EasyChair Preprint

№ 442

An Algebraic Approach for Reasoning About Information Flow

Arthur Américo, Mario S. Alvim and Annabelle McIver

August 18, 2018

An Algebraic Approach for Reasoning About Information Flow

Arthur Américo[1], Mário S. Alvim[1], and Annabelle McIver[2]

[1] Universidade Federal de Minas Gerais, Belo Horizonte, Brazil
[2] Macquarie University, Sydney, Australia

Abstract. This paper concerns the analysis of information leaks in security systems. We address the problem of specifying and analyzing large systems in the (standard) channel model used in quantitative information flow (QIF). We propose several operators which match typical interactions between system components. We explore their algebraic properties with respect to the security-preserving refinement relation defined by Alvim et al. and McIver et al. [1,2].
We show how the algebra can be used to simplify large system specifications in order to facilitate the computation of information leakage bounds. We demonstrate our results on the specification and analysis of the Crowds Protocol. Finally, we use the algebra to justify a new algorithm to compute leakage bounds for this protocol.

1 Introduction

Protecting sensitive information from unintended disclosure is a crucial goal for information security. There are, however, many situations in which information leakage is unavoidable. An example is a typical password checker, which must always reveal some information about the secret password—namely whether or not it matches the input provided by the user when trying to log in. Another example concerns election tallies, which reveal information about individual votes by ruling out several configurations of votes (e.g., in the extreme case of an unanimous election, the tally reveals every vote). The field of *Quantitative Information Flow* (QIF) is concerned with quantifying the amount of sensitive information computational systems leak, and it has been extremely active in the past decade [3,4,5,6,7,8,9].

In the QIF framework, systems are described as receiving *secret inputs* from a set of values $\mathcal{X}$, and producing *public*, or *observable*, *outputs* from a set $\mathcal{Y}$. Typical secret inputs are a user's identity, password, or current location, whereas public outputs are anything an adversary can observe about the behavior of the system, such as messages written on the screen, execution time, or power consumption. A system is, then, modeled as an *(information-theoretic) channel*, which is a function mapping each possible pair $x \in \mathcal{X}$, $y \in \mathcal{Y}$ to the conditional probability $p(y \mid x)$ of the system producing output y when receiving input x. Channels abstract technicalities of the system, while retaining the essentials that influence information leakage: the relation between secret input and public output values.

The QIF framework provides a robust theory for deriving security properties from a system's representation as a channel. However, obtaining an appropriate channel to model a system is often a non-trivial task. Moreover, some channels turn out to be so large as to render most security analyses unfeasible in practice.

In this paper we provide an algebra for describing (larger, more complex) channels as compositions of other (smaller, simpler) channels. For that, we define a set of operators, each corresponding to a different way in which components can interact in a system—namely, *parallel* composition, *visible choice* composition, and *hidden choice* composition. We prove a series of algebraic properties of these operators, and use such properties to simplify system specifications so that bounds on the information leakage of a compound system can be inferred from the information leakage of its components. In this way, we allow for leakage analyses of systems which would be intractable with traditional QIF techniques.

This compositional approach seems particularly natural for modeling security protocols, which often involve interactions among various entities. Consider, for instance, the well-known *Dining Cryptographers* anonymity protocol [10]. A group of n cryptographers has been invited for dinner by the NSA (American National Security Agency), who will either pay the bill, or secretly ask one of the cryptographers to be the payer. The cryptographers want to determine whether one among them is the payer, but without revealing which one. For that, they execute the following protocol. In a first phase all participants form a circle, and each tosses a coin and shares the result only with the cryptographer on his right. In a second phase, each cryptographer computes the exclusive-or of the two coins tosses he observed (interpreting *heads* as 0 and *tails* as 1), and publicly announces the result. The only exception is the paying cryptographer (if any), who announces the negation of his exclusive-or. In a third phase, the cryptographers compute the exclusive-or of all announcements. One of them is the payer if, and only if, the result is 1. It has been shown that, if all coins are fair, no information is leaked about who the paying cryptographer is [10].

Despite the Dining Cryptographers relative simplicity, deriving its channel can be a challenging task. Since each of the n cryptographers can announce either 0 or 1, the size of the output set $\mathcal{Y}$, and, consequently, of the channel, increases exponentially with the number of cryptographers. The problem is worsened by the fact that computing the probabilities constituting the channel's entries is not trivial. The algebra we introduce in this paper allows for an intuitive and compositional way of building a protocol's channel from each of its components. To illustrate the concept,

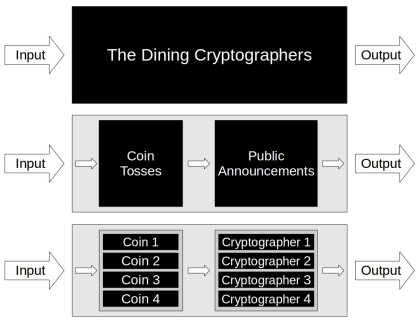

Fig. 1. Schematic representation of the Dining Cryptographers protocol as: (i) a monolithic channel (top); (ii) a composition of two channels (middle); and (ii) a composition of eight channels (bottom).

Figure 1 depicts three alternative representations, using channels, for the Dining Cryptographers with 4 cryptographers and 4 coins. In all models, the input is the identity of the payer (one of the cryptographers or the NSA), and the output are the public announcements of all cryptographers. The top model uses a single (enormous) channel to represent the protocol; the middle one models the protocol as the interaction between two smaller components (the coins and the party of cryptographers); the bottom one uses interactions between even smaller channels (one for each coin and each cryptographer).

The main contributions of this paper are the following.

- We formalize several common operators for channel composition used in the literature, each matching a typical interaction between system components. We prove several relevant algebraic and information-leakage properties of these operators.
- We show that the substitution of components in a system may be subject to unexpected, and perhaps counter-intuitive, results. In particular, we show that overall leakage may increase even when the new component is more secure than the one it is replacing (e.g., Theorems 5 and 6).
- We show how the proposed algebra can be used to simplify large system specifications in order to facilitate the computation of information leakage bounds, given in terms of the g-leakage framework [1,11,12,2].
- We demonstrate our results on the specification and analysis of the Crowds Protocol [13]. We use the proposed algebra to justify a new algorithm to compute leakage bounds for this protocol.

Detailed proofs of all of our technical results can be found in an accompanying technical report[14].

Plan of the paper. The remainder of this paper is organized as follows. In Section 2 we review fundamental concepts from QIF. In Section 3 we introduce our channel operators, and in Section 4 we provide their algebraic properties. In Section 5 we present our main results, concerning information leakage in channel composition. In Section 6 we develop a detailed case study of the Crowds protocol. Finally, in Section 7 we discuss related work, and in Section 8 we conclude.

2 Preliminaries

In this section we review some fundamentals from quantitative information flow.

Secrets, gain functions and vulnerability. A *secret* is some piece of sensitive information that one wants to protect from disclosure. Such sensitive information may concern, for instance, a user's password, identity, personal data, or current location. We represent by $\mathcal{X}$ the set of possible *secret values* the secret may take.

The *adversary* is assumed to have, before observing the system's behaviour, some *a priori* partial knowledge about the secret value. This knowledge is modeled as a probability distribution $\pi \in \mathbb{D}\mathcal{X}$, where $\mathbb{D}\mathcal{X}$ denotes the set of all probability distributions on $\mathcal{X}$. We call π a *prior distribution*, or simply a *prior*.

