Min-Entropy as a Resource

Barbara Espinoza, Geoffrey Smith

School of Computing and Information Sciences
Florida International University
Miami, FL 33199 USA

Abstract

Secrecy is fundamental to computer security, but real systems often cannot avoid leaking some secret information. For this reason, it is useful to model secrecy quantitatively, thinking of it as a "resource" that may be gradually "consumed" by a system. In this paper, we explore this intuition through several dynamic and static models of secrecy consumption, ultimately focusing on (average) vulnerability and min-entropy leakage as especially useful models of secrecy consumption. We also consider several composition operators that allow smaller systems to be combined into a larger system, and explore the extent to which the secrecy consumption of a combined system is constrained by the secrecy consumption of its constituents.

Keywords: secrecy, quantitative information flow, information theory

1. Introduction

Many security goals depend crucially on *secrecy*. Sometimes secrecy is crucial *in itself*, as in the case of anonymity protocols like Crowds [1], which aim to keep the identities of message initiators secret from adversaries who are able to observe network traffic. Other times secrecy is crucial *as a tool*, as in the case of an operating system login program, which must keep user passwords secret in order to achieve *authentication*.

Maintaining secrecy, however, is a challenging issue due to the practical difficulty of preventing all leakage of secret information. As a basic example, a login program that rejects an incorrect password unavoidably reveals that the secret password differs from the one that was entered. Similarly, revealing the tally of votes in an election reveals some information about the secret ballots—if the election happens to be unanimous, for example, then we learn what *all* of the ballots were. More subtly, an adversary may be able to observe *side information* about a system's implementation that may leak secret information. For example, Kocher's celebrated timing attack on RSA [2] demonstrates that the timings of a set of RSA decryptions can allow an adversary to deduce the private key.

Email addresses: bespi009@cis.fiu.edu (Barbara Espinoza), smithg@cis.fiu.edu (Geoffrey Smith)

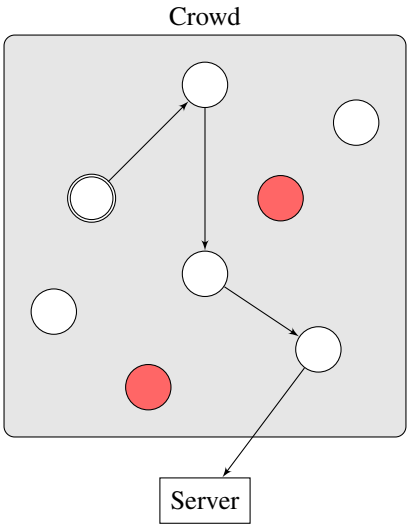

Figure 1: Crowds protocol

One promising way to address information leakage is to consider it *quantitatively*, based on the intuition that a login program is acceptable in practice because it leaks only a "small" amount of information about the secret password. This viewpoint has led to the area of *quantitative information flow*, which has seen growing interest in the past decade. (See, for example, [3, 4, 5, 6, 7, 8, 9, 10].)

In this paper, we explore particularly the idea that secrecy can be viewed as a "resource" that is "created" through a random process, and then gradually "consumed" by a system. As a first intuitive example, suppose that a system takes as input a 32-bit integer X, where we assume that all 2^{32} possible values are equally likely. If the system performs the bitwise "and" operation

```
Y = X & 0x007f;
```

then an adversary $\mathcal{A}$ seeing the value of Y learns exactly the last 7 bits of X, and remains entirely ignorant of the first 25 bits. Thus it seems clear here that the system starts with 32 bits of secrecy and consumes 7 bits of it, leaving 25 bits of remaining secrecy.

But other systems are considerably more subtle. As an example, consider the Crowds anonymity protocol of Reiter and Rubin [1]. In this protocol, a *crowd* of m users cooperates to communicate anonymously with a server. A crowd member wanting to send a message to the server initially sends it to a randomly-chosen *forwarder* (possibly itself). Then, with probability p_f, each forwarder sends the message to another randomly-chosen forwarder (again, possibly itself) or, with probability $1 - p_f$, sends it to the server. Figure 1 illustrates a possible message path. Note that when the server receives a message from user i, i is always a randomly-chosen forwarder, and hence no information about the initiator is revealed. However, we further assume that c of the crowd members are actually *collaborators* trying to compromise anonymity.

(The collaborators are shaded in Figure 1.) If user i forwards a message to a collaborator, then the collaborator reveals i to the server—this weakens anonymity, since i is now more likely than the other crowd members to have been the initiator. But quantifying *how much* secrecy is consumed here is not at all obvious. (We will give answers to this question in Section 2.)

The main goal of this paper is to develop the viewpoint of secrecy as a resource in the general setting of probabilistic systems (like the Crowds protocol), which we model as information-theoretic channels. We begin in Section 2 by considering how to measure the amount of secrecy that a system starts with, and then considering how to measure the amount of secrecy that is consumed by the system's execution. We will first consider a *dynamic* measure of the consumption caused by a particular system execution, but we will argue that this viewpoint does not work very well, both because it makes policy enforcement difficult and because it can result in a system "consuming" a *negative* amount of secrecy. We will then argue that it is more useful to adopt a *static* measure of the consumption caused by the system as a whole. We will see that both worst-case and average-case static measures can be useful, but we will finally focus our attention on an average-case model, namely the *min-entropy leakage* model of [9, 11, 12].

In Section 3, we review the basic properties of min-entropy leakage already established in the literature. Then, in Section 4, we consider a variety of system *composition* operators that allow smaller systems to be combined into a larger system. We explore the extent to which the secrecy consumption of a combined system is constrained by the secrecy consumption of its constituents. We collect bounds shown in the recent literature, and also consider their tightness, giving examples that show that the bounds cannot be strengthened in general.

Finally, Sections 5 and 6 discuss related work and conclude.

While the main goal of this paper is a unified presentation of the existing min-entropy leakage model of [9, 11, 12] from the perspective of *secrecy as a resource*, this paper also makes a number of technical contributions, including

- in Sections 2.2 and 2.3, exploration of new dynamic and worst-case leakage measures,
- in Section 2.5, an analysis of the average-case and worst-case min-entropy leakage of the Crowds protocol [1],
- in Section 4, refinements to compositionality results that have appeared in the literature, and
- throughout, a number of new illustrative examples.

2. Measuring secrecy and the consumption of secrecy

In this section, we explore the idea that secrecy can be viewed as a *resource* that is gradually *consumed* by a system.

To begin with, we assume that the secret is *created* by a random process, so that the secret is a random variable X that ranges over some finite set $\mathcal{X}$ according to some

distribution π. We make the worst-case assumption that π is known to adversary $\mathcal{A}$ that tries to guess the value of X.

How can we measure the amount of secrecy that we start with? It is tempting to turn to information theory, and to adopt Shannon's entropy [13] as a measure of $\mathcal{A}$'s uncertainty about X:

$$H(\pi) = -\sum_{x \in \mathcal{X}} \pi[x] \log \pi[x].$$

However, Shannon entropy does not work very well as a secrecy measure. The trouble is that a secret that is highly vulnerable to being guessed correctly by $\mathcal{A}$ may nonetheless have arbitrarily high Shannon entropy.

