Recent Developments in Quantitative Information Flow

(Invited Tutorial)

Geoffrey Smith
School of Computing and Information Sciences
Florida International University
Miami, Florida USA
Email: smithg@cis.fiu.edu

***Abstract*—In computer security, it is frequently necessary in practice to accept some leakage of confidential information. This motivates the development of theories of *Quantitative Information Flow* aimed at showing that some leaks are "small" and therefore tolerable. We describe the fundamental view of channels as mappings from prior distributions on secrets to *hyper-distributions*, which are distributions on posterior distributions, and we show how g-leakage provides a rich family of operationally-significant measures of leakage. We also discuss two approaches to achieving robust judgments about leakage: notions of *capacity* and a robust leakage ordering called *composition refinement*.**

***Keywords*-security, confidentiality, information theory.**

I. Introduction

A fundamental and vexing problem in computer security is to control the leakage of sensitive information. In the past year, numerous large-scale compromises (e.g. at Home Depot, Target, and JPMorgan Chase) showed how just far we are from being able to solve this problem. When breaches of this kind occur, one important and natural response is to investigate the precise vulnerabilities that enabled the breach, and to seek ways to remedy them. But it appears more and more doubtful that the approach of patching vulnerabilities as they are discovered will *ever* lead us to a secure and trustworthy cyber-infrastructure. Fundamentally, achieving a real solution requires the development of a true science of information flow, a science that supports the construction and analysis of systems with precise information-flow guarantees.

Of course there are profound challenges in the development of such a science. A first issue concerns the modeling of systems. As Lynch and Fischer wrote long ago [1], "one would like simple mathematical models that exhibit the essential features of these systems while abstracting away irrelevant details". But what is *essential* and what is *irrelevant*? The history of computer security is full of "irrelevant details" that turned out to be essential—indeed, many low-level system details (e.g. timing, power consumption, caching behavior), which a mathematical model would naturally abstract away, have turned out to afford powerful *side channels* leaking sensitive information. Of course the modeling issue is not one that can be solved in generality; we always need to be mindful of the abstractions that we make and sensitive to the issues that we may be overlooking.

Another key issue is the information flow *policies* to be enforced. Early work in information flow focused on completely preventing leakage of sensitive information, requiring that the observable output of a system be independent of the secret input; this property is known as *noninterference* [2], [3]. Noninterfence is of course desirable whenever it can be achieved, and it has been found possible to guarantee it through the use of type systems (see, for example, [4], [5]). But it is more typical that noninterference is too strong, because *some* leakage of sensitive information is unavoidable in practice.

To illustrate, let us consider several motivating examples.

- A *password checker* stores a secret password, which must not be leaked, but when the checker rejects an incorrect guess it thereby reveals that the secret password differs from the guess.
- The *Crowds anonymity protocol* [6] enables anonymous communication with a server through randomized forwarding within the "crowd" of users. But if some crowd members are *collaborators* that report users that send messages to them, then some information about the initiator of the message is leaked.
- In a *statistical database*, the database entries may be considered to be confidential, but the results of statistical queries (e.g. average salary) are made public, revealing some information about the confidential entries. Noise may be added to query results in order to control the leakage, as in differential privacy [7].
- In typical implementations, the *time* required to do an RSA decryption depends on the RSA secret key, and these timing leaks can be used to recover the secret key [8]. Notice that this is a case where the choice of *system model* is crucial, since here we need a system model that includes running time as an observable output.

In each of these examples, noninterference is violated, yet we might sense intuitively that the "amount" of confidential information leaked is small, or perhaps can be made small through some mechanism. This perspective motivates the study of *Quantitative Information Flow* [*QIF*], an area that has seen growing interest over the past decade, including works with a foundational focus [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], works aimed at the development of verification techniques [23], [24], [25], [26], [27], [28], [29], [30], and works analyzing the leakage in real system vulnerabilities [31], [32], [33], [34].

As a first example, if X and Y are 32-bit integers, then we would intuitively expect that the C assignment statement

```
X = Y & 0x1ff;
```

should leak 9 bits out of 32, since the bitwise "and" operation `&` masks out the first 23 bits of X. But how should leakage be defined in general? And what security guarantees does leakage analysis offer?

The goal of this tutorial paper is to give a unified presentation of some recent developments in Quantitative Information Flow. It is organized as follows. Section II presents fundamental concepts, including channels as mappings from prior distributions to hyper-distributions, vulnerability, min-entropy leakage, and g-leakage. Section III then discusses the importance of *robust* judgments about leakage, Section IV discusses *capacity*, and Section V discusses robust channel ordering. Finally, Section VI discusses some future directions and concludes.

II. Fundamental Concepts of QIF

Given a confidential value X, its *secrecy* can be modeled by a *prior probability distribution* π on its set $\mathcal{X}$ of possible values, which we assume to be finite. For example, a uniform distribution gives X maximum secrecy, while a point distribution gives it no secrecy at all. We assume that π is known to the adversary $\mathcal{A}$, and it thus reflects the adversary's *prior knowledge* about X.

We remark that, philosophically, the distribution π is clear in the case when X is a randomly-generated string of bits. In contrast, if X is a secret like *my mother's maiden name*, then π can be seen as reflecting $\mathcal{A}$'s knowledge of the population I come from.