To quantify how *vulnerable* a secret is—i.e., how prone it is to exploitation by the adversary— we employ a function that maps probability distributions to the real numbers (or, more in general, to any ordered set). Many functions have been used in the literature, such as *Shannon entropy* [15], guessing-entropy [16], Bayes vulnerability [17], and *Rényi min-entropy* [6]. Recently, the *g-leakage* [1] framework was proposed, and it proved to be very successful in capturing a variety of different scenarios, including those in which the adversary benefits from guessing part of secret, guessing a secret approximately, guessing the secret within a number of tries, or gets punished for guessing wrongly. In particular, the framework has been shown to be able to capture all functions mentioned above [12]. In this framework, a finite set $\mathcal{W}$ of *actions* is available to the adversary, and a *gain-function* $g{:}\mathcal{W}\times\mathcal{X}\to[0,1]$ is used to describe the benefit $g(w,x)$ an adversary obtains when he performs action $w\in\mathcal{W}$, and the secret value is $x\in\mathcal{X}$. Given an appropriate gain-function g, the secret's *(prior) g-vulnerability* is defined as the expected value of the adversary's gain if he chooses a best possible action,

$$V_g[\pi] = \max_{w\in\mathcal{W}} \sum_{x\in\mathcal{X}} \pi(x)g(w,x),$$

and the greater its value, the more vulnerable, or insecure, the secret is.

Channels and posterior vulnerabilities In the QIF framework, a system is usually modeled as an *(information theoretic) channel* taking a *secret input* $x\in\mathcal{X}$, and producing a *public*, or *observable*, *output* $y\in\mathcal{Y}$. Each element of $\mathcal{Y}$ represents a behaviour from the system that can be discerned by the adversary. Formally, a channel is a function $C{:}\mathcal{X}\times\mathcal{Y}\to\mathbb{R}$ such that $C(x,y)$ is the conditional probability $p(y|x)$ of the system producing output $y\in\mathcal{Y}$ when input is $x\in\mathcal{X}$.

A channel C together with a prior π induce a joint probability distribution p on the set $\mathcal{X}\times\mathcal{Y}$, given by $p(x,y)=\pi(x)C(x,y)$. From this joint distribution we can derive, for every $x\in\mathcal{X}$ and $y\in\mathcal{Y}$, the marginal probabilities $p(x)=\sum_y p(x,y)$ and $p(y)=\sum_x p(x,y)$, and the conditional probabilities $p(x|y)={}^{p(x,y)}/_{p(y)}$ and $p(y|x)={}^{p(x,y)}/_{p(x)}$. Note that $p(x)=\pi(x)$ and, if $p(x)\neq 0$, $p(y|x)=C(x,y)$.

By observing the output produced by the system, the adversary can update his knowledge about the secret value. More specifically, if the system outputs $y\in\mathcal{Y}$, an adversary can update the prior π to a revised *posterior distribution* $p_{X|y}\in\mathbb{D}\mathcal{X}$ on $\mathcal{X}$ given y, defined for all $x\in\mathcal{X}$ and $y\in\mathcal{Y}$ as $p_{X|y}(x)=p(x|y)$.

Example 1. Let $\mathcal{X}=\{x_1,x_2,x_3\}$ and $\mathcal{Y}=\{y_1,y_2,y_3,y_4\}$ be input and output sets. Let $\pi=({}^1/_2,{}^1/_3,{}^1/_6)$ be a prior, and C be the channel below. The combination of π and C yield a joint probability p, according to the tables below.

C	y_1	y_2	y_3	y_4
x_1	${}^1/_6$	${}^2/_3$	${}^1/_6$	0
x_2	${}^1/_2$	${}^1/_4$	${}^1/_4$	0
x_3	${}^1/_2$	${}^1/_3$	0	${}^1/_6$

$\xrightarrow{\pi}$

p	y_1	y_2	y_3	y_4
x_1	${}^1/_{12}$	${}^1/_3$	${}^1/_{12}$	0
x_2	${}^1/_6$	${}^1/_{12}$	${}^1/_{12}$	0
x_3	${}^1/_{12}$	${}^1/_{18}$	0	${}^1/_{36}$

By summing the columns of the second table, we obtain the marginal probabilities $p(y_1)=1/3$, $p(y_2)=17/36$, $p(y_3)=1/6$ and $p(y_4)=1/36$. These marginal probabilities yield the posterior distributions $p_{X|y_1}=(1/4, 1/2, 1/4)$, $p_{X|y_2}=(12/17, 3/17, 2/17)$, $p_{X|y_3}=(1/2, 1/2, 0)$, and $p_{X|y_4}=(0, 0, 1)$. □

The *posterior g-vulnerability* of a prior π and a channel C is defined as the expected value of the secret's g-vulnerability after the execution of the system:

$$V_g[\pi \rangle C] = \sum_{y\in\mathcal{Y}} \max_{w\in\mathcal{W}} \sum_{x\in\mathcal{X}} C(x,y)\pi(x)g(x,w).$$

The *information leakage* of a prior and a channel is a measure of the increase in secret vulnerability caused by the observation of the system's output. Leakage is, thus, defined as a comparison between the secret's prior and posterior vulnerabilities. Formally, for a gain-function g, and given prior π and channel C, the *multiplicative* and the *additive* versions of g-leakage are defined, respectively, as

$$\mathcal{L}_g[\pi \rangle C] = {}^{V_g[\pi \rangle C]}/_{V_g[\pi]}, \qquad \text{and} \qquad \mathcal{L}^+_g[\pi \rangle C] = V_g[\pi \rangle C] - V_g[\pi].$$

Since prior vulnerability does not depend on the channel, we have that

$$\mathcal{L}_g[\pi \rangle C_1] \geq \mathcal{L}_g[\pi \rangle C_2] \;\Leftrightarrow\; \mathcal{L}^+_g[\pi \rangle C_1] \geq \mathcal{L}^+_g[\pi \rangle C_2] \;\Leftrightarrow\; V_g[\pi \rangle C_1] \geq V_g[\pi \rangle C_2],$$

and, hence, the posterior vulnerability of a channel is greater than that of another if, and only if, both multiplicative and additive leakage also are.

Channel Ordering and the Coriaceous Theorem. We now define a common composition of channels, called *cascading*. This operation can be interpreted as the result of a channel post-processing the output of another channel. Formally, given two channels $C{:}\mathcal{X}\times\mathcal{Y}\to\mathbb{R}$ and $D{:}\mathcal{Y}\times\mathcal{Z}\to\mathbb{R}$, their cascading is defined as

$$(CD)(x,z) = \sum_{y\in\mathcal{Y}} C(x,y)D(y,z),$$

for all $x\in\mathcal{X}$ and $z\in\mathcal{Z}$. If we represent channels as tables, as we did in Example 1, the cascading operation corresponds to a simple matrix multiplication.

An important question in QIF is to decide whether a channel C_2 is always *at least as secure as* a channel C_1, meaning that C_2 never leaks more information than C_1, for whatever choice of gain function g and of prior π. Let us write $C_1 \sqsubseteq_\circ C_2$ (read as C_2 *refines* C_1) to denote that there exists a channel D such that $C_1D = C_2$. We write $C_1 \approx C_2$, and say that C_1 is *equivalent* to C_2, when both $C_1 \sqsubseteq_\circ C_2$ and $C_2 \sqsubseteq_\circ C_1$ hold. The *Coriaceous Theorem* [1,2] states that, $C_1 \sqsubseteq_\circ C_2$ if, and only if, $V_g[\pi \rangle C_1] \geq V_g[\pi \rangle C_2]$ for all π, g. This result reduces the comparison of channel security to a simple algebraic test.

