})$ is equal to $\frac{1}{|P^{-1}(\vec{v},\vec{o})|}$ if $s \in P^{-1}(\vec{v}, \vec{o})$ and 0 otherwise. Therefore, by Eq. (2) we have that

$$\mu(q \mid w; \vec{v}, \vec{o}) = \frac{\frac{1}{|P^{-1}(\vec{v},\vec{o})|}}{\frac{1}{|P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q)|}} = \frac{|P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q)|}{|P^{-1}(\vec{v}, \vec{o})|},$$

when $P^{-1}(\vec{v}, \vec{o})$ and $P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q)$ are non-empty, and 0 otherwise. Let us call the setting *uniform-deterministic*.

For uniform-deterministic games, Theorem III.2 implies that a program P is (f, ϵ)-secure if and only if for all $\ell \in \mathbb{N}^+$, Defender wins the $f(\ell)$-round $\epsilon(\ell)$-confidence game, where at each round at position $\langle \vec{v}, \vec{o}, r\rangle$, Attacker has to ensure that $r = \sum_{q\in\mathcal{O}} \frac{|P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q)|}{|P^{-1}(\vec{v},\vec{o})|} \cdot r_w(q)$, and at the last round (at some position $\langle \vec{v}, \vec{o}, r\rangle$), it holds that $\frac{1}{|P^{-1}(\vec{v},\vec{o})|} < r$. Furthermore, when Defender internally selects a secret value s, this value directly determines the observation o she replies with, as for any $w \in \mathcal{I}$, $P(s, w)$ can only take one value.

Given the simpler setting, it is now easier to provide some intuition for the game and why Attacker is allowed to change the weight for the different choices by Defender, using a simple example. Consider a very simple game with only one round, where the initial set of secrets is $\mathcal{S}_4$ with 16 secrets, and there are exactly two possible observations o_1 and o_2 in $\mathcal{O}$, such that for any $v \in \mathcal{I}$, $|P^{-1}(v, o_1)| = 12$ and $|P^{-1}(v, o_2)| = 4$. In this game, Defender would always reply with the observation o_1, as the number of secrets compatible with that observation is a lot higher, and as a result $\mu(s \mid v, o_1)$ (equal to $\frac{1}{12}$) is a lot lower than $\mu(s \mid v, o_2)$ (equal to $\frac{1}{4}$). The actual probability Attacker can guess the secret is $\frac{1}{16}\cdot 12 \cdot \frac{1}{12} + \frac{1}{16}\cdot 4 \cdot \frac{1}{4} = \frac{1}{8}$ (after one round, the probability of correctly guessing the secret is $\frac{1}{12}$ for 12 of the 16 secrets and $\frac{1}{4}$ for 4 of them). Without any further condition, if Defender chose o_1 as her move, then she could make it seem like the probability of Attacker's success is $\frac{1}{12}$, which is less than the actual one (i.e., $\frac{1}{8}$).

In our games, Attacker has way of dealing with this issue, by defining the function $r_v : \mathcal{O} \to [0, 1]$. This way, Attacker can define $r_v(o_1)$ to be lower than $r_v(o_2)$, giving the incentive to Defender to also consider o_2 as a move. Concretely, suppose that Attacker chose the function r_v to be such that $r_v(o_1) = 0$. Then if Defender chose o_1 as her move, she would require $\frac{1}{|P^{-1}(v,o_1)|} < 0$ to win, which is not possible. She would then definitely choose o_2. It is, however, not to Attacker's advantage to choose such extreme valuations of r_v. Recall that the game enforces the constraint $r = \sum_{q\in\mathcal{O}} \mu(q \mid v; \vec{v}, \vec{o}) \cdot r_v(q)$ where $\langle \vec{v}, \vec{o}, r\rangle$ is the current game position. This implies that, in our example, r_v must satisfy $r = \sum_{i\in\{1,2\}} \mu(o_i \mid v; \varepsilon, \varepsilon) \cdot r_v(o_i)$ where r is the initial confidence parameter. With $r_v(o_1)$ being 0, $r_v(o_2)$ would have to be $\frac{1}{2}$, if he wants to win the $\frac{1}{8}$-confidence game and satisfy the constraint with his move. But with such a high value of $r_v(o_2)$, Defender can win that position since $\frac{1}{|P^{-1}(v,o_2)|} < \frac{1}{2}$. The best move for Attacker is to choose $r_v(o_1)$ to be $\frac{1}{12}$ and $r_v(o_2)$ to be $\frac{1}{4}$. This way for both o_i, $\frac{1}{|P^{-1}(v,o_i)|} \geq r_v(o_i)$, and

$$r_v(o_1) \cdot \tfrac{|P^{-1}(v,o_1)|}{|\mathcal{S}_4|} + r_v(o_2) \cdot \tfrac{|P^{-1}(v,o_2)|}{|\mathcal{S}_4|} = \\ = \tfrac{1}{12} \cdot \tfrac{12}{16} + \tfrac{1}{4} \cdot \tfrac{4}{16} = \quad \tfrac{1}{8}.$$

For the rest of this section, $\Pr_{s\leftarrow Q}[\mathcal{A}(\lambda v.P(s, v), n) = s]$ denotes the probability that the attacker algorithm can find the secret s (using at most n calls to the program), when the secret is drawn uniformly at random from the set Q. In contrast, for $\Pr[\mathcal{A}(\lambda v.P(s, v), n) = s]$, where the subscript $s \leftarrow Q$ is missing, denotes $\Pr_{s\leftarrow\{s\}}[\mathcal{A}(\lambda v.P(s, v), n) = s]$.

Example IV.1. Let P_{sec} be the program from Section I. Let $f(\ell) = 2 \cdot \ell$ and $\epsilon(\ell) = 1$ for all $\ell \in \mathbb{N}^+$. We show that the program is $(2 \cdot \ell, 1)$-secure, that is Attacker *cannot* find the secret with absolute certainty after $2 \cdot \ell$ many tries. In order for Defender to win the game, after $f(\ell)$ rounds, $\frac{1}{|P^{-1}_{\mathsf{sec}}(\vec{v},\vec{o})|}$ must be strictly less than 1, or in other words, $|P^{-1}_{\mathsf{sec}}(\vec{v}, \vec{o})|$must be strictly more than 1. That implies that the probability that the attacker algorithm will guess the secret is less than 1, as expected. ▲

We go through the simple example P_{ins} and show that it is vulnerable, with only a linear number of tries.

Proposition IV.2. *Attacker has a winning strategy for the $(\ell + 1)$-round 1-confidence game on program P_{ins}.*

Proof: We can show by induction on the number of rounds that Attacker has a strategy, such that at each round i, at position $\langle \vec{v}, \vec{o}, 1\rangle$, there exists $w \in \{0,1\}^{i-1}$ such that $P^{-1}_{\mathsf{ins}}(\vec{v}, \vec{o}) \subseteq \{w \cdot z \in \mathcal{S}_\ell \mid z \in \{0,1\}^{\ell-i+1}\}$. This implies that at round $\ell + 1$, $|P^{-1}_{\mathsf{ins}}(\vec{v}, \vec{o})| = 1$. We remind the reader, that according to Remark III.3, Attacker needs to choose only an input $w \in \mathcal{I}$ at each move, and not a mapping $r_w : \mathcal{O} \to [0, 1]$. For the base case, let $i = 1$. Then at the first round $P^{-1}_{\mathsf{ins}}(\varepsilon, \varepsilon) = \{z \in \mathcal{S}_\ell \mid z \in \{0,1\}^\ell\}$ as required. For the inductive case, suppose that the statement holds for all $i \leq I$, for some $I \in \mathbb{N}$, and consider the case where $i = I + 1$. Then, by the inductive hypothesis, there exists $w \in \{0,1\}^{I-1}$ such that $P^{-1}_{\mathsf{ins}}(\vec{v}, \vec{o}) \subseteq \{w \cdot z \in \mathcal{S}_\ell \mid z \in \{0,1\}^{\ell-I+1}\}$. Attacker then chooses v_I to be equal to any string that starts with w and Defender replies with $o_I \in \mathcal{O}$. Notice that if Defender replies with $o_I < |w| = I - 1$, then $P^{-1}_{\mathsf{ins}}(\vec{v}\cdot v_I, \vec{o}\cdot o_I)$ is empty, since for all $s \in P^{-1}_{\mathsf{ins}}(\vec{v}, \vec{o})$, $s = w \cdot z$ for some $z \in \{0,1\}^{\ell-I+1}$ and therefore for all such s, $\mathsf{prefix}(s, v_I) \geq |w|$. Assume then that Defender replies with $o_I \geq |w| = I - 1$. Let y_I be the prefix of v_I of length $o_I + 1$, and notice that by definition $P^{-1}_{\mathsf{ins}}(\vec{v}\cdot v_I, \vec{o}\cdot o_I)$ contains no secret with prefix y_I. Let w_I be equal to the string obtained from y_I by flipping the last bit in that string. Then w_I, of length $o_I + 1$, is such that $P^{-1}_{\mathsf{ins}}(\vec{v}\cdot v_I, \vec{o}\cdot o_I) \subseteq \{w_I \cdot z \in \mathcal{S}_\ell \mid z \in \{0,1\}^{\ell-o_I+1}\}$. Since

$o_I \geq I-1$, it follows that there exists u of length I, such that $P_{\mathsf{ins}}^{-1}(\vec{v}\cdot v_I, \vec{o}\cdot o_I) \subseteq \{u\cdot z \in \mathcal{S}_\ell \mid z \in \{0,1\}^{\ell-I}\}$ as required. ■

A. Adaptive vs. Non-adaptive attacks

Adaptive attacks are naturally a lot more powerful than non-adaptive attacks, and here we demonstrate some strong lower bounds for the gap between the number of moves or tries required by a non-adaptive attacker compared to the number of moves required by an adaptive attacker. First we need to define a non-adaptive game. Such a game would be like the one defined for adaptive attacks, but where Attacker's choices do not depend on Defender's moves at all. In such a game, Attacker can perform all his moves at the very beginning, and Defender can reply with her moves after Attacker has finished moving. The winning condition in the end is the same.

The following example illustrates the gap between the power of an adaptive attacker and a non-adaptive one. It is an adaptation of an example presented in [7], generalized to the setting where the security bounds are parametric to the size of secrets. Even though the generalization is easy to produce, we present it here for completeness.

