Journal of Computer Security 28 (2020) 607–634
DOI 10.3233/JCS-191356
IOS Press

Bucketing and information flow analysis for provable timing attack mitigation

Tachio Terauchi [a,*] and Timos Antonopoulos [b]
[a] *Department of Computer Science and Engineering, Waseda University, Tokyo, Japan*
E-mail: terauchi@waseda.jp
[b] *Department of Computer Science, Yale University, New Haven, CT, U.S.A.*
E-mail: 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. It is interesting to note that our results make non-trivial (and somewhat unconventional) uses of ideas from information flow research such as *channel capacity* and *refinement order relation*.

Keywords: Side-channel attacks, timing attacks, bucketing, information flow

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 [13,18,20,21,25,27–29,36,46]. 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 [1,3,4,6,8,11,22,24,26,30,31,35,51].

Bucketing is a technique proposed for mitigating timing attacks [8,16,30,31,51]. It restricts the system's outputs to only occur at pre-determined and input-independent (but possibly non-periodic and security-parameter-dependent) 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 the system's

*Corresponding author. E-mail: terauchi@waseda.jp.

performance because outputs must be delayed to the next bucket time. Nonetheless, in comparison to the *constant-time implementation* technique [1,3,6,11,24,26] 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 [30,31].[1] For example, bucketing may be implemented in a blackbox-style by a monitor that buffers and delays outputs [8,51]. On the other hand, while it is clear that constant-time implementation entirely removes information leaks from timing channels, bucketing only mitigates the leaks and the precise degree of security achieved by the mitigation has been little understood. *A main motivation of the paper is to clarify the degree of security achieved by bucketing.*

In this paper, we formally study the effect of bucketing on security against *adaptive* timing attacks. To this end, first, we review a formal notion of security against adaptive side-channel-observing adversaries, called (f, ϵ)-*security*, which was introduced in our previous work [7,43] inspired by a closely related notion from the DARPA STAC program [17]. 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 disjoint subsets of secrets such that, when the space of possible secrets is restricted to each subset, (1) the system's side channel reveals no more information than the regular channel and (2) the system is secure against adversaries who only observe the regular channel. The degree of security (i.e., f and ϵ) is proportional to those against regular-channel-only-observing adversaries for the restricted subset of secrets and the sizes 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 *bounded low-input side-channel capacity* (BLISCC). We show that applying bucketing to a system that satisfies the condition would result in a system that satisfies SRSCR. Therefore, BLISCC is a sufficient condition for security under the bucketing technique. Roughly, BLISCC says that (1) the attacker-controlled inputs only have some bounded influence on the side channel (while secrets are allowed to influence the side channel arbitrarily), 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, the degree of attacker-controlled-input influence on the side channel, and the granularity of buckets. A main benefit of the conditions SRSCR and BLISCC 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.

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 is BLISCC, 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.

[1] Sometimes, the terminology "constant-time implementation" is used to mean even stricter requirements, such as requiring control flows to be secret independent [3,11]. In this paper, we use the terminology for a more permissive notion in which only the running time is required to be secret independent.

- 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 BLISCC 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).

As remarked before, these results are given in terms of a formal notion of security called (f, ϵ)-security that we will review in Section 2.

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.

Our results make use of ideas from (quantitative) information flow research such as refinement order relation [33,34,38,48] and channel capacity [5,9,32,39,48,49]. It is interesting to note that we use the ideas somewhat unconventionally. Rather than using them directly to, for example, derive the degree of security in terms of information theoretic measures such as Shannon entropy, we use them *indirectly*, for example, to bound the degree of influence that attacker-controlled inputs have on side-channel outputs. We show that, combined with certain additional conditions and program transformations, this allows us to derive the security of the final system.

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 BLISCC. 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 BLISCC 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.

A preliminary version of this paper appeared in [43]. The present paper substantially improves that version by extending the key results. Namely, the conditions SRSCR and LISCNI and their compositional application have been substantially extended. The extensions allow derivation of much better security bounds. Accordingly, the theorems stating the derivable security bounds and the proofs of the theorems have also been renovated. We list the main changes below:

(1) SRSCR is extended to allow different secret subsets to be assigned different regular-channel adversary error probability bounds. The extension allows the derivation of much better security bounds. SRSCR in [43] is a restricted case of the new one where the same bound was assigned to every subset. We also correct a bug in [43] that allowed non-disjoint secret subsets (Definition 3.3, Lemma 3.2 and Theorem 3.4 in Section 3.1).

(2) The condition called low-input side-channel non-interference (LISCNI) in [43] is extended to what we now call *bounded low-input side-channel capacity* (BLISCC) condition. BLISCC is a generalization of LISCNI in that the latter is a special case of the former where the capacity bound is restricted to 1 (Definition 3.8 in Section 3.2).

(3) The extensions (1) and (2) together improve the BLISCC (LISCNI in [43]) soundness theorem and the compositionality theorem to allow derivation of tighter security bounds (Lemma 3.10, Theorem 3.11 and Corollary 3.12 in Section 3.2, and Theorem 3.21 and Corollary 3.22 in Section 3.3).
(4) An additional example demonstrating the usefulness of the BLISCC condition is added (Example 3.14).
(5) An extended analysis of the leaky login program is added. The analysis shows a limitation with the permissive definition of bucketing that is used in this paper (and in [43]) (Example 3.15 in Section 3.2).

In summary, the main technical contributions given in Sections 3.1, 3.2 and 3.3 (same section numbers are used in [43]) have all been substantially extended to allow improved security analysis. Also, as a side effect of the extended results, we were able to "fix" the bugs in the examples of [43] where we claimed security bounds that cannot be derived by the results in that version (the authors' online copy of [43] corrects the bug by stating weaker bounds [44]). The extensions in this submission allow the derivation of the strong bounds.

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. Furthermore, $\mathcal{S}$ 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. We assume that each $\mathcal{S}_n$ and $\mathcal{I}_n$ are finite. 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.[2] 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 the parameter C 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* [29].

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.

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

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 $Win^C_{\mathcal{A}}(n, f)$ to be the event that the following game outputs true.

$s \leftarrow \mathcal{S}_n$

$s' \leftarrow \mathcal{A}^{C_n(s)}(f(n))$

Output $s = s'$

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 $Win^C_{\mathcal{A}}(n, f)$. Now, we are ready to give the definition of (f, ϵ)-security.

Definition 2.1 ((f, ϵ)-security)**.** Let $f : \mathbb{N} \to \mathbb{N}$ and $\epsilon : \mathbb{N} \to \mathbb{R}$ be such that $0 < \epsilon(n) \leqslant 1$ for all $n \in \mathbb{N}$. We say that a system C is (f, ϵ)-*secure* if there exists $N \in \mathbb{N}$ such that for all adversaries $\mathcal{A}$ and $n \geqslant N$, it holds that $\Pr[Win^C_{\mathcal{A}}(n, f)] < \epsilon(n)$.

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 $\epsilon(n)$, assuming that the secrets are uniformly distributed. It should be noted that this definition of security is also used in our previous works [7,43] and it also corresponds closely to the definition used in the DARPA STAC program [17].[3]

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 $\epsilon \leqslant \epsilon'$ when $\epsilon(n) \leqslant \epsilon'(n)$ for all sufficiently large n, and likewise for $\epsilon < \epsilon'$.

Example 2.2 (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.

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

Fig. 1. Timing insecure login program.

[3] A slight difference is that the definition in [7] is not asymptotic, i.e., it asserts $\Pr[Win^C_{\mathcal{A}}(n, f)]$ for all n. The relaxed definition in this paper facilitates derivations of asymptotic bounds.

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

- $\mathcal{S}_n = \mathcal{I}_n = \{0,1\}^n$;
- $\mathcal{O}^{\mathsf{rc}}_n = \{\texttt{true}, \texttt{false}\}$ and $\mathcal{O}^{\mathsf{sc}}_n = \{i \in \mathbb{N} \mid i \leqslant n\}$;
- 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 2.3 (Bucketed leaky login)**.** Next, we consider the security of the program from Example 2.2 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 natural $i \leqslant k$.[4] 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 2.2;
- For all $n \in \mathbb{N}$, $\mathcal{O}^{\mathsf{sc}}_n = \{i \in \mathbb{N} \mid i \leqslant k\}$; and
- For all $n \in \mathbb{N}$ and $(s, v) \in \mathcal{S}_n \times \mathcal{I}_n$, $\mathsf{sc}_n(s, v) = bkt(\mathrm{argmax}_i\, s\restriction_i = v\restriction_i, n/k)$

where $bkt(i, j)$ is the smallest $a \in \mathbb{N}$ such that $i \leqslant a \cdot j$. It is easy to see that the system is not constant-time for any $k > 1$. Nonetheless, with the analysis method that we will present in Section 3.1, 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 $1 \leqslant N < 2^{n/k}$ (cf. Example 3.5). 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.[5]

Remark 2.4 (Parameter correlation in (f, ϵ)-security)**.** Note that in Example 2.3, the parameters f and ϵ in (f, ϵ)-security are correlated. For instance, when ϵ is maximized (i.e., letting $\epsilon = 1$ by letting $N = 1$), the corresponding f is also maximized to be $2^{n/k} - 2$. Typically, a larger ϵ implies a larger f and vice versa. Such a correlation is expected as requiring the attacker to succeed with a higher probability by making ϵ larger means that the attacker has to work harder by making more queries which implies enlarging f.

As we shall see later in the paper, the main results of the paper (cf. Theorem 3.4, Theorem 3.11, Corollary 3.12 and Theorem 3.21) show, given a system with or without bucketing, how to transfer the *regular-channel security* of the system in which the adversary only observes the regular channel (cf. Section 3.1) to that for the case when the adversary observes both the regular and the side channel. Some amount of security is lost in the transfer and each theorem stipulates how much security is lost

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

[5] Roughly, the bound for the ideal login program follows from the fact that the probability of guessing an element in a set of size M in X tries is $X/M + (1 - X/M)(1/(M - X))$ and letting $X = 2^n - N$ and $M = 2^n$. We refer to [7] for a method for formally deriving such a bound.

in terms of the change in the f and ϵ parameters. As we shall show, the statements of the theorems say that only the ϵ parameter is affected, possibly giving a false impression that bucketing only affects the probability of the attack success. However, due to the parameter correlation, the results actually imply that the change in the ϵ parameter may need to be compensated by a change in the f parameter, that is, the transfer also affects the number of queries the adversary needs to make to recover the secret.

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 [23,47]) 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 $\mathcal{I}_n = \{i \in \mathbb{N} \mid i \leqslant n\}$;
- $\mathcal{O}_n^{\mathsf{rc}} = \{\bullet\}$ and $\mathcal{O}_n^{\mathsf{sc}} = \{0, 1\}$;
- 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 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 as shown in the program below.

```
if (s[v] == 0) {
  sleep(100);
}
return true;
```

We remark that the above attack is *efficient*. That is, the attacker recovers the secret with a high probability with only a small amount of queries. This is in contrast to the bucketed leaky login program from Example 2.3 which, while still leaking information through the side channel, is secure in the sense that any attack that recovers the secret with a high probability must make an exponential number of queries. Secondly, we remark that leaking information does not mean that attacker can always recover the secret with a high probability, even with an unlimited number of queries. For instance, if a line `v = 0;` is added at the beginning of the program above, the attacker would have little chance of recovering the secret with any number of queries, even though the program still leaks information for any query (namely, the first bit of s). Finally, 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.