The *refinement relation* $\sqsubseteq_\circ$ is a preorder on the set of all channels having the same input set. This preorder can be made into a partial order by using *abstract channels* [2], an equivalence relation that equates all channels presenting same leakage for all priors and gain functions. This partial order coincides with how much information channels leak, being the least secure channel (i.e., the "most leaky" one) at its bottom, and the most secure (i.e., the "least leaky") at its top.

3 Operators on channel composition

We shall say that two channels are *compatible* if they have the same input set. Given a set $\mathcal{X}$, we denote by $\mathcal{C}_\mathcal{X}$ the set of all channels that have $\mathcal{X}$ as input set. Two compatible channels with same output set are said to be of the *same type*.

In this section we introduce several *binary operators*—i.e., functions of type $(\mathcal{C}_\mathcal{X}\times\mathcal{C}_\mathcal{X})\to\mathcal{C}_\mathcal{X}$—matching typical interactions between system components, and prove relevant algebraic properties of these operators. We refer to the result of an operator as a *compound system*, and we refer to its arguments as *components*.

3.1 The parallel composition operator $\parallel$

The *parallel composition operator* $\parallel$ models the composition of two independent channels in which the same input is fed to both of them, and their outputs are then observed. By *independent*, we mean that the output of one channel does not interfere with that of the other. This assumption, while not universal, captures a great variety of real-world scenarios, and is, hence, of practical interest.

For example, *side-channel attacks* occur when the adversary combines his observation of the system's output with some alternative way of inferring information about the secret (e.g., by observing physical properties of the system execution, such as time elapsed [18,19] or change in magnetic fields [20]). In such attacks, the channel used by the adversary to infer information about the secret can be modeled as the composition of a channel representing the program's intended behaviour in parallel with a channel modeling the relation between the secret and the physical properties of the hardware.

Definition 1 (Parallel composition operator $\parallel$). *Given compatible channels* $C_1{:}\mathcal{X}\times\mathcal{Y}_1\to\mathbb{R}$ *and* $C_2{:}\mathcal{X}\times\mathcal{Y}_2\to\mathbb{R}$*, their* parallel composition $C_1 \parallel C_2 : \mathcal{X}\times(\mathcal{Y}_1\times\mathcal{Y}_2)\to\mathbb{R}$ *is defined as, for all* $x\in\mathcal{X}$*,* $y_1\in\mathcal{Y}_1$*, and* $y_2\in\mathcal{Y}_2$*,*

$$(C_1 \parallel C_2)(x,(y_1,y_2)) = C_1(x,y_1)C_2(x,y_2).$$

Notice that this definition comes from the independence property, as we have $C_1(x,y_1)C_2(x,y_2) = p(y_1|x)p(y_2|x) = p(y_1,y_2|x)$.

3.2 The visible choice operator ${}_p\sqcup$

The *visible choice operator* ${}_p\sqcup$ models a scenario in which the system has a choice among two different components to process the secret it was fed as input. With probability p, the system feeds the secret to the first component, and, with probability $1-p$, it feeds the secret to the second component. In the end, the system reveals the output produced, together with the identification of which component was used (whence, the name "visible choice").

As an example, consider an adversary trying to gain information about a secret processed by a website. The adversary knows that the website has two servers, one of which will be assigned to answer the request according to a

known probability distribution. Suppose, furthermore, that the adversary can identify which server was used by measuring its response time to the request. This adversary's view of the system can be modeled as the visible choice between the two servers, since, although the adversary does not know in advance which server will be used, he learns it when he gets the output from the system.

Before formalizing this operator, we need to define the *disjoint union* of sets. Given any sets $\mathcal{A}$ and $\mathcal{B}$, their disjoint union is $\mathcal{A} \sqcup \mathcal{B} = (\mathcal{A} \times \{1\}) \cup (\mathcal{B} \times \{2\})$.

Definition 2 (Visible choice operator $_p\sqcup$). *Given compatible channels $C_1 : \mathcal{X}\times\mathcal{Y}_1\to\mathbb{R}$ and $C_2{:}\mathcal{X}\times\mathcal{Y}_2\to\mathbb{R}$, their* visible choice *is the channel $C_1 \ {}_p\!\sqcup\ C_2 : \mathcal{X}\times(\mathcal{Y}_1 \sqcup \mathcal{Y}_2)\to\mathbb{R}$ defined as, for all $x\in\mathcal{X}$ and $(y, i) \in \mathcal{Y}_1 \sqcup \mathcal{Y}_2$,*

$$(C_1 \ {}_p\!\sqcup\ C_2)(x, (y, i)) = \begin{cases} pC_1(x, y), & \text{if } i = 1, \\ (1{-}p)C_2(x, y), & \text{if } i = 2. \end{cases}$$

3.3 The hidden choice operator $_p\oplus$

Similarly to the visible choice case, the *hidden choice operator* $_p\oplus$ models a scenario in which the system has a choice of feeding its secret input to one component (with probability p), or to another component (with probability $1{-}p$). In the end, the system reveals the output produced, but, unlike the visible choice case, the component which was used is not revealed. Hence, when the same observations are randomized between the two channels, the adversary cannot identify which channel produced the observation (whence, the name "hidden choice").

As an example, consider statistical surveys that ask some sensitive yes/no question, such as whether the respondent has made use of any illegal substances. To encourage individuals to participate on the survey, it is necessary to control leakage of their sensitive information, while preserving the accuracy of statistical information in the ensemble of their answers. A common protocol to achieve this goal works as follows [21]. Each respondent throws a coin, without letting the questioner know the corresponding result. If the result is heads, the respondent answers the question honestly, and if the result is tails, he gives a random response (obtained, for example, according to the result of a second coin toss). If the coins are fair, this protocol can be modeled as the hidden choice $T \ {}_{1/2}\!\oplus\ C$ between a channel T representing an honest response (revealing the secret completely), and a channel C representing a random response (revealing nothing about the secret). The protocol is, hence, a channel that masks the result of T.

Definition 3 (Hidden choice operator $_p\oplus$). *Given compatible channels $C_1 : \mathcal{X}\times\mathcal{Y}_1\to\mathbb{R}$ and $C_2{:}\mathcal{X}\times\mathcal{Y}_2\to\mathbb{R}$, their* hidden choice *is the channel $C_1 \ {}_p\!\oplus\ C_2 : \mathcal{X}\times(\mathcal{Y}_1 \cup \mathcal{Y}_2)\to\mathbb{R}$ defined as, for all $x\in\mathcal{X}$ and $y\in\mathcal{Y}_1 \cup \mathcal{Y}_2$,*

$$(C_1 \ {}_p\!\oplus\ C_2)(x, y) = \begin{cases} pC_1(x, y) + (1{-}p)C_2(x, y), & \text{if } y \in \mathcal{Y}_1 \cap \mathcal{Y}_2, \\ pC_1(x, y), & \text{if } y \in \mathcal{Y}_1 \setminus \mathcal{Y}_2, \\ (1{-}p)C_2(x, y), & \text{if } y \in \mathcal{Y}_2 \setminus \mathcal{Y}_1. \end{cases}$$

Note that when the output sets of C_1 and C_2 are disjoint the adversary can always identify the channel used, and we have $C_1 \ {}_p\!\sqcup\ C_2 \approx C_1 \ {}_p\!\oplus\ C_2$.

3.4 A compositional description of the Dining Cryptographers

We now revisit the Dining Cryptographers protocol example from Section 1, showing how it can be modeled using our composition operators.

We consider that there are 4 cryptographers and 4 coins, and denote the protocol's channel by *Dining*. The channel's input set is $\mathcal{X} = \{c_1, c_2, c_3, c_4, n\}$, in which c_i represents that cryptographer i is the payer, and n represents that the NSA is the payer. The channel's output set is $\mathcal{Y} = \{0,1\}^4$, i.e., all 4-tuples representing possible announcements by all cryptographers, in order.