Example IV.3. Let P_{dis} be a program that discloses all the secret information but in two steps, parameterized by the secret length ℓ. For any ℓ, the program has two inputs as usual, $v \in \mathcal{I}$ and $s \in \mathcal{S}$. For $s \in \mathcal{S}_\ell$, define s^1 and s^2 to be such that $s = s^1 \cdot s^2$, and $|s^1| = \ell - 1$. Given v, we define v^1 and v^2, similarly. The program P_{dis} then operates as follows on inputs v and s. If $v^1 \cdot v^2$ is equal to $s^1 \cdot s^2$ then the program returns 1. Otherwise, if v^1 is equal to s^1, the program returns s^2. Finally, if none of the above hold, the program returns s^1.

It is easy to see that an adaptive attacker can find the secret in 2 steps. First they try any input v, on which the program, in the worst case, will return the string s^1. At the second step, Attacker tries any string with prefix s^1, at which point the program will return s^2 and the secret is simply $s^1 \cdot s^2$. ▲

Lemma IV.4. *There exists program P such that for each $\ell \in \mathbb{N}$, Attacker wins the 2-round 1-confidence adaptive game on P, but for all ℓ larger than some $\ell_0 \in \mathbb{N}$, Attacker does not win the $(2^{\ell-1}-1)$-round 1-confidence non-adaptive game.*

Proof: Consider the program P_{dis} from Example IV.3. It was shown that Attacker can win the 2-round 1-confidence game. We need to show that for large enough $\ell \in \mathbb{N}$, the Defender wins the $(2^{\ell-1}-1)$-round 1-confidence non-adaptive game. Let $m = 2^{\ell-1}-1$ and let $\vec{x} = x_1, \ldots, x_m$ be the sequence of moves performed by the Attacker in the non-adaptive game. Then, there is $s^1 \in \{0,1\}^{\ell-1}$, that is not a prefix of any of the x_i. Let S be the set of strings of length ℓ that start with s^1. Defender then defines $\vec{o}$ to be the sequence where for all $1 \leq i \leq m$, $o_i = s^1$. Then notice that (i) for any s in S, and any x_i, $P_{\mathsf{dis}}(s, x_i) = o_i$, and (ii) there are at least two secrets s and s' in S, namely $s^1 \cdot 0$ and $s^1 \cdot 1$. Therefore, by (i) $S \subseteq P_{\mathsf{dis}}^{-1}(\vec{x}, \vec{o})$, and by (ii) $|S| \geq 2$. Hence $\frac{1}{|P_{\mathsf{dis}}^{-1}(\vec{x},\vec{o})|} < 1$, as required for Defender to win the game. ■

Corollary IV.5. *There exists a program P, that is $(2,1)$-insecure in the adaptive setting, but $(2^{\ell/2},1)$-secure in the non-adaptive setting.*

V. Attack and Defense Slopes

In this section, we use the game-characterization developed earlier, to approximate the search of attack strategies on programs, or show they are secure against attacks of certain length. We define the notion of *slopes*, which intuitively describe how quickly an attacker can refine their knowledge about the secret. In this section, we carry over the uniform-deterministic setting from Section IV.

Attack Slope: Suppose a program P is such that there exists $h \in [0,1]$ and $k \in \mathbb{N}^+$, where for each $\vec{v} \in \mathcal{I}^*$ and $\vec{o} \in \mathcal{O}^*$ of the same length, there is a sequence $\vec{w} \in \mathcal{I}^k$ of length k, such that for any sequence $\vec{q} \in \mathcal{O}^k$, $\frac{|P^{-1}(\vec{v}\cdot\vec{w},\vec{o}\cdot\vec{q})|}{|P^{-1}(\vec{v},\vec{o})|} \leq h$, or $|P^{-1}(\vec{v}\cdot\vec{w},\vec{o}\cdot\vec{q})| = 1$. We call such a pair of numbers (k,h), the *attack slope* for P.

Lemma V.1. *Let P be a program with a (k,h) attack slope, with $k \in \mathbb{N}^+$ and $h \in [0,1]$. Then, for any $n \in \mathbb{N}^+$, P is $(n\cdot k, \frac{1}{h^n \cdot 2^\ell})$-insecure.*

Proof: We will show by induction on n that Attacker wins the $(n\cdot k)$-round r-confidence game from any position $\langle \vec{v}, \vec{o}, r\rangle$, for $r \leq \frac{1}{h^n \cdot |P^{-1}(\vec{v},\vec{o})|}$. For what follows, fix arbitrary $\vec{v} \in \mathcal{I}^*$ and $\vec{o} \in \mathcal{O}^*$. For the base case, suppose that $n = 0$. Then any position $\langle \vec{v}, \vec{o}, r\rangle$ is a winning position for Attacker, for any $r \leq \frac{1}{|P^{-1}(\vec{v},\vec{o})|}$, by definition.

For the inductive case, notice that by definition, there is a sequence $\vec{w} \in \mathcal{I}^k$, such that for any sequence $\vec{q} \in \mathcal{O}^k$, $\frac{|P^{-1}(\vec{v}\cdot\vec{w},\vec{o}\cdot\vec{q})|}{|P^{-1}(\vec{v},\vec{o})|} \leq h$. Let $r_{w_1}, r_{w_2}, \ldots, r_{w_k}$ be functions that conform to the allowed moves of Attacker. We will later define the function r_{w_k}, and its valuation will determine that of all the other functions. It is worth noting again, that the choice of each w_i, for $0 < i \leq k$ does not depend on the choice of q_j, for $j < i$, from Defender, and Attacker needs to consider the previous moves of Defender to determine the whole sequence $\vec{w}$, only at the very beginning of these k rounds. By definition of r_{w_i}, for each $1 \leq i < k$, we know that

$$\begin{aligned} r = &\quad \textstyle\sum_{q_1\in\mathcal{O}} \frac{|P^{-1}(\vec{v}\cdot w_1,\vec{o}\cdot q_1)|}{|P^{-1}(\vec{v},\vec{o})|} \cdot r_{w_1}(q_1), \text{ and} \\ r_{w_i}(q_i) = &\quad \textstyle\sum_{q_{i+1}\in\mathcal{O}} \frac{|P^{-1}(\vec{v}\cdot w_1\cdots w_{i+1},\vec{o}\cdot q_1\cdots q_{i+1})|}{|P^{-1}(\vec{v}\cdot w_1\cdots w_i,\vec{o}\cdot q_1\cdots q_i)|} \cdot r_{w_{i+1}}(q_{i+1}). \end{aligned}$$

Without being explicit in these expressions, we take the sum over only those observations for which the respective probability $\mu(q \mid w; \vec{v}, \vec{o}) = \frac{|P^{-1}(\vec{v}\cdot w,\vec{o}\cdot q)|}{|P^{-1}(\vec{v},\vec{o})|}$ is non-zero. From the equations above, it follows that

$$\begin{aligned} r = &\quad \textstyle\sum_{q_1\in\mathcal{O}} \frac{|P^{-1}(\vec{v}\cdot w_1,\vec{o}\cdot q_1)|}{|P^{-1}(\vec{v},\vec{o})|} \cdot \\ &\quad \textstyle\sum_{q_2\in\mathcal{O}} \frac{|P^{-1}(\vec{v}\cdot w_1\cdot w_2,\vec{o}\cdot q_1\cdot q_2)|}{|P^{-1}(\vec{v}\cdot w_1,\vec{o}\cdot q_1)|} \cdot \\ &\quad \cdots \\ &\quad \textstyle\sum_{q_k\in\mathcal{O}} \frac{|P^{-1}(\vec{v}\cdot\vec{w},\vec{o}\cdot\vec{q})|}{|P^{-1}(\vec{v}\cdot w_1\cdots w_{k-1},\vec{o}\cdot q_1\cdots q_{k-1})|} r_{w_k}(q_k), \end{aligned}$$

which is equal to

$$\begin{aligned} r = & \sum_{q_1 \in \mathcal{O}} \sum_{q_2 \in \mathcal{O}} \cdots \sum_{q_k \in \mathcal{O}} \frac{|P^{-1}(\vec{v}\cdot\vec{w}, \vec{o}\cdot\vec{q})|}{P^{-1}(\vec{v},\vec{o})} \cdot r_{w_k}(q_k) \\ = & \sum_{\vec{q} \in \mathcal{O}^k} \frac{|P^{-1}(\vec{v}\cdot\vec{w}, \vec{o}\cdot\vec{q})|}{P^{-1}(\vec{v},\vec{o})} \cdot r_{w_k}(q_k). \end{aligned} \tag{5}$$

Without loss of generality, assume that r is equal to $\frac{1}{h^n \cdot |P^{-1}(\vec{v},\vec{o})|}$. Furthermore, define $r_{w_k}(q_k)$ to be equal to $\frac{1}{h^n \cdot |P^{-1}(\vec{v},\vec{o})|}$. This valuation of r_{w_k} conforms with Eq. 5, since for any $\vec{w} \in \mathcal{I}^k$, $\sum_{\vec{q} \in \mathcal{O}^k} |P^{-1}(\vec{v}\cdot\vec{w}, \vec{o}\cdot\vec{q})| = |P^{-1}(\vec{v},\vec{o})|$ and thus $\sum_{\vec{q} \in \mathcal{O}^k} \frac{|P^{-1}(\vec{v}\cdot\vec{w}, \vec{o}\cdot\vec{q})|}{|P^{-1}(\vec{v},\vec{o})|} = 1$. Then, notice that

$$\begin{aligned} r_{w_k}(q_k) = & \frac{1}{h^n \cdot |P^{-1}(\vec{v},\vec{o})|} \\ = & \frac{1}{h^{n-1} \cdot h \cdot |P^{-1}(\vec{v},\vec{o})|} \\ \leq & \frac{1}{h^{n-1} \cdot |P^{-1}(\vec{v}\cdot\vec{w}, \vec{o}\cdot\vec{q})|}, \end{aligned}$$

for any $\vec{q} \in \mathcal{O}^k$. Thus, by the inductive hypothesis, Attacker wins the $(n-1)\cdot k$-round $r_{w_k}(q_k)$-confidence game from the position $\langle \vec{v}\cdot\vec{w}, \vec{o}\cdot\vec{q}, r_{w_k}(q_k)\rangle$. Since this is the case for any $\vec{q} \in \mathcal{O}^k$, it follows that Attacker wins the n-round r-confidence game from the position $\langle \vec{v}, \vec{o}, r\rangle$.

Finally, by applying the induction statement, since $|P^{-1}(\underline{\varepsilon}, \underline{\varepsilon})| = |\mathcal{S}_\ell| = 2^\ell$, we have that Attacker can win the game with $n \cdot k$ moves from $\langle \underline{\varepsilon}, \underline{\varepsilon}, \frac{1}{h^n \cdot 2^\ell}\rangle$.