Example 1. Let

$$\pi = (1/2, 2^{-1000}, 2^{-1000}, 2^{-1000}, \ldots, 2^{-1000}).$$

Then $H(\pi) = 1/2 \log 2 + 2^{999} \cdot 2^{-1000} \cdot \log 2^{1000} = 500.5$ bits, even though $\mathcal{A}$ has probability $1/2$ of guessing the value of X correctly in one try. □

For this reason, Smith [9] proposed to focus instead on the secret's *vulnerability* to being guessed correctly by the adversary in one try; this is simply the maximum probability in π:

Definition 1. The *vulnerability* of π is given by $V(\pi) = \max_{x \in \mathcal{X}} \pi[x]$. □

We can obtain a measure (in bits) of uncertainty by taking the negative logarithm of the vulnerability; this turns out to be Rényi's min-entropy [14]:

Definition 2. The *min-entropy* of π is given by $H_\infty(\pi) = -\log V(\pi)$. □

With min-entropy, the secrecy of the distribution in Example 1 is now

$$H_\infty(\pi) = -\log V(\pi) = -\log 1/2 = 1,$$

a quantity that accurately reflects the large threat to the secret.

Of course min-entropy is a rather crude measure, in that it depends only on the maximum probability in π. So, for example, the min-entropy is also 1 on the distribution $\pi' = (1/2, 1/2)$, which is clearly less secure than π. Still, it is reasonable to say that both π and π' are situations where there is little initial secrecy.

2.1. Consumption of secrecy

We now turn our attention to the *consumption* of secrecy in a system. We model the system as an information-theoretic channel [15].

A *channel* is a triple $(\mathcal{X}, \mathcal{Y}, C)$, where $\mathcal{X}$ and $\mathcal{Y}$ are finite sets (of secret input values and observable output values) and C is a *channel matrix*, an $|\mathcal{X}| \times |\mathcal{Y}|$ matrix whose entries are between 0 and 1 and whose rows each sum to 1; the intent is that $C[x, y]$ is the probability of getting output y when the input is x. Channel C is *deterministic* if each entry of C is either 0 or 1, implying that each row contains exactly one 1, which means that each input produces a unique output.

Given a *prior distribution* π on $\mathcal{X}$, we can define a *joint distribution* p on $\mathcal{X} \times \mathcal{Y}$ by

$$p(x, y) = \pi[x]C[x, y].$$

This gives jointly distributed random variables X and Y with marginal probabilities $p(x) = \sum_y p(x, y)$ and $p(y) = \sum_x p(x, y)$, and conditional probabilities

$$p(y|x) = \frac{p(x, y)}{p(x)},$$

provided that $p(x)$ is nonzero. As shown in [16], p is the *unique* joint distribution that recovers π and C, in that $p(x) = \pi[x]$ and $p(y|x) = C[x, y]$ (if $p(x)$ is nonzero).

Of particular interest are the conditional probabilities $p(x|y)$, which can be computed using *Bayes's theorem*:

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} = \frac{p(x, y)}{p(y)}.$$

Hence for every $y \in \mathcal{Y}$ (with $p(y)$ nonzero), we get a *posterior distribution* $p_{X|y}$.[1] If we make the additional worst-case assumption that adversary $\mathcal{A}$ knows the prior distribution π and the channel matrix C, then the posterior distribution $p_{X|y}$ gives $\mathcal{A}$'s updated knowledge about X if it sees output y.

Example 2. Consider channel $(\{x_1, x_2, x_3, x_4\}, \{y_1, y_2, y_3, y_4\}, C)$, where

$$C = \begin{pmatrix} 0 & 0 & 1/3 & 2/3 \\ 0 & 4/5 & 0 & 1/5 \\ 4/7 & 0 & 2/7 & 1/7 \\ 4/9 & 0 & 4/9 & 1/9 \end{pmatrix}$$

Then, if the prior is $\pi = (3/16, 5/16, 7/32, 9/32)$, we get the following joint matrix:

$$p_{XY} = \begin{pmatrix} 0 & 0 & 1/16 & 1/8 \\ 0 & 1/4 & 0 & 1/16 \\ 1/8 & 0 & 1/16 & 1/32 \\ 1/8 & 0 & 1/8 & 1/32 \end{pmatrix}$$

By summing the columns of the joint matrix we obtain the output distribution $p_Y = (1/4, 1/4, 1/4, 1/4)$ which happens to be uniform in this case. And by normalizing the columns of the joint matrix we get four posterior distributions:

$$\begin{aligned} p_{X|y_1} &= (0, 0, 1/2, 1/2) \\ p_{X|y_2} &= (0, 1, 0, 0) \\ p_{X|y_3} &= (1/4, 0, 1/4, 1/2) \\ p_{X|y_4} &= (1/2, 1/4, 1/8, 1/8) \end{aligned}$$

□

[1] We will write subscripts on p when necessary to avoid ambiguity.

2.2. Dynamic consumption of secrecy

One view of the consumption of secrecy is *dynamic*, considering the change of secrecy when the adversary observes a *particular* channel output during an individual run of the channel.

Definition 3. Given channel $(\mathcal{X}, \mathcal{Y}, C)$ and prior π, the *dynamic min-entropy leakage* associated with output y is the decrease in min-entropy caused by y:

$$\mathcal{L}^{dynamic}(\pi, C, y) = H_\infty(\pi) - H_\infty(p_{X|y}).$$

Equivalently, it is the logarithm of the factor by which y increases the vulnerability:

$$\mathcal{L}^{dynamic}(\pi, C, y) = \log \frac{V(p_{X|y})}{V(\pi)}.$$

□

For instance, in Example 2, observing output y_2 reveals that the secret must be x_2, since $p_{X|y_2} = (0, 1, 0, 0)$. In this case, we can say that the secrecy of X decreases from $H_\infty(\pi) = -\log 5/16 \approx 1.678$ down to $H_\infty(p_{X|y_2}) = -\log 1 = 0$. Hence we have $\mathcal{L}^{dynamic}(\pi, C, y_2) \approx 1.678$.

Moreover, if the adversary can run the channel multiple times, using the same value of X each time, then it can repeatedly refine the posterior distribution on $\mathcal{X}$, by using the posterior distribution from one output as the prior distribution for the next run.

Example 3. If we run repeatedly the channel from Example 2 and observe the output sequence y_3, y_1 and y_3, the distribution on $\mathcal{X}$ is refined as follows:

$$\begin{aligned} p_X &= (3/16, 5/16, 7/32, 9/32) \\ p_{X|y_3} &= (1/4, 0, 1/4, 1/2) \\ p_{X|y_3,y_1} &= (0, 0, 9/23, 14/23) \\ p_{X|y_3,y_1,y_3} &= (0, 0, 81/277, 196/277) \end{aligned}$$

Respectively, the min-entropy for these distributions is:

$$\begin{aligned} H_\infty(p_X) &= -\log(5/16) \approx 1.678 \\ H_\infty(p_{X|y_3}) &= -\log(1/2) = 1 \\ H_\infty(p_{X|y_3,y_1}) &= -\log(14/23) \approx 0.716 \\ H_\infty(p_{X|y_3,y_1,y_3}) &= -\log(196/277) \approx 0.708 \end{aligned}$$

These decreasing entropy values reflect the gradual consumption of the secrecy of X through the repeated observations. □

While this dynamic view of leakage is natural, it suffers from some significant drawbacks. First, dynamic (run-time) policy enforcement may actually reveal information about the secret. For instance, an execution monitor could track the amount of

remaining min-entropy, verifying that it stays above some threshold. But if a run produces an output that leaks too much, what can the monitor do? It might try to respond by aborting the execution, but the very act of aborting might in itself leak a lot of information to the adversary. For example, in the case of a password checker, aborting the execution when the adversary enters the correct password reveals the entire password to the adversary (and also makes the password checker useless).

