A Formal Analysis of Timing Channel Security via Bucketing

Tachio Terauchi[1] and Timos Antonopoulos[2]

[1] Waseda University, Tokyo, Japan
terauchi@waseda.jp
[2] Yale University, New Haven, U.S.A.
timos.antonopoulos@yale.edu

Abstract. This paper investigates the effect of *bucketing* in security against timing channel attacks. Bucketing is a technique proposed to mitigate timing-channel attacks by restricting a system's outputs to only occur at designated time intervals, and has the effect of reducing the possible timing-channel observations to a small number of possibilities. However, there is little formal analysis on when and to what degree bucketing is effective against timing-channel attacks. In this paper, we show that bucketing is in general insufficient to ensure security. Then, we present two conditions that can be used to ensure security of systems against adaptive timing channel attacks. The first is a general condition that ensures that the security of a system decreases only by a limited degree by allowing timing-channel observations, whereas the second condition ensures that the system would satisfy the first condition when bucketing is applied and hence becomes secure against timing-channel attacks. A main benefit of the conditions is that they allow *separation of concerns* whereby the security of the regular channel can be proven independently of concerns of side-channel information leakage, and certain conditions are placed on the side channel to guarantee the security of the whole system. Further, we show that the bucketing technique can be applied compositionally in conjunction with the constant-time-implementation technique to increase their applicability. While we instantiate our contributions to timing channel and bucketing, many of the results are actually quite general and are applicable to any side channels and techniques that reduce the number of possible observations on the channel.

1 Introduction

Side-channel attacks aim to recover a computer system's secret information by observing the target system's side channels such as cache, power, timing and electromagnetic radiation [23, 24, 17, 31, 21, 36, 25, 16, 15, 11]. They are well recognized as a serious threat to the security of computer systems. *Timing-channel* (or simply *timing*) *attacks* are a class of side-channel attacks in which the adversary makes observations on the system's running time. Much research has been done to detect and prevent timing attacks [1, 22, 20, 26, 7, 27, 41, 4, 18, 3, 30, 6, 9].

Bucketing is a technique proposed for mitigating timing attacks [26, 27, 7, 41, 14]. It restricts the system's outputs to only occur at designated time intervals. Therefore, bucketing has the effect of reducing the possible timing-channel observations to a small number of possibilities. This is at some cost of system's performance because outputs must be delayed to the next bucket time. Nonetheless, in comparison to the *constant-time implementation* technique [1, 22, 20, 3, 6, 9] which restricts the system's running time to be independent of secrets, bucketing is often said to be more efficient and easier to implement as it allows running times to vary depending on secrets [26, 27].[3] For example, bucketing may be implemented in a blackbox-style by a monitor that buffers and delays outputs [7, 41].

In this paper, we formally study the effect of bucketing on security against *adaptive* timing attacks. To this end, first, we give a formal notion of security against adaptive side-channel-observing adversaries, called (f, ϵ)-*security*. Roughly, (f, ϵ)-security says that the probability that an adversary can recover the secret by making at most $f(n)$ many queries to the system is bounded by $\epsilon(n)$, where n is the security parameter.

Next, we show that bucketing alone is in general insufficient to guarantee security against adaptive side-channel attacks by presenting a counterexample that has only two timing observations and yet is efficiently attackable. This motivates a search for conditions sufficient for security. We present a condition, called *secret-restricted side-channel refinement* (SRSCR), which roughly says that a system is secure if there are sufficiently large subsets of secrets such that (1) the system's side channel reveals no more information than the regular channel on the subsets and (2) the system is secure on the subsets against adversaries who only observe the regular channel. The degree of security (i.e., f and ϵ) is proportional to that against regular-channel-only-observing adversaries and the size of the subsets.

Because of the insufficiency of bucketing mentioned above, applying bucketing to an arbitrary system may not lead to a system that satisfies SRSCR (for good f and ϵ). To this end, we present a condition, called *low-input side-channel non-interference* (LISCNI). We show that applying bucketing to a system that satisfies the condition would result in a system that satisfies SRSCR. Therefore, LISCNI is a sufficient condition for security under the bucketing technique. Roughly, LISCNI says that (1) the side-channel observation does not depend on attacker-controlled inputs (but may depend on secrets) and (2) the system is secure against adversaries who only observe the regular channel. The degree of security is proportional to that against regular-channel-only-observing adversaries and the granularity of buckets. A main benefit of the conditions SRSCR and LISCNI is that they allow *separation of concerns* whereby the security of the regular channel can be proven independently of concerns of side-channel

[3] Sometimes, the terminology "constant-time implementation" is used to mean even stricter requirements, such as requiring control flows to be secret independent [3, 9]. In this paper, we use the terminology for a more permissive notion in which only the running time is required to be secret independent.

information leakage, and certain conditions are placed on the side channel to guarantee the security of the whole system.

Finally, we show that the bucketing technique can be applied in a compositional manner with the constant-time implementation technique. Specifically, we show that when a system is a sequential composition of components in which one component is constant-time and the other component LISCNI, the whole system can be made secure by applying bucketing only to the non-constant-time part. We show that the combined approach is able to ensure security of some non-constant-time systems that cannot be made secure by applying bucketing to the whole system. We summarize the main contributions below.

- A formal notion of security against adaptive side-channel-observing adversaries, called (f, ϵ)-security. (Section 2)
- A counterexample which shows that bucketing alone is insufficient for security against adaptive side-channel attacks. (Section 2.1)
- A condition SRSCR which guarantees (f, ϵ)-security. (Section 3.1)
- A condition LISCNI which guarantees that the system satisfying it becomes one that satisfies SRSCR and therefore becomes (f, ϵ)-secure after suitable bucketing is applied. (Section 3.2)
- A compositional approach that combines bucketing and the constant-time technique. (Section 3.3)

While the paper focuses on timing channels and bucketing, many of the results are actually quite general and are applicable to side channels other than timing channels. Specifically, aside from the compositional bucketing result that exploits the "additive" nature of timing channels (cf. Section 3.3), the results are applicable to any side channels and techniques that reduce the number of possible side-channel observations

The rest of the paper is organized as follows. Section 2 formalizes the setting, and defines (f, ϵ)-security which is a formal notion of security against adaptive side-channel attacks. We also show that bucketing is in general insufficient to guarantee security of systems against adaptive side-channel attacks. Section 3 presents sufficient conditions for ensuring (f, ϵ)-security: SRSCR and LISCNI. We show that they facilitate proving the security of systems by allowing system designers to prove the security of regular channels separately from the concern of side channels. We also show that the LISCNI condition may be used in combination with the constant-time implementation technique in a compositional manner so as to prove the security of systems that are neither constant-time nor can be made secure by (globally) applying bucketing. Section 4 discusses related work. Section 5 concludes the paper with a discussion on future work.

2 Security against Adaptive Side-Channel Attacks

Formally, a *system* (or, *program*) is a tuple $(\mathsf{rc}, \mathsf{sc}, \mathcal{S}, \mathcal{I}, \mathcal{O}^{\mathsf{rc}}, \mathcal{O}^{\mathsf{sc}})$ where rc and sc are indexed families of functions (indexed by the security parameter) that

represent the regular-channel and side-channel input-output relation of the system, respectively. $\mathcal{S}$ is a security-parameter-indexed family of sets of *secrets* (or, *high inputs*) and $\mathcal{I}$ is a security-parameter-indexed family of sets of *attacker-controlled inputs* (or, *low inputs*). A *security parameter* is a natural number that represents the size of secrets, and we write $\mathcal{S}_n$ for the set of secrets of size n and $\mathcal{I}_n$ for the set of corresponding attacker-controlled inputs. Each indexed function rc_n (respectively sc_n) is a function from $\mathcal{S}_n \times \mathcal{I}_n$ to $\mathcal{O}^{\mathsf{rc}}_n$ (resp. $\mathcal{O}^{\mathsf{sc}}_n$), where $\mathcal{O}^{\mathsf{rc}}$ and $\mathcal{O}^{\mathsf{sc}}$ are indexed families of sets of possible regular-channel and side-channel outputs, respectively. For $(s, v) \in \mathcal{S}_n \times \mathcal{I}_n$, we write $\mathsf{rc}_n(s, v)$ (resp. $\mathsf{sc}_n(s, v)$) for the regular-channel (resp. side-channel) output given the secret s and the attacker-controlled input v.[4] For a system $C = (\mathsf{rc}, \mathsf{sc}, \mathcal{S}, \mathcal{I}, \mathcal{O}^{\mathsf{rc}}, \mathcal{O}^{\mathsf{sc}})$, we often write $\mathsf{rc}\langle C\rangle$ for rc, $\mathsf{sc}\langle C\rangle$ for sc, $\mathcal{S}\langle C\rangle$ for $\mathcal{S}$, $\mathcal{I}\langle C\rangle$ for $\mathcal{I}$, $\mathcal{O}^{\mathsf{rc}}\langle C\rangle$ for $\mathcal{O}^{\mathsf{rc}}$, and $\mathcal{O}^{\mathsf{sc}}\langle C\rangle$ for $\mathcal{O}^{\mathsf{sc}}$. We often omit "$\langle C\rangle$" when it is clear from the context.