■

Example V.2. Consider the program P_{ins} from Section I, *i.e.* the Leaky Login program in Fig. 1. This program has a $(1, \frac{1}{2})$ attack slope: let $\langle \vec{v}, \vec{o}, 1\rangle$ be any position in the game. Remember that $\mathcal{O}$ is a subset of $\mathbb{N}$, where each observation corresponds to the largest common prefix of the secret and the input. Then, let $o \in \vec{o}$, be the highest value, and let $v \in \vec{v}$ be the input that generated that response. It follows that $P^{-1}(\vec{v},\vec{o})$ is the set of all secrets whose prefix of length o is equal to the prefix of length o of v, but the next character in the secret differs from the one in v. Then, let w be equal to the string formed by flipping the $(o+1)$-th character of v. Also let $r_w(q) = 1$, for all $q \in \mathcal{O}$. Let q be any response of Defender to w, and notice that any such response will at the very least reveal the value of the $(o+2)$-th character of the secret. Thus $\frac{|P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q)|}{|P^{-1}(\vec{v},\vec{o})|}$ is less than or equal to $\frac{1}{2}$. By Lemma V.1, P_{ins} is $(n, \frac{2^n}{2^\ell})$-insecure, and in particular it is $(\ell, 1)$-insecure, agreeing with the statement of Proposition IV.2. ▲

Program `Index Query`

```
def indexQuery(secret, index):
  return (secret[index-1] == 1)
```

Fig. 2: Index Query program

Example V.3. As a further example, we consider the simple program `IQ` presented in Fig. 2, over public inputs $\mathcal{I}_\ell = \{1, \ldots, \ell\}$ and observations $\{\mathsf{true}, \mathsf{false}\}$. On inputs $s \in \mathcal{S}_\ell$ and $v \in \mathcal{I}_\ell$, the program returns true if the v-th symbol of s is 1, and false otherwise. This is a trivially vulnerable program, where the attacker simply has to query one index at a time to gain information about all the bits of the secret s. The same result can just as easily be established by showing that the program has a $(\frac{1}{2}, 1)$ attack slope. Let $\vec{v} \in \mathcal{I}^*$ and $\vec{o} \in \mathcal{O}^*$ be two sequences of public inputs and observations respectively, of length less than ℓ. Then let v' be any natural less than ℓ that is not in the sequence $\vec{v}$. It is not difficult to see that, for any $q \in \mathcal{O}$, $\frac{|P^{-1}(\vec{v}\cdot v', \vec{o}\cdot q)|}{|P^{-1}(\vec{v},\vec{o})|} < \frac{1}{2}$, as the v'-th bit of s is undetermined before, and determined by feeding v' as a public input. This attack slope implies that `IQ` is $(\ell, 1)$-insecure. ▲

Defense Slope: Suppose a program P is such that there exists $h \in \mathbb{N}$, where for each $\vec{v} \in \mathcal{I}^*$ and $\vec{o} \in \mathcal{O}^*$ of the same length, and for every $w \in \mathcal{I}$, there exists $q_w \in \mathcal{O}$ such that $|P^{-1}(\vec{v},\vec{o})| - |P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q_w)| < h$. We call such a number h, the *defense slope* for P. A defense slope of h, indicates that Attacker, at each round can exclude at best a constant number (bounded by h) of secrets.

Lemma V.4. *Let P be a program with a defense slope h. Then for all $n \in \mathbb{N}$, P is $(n, \frac{(n+2)h}{2^\ell})$-secure.*

Proof: We show by induction on n, that if P has a defense slope h, then for any $\vec{v} \in \mathcal{I}^*$ and $\vec{o} \in \mathcal{O}^*$, Defender wins the n-round game starting at position $\langle \vec{v}, \vec{o}, r\rangle$, for $r \geq \frac{(n+2)h}{|P^{-1}(\vec{v},\vec{o})|}$. This would imply that Defender wins the n-round game from position $\langle \underline{\varepsilon}, \underline{\varepsilon}, \frac{(n+2)h}{2^\ell}\rangle$, and thus P is $(n, \frac{(n+2)h}{2^\ell})$-secure.

We note that $h > 0$ since $|P^{-1}(\vec{v},\vec{o})|$ cannot be less than $|P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q_w)|$, and h is by assumption strictly larger than the difference. For the base case, let $n = 0$. Then for any position $\vec{v} \in \mathcal{I}^*$ and $\vec{o} \in \mathcal{O}^*$, Defender wins the 0-round game if $\frac{1}{|P^{-1}(\vec{v},\vec{o})|} < r$ which holds since $r \geq \frac{2h}{|P^{-1}(\vec{v},\vec{o})|}$ by assumption, and $h \in \mathbb{N}^+$.

For the inductive case, suppose that for any $w \in \mathcal{I}$, there exists $q_w \in \mathcal{O}$, such that $|P^{-1}(\vec{v},\vec{o})| - |P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q_w)| < h$. Let $w \in \mathcal{I}$ be any choice of Attacker. By definition of a game, Attacker has to chose a function $r_w : \mathcal{O} \to [0,1]$, such that $r = \sum_{q \in \mathcal{O}} \frac{|P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q)|}{|P^{-1}(\vec{v},\vec{o})|} \cdot r_w(q_w)$. The latter is equal to

$$\sum_{q \in \mathcal{O}\setminus\{q_w\}} \frac{|P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q)|}{|P^{-1}(\vec{v},\vec{o})|} \cdot r_w(q) + \frac{|P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q_w)|}{|P^{-1}(\vec{v},\vec{o})|} \cdot r_w(q_w)$$

Let R be equal to $\max_{q \in \mathcal{O}\setminus\{q_w\}} r_w(q)$. Then

$$\begin{aligned} r \leq & \sum_{q \in \mathcal{O}\setminus\{q_w\}} \frac{|P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q)|}{|P^{-1}(\vec{v},\vec{o})|} \cdot R + \frac{|P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q_w)|}{|P^{-1}(\vec{v},\vec{o})|} \cdot r_w(q_w) \\ = & \frac{R}{|P^{-1}(\vec{v},\vec{o})|} \cdot \Big(\sum_{q \in \mathcal{O}\setminus\{q_w\}} |P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q)|\Big) \\ & + \frac{|P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q_w)|}{|P^{-1}(\vec{v},\vec{o})|} \cdot r_w(q_w). \end{aligned}$$

Notice that $\sum_{q \in \mathcal{O}\setminus\{q_w\}} |P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q)| = |P^{-1}(\vec{v},\vec{o})| - |P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q_w)|$, and hence, by assumption, the latter is strictly less than

$$\frac{R}{|P^{-1}(\vec{v},\vec{o})|} \cdot h + \frac{|P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q_w)|}{|P^{-1}(\vec{v},\vec{o})|} \cdot r_w(q_w).$$

By assumption $r \geq \frac{(n+2)h}{|P^{-1}(\vec{v},\vec{o})|}$, and thus it follows that

$$(n+2)h < R \cdot h + |P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q_w)| \cdot r_w(q_w).$$

which implies that

$$r_w(q_w) > \frac{(n+2)h - R\cdot h}{|P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q_w)|}.$$

Furthermore, $R \leq 1$, and hence

$$r_w(q_w) > \frac{(n+1)\cdot h}{|P^{-1}(\vec{v}\cdot w, \vec{o}\cdot q_w)|}.$$

By the inductive hypothesis, Defender wins the $(n-1)$-round game from position $\langle \vec{v}, \vec{o}, r_w(q_w)\rangle$, which completes the argument. ∎

It is worth mentioning that the definition of defense slope does not put any constraints on the function $r_w : \mathcal{O} \to [0,1]$ selected at each round by Attacker.

Example V.5. Let P_{sec} be the program from Section I. This program has a defense slope of 2: let $\vec{v} \in \mathcal{I}^*$ and $\vec{o} \in \mathcal{O}^*$. For any $w \in \mathcal{I}$, let $q_w = \mathsf{false}$. Then $|P^{-1}(\vec{v}, \vec{o})| - |P^{-1}(\vec{v}\cdot w, \vec{o}\cdot \mathsf{false})| < 2$ since the only secret excluded is the one equal to w. Then Lemma V.4 says that P_{sec} is, for example, $(2^{\ell-2}, \frac{2^{\ell-1}+4}{2^\ell})$-secure. For $\ell > 2$, $1 \geq \frac{2^{\ell-1}+4}{2^\ell}$, and therefore it follows that P_{sec} is even $(2^{\ell-2}, 1)$-secure. In other words, even with $2^{\ell-2}$ moves, Attacker cannot know with absolute certainty what the secret is. ▲

While defense slopes provide sound bound on security, it is incomplete. That is, secure programs do not need to have a strong defense slope in general. The intuition is that sometimes a steep slope may be achievable for a restricted number of moves and then flatten as the game proceeds. This is illustrated with the next example.

Example V.6. Let $P_{\mathsf{par}}(s, v)$, be the program, that for any input $v \in \mathcal{I}$, it returns the parity of 1's in s. Since the output and observation of P_{par} does not depend on the public input, all the information Attacker can gain from this program is whether the number of 1's in the secret is even or odd, and additional queries to the program will not provide any more information. Thus, for any $\epsilon > \frac{1}{2}$ and any $f > 1$, P_{par} is (f, ϵ)-secure. For large enough ℓ it is then $(\ell, \frac{(\ell+2)\cdot 2}{2^\ell})$-secure, but notice that it does not have a defense slope of 2. This is because $\mathcal{O} = \{0, 1\}$ has exactly two values corresponding to the parity of s. No matter what input w we feed initially to the program, and no matter which of the two parities q_w we observe, the size of $|P^{-1}(w, q_w)|$ will be half the size of all secrets. Clearly $|P^{-1}(\varepsilon, \varepsilon)| - |P^{-1}(w, q_w)| \geq 2$, and thus the defense slope of P_{par} is not 2. ▲

Conversely, an insecure program may not necessarily have a steep attack slope. That could happen when after some sequences $\vec{v} \in \mathcal{I}$ and $\vec{o} \in \mathcal{O}$, there is not always a sequence of moves $\vec{w} \in \mathcal{I}$ by Attacker that for any response $\vec{q} \in \mathcal{O}$ would cause the sets $\frac{P^{-1}(\vec{v}\cdot\vec{w}, \vec{o}\cdot\vec{q})}{P^{-1}(\vec{v}, \vec{o})}$ to be less than a slope h, but later moves may compensate for that with a slope even steeper than h. This problem can sometimes be alleviated by reasoning over a sequence of k moves $\vec{w} \in \mathcal{I}^k$ and $\vec{q} \in \mathcal{O}^k$ as the definition of attack slope allows.