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 BLISCC 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 BLISCC 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 disjoint 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 just observing the regular channel, and that the side channel reveals no more information than the regular channel for those sets of secrets. Then, because each S'' is large, the entire system is also ensured to be secure with high probability. We adopt *refinement order relation* [33,34,38,48], which had been studied in quantitative information flow (QIF) 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 begin by introducing preliminary notions that we shall use to state the SRSCR condition.

Regular-channel security. 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 say that the system 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.

Secret restriction. Let us write $(0, 1]$ for the set $\{a \in \mathbb{R} \mid 0 < a \leqslant 1\}$. Let us fix a system $C = (\mathsf{rc}, \mathsf{sc}, \mathcal{S}, \mathcal{I}, \mathcal{O}^{\mathsf{rc}}, \mathcal{O}^{\mathsf{sc}})$. We call X a *secret restriction* of $\mathcal{S}$ when (1) X is an indexed family such that $X_n \subseteq \mathcal{P}(\mathcal{S}_n) \times (0, 1]$ for each $n \in \mathbb{N}$ and (2) for each $n \in \mathbb{N}$, $(S_1, p_1) \in X_n$ and $(S_2, p_2) \in X_n$, $(S_1, p_1) \neq (S_2, p_2)$ implies $S_1 \cap S_2 = \emptyset$. Let us write $\#_1(S, p)$ for S and $\#_2(S, p)$ for p. For a secret restriction X, we write $U \prec X$ when U is an indexed family such that $U_n \in X_n$ for each n. Note that such U satisfies $U_n \in \mathcal{P}(\mathcal{S}_n) \times (0, 1]$ for each n. For an indexed family of sets of secrets T such that $T_n \subseteq \mathcal{S}_n$ for each $n \in \mathbb{N}$, we write $C|_T$ for the system $(\mathsf{rc}, \mathsf{sc}, T, \mathcal{I}, \mathcal{O}^{\mathsf{rc}}, \mathcal{O}^{\mathsf{sc}})$. We write $C|_U$ for $C|_{\#_1 U}$, and write err_U for the function from $\mathbb{N}$ to $(0, 1]$ such that $err_U(n) = (\#_2 U)_n$ for each n. Here, $(\#_1 U)_n = \#_1(U_n)$ and $(\#_2 U)_n = \#_2(U_n)$ for each n. Roughly, err_U is the error bound function stipulated by U.

Refinement order relation. Let us fix a system $C = (\mathsf{rc}, \mathsf{sc}, \mathcal{S}, \mathcal{I}, \mathcal{O}^{\mathsf{rc}}, \mathcal{O}^{\mathsf{sc}})$. Let T be an indexed family of sets of secrets such that $T_n \subseteq \mathcal{S}_n$ for each $n \in \mathbb{N}$. We say that C satisfies the *side-channel refinement order relation condition* on the secrets T, written $RR(T)$, if for all $n \in \mathbb{N}$, $s_1 \in T_n$, $s_2 \in T_n$ and $v \in \mathcal{I}_n$, it holds that $\mathsf{sc}_n(s_1, v) \neq \mathsf{sc}_n(s_2, v) \Rightarrow \mathsf{rc}_n(s_1, v) \neq \mathsf{rc}_n(s_2, v)$. Roughly, $RR(T)$ says that the system's side channel reveals no more information than its regular-channel on the secrets T. We formalize the intuition in Lemma 3.2.

Lemma 3.1. *Let T be an indexed family of sets of secrets such that $T_n \subseteq \mathcal{S}_n$ for each $n \in \mathbb{N}$. Suppose that C satisfies $RR(T)$. Then, for any $n \in \mathbb{N}$, $o^{\mathsf{rc}} \in \mathcal{O}_n^{\mathsf{rc}}$ and $v \in \mathcal{I}_n$, there exists $o^{\mathsf{sc}} \in \mathcal{O}_n^{\mathsf{sc}}$ such that for any $s \in T_n$, $\mathsf{rc}_n(s, v) = o^{\mathsf{rc}} \Rightarrow \mathsf{sc}_n(s, v) = o^{\mathsf{sc}}$.*

Proof. Let $o^{\mathsf{rc}} \in \mathcal{O}_n^{\mathsf{rc}}$, $v \in \mathcal{I}_n$ and $s \in T_n$ be such that $\mathsf{rc}_n(s, v) = o^{\mathsf{rc}}$. Let $o^{\mathsf{sc}} = \mathsf{sc}_n(s, v)$. Then, by $RR(T)$, for any $s' \in T_n$ such that $\mathsf{rc}_n(s', v) = o^{\mathsf{rc}}$, we have $\mathsf{sc}_n(s', v) = o^{\mathsf{sc}}$. □

Lemma 3.2. *Let T be an indexed family of sets of secrets such that $T_n \subseteq \mathcal{S}_n$ for each $n \in \mathbb{N}$. Suppose that C satisfies $RR(T)$ and $C|_T$ is regular-channel (f, ϵ)-secure. Then, $C|_T$ is (f, ϵ)-secure.*

Proof. Suppose for contradiction that $C|_T$ is not (f, ϵ)-secure. Then, there exists $\mathcal{A}$ such that $Win_{\mathcal{A}}^{C|_T}(n, f) > \epsilon(n)$ for infinitely many $n \in \mathbb{N}$. We prove the lemma by constructing a (regular-channel) adversary $\mathcal{A}'$ such that $\Pr[Win_{\mathcal{A}'}^{C|_T^\bullet}(n, f)] > \epsilon(n)$ for infinitely many $n \in \mathbb{N}$.

We construct such $\mathcal{A}'$ by letting it *simulate* $\mathcal{A}$. Fix $n \in \mathbb{N}$. By $RR(T)$ and Lemma 3.1, for any $o^{\mathsf{rc}} \in \mathcal{O}_n^{\mathsf{rc}}$ and $v \in \mathcal{I}_n$, there exists $o^{\mathsf{sc}} \in \mathcal{O}_n^{\mathsf{sc}}$ such that for any $s \in T_n$, $\mathsf{rc}_n(s, v) = o^{\mathsf{rc}} \Rightarrow \mathsf{sc}_n(s, v) = o^{\mathsf{sc}}$. Let $cor_n : \mathcal{O}_n^{\mathsf{rc}} \times \mathcal{I}_n \to \mathcal{O}_n^{\mathsf{sc}}$ be the mapping where $cor_n(o^{\mathsf{rc}}, v)$ is such o^{sc}, for each $o^{\mathsf{rc}} \in \mathcal{O}_n^{\mathsf{rc}}$ and $v \in \mathcal{I}_n$. Then, $\mathcal{A}'$ works by running $\mathcal{A}$ but hijacking its process whenever it makes a query to the system. That is, whenever $\mathcal{A}$ attempts to query the system with an input $v \in \mathcal{I}_n$, $\mathcal{A}'$ sends the query to the regular-channel-only system to obtain the regular-channel output $o^{\mathsf{rc}} = \mathsf{rc}(s, v)$ and communicates back to $\mathcal{A}$ the regular-and-side-channel output $(o^{\mathsf{rc}}, cor_n(o^{\mathsf{rc}}, v))$. It is easy to see that $\mathcal{A}'$ and $\mathcal{A}$ behave in exactly the same manner on secrets from T_n. Therefore, $\Pr Win_{\mathcal{A}'}^{C|_T^\bullet}(n, f)] > \epsilon(n)$ for infinitely many $n \in \mathbb{N}$. □

We are now ready to formally state the SRSCR condition.

Definition 3.3 (SRSCR)**.** Let $C = (\mathsf{rc}, \mathsf{sc}, \mathcal{S}, \mathcal{I}, \mathcal{O}^{\mathsf{rc}}, \mathcal{O}^{\mathsf{sc}})$. Let $f : \mathbb{N} \to \mathbb{N}$ and X be a secret restriction of $\mathcal{S}$. We say that C satisfies the *secret-restricted side-channel refinement* condition with f and X written $\mathsf{SRSCR}(f, X)$, if the following condition holds

(1) For all $U \prec X$, $C|_U$ is regular-channel (f, err_U)-secure; and
(2) For all $U \prec X$, $RR(\#_1 U)$ holds.[6]

We informally describe why SRSCR is a sufficient condition for security. Let $(S, p) \in X_n$. The condition (2) guarantees that an attacker gains no additional information by observing the side-channel compared to what he already knew by observing the regular-channel to distinguish secrets in S. As seen in Lemma 3.2, this implies that regular-channel security transfers to the side-channel aware case when the secrets are restricted to S. And, the condition (1) says that the regular-channel attack succeeds with the probability at most p (technically, the condition only stipulates that this holds for all but finitely many n's; for simplicity, we assume the stronger non-asymptotic condition in this informal discussion). Therefore, the probability that a secret is selected from S and that the attack fails to recover the secret is bounded below by $\frac{|S|}{|\mathcal{S}_n|}(1 - p)$. Then, since the secret subsets in X_n are disjoint (i.e., $(S_1, p_1) \neq (S_2, p_2)$ implies $S_1 \cap S_2 = \emptyset$ for any $(S_1, p_1) \in X_n$ and $(S_2, p_2) \in X_n$), the attack must fail with the probability at least $\sum_{(S,p)\in X_n} \frac{|S|}{|\mathcal{S}_n|}(1 - p)$. The theorem below formalizes the intuition.

[6] It is easy to relax the condition to also be asymptotic so that the refinement order relation only needs to hold for large n.

Theorem 3.4 (SRSCR soundness)**.** *Suppose C satisfies* SRSCR(f, X)*. Then, C is (f, ϵ)-secure where $\epsilon : \mathbb{N} \to (0, 1]$ is any function satisfying the condition below:*

(♠) *There exists $N \in \mathbb{N}$ such that for all $n \geqslant N$, $1 - \sum_{(S,p)\in X_n} \frac{|S|}{|\mathcal{S}_n|}(1-p) \leqslant \epsilon(n)$.*

Proof. Suppose for contradiction that C is not (f, ϵ)-secure for ϵ satisfying ♠. That is, there exists an adversary $\mathcal{A}$ such that $\Pr[Win^C_{\mathcal{A}}(n, f)] \geqslant \epsilon(n)$ for infinitely many $n \in \mathbb{N}$. To prove the theorem, it suffices to construct an adversary $\mathcal{A}'$ and $U \prec X$ such that $\Pr[Win^{C|_U}_{\mathcal{A}'}(n, f)] \geqslant err_U(n)$ for infinitely many $n \in \mathbb{N}$ because that would contradict the condition (1) of SRSCR(f, X).