For a system C and $s \in \mathcal{S}_n$, we write $C_n(s)$ for the *oracle* which, given $v \in \mathcal{I}_n$, returns a pair of outputs $(o_1, o_2) \in \mathcal{O}^{\mathsf{rc}}_n \times \mathcal{O}^{\mathsf{sc}}_n$ such that $\mathsf{rc}_n(s, v) = o_1$ and $\mathsf{sc}_n(s, v) = o_2$. An *adversary* $\mathcal{A}$ is an algorithm that attempts to discover the secret by making some number of oracle queries. As standard, we assume that $\mathcal{A}$ has the full knowledge of the system. For $i \in \mathbb{N}$, we write $\mathcal{A}^{C_n(s)}(i)$ for the adversary $\mathcal{A}$ that makes at most i oracle queries to $C_n(s)$. We impose no restriction on how the adversary chooses the inputs to the oracle. Importantly, he may choose the inputs based on the outputs of previous oracle queries. Such an adversary is said to be *adaptive* [25].

Also, for generality, we intentionally leave the computation class of adversaries unspecified. The methods presented in this paper work for any computation class, including the class of polynomial time randomized algorithms and the class of resource-unlimited randomized algorithms. The former is the standard for arguing the security of cryptography algorithms, and the latter ensures information theoretic security. In what follows, unless specified otherwise, we assume that the computation class of adversaries is the class of resource-unlimited randomized algorithms.

As standard, we define security as the bound on the probability that an adversary wins a certain game. Let f be a function from $\mathbb{N}$ to $\mathbb{N}$. We define $\mathit{Win}_{\mathcal{A}}(n, f)$ to be the event that the following game outputs true.

$$
\begin{array}{l}
s \leftarrow \mathcal{S}_n \\
s' \leftarrow \mathcal{A}^{C_n(s)}(f(n)) \\
\mathsf{Output}\ s = s'
\end{array}
$$

Here, the first line selects s uniformly at random from $\mathcal{S}_n$. We note that, while we restrict to deterministic systems, the adversary algorithm $\mathcal{A}$ may be probabilistic and also the secret s is selected randomly. Therefore, the full range of probabilities is possible for the event $\mathit{Win}_{\mathcal{A}}(n, f)$. Now, we are ready to give the definition of (f, ϵ)-security.

[4] We restrict to deterministic systems in this paper. Extension to probabilistic systems is left for future work.

```
i = 0;
while (i < n) {
  if (pass[i] != guess[i]) return false;
  i++;
}
return true;
```

Fig. 1. Timing insecure login program

Definition 1 ((f,ϵ)-security). Let $f : \mathbb{N} \to \mathbb{N}$ and $\epsilon : \mathbb{N} \to \mathbb{R}$ be such that $0 < \epsilon(n) \leq 1$ for all $n \in \mathbb{N}$. We say that a system is (f,ϵ)-*secure* if there exists $N \in \mathbb{N}$ such that for all adversaries $\mathcal{A}$ and $n \geq N$, it holds that $\Pr[Win_{\mathcal{A}}(n,f)] < \epsilon(n)$.

Roughly, (f,ϵ)-secure means that, for all sufficiently large n, there is no attack that is able to recover secrets in $f(n)$ number of queries with the probability of success $\epsilon(n)$.

By abuse of notation, we often implicitly treat an expression e on the security parameter n as the function $\lambda n{\in}\mathbb{N}.e$. Therefore, for example, (n,ϵ)-secure means that there is no attack that is able to recover secrets in n many queries with the probability of success $\epsilon(n)$, and $(f,1)$-secure means that there is no attack that makes at most $f(n)$ number of queries and is always successful. Also, by abuse of notation, we often write $\epsilon \leq \epsilon'$ when $\epsilon(n) \leq \epsilon'(n)$ for all sufficiently large n, and likewise for $\epsilon < \epsilon'$.

Example 1 (Leaky Login). Consider the program shown in Fig. 1 written in a C-like language. The program is an abridged version of the timing insecure login program from [6]. Here, `pass` is the secret and `guess` is the attacker-controlled input, each represented as a length n bit array. We show that there is an efficient adaptive timing attack against the program that recovers the secret in a linear number of queries.

We formalize the program as the system C where for all $n \in \mathbb{N}$,

- $\mathcal{S}_n = \mathcal{I}_n = \{0,1\}^n$;
- $\mathcal{O}_n^{\mathsf{rc}} = \{\texttt{true}, \texttt{false}\}$ and $\mathcal{O}_n^{\mathsf{sc}} = \{i \in \mathbb{N} \mid i \leq n\}$;
- For all $(s,v) \in \mathcal{S}_n \times \mathcal{I}_n$, $\mathsf{rc}_n(s,v) = \texttt{true}$ if $s = v$ and $\mathsf{rc}_n(s,v) = \texttt{false}$ if $s \neq v$; and
- For all $(s,v) \in \mathcal{S}_n \times \mathcal{I}_n$, $\mathsf{sc}_n(s,v) = (\mathrm{argmax}_i\, s{\restriction_i} = v{\restriction_i})$.

Here, $a{\restriction_i}$ denotes the length i prefix of a. Note that sc expresses the timing-channel observation, as its output corresponds to the number of times the loop iterated.

For a secret $s \in \mathcal{S}_n$, the adversary $\mathcal{A}^{C_n(s)}(n)$ efficiently recovers s as follows. He picks an arbitrary $v_1 \in \mathcal{I}_n$ as the initial guess. By seeing the timing-channel output $\mathsf{sc}_n(s,v_1)$, he would be able to discover at least the first bit of s, $s[0]$, because $s[0] = v_1[0]$ if and only if $\mathsf{sc}_n(s,v_1) > 0$. Then, he picks an arbitrary $v_2 \in \{0,1\}^n$ satisfying $v_2[0] = s[0]$, and by seeing the timing-channel output, he would be able to discover at least up to the second bit of s. Repeating the process

n times, he will recover all n bits of s. Therefore, the system is not (n, ϵ)-secure for any ϵ. This is an example of an adaptive attack since the adversary crafts the next input by using the knowledge of previous observations. ▲

Example 2 (Bucketed Leaky Login). Next, we consider the security of the program from Example 1 but with bucketing applied. Here, we assume a constant number of buckets, k, such that the program returns its output at time intervals $i \cdot n/k$ for $i \in \{j \in \mathbb{N} \mid j \leq k\}$.[5] (For simplicity, we assume that n is divisible by k.) The bucketed program can be formalized as the system where

- rc, sc, $\mathcal{I}$, $\mathcal{O}^{\mathsf{rc}}$ are as in Example 1;
- For all $n \in \mathbb{N}$, $\mathcal{O}^{\mathsf{sc}}_n = \{i \in \mathbb{N} \mid i \leq k\}$; and
- For all $n \in \mathbb{N}$ and $(s, v) \in \mathcal{S}_n \times \mathcal{I}_n$, $\mathsf{sc}_n(s, v) = \mathit{bkt}(\mathrm{argmax}_i\, s{\restriction_i} = v{\restriction_i}, n/k)$

where $\mathit{bkt}(i, j)$ is the smallest $a \in \mathbb{N}$ such that $i \leq a \cdot j$. It is easy to see that the system is not constant-time for any $k > 1$. Nonetheless, we can show that the system is (f, ϵ)-secure where $f(n) = 2^{n/k} - (N + 1)$ and $\epsilon(n) = 1 - \frac{N-1}{2^{n/k}}$ for any $1 \leq N < 2^{n/k}$. Note that as k approaches 1 (and hence the system becomes constant-time), f approaches $2^n - (N + 1)$ and ϵ approaches $1 - \frac{N-1}{2^n}$, which match the security bound of the ideal login program that only leaks whether the input guess matched the password or not. We will show that the approach presented in Section 3.1 can be used to derive such a bound. ▲

2.1 Insufficiency of Bucketing

We show that bucketing is in general insufficient to guarantee the security of systems against adaptive side-channel attacks. In fact, we show that bucketing with even just two buckets is insufficient. (Two is the minimum number of buckets that can be used to show the insufficiency because having only one bucket implies that the system is constant-time and therefore is secure.) More generally, our result applies to any side channels, and it shows that there are systems with just two possible side-channel outputs and completely secure (i.e., non-interferent [19, 37]) regular channel that is efficiently attackable by side-channel-observing adversaries.

Consider the system such that, for all $n \in \mathbb{N}$,

- $\mathcal{S}_n = \{0, 1\}^n$ and $\mathcal{I}_n = \{i \in \mathbb{N} \mid i \leq n\}$;
- $\mathcal{O}^{\mathsf{rc}}_n = \{\bullet\}$ and $\mathcal{O}^{\mathsf{sc}}_n = \{0, 1\}$;
- For all $(s, v) \in \mathcal{S}_n \times \mathcal{I}_n$, $\mathsf{rc}_n(s, v) = \bullet$; and
- For all $(s, v) \in \mathcal{S}_n \times \mathcal{I}_n$, $\mathsf{sc}_n(s, v) = s[v]$.