We conclude this section with a small remark about the restriction of these approximations on deterministic programs over uniform distributions of secrets. Both the attack and defense slopes depend on measuring in a combinatorial way the amount of change in knowledge Attacker has after successive moves. For deterministic systems with uniform distributions, this amounts to measuring how many secrets are still possible under the sequence of observations from multiple executions of the program under attack. When the programs are probabilistic, or in other words there is a non-trivial probability distribution over the observations, given a public input and a secret, comparisons between distributions over the secrets is not immediate. There is no direct way of comparing two distributions over the secrets and determining that one corresponds to "better" knowledge than the other without actually playing the game. It is not clear how to compare even two distributions where the support of one is strictly larger than the support of the other.

VI. Reductions

In this section we present a way of reducing a game strategy on one program to that of another. Calculating the optimal strategy for one of the two players in a game hinges on a sequence of alternations in optimizations, resulting in high complexity. As such, taking advantage of an existing strategy for a game to produce one for a new game in a new setting is highly desirable, especially if such a reduction depends on mapping each level of alternations of one game to one level, or a bounded number of levels of the other. Such reasoning is inherently more local, and is computationally more efficient.

The general idea behind a reduction from a program P_1 to a program P_2 is as follows. Suppose that it has already been established that Attacker has a winning strategy on P_2 for the game with some number of rounds n and confidence r. To establish a similar Attacker strategy for P_1, we check what move the unbeatable Attacker on P_2 would make, and use the mapping Φ to make a corresponding move on the P_1 game. Then for any response by Defender on that game on P_1, we use the mapping Ω to make a Defender response move back on the game on P_2. Since we know that Attacker on P_2 is unbeatable, and assuming the conditions for the mappings Φ, Ψ and Ω hold, we can deduce that Attacker can also win the game on P_1. See Fig. 3 for a depiction. We remark that a reduction from a secure program P_1 to a program P_2 establishes security of the latter, while a reduction of a program P_2 to an insecure program P_1 establishes that P_2 is insecure. Thus, the reduction can be used for both proving and disproving security.

Let P be a program defined over a set of secrets $\mathcal{S}$ with a probability distribution $\mu_{\mathcal{S}}$, a set of public inputs $\mathcal{I}$ and a set of observations $\mathcal{O}$ determined by the probability distribution $\mu_{\mathcal{O}}$. Namely for any $s \in \mathcal{S}$ and $v \in \mathcal{I}$, the probability of observing $o \in \mathcal{O}$ is given by $\mu_{\mathcal{O}}(o \mid s, v)$. We denote this using the tuple $P = (\mathcal{S}, \mathcal{I}, \mathcal{O}, \mu_{\mathcal{S}}, \mu_{\mathcal{O}})$. For $\vec{v} \in \mathcal{I}^*$ and $\vec{o} \in \mathcal{O}^*$, we define $\mu_{\mathcal{S}}^{(\vec{v},\vec{o})}$ to be the probability distribution that maps every $s \in \mathcal{S}$ to $\mu_{\mathcal{S}}(s \mid \vec{v}, \vec{o})$.

Let P_1 be the program defined by $(\mathcal{S}_1, \mathcal{I}_1, \mathcal{O}_1, \mu_{\mathcal{S}_1}, \mu_{\mathcal{O}_1})$ and P_2 be the program defined by $(\mathcal{S}_2, \mathcal{I}_2, \mathcal{O}_2, \mu_{\mathcal{S}_2}, \mu_{\mathcal{O}_2})$. Let $\Xi = (\Psi, \Phi, \Omega)$ be a tuple of mappings of type:

$$\Phi : \mathcal{I}_2 \to \mathcal{I}_1$$
$$\Psi : \mathcal{S}_2 \to \mathcal{S}_1, \text{ that is injective and total,}$$
$$\Omega : \mathcal{O}_1 \times \mathcal{I}_2 \to \mathcal{O}_2.$$

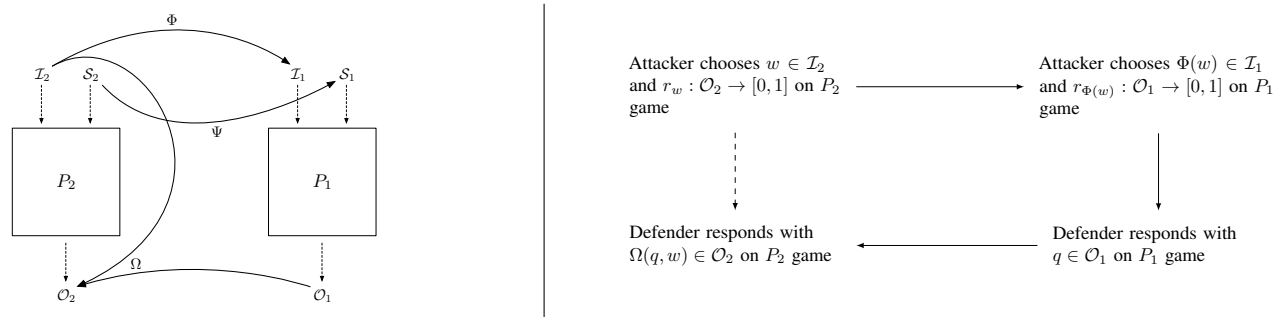

Fig. 3: Depiction of the mappings Φ, Ψ and Ω of a reduction: *for all* $s \in \mathcal{S}_2$, $w \in \mathcal{I}_2$ *and* $q \in \mathcal{O}_1$, $\mu_{\mathcal{O}_1}(q \mid \Psi(s), \Phi(w)) = \mu_{\mathcal{O}_2}(\Omega(q,w) \mid s, w)$.

Then, given probability distributions $\mu_{\mathcal{S}_1}^{(\vec{v}_1,\vec{o}_1)}$ and $\mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)}$, we say that $\mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)} = \Psi\langle\mu_{\mathcal{S}_1}^{(\vec{v}_1,\vec{o}_1)}\rangle$, if

$$\forall s \in \mathcal{S}_2,\ \mu_{\mathcal{S}_2}(s \mid \vec{v}_2, \vec{o}_2) = \mu_{\mathcal{S}_1}(\Psi(s) \mid \vec{v}_1, \vec{o}_1).$$

Definition VI.1. Given probability distributions $\mu_{\mathcal{S}_1}^{(\vec{v}_1,\vec{o}_1)}$ and $\mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)}$, we say that the program P_1 *reduces via* Ξ *to the program* P_2 and write it as $(P_1, \mu_{\mathcal{S}_1}^{(\vec{v}_1,\vec{o}_1)}) \preccurlyeq_\Xi (P_2, \mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)})$, if:

- for all $s \in \mathcal{S}_2$, $w \in \mathcal{I}_2$ and $q \in \mathcal{O}_1$,

$$\mu_{\mathcal{O}_1}(q \mid \Psi(s), \Phi(w)) = \mu_{\mathcal{O}_2}(\Omega(q,w) \mid s, w)$$

  and
- $\mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)} = \Psi(\mu_{\mathcal{S}_1}^{(\vec{v}_1,\vec{o}_1)})$.

In such a case, we call the tuple of mappings Ξ a reduction of P_1 to P_2.

First we present two lemmas that are used in the proof of Theorem VI.4.

Lemma VI.2. *Let* $P_1 = (\mathcal{S}_1, \mathcal{I}_1, \mathcal{O}_1, \mu_{\mathcal{S}_1}, \mu_{\mathcal{O}_1})$ *and* $P_2 = (\mathcal{S}_2, \mathcal{I}_2, \mathcal{O}_2, \mu_{\mathcal{S}_2}, \mu_{\mathcal{O}_2})$ *be two programs and let* $\Xi = (\Psi, \Phi, \Omega)$ *and* $\vec{v}_1 \in \mathcal{I}_1, \vec{v}_2 \in \mathcal{I}_2, \vec{o}_1 \in \mathcal{O}_1, \vec{o}_2 \in \mathcal{O}_2$ *be such that* $(P_1, \mu_{\mathcal{S}_1}^{(\vec{v}_1,\vec{o}_1)}) \preccurlyeq_\Xi (P_2, \mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)})$. *Then, for all* $w \in \mathcal{I}_2, q \in \mathcal{O}_1$, *and* $\ell \in \mathbb{N}$,

$$\textstyle\sum_{s' \in (\mathcal{S}_2)_\ell} \mu_{\mathcal{O}_2}(\Omega(q,w) \mid s', w) \cdot \mu_{\mathcal{S}_2}(s' \mid \vec{v}_2, \vec{o}_2) = \sum_{s' \in (\mathcal{S}_1)_\ell} \mu_{\mathcal{O}_1}(q \mid s', \Phi(w)) \cdot \mu_{\mathcal{S}_1}(s' \mid \vec{v}_1, \vec{o}_1)).$$

Proof: By assumption that $(P_1, \mu_{\mathcal{S}_1}^{(\vec{v}_1,\vec{o}_1)}) \preccurlyeq_\Xi (P_2, \mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)})$, we know that for all $s' \in \mathcal{S}_2$, $\mu_{\mathcal{S}_2}(s' \mid \vec{v}_2, \vec{o}_2) = \mu_{\mathcal{S}_1}(\Psi(s') \mid \vec{v}_1, \vec{o}_1)$. This also implies that for all $s'' \in \mathcal{S}_1$ that are not in the image of Ψ, $\mu_{\mathcal{S}_1}(s'' \mid \vec{v}_1, \vec{o}_1)$ must be equal to 0, since for all $\ell \in \mathbb{N}$ $\sum_{s' \in (\mathsf{Im}(\Psi))_\ell} \mu_{\mathcal{S}_1}(\Psi(s') \mid \vec{v}_1, \vec{o}_1) = \sum_{s' \in (\mathcal{S}_2)_\ell} \mu_{\mathcal{S}_2}(s' \mid \vec{v}_2, \vec{o}_2) = 1$ and by definition $\sum_{s' \in (\mathcal{S}_1)_\ell} \mu_{\mathcal{S}_1}(\Psi(s') \mid \vec{v}_1, \vec{o}_1)$ is also equal to 1. Furthermore, by assumption, for all $s \in \mathcal{S}_2$, $w \in \mathcal{I}_2$ and $q \in \mathcal{O}_1$, $\mu_{\mathcal{O}_2}(\Omega(q,w) \mid s, w) = \mu_{\mathcal{O}_1}(q \mid \Psi(s), \Phi(w))$, and therefore