Moreover, under the dynamic view it turns out that min-entropy need not decrease monotonically—it can actually *increase* as a result of an observation.

Example 4. Let channel $(\mathcal{X}, \mathcal{Y}, C)$ be as follows:

$$C = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}$$

and suppose that $\pi = [9/10, 1/40, 1/40, 1/40]$. Then $V(\pi) = 9/10$ and $H_\infty(\pi) \approx 0.152$. However, after observing output y_2 the vulnerability *decreases* to $V(p_{X|y_2}) = 1/4$ and the secrecy *increases* to $H_\infty(p_{X|y_2}) = 2$. Hence we get *negative leakage*:

$$\mathcal{L}^{dynamic}(\pi, C, y_2) = \log \frac{1/4}{9/10} = \log 5/18 \approx -1.848.$$

□

A real-world scenario corresponding to this example is the case of a doctor trying to diagnose an unknown disease. Based on the symptoms, there might be only one likely diagnosis, making the prior "secrecy" small. But if a medical test refutes that diagnosis, then the doctor is left with no idea of the disease, making the posterior "secrecy" large.

We conclude that, under the dynamic perspective, min-entropy does not behave as a reasonable resource. For this reason, we henceforth restrict our attention to *static* viewpoints.

2.3. Worst-case consumption of secrecy

Perhaps the most straightforward way to achieve a static measure of the secrecy consumption of a channel is to consider the *worst-case* loss of min-entropy, over all channel outputs. (Such worst-case measures were considered previously by Köpf and Basin [6], although using guessing entropy rather than min-entropy, and by Mardziel et al. [17]; we will discuss the latter paper further in Section 5.)

We first define worst-case posterior vulnerability:

Definition 4. Given prior π and channel C, the *worst-case posterior vulnerability* is given by

$$V^{worst}(\pi, C) = \max_{y \in \mathcal{Y}} V(p_{X|y}).$$

□

Notice that the worst-case posterior vulnerability is simply the maximum posterior probability over all the inputs and outputs:

$$V^{worst}(\pi, C) = \max_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x|y).$$

We define the *worst-case posterior min-entropy* by taking the negative logarithm, as before:

Definition 5. $H_\infty^{worst}(\pi, C) = -\log V^{worst}(\pi, C)$. □

Finally, we define *worst-case min-entropy leakage* as the difference between the prior and posterior min-entropy; equivalently, it is the logarithm of the ratio of the posterior and prior vulnerability:

Definition 6.

$$\mathcal{L}^{worst}(\pi, C) = H_\infty(\pi) - H_\infty^{worst}(\pi, C) = \log \frac{V^{worst}(\pi, C)}{V(\pi)}.$$

□

Worst-case min-entropy leakage is sometimes a useful measure, but it has the serious drawback that it is highly sensitive to a channel's worst output, even if that output is very unlikely. For instance a password checker

```
if (X == g) Y = 1; else Y = 0;
```

has the *same* worst-case leakage as a program that always leaks the entire secret:

```
Y = X;
```

To see this, notice that under the password checker $p_{X|1}(g) = p(X = g|Y = 1) = 1$, which means that $V(p_{X|1}) = 1$.

2.4. Average-case consumption of secrecy

To minimize the sensitivity to unlikely "bad" outputs, it seems generally more useful to define leakage by considering the *average* posterior vulnerability over all outputs.

Definition 7. Given prior π and channel C, the *average posterior vulnerability* is given by

$$V^{average}(\pi, C) = \sum_{y \in \mathcal{Y}} p(y) V(p_{X|y}).$$

□

Note that average posterior vulnerability can be calculated easily as the sum of the column maximums of the joint matrix; it can also be calculated directly from the prior

π and channel matrix C:

$$\begin{aligned} V^{average}(\pi, C) &= \sum_{y \in \mathcal{Y}} p(y) V(p_{X|y}) \\ &= \sum_{y \in \mathcal{Y}} p(y) \max_{x \in \mathcal{X}} p(x|y) \\ &= \sum_{y \in \mathcal{Y}} \max_{x \in \mathcal{X}} p(x, y) \\ &= \sum_{y \in \mathcal{Y}} \max_{x \in \mathcal{X}} \pi[x] C[x, y]. \end{aligned}$$

Now we define entropy and leakage as before:

Definition 8. The *average posterior min-entropy* is given by

$$H_\infty^{average}(\pi, C) = -\log V^{average}(\pi, C),$$

and the *average min-entropy leakage* is given by

$$\mathcal{L}^{average}(\pi, C) = H_\infty(\pi) - H_\infty^{average}(\pi, C) = \log \frac{V^{average}(\pi, C)}{V(\pi)}.$$

□

Note that $V(\pi)$, $V^{average}(\pi, C)$, $H_\infty^{average}(\pi, C)$, and $\mathcal{L}^{average}(\pi, C)$ are the leakage measures that are advocated by Smith [9, 12] and (with a slight variation) by Braun, Chatzikokolakis, and Palamidessi [11].[2]

Example 5. Returning to Example 2, we have

$$V^{average}(\pi, C) = \sum_{y \in \mathcal{Y}} \max_{x \in \mathcal{X}} p(x, y) = 1/8 + 1/4 + 1/8 + 1/8 = 5/8.$$

Hence we get

$$\mathcal{L}^{average}(\pi, C) = \log \frac{5/8}{5/16} = \log 2 = 1,$$

reflecting the fact that the average vulnerability is doubled by C.

Note that adversary's best prior guess for X is x_2, since $\pi[x_2] = 5/16$ is maximal. Interestingly, the maximums in the calculation of $V^{average}$ reflect the adversary's best guess about X, given each output. On output y_1, the best guess is x_3 (or x_4); on output y_2, the best guess is x_2; on output y_3, the best guess is x_4; and on output y_4, the best guess is x_1. □

In contrast to what we observed about dynamic posterior vulnerability, the average posterior vulnerability cannot be less than the prior vulnerability:

[2]We deviate, however, from the notation $V(X)$, $V(X|Y)$, $H_\infty(X|Y)$, and $\mathcal{L}_{XY}$ used in [9, 12]. Instead we follow [18] in adopting a notation that makes explicit the dependence on X's prior distribution.

Theorem 1. *For any π and C, $V(\pi) \leq V^{average}(\pi, C)$. Hence $\mathcal{L}^{average}(\pi, C) \geq 0$.*

Proof. We have

$$V(\pi) = \max_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x, y) \leq \sum_{y \in \mathcal{Y}} \max_{x \in \mathcal{X}} p(x, y) = V^{average}(\pi, C).$$

□

Also, as expected, the average posterior vulnerability is upper bounded by the worst-case posterior vulnerability:

Theorem 2. *For any π and C, $V^{average}(\pi, C) \leq V^{worst}(\pi, C)$.*

Proof.

$$V^{average}(\pi, C) = \sum_{y \in \mathcal{Y}} p(y) V(p_{X|y}) \leq \sum_{y \in \mathcal{Y}} p(y) V^{worst}(\pi, C) = V^{worst}(\pi, C).$$

□

As an immediate consequence, worst-case min-entropy leakage must be non-negative and must be an upper bound on average min-entropy leakage.