Note that the regular channel rc only has one possible output and therefore is non-interferent. The side channel sc has just two possible outputs. The side channel, given an attacker-controlled input $v \in \mathcal{I}_n$, reveals the v-th bit of s. It is easy to see that the system is linearly attackable. That is, for any secret $s \in \mathcal{S}_n$, the adversary may recover the entire n bits of s by querying with each

[5] A similar analysis can be done for any strictly sub-linear number of buckets.

of the n-many possible attacker-controlled inputs. Therefore, the system is not (n, ϵ)-secure for any ϵ. Note that the side channel is easily realizable as a timing channel, for example, by having a branch with the branch condition "$s[v] = 0$" and different running times for the branches.

We remark that the above attack is not adaptive. Therefore, the counterexample actually shows that bucketing can be made ineffective by just allowing multiple non-adaptive side-channel observations. We also remark that the counterexample shows that some previously proposed measures are insufficient. For example, the *capacity* measure [28, 33, 5, 39] would not be able to detect the vulnerability of the example, because the measure is equivalent to the log of the number of possible outputs for deterministic systems.

3 Sufficient Conditions for Security against Adaptive Side-Channel Attacks

In this section, we present conditions that guarantee the security of systems against adaptive side-channel-observing adversaries. The condition SRSCR presented in Section 3.1 guarantees that systems that satisfy it are secure, whereas the condition LISCNI presented in Section 3.2 guarantees that systems that satisfy it become secure once bucketing is applied. We shall show that the conditions facilitate proving (f, ϵ)-security of systems by separating the concerns of regular channels from those of side channels. In addition, we show in Section 3.3 that the LISCNI condition may be used in combination with constant-time implementation techniques in a compositional manner so as to prove the security of systems that are neither constant-time nor can be made secure by (globally) applying bucketing.

3.1 Secret-Restricted Side-Channel Refinement Condition

We present the *secret-restricted side-channel refinement* condition (SRSCR). Informally, the idea here is to find large subsets of secrets $S' \subseteq \mathcal{P}(\mathcal{S}_n)$ such that for each $S'' \in S'$, the secrets are difficult for an adversary to recover by only observing the regular channel, and that the side channel reveals no more information than the regular channel for those sets of secrets. Then, because S' is large, the entire system is also ensured to be secure with high probability. We adopt *refinement order* [38, 29], which had been studied in quantitative information flow research, to formalize the notion of "reveals no more information". Roughly, a channel C_1 is said to be a refinement of a channel C_2 if, for every attacker-controlled input, every pair of secrets that C_2 can distinguish can also be distinguished by C_1.

We write $\mathcal{O}^\bullet$ for the indexed family of sets such that $\mathcal{O}^\bullet_n = \{\bullet\}$ for all $n \in \mathbb{N}$. Also, we write $\mathsf{sc}^\bullet$ for the indexed family of functions such that $\mathsf{sc}^\bullet_n(s, v) = \bullet$ for all $n \in \mathbb{N}$ and $(s, v) \in \mathcal{S}_n \times \mathcal{I}_n$. For $C = (\mathsf{rc}, \mathsf{sc}, \mathcal{S}, \mathcal{I}, \mathcal{O}^{\mathsf{rc}}, \mathcal{O}^{\mathsf{sc}})$, we write $C^\bullet$ for the system $(\mathsf{rc}, \mathsf{sc}^\bullet, \mathcal{S}, \mathcal{I}, \mathcal{O}^{\mathsf{rc}}, \mathcal{O}^\bullet)$. We define the notion of *regular-channel security*.

Definition 2 (Regular-channel (f,ϵ)-security). We say that the C is *regular-channel (f,ϵ)-secure* if $C^\bullet$ is (f,ϵ)-secure.

Roughly, regular-channel security says that the system is secure against attacks that only observe the regular channel output.

Let us fix a system $C = (\mathsf{rc}, \mathsf{sc}, \mathcal{S}, \mathcal{I}, \mathcal{O}^{\mathsf{rc}}, \mathcal{O}^{\mathsf{sc}})$. For an indexed family of sets of sets of secrets S' (i.e., $S'_n \subseteq \mathcal{P}(\mathcal{S}_n)$ for each n), we write $S'' \prec S'$ when S'' is an indexed family of sets of secrets such that $S''_n \in S'_n$ for each n. Note that such S'' satisfies $S''_n \subseteq \mathcal{S}_n$ for each n. Also, for $S'' \prec S'$, we write $C|_{S''}$ for the system that is equal to C except that its secrets are restricted to S'', that is, $(\mathsf{rc}, \mathsf{sc}, S'', \mathcal{I}, \mathcal{O}^{\mathsf{rc}}, \mathcal{O}^{\mathsf{sc}})$. For a set of sets X, we say that X is *mutually disjoint* if for all $S_1 \in X$ and $S_2 \in X$, $S_1 \cap S_2 = \emptyset$. Next, we formalize the SRSCR condition.

Definition 3 (Secret-Restricted Side-Channel Refinement). Let $f : \mathbb{N} \to \mathbb{N}$, $\epsilon : \mathbb{N} \to (0,1]$, and $0 < r \leq 1$. We say that the system $C = (\mathsf{rc}, \mathsf{sc}, \mathcal{S}, \mathcal{I}, \mathcal{O}^{\mathsf{rc}}, \mathcal{O}^{\mathsf{sc}})$ satisfies the *secret-restricted side-channel refinement* condition with f, ϵ, and r, written $\mathsf{SRSCR}(f,\epsilon,r)$, if there exists an indexed family of sets of sets of secrets S^{res} such that S^{res}_n is mutually disjoint and $S^{res}_n \subseteq \mathcal{P}(\mathcal{S}_n)$ for all $n \in \mathbb{N}$, and:

(1) For all $n \in \mathbb{N}$, $r \leq |\bigcup S^{res}_n| / |\mathcal{S}_n|$;
(2) For all $S'' \prec S^{res}$, $C|_{S''}$ is regular-channel (f,ϵ)-secure; and
(3) For all $n \in \mathbb{N}$, $S \in S^{res}_n$, $v \in \mathcal{I}_n$ and $s_1, s_2 \in S$, it holds that $\mathsf{sc}_n(s_1, v) \neq \mathsf{sc}_n(s_2, v) \Rightarrow \mathsf{rc}_n(s_1, v) \neq \mathsf{rc}_n(s_2, v)$.

Condition (2) says that the system is regular-channel (f,ϵ)-secure when restricted to any subset of secrets $S'' \prec S^{res}$. Condition (3) says that the system's side channel reveals no more information than its regular channel for the restricted secret subsets. Condition (1) says that the ratio of the restricted set over the entire space of secrets is at least r.[6]

We informally describe why SRSCR is a sufficient condition for security. The condition guarantees that, for the restricted secrets S^{res}, the attacker gains no additional information by observing the side-channel compared to what he already knew by observing the regular channel. Then, because r is a bound on the probability that a randomly selected secret falls in S^{res}, the system is secure provided that r is suitably large and the system is regular-channel secure. The theorem below formalizes the above intuition.

Theorem 1 (SRSCR Soundness). *Suppose C satisfies $\mathsf{SRSCR}(f,\epsilon,r)$. Then, C is (f,ϵ')-secure, where $\epsilon' = 1 - r(1-\epsilon)$.*

Proof. Let S^{res} be an indexed family of sets of secret subsets that satisfies conditions (1), (2), and (3) of $\mathsf{SRSCR}(f,\epsilon,r)$. By condition (2), for all sufficiently large n and adversaries $\mathcal{A}$, $\Pr[Win^{\bullet,res}_{\mathcal{A}}(n,f)] < \epsilon(n)$ where $Win^{\bullet,res}_{\mathcal{A}}(n,f)$ is the modified game in which the oracle oracle $C_n(s)$ always outputs $\bullet$ as its side-channel output and the secret s is selected randomly from $\bigcup S^{res}_n$ (rather than from $\mathcal{S}_n$).

[6] It is easy to relax the notion to be asymptotic so that the conditions need to hold only for $n \geq N$ for some $N \in \mathbb{N}$.

For any n, the probability that a randomly selected element from $\mathcal{S}_n$ is in $\bigcup S_n^{res}$ is at least r by condition (1). That is, $\Pr[s \in \bigcup S_n^{res} \mid s \leftarrow \mathcal{S}_n] \geq r$. Also, $\Pr[\neg \mathit{Win}_{\mathcal{A}}^{\bullet,res}(n,f)] > 1-\epsilon(n)$ (for sufficiently large n) for any $\mathcal{A}$ by the argument above. Therefore, by condition (3), for sufficiently large n,

$$\Pr[\neg \mathit{Win}_{\mathcal{A}}(n,f)] \geq \Pr[s \in S_n^{res} \mid s \leftarrow \bigcup \mathcal{S}_n] \cdot \Pr[\neg \mathit{Win}_{\mathcal{A}}^{\bullet,res}(n,f)] > r \cdot (1-\epsilon(n))$$

Therefore, $\Pr[\mathit{Win}_{\mathcal{A}}(n,f)] < 1 - r(1-\epsilon(n))$ for sufficiently large n. □

As a special case where the ratio r is 1, Theorem 1 implies that if a system satisfies $\mathsf{SRSCR}(f,\epsilon,1)$ then it is (f,ϵ)-secure.