$$\begin{aligned} &\textstyle\sum_{s' \in (\mathcal{S}_2)_\ell} \mu_{\mathcal{O}_2}(\Omega(q,w) \mid s', w) \cdot \mu_{\mathcal{S}_2}(s' \mid \vec{v}_2, \vec{o}_2) = \\ =\ &\textstyle\sum_{s' \in (\mathcal{S}_2)_\ell} \mu_{\mathcal{O}_1}(q \mid \Psi(s'), \Phi(w)) \cdot \mu_{\mathcal{S}_1}(\Psi(s') \mid \vec{v}_1, \vec{o}_1)) = \\ =\ &\textstyle\sum_{s'' \in (\mathcal{S}_1)_\ell} \mu_{\mathcal{O}_1}(q \mid s'', \Phi(w)) \cdot \mu_{\mathcal{S}_1}(s'' \mid \vec{v}_1, \vec{o}_1)), \end{aligned}$$

as required. ∎

Lemma VI.3. *Let* $P_1 = (\mathcal{S}_1, \mathcal{I}_1, \mathcal{O}_1, \mu_{\mathcal{S}_1}, \mu_{\mathcal{O}_1})$ *and* $P_2 = (\mathcal{S}_2, \mathcal{I}_2, \mathcal{O}_2, \mu_{\mathcal{S}_2}, \mu_{\mathcal{O}_2})$ *be two programs and let* $\Xi = (\Psi, \Phi, \Omega)$ *and* $\vec{v}_1 \in \mathcal{I}_1, \vec{v}_2 \in \mathcal{I}_2, \vec{o}_1 \in \mathcal{O}_1, \vec{o}_2 \in \mathcal{O}_2$ *be such that* $(P_1, \mu_{\mathcal{S}_1}^{(\vec{v}_1,\vec{o}_1)}) \preccurlyeq_\Xi (P_2, \mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)})$. *Then for any* $\vec{v} \in \mathcal{I}_2^*$ *and* $\vec{o} \in \mathcal{O}_1^*$, $(P_1, \mu_{\mathcal{S}_1}^{(\vec{v}_1 \cdot \Phi(\vec{v}), \vec{o}_1 \cdot \vec{o})}) \preccurlyeq_\Xi (P_2, \mu_{\mathcal{S}_2}^{(\vec{v}_2 \cdot \vec{v}, \vec{o}_2 \cdot \Omega(\vec{o}, \vec{v}))})$.

Proof: We need to show that for any $\vec{v} \in \mathcal{I}_2^*$ and $\vec{o} \in \mathcal{O}_1^*$, and for any $s \in \mathcal{S}_2$, $w \in \mathcal{I}_2$ and $q \in \mathcal{O}_1$, it holds that

- $\mu_{\mathcal{O}_1}(q \mid \Psi(s), \Phi(w)) = \mu_{\mathcal{O}_2}(\Omega(q,w) \mid s, w)$, and
- $\mu_{\mathcal{S}_2}^{(\vec{v}_2 \cdot \vec{v}, \vec{o}_2 \cdot \Omega(\vec{o}, \vec{v}))} = \Psi(\mu_{\mathcal{S}_1}^{(\vec{v}_1 \cdot \Phi(\vec{v}), \vec{o}_1 \cdot \vec{o})})$.

The first condition holds by assumption that $(P_1, \mu_{\mathcal{S}_1}^{(\vec{v}_1,\vec{o}_1)}) \preccurlyeq_\Xi (P_2, \mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)})$. For the second condition it suffices to show that for all $\vec{v} \in \mathcal{I}_2^*$, $\vec{o} \in \mathcal{O}_1^*$, and $s \in \mathcal{S}_2$,

$$\mu_{\mathcal{S}_2}(s \mid \vec{v}_2 \cdot \vec{v}, \vec{o}_2 \cdot \Omega(\vec{o}, \vec{v})) = \mu_{\mathcal{S}_1}(\Psi(s) \mid \vec{v}_1 \cdot \Phi(\vec{v}), \vec{o}_1 \cdot \vec{o}).$$

We proceed by induction on the length $|\vec{v}| = |\vec{o}|$. For the *base case*, suppose that $|\vec{v}| = |\vec{o}| = 0$. Then the statement holds by the assumption that $\mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)} = \Psi(\mu_{\mathcal{S}_1}^{(\vec{v}_1,\vec{o}_1)})$. For the *inductive case*, suppose that the statement holds for all $\vec{v}$ and $\vec{o}$ whose size is less than N, for some $N \in \mathbb{N}$, and consider $\vec{v}$ and $\vec{o}$ whose size is exactly N. We write $\vec{v}$ as $\vec{v}' \cdot w$ and $\vec{o}$ as $\vec{o}' \cdot q$, where $|\vec{v}'| = |\vec{o}'| = N - 1$. Then, by the inductive hypothesis we have that $(P_1, \mu_{\mathcal{S}_1}^{(\vec{v}_1 \cdot \Phi(\vec{v}'), \vec{o}_1 \cdot \vec{o}')}) \preccurlyeq_\Xi (P_2, \mu_{\mathcal{S}_2}^{(\vec{v}_2 \cdot \vec{v}', \vec{o}_2 \cdot \Omega(\vec{o}', \vec{v}'))})$, which immediately implies that for any $s \in \mathcal{S}_2$, $w \in \mathcal{I}_2$ and $q \in \mathcal{O}_1$, it holds that

- $\mu_{\mathcal{O}_1}(q \mid \Psi(s), \Phi(w)) = \mu_{\mathcal{O}_2}(\Omega(q,w) \mid s, w)$, and
- $\mu_{\mathcal{S}_2}^{(\vec{v}_2 \cdot \vec{v}', \vec{o}_2 \cdot \Omega(\vec{o}', \vec{v}'))} = \Psi(\mu_{\mathcal{S}_1}^{(\vec{v}_1 \cdot \Phi(\vec{v}'), \vec{o}_1 \cdot \vec{o}')})$.

Furthermore, by definition,

$$\mu_{\mathcal{S}_2}(s \mid \vec{v}_2 \cdot \vec{v}' \cdot w, \vec{o}_2 \cdot \Omega(\vec{o}', \vec{v}') \cdot \Omega(q,w)) = \frac{\mu_{\mathcal{O}_2}(\Omega(q,w) \mid s, w) \cdot \mu_{\mathcal{S}_2}(s \mid \vec{v}_2 \cdot \vec{v}', \vec{o}_2 \cdot \Omega(\vec{o}', \vec{v}'))}{\sum_{s' \in (\mathcal{S}_2)_\ell} \mu_{\mathcal{O}_2}(\Omega(q,w) \mid s', w) \cdot \mu_{\mathcal{S}_2}(s' \mid \vec{v}_2 \cdot \vec{v}', \vec{o}_2 \cdot \Omega(\vec{o}', \vec{v}'))}, \tag{6}$$

and by Lemma VI.2, the denominator of the latter is equal to

$$\sum_{s'' \in (\mathcal{S}_1)_\ell} \mu_{\mathcal{O}_1}(q \mid s'', \Phi(w)) \cdot \mu_{\mathcal{S}_1}(s'' \mid \vec{v}_1 \cdot \Phi(\vec{v}'), \vec{o}_1 \cdot \vec{o}')).$$

Thus $\mu_{\mathcal{S}_2}(s \mid \vec{v}_2 \cdot \vec{v}' \cdot w, \vec{o}_2 \cdot \Omega(\vec{o}', \vec{v}') \cdot \Omega(q,w))$ is equal to

$$\frac{\mu_{\mathcal{O}_1}(q \mid \Psi(s), \Phi(w)) \cdot \mu_{\mathcal{S}_1}(\Psi(s) \mid \vec{v}_1 \cdot \Phi(\vec{v}'), \vec{o}_1 \cdot \vec{o}')}{\sum_{s'' \in (\mathcal{S}_1)_\ell} \mu_{\mathcal{O}_1}(q \mid s'', \Phi(w)) \cdot \mu_{\mathcal{S}_1}(s'' \mid \vec{v}_1 \cdot \Phi(\vec{v}'), \vec{o}_1 \cdot \vec{o}')}.$$

which is equal, by definition, to $\mu_{\mathcal{S}_1}(\Psi(s) \mid \vec{v}_1 \cdot \Phi(\vec{v}), \vec{o}_1 \cdot \vec{o})$, as required. ∎

It follows that if $(P_1, \mu_{\mathcal{S}_1}^{(\varepsilon,\varepsilon)}) \preccurlyeq_\Xi (P_2, \mu_{\mathcal{S}_2}^{(\varepsilon,\varepsilon)})$, then for any $\vec{v} \in \mathcal{I}_2^*$ and $\vec{o} \in \mathcal{O}_1^*$, $(P_1, \mu_{\mathcal{S}_1}^{(\Phi(\vec{v}),q)}) \preccurlyeq_\Xi (P_2, \mu_{\mathcal{S}_2}^{(\vec{v},\Omega(\vec{o},\vec{v}))})$.

The following theorem establishes the connection between reductions of one program to another, and simulating a strategy of the latter as a strategy of the former.

Theorem VI.4. *Let $P_1 = (\mathcal{S}_1, \mathcal{I}_1, \mathcal{O}_1, \mu_{\mathcal{S}_1}, \mu_{\mathcal{O}_1})$ and $P_2 = (\mathcal{S}_2, \mathcal{I}_2, \mathcal{O}_2, \mu_{\mathcal{S}_2}, \mu_{\mathcal{O}_2})$ be two programs, let $\vec{v}_1 \in \mathcal{I}_1, \vec{v}_2 \in \mathcal{I}_2, \vec{o}_1 \in \mathcal{O}_1$ and $\vec{o}_2 \in \mathcal{O}_2$, and let $\Xi = (\Psi, \Phi, \Omega)$ be such that $(P_1, \mu_{\mathcal{S}_1}^{(\vec{v}_1,\vec{o}_1)}) \preccurlyeq_\Xi (P_2, \mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)})$. Suppose also that Attacker can win the n-round r-confidence game on P_2, starting from the initial distribution of secrets $\mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)}$ for some $n \in \mathbb{N}$ and $r \in [0,1]$. Then Attacker can win the n-round r-confidence game on P_1 starting with initial distribution of secrets $\mu_{\mathcal{S}_1}^{(\vec{v}_1,\vec{o}_1)}$.*

Proof: In what follows, when Attacker can win the n-round r-confidence game starting with initial probability distribution of secrets $\mu_{\mathcal{S}}$, we say that $\langle \mu_{\mathcal{S}}, n, r\rangle$ is a winning configuration for Attacker. It should be noted that what we call a position $\langle \vec{v}, \vec{o}, r\rangle$ in the game, with n remaining rounds, corresponds to the configuration $\langle \mu_{\mathcal{S}}^{(\vec{v},\vec{o})}, n, r\rangle$, and vice versa.