We now show the construction of $\mathcal{A}'$ and U ($\mathcal{A}'$ turns out to be simply $\mathcal{A}$). Because of the condition (2) of SRSCR(f, X) and Lemma 3.2, it suffices to allow such an adversary $\mathcal{A}'$ to also make side-channel observations. That is, it suffices to construct $\mathcal{A}'$ and U such that $\Pr[Win^{C|_U}_{\mathcal{A}'}(n, f)] \geqslant err_U(n)$ for infinitely many $n \in \mathbb{N}$. By condition (♠), there exists $N \in \mathbb{N}$ such that $1 - \sum_{(S,p)\in X_n} \frac{|S|}{|\mathcal{S}_n|}(1-p) \leqslant \epsilon(n)$ for all $n \geqslant N$ (♢). Let $Y = \{n \in \mathbb{N} \mid n \geqslant N \wedge \Pr[Win^C_{\mathcal{A}}(n, f)] \geqslant \epsilon(n)\}$. Note that Y is an infinite set. Fix $n \in Y$. For $S \subseteq \mathcal{S}_n$, let us write $Win^{C|_S}(n, \cdot)$ for the winning event of the modified game in which the space of secrets (of size n) is S. Note that, given $S \subseteq \mathcal{S}_n$, the probability that a secret randomly selected from $\mathcal{S}_n$ belongs to S is $|S|/|\mathcal{S}_n|$, that is, $\Pr[s \in S \mid s \leftarrow \mathcal{S}_n] = |S|/|\mathcal{S}_n|$. Therefore, by disjointness of the secret subsets in X_n, it follows that

$$\begin{aligned}\Pr\big[Win^C_{\mathcal{A}}(n, f)\big] &\leqslant 1 - \sum_{(S,\_)\in X_n} \Pr[s \in S \mid s \leftarrow \mathcal{S}_n] \cdot \big(1 - \Pr\big[Win^{C|_S}_{\mathcal{A}}(n, f)\big]\big) \\ &= 1 - \sum_{(S,\_)\in X_n} \frac{|S|}{|\mathcal{S}_n|} \cdot \big(1 - \Pr\big[Win^{C|_S}_{\mathcal{A}}(n, f)\big]\big).\end{aligned}$$

Then, because $\Pr[Win^C_{\mathcal{A}}(n, f)] \geqslant \epsilon(n)$, from (♢) above, we have that

$$1 - \sum_{(S,p)\in X_n} \frac{|S|}{|\mathcal{S}_n|}(1-p) \leqslant \epsilon(n) \leqslant \Pr\big[Win^C_{\mathcal{A}}(n, f)\big] \leqslant 1 - \sum_{(S,\_)\in X_n} \frac{|S|}{|\mathcal{S}_n|} \cdot \big(1 - \Pr\big[Win^{C|_S}_{\mathcal{A}}(n, f)\big]\big),$$

and thus

$$\sum_{(S,p)\in X_n} \frac{|S|}{|\mathcal{S}_n|}(1-p) \geqslant \sum_{(S,\_)\in X_n} \frac{|S|}{|\mathcal{S}_n|} \cdot \big(1 - \Pr\big[Win^{C|_S}_{\mathcal{A}}(n, f)\big]\big).$$

Therefore there must be $(S', p') \in X_n$ such that $p' \leqslant \Pr[Win^{C_{S'}}_{\mathcal{A}}(n, f)]$. Denote by S^{res}_n any such S', for each $n \in Y$. Let $U \prec X$ be such that $\#_1 U_n = S^{res}_n$ for each $n \in Y$ (U_n is allowed to be an arbitrary element of X_n for $n \notin Y$), and let $\mathcal{A}' = \mathcal{A}$. Then, it follows that $\Pr[Win^{C|_U}_{\mathcal{A}'}(n, f)] \geqslant err_U(n)$ for each $n \in Y$. □

We remark about a restricted case of the SRSCR soundness theorem. Let X be a secret restriction and suppose that there exist $\epsilon : \mathbb{N} \to (0, 1]$ and $r \in (0, 1]$ satisfying the following: for all $n \in \mathbb{N}$, $p = \epsilon(n)$ for all $(\_, p) \in X_n$ and $\frac{|\bigcup_{(S,\_)\in X_n} S|}{|\mathcal{S}_n|} \leqslant r$. Then, Theorem 3.4 implies that a system satisfying SRSCR(f, X) is $(f, 1 - r(1-\epsilon))$ secure. The restricted case corresponds to the version of the theorem presented in [43] (with the disjointness bug corrected). Theorem 3.4 of this paper is more general

because the improved SRSCR condition allows assignments of different error bounds to different secret subsets, stipulated by the secret restriction parameter X. Importantly, as we shall show in the next sections, the more general result allows derivations of substantially tighter security bounds (cf. Corollaries 3.12 and 3.22).

Example 3.5. Recall the bucketed leaky login program from Example 2.3. We show that the program satisfies the SRSCR condition. For each $n \in \mathbb{N}$, $a \in \{0,1\}^{n-n/k}$, and $0 \leqslant i < k$, 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 as in a (and therefore same). That is,

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

where $s[m, \ldots, m']$ and $a[m, \ldots, m']$, for $m > m'$, are arrays with no elements and in which case $s[m, \ldots, m'] = a[m, \ldots, m']$. Fix some $i \in \{0, \ldots, k-1\}$ and $N \in \{1, \ldots, 2^{n/k} - 1\}$. Recall $\epsilon : \mathbb{N} \to (0, 1]$ such that $\epsilon(n) = 1 - \frac{N-1}{2^{n/k}}$ from Example 2.3. Let X be an indexed family such that $X_n = \{(S_n^{a,i}, \epsilon(n)) \mid a \in \{0,1\}^{n-n/k}\}$ for each $n \in \mathbb{N}$. Note that X is a secret restriction for the system because $S_n^{a,i} \cap S_n^{a',i} = \emptyset$ for any $a \neq a'$. Also, $\bigcup_{S \in X_n} S = \mathcal{S}_n$ for each n.

Recall f such that $f(n) = 2^{n/k} - (N+1)$ from Example 2.3. We argue that the system satisfies SRSCR(f, X). We note that condition (1) 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 $2^{n/k}$.[7] To see why condition (2) 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 3.4, it follows that bucketed leaky login program is (f, ϵ)-secure. Note that the bound matches the one given in Example 2.3.

To effectively apply Theorem 3.4, one needs to find a suitable secret restriction X on which the system's regular channel is secure and the side channel satisfies the refinement order relation with respect to the regular channel. As also observed in a prior work [48], the refinement order relation is a 2-safety property [15,42] for which there are a number of effective verification methods [2,6,12,37,40]. For instance, self-composition [3,4,10,19,42] is a well-known technique that can be used to verify arbitrary 2-safety properties.

We note that a main benefit of Theorem 3.4 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 (e.g., by following the method proposed in [7]), while the side-channel conditions are checked, possibly automatically, by established program verification methods such as self composition.

Remark 3.6. We make some additional observations regarding the SRSCR condition. First, while Theorem 3.4 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 3.5 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

[7] The latter follows by an argument similar to the one given in footnote 5 in Example 2.3.

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 (i.e., we fixed an arbitrary $i \in \{0, \ldots, k-1\}$ to construct the restriction X). Extending the technique to enable tighter analyses is left for future work.

Secondly, Theorem 3.4 says that, given a secret restriction X, if the regular channel of the system is (f, err_U)-secure when the secrets are restricted to the indexed family of subsets of secrets $\#_1 U$ for each $U \prec X$, then the whole system is (f, ϵ)-secure under certain conditions for some ϵ determined by the parameter X. 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 remarked in Remark 2.4, the two parameters are often correlated so that smaller f implies smaller ϵ and vice versa. Therefore, Theorem 3.4 suggests that the change in the probability parameter (i.e., from err_U 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 (1) of SRSCR stipulates that the regular channel is (f, err_U)-secure for each restricted indexed family of subsets of secrets $\#_1 U$ 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 $U \prec X$ used in the analysis of Example 3.5, 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 X such that each $U \in X$ is large and satisfies the conditions is important for deriving good security bounds, even when the ratio $|\bigcup_{S \in X_n} S|/|\mathcal{S}_n|$ is large as in the analysis of the bucketed leaky login program.

3.2. Bounded low-input side-channel capacity condition

While SRSCR facilitates proving security of systems by separating regular channels from side channels, it requires one to identify a suitable secret restriction that satisfies the conditions. This can be a hurdle to applying the proof method. To this end, this section presents a condition, called *bounded low-input side-channel capacity* (BLISCC), 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 restricted secret subsets. Roughly, the condition stipulates that the regular channel is secure (for the entire space of secrets) and that the side-channel outputs depend on attacker-controlled inputs only in a *bounded* way. As we shall show, the degree of dependency that we will use is related to *channel capacity*, a notion studied in QIF research [5,9,32,39,48,49], which measures the number of possible observations one can make on the channel.

We show that the system satisfying BLISCC becomes a system satisfying SRSCR once bucketing is applied, where the degree of security (i.e., the parameters f and X of SRSCR) will be proportional to the degree of regular-channel security, capacity bound and the granularity of buckets. Roughly, this holds because for a system whose side-channel capacity relative to attacker-controlled inputs is bounded, bucketing is guaranteed to partition the secrets into a small number of sets (relative to the bucket granularity and the capacity bound) 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 2.2, 2.3 and 3.5), it covers interesting scenarios such as fast modular exponentiation (cf. Example 3.13). Also, as we shall show in Section 3.3, the condition may be used compositionally in combination with the constant-time implementation technique [1,3,11,26] to further widen its applicability.

For a set A and an equivalence relation $\sim \subseteq A \times A$, we write $A/\sim$ for the set of equivalence classes of A. That is, $A/\sim = \{[a]_\sim \mid a \in A\}$ where $[a]_\sim = \{x \in A \mid x \sim a\}$. For a system C and $n \in \mathbb{N}$, we say that $v_1 \in \mathcal{I}_n$ and $v_2 \in \mathcal{I}_n$ are *low-input side-channel equivalent*, written $v_1 \sim_n^{lisc} v_2$, if $\mathsf{sc}_n(s, v_1) = \mathsf{sc}_n(s, v_2)$ for all $s \in \mathcal{S}_n$. It is easy to see that $\sim_n^{lisc}$ is an equivalence relation on $\mathcal{I}_n$.

Definition 3.7 (*LISCC*)**.** We define the *low-input side-channel capacity* of C for $n \in \mathbb{N}$, $LISCC(C, n)$, as follows: $LISCC(C, n) = |\mathcal{I}_n/\sim_n^{lisc}|$.

Intuitively, low-input side-channel capacity measures the number of equivalence classes of low inputs (i.e., attacker-controlled inputs) that can be obtained by observing the side-channel output, where two low inputs are considered equivalent if they yield the same side-channel output for each high input (i.e., secret). It is interesting to note that this is the standard definition of channel capacity for deterministic systems [5,9,32,39,48,49], except that the roles of high inputs and low inputs are reversed.[8] That is, while the standard channel capacity measures the amount of influence that the high inputs exert on the channel output, our low-input channel capacity measures that exerted by the low inputs.

As a special case, we say that C is *low-input side-channel non-interferent* when $LISCC(C, n) = 1$ for all n. Note that low-input side-channel non-interference is equivalent to the following condition which is the standard definition of (side-channel) non-interference [23,47] with the roles of high inputs and low inputs reversed: for all $n \in \mathbb{N}$, $v_1 \in \mathcal{I}_n$, $v_2 \in \mathcal{I}_n$ and $s \in \mathcal{S}_n$, $\mathsf{sc}_n(s, v_1) = \mathsf{sc}_n(s, v_2)$.

We now define the bounded low-input side-channel capacity condition.

Definition 3.8 (BLISCC)**.** Let $f : \mathbb{N} \to \mathbb{N}$, $\epsilon : \mathbb{N} \to (0, 1]$, and $\ell \in \mathbb{N}$. We say that the system C satisfies the *bounded low-input side-channel capacity* condition with f, ϵ and ℓ, written $\mathsf{BLISCC}(f, \epsilon, \ell)$, if the following conditions are satisfied:

(1) C is regular-channel (f, ϵ)-secure; and
(2) For all $n \in \mathbb{N}$, $LISCC(C, n) \leqslant \ell$.[9]

We remark that, when $\ell = 1$, $\mathsf{BLISCC}(f, \epsilon, \ell)$ implies that the side channel is low-input non-interferent. This restricted case corresponds to the LISCNI condition presented in [43].