2.5. A case study: the Crowds protocol

We conclude this section with a case study in which we calculate the average and worst-case min-entropy leakage associated with the Crowds protocol of Reiter and Rubin [1], which was briefly described in Section 1.

We assume that there are n honest users and c collaborators in the Crowd, whose total size is m, where $m = n + c$. An initiator wishing to send a message to the server first sends it to a randomly-chosen forwarder, possibly itself, each with probability $\frac{1}{m}$. With probability p_f, each forwarder will forward the message again to another randomly-chosen forwarder, possibly itself; with probability $1 - p_f$, it sends the message to the server. If some user forwards a message to a collaborator, then that user is said to be *detected*. Once a user is detected or the message reaches the server, the protocol stops. We assume that the forwarding probability p_f satisfies $0 \leq p_f < 1$, since if $p_f = 1$ then the message can never reach the server.

As a channel, the set of secret inputs $\mathcal{X} = \{u_1, \ldots, u_n\}$, where u_i means that user i is the initiator. The set of observable outputs $\mathcal{Y} = \{d_1, \ldots, d_n, s\}$, where d_i means that user i was detected, and s means that the message reached the server without ever going to a collaborator.

Now we calculate the channel matrix C, using techniques like those in [1]. To compute $p(s|u_i)$, we observe that output s occurs iff the message is forwarded one or

more times among honest users, and then is sent to the server:

$$\begin{aligned} p(s|u_i) &= \sum_{j=0}^{\infty} \frac{n}{m} \left(p_f \frac{n}{m}\right)^j (1-p_f) \\ &= \frac{n}{m}(1-p_f)\left(\frac{1}{1-\frac{p_f n}{m}}\right) \\ &= \frac{n-p_f n}{m-p_f n} \end{aligned}$$

Let D be the event that *some* user is detected. We have

$$p(D|u_i) = 1 - p(s|u_i) = 1 - \frac{n-p_f n}{m-p_f n} = \frac{c}{m-p_f n}$$

Now let D_2 be the event that some user is detected after two or more steps. We have

$$p(D_2|u_i) = p(D|u_i) - \frac{c}{m} = \frac{c(m-(m-p_f n))}{m(m-p_f n)} = \frac{cp_f n}{m(m-p_f n)}$$

Note that *every* user is equally likely to be detected after two or more steps, since forwarders are all randomly chosen. Hence we can compute $p(d_i|u_i)$ as follows:

$$p(d_i|u_i) = \frac{c}{m} + \frac{1}{n}\frac{cp_f n}{m(m-p_f n)} = \frac{c(m-p_f(n-1))}{m(m-p_f n)}$$

Finally, we can calculate $p(d_j|u_i)$, for $j \neq i$, by noting that user j cannot be detected in one step when i is the initiator:

$$p(d_j|u_i) = \frac{cp_f}{m(m-p_f n)}$$

Assume now that the prior distribution π is uniform, so that each user has probability $\frac{1}{n}$ of being the initiator. This means that the prior vulnerability is

$$V(\pi) = \frac{1}{n}.$$

We now determine the posterior vulnerability for each output. First, since $p(s|u_i)$ is the same, for every i, it is immediate that $p_{X|s}$ is a uniform distribution. Hence

$$V(p_{X|s}) = \frac{1}{n}.$$

Next consider $p_{X|d_i}$, for any i. We have seen that column d_i of the channel matrix contains its largest entry at $p(d_i|u_i)$, and $n-1$ smaller entries $p(d_i|u_j)$, for $j \neq i$. Moreover the sum of the column can be calculated easily, using the symmetries of the matrix:

$$\sum_{j=1}^{n} p(d_i|u_j) = \sum_{j=1}^{n} p(d_j|u_i) = p(D|u_i)$$

Hence we have

$$V(p_{X|d_i}) = \frac{p(d_i|u_i)}{p(D|u_i)} = \frac{c(m - p_f(n-1))}{m(m-p_f n)} \frac{m - p_f n}{c} = \frac{m - p_f(n-1)}{m}.$$

Note that $V(p_{X|d_i})$ does not depend on i.

Now we calculate the average posterior vulnerability:

$$\begin{aligned} V^{average}(\pi, C) &= p(s)V(p_{X|s}) + \sum_{i=1}^{n} p(d_i)V(p_{X|d_i}) \\ &= \frac{n - p_f n}{m - p_f n}\frac{1}{n} + \left(1 - \frac{n - p_f n}{m - p_f n}\right)\frac{m - p_f(n-1)}{m} \\ &= \frac{m - p_f n - p_f c + cm - cp_f n + cp_f}{m(m - p_f n)} \\ &= \frac{m(c+1) - p_f n(c+1)}{m(m - p_f n)} \\ &= \frac{c+1}{c+n}. \end{aligned}$$

And from this, we can calculate the average min-entropy leakage:

$$\mathcal{L}^{average}(\pi, C) = \log \frac{(c+1)n}{c+n}$$

Remarkably, we see that the average posterior vulnerability and the average min-entropy leakage *do not depend* on the forwarding probability p_f.

But this is not to say that p_f is irrelevant to the effectiveness of the Crowds protocol. For example, if $p_f = 0$, then on output d_i the adversary *knows* that user i was the initiator. For this reason it is interesting to consider as well the worst-case posterior vulnerability:

$$V^{worst}(\pi, C) = \frac{m - p_f(n-1)}{m} = \frac{c + (1-p_f)n + p_f}{c+n}$$

and the worst-case min-entropy leakage:

$$\mathcal{L}^{worst}(\pi, C) = \log \frac{(c + (1-p_f)n + p_f)n}{c+n}.$$

Note that the worst-case vulnerability is maximized by $p_f = 0$, which gives a vulnerability of 1. It is minimized by choosing p_f close to 1, but of course this has the drawback of making message transmission slower and also increasing the probability that the message will go at some point to a collaborator.

The worst-case vulnerability analysis of Crowds gives useful insight, showing the importance of carefully choosing information flow measures based the characteristics of the scenario being studied. However, considering that worst-case vulnerability is highly sensitive to a channel's worst output (no matter how unlikely it may be), it usually seems more informative to focus on the average posterior vulnerability and average min-entropy leakage. For this reason, in the rest of this paper we will use the average-case measures by default, and we will henceforth drop the superscripts in $V^{average}$, $H_\infty^{average}$, and $\mathcal{L}^{average}$.

3. Properties of min-entropy leakage

In this section, we review the main properties of (average) min-entropy leakage that have been established in the literature. We begin by discussing *min-capacity*:

Definition 9. The *min-capacity* of channel C is the maximum min-entropy leakage over all priors π:

$$\mathcal{ML}(C) = \sup_{\pi} \mathcal{L}(\pi, C).$$

Min-capacity is a useful measure in situations where the prior π is unknown. Moreover, in contrast to Shannon capacity, min-capacity is easy to calculate, as it is simply the logarithm of the sum of the column maximums of C; this result was proved in [11, 10].

Theorem 3. $\mathcal{ML}(C) = \log \sum_{y \in \mathcal{Y}} \max_{x \in \mathcal{X}} C[x, y]$*, and it is realized on a uniform prior* π.