Following the scheme in Figure 1 (middle), we begin by modeling the protocol as the interaction between two channels, *Coins* and *Announcements*, representing, respectively, the coin tosses and the cryptographers' public announcements. Since in the protocol first the coins are tossed, and only then the corresponding results are passed on to the party of cryptographers, *Dining* can be described as the cascading of these two channels:

$$Dining = (Coins)(Announcements).$$

To specify channel *Coins*, we use the parallel composition of channels $Coin_1$, $Coin_2$, $Coin_3$ and $Coin_4$, each representing one coin toss. Letting p_i denote the probability of coin i landing on tails, these channels are defined as on Table 1.

$Coin_i$	Tails	Heads
c_1	p_i	$1-p_i$
c_2	p_i	$1-p_i$
c_3	p_i	$1-p_i$
c_4	p_i	$1-p_i$
n	p_i	$1-p_i$

Table 1. Channel representing toss of coin $Coin_i$.

Besides the result of the tosses, *Coins* also needs to pass on to *Announcements* the identity of the payer. We then introduce a fifth channel, $I{:}\mathcal{X}\times\mathcal{X}\to\mathbb{R}$, that simply outputs the secret, i.e., $I(x_1, x_2) = 1$ if $x_1 = x_2$, and 0 otherwise. Hence, a complete definition of channel *Coins* is

$$Coins = Coin_1 \parallel Coin_2 \parallel Coin_3 \parallel Coin_4 \parallel I.$$

As we will show in Section 4, parallel composition is associative, allowing us to omit parentheses in the equation above.

We now specify the channel *Announcements*, which takes as input a 5-tuple with five terms whose first four elements are the results of the coin tosses, and the fifth is the identity of the payer. For that end, we describe each cryptographer as a channel with this 5-tuple as input, and with the set of possible announcements $\{0,1\}$ as output set. $Crypto_1$ below describes the first cryptographer.

$$Crypto_1(t_1, t_2, t_3, t_4, x) = \begin{cases} 1, \text{ if } t_4 = t_1 \text{ and } x = c_1, \text{ or } t_4 \neq t_1 \text{ and } x \neq c_1 \\ 0, \text{ otherwise} \end{cases}$$

Channels $Crypto_2$, $Crypto_3$ and $Crypto_4$ describing the remaining cryptographers are defined analogously. Channel *Announcements* is, hence, defined as

$$Announcements = Crypto_1 \parallel Crypto_2 \parallel Crypto_3 \parallel Crypto_4.$$

Note that our operators allow for an intuitive and succinct representation of the channel *Dining* modeling the Dining Cryptographers protocol, even when the number of cryptographers and coins is large. Moreover, the channel is easy to compute: we need only to first calculate the parallel compositions within channels *Crypto* and *Announcements*, and then multiply these channels' matrices.

4 Algebraic properties of channel operators

In this section we prove a series of relevant algebraic properties of our channel operators. These properties are the key for building channels in a compositional way, and, more importantly, for deriving information flow properties of a compound system in terms of those of its components.

We begin by defining a notion of equivalence stricter than $\approx$, which equates any two channels that are identical modulo a permutation of their columns.

Definition 4 (Channel equality). *Let $C_1 : \mathcal{X}\times\mathcal{Y}_1\to\mathbb{R}$ and $C_2 : \mathcal{X}\times\mathcal{Y}_2\to\mathbb{R}$ be compatible channels. We say that C_1 and C_2 are* equal up to a permutation *, and write $C_1 \overset{\circ}{=} C_2$, if there is a bijection $\psi{:}\mathcal{Y}_1\to\mathcal{Y}_2$ such that $C_1(x,y){=}C_2(x,\psi(y))$ for all $x{\in}\mathcal{X}$, $y{\in}\mathcal{Y}_1$.*

Note that, if $C_1 \overset{\circ}{=} C_2$, then $C_1{\approx}C_2$. [3]

In remaining of this section, let $C_1 : \mathcal{X}\times\mathcal{Y}_1\to\mathbb{R}$, $C_2 : \mathcal{X}\times\mathcal{Y}_2\to\mathbb{R}$ and $C_3 : \mathcal{X}\times\mathcal{Y}_3\to\mathbb{R}$ be compatible channels, and $p, q \in [0,1]$ be probability values.

4.1 Properties regarding channel operators

We first establish our operators' associativity and commutativity properties.

Proposition 1 (Commutative Properties).

$$C_1 \| C_2 \overset{\circ}{=} C_2 \| C_1, \quad C_1 \ {}_p\!\sqcup\!\!\sqcup\ C_2 \overset{\circ}{=} C_2 \ {}_{(1-p)}\!\sqcup\!\!\sqcup\ C_1, \text{ and } \ C_1 \ {}_p\!\oplus\ C_2 = C_2 \ {}_{(1-p)}\!\oplus\ C_1.$$

Proposition 2 (Associative Properties).

$$(C_1 \parallel C_2) \parallel C_3 \overset{\circ}{=} C_1 \parallel (C_2 \parallel C_3), \qquad (C_1 \ {}_p\!\sqcup\!\!\sqcup\ C_2) \ {}_q\!\sqcup\!\!\sqcup\ C_3 \overset{\circ}{=} C_1 \ {}_{p'}\!\sqcup\!\!\sqcup\ (C_2 \ {}_{q'}\!\sqcup\!\!\sqcup\ C_3),$$

$$\text{and } \ (C_1 \ {}_p\!\oplus\ C_2) \ {}_q\!\oplus\ C_3 = C_1 \ {}_{p'}\!\oplus\ (C_2 \ {}_{q'}\!\oplus\ C_3), \ \text{ s.t. } p'{=}pq \text{ and } q'{=}{}^{(q-pq)}\!/_{(1-pq)}.$$

We now turn our attention to two kinds of channels that will be recurrent building blocks for more complex channels. A *null channel* is any channel $\overline{0} : \mathcal{X}\times\mathcal{Y}\to\mathbb{R}$ such that, for every prior π and gain-function g, $V_g[\pi \rangle \overline{0}] = V_g[\pi]$. That is, a null channel never leaks any information. A channel $\overline{0}$ is null if, and only if, $\overline{0}(x,y) = \overline{0}(x',y)$ for all $y{\in}\mathcal{Y}$ and $x, x'{\in}\mathcal{X}$. On the other hand, a *transparent channel* is any channel $\overline{I} : \mathcal{X}\times\mathcal{Y}\to\mathbb{R}$ that leaks at least as much information as

[3] A complete, formal definition of such bijections can be found in an accompanying technical report[14].

any other compatible channel, for every prior and gain-function. A channel $\overline{I}$ is transparent if, and only if, for each $y \in \mathcal{Y}$, there is at most one $x \in \mathcal{X}$ such that $\overline{I}(x,y) > 0$. The following properties hold for any null channel $\overline{0}$ and transparent channel $\overline{I}$ compatible with C_1, C_2 and C_3.

Proposition 3 (Null and Transparent Channel Properties).

$$\begin{array}{rl} \textit{null channel:} & (C_1 \parallel \overline{0}) \approx C_1, \quad C_1 \sqsubseteq_\circ (C_1 \ {}_p\!\sqcup \overline{0}), \quad C_1 \sqsubseteq_\circ (C_1 \ {}_p\!\oplus \overline{0}). \\ \textit{transparent channel:} & (C_1 \parallel \overline{I}) \approx \overline{I}, \qquad (C_1 \ {}_p\!\sqcup \overline{I}) \sqsubseteq_\circ C_1. \end{array}$$

Note that, in general, $(C_1 \ {}_p\!\oplus \overline{I}) \not\sqsubseteq_\circ C_1$. To see why, consider the two transparent channels $\overline{I}_1$ and $\overline{I}_2$, with both input and output sets equal $\{1,2\}$, given by $\overline{I}_1(x,x') = 1$ if $x{=}x'$, and 0 otherwise, and $\overline{I}_2(x,x') = 0$ if $x{=}x'$, and 1 otherwise Then, $\overline{I}_1 \ {}_p\!\oplus \overline{I}_2$ is a null channel, and the property does not hold for $C_1 = \overline{I}_1$, $\overline{I} = \overline{I}_2$.