The proof proceeds by induction on $n \in \mathbb{N}$. For the *base case*, suppose that $n = 0$. Then, since $\langle \mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)}, 0, r\rangle$ is a winning configuration for Attacker in P_2, it follows that $\max_{s\in\mathcal{S}_2} \mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)}(s) \geq r$. Let s be such that $\mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)}(s) \geq r$. Since $\mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)} = \Psi(\mu_{\mathcal{S}_1}^{(\vec{v}_1,\vec{o}_1)})$, it follows that $\mu_{\mathcal{S}_1}^{(\vec{v}_1,\vec{o}_1)}(\Psi(s)) \geq r$, and therefore, $\langle \mu_{\mathcal{S}_1}^{(\vec{v}_1,\vec{o}_1)}, 0, r\rangle$ is a winning configuration for Attacker in P_1.

For the *inductive case*, we distinguish the two attackers and defenders by writing Attacker$_1$ and Defender$_1$ for the ones on P_1, and Attacker$_2$ and Defender$_2$ for the ones on P_2. Suppose the statement holds for all $n < N$, for some $N \in \mathbb{N}$, and suppose $\langle \mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)}, N, r\rangle$ is a winning configuration for Attacker$_2$ in P_2. Then, Attacker$_2$ can choose $w \in \mathcal{I}_2$ and function $r_w : \mathcal{O}_2 \to [0,1]$, according to his winning strategy. Let Attacker$_1$ choose $\Phi(w) \in \mathcal{I}_1$ and function $r_{\Phi(w)} : \mathcal{O}_1 \to [0,1]$ defined as the function that maps any $q \in \mathcal{O}_1$ to $r_w(\Omega(q,w))$. First we show that $r_{\Phi(w)}$ satisfies the constraints of a valid move by showing that $\sum_{q\in\mathcal{O}_1} \mu_{\mathcal{O}_1}(q \mid \Phi(w); \vec{v}_1, \vec{o}_1) \cdot r_{\Phi(w)}(q) = r$. Since for all $q \in \mathcal{O}_1$, $r_{\Phi(w)}(q) = r_w(\Omega(q,w))$, it suffices to show that for all $q \in \mathcal{O}_1$, $\mu_{\mathcal{O}_1}(q \mid \Phi(w); \vec{v}_1, \vec{o}_1) = \mu_{\mathcal{O}_2}(\Omega(q,w) \mid w; \vec{v}_2, \vec{o}_2)$, and that for any q' not in the image of Ω, $\mu_{\mathcal{O}_2}(q' \mid w; \vec{v}_2, \vec{o}_2) = 0$. It should be noted that the second statement is implied by the first, since $\sum_{q\in\mathcal{O}_2} \mu_{\mathcal{O}_2}(q \mid w; \vec{v}_2, \vec{o}_2) = 1$ and also $\sum_{q\in\mathcal{O}_1} \mu_{\mathcal{O}_1}(q \mid \Phi(w); \vec{v}_1, \vec{o}_1)) = 1$.

By definition, $\mu_{\mathcal{O}_1}(q \mid \Phi(w); \vec{v}_1, \vec{o}_1))$ is equal to

$$\sum_{s'\in(\mathcal{S}_1)_\ell} \mu_{\mathcal{O}_1}(q \mid s', \Phi(w)) \cdot \mu_{\mathcal{S}_2}(s' \mid \vec{v}_1, \vec{o}_1),$$

and $\mu_{\mathcal{O}_2}(\Omega(q,w) \mid w; \vec{v}_2, \vec{o}_2)$ is equal to

$$\sum_{s'\in(\mathcal{S}_2)_\ell} \mu_{\mathcal{O}_2}(\Omega(q,w) \mid s', w) \cdot \mu_{\mathcal{S}_2}(s' \mid \vec{v}_2, \vec{o}_2).$$

Thus, by Lemma VI.2, for all $q \in \mathcal{O}_1$, $\mu_{\mathcal{O}_1}(q \mid \Phi(w); \vec{v}_1, \vec{o}_1)) = \mu_{\mathcal{O}_2}(\Omega(q,w) \mid w; \vec{v}_2, \vec{o}_2)$.

Therefore $r_{\Phi(w)}$ satisfies the constraints of a valid move by Attacker$_1$. Let $q \in \mathcal{O}_1$ be any reply from Defender$_1$. By assumption that $\langle \mu_{\mathcal{S}_2}^{(\vec{v}_2,\vec{o}_2)}, n, r\rangle$ is a winning configuration for Attacker$_2$, it follows that $\langle \mu_{\mathcal{S}_2}^{(\vec{v}_2\cdot w,\vec{o}_2\cdot\Omega(q,w))}, n-1, r_w(\Omega(q,w))\rangle$ is also a winning configuration for Attacker$_2$. By Lemma VI.3, $(P_1, \mu_{\mathcal{S}_1}^{(\vec{v}_1\cdot\Phi(w),\vec{o}_1\cdot q)}) \preccurlyeq_\Xi (P_2, \mu_{\mathcal{S}_2}^{(\vec{v}_2\cdot w,\vec{o}_2\cdot\Omega(q,w))})$. Furthermore $r_{\Phi(w)}(q) = r_w(\Omega(q,w))$. Thus, we can apply the inductive hypothesis, and conclude that $\langle \mu_{\mathcal{S}_1}^{(\vec{v}_1\cdot\Phi(w),\vec{o}_1\cdot q)}, n-1, r_{\Phi(w)}(q)\rangle$ is a winning configuration for Attacker$_1$. As this is the case for any move q by Defender$_1$, it holds that $\langle \mu_{\mathcal{S}_1}^{(\vec{v}_1,\vec{o}_1)}, n, r\rangle$ is a winning configuration for Attacker$_1$ as required. ∎

A. Examples on simulating strategies

In Fig. 4, we present the two programs below called PR2 and PR3 whose structure is identical and with the only difference between them being that PR2 returns as output the number of indices in the two input strings where the bits agree, modulo 2, whereas PR3 returns the same number but modulo 3. They both also return a boolean indicating whether the two strings are identical.

The interesting, and somewhat unintuitive, observation about these two programs, is that PR2 is $(2^\ell - 1, 1)$-secure under a uniform initial distribution over the secrets, whereas PR3 is $(\ell + 1, 1)$-insecure over the same distribution of secrets; a vast difference in security for such similar programs. In what follows we aim to establish two strategy simulations, as defined in Definition VI.1, in order to show that PR2 is $(2^\ell - 1, 1)$-secure and PR3 is $(\ell + 1, 1)$-insecure. For this we will use the previous results where we showed that SL (*i.e.* program P_{sec} from Section I) is $(2^\ell - 1, 1)$-secure and IQ is $(\ell, 1)$-insecure.

As explained earlier, to show security of a program P, we use a reduction from a known secure program to P, and to show insecurity of a program P', we use a reduction from P' to a known insecure program. As such, we present:

- a reduction from the secure program SL to PR2, and
- a reduction from PR3 to the insecure program IQ.

These two reductions presented below are not necessarily trivial, but they don't require reasoning over extended sequences of move alternations, as showing security or insecurity directly over the games would require. Instead, the point of the reductions is to take advantage of the more global argument that was made for the established results proving security or insecurity of the programs SL and IQ respectively. We use these two programs to demonstrate how reductions can be used, but arguably, for these two particular programs, there are easier ways to prove security and insecurity.

1) Reduction from SL to PR2: Let P_1 be SL and P_2 be PR2. We fix a length of input secrets $\ell \in \mathbb{N}$ for both programs, so that by $\mathcal{S}_1, \mathcal{I}_1, \mathcal{S}_2$, and $\mathcal{I}_2$ we mean respectively $(\mathcal{S}_1)_\ell$, $(\mathcal{I}_1)_\ell$, $(\mathcal{S}_2)_\ell$, and $(\mathcal{I}_2)_\ell$. We use the notation ℓ_2 to denote the value of ℓ mod 2. Also, because the initial distribution of secrets is uniform, for $i \in \{1,2\}$ and any $s \in \mathcal{S}_i$, $\mu_{\mathcal{S}_i}^{(\varepsilon,\varepsilon)}(s) = \frac{1}{|\mathcal{S}_i|}$. We adopt the following notation. Since the set of observations $\mathcal{O}_2$ of P_2 are

Program PR2

```
def PR2(sec, val):
  if (sec == val == ""):
    return (0, True)
  else:
    res = PR2(sec.tail, val.tail)
    if (sec.head == val.head):
      return ((1+res[1])%2, res[2])
    else:
      return (res[1], False)
```

Program PR3

```
def PR3(sec, val):
  if (sec == val == ""):
    return (0, True)
  else:
    res = PR3(sec.tail, val.tail)
    if (sec.head == val.head):
      return ((1+res[1])%3, res[2])
    else:
      return (res[1], False)
```

Fig. 4: Programs PR2 and PR3 with the differences between two programs highlighted in red.

tuples over $\{0,1\} \times \{\mathsf{true}, \mathsf{false}\}$, for $o \in \mathcal{O}_2$, we denote with $o[1]$ the first component of o in $\{0,1\}$ and with $o[2]$ its second component in $\{\mathsf{true}, \mathsf{false}\}$. Given two inputs $v, w \in \mathcal{I}_2$, we define $\mathsf{diff}(v,w)$ to be the number of indices in the two inputs where the bits differ, modulo 2. Furthermore, for two values $p, q \in \{0,1\}$, we denote with $p \oplus q$ their sum modulo 2. Given $v \in \mathcal{I}_2$ and $o \in \mathcal{O}_2$, let then $\Phi : \mathcal{I}_2 \to \mathcal{I}_1$ and $\Psi : \mathcal{S}_2 \to \mathcal{S}_1$ be the identity functions, and let $\Omega_{(v,o)} : \mathcal{O}_1 \times \mathcal{I}_2 \to \mathcal{O}_2$ be the function that maps (b,w) to $(\mathsf{diff}(v,w) \oplus o[1], b)$ where $b \in \{\mathsf{true}, \mathsf{false}\} = \mathcal{O}_1$ and $w \in \mathcal{I}_2$. Let $\Xi_{(v,o)}$ be the tuple $(\Phi, \Psi, \Omega_{(v,o)})$ of mappings, where $\Omega_{(v,o)}$ is parameterized by $v \in \mathcal{I}_2$ and $o \in \mathcal{O}_2$. With $\overrightarrow{\mathsf{false}}$ we refer to a sequence of observations, all of whose value is false, and where the length of the sequence is determined by the context.