Example 3. Recall the bucketed leaky login program from Example 2. We show that the program satisfies the SRSCR condition. For each n, $a \in \{0,1\}^n$, and $0 \leq i < k$, let $S_n^{a,i} \subseteq \mathcal{S}_n$ be the set of secrets whose sub-bits from $i \cdot n/k$ to $(i+1) \cdot n/k - 1$ may differ but the remaining $n - n/k$ bits are a (and therefore same). That is,

$$S_n^{a,i} = \{s \in \mathcal{S}_n \mid s[0,\ldots,i \cdot n/k - 1] = a[0,\ldots,i \cdot n/k - 1] \\ \text{and } s[(i+1) \cdot n/k,\ldots,n-1] = a[(i+1) \cdot n/k,\ldots,n-1]\}$$

Let S^{res} be the indexed family of sets of sets of secrets such that $S_n^{res} = \{S_n^{a,i} \mid a \in \{0,1\}^n\}$ for some i. Then, the system satisfies conditions (1), (2), and (3) of $\mathsf{SRSCR}(f,\epsilon,r)$ with $r=1$, $f(n) = 2^{n/k} - (N+1)$, and $\epsilon = 1 - \frac{N-1}{2^{n/k}}$ for any $1 \leq N < 2^{n/k}$. Note that (1) is satisfied with $r=1$ because $\mathcal{S}_n = \bigcup S_n^{res}$, and (2) is satisfied because $|S_n^{a,i}| = 2^{n/k}$ and (f,ϵ) matches the security of the ideal login program without side channels for the set of secrets of size $2^{n/k}$. To see why (3) is satisfied, note that for any $v \in \mathcal{I}_n$ and $s \in S_n^{a,i}$, $\mathsf{sc}_n(s,v) = i$ if $s \neq v$, and $\mathsf{sc}_n(s,v) = k$ if $s = v$. Hence, for any $v \in \mathcal{I}_n$ and $s_1, s_2 \in S_n^{a,i}$, $\mathsf{sc}_n(s_1,v) \neq \mathsf{sc}_n(s_2,v) \Rightarrow \mathsf{rc}_n(s_1,v) \neq \mathsf{rc}_n(s_2,v)$. Therefore, by Theorem 1, it follows that bucketed leaky login program is (f,ϵ)-secure. Note that the bound matches the one given in Example 2. ▲

To effectively apply Theorem 1, one needs to find suitable subsets of secrets S^{res} on which the system's regular channel is (f,ϵ)-secure and the side channel satisfies the refinement relation with respect to the regular channel. As also observed in prior works [38, 29], the refinement relation is a 2-safety property [35, 13] for which there are a number of effective verification methods [32, 10, 34, 2, 6]. For instance, self-composition [35, 8, 4, 3] is a well-known technique that can be used to verify arbitrary 2-safety properties.

We note that a main benefit of Theorem 1 is *separation of concerns* whereby the security of regular channel can be proven independently of side channels, and the conditions required for side channels can be checked separately. For instance, a system designer may prove the regular-channel (f,ϵ)-security by an elaborate manual reasoning, while the side-channel conditions are checked, possibly automatically, by established program verification methods such as self composition.

Remarks. We make some additional observations regarding the SRSCR condition. First, while Theorem 1 derives a sound security bound, the bound may not be the tightest one. Indeed, when the adversary's error probability (i.e., the "ϵ" part of (f, ϵ)-security) is 1, the bucketed leaky login program can be shown to be actually $(k(2^{n/k} - 2), 1)$-secure, whereas the bound derived in Example 3 only showed that it is $(2^{n/k} - 2, 1)$-secure. That is, there is a factor k gap in the bounds. Intuitively, the gap occurs for the example because the buckets partition a secret into k number of n/k bit blocks, and while an adversary needs to recover the bits of every block in order to recover the entire secret, the analysis derived the bound by assessing only the effort required to recover bits from one of the blocks. Extending the technique to enable tighter analyses is left for future work.

Secondly, the statement of Theorem 1 says that when regular channel of the system is (f, ϵ)-secure for certain subsets of secrets, then the whole system is (f, ϵ')-secure under certain conditions. This may give an impression that only the adversary-success probability parameter (i.e., ϵ) of (f, ϵ)-security is affected by the additional consideration of side channels, leaving the number of oracle queries parameter (i.e., f) unaffected. However, as also seen in Example 2, the two parameters are often correlated so that smaller f implies smaller ϵ and vice versa. Therefore, Theorem 1 suggests that the change in the probability parameter (i.e., from ϵ to ϵ') may need to be compensated by a change in the degree of security with respect to the number of oracle queries.

Finally, condition (2) of SRSCR stipulates that the regular channel is (f, ϵ)-secure for each restricted family of sets of secrets $S'' \prec S^{res}$ rather than the entire space of secrets $\mathcal{S}$. In general, a system can be less secure when secrets are restricted because the adversary has a smaller space of secrets to search. Indeed, in the case when the error probability is 1, the regular channel of the bucketed leaky login program can be shown to be $(2^n - 2, 1)$-secure, but when restricted to each $S'' \prec S^{res}$ used in the analysis of Example 3, it is only $(2^{n/k} - 2, 1)$-secure. That is, there is an implicit correlation between the sizes of the restricted subsets and the degree of regular-channel security. Therefore, finding S^{res} such that each $S'' \in S^{res}_n$ is large and satisfies the conditions is important for deriving good security bounds, even when the ratio $|\bigcup S^{res}_n|/|\mathcal{S}_n|$ is large as in the analysis of the bucketed leaky login program.

3.2 Low-Input Side-Channel Non-Interference Condition

While SRSCR facilitates proving security of systems by separating regular channels from side channels, it requires one to identify suitable subsets of secrets S^{res} that satisfy the conditions. This can be a hurdle to applying the proof method. To this end, this section presents a condition, called *low-input side-channel non-interference* (LISCNI), which guarantees that a system satisfying it becomes secure after applying bucketing (or other techniques) to reduce the number of side-channel outputs. Unlike SRSCR, the condition does not require identifying secret subsets. Roughly, the condition stipulates that the regular channel is secure (for the entire space of secrets) and that the side-channel outputs are independent of attacker-controlled inputs.

We show that the system satisfying the condition becomes a system satisfying SRSCR once bucketing is applied, where the degree of security (i.e., the parameters f, ϵ, r of SRSCR) will be proportional to the degree of regular-channel security and the granularity of buckets. Roughly, this holds because for a system whose side-channel outputs are independent of attacker-controlled inputs, bucketing is guaranteed to partition the secrets into a small number of sets (relative to the bucket granularity) such that for each of the sets, the side channel cannot distinguish the secrets in the set, and the regular-channel security transfers to a certain degree to the case when the secrets are restricted to the ones in the set.

As we shall show next, while the condition is not permissive enough to prove security of the leaky login program (cf. Examples 1, 2 and 3), it covers interesting scenarios such as fast modular exponentiation (cf. Example 4). Also, as we shall show in Section 3.3, the condition may be used compositionally in combination with the constant-time implementation technique [1, 22, 3, 9] to further widen its applicability.

Definition 4 (Low-Input Side-Channel Non-Interference). Let $f : \mathbb{N} \to \mathbb{N}$ and $\epsilon : \mathbb{N} \to (0, 1]$. We say that the system C satisfies the *low-input side-channel non-interference* condition with f and ϵ, written $\mathsf{LISCNI}(f, \epsilon)$, if the following conditions are satisfied:

(1) C is regular-channel (f, ϵ)-secure; and
(2) For all $n \in \mathbb{N}$, $s \in \mathcal{S}_n$, and $v_1, v_2 \in \mathcal{I}_n$, it holds that $\mathsf{sc}_n(s, v_1) = \mathsf{sc}_n(s, v_2)$.

Condition (2) says that the side-channel outputs are independent of low inputs (i.e., attacker-controlled inputs). We note that this is *non-interference* with respect to low inputs, whereas the usual notion of non-interference says that the outputs are independent of high inputs (i.e., secrets) [19, 37].[7]

The LISCNI condition ensures the security of systems after bucketing is applied. We next formalize the notion of "applying bucketing".

Definition 5 (Bucketing). Let C be a system and $k \in \mathbb{N}$ such that $k > 0$. The system C after *k-bucketing* is applied, written $Bkt_k(C)$, is a system C' that satisfies the following:

(1) $\mathsf{rc}\langle C' \rangle = \mathsf{rc}\langle C \rangle$, $\mathcal{S}\langle C' \rangle = \mathcal{S}\langle C \rangle$, $\mathcal{I}\langle C' \rangle = \mathcal{I}\langle C \rangle$, and $\mathcal{O}^{\mathsf{rc}}\langle C' \rangle = \mathcal{O}^{\mathsf{rc}}\langle C \rangle$;
(2) For all $n \in \mathbb{N}$, $\mathcal{O}^{\mathsf{sc}}\langle C' \rangle_n = \{\star_1, \ldots, \star_k\}$ where $\star_i \neq \star_j$ for each $i \neq j$; and
(3) For all $n \in \mathbb{N}$, $s_1, s_2 \in \mathcal{S}_n$ and $v_1, v_2 \in \mathcal{I}_n$, $\mathsf{sc}\langle C \rangle_n(s_1, v_1) = \mathsf{sc}\langle C \rangle_n(s_2, v_2) \Rightarrow \mathsf{sc}\langle C' \rangle_n(s_1, v_2) = \mathsf{sc}\langle C' \rangle_n(s_2, v_2)$.