Proof. For any prior π, we have

$$\begin{aligned}
\mathcal{L}(\pi, C) &= \log \frac{V(\pi, C)}{V(\pi)} \\
&= \log \frac{\sum_{y \in \mathcal{Y}} \max_{x \in \mathcal{X}} (\pi[x] C[x, y])}{\max_{x \in \mathcal{X}} \pi[x]} \\
&\leq \log \frac{\sum_{y \in \mathcal{Y}} \max_{x \in \mathcal{X}} C[x, y] (\max_{x \in \mathcal{X}} \pi[x])}{\max_{x \in \mathcal{X}} \pi[x]} \\
&= \log \sum_{y \in \mathcal{Y}} \max_{x \in \mathcal{X}} C[x, y]
\end{aligned}$$

The upper bound is realized when π is uniform. (It can also be realized on nonuniform π, provided that some proper subset of the rows of C includes at least one maximum from each column.) □

Theorem 3 gives two useful corollaries:

Corollary 1. *If C is deterministic, then $\mathcal{ML}(C)$ is the logarithm of the number of feasible outputs.*

Corollary 2. $\mathcal{ML}(C) = 0$ *iff the rows of C are identical.*

If we turn from min-capacity to min-entropy leakage, we find that leakage of 0 is more subtle. In fact, we have $\mathcal{L}(\pi, C) = 0$ whenever the adversary's best guess is unaffected by the output y. This can sometimes be surprising, as in the following example from [12], which illustrates the so-called *base-rate fallacy*.

Example 6. Suppose that C is the channel matrix of a good, but imperfect, test for cancer:

	positive	*negative*
cancer	0.90	0.10
no cancer	0.07	0.93

Moreover, suppose that for the population under consideration (say, age 40–50, no symptoms, no family history) the prior π is

$$\pi[\textit{cancer}] = 0.008 \quad \pi[\textit{no cancer}] = 0.992$$

Then, although the channel might appear to be quite reliable, we find that the min-entropy leakage is 0. For we find that the joint matrix p_{XY} is

	positive	*negative*
cancer	0.00720	0.00080
no cancer	0.06944	0.92256

Hence the sum of the column maximums coincides with $\pi[\textit{no cancer}]$, since both column maximums occurs in the *no cancer* row. This implies that $V(\pi) = V(\pi, C)$, giving $\mathcal{L}(\pi, C) = 0$. Operationally, this reflects the fact that the adversary should guess *no cancer*, regardless of whether the test was *positive* or *negative*. (In particular, $p(\textit{cancer}|\textit{positive}) \approx 0.094$, which is much greater than $p(\textit{cancer}) = 0.008$, but still much less than 0.500.) □

Because min-capacity is relatively simple to calculate, particularly in the case of deterministic channels, it may facilitate the design of practical quantitative information flow analyses. For whenever we can show that min-capacity is small, we know that min-entropy leakage must also be small, whatever the prior π may be. On the other hand, for a particular prior π we may find that $\mathcal{L}(\pi, C)$ is far less than $\mathcal{ML}(C)$. So in cases where the prior π is known, it is more precise to use the min-entropy leakage with respect to π.

We should finally emphasize that vulnerability is information theoretic, rather than computational. To see this, consider the following example from [12]. Let C be a channel that takes as input a uniformly-distributed 100-digit prime p, and that outputs pq, where q is a uniformly-distributed 101-digit prime. Then the posterior vulnerability $V(\pi, C) = 1$, since each column of the channel matrix has a unique nonzero entry. Hence C's min-entropy leakage exceeds 322 bits, since (by the prime number theorem) $V(\pi) < 10^{-97}$. Nevertheless, *finding* the input given the output requires factoring a very large number into the product of two roughly equally-sized primes, a problem strongly believed to be computationally hard.

4. Min-entropy consumption in combined channels

We now turn our attention to the behavior of min-entropy leakage when we combine multiple channels. Ideally, we would be able to compute the leakage of a system in terms of the leakage of its constituents. However, such closed formulas are typically not possible. Still, as the following subsections describe, we can derive a number of useful bounds depending on what mechanism we use to combine the channels.

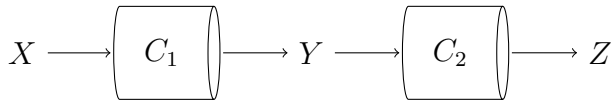

Figure 2: Cascade of channels C_1 and C_2

4.1. Cascades of channels

We begin by exploring channels in *cascade*, where the output of the first channel is used as input to the second channel, as depicted in Figure 2. Preliminary versions of the results in this section appeared in [16].

Following [16], we formalize the behavior of a cascade of two channels in terms of the joint distribution of the random variables corresponding to its input, its intermediate output, and its final output.

Definition 10. The *cascade* of channels $(\mathcal{X}, \mathcal{Y}, C_1)$ and $(\mathcal{Y}, \mathcal{Z}, C_2)$ under prior distribution π has joint distribution p_{XYZ}, where

$$p_{XYZ}(x, y, z) = \pi[x]C_1[x, y]C_2[y, z].$$

□

This definition tells us that in a cascade of channels the final output Z depends only on the intermediate output Y and not on the secret input X. We express this formally with the following theorem.

Theorem 4. *Let $(\mathcal{X}, \mathcal{Z}, C)$ be the cascade of $(\mathcal{X}, \mathcal{Y}, C_1)$ and $(\mathcal{Y}, \mathcal{Z}, C_2)$. Let π be the prior distribution on $\mathcal{X}$. Then, for any values $x \in \mathcal{X}$, $y \in \mathcal{Y}$, and $z \in \mathcal{Z}$ such that $p(x, y) > 0$, we have*

$$p(z|x, y) = p(z|y).$$

Proof. We have

$$\begin{aligned} p(z|x, y) &= \frac{p(x, y, z)}{p(x, y)} \\ &= \frac{\pi[x]C_1[x, y]C_2[y, z]}{\sum_{z \in \mathcal{Z}} \pi[x]C_1[x, y]C_2[y, z]} \\ &= \frac{\pi[x]C_1[x, y]C_2[y, z]}{\pi[x]C_1[x, y]} \\ &= C_2[y, z] \\ &= p(z|y) \end{aligned}$$

□

Moreover, given the joint distribution p_{XYZ} in a cascade, we can infer the conditional probability $p(z|x)$ of the final output given the secret input whenever it is

defined:

$$\begin{aligned} p(z|x) &= \frac{p(x,z)}{\pi[x]} \\ &= \frac{\sum_{y\in\mathcal{Y}} p(x,y,z)}{\pi[x]} \\ &= \frac{\sum_{y\in\mathcal{Y}} \pi[x]C_1[x,y]C_2[y,z]}{\pi[x]} \\ &= \sum_{y\in\mathcal{Y}} C_1[x,y]C_2[y,z] \\ &= C_1C_2[x,z]. \end{aligned}$$

This motivates the following definition of the channel matrix of a cascade as the product of the channel matrices of its links.

Definition 11. The *cascade* of channels $(\mathcal{X},\mathcal{Y},C_1)$ and $(\mathcal{Y},\mathcal{Z},C_2)$ is the channel $(\mathcal{X},\mathcal{Z},C_1C_2)$. □

Now that we understand how to build the channel matrix of the combined channel, we are interested in finding a relationship between the min-entropy leakage of the cascade and the min-entropy leakage of its constituents.