We will show that for any $v \in \mathcal{I}_2$, any $o \in \mathcal{O}_2$, there exists a sequence $\vec{x} \in \mathcal{I}_1$ of length $2^\ell/2$, such that $(P_1, \mu_{\mathcal{S}_1}^{(v\cdot\vec{x}, o[2]\cdot\vec{y})}) \preccurlyeq_{\Xi_{(v,o)}} (P_2, \mu_{\mathcal{S}_2}^{(v,o)})$, where $\vec{y} = \overrightarrow{\mathsf{false}}$ and is of length $2^\ell/2$. Then, by Theorem VI.4, it follows that if Attacker can win the game on P_2 (*i.e.* PR2) from $\langle \mu_{\mathcal{S}_2}^{(v,o)}, n, r\rangle$ then Attacker can also win the game on P_1 (*i.e.* SL) from $\langle \mu_{\mathcal{S}_1}^{(v\cdot\vec{x}, o[2]\cdot\vec{y})}, n, r\rangle$. The sequence $\vec{x}$ will be such that Defender has a winning strategy on P_1 from $\langle \mu_{\mathcal{S}_1}^{(v\cdot\vec{x}, o[2]\cdot\vec{y})}, 2^{\ell-1}-2, 1\rangle$ when $o[2]$ is false. Thus it will follow that Defender has a winning strategy on the program PR2 from $\langle \mu_{\mathcal{S}_2}^{(v,o)}, 2^{\ell-1}-2, 1\rangle$, when o is such that $o[2] = \mathsf{false}$. Since for any $v \in \mathcal{I}_2$ Defender can reply with some $o \in \mathcal{O}_2$ such that $o[2] = \mathsf{false}$, Defender has a winning strategy from $\langle \mu_{\mathcal{S}_2}^{(\varepsilon,\varepsilon)}, 2^{\ell-1}-1, 1\rangle$, and the program PR2 is thus $(2^{\ell-1}-1, 1)$-secure.

Given $v \in \mathcal{I}_2$, we define X_v to be the set of secrets $x \in \mathcal{S}_1$ of length ℓ such that $\mathsf{diff}(v,x) = 1$. Notice that X contains exactly half the secrets from $\mathcal{S}_1$. We then let $\vec{x}$ be a sequence comprising the elements of X_v in any order. When $o[2]$ is false, notice that Defender has a winning strategy on P_1 from $\langle \mu_{\mathcal{S}_1}^{(v\cdot\vec{x}, o[2]\cdot\vec{y})}, 2^{\ell-1}-2, 1\rangle$. We assume that this has been established earlier for the program P_1, but informally this is because each guess from $v \cdot \vec{x}$ by Attacker disqualifies only that guess as a secret.

It remains to show that for any $v \in \mathcal{I}_2$ and $o \in \mathcal{O}_2$, $(P_1, \mu_{\mathcal{S}_1}^{(v\cdot\vec{x}, o[2]\cdot\vec{y})}) \preccurlyeq_{\Xi_{(v,o)}} (P_2, \mu_{\mathcal{S}_2}^{(v,o)})$, where $\vec{y} = \overrightarrow{\mathsf{false}}$. To show the latter, by Definition VI.1, it suffices to show that

- for all $s \in \mathcal{S}_2$, $w \in \mathcal{I}_2$ and $q \in \mathcal{O}_1$, $\mu_{\mathcal{O}_1}(q \mid s, w) = \mu_{\mathcal{O}_2}((\mathsf{diff}(v,w) \oplus o[1], q) \mid s, w)$, and
- for all $s \in \mathcal{S}_2$, $\mu_{\mathcal{S}_2}(s \mid v, o) = \mu_{\mathcal{S}_1}(s \mid v \cdot \vec{x}, o[2] \cdot \vec{y})$.

Fix $v \in \mathcal{I}_2$ and $o \in \mathcal{O}_2$, where $o = (p, b)$, for $p \in \{0,1\}$ and $b \in \{\mathsf{true}, \mathsf{false}\}$. For the first condition, let $s \in \mathcal{S}_2$, $w \in \mathcal{I}_2$ and $q \in \mathcal{O}_1 = \{\mathsf{true}, \mathsf{false}\}$. Since both SL and PR2 are deterministic, there is exactly one possible output for $\mathtt{SL}(s,w)$, either true or false. By definition, $o[1]$, being equal to the first component of the observation $\mathtt{PR2}(s,v)$, is the number of common bits between s and v, modulo 2, and thus it is also equal to the number of different bits modulo 2, summed to the length of s modulo 2. In other words, $o[1]$ is equal to $\mathsf{diff}(s,v) \oplus \ell_2$, meaning that $\mathsf{diff}(s,v) = o[1] \oplus \ell_2$. Using the same reasoning, notice that $\mathtt{PR2}(s,w)[1] = \mathsf{diff}(s,w) \oplus \ell_2$. The latter is equal to $\mathsf{diff}(s,v) \oplus \mathsf{diff}(v,w) \oplus \ell_2$, and consequently equal to $o[1] \oplus \mathsf{diff}(v,w)$. Therefore, $\mathsf{diff}(s,w) = \mathsf{diff}(v,w) \oplus o[1] \oplus \ell_2$. Hence $\mathtt{PR2}(s,w)[1] = \mathsf{diff}(v,w) \oplus o[1] \oplus \ell_2 \oplus \ell_2 = \mathsf{diff}(v,w) \oplus o[1]$. We know that $\mu_{\mathcal{O}_1}(q \mid s, w) = 1$ if and only if $\mu_{\mathcal{O}_2}((\mathtt{PR2}(s,w)[1], q) \mid s, w) = 1$, simply because SL returns true on inputs $s \in \mathcal{S}_1$ and $w \in \mathcal{I}_1$ if and only if the second component of PR2 is true on the same inputs (while the first component is by definition $\mathtt{PR2}(s,w)[1]$). From the argument above, it is the case that $\mu_{\mathcal{O}_2}((\mathtt{PR2}(s,w)[1], q) \mid s, w) = 1$ if and only if $\mu_{\mathcal{O}_2}((\mathsf{diff}(v,w) \oplus o[1], q) \mid s, w) = 1$, and the result follows.

For the second condition, notice that $\mu_{\mathcal{S}_2}(s \mid v, o) = \mu_{\mathcal{S}_1}(s \mid v\cdot\vec{x}, o[2]\cdot\vec{y})$, since in the case that $o[2]$ is true, both $\mu_{\mathcal{S}_2}(s \mid v, o)$ and $\mu_{\mathcal{S}_1}(s \mid v, o[2])$ are 1 for $s = v$, and 0 for all other values of s. Meanwhile, in the case where $o[2]$ is false, both $\mu_{\mathcal{S}_2}(s \mid v, o)$ and $\mu_{\mathcal{S}_1}(s \mid v\cdot\vec{x}, o[2]\cdot\vec{y})$ are 0 for all $s \in \mathcal{S}_2$ such that $\mathsf{diff}(s,v) = 1$, and for $s = v$, and equal to $\frac{1}{|\mathcal{S}_2|/2-1}$ for the remaining $|\mathcal{S}_2|/2 - 1$ secrets. This completes the argument for the reduction.

2) Reduction from PR3 to IQ: We define P_1 to be PR3 and P_2 to be IQ. We again fix $\ell \in \mathbb{N}$ and restrict the sets $\mathcal{S}_1, \mathcal{S}_2, \mathcal{I}_1$ and $\mathcal{I}_2$ to inputs related to the security parameter ℓ as before. We remind the reader that for IQ, for the security parameter ℓ, $(\mathcal{I}_2)_\ell$ is defined to be the set of integers $\{1, \ldots, \ell\}$. We want to show that for any $v \in \mathcal{I}_1$ and $(o,b) \in \mathcal{O}_1$, there exists an initial probability distribution of secrets $\mu_{\mathcal{S}_2}^{(\varepsilon,\varepsilon)}$ and a reduction $\Xi_{(v,o)}$, such that $(P_1, \mu_{\mathcal{S}_1}^{(v,o)}) \preccurlyeq_{\Xi_{(v,o)}} (P_2, \mu_{\mathcal{S}_2}^{(\varepsilon,\varepsilon)})$. Notice that for any initial distribution of secrets $\mu_{\mathcal{S}_2}^{(\varepsilon,\varepsilon)}$, the program IQ (*i.e.* P_2) is $(\ell, 1)$-insecure, as shown earlier in Example V.3. Given the above reduction $\Xi_{(v,o)}$, for any $v \in \mathcal{I}_1$ and $(o,b) \in \mathcal{O}_1$, and Theorem VI.4, it follows that PR3 is $(\ell+1, 1)$-insecure.

It thus remains to be shown that for any $v \in \mathcal{I}_1$, and any

$(o,b) \in \mathcal{O}_1$, there exists $\mu_{\mathcal{S}_2}^{(\underline{\varepsilon},\underline{\varepsilon})}$ and $\Xi_{(v,o)} = (\Phi, \Psi, \Omega)$ such that

- for all $s \in \mathcal{S}_2$, $w \in \mathcal{I}_2$ and $q \in \mathcal{O}_1$,

$$\mu_{\mathcal{O}_1}(q \mid \Psi(s), \Phi(w)) = \mu_{\mathcal{O}_2}(\Omega(q,w) \mid s, w)$$

  and
- $\forall s \in \mathcal{S}_2$, $\mu_{\mathcal{S}_2}(s \mid \underline{\varepsilon}, \underline{\varepsilon}) = \mu_{\mathcal{S}_1}(s \mid v, o)$.

In fact, defining $\mu_{\mathcal{S}_2}^{(\underline{\varepsilon},\underline{\varepsilon})}$ to be equal to $\mu_{\mathcal{S}_1}^{(v,o)}$ suffices to handle the second condition above. This is because `IQ` is $(\ell, 1)$-insecure for any initial distribution of secrets under the same strategy for Attacker.