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

Definition 3.9 (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 $v_1, v_2 \in \mathcal{I}_n$, $\mathsf{sc}\langle C\rangle_n(s_1, v_1) = \mathsf{sc}\langle C\rangle_n(s_2, v_2) \Rightarrow \mathsf{sc}\langle C'\rangle_n(s_1, v_2) = \mathsf{sc}\langle C'\rangle_n(s_2, v_2)$.

[8] In QIF literature, channel capacity is typically defined to be the log of the number of equivalence classes. Here, we use the non-log form which turns out to be more convenient for our purposes.

[9] It is easy to relax the notion to be asymptotic so that channel capacity only needs to be bounded for large n.

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 BLISCC only requires the fact that side channel outputs are partitioned into a small number of buckets. This makes our results applicable to any techniques (beyond the usual bucketing technique for timing channels [8,16,30,31,51]) that reduce the number of possible side-channel outputs. We remark that $\mathit{LISCC}(C, n) \leqslant \ell$ implies that $\mathit{LISCC}(\mathit{Bkt}_k(C), n) \leqslant \ell$.

We shall make use of the following lemma in the proof of BLISCC soundness theorem.

Lemma 3.10. *Let C be a system, $n \in \mathbb{N}$, and $\ell \geqslant 1$. Then, $\mathit{LISCC}(C, n) \leqslant \ell$ if and only if there exist $\mathcal{I}^1, \ldots, \mathcal{I}^\ell$ such that the following conditions hold:*

(a) $\mathcal{I}_n = \bigcup_{i=1}^{\ell} \mathcal{I}^i$, *and* $\mathcal{I}^i \cap \mathcal{I}^j = \emptyset$ *for* $i \neq j$; *and*
(b) *For each* $i \in \{1, \ldots, \ell\}$, $s \in \mathcal{S}_n$, $v_1 \in \mathcal{I}^i$ *and* $v_2 \in \mathcal{I}^i$, $\mathsf{sc}_n(s, v_1) = \mathsf{sc}_n(s, v_2)$.

Proof. We show the if direction. We have that $|\mathcal{I}_n/{\sim_n^{\mathit{lisc}}}| \leqslant \ell$. Therefore, we can construct $\mathcal{I}^1, \ldots, \mathcal{I}^\ell$ satisfying (a) and (b) by letting $\mathcal{I}^1, \ldots, \mathcal{I}^m$ be the elements of $\mathcal{I}_n/{\sim_n^{\mathit{lisc}}}$, and $\mathcal{I}^i = \emptyset$ for each $m < i \leqslant \ell$ where $m = |\mathcal{I}_n/{\sim_n^{\mathit{lisc}}}|$.

We show the only-if direction. Let $\mathcal{I}^1, \ldots, \mathcal{I}^\ell$ satisfy (a) and (b). For each $\mathcal{I}^i \in \{\mathcal{I}^1, \ldots, \mathcal{I}^\ell\}$, there must be an equivalence class $[v]_{\sim_n^{\mathit{lisc}}} \in \mathcal{I}_n/{\sim_n^{\mathit{lisc}}}$ such that $\mathcal{I}^i \subseteq [v]_{\sim_n^{\mathit{lisc}}}$. Let us choose one such $[v]_{\sim_n^{\mathit{lisc}}}$ and identify it by $[v_i]_{\sim_n^{\mathit{lisc}}}$, for each $\mathcal{I}^i \in \{\mathcal{I}^1, \ldots, \mathcal{I}^\ell\}$. Then, $\{[v_i]_{\sim_n^{\mathit{lisc}}} \mid i \in \{1, \ldots, \ell\}\}$ is the set of equivalence classes $\mathcal{I}_n/{\sim_n^{\mathit{lisc}}}$. Therefore, $|\mathcal{I}_n/{\sim_n^{\mathit{lisc}}}| = |\{[v_i]_{\sim_n^{\mathit{lisc}}} \mid i \in \{1, \ldots, \ell\}\}| \leqslant \ell$. □

We now state and prove the BLISCC soundness theorem. Informally, the soundness theorem says that a system satisfying the BLISCC condition becomes one that satisfies the SRSCR condition after some suitable bucketing is applied.

Theorem 3.11 (BLISCC soundness)**.** *Suppose that C satisfies* $\mathsf{BLISCC}(f, \epsilon, \ell)$. *Let $k > 0$ be such that $k^\ell \epsilon \leqslant 1$. Then, $\mathit{Bkt}_k(C)$ satisfies* $\mathsf{SRSCR}(f, X)$ *for some X satisfying the following:*

(♣) *For all* $n \in \mathbb{N}$, $1 - \sum_{(S,p) \in X_n} \frac{|S|}{|\mathcal{S}_n|}(1 - p) \leqslant k^\ell \cdot \epsilon(n)$.

Proof. Let $C' = \mathit{Bkt}_k(C)$. We shall prove by constructing an indexed family X such that X satisfies (♣) and C' satisfies $\mathsf{SRSCR}(f, X)$. First, by condition (2) of BLISCC, it follows that $\mathit{LISCC}(C, n) \leqslant \ell$ for all $n \in \mathbb{N}$ and therefore $\mathit{LISCC}(C', n) \leqslant \ell$ for all $n \in \mathbb{N}$. Therefore, by Lemma 3.10, for each $n \in \mathbb{N}$, there exist $\mathcal{I}_n^1, \ldots, \mathcal{I}_n^\ell$ such that (a) $\mathcal{I}_n^1, \ldots, \mathcal{I}_n^\ell$ partition $\mathcal{I}_n$, and (b) $\mathsf{sc}\langle C'\rangle_n(s, v_1) = \mathsf{sc}\langle C'\rangle_n(s, v_2)$ for all $s \in \mathcal{S}_n$, $v_1 \in \mathcal{I}_n^i$ and $v_2 \in \mathcal{I}_n^i$. set $\{i \in \mathbb{N} \mid 1 \leqslant i \leqslant a\}$. For each $n \in \mathbb{N}$ and $p : [1, \ell] \to [1, k]$, let $S_n^p = \{s \in \mathcal{S}_n \mid \bigwedge_{i \in [1,\ell]} \forall v \in \mathcal{I}_n^i.\mathsf{sc}\langle C'\rangle_n(s, v) = \star_{p(i)}\}$. From (a) and (b), it follows that, for each $s \in \mathcal{S}_n$, there is a unique $p : [1, \ell] \to [1, k]$ satisfying $\bigwedge_{i \in [1,\ell]} \forall v \in I_n^i.\mathsf{sc}_n(s, v) = \star_{p(i)}$. Therefore, S_n^p's partition $\mathcal{S}_n$, that is, $\mathcal{S}_n = \bigcup_{p:[1,\ell]\to[1,k]} S_n^p$ and $\mathcal{S}_n^{p_1} \cap \mathcal{S}_n^{p_2} = \emptyset$ for $p_1 \neq p_2$. Let X be the indexed family defined by

$$X_n = \left\{ \left(S_n^p, \frac{|\mathcal{S}_n|}{|S_n^p|}\epsilon(n) \right) \;\middle|\; p : [1, \ell] \to [1, k] \wedge S_n^p \neq \emptyset \wedge \left(|\mathcal{S}_n|/\left|S_n^p\right|\right)\epsilon(n) \leqslant 1 \right\} \quad \text{for each } n \in \mathbb{N}.$$

We show that X satisfies (♣). Fix $n \in \mathbb{N}$. Let $A = \{S \mid (S, \_) \in X_n\}$ and let A^c be the complement of A, that is, $A^c = \{S_n^p \mid p : [1, \ell] \to [1, k] \wedge S_n^p \notin A\}$. Note that $|A| + |A^c| \leqslant k^\ell$ and $|S_n|\epsilon(n) > |S|$ for

each $S \in A^c$. Therefore,

$$\begin{aligned}
1 - \sum_{(S,p)\in X_n} \frac{|S|}{|\mathcal{S}_n|}(1-p) &= 1 - \sum_{S\in A} \frac{|S|}{|\mathcal{S}_n|}\left(1 - \frac{|S_n|}{|S|}\epsilon(n)\right) \\
&= 1 - \sum_{S\in A} \frac{|S|}{|\mathcal{S}_n|} + |A|\epsilon(n) \\
&= 1 - \left(1 - \sum_{S\in A^c} \frac{|S|}{|\mathcal{S}_n|}\right) + |A|\epsilon(n) \\
&\leqslant 1 - \left(1 - \left|A^c\right| \max_{S\in A^c} \frac{|S|}{|\mathcal{S}_n|}\right) + |A|\epsilon(n) \\
&\leqslant 1 - \left(1 - \left|A^c\right|\epsilon(n)\right) + |A|\epsilon(n) \leqslant k^\ell \epsilon(n).
\end{aligned}$$

Above, the third line follows from the fact that S_n^p's partition $\mathcal{S}_n$, that is, $\bigcup(A \cup A^c) = \mathcal{S}_n$.

Therefore, it suffices to show that C' satisfies $\mathsf{SRSCR}(f, X)$ (with the X we constructed). First, we show that the condition (2) of $\mathsf{SRSCR}(f, X)$ is satisfied. From the construction of S_n^p's, it follows that $\mathsf{sc}\langle C'\rangle_n(s_1, v) = \mathsf{sc}\langle C'\rangle_n(s_2, v) = \star_{p(i)}$ for all $s_1 \in S_n^p$, $s_2 \in S_n^p$ and $v \in \mathcal{I}_n$ where v belongs to the (unique) low-input partition $\mathcal{I}_n^i$. That is, the side channel of C' is non-interferent (with respect to high inputs) for the subset S_n^p. Therefore, C' satisfies $RR(\#_1 U)$ for any $U \prec X$.

It remains to show that the condition (1) of $\mathsf{SRSCR}(f, X)$ is satisfied. For contradiction, suppose that $C'|_U$ is not regular-channel (f, err_U)-secure for some $U \prec X$. Let $\mathcal{A}$ be a regular-channel adversary for $C'|_U$ such that for infinitely many $n \geqslant N$, $\mathcal{A}$ queries (the regular channel of) $C'|_U$ at most $f(n)$ many times and successfully recovers the secret with probability at least $err_U(n)$. We show that running $\mathcal{A}$ against C' (by allowing arbitrary behavior when the secret is not from U) successfully recovers the secret with probability at least $\epsilon(n)$ for infinitely many n. To see this, let $n \in \mathbb{N}$ be such that $\mathcal{A}$ successfully recovers the secret from $C'|_U$ with probability at least $err_U(n)$. Note that the probability that a secret randomly selected from $\mathcal{S}_n$ belongs to S_n^p is $|S_n^p|/|\mathcal{S}_n|$, i.e., $\Pr[s \in S_n^p \mid s \leftarrow \mathcal{S}_n] = |S_n^p|/|\mathcal{S}_n|$. Then, because $err_U(n) = (|\mathcal{S}_n|/|S_n^p|)\epsilon(n)$ where $S_n^p = \#_1 U_n$, it follows that

$$\Pr\left[Win_{\mathcal{A}}^{C'^\bullet}(n, f)\right] \geqslant \Pr[s \in S_n^p \mid s \leftarrow \mathcal{S}_n] \cdot \Pr\left[Win_{\mathcal{A}}^{C'|_U^\bullet}(n, f)\right] \geqslant \frac{|S_n^p|}{|\mathcal{S}_n|} \cdot err_U(n) = \epsilon(n).$$

Because $\Pr[Win_{\mathcal{A}}^{C^\bullet}(n, f)] = \Pr[Win_{\mathcal{A}}^{C'^\bullet}(n, f)]$, this contradicts condition (1) of $\mathsf{BLISCC}(f, \epsilon, \ell)$ which says that C is regular-channel (f, ϵ)-secure. Therefore, $C'|_U$ is regular-channel (f, err_U)-secure. □

As a corollary of Theorems 3.4 and 3.11, we have the following.