We now consider idemptotency.

Proposition 4 (Idempotency).

$$C_1 \parallel C_1 \sqsubseteq_\circ C_1, \qquad C_1 \ {}_p\!\sqcup C_1 \approx C_1, \qquad \textit{and} \qquad C_1 \ {}_p\!\oplus C_1 = C_1.$$

Note that $C_1 \| C_1 \approx C_1$ holds only when C_1 is deterministic or equivalent to a deterministic channel.

Finally, we consider distributive properties. In particular, we explore interesting properties when an operator is "distributed" over itself.

Proposition 5 (Distribution over the same operator).

$$\begin{array}{c} (C_1 \parallel C_2) \parallel (C_1 \parallel C_3) \sqsubseteq_\circ C_1 \parallel (C_2 \parallel C_3), \\ C_1 \ {}_p\!\sqcup (C_2 \ {}_q\!\sqcup C_3) \approx (C_1 \ {}_p\!\sqcup C_2) \ {}_q\!\sqcup (C_1 \ {}_p\!\sqcup C_3), \\ C_1 \ {}_p\!\oplus (C_2 \ {}_q\!\oplus C_3) = (C_1 \ {}_p\!\oplus C_2) \ {}_q\!\oplus (C_1 \ {}_p\!\oplus C_3). \end{array}$$

Proposition 6 (Distribution over different operators).

$$\begin{array}{c} C_1 \parallel (C_2 \ {}_p\!\sqcup C_3) \stackrel{\circ}{=} (C_1 \parallel C_2) \ {}_p\!\sqcup (C_1 \parallel C_3), \\ C_1 \parallel (C_2 \ {}_p\!\oplus C_3) = (C_1 \parallel C_2) \ {}_p\!\oplus (C_1 \parallel C_3), \\ C_1 \ {}_p\!\sqcup (C_2 \ {}_q\!\oplus C_3) = (C_1 \ {}_p\!\sqcup C_2) \ {}_q\!\oplus (C_1 \ {}_p\!\sqcup C_3). \end{array}$$

Unfortunately, the distribution of ${}_p\!\sqcup$ over $\parallel$, ${}_p\!\oplus$ over $\parallel$, or ${}_p\!\oplus$ over ${}_p\!\sqcup$ is not as well behaved. A complete discussion is avaiable in the technical report[14]

4.2 Properties regarding cascading

We conclude this section by exploring how our operators behave w.r.t. cascading (defined in Section 2). Cascading of channels is fundamental in QIF, as it captures the concept of a system's *post-processing* of another system's outputs, and it is also the key to the partial order on channels discussed in Section 2.

The next propositions explore whether it is possible to express a composition of two post-processed channels by a post-processing of their composition.

Proposition 7. *Let $D_1 : \mathcal{Y}_1 \times \mathcal{Z}_1 \to \mathbb{R}$, $D_2 : \mathcal{Y}_2 \times \mathcal{Z}_2 \to \mathbb{R}$ be channels. Then,*

$$(C_1 D_1) \parallel (C_2 D_2) = (C_1 \parallel C_2) D^{\parallel},$$

where $D^{\parallel} : (\mathcal{Y}_1 \times \mathcal{Y}_2) \times (\mathcal{Z}_1 \times \mathcal{Z}_2) \to \mathbb{R}$ is defined, for all $y_1 \in \mathcal{Y}_1$, $y_2 \in \mathcal{Y}_2$, $z_1 \in \mathcal{Z}_1$, and $z_2 \in \mathcal{Z}_2$, as $D^{\parallel}((y_1, y_2), (z_1, z_2)) = D_1(y_1, z_1) D_2(y_2, z_2)$.

Proposition 8. *Let $D_1 : \mathcal{Y}_1 \times \mathcal{Z}_1 \to \mathbb{R}$, $D_2 : \mathcal{Y}_2 \times \mathcal{Z}_2 \to \mathbb{R}$ be channels. Then,*

$$(C_1 D_1) \ {}_p{\sqcup\!\!\cdot} \ (C_2 D_2) = (C_1 \ {}_p{\sqcup\!\!\cdot} \ C_2) D^{\sqcup\!\!\cdot},$$

where $D^{\sqcup\!\!\cdot} : (\mathcal{Y}_1 \sqcup \mathcal{Y}_2) \times (\mathcal{Z}_1 \sqcup \mathcal{Z}_2) \to \mathbb{R}$ is defined as $D^{\sqcup\!\!\cdot}((y, i), (z, j)) = D_1(y, z)$ if $i = j = 1$, or $D_2(y, z)$ if $i = j = 2$, or 0 otherwise, for all $y_1 \in \mathcal{Y}_1$, $y_2 \in \mathcal{Y}_2$, $z_1 \in \mathcal{Z}_1$, $z_2 \in \mathcal{Z}_2$.

A similar rule, however, does not hold for hidden choice. For example, let C_1 and C_2 be channels with input and output sets $\{1, 2\}$, such that $C_1(x, x') = 1$ if $x = x'$, or 0 otherwise, and $C_2(x, x') = 0$ if $x = x'$, or 1 otherwise. Let D_1 and D_2 be transparent channels whose output sets are disjoint. Then, $(C_1 D_1) \ {}_{1/2}\oplus \ (C_2 D_2)$ is a transparent channel, but $C_1 \ {}_{1/2}\oplus \ C_2$ is a null channel. Thus, it is impossible to describe $(C_1 D_1) \ {}_{1/2}\oplus \ (C_2 D_2)$ as $C_1 \ {}_{1/2}\oplus \ C_2$ post-processed by some channel. However, we can establish a less general, yet relevant, equivalence.

Proposition 9. *Let $C_1 : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ and $C_2 : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ be channels of the same type. Let $D : \mathcal{Y} \times \mathcal{Z} \to \mathbb{R}$ be a channel. Then, $(C_1 D) \ {}_p\oplus \ (C_2 D) = (C_1 \ {}_p\oplus \ C_2) D$.*

5 Information leakage of channel operators

This section presents the main contribution of our paper: a series of results showing how, using the proposed algebra, we can facilitate the security analysis of compound systems. Our results are given in terms of the g-leakage framework introduced in Section 2, and we focus on two central problems. For the remaining of the section, let $C_1 : \mathcal{X} \times \mathcal{Y}_1 \to \mathbb{R}$ and $C_2 : \mathcal{X} \times \mathcal{Y}_2 \to \mathbb{R}$ be compatible channels.

5.1 The problem of compositional vulnerability

The first problem consists in estimating the information leakage of a compound system in terms of the leakage of its components. This is formalized as follows.