For the other condition we proceed as follows. First we define the function $\mathsf{flip} : \mathcal{I}_1 \times \mathbb{N} \to \mathcal{I}_1$ such that, given $v = a_1 \dots a_i \dots a_n$ and $i \in \{1, \dots, n\}$, $\mathsf{flip}(v, i) = a_1 \dots \overline{a_i} \dots a_n$ where $\overline{0} = 1$ and $\overline{1} = 0$. In other words, $\mathsf{flip}(v, i)$ is the string obtained by flipping the i-th bit of v. Fix $v \in \mathcal{I}_1$ and $(o, b) \in \mathcal{O}$. We define $\Xi_{(v,o)} = (\Phi, \Psi, \Omega)$ to be such that for all $s \in \mathcal{S}_2, w \in \mathcal{I}_2$ and $q = (m, b) \in \mathcal{O}_1$, $\Phi(w) = \mathsf{flip}(v, w)$, $\Psi(s) = s$ and $\Omega((m, b), w)$ to be true exactly when m is equal to $o+1 \bmod 3$ and the w-th bit of v is 0, or when m is equal to $o - 1 \bmod 3$ and the w-th bit of v is 1. We define the value to be false otherwise. Notice that $\mu_{\mathcal{O}_1}((o \bmod 3, b) \mid s, \mathsf{flip}(v, w))$ is 0, since we flipped exactly one bit from v, and by definition $\mu_{\mathcal{O}_1}((o \bmod 3, b') \mid s, v)) = 1$ for some $b' \in \{\mathsf{true}, \mathsf{false}\}$. As a result, for $v = a_1 \cdots a_\ell$, we have the following cases:

When $a_w = 0$:
$\mu_{\mathcal{O}_1}((o + 1 \bmod 3, b) \mid s, \mathsf{flip}(v, w)) = \mu_{\mathcal{O}_2}(\mathsf{true} \mid s, w)$,
$\mu_{\mathcal{O}_1}((o - 1 \bmod 3, b) \mid s, \mathsf{flip}(v, w)) = \mu_{\mathcal{O}_2}(\mathsf{false} \mid s, w)$,
When $a_w = 1$:
$\mu_{\mathcal{O}_1}((o + 1 \bmod 3, b) \mid s, \mathsf{flip}(v, w)) = \mu_{\mathcal{O}_2}(\mathsf{false} \mid s, w)$,
$\mu_{\mathcal{O}_1}((o - 1 \bmod 3, b) \mid s, \mathsf{flip}(v, w)) = \mu_{\mathcal{O}_2}(\mathsf{true} \mid s, w)$.

To illustrate this, consider the case where $a_w = 0$. Let $s \in \mathcal{S}_1$ be any secret, and suppose that for $v \in \mathcal{I}_1$ with its w-th bit being 0, $P_1(s, v) = (o, b)$ for some $(o, b) \in \mathcal{O}_1$. Then when we flip the w-th bit of v, and only that bit, the output of $P_1(s, \mathsf{flip}(v, w))$ will either be $(o + 1 \bmod 3, b)$ or $(o - 1 \bmod 3, b)$ for some $b \in \{\mathsf{true}, \mathsf{false}\}$. In the first case, it means that the w-th bit of s is 1, since $o + 1 \bmod 3$ implies the number of correct bits between $\mathsf{flip}(v, w)$ and s has increased by 1 and the w-th bit of v is 0. Again, this can be inferred because we flipped exactly one bit in v. Similarly, if $P_1(s, \mathsf{flip}(v, w)) = (o - 1 \bmod 3, b)$ it means that the w-th bit of s is 0. Hence $P_2(s, w)$ will be true in the first case and false in the second case, as described above. The case where $a_w = 1$ is similar. This completes the argument.

VII. Related Work

Closely related to our work is the work by Köpf and Basin [10], [11] that laid the foundation for formalizing security under adaptive adversaries for deterministic systems. Importantly, they have introduced the notion of an *attack tree* that captures an attacker's strategy. Each node of the tree represents the attacker's current knowledge (or, *uncertainty*) about the secrets, with the root node being the initial state of his knowledge. Each node determines the next query input chosen by the attacker and a child node is added for each possible query output. They use quantitative information flow (QIF) to measure the amount of information an attacker gains by running a strategy. As standard in QIF [4], [8], [19], [25], this is defined as the change in the attacker's knowledge before and after the attack. They also show that maximum information leakage over all attack strategies of some bounded length can be computed by enumerating those strategies and computing the corresponding QIF. Boreale and Pampaloni [7] extended the line of work to probabilistic systems.

Our work is inspired by, and builds on the above characterization of attack strategies as attack trees. New in our work is the introduction of the *defender* player, which was missing in the previous works, thus making the formalism a full-blown *game*. Our game exactly captures security of systems, that is, the existence of a winning strategy for the defender player implies that the system is secure whereas that for the attacker player implies that the system is insecure. We remark that a correspondence between a particular winning strategy for these games and the attack trees could be made. For the attack trees of [7], a strategy for an n-round game would naturally correspond to a tree of height n, each of whose nodes would additionally be labeled with the $r_w(q)$ values chosen by Attacker. A similar correspondence would be produced in the deterministic case, for the attack trees of [10], [11].

Also, our results are given in the form of (f, ϵ)-security which asserts precise bounds on the number of attacker queries and the probability of his success, where the bounds can be *parametric* to the size of secrets. As we have shown, the game-based characterization is useful for deriving such parametric security bounds, as it allows strategy constructions that are parametric to the security parameter. In addition, building on the game-based formalism, we have proposed slopes and game reductions which can expedite the process of proving or disproving security.

As remarked before, the defense player's move in our game can be seen as her choosing a secret s internally and then exhibiting the corresponding output $o \in \mathsf{supp}(P(s, v))$ where v is the input chosen by the attacker. This may be seen as if the defender *changes* the secret as the game progresses. Mardziel et al. [14] investigates security against adaptive adversaries in a setting where the defender may change secrets across multiple attack queries. However, their work is orthogonal to ours since whereas their change of secrets is a physical one that actually changes the secret of the running system, ours is a conceptual means for proving or disproving security (defined by attackers' probability of success) over all secrets distributed according to some prior.

Finally, we remark that there are large body of works on using game theory for security [2], [3], [18], [21]. Like the work by Mardziel et al., these works are orthogonal to ours as they concern settings in which the defender makes choices that exert physical influences, such as choosing which system to run given some finite number of possibilities [2], [3]. It is interesting to note that, as shown in Remark III.4, probabilistic choices give no additional power to attackers in our setting

(this was also conjectured in [7], but not proved). This is in stark contrast to the setting with defenders making physical choices where probabilistic attackers are shown to be strictly more powerful than deterministic ones [2].

VIII. Conclusion

We have presented a new game-based characterization of security against adaptive adversaries. Our game is played by two players, Attacker and Defender, and it can be used to derive precise security bounds of deterministic and probabilistic systems. Importantly, the game allows one to derive security bounds that are parametric to the size of secrets.

In addition, leveraging the game, we have proposed techniques, called slopes and reductions, that can be used to expedite the process of deriving security bounds. The former provides a sound approximation of the bounds (for deterministic programs), and the latter can be used to convert an attacker strategy over one program to that of another, thereby allowing one reuse previously established secure or insecure instances to show security or insecurity of new ones.

Acknowledgments

We thank the anonymous reviewers for their useful comments and the meticulous examination they performed on this work. This work was supported by JSPS KAKENHI Grant Numbers 17H01720 and 18K19787, JSPS Core-to-Core Program, A.Advanced Research Networks, Office of Naval Research (ONR) award N00014-17-1-2787 and DARPA award FA8750-15-2-0104.

References

[1] J. Agat. Transforming out timing leaks. In *POPL*, 2000.

[2] M. S. Alvim, K. Chatzikokolakis, Y. Kawamoto, and C. Palamidessi. Information leakage games. In *8th International Conference on Game Theory for Security*, 2017.

[3] M. S. Alvim, K. Chatzikokolakis, Y. Kawamoto, and C. Palamidessi. Leakage and protocol composition in a game-theoretic perspective. In *POST*, 2018.

[4] M. S. Alvim, K. Chatzikokolakis, C. Palamidessi, and G. Smith. Measuring information leakage using generalized gain functions. In *CSF*, 2012.

[5] T. Antonopoulos, P. Gazzillo, M. Hicks, E. Koskinen, T. Terauchi, and S. Wei. Decomposition instead of self-composition for proving the absence of timing channels. In *PLDI*, 2017.

[6] G. Barthe, P. R. D'Argenio, and T. Rezk. Secure information flow by self-composition. *Mathematical Structures in Computer Science*, 21(6), 2011.

[7] M. Boreale and F. Pampaloni. Quantitative information flow under generic leakage functions and adaptive adversaries. *Logical Methods in Computer Science*, 11(4), 2015.

[8] D. Clark, S. Hunt, and P. Malacaria. Quantitative analysis of the leakage of confidential data. *Electr. Notes Theor. Comput. Sci.*, 59(3), 2001.

[9] G. Doychev, B. Köpf, L. Mauborgne, and J. Reineke. Cacheaudit: A tool for the static analysis of cache side channels. *ACM Transactions on Information and System Security*, 18(1), 2015.

[10] B. Köpf and D. A. Basin. An information-theoretic model for adaptive side-channel attacks. In *CCS*, 2007.

[11] B. Köpf and D. A. Basin. Automatically deriving information-theoretic bounds for adaptive side-channel attacks. *Journal of Computer Security*, 19(1), 2011.

[12] S. Langkemper. The password guessing bug in Tenex. https://www.sjoerdlangkemper.nl/2016/11/01/tenex-password-bug/, 2016.

[13] P. Li and S. Zdancewic. Downgrading policies and relaxed noninterference. In *POPL*, 2005.

[14] P. Mardziel, M. S. Alvim, M. W. Hicks, and M. R. Clarkson. Quantifying information flow for dynamic secrets. In *IEEE Symposium on Security and Privacy*, 2014.

[15] C. S. Pasareanu, Q. Phan, and P. Malacaria. Multi-run side-channel analysis using symbolic execution and max-SMT. In *CSF*, 2016.

[16] A. Sabelfeld and A. C. Myers. Language-based information-flow security. *IEEE Journal on Selected Areas in Communications*, 21(1), 2003.

[17] A. Sabelfeld and A. C. Myers. A model for delimited information release. In *ISSS*, 2003.

[18] A. Sinha, F. Fang, B. An, C. Kiekintveld, and M. Tambe. Stackelberg security games: Looking beyond a decade of success. In *IJICAI*, 2018.

[19] G. Smith. On the foundations of quantitative information flow. In *FOSSACS*, 2009.

[20] DARPA space/time analysis for cybersecurity (STAC) program. http://www.darpa.mil/program/space-time-analysis-for-cybersecurity, 2017.

[21] M. Tambe. *Security and game theory: algorithms, deployed systems, lessons learned*. Cambridge university press, 2011.

[22] T. Terauchi and A. Aiken. Secure information flow as a safety problem. In *SAS*, 2005.

[23] T. Terauchi and T. Antonopoulos. A formal analysis of timing channel security via bucketing. In *POST*, 2019.

[24] D. M. Volpano, C. E. Irvine, and G. Smith. A sound type system for secure flow analysis. *Journal of Computer Security*, 4(2/3), 1996.

[25] H. Yasuoka and T. Terauchi. On bounding problems of quantitative information flow. *Journal of Computer Security*, 19(6), 2011.

[26] S. Zdancewic and A. C. Myers. Robust declassification. In *CSFW*, 2001.

[27] D. Zhang, A. Askarov, and A. C. Myers. Language-based control and mitigation of timing channels. In *PLDI*, 2012.