A. Channels are mappings from priors to hyper-distributions

The motivating examples mentioned in Section I can all be modeled as information-theoretic *channels*. A channel C that (perhaps probabilistically) maps secret input X to some observable output Y can be modeled as an information-theoretic *channel matrix* [35], whose rows show the distribution of outputs corresponding to each possible input. For example,

C	y_1	y_2	y_3	y_4
x_1	$1/2$	$1/2$	0	0
x_2	0	$1/4$	$1/2$	$1/4$
x_3	$1/2$	$1/3$	$1/6$	0

A significant special case is a *deterministic* channel, in which each row contains exactly one 1, with all other entries 0.

Assuming that the adversary $\mathcal{A}$ knows prior π and channel C, what does the channel's output Y reveal about X? It turns out that the effect of C is to map the prior distribution π to a *distribution on posterior distributions*, as we now illustrate with an example.

Given prior $\pi = (1/4, 1/2, 1/4)$, we form the *joint matrix* J by multiplying each row of C by the prior probability:

J	y_1	y_2	y_3	y_4
x_1	$1/8$	$1/8$	0	0
x_2	0	$1/8$	$1/4$	$1/8$
x_3	$1/8$	$1/12$	$1/24$	0

Summing the columns of J gives the marginal distribution $p_Y = (1/4, 1/3, 7/24, 1/8)$. Each possible value of Y gives rise to a *posterior distribution* on X, by Bayesian updating; these can be calculated by normalizing the columns of J:

	$p_{X\|y_1}$	$p_{X\|y_2}$	$p_{X\|y_3}$	$p_{X\|y_4}$
x_1	$1/2$	$3/8$	0	0
x_2	0	$3/8$	$6/7$	1
x_3	$1/2$	$1/4$	$1/7$	0

These posterior distributions reflect the knowledge about X obtained from each of the four possible output values. For instance, the posterior distribution $p_{X|y_4}$ shows that on output y_4, the adversary learns that X must be x_2. Notice that because we have a distribution (p_Y) on the four output values, we also have a *distribution on the posterior distributions*.

Finally, we observe that the output *labels* do not matter; all that matters is the distribution on posterior distributions; we call this a *hyper-distribution* [16], and denote it by $[\pi, C]$:

$[\pi, C]$	$1/4$	$1/3$	$7/24$	$1/8$
x_1	$1/2$	$3/8$	0	0
x_2	0	$3/8$	$6/7$	1
x_3	$1/2$	$1/4$	$1/7$	0

Figure 1 shows a graphical representation (using barycentric coordinates) of the prior π and the hyper-distribution $[\pi, C]$, showing how channel C "explodes" the prior π into four posterior distributions.[1]

In conclusion, the information-theoretic essence of C is a mapping from priors π to hyper-distributions $[\pi, C]$.

Note that this implies that classic channel matrices have structural redundancies: *labels of outputs*, *columns that are multiples of one another*, and *all-zero columns*, which are irrelevant with respect to (information-theoretic) leakage.[2] Eliminating these redundancies gives *abstract channels*, as studied in [20].

[1] Interestingly, if the four posterior distributions are weighted according to their probabilities $(1/4, 1/3, 7/24, 1/8)$, then the *center of mass* is located at the prior π.

[2] All-zero columns represent outputs that never occur, while columns that are multiples of one other give rise to the *same* posterior distribution.

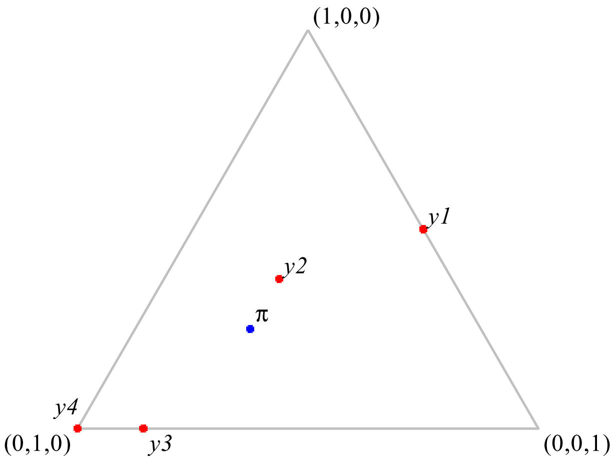

Figure 1. Graphical representation of π and hyper-distribution $[\pi, C]$.

B. Leakage as mutual information?

Early research in QIF (e.g. [9]) noted the similarity to *information theory* [35] and chose to quantify the leakage from X to Y using the classic concepts of *Shannon entropy* and *mutual information* [36]. This approach defines the prior uncertainty about X to be the Shannon entropy

$$H(X) = -\sum_x \pi_x \log \pi_x,$$

the posterior uncertainty about X to be the conditional Shannon entropy (which is simply the average Shannon entropy in the hyper-distribution)

$$H(X|Y) = \sum_y p(y) H(p_{X|y}),$$

and the leakage to be the mutual information

$$I(X;Y) = H(X) - H(X|Y).$$

Unfortunately, these measures do not give good security guarantees with respect to confidentiality. The key issue is that mutual information is concerned with the rate at which information can be transmitted *reliably*, while confidentiality is concerned with the risk that a secret *might* be compromised. For example, consider the following probabilistic program:

$$Y = X \;\; {}_{\frac{1}{8}}\!\oplus \;\; Y = -1$$

which chooses its left command ($Y = X$) with probability $\frac{1}{8}$ and its right command ($Y = -1$) with probability $\frac{7}{8}$. Suppose that X is a uniformly-distributed 64-bit unsigned integer, so that the initial uncertainty $H(X)$ is 64 bits. Then we find that the posterior uncertainty is 56 bits:

$$H(X|Y) = \frac{1}{8} \cdot 0 + \frac{7}{8} \cdot 64 = 56,$$

which means that the mutual information leakage $I(X;Y)$ is just 8 bits.