The problem of compositional vulnerability: *Given a composition operator $*$ on channels, a prior $\pi \in \mathbb{D}\mathcal{X}$, and a gain function g, how can we estimate $V_g[\pi \rangle C_1 * C_2]$ in terms of $V_g[\pi \rangle C_1]$ and $V_g[\pi \rangle C_2]$?*

Theorem 1 (Upper and lower bounds for V_g w.r.t. $\parallel$). *For all gain functions g and $\pi \in \mathbb{D}\mathcal{X}$, let $\mathcal{X}' = \{x \in \mathcal{X} \mid \exists w \in \mathcal{W} \text{ s.t. } \pi(x) g(w, x) > 0\}$. Then*

$$V_g[\pi \rangle C_1 \| C_2] \geq \max(V_g[\pi \rangle C_1], V_g[\pi \rangle C_2]), \qquad \text{and}$$

$$V_g[\pi \rangle C_1 \| C_2] \leq \min\left(V_g[\pi \rangle C_1] \sum_{y_2} \max_{x \in \mathcal{X}'} C_2(x, y_2), V_g[\pi \rangle C_2] \sum_{y_1} \max_{x \in \mathcal{X}'} C_1(x, y_1) \right).$$

Theorem 2 (Linearity of V_g w.r.t. ${}_p\sqcup$). *For all gain functions g, $\pi \in \mathbb{D}\mathcal{X}$ and $p \in [0,1]$,*

$$V_g[\pi \rangle C_1 \ {}_p\sqcup \ C_2] = pV_g[\pi \rangle C_1] + (1-p)V_g[\pi \rangle C_2].$$

Theorem 3 (Upper and lower bounds for V_g w.r.t. ${}_p\oplus$). *For all gain functions g, $\pi \in \mathbb{D}\mathcal{X}$ and $p \in [0,1]$,*

$$\begin{aligned} &V_g[\pi \rangle C_1 \ {}_p\oplus \ C_2] \geq \max(pV_g[\pi \rangle C_1], (1-p)V_g[\pi \rangle C_2]), \qquad \textit{and} \\ &V_g[\pi \rangle C_1 \ {}_p\oplus \ C_2] \leq pV_g[\pi \rangle C_1] + (1-p)V_g[\pi \rangle C_2]. \end{aligned}$$

The three theorems above yield an interesting order between the operators

Corollary 1 (Ordering between operators). *Let $\pi \in \mathbb{D}\mathcal{X}$, g be a gain function and $p \in [0,1]$. Then $V_g[\pi \rangle C_1 \parallel C_2] \geq V_g[\pi \rangle C_1 \ {}_p\sqcup \ C_2] \geq V_g[\pi \rangle C_1 \ {}_p\oplus \ C_2]$.*

5.2 The problem of relative monotonicity

The second problem concerns establishing whether a component channel of a larger system can be safely substituted with another component, i.e., whether substituting a component with another can cause an increase in the information leakage of the system as a whole. This is formalized as follows.

The problem of relative monotonicity: *Given a composition operator $*$ on channels, a prior $\pi \in \mathbb{D}\mathcal{X}$, and a gain function g, is it the case that $V_g[\pi \rangle C_1] \leq V_g[\pi \rangle C_2] \Leftrightarrow \forall C \in \mathcal{C}_{\mathcal{X}}.\ V_g[\pi \rangle C_1 * C] \leq V_g[\pi \rangle C_2 * C]$?*

We start by showing that relative monotonicity holds for visible choice. Note, however, that because $V_g[\pi \rangle C_1 \ {}_p\sqcup \ C] \leq V_g[\pi \rangle C_2 \ {}_p\sqcup \ C]$ is vacuously true if $p = 0$, we consider only $p \in (0,1]$.

Theorem 4 (Relative monotonicity for ${}_p\sqcup$). *For all gain functions g, $\pi \in \mathbb{D}\mathcal{X}$ and $p \in (0,1]$,*

$$V_g[\pi \rangle C_1] \leq V_g[\pi \rangle C_2] \Leftrightarrow \forall C.\ V_g[\pi \rangle C_1 \ {}_p\sqcup \ C] \leq V_g[\pi \rangle C_2 \ {}_p\sqcup \ C].$$

Interestingly, relative monotonicity does not hold for the parallel operator. This means that the fact that a channel C_1 is always more secure than a channel C_2 does not guarantee that if we replace C_1 for C_2 in a parallel context we necessarily obtain a more secure system.[4] However, when the adversary's knowledge (represented by the prior π) or preferences (represented by the gain-function g) are known, we can obtain a constrained result on leakage by fixing only π or g.

[4] As a counter-example, consider channels $C_1 = \left(\begin{smallmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{smallmatrix}\right)$ and $C_2 = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{smallmatrix}\right)$. Let $\pi_u = \{1/3, 1/3, 1/3\}$ and $g_{id} : \mathcal{X} \times \mathcal{X} \to [0,1]$ s.t. $g_{id}(x_1, x_2) = 1$ if $x_1 = x_2$ and 0 otherwise. Then, $V_{g_{id}}[\pi_u \rangle C_2] \leq V_{g_{id}}[\pi_u \rangle C_1]$, but $V_{g_{id}}[\pi_u \rangle C_2 \parallel C_1] > V_{g_{id}}[\pi_u \rangle C_1 \parallel C_1]$.

Theorem 5 (Relative monotonicity for $\|$). *For all gain functions g and $\pi \in \mathbb{D}\mathcal{X}$*

$$\forall \pi'.\, V_g[\pi' \rangle C_1] \leq V_g[\pi' \rangle C_2] \Leftrightarrow \forall \pi', C.\, V_g[\pi' \rangle C_1 \parallel C] \leq V_g[\pi' \rangle C_2 \parallel C], \quad \textit{and}$$
$$\forall g'.\, V_{g'}[\pi \rangle C_1] \leq V_{g'}[\pi \rangle C_2] \Leftrightarrow \forall g', C.\, V_{g'}[\pi \rangle C_1 \parallel C] \leq V_{g'}[\pi \rangle C_2 \parallel C].$$

Perhaps surprisingly, hidden choice does not respect relative monotonicity, even when we only consider channels that respect the refinement relation introduced in section 2.

Theorem 6 (Relative monotonicity for ${}_p\oplus$). *For all $p \in (0,1)$, there are $C_1 : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ and $C_2 : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ such that*

$$\forall \pi, g.\, V_g[\pi \rangle C_1] \leq V_g[\pi \rangle C_2] \textit{ and } \exists \pi', g', C.\, V_{g'}[\pi' \rangle C_1 \;{}_p\oplus C] > V_{g'}[\pi' \rangle C_2 \;{}_p\oplus C],$$

The converse, however, is true.

Theorem 7 (Relative monotonicity for ${}_p\oplus$, cont.). *For all gain functions g, $\pi \in \mathbb{D}\mathcal{X}$ and $p \in (0,1]$,*

$$\forall C.\, V_g[\pi \rangle C_1 \;{}_p\oplus C] \leq V_g[\pi \rangle C_2 \;{}_p\oplus C] \Rightarrow V_g[\pi \rangle C_1] \leq V_g[\pi \rangle C_2].$$

6 Case study: the Crowds protocol

In this section we apply the theoretical techniques developed in this paper to the well-known *Crowds* anonymity protocol [13]. Crowds was designed to protect the identity of a group of *users* who wish to anonymously send requests to a *server*, and it is the basis of the widely used protocols *Onion Routing* [22] and *Tor* [23].

The protocol works as follows. When a user wants to send a request to the server, he first randomly picks another user in the group and forwards the request to that user. From that point on, each user, upon receiving a request from another user, sends it to the server with probability $p \in (0,1]$, or forwards it to another user with probability $1-p$. This second phase repeats until the message reaches the server.

It is assumed that the adversary controls the server and some *corrupt* users among the regular, *honest*, ones. When a corrupt user receives a forwarded request, he shares the forwarder's identity with the server, and we say that the forwarder was *detected*. As no information can be gained after a corrupt user intercepts a request, we need only consider the protocol's execution until a detection occurs, or the message reaches the server.