Roughly, k-bucketing partitions the side channel outputs into k number of buckets. We note that our notion of "bucketing" is quite general in that it does not specify how the side channel outputs are partitioned into the buckets. Indeed, as we shall show next, the security guarantee derived by LISCNI only requires the fact that side channel outputs are partitioned into a small number of buckets.

[7] As with SRSCR, it is easy to relax the notion to be asymptotic so that condition (2) only needs to hold for large n.

This makes our results applicable to any techniques (beyond the usual bucketing technique for timing channels [26, 27, 7, 41, 14]) that reduce the number of possible side-channel outputs.

Below states that a system satisfying the LISCNI condition becomes one that satisfies the SRSCR condition after suitable bucketing is applied.

Theorem 2 (LISCNI Soundness). *Suppose that C satisfies* $\mathsf{LISCNI}(f, \epsilon)$. *Let $k > 0$ be such that $k \cdot \epsilon \leq 1$. Then, $Bkt_k(C)$ satisfies* $\mathsf{SRSCR}(f, k \cdot \epsilon, 1/k)$.

Proof. Let $C' = Bkt_k(C)$. By condition (2) of k-bucketing and condition (2) of $\mathsf{LISCNI}(f, \epsilon)$, we have that for all $n \in \mathbb{N}$, $s \in \mathcal{S}_n$ and $v_1, v_2 \in \mathcal{I}_n$, $\mathsf{sc}\langle C' \rangle_n(s, v_1) = \mathsf{sc}\langle C' \rangle_n(s, v_2)$. Therefore, by k-bucketing, there must be an indexed family of sets of secrets S' such that for all n, (a) $S'_n \subseteq \mathcal{S}_n$, (b) $|S'_n| \geq |\mathcal{S}_n|/k$, and (c) for all $s_1, s_2 \in S'_n$ and $v_1, v_2 \in \mathcal{I}_n$, $\mathsf{sc}\langle C' \rangle_n(s_1, v_1) = \mathsf{sc}\langle C' \rangle_n(s_2, v_2)$. Note that such S' can be found by, for each n, choosing a bucket into which a maximal number of secrets fall. We define an indexed family of sets of sets of secrets S^{res} to be such that S^{res}_n is the singleton set $\{S'_n\}$ for each n.

We show that C' satisfies conditions (1), (2), and (3) of $\mathsf{SRSCR}(f, k \cdot \epsilon, 1/k)$ with the restricted secret subsets S^{res} defined above. Firstly, (1) is satisfied because $|S'_n| \geq |\mathcal{S}_n|/k$. Also, (3) is satisfied because of property (c) above (i.e., the side channel is non-interferent for the subset).

It remains to show that (2) is satisfied. That is, $C'|_{S'}$ is regular-channel $(f, k \cdot \epsilon)$-secure. For contradiction, suppose that $C'|_{S'}$ is not regular-channel $(f, k \cdot \epsilon)$-secure, that is, there exists a regular-channel attack $\mathcal{A}$ that queries (the regular channel of) $C'|_{S'}$ at most $f(n)$ many times and successfully recovers the secret with probability at least $k \cdot \epsilon(n)$. Then, we can construct a regular-channel adversary for C which simply runs $\mathcal{A}$ (on any secret from $\mathcal{S}_n$). Note that the adversary makes at most $f(n)$ many queries. We argue that the probability that the adversary succeeds in recovering the secret is at least ϵ. That is, we show that $\Pr[Win^{\bullet}_{\mathcal{A}}(n, f)] \geq \epsilon(n)$ (for sufficiently large n) where $Win^{\bullet}_{\mathcal{A}}(n, f)$ is the modified game in which the oracle always outputs $\bullet$ as its side-channel output.

To see this, note that the probability that a secret randomly selected from $\mathcal{S}_n$ is in S'_n is at least $1/k$, that is, $\Pr[s \in S'_n \mid s \leftarrow \mathcal{S}_n] \geq 1/k$. Also, $\mathcal{A}$'s regular-channel attack succeeds with probability at least $k \cdot \epsilon$ given a randomly chosen secret from S'_n, that is, $\Pr[Win^{\bullet,res}_{\mathcal{A}}(n, f)] \geq k \cdot \epsilon(n)$ where $Win^{\bullet,res}_{\mathcal{A}}(n, f)$ is the modified game in which the oracle always outputs $\bullet$ as its side-channel output and the secret is selected randomly from S'_n (rather than from $\mathcal{S}_n$). Therefore, for sufficiently large n, we have:

$$\Pr[Win^{\bullet}_{\mathcal{A}}(n, f)] \geq \Pr[s \in S'_n \mid s \leftarrow \mathcal{S}_n] \cdot \Pr[Win^{\bullet,res}_{\mathcal{A}}(n, f)] \geq 1/k \cdot (k \cdot \epsilon(n)) = \epsilon(n)$$

This contradicts condition (1) of $\mathsf{LISCNI}(f, \epsilon)$ which says that C is regular-channel (f, ϵ)-secure. Therefore, $C'|_{S'}$ is regular-channel $(f, k \cdot \epsilon)$-secure. □

As a corollary of Theorems 1 and 2, we have the following.

Corollary 1. *Suppose that C satisfies* $\mathsf{LISCNI}(f, \epsilon)$. *Let $k > 0$ be such that $k \cdot \epsilon \leq 1$. Then, $Bkt_k(C)$ is (f, ϵ')-secure where $\epsilon' = 1 - 1/k + \epsilon$.*

```
i = 0;
a = 1;
while (i < n) {
  if (x[i] == 1) {
    r = (a * y) % m;
  } else {
    r = a;
  }
  a = (r * r) % m;
  i++;
}
return r;
```

Fig. 2. Fast modular exponentiation

Note that as k approaches 1 (and hence the system becomes constant-time), the security bound of $Bkt_k(C)$ approaches (f, ϵ), matching the regular-channel security of C. As with Theorem 1, Theorem 2 may give an impression that the conditions only affect the adversary-success probability parameter (i.e., ϵ) of (f, ϵ)-security, leaving the number of queries parameter (i.e., f) unaffected. However, as also remarked in Section 3.1, the two parameters are often correlated so that a change in one can affect the other. Also, like SRSCR, LISCNI separates the concerns regarding regular channels from those regarding side channels. A system designer may check the security of the regular channel while disregarding the side channel, and separately prove the condition on the side channel.

Example 4 (Fast Modular Exponentiation). Fast modular exponentiation is an operation that is often found in cryptography algorithms such as RSA [23, 30]. Fig. 2 shows its implementation written in a C-like language. It computes $\mathtt{y}^{\mathtt{x}}\ \mathtt{mod}\ \mathtt{m}$ where `x` is the secret represented as a length `n` bit array and `y` is an attacker controlled-input. The program is not constant-time (assuming that then and else branches in the loop have different running times), and effective timing attacks have been proposed for the program [23, 30].

However, assuming that running time of the operation `(a * y) % m` is independent of `y`, it can be seen that the program satisfies the LISCNI condition.[8] Under the assumption, the program can be formalized as the system C where, for all $n \in \mathbb{N}$,

- $\mathcal{S}_n = \mathcal{I}_n = \{0,1\}^n$;
- $\mathcal{O}_n^{\mathsf{rc}} = \mathcal{O}_n^{\mathsf{sc}} = \mathbb{N}$;
- For all $(s, v) \in \mathcal{S}_n \times \mathcal{I}_n$, $\mathsf{rc}_n(s, v) = v^s \ mod\ \mathtt{m}$; and
- For all $(s, v) \in \mathcal{S}_n \times \mathcal{I}_n$, $\mathsf{sc}_n(s, v) = time_{\mathtt{t}} \cdot num(s, 1) + time_{\mathtt{f}} \cdot num(s, 0)$.

[8] This is admittedly an optimistic assumption. Indeed, proposed timing attacks exploit the fact that the running time of the operation can depend on `y` [23, 30]. Here, we assume that the running time of the operation is made independent of `y` by some means (e.g., by adopting the constant-time implementation technique).

Here, $num(s, b) = |\{i \in \mathbb{N} \mid i < n \wedge s[i] = b\}|$ for $b \in \{0, 1\}$, and $time_{\mathtt{t}}$ (resp. $time_{\mathtt{f}}$) is the running time of the then (resp. else) branch.

Let the computation class of adversaries be the class of randomized polynomial time algorithms. Then, under the standard computational assumption that inverting modular exponentiation is hard, one can show that C satisfies $\mathsf{LISCNI}(f, \epsilon)$ for any f and negligible ϵ. This follows because the side-channel outputs are independent of low inputs, and the regular-channel is (f, ϵ)-secure for any f and negligible ϵ under the assumption.[9] Therefore, it can be made $(f, 1 - 1/k + \epsilon)$-secure for any f and negligible ϵ by applying bucketing. ▲

Remarks. We make some additional observations regarding the LISCNI condition. First, similar to condition (3) of SRSCR, the low-input independence condition of LISCNI (condition (2)) is a 2-safety property and is amenable to various verification methods proposed for the class of properties. In fact, because the condition is essentially side-channel non-interference but with respect to low inputs instead of high inputs, it can be checked by the methods for checking ordinary side-channel non-interference by reversing the roles of high inputs and low inputs [1, 20, 3, 6, 9].