One might expect that the posterior entropy $H(X|Y)$ of 56 bits would imply that X is safe from discovery by the adversary. Yet $1/8$ of the time the adversary learns X exactly!

C. Vulnerability

For confidentiality, it seems more useful to measure leakage based on X's *vulnerability* [15] to be guessed correctly by the adversary in one try. The prior vulnerability is simply the largest probability in π:

$$V[\pi] = \max_x \pi_x$$

and the posterior vulnerability is

$$V[\pi, C] = \sum_y p(y) V[p_{X|y}]$$

which is the average vulnerability in the hyper-distribution. It is also the complement of the *Bayes risk*.[3]

A strength of vulnerability is its clear operational significance with respect to confidentiality. Indeed we can understand $V[\pi]$ as an optimal adversary $\mathcal{A}$'s probability of winning the following game, in which a value x is sampled according to π and $\mathcal{A}$ tries to guess it:

```
$x \overset{\$}{\leftarrow} \pi$;
$w \overset{\$}{\leftarrow} \mathcal{A}(\pi)$;
if $w = x$ then **win** else **lose**
```

Similarly, $V[\pi, C]$ is an optimal adversary $\mathcal{A}$'s probability of winning the following game, in which $\mathcal{A}$ is also given an output y sampled according to row x of C:

```
$x \overset{\$}{\leftarrow} \pi$;
$y \overset{\$}{\leftarrow} C_{x,-}$;
$w \overset{\$}{\leftarrow} \mathcal{A}(\pi, C, y)$;
if $w = x$ then **win** else **lose**
```

D. Min-entropy leakage

It is natural to quantify leakage in terms of the prior and posterior vulnerabilities, $V[\pi]$ and $V[\pi, C]$. However, there are a number of plausible ways that we might define the leakage $\mathcal{L}(\pi, C)$, including

- multiplicative: $V[\pi, C]/V[\pi]$
- "logged" multiplicative: $\log(V[\pi, C]/V[\pi])$
- additive: $V[\pi, C] - V[\pi]$.

Notice that the two multiplicative definitions focus on the *relative* difference between the posterior and prior vulnerabilities (with "log" just changing the scale), while the additive definition focuses on the *absolute* difference.

[3] A more conservative definition [37], [38] uses instead *worst-case* posterior vulnerability, taking the *maximum* of the vulnerabilities of the distributions in the hyper-distribution. But this has the drawback of being highly sensitive to a channel's worst output, even if that output is very unlikely. For instance it judges a password checker to be as bad as a channel that always leaks the entire secret.

While it is not clear that any of these definitions is canonical, a pleasant fact is that if we are just interested in *comparing* the leakage caused by two channels A and B, then *all* of these definitions give the same answer—we always get

$$\mathcal{L}(\pi, A) \leq \mathcal{L}(\pi, B) \quad \text{iff} \quad V[\pi, A] \leq V[\pi, B]\,.$$

In [15], *min-entropy leakage* is defined using the "logged" multiplicative definition. The reason for the name is that Rényi's min-entropy [39] is the negative logarithm of the vulnerability:

$$H_\infty(X) = -\log V[\pi].$$

So, since

$$\begin{aligned}\mathcal{L}(\pi, C) &= \log(V[\pi, C]/V[\pi]) \\ &= \log V[\pi, C] - \log V[\pi] \\ &= (-\log V[\pi]) - (-\log V[\pi, C])\end{aligned}$$

we see that $\mathcal{L}(\pi, C)$ is equivalently the difference between the prior and posterior min-entropy of X.

An interesting property of min-entropy leakage is that $\mathcal{L}(\pi, C) = 0$ whenever the adversary's best guess is unaffected by the output y. This can sometimes be surprising, as in the following example, which illustrates the so-called *base-rate fallacy*. Suppose that C is the channel matrix of a good, but imperfect, test for cancer:

C	*positive*	*negative*
cancer	$9/10$	$1/10$
no cancer	$1/10$	$9/10$

Moreover, suppose that for the population under consideration (say, age 40–50, no symptoms, no family history) the prior π is heavily biased towards "no cancer":

$$\pi = (1/100, 99/100).$$

Then, although the channel might appear to be quite reliable, we find that the min-entropy leakage is 0. For we find that the hyper-distribution $[\pi, C]$ is

$[\pi, C]$	$27/250$	$223/250$
cancer	$1/12$	$1/892$
no cancer	$11/12$	$891/892$

We see that a positive test result increases the probability of cancer from $1/100$ to $1/12$, while a negative test result decreases it to $1/892$. Nevertheless, C's output is useless in winning the second game above, since $\mathcal{A}$'s best guess is always "no cancer". And we see that

$$V[\pi, C] = 27/250 \cdot 11/12 + 223/250 \cdot 891/892 = 99/100 = V[\pi],$$

implying that the min-entropy leakage

$$\mathcal{L}(\pi, C) = \log(V[\pi, C]/V[\pi]) = \log 1 = 0.$$

E. *Generalizing to g-vulnerability and g-leakage*

While vulnerability is clearly a fundamental security metric, it is certainly not appropriate in all scenarios—for instance, in the context of password checker, the adversary might be allowed five tries before being locked out. For this reason, [18] introduced *g-vulnerability* V_g, which parameterizes vulnerability with a *gain function* g that can model diverse operational scenarios, including those where the adversary gains by guessing the secret partially, approximately, or in k tries, or where there is a penalty for incorrect guesses.