In Crowds' original description, all users have equal probability of being forwarded a message, regardless of the forwarder. The channel modeling such a case is easily computed, and well-known in the literature. Here we consider the more general case in which each user may employ a different probability distribution when choosing which user to forward a request to. Thus, we can capture scenarios in which not all users can easily reach each other (a common

problem in, for instance, *ad-hoc* networks). We make the simplifying assumption that corrupt users are evenly distributed, i.e., that all honest users have the same probability $q \in (0,1]$ of choosing a corrupt user to forward a request to.

We model Crowds as a channel $Crowds{:}\mathcal{X}\times\mathcal{Y}\to\mathbb{R}$. The channel's input, taken from set $\mathcal{X}=\{u_1, u_2, \ldots, u_{n_c}\}$, represents the identity u_i of the honest user (among a total of n_c honest users) who initiated the request. The channel's output is either the identity of a detect user—i.e., a value from $\mathcal{D}=\{d_1, d_2, \ldots, d_{n_c}\}$, where where d_i indicates user u_i was detected—or the identity of a user who forwarded the message to the server—i.e., a value from $\mathcal{S}=\{s_1, s_2, \ldots, s_{n_c}\}$, where s_i indicates user u_i forwarded a message to the server. Note that $\mathcal{D}$ and $\mathcal{S}$ are disjoint, and the channel's output set is $\mathcal{Y} = \mathcal{D} \cup \mathcal{S}$.

To compute the channel's entries, we model the protocol as a time-stationary Markov chain $M = (\mathcal{U}, \boldsymbol{P})$, where the set of states is the set of honest users $\mathcal{U}$, and its transition function is such that $\boldsymbol{P}(u_i, u_j)$ is the probability of u_j being the recipient of a request forwarded by u_i, given that u_i will not be detected.

We then define four auxiliary channels. Transparent channels $I_d{:}\mathcal{U}\times\mathcal{D}\to\mathbb{R}$ and $I_s{:}\mathcal{U}\times\mathcal{S}\to\mathbb{R}$ are defined as $I_d(u_i, d_j)=1$ if $i=j$, or 0 otherwise, and $I_s(u_i, s_j) = 1$ if $i=j$, or 0 otherwise; and two other channels $P_d{:}\mathcal{D}\times\mathcal{D}\to\mathbb{R}$ and $P_s{:}\mathcal{S}\times\mathcal{S}\to\mathbb{R}$, based on our Markov chain M, are defined as $P_d(d_i, d_j)=P_s(s_i, s_j)=\boldsymbol{P}(u_i, u_j)$.

We begin by reasoning about what happens if each request can be forwarded only once. There are two possible situations: either the initiator is detected, or he forwards the request to an honest user, who will in turn send it to the server. The channel corresponding to the initiator being detected is I_d, since in this case the output has to be d_i whenever u_i is the initiator. The channel corresponding to the latter situation is I_sP_s—i.e., the channel I_s postproccessed by P_s. This is because, being P_s based on the transition function of M, the entry $(I_sP_s)(u_i, s_j)$ gives us exactly the probability that user u_j received the request originated by user u_i after it being forwarded once. Therefore, when Crowds is limited to one forwarding, it can be modeled by the channel $I_d \;{}_q\!\oplus\; I_sP_s$ [5], representing the fact that: (1) with probability q the initiator is detected, and the output is generated by I_d; and (2) with probability $1 - q$ the output is generated by I_sP_s.

Let us now cap our protocol to at most two forwards. If the initiator is not immediately detected, the first recipient will have a probability p of sending the message to the server. If the recipient forwards the message instead, he may be detected. Because the request was already forwarded once, the channel that will produce the output in this case is I_dP_d (notice that, despite this channel being equivalent to I_sP_s, it is of a different type). On the other hand, if the first recipient forwards the message to an honest user, this second recipient will now send the message to the server, making the protocol produce an output according to $I_sP_sP_s$ (or simply $I_sP_s^2$), since $(I_sP_s^2)(u_i, s_j)$ is the probability that user u_j received the request originated by user u_i after it being forwarded twice. Therefore, when Crowds is limited to two forwardings, it can be modeled by the channel $I_d \;{}_q\!\oplus\; (I_sP_s \;{}_p\!\oplus\; (I_dP_d \;{}_q\!\oplus\; I_sP_s^2))$. Note the disposition of the parenthesis

[5] To simplify notation, we assume cascading has precedence over hidden choice, i.e., $AB \;{}_p\!\oplus\; CD = (AB) \;{}_p\!\oplus\; (CD)$.

reflects the order in which the events occur. First, there is a probability q of the initiator being detected, and $1-q$ of the protocol continuing. Then, there is a probability p of the first recipient sending it to the server, and so on.

Proceeding this way, we can inductively construct a sequence $\{C_i\}_{i\in\mathbb{N}^*}$,

$$C_i = I_d \ {}_q\oplus (I_s P_s \ {}_p\oplus (I_d P_d \ {}_q\oplus (\dots \ {}_p\oplus (I_d P_d^{i-1} \ {}_q\oplus I_s P_s^i)\dots))),$$

in which each C_i represents our protocol capped at i forwards per request. We can then obtain *Crowds* by taking $\lim_{i\to\infty} C_i$. From that, Theorem 3 and Proposition 2, we can derive the following bounds on the information leakage of Crowds.

Theorem 8. *Let $\{t_i\}_{i\in\mathbb{N}}$ be the sequence in which $t_{2i}=1-(1-q)^{i+1}(1-p)^i$ and $t_{(2i+1)} = 1-(1-q)^{i+1}(1-p)^{i+1}$ for all $i\in\mathbb{N}$.*

Let $K_m=((\dots(I_d \ {}_{t_0/t_1}\oplus I_s P_s) \ {}_{t_1/t_2}\oplus \dots) \ {}_{t_{2m-1}/t_{2m}}\oplus (I_d P_d^m)$. Then, $\forall m\in\mathbb{N}^*$,

$$V_g[\pi\rangle \lim_{i\to\infty} C_i] \ \geq \ t_{2m} V_g[\pi\rangle K_m], \tag{1}$$

$$V_g[\pi\rangle \lim_{i\to\infty} C_i] \ \leq \ t_{2m} V_g[\pi\rangle K_m] + (1-t_{2m}) V_g[\pi\rangle I_s P_s^{m+1}], \quad \textit{and} \tag{2}$$

$$(1-t_{2m}) V_g[\pi\rangle I_s P_s^{m+1}] \ \leq \ (1-q)^{m+1}(1-p)^m. \tag{3}$$

Equations (1) and (2) provide an effective way to approximate the g-leakage of information of the channel *Crowds* with arbitrary precision, whereas Equation (3) lets us estimate how many interactions are needed for that.

To obtain K_m, we need to calculate m matrix multiplications, which surpass the cost of computing the m hidden choices (which are only matrix additions). Thus, Theorem 8 implies we can obtain a channel whose posterior vulnerability differs from that of *Crowds* by at most $(1-q)^{m+1}(1-p^m)$ in $\approx O(mn_c^{2.807})$ time (using the Strassen algorithm for matrix multiplication [24]). Since p is typically high, $(1-q)^{m+1}(1-p)^m$ decreases very fast. For instance, for a precision of 0.001 on the leakage bound, we need m=10 when $(1-q)(1-p)$ is 0.5, m=20 when it is 0.7, and m=66 when it is 0.9, regardless of the number n_c of honest users.

Therefore, our method has time complexity $O(n_c^{2.807})$ when the number of users is large (which is the usual case for Crowds), and reasonable values of forward probability p, and precision. To the best of our knowledge this method is the fastest in the literature, beating the previous $O(n_c^{3.807})$ that can be achieved by modifying the method presented in [25]—although their method does not require our assumption of corrupt users being evenly distributed.