Corollary 3.12. *Suppose that C satisfies $\mathsf{BLISCC}(f, \epsilon, \ell)$. Let $k > 0$ be such that $k^\ell \cdot \epsilon \leqslant 1$. Then, $Bkt_k(C)$ is $(f, k^\ell \cdot \epsilon)$-secure.*

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 3.4, Theorem 3.11 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 Remark 2.4, the two parameters are often correlated so that a change in one can affect the other. Also, like SRSCR, BLISCC 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.

We remark that Theorem 3.11 and Corollary 3.12 substantially improve the corresponding results from [43]. First, the results in [43] were restricted only to the low-input side-channel *non-interference*, that is, the case where ℓ is restricted to be 1 in BLISCC(f, ϵ, ℓ). Secondly, even for the low-input side-channel non-interferent case, it could only deduce rather pessimistic security bounds, that is, $(f, 1 - 1/k + \epsilon)$-security. By contrast, this paper generalizes the results to be applicable to cases where the system is not low-input side-channel non-interferent but has a bounded low-input side-channel capacity, that is, the cases where $\ell > 1$. In addition, the new results are able to derive much tighter security bounds. Namely, when $\ell = 1$, we are able to derive $(f, k \cdot \epsilon)$-security, substantially improving the previously derivable bound (note that $k \cdot \epsilon \leqslant 1 - 1/k + \epsilon$ for all valid k and ϵ). For example, this allows the derivation of a strong security bound in Example 3.13 below.[10] We remark that the improvements are made possible by the extended SRSCR soundness theorem of this paper (cf. Theorem 3.4) which generalized the corresponding theorem from [43] by allowing different error bounds to be assigned to different secret subsets.

Example 3.13 (Fast modular exponentiation)**.** Fast modular exponentiation is an operation that is often found in cryptography algorithms such as RSA [27,35]. Figure 2 shows its implementation written in a C-like language. It computes $\texttt{y}^{\texttt{x}}$ `mod` `m` where `x` is the secret represented as a length `n` bit array, `y` is an attacker controlled-input, and `m` is a constant. 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 [27,35].

```
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.

[10] For the same example, [43] erroneously asserts the strong bound which cannot be actually derived by the results in that paper (the error is corrected in the authors' online copy of that paper by asserting a weaker bound [44]).

However, assuming that running time of the operation `(a * y) % m` is independent of `y`, it can be seen that the program satisfies the BLISCC condition.[11] Under the assumption, the program can be formalized as the system C where, for all $n \in \mathbb{N}$,

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

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 $\mathsf{BLISCC}(f, \epsilon, 1)$ 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.[12] Therefore, by applying bucketing, it can be made $(f, k \cdot \epsilon)$-secure for any f and negligible ϵ (and hence (f, ϵ)-secure for any f and negligible ϵ). We remark that the same security bound can also be obtained even when the assumption is relaxed so that low input `y` can exert a non-zero (but bounded) influence on the running time (i.e., when the program satisfies $\mathsf{BLISCC}(f, \epsilon, \ell)$ for some $\ell > 1$).

Example 3.14 (Timeout options)**.** We show another example demonstrating the usefulness of the BLISCC condition. This is a *meta* example, in which we add timeout options to the given program. Suppose that we are given a system C that is regular-channel (f, ϵ)-secure and $LISCC(C, n) \leqslant \ell$ for all $n \in \mathbb{N}$. Let $T = \{t_1, \ldots, t_m\} \subseteq \mathbb{N} \cup \{\infty\}$ be a finite set of *timeout options* where $t < \infty$ for any $t \in \mathbb{N}$. Then, C with the timeout option T is a system C' such that

- $\mathsf{rc}\langle C' \rangle = \mathsf{rc}\langle C \rangle$, $\mathcal{S}\langle C' \rangle = \mathcal{S}\langle C \rangle$, and $\mathcal{O}^{\mathsf{rc}}\langle C' \rangle = \mathcal{O}^{\mathsf{rc}}\langle C \rangle$;
- For all $n \in \mathbb{N}$, $\mathcal{I}\langle C' \rangle_n = \mathcal{I}\langle C \rangle_n \times T$;
- For all $n \in \mathbb{N}$, $\mathcal{O}^{\mathsf{sc}}\langle C' \rangle_n = \mathcal{O}^{\mathsf{sc}}\langle C \rangle_n \cup \{\bot\}$; and
- For all $n \in \mathbb{N}$, $s \in \mathcal{S}_n$ and $(v, t) \in \mathcal{I}\langle C' \rangle_n$, $\mathsf{sc}\langle C' \rangle_n(s, (v, t)) = timeout(t, \mathsf{sc}\langle C \rangle_n(s, v))$ where $timeout = \lambda t_1, t_2.\mathtt{if}\ t_1 < t_2\ \mathtt{then} \perp \mathtt{else}\ t_2$.

Roughly, C' takes an additional timeout parameter in its low input so that the execution is halted at the time given as the parameter and using ∞ for the parameter means no timeout. The occurrence of timeout is represented by the side channel output $\bot$. As we shall show below, BLISCC soundness theorem can be used to deduce that, even with the addition of the timeout option, the security of the system is ensured to a certain degree when bucketing is applied.[13]

First, we show that $LISCC(C', n) \leqslant \ell \times m$ for all $n \in \mathbb{N}$ (recall that $|T| = m$). Let $n \in \mathbb{N}$. Because $LISCC(C, n) \leqslant \ell$, by Lemma 3.10, there exists a partition $\mathcal{I}^1, \ldots, \mathcal{I}^\ell$ of $\mathcal{I}\langle C \rangle_n$ such that for all $i \in$

[11] This is admittedly an optimistic assumption. Indeed, proposed timing attacks exploit the fact that the running time of the operation can depend on `y` [27,35]. 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).

[12] 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.

[13] We have allowed C' to output the (actual) regular-channel output even when timeout has reached. This is sufficient for deriving a security bound as it only increases the attacker's knowledge.

$\{1,\ldots,\ell\}$, $s \in \mathcal{S}_n$, $v_1 \in \mathcal{I}^i$ and $v_2 \in \mathcal{I}^i$, $\mathsf{sc}\langle C\rangle_n(s,v_1) = \mathsf{sc}\langle C\rangle_n(s,v_2)$. For each $s \in \mathcal{S}_n$ and $i \in \{1,\ldots,\ell\}$, let $o_{s,i}$ be the unique $o \in \mathcal{O}^{\mathsf{sc}}\langle C\rangle_n$ satisfying $\forall v \in \mathcal{I}^i.\mathsf{sc}\langle C\rangle_n(s,v) = o$. For each $i \in \{1,\ldots,\ell\}$ and $j \in \{1,\ldots,m\}$, let $\mathcal{I}^{i,j} = \mathcal{I}^i \times \{t_j\}$. Clearly, $\mathcal{I}^{1,1},\ldots,\mathcal{I}^{\ell,m}$ partition $\mathcal{I}\langle C'\rangle_n$. Also, for all $s \in \mathcal{S}_n$, $(v_1,t_j) \in \mathcal{I}^{i,j}$ and $(v_2,t_j) \in \mathcal{I}^{i,j}$, $\mathsf{sc}\langle C'\rangle_n(s,v_1) = \mathsf{sc}\langle C'\rangle_n(s,v_2) = \mathit{timeout}(t_j,o_{s,i})$. Therefore, by Lemma 3.10, $\mathit{LISCC}(C',n) \leqslant \ell \times m$. Therefore, $\mathsf{BLISCC}(f,\epsilon,m\ell)$ is satisfied. Therefore, by Corollary 3.12, $\mathit{Bkt}_k(C')$ is $(f,k^{m\ell}\cdot\epsilon)$-secure, for k satisfying $k^{m\ell}\cdot\epsilon \leqslant 1$.

Example 3.15 (Failure of BLISCC-like conditions on the leaky login program)**.** We cannot apply BLISCC to the leaky login program from Example 2.2 to prove its security after applying bucketing. This is because the program's low-input side-channel capacity is unbounded. Specifically, it can be shown that $\mathit{LISCC}(C,n) = |\mathcal{I}_n| = 2^n$, because for every pair of low inputs $v_1 \in \mathcal{I}_n$ and $v_2 \in \mathcal{I}_n$ such that $v_1 \neq v_2$, there exists a secret $s \in \mathcal{S}_n$ that distinguishes v_1 and v_2 (namely, s whose lengths of matching prefixes are different for v_1 and v_2). Nevertheless, as we have shown in Example 3.5, the program can be made secure by applying bucketing. In fact, as we have shown there, it becomes one that satisfies SRSCR after applying bucketing. Ideally, we would like to find a relatively simple condition (on systems before bucketing is applied) that can cover many systems that would become secure by applying bucketing, such as the leaky login program.

Unfortunately, as we shall show next, there cannot be such a condition if bucketing is formalized in the very permissive way as we have done in this paper. Recall that in Definition 3.9, we have formalized bucketing to be a program transformation that partitions the side-channel outputs into some number of buckets, but placed no restriction on which side-channel outputs are put into which bucket. Below, we show a bucketed version of the leaky login program that is efficiently attackable by an adaptive adversary. As we shall show, bucketing partitions the side-channel outputs into only two buckets, but does so in a rather *unrealistic* manner. This bucketed leaky login program is formalized as the following system:

- rc, sc, $\mathcal{I}$, $\mathcal{O}^{\mathsf{rc}}$ are as in Example 2.2;
- For all $n \in \mathbb{N}$, $\mathcal{O}^{\mathsf{sc}}_n = \{0,1\}$; and
- For all $n \in \mathbb{N}$ and $(s,v) \in \mathcal{S}_n \times \mathcal{I}_n$, $\mathsf{sc}_n(s,v) = (\mathrm{argmax}_i\, s\restriction_i = v\restriction_i) \bmod 2$.

Note that the side-channel outputs are partitioned into two buckets, that is, the side channel outputs 0 if the length of the matching prefix is even and outputs 1 if it is odd.

We show that the system is efficiently attackable. For $b \in \{0,1\}$, let $\overline{b}$ be the complement of b, that is, $\overline{0} = 1$ and $\overline{1} = 0$. For $v \in \{0,1\}^n$ and $0 \leqslant i \leqslant n-1$, let $\mathit{flip}(v,i)$ be the n-bit sequence such that $\mathit{flip}(v,i)[i] = \overline{v[i]}$ and $\mathit{flip}(v,i)[j] = v[j]$ for $j \neq i$. For $s \in \mathcal{S}_n$ and $v \in \mathcal{I}_n$, let us write $\mathit{mplen}(s,v)$ to be the length of the matching prefix, that is, $\mathit{mplen}(s,v) = \mathrm{argmax}_i\, s\restriction_i = v\restriction_i$.

The attack on the system proceeds as follows. The adversary $\mathcal{A}$ chooses an arbitrary $v_1 \in \mathcal{I}_n$ as the initial input and records the output o_1. Note that $o_1 = 0$ if and only if $\mathit{mplen}(s,v_1)$ is even. Then, $\mathcal{A}$ chooses $v_{1,1} = \mathit{flip}(v_1, o_1+1)$ as the next input and observes the output $o_{1,1}$. Note that $o_1 = o_{1,1}$ if and only if $\mathit{mplen}(s,v_1) = o_1$. $\mathcal{A}$ continues the process, each time choosing $v_{1,j} = \mathit{flip}(v_1, o_1+2j-1)$ as the input for each $j \in \{j \in \mathbb{N} \mid j \geqslant 1 \wedge o_1+2j-1 < n\}$ in the increasing order, until he observes $o_{1,j}$ such that $o_1 = o_{1,j}$ (if no such j is found and $s \neq v_1$ which $\mathcal{A}$ can detect by observing the regular channel, then $s = \mathit{flip}(v_1, n-1)$ and so $\mathcal{A}$ has recovered the entire s). At such a point, $\mathcal{A}$ has discovered that $\mathit{mplen}(s,v_1) = o_1+2j-2$. Let $x_1 = \mathit{mplen}(s,v_1)$. Note that $\mathcal{A}$ at this point has recovered the first x_1+1 bits of s (the x_1+1-th bit, $s[x_1]$, is equal to $\overline{v_1[x_1]}$). Also, $\mathcal{A}$ makes $j+1 = (x_1-o_1)/2+2 \leqslant 2(x_1+1)$ queries for this (i.e., the queries $v_1, v_{1,1}, \ldots, v_{1,j}$). $\mathcal{A}$ then chooses as the next input an arbitrary $v_2 \in \mathcal{I}_n$ satisfying $v_2[i] = s[i]$ for all $0 \leqslant i < x_1+1$. $\mathcal{A}$ now uses the same process that he used to discover x_1

to discover $x_2 = \mathit{mplen}(s, v_2)$, except that this time, he starts with the query $v_{2,1} = \mathit{flip}(v_2, x_1 + o_2 + 2)$ where o_2 is the output given the input v_2. By this process, $\mathcal{A}$ will recover up to the $x_2 + 1$-th bit of s in at most $2(x_2 - x_1)$ additional queries. Repeating this, $\mathcal{A}$ makes at most $2n$ queries total to recover the entire bits of s. Thus, the system is not $(2n, \epsilon)$-secure for any ϵ.

The above shows that our definition of bucketing is too permissive for proving its effectiveness on the leaky login program. The issue here is that the definition allows an arbitrary, even *unrealistic*, partitioning of side-channel outputs. Note that partitioning odd timings from even timings cannot happen in reality because for any timing outputs o_1, o_2, o_3 such that $o_1 \leqslant o_2 \leqslant o_3$, if o_1 and o_3 are put in a same bucket then so must o_2 (recall that bucketing is a technique that buffers and delays the output until the next time interval). A possible remedy for the issue is to incorporate the ordering constraint in the definition of bucketing so that an unrealistic partitioning like the above would be disallowed. We leave the issue for future work.

Remark 3.16. We make some additional observations regarding the BLISCC condition. First, as we have noted before, low-input side-channel capacity is equivalent to the ordinary (i.e., high-input) side-channel capacity but with the roles of high inputs and low inputs reversed. Therefore, it is amenable to various techniques proposed for checking and inferring channel capacity [5,9,32,39,48,49]. Furthermore, as we have noted before, in the case where the bound is 1, the problem becomes low-input side-channel non-interference. Similar to the refinement order relation (i.e., condition (2) of SRSCR), low-input side-channel non-interference is a 2-safety property and can be checked by the methods for checking ordinary side-channel non-interference by reversing the roles of high inputs and low inputs [1,3,6,11,24].

Secondly, we remark that it is easy to extend the channel capacity parameter ℓ in $\mathsf{BLISCC}(f, \epsilon, \ell)$ to be a function on the security parameter (i.e., n) similar to f and ϵ. Such an extension would allow us to reason about the case where the low-input side-channel capacity can vary for different sizes of secrets (however, because ℓ appears in the exponent of the deduced security bound, the extension will only be useful for the case when the channel capacity grows very slowly). Likewise, while we restricted the number of buckets parameter k in the definition of bucketing constant, it is easy to extend it to be a function on the security parameter, which would allow us to analyze the security guarantee from bucketing strategies that use different numbers of buckets for different sizes of secrets.

3.3. Combining bucketing and constant-time implementation compositionally

We show that the BLISCC condition may be applied compositionally with the constant-time implementation technique (technically, we will only apply the condition (2) of BLISCC compositionally). As we shall show next, the combined approach is able to ensure security of some non-constant-time systems that cannot be made secure 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 3.17 (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 3.18 (CT soundness)**.** *If C satisfies $\mathsf{CT}(f, \epsilon)$, then C is (f, ϵ)-secure.*

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

Example 3.19. Figure 3 shows a simple, albeit contrived, program that we will use to motivate the combined approach. Here, `sec` is an n-bit secret and `inp` is an 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 the system $C_{\texttt{comp}}$ where for all $n \in \mathbb{N}$,

- $\mathcal{S}_n = \mathcal{I}_n = \{0, 1\}^n$;
- $\mathcal{O}^{\mathsf{rc}}_n = \{\bullet\}$;
- $\mathcal{O}^{\mathsf{sc}}_n = \{i \in \mathbb{N} \mid i \leqslant 2^{n+1}\}$;
- 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 $\mathsf{BLISCC}(f, \epsilon, \ell)$ for any f, ϵ and ℓ either. This is because $\mathsf{sc}_n(s, v_1) \neq \mathsf{sc}_n(s, v_2)$ for any s and $v_1 \neq v_2$, and therefore $\mathit{LISCC}(C_{\texttt{comp}}, n) = |\mathcal{I}_n| = 2^n$. 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 $C_{\texttt{comp}}$;
- 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 $s + v \leqslant 2^n$, and $\mathsf{sc}\langle C'\rangle_n(s, v) = 1$ otherwise.