In each scenario, there will be some set $\mathcal{W}$ of *guesses* (or *actions*) that the adversary could make about the secret, and for any guess w and secret value x, there will be some *gain* $g(w, x)$ that the adversary gets by having chosen w when the secret's actual value was x. Formally, $g : \mathcal{W} \times \mathcal{X} \to [0, 1]$, where $\mathcal{W}$ is a finite, non-empty set.[4]

Given a gain function g, the prior g-vulnerability is defined as the maximum expected gain over all possible guesses:

$$V_g[\pi] = \max_w \sum_x \pi_x g(w, x)\,.$$

The posterior g-vulnerability is then defined as before:

$$V_g[\pi, C] = \sum_y p(y) V_g[p_{X|y}]\,.$$

And then we can define *g-leakage* in terms of the prior and posterior g-vulnerabilities, either multiplicatively or additively.

Note that (ordinary) vulnerability is a *special case* of g-vulnerability, corresponding to the *identity gain function* $g_{id} : \mathcal{X} \times \mathcal{X} \to [0, 1]$ given by

$$g_{id}(w, x) = \begin{cases} 1, & \text{if } w = x, \\ 0, & \text{if } w \neq x. \end{cases}$$

(Notice that when written as a matrix, g_{id} is the *identity matrix.*)

But gain functions can do much more. As explained in [18], they can model a wide variety of practical operational scenarios, including those where the adversary benefits from guessing a value *close* to the secret, guessing a *part* of the secret, guessing a *property* of the secret or guessing the secret within some bounded number of tries. They can also model scenarios where there is a *penalty* for incorrect guesses, or where some parts of the secret are worth more than others.

III. Robustness in QIF analysis

Using g-leakage, we can measure leakage in a rich variety of operational scenarios. But we could worry about the

[4] It is, however, of interest to generalize gain functions in various ways, allowing the range to be larger than the interval $[0, 1]$ and allowing the set $\mathcal{W}$ to be countably infinite.

robustness of our conclusions about leakage. The issue is that the g-leakage $\mathcal{L}_g(\pi, C)$ depends on both prior π, which models the adversary's prior knowledge about X, and gain function g, which models (among other things) what is valuable to the adversary. Also, we can choose to calculate leakage additively or multiplicatively. How confident can we be about these? Can we minimize our sensitivity to questionable assumptions about π and g?

There has been considerable interest in achieving robustness in QIF analysis. One important approach is to consider *capacity*, the maximum leakage over all priors π and/or all gain functions g; we can also consider both multiplicative and additive versions of capacity. A second approach to robustness concerns *comparison* of channels, aimed at showing that one channel *never* leaks more than another, regardless of the prior or gain function. We discuss these two approaches in the following sections.

IV. Capacity

One important approach to robustness is to abstract away from the prior π and/or the gain function g, considering instead the *capacity* or maximum leakage over *all* priors and/or gain functions.[5] We separately consider both multiplicative and additive versions of capacity.

A. Multiplicative capacity

Definition 4.1: The *min-capacity of* C, denoted $\mathcal{ML}(C)$, is the maximum min-entropy leakage of C over all π:

$$\mathcal{ML}(C) = \sup_{\pi} \log(V[\pi, C]/V[\pi]).$$

Pleasantly, min-capacity satisfies a number of useful theorems.

First, as shown in [40], [31], min-capacity is easy to calculate:

Theorem 4.1: $\mathcal{ML}(C)$ is the logarithm of the sum of the column maximums of C, and it is always realized on a uniform prior π.

Corollary 4.2: $\mathcal{ML}(C) = 0$ iff the rows of C are identical.

Corollary 4.3: For deterministic C, $\mathcal{ML}(C)$ is the logarithm of the number of feasible output values.

Another important result about min-capacity concerns *cascading*. Given channel A from X to Y and channel B from Y to Z, the *cascade* [41] is the channel AB from X to Z formed by *multiplying* channel matrices A and B. As shown in [38], the min-capacity of a cascade AB is upper bounded by the min-capacities of A and B:

Theorem 4.4: $\mathcal{ML}(AB) \leq \min\{\mathcal{ML}(A), \mathcal{ML}(B)\}$.

Min-capacity is also interesting in terms of its relationship with *other* leakage measures. Interestingly, min-capacity and Shannon capacity *coincide* on *deterministic* channels [42], and min-capacity is an *upper bound* on Shannon capacity on arbitrary channels [18].

More remarkably, [18] shows that min-capacity is an upper bound on "logged" multiplicative g-leakage, for *every* prior and *every* gain function:

Theorem 4.5 (Miracle): For all priors π and gain functions g, $\mathcal{ML}(C) \geq \log(V_g[\pi, C]/V_g[\pi])$.

The Miracle Theorem shows that if we can prove that C's min-capacity is small, then we know that its "logged" multiplicative g-leakage is also small, regardless of the prior π and gain function g.

To illustrate, suppose that the secret is an array X containing 10-bit, uniformly distributed passwords for 1000 users. Consider the following probabilistic channel C, which leaks *some* randomly-chosen user's password:

$$\begin{array}{c} u \overset{\$}{\leftarrow} \{0..999\}; \\ Y = (u, X[u]) \end{array}$$

With respect to the uniform prior π_u, we have $V[\pi_u] = 2^{-10000}$ and $V[\pi_u, C] = 2^{-9990}$, so (using Theorem 4.1) we conclude that $\mathcal{ML}(C) = 10$. But this measures the leakage caused by C in terms of the benefit to an adversary trying to guess the *entire* array of passwords.