7 Related work

Compositionality is a fundamental notion in computer science, and it has been subject of growing interest in the QIF community.

Espinoza and Smith [26] derived a number of *min-capacity* bounds for different channel compositions, including cascading and parallel composition.

However, it was not until recently that compositionality results regarding the more general metrics of g-leakage started to be explored. Kawamoto et al.

[27] defined a generalization of the parallel operator for channels with different input sets, and gave upper bounds for the corresponding information leakage. Our bounds for compatible channels (Theorem 1) are tighter than theirs.

Recently, Engelhardt [28] defined the *mix operator*, another generalization of parallel composition, and derived results similar to ours regarding the parallel operator. Specifically, he provided commutative and associative properties (Propositions 1 and 2), and from his results the lower bound of Theorem 1 can be inferred. He also proved properties similar to the ones in Proposition 3, albeit using more restrictive definitions of null and transparent channels.

Both Kawamoto et al. and Engelhardt provided results similar to Theorem 5, but ours is not restricted to when one channel is refined by the other.

Just recently, Alvim et. al investigated algebraic properties of hidden and visible choice operators in the context of game-theoretic aspects of QIF [29], and derived the upper bounds of Theorems 2 and 3. Here we expanded the algebra to the interaction among operators, including parallel composition, derived more comprehensive bounds on their leakage, and applied our results to the Crowds protocol.

8 Conclusions and future work

In this paper we proposed an algebra to express numerous component compositions in systems that arise from typical ways in components interact in practical scenarios. We provided fundamental algebraic properties of these operators, and studied several of their leakage properties. In particular, we obtained new results regarding their motonicity properties and stricter bounds for the parallel and hidden choice operators. These results are of practical interest for the QIF community, as they provide helpful tools for modeling large systems and analyzing their security properties.

The list of operators we explored in this paper, however, does not seem to capture every possible interaction of components of real systems. As future work we wish to find other operators and increase the expressiveness of our approach.

Acknowledgments Arthur Américo and Mário S. Alvim are supported by CNPq, CAPES, and FAPEMIG. Annabelle McIver is supported by ARC grant DP140101119.

References

1. Alvim, M.S., Chatzikokolakis, K., Palamidessi, C., Smith, G.: Measuring information leakage using generalized gain functions. In: Proc. of CSF. (2012) 265–279
2. McIver, A., Morgan, C., Smith, G., Espinoza, B., Meinicke, L.: Abstract channels and their robust information-leakage ordering. In: Proc. of POST. Volume 8414 of LNCS., Springer (2014) 83–102
3. Clark, D., Hunt, S., Malacaria, P.: Quantitative information flow, relations and polymorphic types. J. of Logic and Computation **18**(2) (2005) 181–199
4. Köpf, B., Basin, D.A.: An information-theoretic model for adaptive side-channel attacks. In: Proc. of CCS, ACM (2007) 286–296

5. Chatzikokolakis, K., Palamidessi, C., Panangaden, P.: On the Bayes risk in information-hiding protocols. J. of Comp. Security **16**(5) (2008) 531–571
6. Smith, G.: On the foundations of quantitative information flow. In: Proc. of FOSSACS. Volume 5504 of LNCS., Springer (2009) 288–302
7. McIver, A., Meinicke, L., Morgan, C.: Compositional closure for bayes risk in probabilistic noninterference. In: Proc. of ICALP. Volume 6199 of LNCS., Springer (2010) 223–235
8. Boreale, M., Pampaloni, F.: Quantitative information flow under generic leakage functions and adaptive adversaries. Logical Methods in Computer Science **11**(4) (2015)
9. Alvim, M.S., Chatzikokolakis, K., McIver, A., Morgan, C., Palamidessi, C., Smith, G.: Axioms for information leakage. In: Proc. of CSF. (2016) 77–92
10. Chaum, D.: The dining cryptographers problem: Unconditional sender and recipient untraceability. Journal of Cryptology **1**(1) (1988) 65–75
11. Alvim, M.S., Chatzikokolakis, K., McIver, A., Morgan, C., Palamidessi, C., Smith, G.: Additive and multiplicative notions of leakage, and their capacities. In: Proc. of CSF, IEEE (2014) 308–322
12. Alvim, M.S., Chatzikokolakis, K., McIver, A., Morgan, C., Palamidessi, C., Smith, G.: Axioms for information leakage. In: Proc. of CSF. (2016) 77–92
13. Reiter, M.K., Rubin, A.D.: Crowds: anonymity for Web transactions. ACM Trans. on Information and System Security **1**(1) (1998) 66–92
14. Américo, A., Alvim, M.S., McIver, A.: An algebraic approach for reasoning about information flow. CoRR **abs/1801.08090** (2018)
15. Shannon, C.E.: A mathematical theory of communication. Bell System Technical Journal **27** (1948) 379–423, 625–56
16. Massey: Guessing and entropy. In: Proceedings of the IEEE Int. Symposium on Information Theory, IEEE (1994) 204
17. Braun, C., Chatzikokolakis, K., Palamidessi, C.: Quantitative notions of leakage for one-try attacks. In: Proc. of MFPS. Volume 249 of ENTCS., Elsevier (2009) 75–91
18. Kocher, P.C.: Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In Koblitz, N., ed.: Advances in Cryptology — CRYPTO '96, Berlin, Heidelberg, Springer Berlin Heidelberg (1996) 104–113
19. Brumley, D., Boneh, D.: Remote timing attacks are practical. In: Proceedings of the 12th Conference on USENIX Security Symposium - Volume 12. SSYM'03, Berkeley, CA, USA, USENIX Association (2003) 1–1
20. Nohl, K., Evans, D., Starbug, S., Plötz, H.: Reverse-engineering a cryptographic rfid tag. In: Proceedings of the 17th Conference on Security Symposium. SS'08, Berkeley, CA, USA, USENIX Association (2008) 185–193
21. Warner, S.L.: Randomized response: A survey technique for eliminating evasive answer bias. Journal of the American Statistical Association **60**(309) (1965) 63–69 PMID: 12261830.
22. Goldschlag, D.M., Reed, M.G., Syverson, P.F.: Hiding routing information. In Anderson, R., ed.: Information Hiding, Berlin, Heidelberg, Springer Berlin Heidelberg (1996) 137–150
23. Dingledine, R., Mathewson, N., Syverson, P.F.: Tor: The second-generation onion router. In: Proceedings of the 13th USENIX Security Symposium, USENIX (2004) 303–320
24. Strassen, V.: Gaussian elimination is not optimal. Numer. Math. **13**(4) (August 1969) 354–356

25. Andrés, M.E., Palamidessi, C., van Rossum, P., Smith, G.: Computing the leakage of information-hiding systems. In: Proc. of TACAS. Volume 6015 of LNCS., Springer (2010) 373–389
26. Espinoza, B., Smith, G.: Min-entropy as a resource. Inf. and Comp. **226** (2013) 57–75
27. Kawamoto, Y., Chatzikokolakis, K., Palamidessi, C.: On the Compositionality of Quantitative Information Flow. Logical Methods in Computer Science **Volume 13, Issue 3** (August 2017)
28. Engelhardt, K.: A better composition operator for quantitative information flow analyses. In: European Symposium on Research in Computer Security, Proceedings, Part I. (2017) 446–463
29. Alvim, M.S., Chatzikokolakis, K., Kawamoto, Y., Palamidessi, C.: Leakage and protocol composition in a game-theoretic perspective. In: Proc. of POST