```
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.

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 $0 \leqslant s \leqslant 2^{n-1}$ (i.e., the side-channel output was 0) or $2^{n-1} < s \leqslant 2^n$ (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 3.21) states that the sequential composition of a constant-time system with a system whose side channel has a bounded low-input channel capacity can be made secure by applying bucketing to only the non-constant-time component. As with BLISCC, the degree of security of the composed system is relative to that of the regular channel, the channel capacity quantity, and the granularity of buckets.

To state the compositionality theorem, we explicitly separate the conditions on side channels of CT and BLISCC 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 $\mathsf{CT}(\_, \_)$, 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 $\mathsf{BLISCC}^{\mathsf{sc}}(\ell)$ if it satisfies condition (2) of $\mathsf{BLISCC}(\_, \_, \ell)$, that is, for all $n \in \mathbb{N}$, $\mathit{LISCC}(C, n) \leqslant \ell$. Next, we define sequential composition of systems.

Definition 3.20 (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 3.21 (Compositionality)**.** *Let $C^\dagger$ be a system that satisfies $\mathsf{BLISCC}^{\mathsf{sc}}(\ell)$ and $C^\ddagger$ be a system that satisfies $\mathsf{CT}^{\mathsf{sc}}$. Let $k > 0$. Suppose that $\mathit{Bkt}_k(C^\dagger); C^\ddagger$ is regular-channel (f, ϵ)-secure where $k \cdot \epsilon \leqslant 1$. Then, $\mathit{Bkt}_k(C^\dagger); C^\ddagger$ satisfies $\mathsf{SRSCR}(f, X)$ for some X satisfying the following:*

(♣) *For all $n \in \mathbb{N}$, $1 - \sum_{(S,p)\in X_n} \frac{|S|}{|\mathcal{S}_n|}(1 - p) \leqslant k^\ell \cdot \epsilon(n)$.*

Proof. Let $C = \mathit{Bkt}_k(C^\dagger); C^\ddagger$. The proof is similar to that of Theorem 3.11, and we prove by constructing an indexed family X such that X satisfies (♣) and C satisfies $\mathsf{SRSCR}(f, X)$. First, by $\mathsf{BLISCC}^{\mathsf{sc}}$, it follows that $\mathit{LISCC}(\mathit{Bkt}_k(C^\dagger), n) \leqslant \ell$ for all $n \in \mathbb{N}$. Therefore, by Lemma 3.10, for each $n \in \mathbb{N}$, there exist $\mathcal{I}_n^1, \ldots, \mathcal{I}_n^\ell$ such that (a) $\mathcal{I}_n^1, \ldots, \mathcal{I}_n^\ell$ partition $\mathcal{I}_n$, and (b) for each $i \in \{1, \ldots, \ell\}$, $\mathsf{sc}\langle \mathit{Bkt}_k(C^\dagger)\rangle_n(s, v_1) = \mathsf{sc}\langle \mathit{Bkt}_k(C^\dagger)\rangle_n(s, v_2)$ for all $s \in \mathcal{S}_n$, $v_1 \in \mathcal{I}_n^i$ and $v_2 \in \mathcal{I}_n^i$. For each $n \in \mathbb{N}$ and $p : [1, \ell] \to [1, k]$, let $S_n^p = \{s \in \mathcal{S}_n \mid \bigwedge_{i\in[1,\ell]} \forall v \in \mathcal{I}_n^i.\mathsf{sc}\langle \mathit{Bkt}_k(C^\dagger)\rangle_n(s, v) = \star_{p(i)}\}$. From (a) and (b), it is immediate that S_n^p's partition $\mathcal{S}_n$, that is, $\mathcal{S}_n = \bigcup_{p:[1,\ell]\to[1,k]} S_n^p$ and $\mathcal{S}_n^{p_1} \cap \mathcal{S}_n^{p_2} = \emptyset$ for $p_1 \neq p_2$. Let

X be the indexed family defined by

$$X_n = \left\{ \left(S_n^p, \frac{|\mathcal{S}_n|}{|S_n^p|}\epsilon(n) \right) \;\middle|\; p : [1,\ell] \to [1,k] \wedge S_n^p \neq \emptyset \wedge \left(|\mathcal{S}_n|/\left|S_n^p\right|\right)\epsilon(n) \leqslant 1 \right\} \quad \text{for each } n \in \mathbb{N}.$$

By an argument similar to that in the proof of Theorem 3.11, we can show that X satisfies (♣). Therefore, it suffices to show that C satisfies $\mathsf{SRSCR}(f, X)$ (with the X we constructed). We can show the condition (1) of $\mathsf{SRSCR}(f, X)$ by an argument similar to that in the proof of Theorem 3.11 as the argument only concerns the regular channel. Therefore, it remains to show the condition (2) of $\mathsf{SRSCR}(f, X)$. From the construction of S_n^p's, it follows that $\mathsf{sc}\langle Bkt_k(C^\dagger)\rangle_n(s_1, v) = \mathsf{sc}\langle Bkt_k(C^\dagger)\rangle_n(s_2, v) = \star_{p(i)}$ for all $s_1 \in S_n^p$, $s_2 \in S_n^p$ and $v \in \mathcal{I}_n$ where $\mathcal{I}_n^i$ is the unique low input partition such that $v \in \mathcal{I}_n^i$. Then, 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 $C^\ddagger$, for all $v \in \mathcal{I}_n$, $s_1 \in S_n^p$ and $s_2 \in S_n^p$, we have

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

Therefore, the side channel of C is non-interferent (with respect to high inputs) for the subset S_n^p. Therefore, C satisfies $RR(\#_1 U)$ for any $U \prec X$. □

As a corollary of Theorems 3.4 and 3.21, we have the following.