We might instead focus on an adversary $\mathcal{A}$ who simply wants to guess *some* user's password, with no preference as to whose it is. To achieve this, we define

$$\mathcal{W} = \{(u, x) \mid 0 \leq u \leq 999 \text{ and } 0 \leq x \leq 1023\}$$

and

$$g((u, x), X) = \begin{cases} 1, & \text{if } X[u] = x \\ 0, & \text{otherwise.} \end{cases}$$

The Miracle Theorem assures us that the "logged" multiplicative g-leakage cannot exceed 10, meaning that the g-vulnerability cannot increase by more than a factor of 2^{10}. In fact, this is precisely what happens here. We have $V_g[\pi_u] = 2^{-10}$, since $\mathcal{A}$'s best choice *a priori* is simply to guess some password x for some user u, which gives expected gain 2^{-10}. And $V_g[\pi_u, C] = 1$, since given the output of C the adversary can always achieve a gain of 1.

It is important to note, then, that the Miracle Theorem does not give any direct guarantees about the posterior g-vulnerability; it bounds only the *ratio* between the posterior and prior g-vulnerabilities.

B. Additive capacity

In the case of additive capacity under a fixed gain function g and a universally quantified prior, it is proved in [21] that it challenging to compute the capacity:

Theorem 4.6: When g is g_{id} (giving ordinary vulnerability), it is NP-complete to decide whether the additive capacity of C exceeds a given threshold t.

It should be emphasized that the input here is the *channel matrix* C, rather than a concise program (which invariably leads to intractability).

[5] Recall that Shannon [36] introduced capacity for mutual information and proved that its value is a tight bound on the amount of information that can be transmitted reliably over a channel.

Even more interesting is additive capacity when π is fixed and g is universally quantified. While one might expect a similar computational hardness result, it is shown in [21] that the additive capacity can be computed efficiently by exploiting some abstract mathematics. For the additive capacity is the supremum over g of $V_g[\pi, C] - V_g[\pi]$, which is equivalent to the supremum over V_g of the difference between the expectations of V_g over $[\pi, C]$ and over $[\pi]$, where $[\pi]$ is understood as a point hyper-distribution. This is strikingly similar to the *Kantorovich distance* between $[\pi, C]$ and $[\pi]$, except that the latter takes the supremum over all 1-Lipschitz functions from distributions on X to reals. But, with g restricted to *1-spanning gain functions*,[6] V_g is always 1-Lipschitz. Moreover, gain functions are expressive enough that we actually achieve equality:

Theorem 4.7: The additive capacity (over all 1-spanning gain functions) of C under prior π is equal to the Kantorovich distance between $[\pi, C]$ and $[\pi]$.

This result is exceedingly useful, because by the 1958 *Kantorovich-Rubinstein theorem* [43], it implies that the additive capacity is also the *earth-moving distance* between $[\pi, C]$ and $[\pi]$. And that can be computed straightforwardly in time linear in the size of C.

As an illustration, consider the channel C discussed in Section II-A. With respect to ordinary vulnerability (which corresponds to g_{id}), we can easily calculate the additive leakage:

$$V[\pi] = 1/2$$

and

$$V[\pi, C] = 1/4 \cdot 1/2 + 1/3 \cdot 3/8 + 7/24 \cdot 6/7 + 1/8 \cdot 1 = 5/8,$$

giving additive leakage of $5/8 - 1/2 = 1/8$.

If we are interested in calculating the additive capacity over all 1-spanning gain functions, then by Theorem 4.7 it suffices to calculate the earth-moving distance between $[\pi]$ and $[\pi, C]$. To do this, we first calculate the earth-moving distance between π and each of the posterior distributions. For instance, the distance between π and $p_{X|y_3}$ is $5/14$, since we must move $1/4$ units of "earth" from x_1 to x_2, and $3/28$ units from x_3 to x_2. Overall we get an additive capacity of

$$1/4 \cdot 1/2 + 1/3 \cdot 1/8 + 7/24 \cdot 5/14 + 1/8 \cdot 1/2 = 1/3.$$

Notice that $1/3$ exceeds $1/8$, the additive leakage obtained with g_{id}. This is as expected, since g_{id} is a 1-spanning gain function.

V. Robust channel ordering

Given channels A and B, both taking input X, the question of *which leaks more* will ordinarily depend on the prior and gain function used.[7] As an example, assume that X is a 64-bit unsigned integer, and consider the following channels:

$$\begin{array}{ll} A. & Y = X \ {}_{\frac{1}{8}}\!\oplus \ Y = -1 \\ B. & Y = X \ \mid \ \texttt{0x7} \end{array}$$

Both have a min-capacity of 61.0 bits out of 64, since both have a posterior vulnerability of about $1/8$. But we can distinguish them using gain functions.

We can define a gain function g_3 that allows the adversary 3 tries. Its guesses w are size-3 subsets of $\mathcal{X}$, and

$$g_3(w, x) = \begin{cases} 1, & \text{if } x \in w, \\ 0, & \text{if } x \notin w. \end{cases}$$

Observe that the posterior g_3-vulnerability of B is $3/8$, since B reveals all but the last three bits of X. In contrast, the posterior g_3-vulnerability of A remains about $1/8$, because when $Y \neq -1$, the adversary already knows X exactly, and when $Y = -1$, the adversary can just make 3 stabs in the dark. Hence g_3 makes B leak more than A.