Secondly, we note that the leaky login program from Example 1 does not satisfy LISCNI. This is because the program's side channel is not non-interferent with respect to low inputs. Indeed, given any secret $s \in \mathcal{S}_n$, one can vary the running times by choosing low inputs $v, v' \in \mathcal{I}_n$ with differing lengths of matching prefixes, that is, $(\mathrm{argmax}_i\, s\restriction_i = v\restriction_i) \neq (\mathrm{argmax}_i\, s\restriction_i = v'\restriction_i)$. Nevertheless, as we have shown in Examples 2 and 3, the program becomes secure once bucketing is applied. In fact, it becomes one that satisfies SRSCR as shown in Example 3. Ideally, we would like to find a relatively simple condition (on systems before bucketing is applied) that covers many systems that would become secure by applying bucketing. However, finding such a condition that covers a system like the leaky login program may be non-trivial. Indeed, predicting that the leaky login program become secure after applying bucketing appears to require more subtle analysis of interaction between low inputs and high inputs. (In fact, it can be shown that arbitrarily partitioning the side-channel outputs to a small number of buckets does not ensure security for this program.) Extending the technique to cover such scenarios is left for future work.

3.3 Combining Bucketing and Constant-Time Implementation Compositionally

We show that the LISCNI condition may be applied compositionally with the constant-time implementation technique (technically, we will only apply the condition (2) of LISCNI compositionally). As we shall show next, the combined approach is able to ensure security of some non-constant-time systems that cannot

[9] The latter holds because (f, ϵ)-security is asymptotic and the probability that any regular-channel adversary of the computation class may correctly guess the secret for this system is negligible (under the computational hardness assumption). Therefore, a similar analysis can be done for any sub-polynomial number of buckets.

```
while (sec > 0) {
  sec = sec - 1;
}
while (inp > 0) {
  inp = inp - 1;
}
return true;
```

Fig. 3. A non-constant-time program that cannot be made secure by globally applying bucketing.

be made sure by applying bucketing globally to the whole system. We remark that, in contrast to those of the previous sections of the paper, the results of this section are more specialized to the case of timing channels. First, we formalize the notion of constant-time implementation.

Definition 6 (Constant-Time). Let $f : \mathbb{N} \to \mathbb{N}$ and $\epsilon : \mathbb{N} \to (0, 1]$. We say that a system C satisfies the *constant-time* condition (or, *timing-channel non-interference*) with f and ϵ, written $\mathsf{CT}(f, \epsilon)$, if the following is satisfied:

(1) C is regular-channel (f, ϵ)-secure; and
(2) For all $n \in \mathbb{N}$, $v \in \mathcal{I}_n$, and $s_1, s_2 \in \mathcal{S}_n$, $\mathsf{sc}_n(s_1, v) = \mathsf{sc}_n(s_2, v)$.

Note that CT requires that the side channel is non-interferent (with respect to secrets). The following theorem is immediate from the definition, and states that CT is a sufficient condition for security.

Theorem 3 (CT Soundness). *If C satisfies $\mathsf{CT}(f, \epsilon)$, then C is (f, ϵ)-secure.*

To motivate the combined application of CT and LISCNI, let us consider the following example which is neither constant-time nor can be made secure by (globally) applying bucketing.

Example 5. Fig. 3 shows a simple, albeit contrived, program that we will use to motivate the combined approach. Here, `sec` is a n-bit secret and `inp` is a n-bit attacker-controlled input. Both `sec` and `inp` are interpreted as unsigned n-bit integers where $-$ and $>$ are the usual unsigned integer subtraction and comparison operations. The regular channel always outputs `true` and hence is non-interferent. Therefore, only the timing channel is of concern.

The program can be formalized as $C_{\texttt{comp}}$ where for all $n \in \mathbb{N}$,

- $\mathcal{S}_n = \mathcal{I}_n = \{0, 1\}^n$;
- $\mathcal{O}_n^{\mathsf{rc}} = \{\bullet\}$;
- $\mathcal{O}_n^{\mathsf{sc}} = \{i \in \mathbb{N} \mid i \leq 2^{n+1}\}$;
- For all $(s, v) \in \mathcal{S}_n \times \mathcal{I}_n$, $\mathsf{rc}_n(s, v) = \bullet$; and
- For all $(s, v) \in \mathcal{S}_n \times \mathcal{I}_n$, $\mathsf{sc}_n(s, v) = s + v$.

Note that the side channel outputs the sum of the high input and the low input. It is easy to see that the system is not constant-time (i.e., not $\mathsf{CT}(f, \epsilon)$ for any

f and ϵ). Furthermore, the system is not secure as is, because an adversary can immediately recover the secret by querying with any input and subtracting the input from the side-channel output.

Also, it is easy to see that the system does not satisfy $\mathsf{LISCNI}(f, \epsilon)$ for any f and ϵ either, because its side-channel outputs are not independent of low inputs. In fact, we can show that arbitrarily applying bucketing (globally) to the system does not guarantee security. To see this, let us consider applying bucketing with just two buckets whereby the buckets partition the possible running times in two halves so that running times less than or equal to 2^n fall into the first bucket and those greater than 2^n fall into the other bucket. After applying bucketing, the system is C' where

- $\mathsf{rc}\langle C' \rangle$, $\mathcal{S}\langle C' \rangle$, $\mathcal{I}\langle C' \rangle$, and $\mathcal{O}^{\mathsf{rc}}\langle C' \rangle$ are same as those of $C_{\mathtt{comp}}$;
- For all $n \in \mathbb{N}$, $\mathcal{O}^{\mathsf{sc}}\langle C' \rangle_n = \{0, 1\}$; and
- For all $n \in \mathbb{N}$ and $(s, v) \in \mathcal{S}_n \times \mathcal{I}_n$, $\mathsf{sc}\langle C' \rangle_n(s, v) = 0$ if $s + v \leq 2^n$, and $\mathsf{sc}\langle C' \rangle_n(s, v) = 1$ otherwise.

We show that there exists an efficient adaptive attack against C'. Let $s \in \mathcal{S}_n$. The adversary $\mathcal{A}$ recovers s by only making linearly many queries via the following process. First, $\mathcal{A}$ queries with the input $v_1 = 2^{n-1}$. By observing the side-channel output, $\mathcal{A}$ will know whether $0 \leq s \leq 2^{n-1}$ (i.e., the side-channel output was 0) or $2^{n-1} < s \leq 2^n$ (i.e., the side-channel output was 1). In the former case, $\mathcal{A}$ picks the input $v_2 = 2^{n-1} + 2^{n-2}$ for the next query, and in the latter case, he picks $v_2 = 2^{n-2}$. Continuing the process in a binary search manner and reducing the space of possible secrets by $1/2$ in each query, $\mathcal{A}$ is able to hone in on s within n many queries. Therefore, C' is not (n, ϵ)-secure for any ϵ. ▲

Next, we present the compositional bucketing approach. Roughly, our compositionality theorem (Theorem 4) states that the sequential composition of a constant-time system with a system whose side channel is non-interferent with respect to low inputs can be made secure by applying bucketing to only the non-constant-time component. As with LISCNI, the degree of security of the composed system is relative to the that of the regular channel and the granularity of buckets.

To state the compositionality theorem, we explicitly separate the conditions on side channels of CT and LISCNI from those on regular channels and introduce terminologies that only refer to the side-channel conditions. Let us fix C. We say that C satisfies $\mathsf{CT}^{\mathsf{sc}}$, if it satisfies condition (2) of CT, that is, for all $n \in \mathbb{N}$, $v \in \mathcal{I}_n$, and $s_1, s_2 \in \mathcal{S}_n$, $\mathsf{sc}_n(s_1, v) = \mathsf{sc}_n(s_2, v)$. Also, we say that C satisfies $\mathsf{LISCNI}^{\mathsf{sc}}$ if it satisfies condition (2) of LISCNI, that is, for all $n \in \mathbb{N}$, $s \in \mathcal{S}_n$, and $v_1, v_2 \in \mathcal{I}_n$, $\mathsf{sc}_n(s, v_1) = \mathsf{sc}_n(s, v_2)$. Next, we define sequential composition of systems.

Definition 7 (Sequential Composition). Let $C^\dagger$ and $C^\ddagger$ be systems such that $\mathcal{S}\langle C^\dagger \rangle = \mathcal{S}\langle C^\ddagger \rangle$, $\mathcal{I}\langle C^\dagger \rangle = \mathcal{I}\langle C^\ddagger \rangle$, and for all $n \in \mathbb{N}$, $\mathcal{O}^{\mathsf{sc}}\langle C^\ddagger \rangle_n \subseteq \mathbb{N}$ and $\mathcal{O}^{\mathsf{sc}}\langle C^\ddagger \rangle_n \subseteq \mathbb{N}$. The *sequential composition* of $C^\dagger$ with $C^\ddagger$, written $C^\dagger; C^\ddagger$, is the system C such that

– $\mathcal{S}\langle C\rangle = \mathcal{S}(C^\dagger)$ and $\mathcal{I}\langle C\rangle = \mathcal{I}(C^\dagger)$; and
– For all $n \in \mathbb{N}$ and $(s, v) \in \mathcal{S}_n \times \mathcal{I}_n$, $\mathsf{sc}\langle C'\rangle_n(s, v) = \mathsf{sc}\langle C^\dagger\rangle_n(s, v) + \mathsf{sc}\langle C^\ddagger\rangle_n(s, v)$.