Corollary 3.22. *Let $C^\dagger$ be a system that satisfies $\mathsf{BLISCC}^{\mathsf{sc}}(\ell)$ and $C^\ddagger$ be a system that satisfies $\mathsf{CT}^{\mathsf{sc}}$. Let $k > 0$. Suppose that $Bkt_k(C^\dagger); C^\ddagger$ is regular-channel (f, ϵ)-secure where $k \cdot \epsilon \leqslant 1$. Then, $Bkt_k(C^\dagger); C^\ddagger$ is $(f, k^\ell \cdot \epsilon)$-secure.*

We note that the notion of sequential composition is symmetric. Therefore, Corollary 3.22 implies that 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 that satisfies the constant-time property (i.e., satisfies $\mathsf{CT}^{\mathsf{sc}}$) and a component with a bounded low-input side-channel capacity (i.e., satisfies $\mathsf{BLISCC}^{\mathsf{sc}}$), we can ensure the security of C by proving its regular-channel security and applying bucketing only to the non-constant-time component.

Example 3.23. Let us apply compositional bucketing to the system $C_{\texttt{comp}}$ from Example 3.19. 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 $C_{\texttt{comp}} = C^\dagger; C^\ddagger$ where $C^\dagger$ and $C^\ddagger$ satisfy the following:

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

Roughly, $C_{\texttt{comp}}$ is decomposed so that $C^\dagger$ is the first loop and $C^\ddagger$ is the second loop (cf. Fig. 3).

Note that $C^{\ddagger}$ satisfies $\mathsf{CT}^{\mathsf{sc}}$ as its side-channel outputs are high-input independent, and, $C^{\dagger}$ satisfies $\mathsf{BLISCC}^{\mathsf{sc}}(1)$ 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 $1/|\mathcal{S}_n|$.[14] Therefore, by Corollary 3.22, $Bkt_k(C^{\dagger}); C^{\ddagger}$ is $(f, k^{\ell}\epsilon)$-secure for any f and negligible ϵ, and therefore, $Bkt_k(C^{\dagger}); C^{\ddagger}$ is (f, ϵ)-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 BLISCC is the dual and says that the side-channel outputs have a limited dependency on low inputs but may depend arbitrarily on high inputs. Our compositionality theorem (Theorem 3.21) 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 $\mathsf{CT}^{\mathsf{sc}}$. Likewise, as shown below, sequentially composing components that satisfy $\mathsf{BLISCC}^{\mathsf{sc}}$ results in a system that satisfies $\mathsf{BLISCC}^{\mathsf{sc}}$.

Lemma 3.24. *Suppose that C_1 satisfies $\mathsf{BLISCC}^{\mathsf{sc}}(\ell_1)$ and C_2 satisfies $\mathsf{BLISCC}^{\mathsf{sc}}(\ell_2)$, then $C_1; C_2$ satisfies $\mathsf{BLISCC}^{\mathsf{sc}}(\ell_1 \cdot \ell_2)$.*

Proof. Fix $n \in \mathbb{N}$. By Lemma 3.10, for each $j \in \{1, 2\}$, there exist $\mathcal{I}_j^1, \ldots, \mathcal{I}_j^{\ell_i}$ such that (a) $\mathcal{I}_j^1, \ldots, \mathcal{I}_j^{\ell_j}$ partition $\mathcal{I}_n$, and (b) for each $i \in \{1, \ldots, \ell_j\}$, $\mathsf{sc}\langle C_i\rangle_n(s, v_1) = \mathsf{sc}\langle C'\rangle_n(s, v_2)$ for all $s \in \mathcal{S}_n$, $v_1 \in \mathcal{I}_j^i$ and $v_2 \in \mathcal{I}_j^i$.

For each $(i_1, i_2) \in [1, \ell_1]\times[1, \ell_2]$, let $\mathcal{I}^{i_1,i_2} = \{v \in \mathcal{I}_n \mid v \in \mathcal{I}_1^{i_1} \wedge v \in \mathcal{I}_2^{i_2}\}$. It is easy to see that the sets $\mathcal{I}^{i_1,i_2}$ partition $\mathcal{I}_n$, that is, $\mathcal{I}_n = \bigcup_{(i_1,i_2)\in[1,\ell_1]\times[1,\ell_2]} \mathcal{I}^{i_1,i_2}$ and $\mathcal{I}^{i_1,i_2} \cap \mathcal{I}^{i'_1,i'_2} = \emptyset$ for any $(i_1, i_2) \neq (i'_1, i'_2)$. Also, for any $s \in \mathcal{S}_n$, $v \in \mathcal{I}^{i_1,i_2}$ and $v' \in \mathcal{I}^{i_1,i_2}$, we have

$$\begin{aligned}\mathsf{sc}\langle C_1; C_2\rangle_n(s, v) &= \mathsf{sc}\langle C_1\rangle_n(s, v) + \mathsf{sc}\langle C_2\rangle_n(s, v)\\ &= \mathsf{sc}\langle C_1\rangle_n\big(s, v'\big) + \mathsf{sc}\langle C_2\rangle_n\big(s, v'\big)\\ &= \mathsf{sc}\langle C_1; C_2\rangle_n(s, v)\end{aligned}$$

because $\mathsf{sc}\langle C_1\rangle_n(s, v) = \mathsf{sc}\langle C_1\rangle_n(s, v')$ and $\mathsf{sc}\langle C_2\rangle_n(s, v) = \mathsf{sc}\langle C_2\rangle_n(s, v')$. Therefore, by Lemma 3.10, $C_1; C_2$ satisfies $\mathsf{BLISCC}(\ell_1 \cdot \ell_2)$. □

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.

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

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 been experimental evaluations on the effectiveness of bucketing and other timing-channel mitigation schemes [16,22], and other works have proposed information-theoretic methods for formally analyzing the security of (deterministic and probabilistic) systems against adaptive adversaries [14,29].

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. [30,31] 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 [5,32,39,49,50]. 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. In fact, the results of [30,31] may be seen as an instance of our $\mathsf{BLISCC}(f, \epsilon, \ell)$ condition for the case $\ell = 1$ because blinding makes the behavior of cryptographic algorithms effectively independent of attacker-controlled inputs. 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) [1,3,6,11,24,26]. The previous works have proposed methods for verifying that the given system is constant-time [3,6,11,24] or transforming it to one that is constant-time [1,26]. As we have also discussed in this paper (cf. Theorem 3.18), 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 [8,16,30,31,51], we hope that our results will contribute to a better understanding of the bucketing technique and foster further research on the topic.

Next, we remark on a recent work by Tizpaz-Niari et al. [45] that proposes an interesting mitigation scheme against timing channel attacks. Like every timing channel mitigation schemes, their scheme works by changing the time when an output is released to the adversary-observing environment. However, whereas bucketing simply delays the output of the (unmodified) system until the next time interval by blackbox monitoring, they propose to monitor the internal of the system so that different amounts of delay can be added for different runs of the system. Similar to bucketing, they use the delays to partition the outputs to a small set of possibilities. However, a much more flexible partitioning is possible thanks to the internal monitoring (indeed, an arbitrary partitioning can be achieved by this scheme). They argue the security in terms of quantitative information flow. However, they implement their scheme as a *best effort* method that uses dynamic analyses to estimate the timing channel leakage and adds delays based on run-time characteristics such as basic block calls, and therefore they lack a formal guarantee of security. Also, unlike our work which uses a *parametric* notion of security (i.e., (f, ϵ)-security),

they use what they call a *functional observation* model of security, which amounts to security against an adversary who makes an unlimited number of queries (i.e., with every possible public inputs). Note that a parametric notion of security is required to argue the security of systems like the bucketed leaky login program, because an adversary is able to always recover the secret for that system if an unlimited number of queries to the side channel are allowed.

Finally, we remark on a work by Svenningsson and Sands [41] which is related to the SRSCR condition of this paper in an interesting way. Their work suggests to verify side-channel security by having the programmer provide a *declassification* program (similarly in style to *delimited information release* [38]) whose regular channel leaks at least as much information about the secrets as the side channel of the target system does. That is, the side channel of the target system is in the refinement order relation with the regular channel of the declassification program. This allows the side-channel security of the target system to be reduced to the regular-channel security of the target system and that of the declassification program. As such, their work is related to our SRSCR condition that also relates information leaked by a side channel to that leaked by a regular channel. However, whereas our SRSCR considers refinement order relation on subsets of secrets (for which the regular-channel can be proven secure), their work considers that on the entire space of secrets. Furthermore, their approach requires a separate declassification program whose regular-channel is related to the target system's side-channel whereas SRSCR relates the regular and side channels of the same target system.

5. Conclusion and future work

In this paper, we have presented a formal analysis of the effectiveness of the bucketing technique against adaptive side-channel-observing adversaries. We have shown that bucketing is in general insufficient against such adversaries, and presented two novel conditions, SRSCR and BLISCC, that guarantee security against such adversaries. SRSCR states that a system that satisfies it is secure, whereas BLISCC states that 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 adversaries 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 BLISCC 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. We have opted to adopt an asymptotic notion of security to define (f, ϵ)-security, mainly to simplify the exposition of applying our techniques to programs with only computational security guarantee, such as the fast modular exponentiation program. However, all of our results hold even if the definition of security is changed to be non-asymptotic (i.e., by requiring N to be 0 in Definition 2.1) by only changing the condition ♠ in Theorem 3.4 to be non-asymptotic, that is, $\forall n \in \mathbb{N}. 1 - \sum_{(S,p)\in X_n} \frac{|S|}{|\mathcal{S}_n|}(1-p) \leqslant \epsilon(n)$.

While we have instantiated our results to 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, 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 2.2, 2.3 and 3.5. 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.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Numbers 17H01720, 18K19787, 20H04162 and 20K20625, JSPS Core-to-Core Program, A.Advanced Research Networks, and Office of Naval Research (ONR) award #N00014-17-1-2787.