Alternatively, we can define a gain function g_{tiger} for which making a wrong guess triggers a penalty (say, opening a trap door to a pit of tigers), and which allows the adversary to use the special value $\perp$ to indicate that it chooses not to make a guess. With suitable gain and penalty values, the g_{tiger}-leakage of B is 0, because the output of B gives the adversary only a $1/8$ probability of guessing X correctly, making $\perp$ the best choice. In contrast, the g_{tiger}-leakage of A is greater than 0, because when $Y \neq -1$, the adversary knows X exactly and can make a guess without fear of any penalty. Hence g_{tiger} makes A leak more than B.

The question is then whether there is a *robust* leakage ordering, allowing us to conclude that one channel *never* leaks more than another, regardless of the prior or gain function. Such a robust ordering could support a stepwise refinement methodology for constructing secure systems.

It turns out that there is such an ordering [18], [44]:

Definition 5.1: B composition refines A, written $A \sqsubseteq_\circ B$, if there exists a channel C such that $B = AC$.

Note that composition refinement requires that B can be expressed as the *cascade* of A and C, for some "post-processing" channel C.[8]

The main interest in composition refinement is its relation to g-leakage. First, composition refinement implies a strong g-leakage ordering; this can be seen as an analogue of the classic *data-processing inequality*.[9]

Theorem 5.1 (Data-processing inequality): If $A \sqsubseteq_\circ B$ then the g-leakage of B never exceeds that of A, for any prior π and any gain function g.

More interestingly, the converse implication holds as well:

[6] These are g where for any w the gain values $g(w, x)$ and $g(w, x')$ differ by at most 1, for all x and x'.

[7] It will not, however, depend on whether leakage is defined multiplicatively or additively.

[8] In [18], note that the order of the two channels was *reversed*.

[9] To see that this is indeed the g-leakage version of the data-processing inequality [35], note that if $B = AC$, where B goes from $\mathcal{X}$ to $\mathcal{Z}$, A goes from $\mathcal{X}$ to $\mathcal{Y}$, and C goes from $\mathcal{Y}$ to $\mathcal{Z}$, then for any prior π we have a Markov chain $X \rightarrow Y \rightarrow Z$. The data-processing inequality says that in this case $I(X; Z) \leq I(X; Y)$.

Theorem 5.2 ("Coriaceous"): If the g-leakage of B never exceeds that of A, for any prior π and any gain function g, then $A \sqsubseteq_{\circ} B$.

We hence see that composition refinement is an ordering on channels with both *structural* and *leakage-testing* significance.

These fundamental theorems about composition refinement were proved in [18] and [20], using the *separating hyperplane lemma* [45] to construct the g needed in the contrapositive form of the "coriaceous" direction. However, we later learned that these theorems were in fact discovered already in 1951, by statistician David Blackwell [46].

We conclude this section with some discussion of the structure of channels under composition refinement. We first observe that composition refinement is only a *pre-order* on channel matrices. For example, consider the following channel matrices:

$$\begin{array}{|c|ccc|} \hline C & y_1 & y_2 & y_3 \\ \hline x_1 & 1 & 0 & 0 \\ x_2 & 1/4 & 1/2 & 1/4 \\ x_3 & 1/2 & 1/3 & 1/6 \\ \hline \end{array} \qquad \begin{array}{|c|ccc|} \hline D & z_1 & z_2 & z_3 \\ \hline x_1 & 2/5 & 0 & 3/5 \\ x_2 & 1/10 & 3/4 & 3/20 \\ x_3 & 1/5 & 1/2 & 3/10 \\ \hline \end{array}$$

It turns out that $C \sqsubseteq_{\circ} D$ and also $D \sqsubseteq_{\circ} C$. But this is less significant than it might first appear, because in fact C and D are actually the *same* abstract channel, since they denote the *same* mapping from priors to hyper-distributions. (To see this, note that the second and third columns of C are multiples of one another, as are the first and third columns of D; because such "similar" columns always produce the same posterior distribution, they may as well be merged.)

In [20], it is shown that composition refinement is *antisymmetric* on *abstract channels*, and hence a *partial order*.

VI. Future directions and conclusion

We conclude with a brief discussion of some important future directions.

One important direction is the automated analysis of leakage in systems. A variety of static-analysis techniques have been explored in the literature, based on type systems [23], statistical sampling [27], [28], model checking [24], [25], [32], [30], and abstract interpretation [37], [34]. While considerable progress has been achieved, it remains unclear whether such analyses can be scaled up successfully to large-scale systems.

Differential privacy [7] is a popular approach to controlling the leakage of sensitive information caused by queries to statistical databases. The relationship between differential privacy and QIF has received some study [47], [48], but it would be valuable to achieve a better unification of these two approaches.

Finally, our perspective here has been *information theoretic*, concerned only with a channel's mapping from priors to hyper-distributions, and abstracting from details like the names of outputs. These choices are appropriate if we are interested only in the *information* that a channel provides to the adversary with unbounded computational resources. But of course the resulting conclusions may be overly pessimistic with respect to a resource-constrained adversary.

For example, consider a channel C that takes as input an n-bit prime, assumed uniformly distributed. C then randomly chooses an $(n+1)$-bit prime q and outputs pq. Notice then that each column of the channel matrix of C contains exactly one non-zero entry—each column is labeled with the product pq of an n-bit prime p and an $(n+1)$-bit prime q, and has a non-zero entry only in row p. This means that the hyper-distribution $[\pi_u, C]$ contains only *point* posterior distributions. Hence $V[\pi_u, C] = 1$, meaning that C leaks p completely. But of course adversary $\mathcal{A}$, seeing output label pq, needs to *factor* pq in order to recover p. And, under standard assumptions about the difficulty of factorization, a polynomial-time adversary has a negligible probability of succeeding in doing this.