An important property of cascading is that the first link behaves as a bottleneck for secrecy consumption. If the first link consumes little secrecy, then the cascade must also consume little secrecy. Theorem 5 formalizes this property by establishing that a cascade of channels cannot leak more information than its first link.[3] This theorem was proved in [16] but we provide a more compact proof.

Theorem 5. *Let $(\mathcal{X},\mathcal{Z},C)$ be the cascade of $(\mathcal{X},\mathcal{Y},C_1)$ and $(\mathcal{Y},\mathcal{Z},C_2)$. Then for any prior distribution π, we have*

$$\mathcal{L}(\pi,C) \leq \mathcal{L}(\pi,C_1).$$

[3]Theorem 5 is somewhat reminiscent of Maurer and Massey's cryptographic result [19], where they proved that a cascade cipher is at least as difficult to break as the first component cipher.

Proof. For any prior π,

$$\begin{aligned}
V(\pi, C) &= \sum_{z \in \mathcal{Z}} \max_{x \in \mathcal{X}} C[x, z]\pi[x] \\
&= \sum_{z \in \mathcal{Z}} \max_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} C_1[x, y]C_2[y, z]\pi[x] \\
&\leq \sum_{z \in \mathcal{Z}} \sum_{y \in \mathcal{Y}} \max_{x \in \mathcal{X}} C_1[x, y]C_2[y, z]\pi[x] \\
&= \sum_{y \in \mathcal{Y}} \sum_{z \in \mathcal{Z}} C_2[y, z] \max_{x \in \mathcal{X}} C_1[x, y]\pi[x] \\
&= \sum_{y \in \mathcal{Y}} \left(\sum_{z \in \mathcal{Z}} C_2[y, z] \right) \left(\max_{x \in \mathcal{X}} C_1[x, y]\pi[x] \right) \\
&= \sum_{y \in \mathcal{Y}} \max_{x \in \mathcal{X}} C_1[x, y]\pi[x] \\
&= V(\pi, C_1).
\end{aligned}$$

Then, it follows that

$$\mathcal{L}(\pi, C) = \log \frac{V(\pi, C)}{V(\pi)} \leq \log \frac{V(\pi, C_1)}{V(\pi)} = \mathcal{L}(\pi, C_1).$$

□

Curiously, however, the leakage of a cascade may exceed the leakage of the second link as the following example illustrates.

Example 7.

$$C_1 = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad C_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad C = C_1C_2 = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$$

If $\pi = [1/3, 1/3, 1/3]$, then $p_Y = p_Z = (2/3, 1/3)$, and:

$$\mathcal{L}(\pi, C) = \log \frac{V(\pi, C)}{V(\pi)} = \log \frac{2/3}{1/3} = \log 2$$

$$\mathcal{L}(p_Y, C_2) = \log \frac{V(p_Y, C_2)}{V(p_Y)} = \log \frac{1}{2/3} = \log 3/2$$

Hence, the cascade C leaks more than the second link C_2. Moreover, the knowledge of the output of C_2 could not have possibly doubled the prior vulnerability as in the case of C, since doubling a vulnerability of $2/3$ would mean a posterior vulnerability of $4/3$, a value that falls outside the range of valid probabilities. □

To understand why the second link behaves differently, notice that when we compare the leakages $\mathcal{L}(\pi, C)$ and $\mathcal{L}(\pi, C_1)$ in Theorem 5, we are comparing $\log \frac{V(\pi,C)}{V(\pi)}$ and $\log \frac{V(\pi,C_1)}{V(\pi)}$, which reduces to comparing the numerators $V(\pi, C)$ and $V(\pi, C_1)$. In contrast, when we try to compare $\mathcal{L}(\pi, C)$ and $\mathcal{L}(p_Y, C_2)$, we are actually comparing $\log \frac{V(\pi,C)}{V(\pi)}$ and $\log \frac{V(p_Y,C_2)}{V(p_Y)}$, which differ in both the numerators and denominators.

Fortunately, however, the behavior of min-capacity in a cascade is more intuitive. Both the first and second links constraint the amount of information that can flow through the cascade.

Theorem 6. *If $(\mathcal{X}, \mathcal{Z}, C)$ is the cascade of $(\mathcal{X}, \mathcal{Y}, C_1)$ and $(\mathcal{Y}, \mathcal{Z}, C_2)$, then*

$$\mathcal{ML}(C) \leq \min(\mathcal{ML}(C_1), \mathcal{ML}(C_2)).$$

Proof. By Theorem 5 and the definition of min-capacity we know that for any prior π,

$$\mathcal{L}(\pi, C) \leq \mathcal{L}(\pi, C_1) \leq \mathcal{ML}(C_1).$$

Hence

$$\mathcal{ML}(C) = \sup_{\pi} \mathcal{L}(\pi, C) \leq \mathcal{ML}(C_1).$$

To obtain the upper bound with respect to C_2, we observe that

$$\begin{aligned}
\mathcal{ML}(C) &= \log \sum_{z \in \mathcal{Z}} \max_{x \in \mathcal{X}} C[x, z] \\
&= \log \sum_{z \in \mathcal{Z}} \max_{x \in \mathcal{X}} (C_1 C_2)[x, z] \\
&= \log \sum_{z \in \mathcal{Z}} \max_{x \in \mathcal{X}} \left(\sum_{y \in \mathcal{Y}} C_1[x, y] C_2[y, z] \right) \\
&\leq \log \sum_{z \in \mathcal{Z}} \max_{x \in \mathcal{X}} \left(\sum_{y \in \mathcal{Y}} C_1[x, y] \max_{y' \in \mathcal{Y}} C_2[y', z] \right) \\
&= \log \sum_{z \in \mathcal{Z}} \max_{x \in \mathcal{X}} \left(\max_{y' \in \mathcal{Y}} C_2[y', z] \right) \\
&= \log \sum_{z \in \mathcal{Z}} \max_{y' \in \mathcal{Y}} C_2[y', z] \\
&= \mathcal{ML}(C_2).
\end{aligned}$$

Since we have both $\mathcal{ML}(C) \leq \mathcal{ML}(C_1)$ and $\mathcal{ML}(C) \leq \mathcal{ML}(C_2)$, we have proved that $\mathcal{ML}(C) \leq \min(\mathcal{ML}(C_1), \mathcal{ML}(C_2))$. □

A corollary to Theorem 6 is that the capacity of a cascade cannot exceed the logarithm of the number of intermediate outputs; this result was first proved as a standalone theorem by Köpf and Smith [10].

Corollary 3. *If $(\mathcal{X}, \mathcal{Z}, C)$ is the cascade of $(\mathcal{X}, \mathcal{Y}, C_1)$ and $(\mathcal{Y}, \mathcal{Z}, C_2)$, then*

$$\mathcal{ML}(C) \leq \log |\mathcal{Y}|.$$

Proof. Since (by Theorem 3) the capacity of a channel is upper bounded by the logarithm of the number of outputs of the channel, it follows from Theorem 6 that $\mathcal{ML}(C) \leq \mathcal{ML}(C_1) \leq \log |\mathcal{Y}|$. □

It is interesting to note that Theorem 5 can be viewed as the min-entropy analogue to the classic *data-processing inequality* for *mutual information*, which is the measure of leakage obtained by measuring uncertainty using Shannon entropy. The data-processing inequality tells us that post-processing can only destroy information. This inequality is generally formulated [20] in terms of random variables with fixed probability distributions, but we can provide an alternative and arguably more expressive formulation in terms of cascades. In the following theorems we use the notation $I(X;Y)$ for the mutual information (Shannon leakage) of random variables X and Y.