We note that the definition of sequential composition specifically targets the case when the side channel is a timing channel, and says that the side-channels outputs are numeric values and that the side-channel output of the composed system is the sum of those of the components. Also, the definition leaves the composition of regular channels open, and allows the regular channel of the composed system to be any function from $\mathcal{S}_n \times \mathcal{I}_n$. We are now ready to state the compositionality theorem.

Theorem 4 (Compositionality). *Let $C^\dagger$ be a system that satisfies $\mathsf{LISCNI}^{\mathsf{sc}}$ and $C^\ddagger$ be a system that satisfies $\mathsf{CT}^{\mathsf{sc}}$. Suppose that $Bkt_k(C^\dagger); C^\ddagger$ is regular-channel (f, ϵ)-secure where $k \cdot \epsilon \leq 1$. Then, $Bkt_k(C^\dagger); C^\ddagger$ is (f, ϵ')-secure, where $\epsilon' = 1 - 1/k + \epsilon$.*

Proof. By Theorem 1, it suffices to show that $Bkt_k(C^\dagger); C^\ddagger$ satisfies $\mathsf{SRSCR}(f, k \cdot \epsilon, 1/k)$. By an argument similar to the proof of Theorem 2, there must be an indexed family of sets of secrets S' such that, for all $n \in \mathbb{N}$, (a) $S'_n \subseteq \mathcal{S}_n$, (b) $|S'_n| \geq |\mathcal{S}_n|/k$, and (c) for all $s_1, s_2 \in S'_n$ and $v_1, v_2 \in \mathcal{I}_n$, $\mathsf{sc}\langle Bkt_k(C^\dagger)\rangle_n(s_1, v_1) = \mathsf{sc}\langle Bkt_k(C^\dagger)\rangle_n(s_2, v_2)$. We define an indexed family of sets of sets of secrets S^{res} to be such that S^{res}_n is the singleton set $\{S'_n\}$ for each n.

We show that $C = Bkt_k(C^\dagger); C^\ddagger$ satisfies conditions (1), (2), and (3) of $\mathsf{SRSCR}(f, k \cdot \epsilon, 1/k)$ with the restricted secret subsets S^{res} defined above. Firstly, (1) is satisfied because $|S'_n| \geq |\mathcal{S}_n|/k$. Also, because $Bkt_k(C^\dagger); C^\ddagger$ is regular-channel (f, ϵ)-secure, we can show that (2) is satisfied by an argument similar to the one in the proof of Theorem 2.

It remains to show that (3) is satisfied. It suffices to show that for all $n \in \mathbb{N}$, $v \in \mathcal{I}_n$, and $s_1, s_2 \in S'_n$, $\mathsf{sc}\langle C\rangle_n(s_1, v) = \mathsf{sc}\langle C\rangle_n(s_2, v)$. That is, the side channel of the composed system is non-interferent (with respect to high inputs) for the subset S'. By the definition of the sequential composition, for all $v \in \mathcal{I}_n$ and $s \in \mathcal{S}_n$, $\mathsf{sc}\langle C\rangle_n(s, v) = \mathsf{sc}\langle Bkt_k(C^\dagger)\rangle_n(s, v) + \mathsf{sc}\langle C^\ddagger\rangle_n(s, v)$. Therefore, for all $v \in \mathcal{I}_n$ and $s_1, s_2 \in S'_n$,

$$\begin{aligned}
\mathsf{sc}\langle C\rangle_n(s_1, v) &= \mathsf{sc}\langle Bkt_k(C^\dagger)\rangle_n(s_1, v) + \mathsf{sc}\langle C^\ddagger\rangle_n(s_1, v) \\
&= \mathsf{sc}\langle Bkt_k(C^\dagger)\rangle_n(s_2, v) + \mathsf{sc}(C^\ddagger)_n(s_2, v) \\
&= \mathsf{sc}\langle C\rangle_n(s_2, v)
\end{aligned}$$

because $\mathsf{sc}\langle C^\ddagger\rangle_n(s_1, v) = \mathsf{sc}\langle C^\ddagger\rangle_n(s_2, v)$ by $\mathsf{CT}^{\mathsf{sc}}$ of $C^\ddagger$, and $\mathsf{sc}\langle Bkt_k(C^\dagger)\rangle_n(s_1, v) = \mathsf{sc}\langle Bkt_k(C^\dagger)\rangle_n(s_2, v)$ by (c) above. □

We note that the notion of sequential composition is symmetric. Therefore, Theorem 4 implies that the composing the components in the reverse order, that is, $C^\ddagger; Bkt_k(C^\dagger)$, is also secure provided that its regular channel is secure.

The compositionality theorem suggests the following compositional approach to ensuring security. Given a system C that is a sequential composition of a component whose side channel outputs are independent of high inputs (i.e., satisfies $\mathsf{CT}^{\mathsf{sc}}$) and a component whose side channel outputs are independent of

low inputs (i.e., satisfies $\mathsf{LISCNI}^{\mathsf{sc}}$), we can ensure the security of C by proving its regular-channel security and applying bucketing only to the non-constant-time component.

Example 6. Let us apply compositional bucketing to the system $C_{\mathtt{comp}}$ from Example 5. Recall that the system is neither constant-time nor applying bucketing to the whole system ensures its security. The system can be seen as the sequential composition $C_{\mathtt{comp}} = C^{\dagger}; C^{\ddagger}$ where $C^{\dagger}$ and $C^{\ddagger}$ satisfy the following:

- $\mathcal{S}$ and $\mathcal{I}$ are as in $C_{\mathtt{comp}}$;
- For all $n \in \mathbb{N}$, $\mathcal{O}^{\mathsf{sc}}\langle C^{\dagger}\rangle_n = \mathcal{O}^{\mathsf{sc}}\langle C^{\ddagger}\rangle_n = \{i \in \mathbb{N} \mid i \leq 2^n\}$; and
- For all $n \in \mathbb{N}$ and $(s, v) \in \mathcal{S}_n \times \mathcal{I}_n$, $\mathsf{sc}\langle C^{\dagger}\rangle_n(s, v) = s$ and $\mathsf{sc}\langle C^{\ddagger}\rangle_n(s, v) = v$.

Note that $C^{\ddagger}$ satisfies $\mathsf{CT}^{\mathsf{sc}}$ as its side-channel outputs are high-input independent, and, $C^{\dagger}$ satisfies $\mathsf{LISCNI}^{\mathsf{sc}}$ as its side-channel outputs are low-input independent. By applying bucketing only to the component $C^{\dagger}$, we obtain the system $Bkt_k(C^{\dagger}); C^{\ddagger}$. The regular-channel of $Bkt_k(C^{\dagger}); C^{\ddagger}$ (i.e., that of $C_{\mathtt{comp}}$) is (f, ϵ)-secure for any f and negligible ϵ because it is non-interferent (with respect to high inputs) and the probability that an adversary may recover a secret for such a system is at most $1/|\mathcal{S}_n|$.[10] Therefore, by Theorem 4, $Bkt_k(C^{\dagger}); C^{\ddagger}$ is $(f, 1 - 1/k + \epsilon)$-secure for any f and negligible ϵ. ▲

The above example shows that compositional bucketing can be used to ensure security of non-constant-time systems that cannot be made secure by a whole-system bucketing. It is interesting to observe that the constant-time condition, $\mathsf{CT}^{\mathsf{sc}}$, requires the side-channel outputs to be independent of high inputs but allows dependency on low inputs, while $\mathsf{LISCNI}^{\mathsf{sc}}$ is the dual and says that the side-channel outputs are independent of low inputs but may depend on high inputs. Our compositionality theorem (Theorem 4) states that a system consisting of such parts can be made secure by applying bucketing only to the part that satisfies the latter condition.

It is easy to see that sequentially composing components that satisfy $\mathsf{CT}^{\mathsf{sc}}$ results in a system that satisfies $\mathsf{CT}^{\mathsf{sc}}$, and likewise, sequentially composing components that satisfy $\mathsf{LISCNI}^{\mathsf{sc}}$ results in a system that satisfies $\mathsf{LISCNI}^{\mathsf{sc}}$. Therefore, such compositions can be used freely in conjunction with the compositional bucketing technique of this section. We also conjecture that components that are made secure by compositional bucketing can themselves be sequentially composed to form a secure system (possibly with some decrease in the degree of security). We leave a more detailed investigation for future work.

4 Related Work

As remarked in Section 1, much research has been done on defending against timing attacks and more generally side channel attacks. For instance, there have

[10] Therefore, a similar analysis can be done for any strictly sub-exponential number of buckets.

been experimental evaluation on the effectiveness of bucketing and other timing-channel mitigation schemes [18, 14], and other works have proposed information-theoretic methods for formally analyzing the security of (deterministic and probabilistic) systems against adaptive adversaries [25, 12].