References

[1] J. Agat, Transforming out timing leaks, in: *Proceedings of the 27th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL 2000)*, ACM, 2000, pp. 40–53.
[2] A. Aguirre, G. Barthe, M. Gaboardi, D. Garg and P. Strub, A relational logic for higher-order programs, *PACMPL* **1**(ICFP) (2017), 21:1–21:29.
[3] J.B. Almeida, M. Barbosa, G. Barthe, F. Dupressoir and M. Emmi, Verifying constant-time implementations, in: *USENIX Security Symposium*, 2016, pp. 53–70.
[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), 796–812. doi:10.1016/j.scico.2011.10.008.
[5] M.S. Alvim, K. Chatzikokolakis, C. Palamidessi and G. Smith, Measuring information leakage using generalized gain functions, in: *Proceedings of the 25th IEEE Computer Security Foundations Symposium (CSF 2012)*, IEEE Computer Society, 2012, pp. 265–279. doi:10.1109/CSF.2012.26.
[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: *Proceedings of the 38th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2017)*, ACM, 2017, pp. 362–375. doi:10.1145/3062341.3062378.
[7] T. Antonopoulos and T. Terauchi, Games for security under adaptive adversaries, in: *Proceedings of the 31st IEEE Computer Security Foundations Symposium (CSF 2019)*, IEEE Computer Society, 2019, pp. 216–229. doi:10.1109/CSF.2019.00022.
[8] A. Askarov, D. Zhang and A.C. Myers, Predictive black-box mitigation of timing channels, in: *Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS 2010)*, ACM, 2010, pp. 297–307.
[9] M. Backes, B. Köpf and A. Rybalchenko, Automatic discovery and quantification of information leaks, in: *Proceedings of the 30th IEEE Symposium on Security and Privacy (S&P 2009)*, IEEE Computer Society, 2009, pp. 141–153. doi:10.1109/SP.2009.18.
[10] G. Barthe, P.R. D'Argenio and T. Rezk, Secure information flow by self-composition, *Mathematical Structures in Computer Science* **21**(6) (2011), 1207–1252. doi:10.1017/S0960129511000193.
[11] G. Barthe, B. Grégoire and V. Laporte, Secure compilation of side-channel countermeasures: The case of cryptographic "constant-time", in: *Proceedings of the 31st IEEE Computer Security Foundations Symposium (CSF 2018)*, IEEE Computer Society, 2018, pp. 328–343. doi:10.1109/CSF.2018.00031.
[12] N. Benton, Simple relational correctness proofs for static analyses and program transformations, in: *Proceedings of the 31st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL 2004)*, ACM, 2004, pp. 14–25.
[13] A. Blot, M. Yamamoto and T. Terauchi, Compositional synthesis of leakage resilient programs, in: *Proceedings of the 6th International Conference on Principles of Security and Trust (POST 2017)*, Lecture Notes in Computer Science, Vol. 10204, Springer, 2017, pp. 277–297. doi:10.1007/978-3-662-54455-6_13.
[14] M. Boreale and F. Pampaloni, Quantitative information flow under generic leakage functions and adaptive adversaries, *Logical Methods in Computer Science* **11**(4) (2015). doi:10.2168/LMCS-11(4:5)2015.
[15] M.R. Clarkson and F.B. Schneider, Hyperproperties, *Journal of Computer Security* **18**(6) (2010), 1157–1210. doi:10.3233/JCS-2009-0393.

[16] Y.G. Dantas, R. Gay, T. Hamann, H. Mantel and J. Schickel, An evaluation of bucketing in systems with non-deterministic timing behavior, in: *Proceedings of the 33rd IFIP TC 11 International Conference on ICT Systems Security and Privacy Protection (Section 2018)*, IFIP Advances in Information and Communication Technology, Vol. 529, Springer, 2018, pp. 323–338. doi:10.1007/978-3-319-99828-2_23.
[17] DARPA Space/Time Analysis for Cybersecurity (STAC) program, 2017.
[18] 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), 4:1–4:32. doi:10.1145/2756550.
[19] M. Eilers, P. Müller and S. Hitz, Modular product programs, in: *Proceedings of the 27th European Symposium on Programming (ESOP 2018)*, Lecture Notes in Computer Science, Vol. 10801, Springer, 2018, pp. 502–529.
[20] H. Eldib and C. Wang, Synthesis of masking countermeasures against side channel attacks, in: *Proceedings of the 28th International Conference on Computer Aided Verification (CAV 2014)*, Lecture Notes in Computer Science, Vol. 8559, Springer, 2014, pp. 114–130.
[21] K. Gandolfi, C. Mourtel and F. Olivier, Electromagnetic analysis: Concrete results, in: *Proceedings of the 3rd International Workshop on Cryptographic Hardware and Embedded Systems (CHES 2001)*, Lecture Notes in Computer Science, Vol. 2162, Springer, 2001, pp. 251–261. doi:10.1007/3-540-44709-1_21.
[22] R. Gay, H. Mantel and H. Sudbrock, An empirical bandwidth analysis of interrupt-related covert channels, *IJSSE* **6**(2) (2015), 1–22.
[23] J.A. Goguen and J. Meseguer, Security policies and security models, in: *Proceedings of the 3rd IEEE Symposium on Security and Privacy (S&P 1982)*, IEEE Computer Society, 1982, pp. 11–20. doi:10.1109/SP.1982.10014.
[24] D. Hedin and D. Sands, Timing aware information flow security for a JavaCard-like bytecode, in: *Proceedings of the 1st Workshop on Bytecode Semantics, Verification, Analysis and Transformation (BYTE 2005)*, Electronic Notes in Theoretical Computer Science, Vol. 141, Elsevier, 2005, pp. 163–182.
[25] Y. Ishai, A. Sahai and D.A. Wagner, Private circuits: Securing hardware against probing attacks, in: *Proceedings of the 23rd Annual International Cryptology Conference on Advances in Cryptology (CRYPTO 2003)*, Lecture Notes in Computer Science, Vol. 2729, Springer, 2003, pp. 463–481. doi:10.1007/978-3-540-45146-4_27.
[26] N. Kobayashi and K. Shirane, Type-based information analysis for low-level languages, in: *Proceedings of the 3rd Asian Workshop on Programming Languages and Systems (APLAS 2002)*, 2002, pp. 302–316.
[27] P.C. Kocher, Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems, in: *Proceedings of the 16th Annual International Cryptology Conference on Advances in Cryptology (CRYPTO 1996)*, Lecture Notes in Computer Science, Vol. 1109, Springer, 1996, pp. 104–113.
[28] P.C. Kocher, J. Jaffe and B. Jun, Differential power analysis, in: *Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology (CRYPTO 1999)*, Lecture Notes in Computer Science, Vol. 1666, Springer, 1999, pp. 388–397.
[29] B. Köpf and D.A. Basin, Automatically deriving information-theoretic bounds for adaptive side-channel attacks, *Journal of Computer Security* **19**(1) (2011), 1–31. doi:10.3233/JCS-2009-0397.
[30] B. Köpf and M. Dürmuth, A provably secure and efficient countermeasure against timing attacks, in: *Proceedings of the 22nd IEEE Computer Security Foundations Symposium (CSF 2009)*, IEEE Computer Society, 2009, pp. 324–335. doi:10.1109/CSF.2009.21.
[31] B. Köpf and G. Smith, Vulnerability bounds and leakage resilience of blinded cryptography under timing attacks, in: *Proceedings of the 23rd IEEE Computer Security Foundations Symposium (CSF 2010)*, IEEE Computer Society, 2010, pp. 44–56. doi:10.1109/CSF.2010.11.
[32] P. Malacaria, Assessing security threats of looping constructs, in: *Proceedings of the 34th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL 2007)*, ACM, 2007, pp. 225–235.
[33] P. Malacaria, Algebraic foundations for quantitative information flow, *Mathematical Structures in Computer Science* **25**(2) (2015), 404–428. doi:10.1017/S0960129513000649.
[34] A. McIver, C. Morgan, G. Smith, B. Espinoza and L. Meinicke, Abstract channels and their robust information-leakage ordering, in: *Proceedings of the 3rd International Conference on Principles of Security and Trust (POST 2014)*, 2014, pp. 83–102. doi:10.1007/978-3-642-54792-8_5.
[35] C.S. Pasareanu, Q. Phan and P. Malacaria, Multi-run side-channel analysis using symbolic execution and max-SMT, in: *Proceedings of the 29th IEEE Computer Security Foundations Symposium (CSF 2016)*, IEEE Computer Society, 2016, pp. 387–400. doi:10.1109/CSF.2016.34.
[36] J. Quisquater and D. Samyde, ElectroMagnetic analysis (EMA): Measures and counter-measures for smart cards, in: *Proceedings of the Smart Card Programming and Security, International Conference on Research in Smart Cards (E-Smart 2001)*, Lecture Notes in Computer Science, Vol. 2140, Springer, 2001, pp. 200–210. doi:10.1007/3-540-45418-7_17.
[37] J.C. Reynolds, *The Craft of Programming*, Prentice Hall International Series in Computer Science, Prentice Hall, 1981.

[38] A. Sabelfeld and A.C. Myers, A model for delimited information release, in: *Proceedings of the Software Security – Theories and Systems, Second Mext-NSF-JSPS International Symposium (ISSS 2003)*, Lecture Notes in Computer Science, Vol. 3233, Springer, 2003, pp. 174–191.
[39] G. Smith, On the foundations of quantitative information flow, in: *Proceedings of the 12th International Conference on Foundations of Software Science and Computational Structures (FOSSACS 2009)*, Lecture Notes in Computer Science, Vol. 5504, Springer, 2009, pp. 288–302.
[40] M. Sousa and I. Dillig, Cartesian hoare logic for verifying k-safety properties, in: *Proceedings of the 37th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2016)*, ACM, 2016, pp. 57–69. doi:10.1145/2908080.2908092.
[41] J. Svenningsson and D. Sands, Specification and verification of side channel declassification, in: *Proceedings of the 6th International Workshop on Formal Aspects in Security and Trust (FAST 2009)*, Lecture Notes in Computer Science, Vol. 5983, Springer, 2009, pp. 111–125. doi:10.1007/978-3-642-12459-4_9.
[42] T. Terauchi and A. Aiken, Secure information flow as a safety problem, in: *Proceedings of the 12th International Symposium (SAS 2005)*, Lecture Notes in Computer Science, Vol. 3672, Springer, 2005, pp. 352–367.
[43] T. Terauchi and T. Antonopoulos, A formal analysis of timing channel security via bucketing, in: *Proceedings of the 8th International Conference on Principles of Security and Trust (POST 2019)*, Lecture Notes in Computer Science, Vol. 11426, Springer, 2019, pp. 29–50. doi:10.1007/978-3-030-17138-4_2.
[44] T. Terauchi and T. Antonopoulos, A formal analysis of timing channel security via bucketing, http://www.f.waseda.jp/terauchi.
[45] S. Tizpaz-Niari, P. Cerný and A. Trivedi, Quantitative mitigation of timing side channels, in: *Proceedings of the 31st International Conference on Computer Aided Verification (CAV 2019)*, 2019, pp. 140–160.
[46] E. Tromer, D.A. Osvik and A. Shamir, Efficient cache attacks on AES, and countermeasures, *Journal of Cryptology* **23**(1) (2010), 37–71. doi:10.1007/s00145-009-9049-y.
[47] 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), 167–188. doi:10.3233/JCS-1996-42-304.
[48] H. Yasuoka and T. Terauchi, Quantitative information flow – verification hardness and possibilities, in: *Proceedings of the 23rd IEEE Computer Security Foundations Symposium (CSF 2010)*, IEEE Computer Society, 2010, pp. 15–27. doi:10.1109/CSF.2010.9.
[49] H. Yasuoka and T. Terauchi, On bounding problems of quantitative information flow, *Journal of Computer Security* **19**(6) (2011), 1029–1082. doi:10.3233/JCS-2011-0437.
[50] H. Yasuoka and T. Terauchi, Quantitative information flow as safety and liveness hyperproperties, *Theoretical Compututer Science* **538** (2014), 167–182. doi:10.1016/j.tcs.2013.07.031.
[51] D. Zhang, A. Askarov and A.C. Myers, Language-based control and mitigation of timing channels, in: *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2012)*, ACM, 2012, pp. 99–110.