Theorem 7. *Let $(\mathcal{X}, \mathcal{Z}, C)$ be the cascade of $(\mathcal{X}, \mathcal{Y}, C_1)$ and $(\mathcal{Y}, \mathcal{Z}, C_2)$.*
Then for any prior distribution π, we have

$$I(X;Z) \leq I(X;Y).$$

Proof. From Theorem 4, we know that for any values $x \in \mathcal{X}$, $y \in \mathcal{Y}$, and $z \in \mathcal{Z}$ such that $p(x,y) > 0$, we have $p(z|x,y) = p(z|y)$.

The proof immediately follows, since the data-processing inequality [20, p. 26] tells us that whenever the previous condition holds, we have $I(X;Z) \leq I(X;Y)$. □

The properties of mutual information in the presence of cascades differ from those of min-entropy in that leakage bounds can be established for both links. Concretely, the Shannon leakage of the second link in a cascade is an upper bound for the Shannon leakage of the cascade.

Corollary 4. *Let $(\mathcal{X}, \mathcal{Z}, C)$ be the cascade of $(\mathcal{X}, \mathcal{Y}, C_1)$ and $(\mathcal{Y}, \mathcal{Z}, C_2)$. Then for any prior distribution π, we have*

$$I(X;Z) \leq I(Y;Z).$$

A key to proving this corollary is the symmetry of mutual information: $I(X;Z) = I(Z;X)$. Considering that min-entropy leakage is not symmetric in general, it is easier to understand why a comparable property does not hold for min-entropy leakage.

4.2. Repeated independent runs of a channel

We now turn our attention to the behavior of min-entropy leakage when repeated independent runs of a channel are allowed. We can model this scenario with a channel that receives an input in $\mathcal{X}$, feeds it to multiple copies of a channel $(\mathcal{X}, \mathcal{Y}, C)$, and collects the outputs from each run into a tuple in $\mathcal{Y}^n$. Figure 3 illustrates this combination of channels. The effects of repeated independent runs of a channel on min-entropy were first studied by Köpf and Smith [10] within the concrete scenario of a timing attack against a cryptosystem that implements input blinding and bucketing.

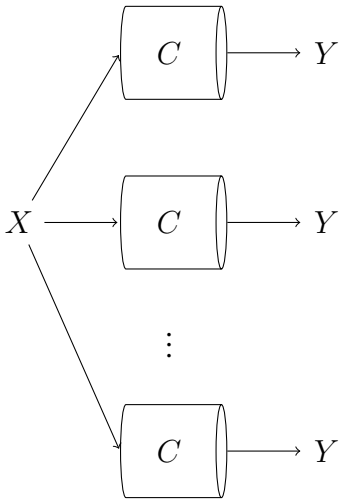

Figure 3: Repeated independent runs of channel C

The resulting combined channel $C^{(n)}$ is a channel from $\mathcal{X}$ to $\mathcal{Y}^n$ and the entries $[x, (y_1, \ldots, y_n)]$ of its matrix give the probability of producing the sequence of outputs $(y_1, \ldots, y_n)$ when the input to the channel is x. Since the probabilities of producing a particular output in each run are conditionally independent given a secret input, we calculate entries of the combined channel matrix by multiplying the individual conditional probabilities.

Definition 12. The matrix of the channel $(\mathcal{X}, \mathcal{Y}^n, C^{(n)})$ that results from n independent runs of channel $(\mathcal{X}, \mathcal{Y}, C)$ is given by

$$C^{(n)}[x, (y_1, \ldots, y_n)] = \prod_{i=1}^{n} C[x, y_i].$$

Note that the study of repeated independent runs of a channel makes sense only in the case a probabilistic channel C; otherwise given a secret input every run (of channel C) would return the same output.

Curiously, even when $\mathcal{L}(\pi, C) = 0$, the leakage under two repeated independent runs $\mathcal{L}(\pi, C^{(2)})$ can be greater than zero.

Example 8. Let the matrix of channel $(\mathcal{X}, \mathcal{Y}, C)$ be

$$C = \begin{pmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{pmatrix}$$

Then, if the prior is given by $\pi = (0.05, 0.95)$ we get $\mathcal{L}(\pi, C) = \log \frac{0.95}{0.95} = 0$. However, if we consider two independent runs of C, the resulting combined matrix $C^{(2)}$ satisfies $C^{(2)}[x, (y_1, y_2)] = C[x, y_1]C[x, y_2]$, giving

$$C^{(2)} = \begin{pmatrix} 0.81 & 0.09 & 0.09 & 0.01 \\ 0.01 & 0.09 & 0.09 & 0.81 \end{pmatrix}$$

But now, $\mathcal{L}(\pi, C^{(2)}) = \log \frac{0.981}{0.95} > 0$. □

It can be shown that the vulnerability of the secret cannot decrease after each additional independent run of channel C. Similarly, neither the leakage nor the capacity of $C^{(n)}$ can decrease as n grows. It is then of interest to understand how fast the capacity $\mathcal{ML}(C^{(n)})$ increases as more repetitions are allowed.

Boreale et al. [21] proved that $\mathcal{ML}(C^{(n)})$ converges exponentially quickly to the logarithm of the number of distinct rows in C. (Intuitively, distinct rows of the channel matrix can be distinguished by repeatedly sampling the output.) More precisely, their Theorem 1 restricted to the case of a uniform prior (which realizes min-capacity) can be restated in the following form:

Theorem 8. *Let K denote the number of distinct rows in C. Then there is an $\epsilon > 0$ such that*

$$\log K \geq \mathcal{ML}(C^{(n)}) \geq \log K + \log r(n),$$

where $r(n) = 1 - (n+1)^{|\mathcal{Y}|} 2^{-n\epsilon}$.

However, the min-capacity $\mathcal{ML}(C^{(n)})$ grows only logarithmically with respect to the number of runs n. This result was first proved by Köpf and Smith [10] within the context of timing attacks against a cryptosystem, but the proof holds in general for any n independent runs channel $C^{(n)}$.

Theorem 9. *For any channel $(\mathcal{X}, \mathcal{Y}, C)$ and number of repetitions n,*

$$\mathcal{ML}(C^{(n)}) \leq |\mathcal{Y}| \log(n+1).$$

Proof. The proof relies on the information-theoretic method of *types*; see for example [15]. The key idea is that, in view of Definition 12, the conditional probability of an output sequence $\bar{y}$ does not depend on the *ordering* of the outputs, but only on the *number of occurrences* of each element of $\mathcal{Y}$ within $\bar{y}$; this information is called the *type* of $\bar{y}$. For example, if $\mathcal{Y} = \{y_1, y_2, y_3, y_4\}$ and $n = 10$, then the output sequence $(y_3, y_2, y_2, y_4, y_2, y_3, y_2, y_2, y_2, y_2)$ has type $(0, 7, 2, 1)$. In general, a type is a length-$|\mathcal{Y}|$ sequence of numbers, each between 0 and n, whose sum is n. We write $t_{\bar{y}}$ to denote the type of $\bar{y}$, $|t_{\bar{y}}|$ to denote the number of sequences with type $t_{\bar{y}}$, and $\mathcal{T}$ to denote the set of all types.