However, few prior works have formally analyzed the effect of bucketing on timing channel security (or similar techniques for other side channels) against adaptive adversaries. Indeed, to our knowledge, the only prior work to do so are the series of works by Köpf et al. [26, 27] who investigated the effect of bucketing applied to blinded cryptography algorithms. They show that applying bucketing to a blinded cryptography algorithm whose regular channel is IND-CCA2 secure results in an algorithm that is IND-CCA2 secure against timing-channel-observing adversaries. In addition, they show bounds on information leaked by such bucketed blinded cryptography algorithms in terms of quantitative information flow [28, 33, 5, 39, 40]. By contrast, we analyze the effect of applying bucketing to general systems, show that bucketing is in general insufficient against adaptive adversaries, and present novel conditions that guarantee security against such adversaries. Also, our results are given in the form of (f, ϵ)-security, which can provide precise bounds on the number of queries needed by adaptive adversaries to recover secrets.

Next, we compare our work with the works on constant-time implementations (i.e., timing-channel non-interference) [1, 22, 20, 3, 6, 9]. The previous works have proposed methods for verifying that the given system is constant-time [20, 3, 6, 9] or transforming it to one that is constant-time [1, 22]. As we have also discussed in this paper (cf. Theorem 3), it is easy to see that the constant-time condition directly transfers the regular-channel-only security to the security for the case with timing channels. By contrast, security implied by bucketing is less straightforward. In this paper, we have shown that bucketing is in general insufficient to guarantee the security of systems even when their regular channel is perfectly secure. And, we have presented results that show that, under certain conditions, the regular-channel-only security can be transferred to the side-channel-observing case to certain degrees. Because there are advantages of bucketing such as efficiency and ease of implementation [26, 27, 7, 41, 14], we hope that our results will contribute to a better understanding of the bucketing technique and foster further research on the topic.

5 Conclusion and Future Work

In this paper, we have presented a formal analysis of the effectiveness of the bucketing technique against adaptive timing-channel-observing adversaries. We have shown that bucketing is in general insufficient against such adversaries, and presented two novel conditions, SRSCR and LISCNI, that guarantee security against such adversaries. SRSCR states that a system that satisfies it is secure, whereas LISCNI states that the a system that satisfies it becomes secure when bucketing is applied. We have shown that both conditions facilitate proving the security of systems against adaptive side-channel-observing adver-

saries by allowing a system designer to prove the security of the system's regular channel separately from the concerns of its side-channel behavior. By doing so, the security of the regular-channel is transferred, to certain degrees, to the full side-channel-aware security. We have also shown that the LISCNI condition can be used in conjunction with the constant-time implementation technique in a compositional manner to further increase its applicability. We have formalized our results via the notion of (f, ϵ)-security, which gives precise bounds on the number of queries needed by adaptive adversaries to recover secrets.

While we have instantiated our results to timing channel and bucketing, many of the results are actually quite general and are applicable to side channels other than timing channels. Specifically, aside from the compositional bucketing result that exploits the "additive" nature of timing channels, the results are applicable to any side channels and techniques that reduce the number of possible side-channel observations.

As future work, we would like to extend our results to probabilistic systems. Currently, our results are limited to deterministic systems, and such an extension would be needed to assess the effect of bucketing when it is used together with countermeasure techniques that involve randomization. We would also like to improve the conditions and the security bounds thereof to be able to better analyze systems such as the leaky login program shown in Examples 1, 2 and 3. Finally, we would like to extend the applicability of the compositional bucketing technique by considering more patterns of compositions, such as sequentially composing components that themselves have been made secure by compositional bucketing.

Acknowledgements. We thank the anonymous reviewers for useful comments. This work was supported by JSPS KAKENHI Grant Numbers 17H01720 and 18K19787, JSPS Core-to-Core Program, A.Advanced Research Networks, JSPS Bilateral Collaboration Research, and Office of Naval Research (ONR) award #N00014-17-1-2787.

References

1. J. Agat. Transforming out timing leaks. In *POPL*, 2000.
2. A. Aguirre, G. Barthe, M. Gaboardi, D. Garg, and P. Strub. A relational logic for higher-order programs. *PACMPL*, 1(ICFP), 2017.
3. J. B. Almeida, M. Barbosa, G. Barthe, F. Dupressoir, and M. Emmi. Verifying constant-time implementations. In *USENIX Security Symposium*, 2016.
4. J. B. Almeida, M. Barbosa, J. S. Pinto, and B. Vieira. Formal verification of side-channel countermeasures using self-composition. *Science of Compututer Programming*, 78(7), 2013.
5. M. S. Alvim, K. Chatzikokolakis, C. Palamidessi, and G. Smith. Measuring information leakage using generalized gain functions. In *CSF*, 2012.
6. 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.

7. A. Askarov, D. Zhang, and A. C. Myers. Predictive black-box mitigation of timing channels. In *CCS*, 2010.
8. G. Barthe, P. R. D'Argenio, and T. Rezk. Secure information flow by self-composition. *Mathematical Structures in Computer Science*, 21(6), 2011.
9. G. Barthe, B. Grégoire, and V. Laporte. Secure compilation of side-channel countermeasures: The case of cryptographic "constant-time". In *CSF*, 2018.
10. N. Benton. Simple relational correctness proofs for static analyses and program transformations. In *POPL*, 2004.
11. A. Blot, M. Yamamoto, and T. Terauchi. Compositional synthesis of leakage resilient programs. In *POST*, 2017.
12. M. Boreale and F. Pampaloni. Quantitative information flow under generic leakage functions and adaptive adversaries. *Logical Methods in Computer Science*, 11(4), 2015.
13. M. R. Clarkson and F. B. Schneider. Hyperproperties. *Journal of Computer Security*, 18(6), 2010.
14. Y. G. Dantas, R. Gay, T. Hamann, H. Mantel, and J. Schickel. An evaluation of bucketing in systems with non-deterministic timing behavior. In *ICT Systems Security and Privacy Protection*, 2018.
15. 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.
16. H. Eldib and C. Wang. Synthesis of masking countermeasures against side channel attacks. In *CAV*, 2014.
17. K. Gandolfi, C. Mourtel, and F. Olivier. Electromagnetic analysis: Concrete results. In *CHES*, 2001.
18. R. Gay, H. Mantel, and H. Sudbrock. An empirical bandwidth analysis of interrupt-related covert channels. *IJSSE*, 6(2), 2015.
19. J. A. Goguen and J. Meseguer. Security policies and security models. In *IEEE Symposium on Security and Privacy*, 1982.
20. D. Hedin and D. Sands. Timing aware information flow security for a JavaCard-like bytecode. 2005.
21. Y. Ishai, A. Sahai, and D. A. Wagner. Private circuits: Securing hardware against probing attacks. In *CRYPTO*, 2003.
22. N. Kobayashi and K. Shirane. Type-based information analysis for low-level languages. In *APLAS*, 2002.
23. P. C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In *CRYPTO*, 1996.
24. P. C. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In *CRYPTO*, 1999.
25. B. Köpf and D. A. Basin. Automatically deriving information-theoretic bounds for adaptive side-channel attacks. *Journal of Computer Security*, 19(1), 2011.
26. B. Köpf and M. Dürmuth. A provably secure and efficient countermeasure against timing attacks. In *CSF*, 2009.
27. B. Köpf and G. Smith. Vulnerability bounds and leakage resilience of blinded cryptography under timing attacks. In *CSF*, 2010.
28. P. Malacaria. Assessing security threats of looping constructs. In *POPL*, 2007.
29. P. Malacaria. Algebraic foundations for quantitative information flow. *Mathematical Structures in Computer Science*, 25(2), 2015.
30. C. S. Pasareanu, Q. Phan, and P. Malacaria. Multi-run side-channel analysis using symbolic execution and max-SMT. In *CSF*, 2016.

31. J. Quisquater and D. Samyde. Electromagnetic analysis (EMA): measures and counter-measures for smart cards. In *Smart Card Programming and Security, International Conference on Research in Smart Cards, E-Smart*, 2001.
32. J. C. Reynolds. *The Craft of Programming*. Prentice Hall International series in computer science. Prentice Hall, 1981.
33. G. Smith. On the foundations of quantitative information flow. In *FOSSACS*, 2009.
34. M. Sousa and I. Dillig. Cartesian Hoare logic for verifying k-safety properties. In *PLDI*, 2016.
35. T. Terauchi and A. Aiken. Secure information flow as a safety problem. In *SAS*, 2005.
36. E. Tromer, D. A. Osvik, and A. Shamir. Efficient cache attacks on AES, and countermeasures. *Journal of Cryptology*, 23(1), 2010.
37. 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.
38. H. Yasuoka and T. Terauchi. Quantitative information flow - verification hardness and possibilities. In *CSF*, 2010.
39. H. Yasuoka and T. Terauchi. On bounding problems of quantitative information flow. *Journal of Computer Security*, 19(6), 2011.
40. H. Yasuoka and T. Terauchi. Quantitative information flow as safety and liveness hyperproperties. *Theoretical Compututer Science*, 538, 2014.
41. D. Zhang, A. Askarov, and A. C. Myers. Language-based control and mitigation of timing channels. In *PLDI*, 2012.