In [20], some preliminary ideas are given about a *computational* version of posterior g-vulnerability. The idea is that the maximization done by $V_g[\pi, C]$ can be described concretely as a *strategy* S that chooses a best guess w given each channel output y; it is then natural to demand that S should be efficiently computable. Developing a theory of *computational g-leakage* would be challenging but interesting, as it might allow QIF to address cryptographic constructions.

Acknowledgements

I am deeply grateful to my collaborators, especially Mário Alvim, Miguel Andrés, Nico Bordenabe, Kostas Chatzikokolakis, Barbara Espinoza, Boris Köpf, Piotr Mardziel, Annabelle McIver, Ziyuan Meng, Carroll Morgan, Catuscia Palamidessi, and Marco Stronati, for their many profound contributions to quantitative information flow. I am also grateful for support by the National Science Foundation under grant CNS-1116318, from Digiteo, and from the INRIA équipe associée Princess.

References

[1] N. A. Lynch and M. J. Fischer, "On describing the behavior and implementation of distributed systems," *Theoretical Computer Science*, vol. 13, pp. 17–43, 1981.

[2] E. Cohen, "Information transmission in computational systems," in *Proc. 6th ACM Symposium on Operating Systems Principles*, 1977, pp. 133–139.

[3] J. Goguen and J. Meseguer, "Security policies and security models," in *IEEE Symposium on Security and Privacy*, 1982, pp. 11–20.

[4] D. Volpano, G. Smith, and C. Irvine, "A sound type system for secure flow analysis," *Journal of Computer Security*, vol. 4, no. 2,3, pp. 167–187, 1996.

[5] A. Banerjee and D. A. Naumann, “Secure information flow and pointer confinement in a Java-like language,” in *Proc. 15th IEEE Computer Security Foundations Workshop (CSFW 2002)*, 2002, pp. 253–267.

[6] M. K. Reiter and A. D. Rubin, “Crowds: Anonymity for web transactions,” *ACM Transactions on Information Systems Security*, vol. 1, no. 1, pp. 66–92, 1998.

[7] C. Dwork, “Differential privacy,” in *Proc. 33rd International Colloquium on Automata, Languages, and Programming (ICALP 2006)*, 2006, pp. 1–12.

[8] P. Kocher, “Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems,” in *Proc. Advances in Cryptology (CRYPTO 1996)*, ser. Lecture Notes in Computer Science, vol. 1109. Springer-Verlag, 1996, pp. 104–113.

[9] D. Clark, S. Hunt, and P. Malacaria, “Quantitative analysis of the leakage of confidential data,” in *Proc. Workshop on Quantitative Aspects of Programming Languages*, ser. Electr. Notes Theor. Comput. Sci, vol. 59 (3), 2001, pp. 238–251.

[10] ——, “Quantitative information flow, relations and polymorphic types,” *Journal of Logic and Computation*, vol. 18, no. 2, pp. 181–199, 2005.

[11] M. Clarkson, A. Myers, and F. Schneider, “Belief in information flow,” in *Proc. 18th IEEE Computer Security Foundations Workshop (CSFW ’05)*, 2005, pp. 31–45.

[12] B. Köpf and D. Basin, “An information-theoretic model for adaptive side-channel attacks,” in *Proc. 14th ACM Conference on Computer and Communications Security (CCS ’07)*, 2007, pp. 286–296.

[13] P. Malacaria, “Assessing security threats of looping constructs,” in *Proc. 34th Symposium on Principles of Programming Languages (POPL ’07)*, 2007, pp. 225–235.

[14] K. Chatzikokolakis, C. Palamidessi, and P. Panangaden, “On the Bayes risk in information-hiding protocols,” *Journal of Computer Security*, vol. 16, no. 5, pp. 531–571, 2008.

[15] G. Smith, “On the foundations of quantitative information flow,” in *Proc. 12th International Conference on Foundations of Software Science and Computational Structures (FoSSaCS ’09)*, ser. Lecture Notes in Computer Science, L. de Alfaro, Ed., vol. 5504, 2009, pp. 288–302.

[16] A. McIver, L. Meinicke, and C. Morgan, “Compositional closure for Bayes risk in probabilistic noninterference,” in *Proc. ICALP’10*, 2010, pp. 223–235.

[17] M. S. Alvim, M. Andrés, and C. Palamidessi, “Probabilistic information flow,” in *Proc. 25th IEEE Symposium on Logic in Computer Science (LICS 2010)*, 2010, pp. 314–321.

[18] M. S. Alvim, K. Chatzikokolakis, C. Palamidessi, and G. Smith, “Measuring information leakage using generalized gain functions,” in *Proc. 25th IEEE Computer Security Foundations Symposium (CSF 2012)*, Jun. 2012, pp. 265–279.

[19] A. McIver, L. Meinicke, and C. Morgan, “A Kantorovich-monadic powerdomain for information hiding, with probability and nondeterminism,” in *Proc. 27th IEEE Symposium on Logic in Computer Science (LICS 2012)*, 2012, pp. 461–470.

[20] A. McIver, C. Morgan, G. Smith, B. Espinoza, and L. Meinicke, “Abstract channels and their robust information-leakage ordering,” in *Proc. 3rd Conference on Principles of Security and Trust (POST 2014)*, 2014, pp. 83–102.

[21] M. S. Alvim, K. Chatzikokolakis, A. McIver, C. Morgan, C. Palamidessi, and G. Smith, “Additive and multiplicative notions of leakage, and their capacities,” in *Proc. 27th IEEE Computer Security Foundations Symposium (CSF 2014)*, 2014, pp. 308–322.

[22] A. McIver, L. Meinicke, and C. Morgan, “Hidden-Markov program algebra with iteration,” *Mathematical Structures in Computer Science*, vol. 25, pp. 320–360, 2015.

[23] D. Clark, S. Hunt, and P. Malacaria, “A static analysis for quantifying information flow in a simple imperative language,” *Journal of Computer Security*, vol. 15, pp. 321–371, 2007.

[24] M. Backes, B. Köpf, and A. Rybalchenko, “Automatic discovery and quantification of information leaks,” in *Proc. 30th IEEE Symposium on Security and Privacy*, 2009, pp. 141–153.

[25] J. Newsome, S. McCamant, and D. Song, “Measuring channel capacity to distinguish undue influence,” in *Proc. Fourth Workshop on Programming Languages and Analysis for Security (PLAS ’09)*, 2009, pp. 73–85.

[26] M. Andrés, C. Palamidessi, P. van Rossum, and G. Smith, “Computing the leakage of information-hiding systems,” in *Tools and Algorithms for the Construction and Analysis of Systems (TACAS ’10)*, ser. Lecture Notes in Computer Science, J. Esparza and R. Majumdar, Eds., vol. 6015, 2010, pp. 373–389.

[27] K. Chatzikokolakis, T. Chothia, and A. Guha, “Statistical measurement of information leakage,” in *Tools and Algorithms for the Construction and Analysis of Systems (TACAS ’10)*, 2010, pp. 390–404.

[28] B. Köpf and A. Rybalchenko, “Approximation and randomization for quantitative information-flow analysis,” in *Proc. 23nd IEEE Computer Security Foundations Symposium (CSF ’10)*, 2010, pp. 3–14.

[29] H. Yasuoka and T. Terauchi, “Quantitative information flow — verification hardness and possibilities,” in *Proc. 23nd IEEE Computer Security Foundations Symposium (CSF ’10)*, 2010, pp. 15–27.

[30] Z. Meng and G. Smith, “Faster two-bit pattern analysis of leakage,” in *Proc. 2nd International Workshop on Quantitative Aspects of Security Assurance (QASA ’13)*, 2013.

[31] B. Köpf and G. Smith, “Vulnerability bounds and leakage resilience of blinded cryptography under timing attacks,” in *Proc. 23nd IEEE Computer Security Foundations Symposium (CSF ’10)*, 2010, pp. 44–56.

[32] J. Heusser and P. Malacaria, “Quantifying information leaks in software,” in *Proc. ACSAC ’10*, 2010, pp. 261–269.

[33] B. Köpf, L. Mauborgne, and M. Ochoa, "Automatic quantification of cache side-channels," in *Proc. 24th International Conference on Computer-Aided Verification (CAV '12)*, 2012, pp. 564–580.

[34] G. Doychev, D. Feld, B. Köpf, L. Mauborgne, and J. Reineke, "Cacheaudit: A tool for the static analysis of cache side channels," in *22nd USENIX Security Symposium (USENIX Security 13)*, 2013, pp. 431–446.

[35] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. John Wiley & Sons, Inc., 2006.

[36] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, 1948.

[37] P. Mardziel, S. Magill, M. Hicks, and M. Srivatsa, "Dynamic enforcement of knowledge-based security policies," in *Proceedings of the Computer Security Foundations Symposium (CSF '11)*, Jun. 2011, pp. 114–128.

[38] B. Espinoza and G. Smith, "Min-entropy as a resource," *Information and Computation (Special Issue on Information Security as a Resource)*, vol. 226, pp. 57–75, Apr. 2013.

[39] A. Rényi, "On measures of entropy and information," in *Proc. 4th Berkeley Symposium on Mathematics, Statistics and Probability 1960*, 1961, pp. 547–561.

[40] C. Braun, K. Chatzikokolakis, and C. Palamidessi, "Quantitative notions of leakage for one-try attacks," in *Proc. 25th Conference on Mathematical Foundations of Programming Semantics (MFPS 2009)*, ser. ENTCS, vol. 249, 2009, pp. 75–91.

[41] C. A. Desoer, "Communication through channels in cascade," Ph.D. dissertation, Massachusetts Institute of Technology, 1953.

[42] G. Smith, "Quantifying information flow using min-entropy," in *Proc. QEST 2011: 8th International Conference on Quantitative Evaluation of SysTems*, 2011, pp. 159–167.

[43] Y. Deng and W. Du, "Kantorovich metric in computer science: A brief survey," *Electronic Notes in Theoretical Computer Science*, vol. 353, no. 3, pp. 73–82, 2009.

[44] A. McIver, L. Meinicke, and C. Morgan, "Compositional closure for Bayes risk in probabilistic noninterference," *CoRR*, vol. abs/1007.1054, 2010, draft full version of [16] with appendices.

[45] K. Trustrum, *Linear Programming*, ser. Library of Mathematics. London: Routledge and Kegan Paul, 1971.

[46] D. Blackwell, "Comparison of experiments," in *Proc. Second Berkeley Symposium on Mathematical Statistics and Probability*, 1951, pp. 93–102.

[47] G. Barthe and B. Köpf, "Information-theoretic bounds for differentially private mechanisms," in *Proc. 24th IEEE Computer Security Foundations Symposium (CSF 2011)*, 2011, pp. 191–204.

[48] M. Alvim, M. Andrés, K. Chatzikokolakis, P. Degano, and C. Palamidessi, "On the information leakage of differentially-private mechanisms," *Journal of Computer Security*, 2015, to appear.