Because two output sequences with the same type have the same conditional probability given the secret, it follows that we can factor $C^{(n)}$ into the cascade of a channel C_1 from $\mathcal{X}$ to $\mathcal{T}$ and a channel C_2 from $\mathcal{T}$ to $\mathcal{Y}^n$. More precisely, we define

$$C_1[x, t_{\bar{y}}] = |t_{\bar{y}}| C^{(n)}[x, \bar{y}]$$

and

$$C_2[t_{\bar{y}}, \bar{y}'] = \begin{cases} \frac{1}{|t_{\bar{y}}|}, & \text{if } \bar{y}' \text{ has type } t_{\bar{y}} \\ 0, & \text{otherwise.} \end{cases}$$

It is easy to see that C_1 and C_2 are well-defined channel matrices and $C^{(n)} = C_1 C_2$.

Hence we can apply Corollary 3 to deduce that $\mathcal{ML}(C^{(n)}) \leq \log |\mathcal{T}|$. Since each type is a length-$|\mathcal{Y}|$ sequence of numbers between 0 and n, we see that $|\mathcal{T}| \leq (n+1)^{|\mathcal{Y}|}$, and the theorem follows. □

The bound on the size of $\mathcal{T}$ in the above proof is quite crude, since it ignores that fact that the numbers in a type must sum to n. Köpf and Smith [10] also showed a tighter bound by calculating $|\mathcal{T}|$ precisely:

Theorem 10. *For any channel $(\mathcal{X}, \mathcal{Y}, C)$ and number of repetitions n,*

$$\mathcal{ML}(C^{(n)}) \leq \log \binom{n + |\mathcal{Y}| - 1}{n}.$$

Proof. We show that

$$|\mathcal{T}| = \binom{n + |\mathcal{Y}| - 1}{n}.$$

Counting $|\mathcal{T}|$ can be viewed as an "Occupancy Problem" as discussed in Section II.5 of Feller [22]. We want to know in how many ways we can place n indistinguishable "balls" (the outputs) into $|\mathcal{Y}|$ "bins" (the possibilities for each output). In general, the number of ways of putting n indistinguishable balls into b bins turns out to be the binomial coefficient

$$\binom{n + b - 1}{n}.$$

To see this, note that each such placement can be represented as a string of n stars (representing the balls) with $b - 1$ bars inserted (representing the boundaries between the bins). For example, with $n = 5$ and $b = 4$, the string

$$**|*||**$$

represents the case when we put 2 balls in the first bin, 1 ball in the second bin, 0 balls in the third bin, and 2 balls in the fourth bin. If the symbols were all distinguishable, then the number of such strings would be $(n + b - 1)!$. But since the n stars and $b - 1$ bars are indistinguishable, then the total number of strings is

$$\frac{(n + b - 1)!}{n!(b - 1)!},$$

which is equal to the above binomial coefficient. □

The following is an example where these bounds are useful to prove formal security guarantees of a system.

Example 9 (Bounding the leakage of timing attacks on cryptosystems). Input blinding and bucketing are countermeasures against timing attacks to public-key cryptosystems. Input blinding consists of randomizing the cyphertext before decryption and de-randomizing it after decryption. Hence, with blinding, the time required to decrypt a cyphertext is a randomized function of the decryption key and is independent of the cyphertext. Bucketing, on the other hand, consists of limiting the decryption operation to take one of only a small number of possible times; this requires sometimes delaying the response of the decryption algorithm.

Köpf and Smith [10] observed that a timing attack against a cryptosystem that implements blinding can be modeled as a repeated independent runs channel $(\mathcal{X}, \mathcal{Y}^n, C^{(n)})$

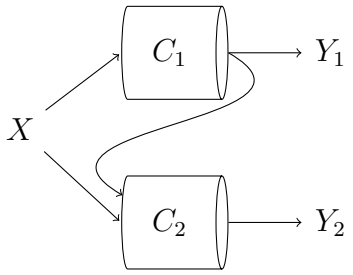

Figure 4: Adaptive composition $C_1 + C_2$

that receives the secret decryption key and outputs a sequence of n timing observations with that key.

Hence, Theorem 10 can be directly applied to establish an upper bound for the min-capacity of the timing attack channel:

$$\mathcal{ML}(C^{(n)}) \leq \log \binom{n + |\mathcal{Y}| - 1}{n}.$$

Moreover, bucketing allows the assumption that $|\mathcal{Y}|$ is small.

Concretely, consider a cryptosystem that implements input blinding and bucketing with a total of 5 buckets ($|\mathcal{Y}| = 5$). Then, the channel of a timing attack with 2^{40} timing observations is given by $(\mathcal{X}, \mathcal{Y}^{2^{40}}, C^{(2^{40})})$ and its capacity is at most

$$\mathcal{ML}(C^{(2^{40})}) \leq \log \binom{2^{40} + 4}{2^{40}} \approx 155.4 \text{ bits}.$$

□

4.3. Adaptive composition $C_1 + C_2$

Barthe and Köpf [23] studied a form of channel composition that is more powerful than cascading. Their model considers two channels, C_1 and C_2, where the second channel receives as input not only the output from C_1 but also the input to C_1, as illustrated by Figure 4.

An example scenario for this kind of composition consists of two privacy preserving randomized queries to the same dataset, where the second query adapts its results depending on the results of the first one. Each of the queries can then be modeled by a probabilistic channel with the special restriction that the channel of the second query must be ready to handle two inputs: the secret dataset and the result from the previous query. The following is a formal definition of adaptive composition:

Definition 13. The adaptive composition of channels $(\mathcal{X}, \mathcal{Y}_1, C_1)$ and $(\mathcal{Y}_1 \times \mathcal{X}, \mathcal{Y}_2, C_2)$ is the channel $(\mathcal{X}, \mathcal{Y}_1 \times \mathcal{Y}_2, C_1 + C_2)$ where

$$(C_1 + C_2)[x, (y_1, y_2)] = C_1[x, y_1] C_2[(y_1, x), y_2].$$

□

Note that this definition makes sense because, when the conditional probabilities are defined, the entries of the matrix $(C_1 + C_2)$ are consistent with the chain rule for probabilities:

$$\begin{aligned} p(y_1, y_2|x) &= (C_1 + C_2)[x, (y_1, y_2)] \\ &= C_1[x, y_1]C_2[(y_1, x), y_2] \\ &= p(y_1|x)p(y_2|y_1, x) \end{aligned}$$

Barthe and Köpf established an upper bound on the min-capacity of $C_1 + C_2$:

Theorem 11. *The min-capacity of the adaptive composition of channels $(\mathcal{X}, \mathcal{Y}_1, C_1)$ and $(\mathcal{Y}_1 \times \mathcal{X}, \mathcal{Y}_2, C_2)$ is at most the sum of the min-capacities of the channels:*

$$\mathcal{ML}(C_1 + C_2) \leq \mathcal{ML}(C_1) + \mathcal{ML}(C_2).$$

Contrast this with the case of cascading, where Theorem 6 tells us that the cascade of two channels cannot consume more min-entropy than *either* channel. Both bounds provide nice guarantees in terms of the behavior of channels when combined under specific conditions. However, we observe that the more restrictive conditions of cascading result in stronger capacity bounds.

We can also think of composing channels C_1 and C_2 *non-adaptively* by having C_2 "ignore" the output from C_1. We denote this kind of composition with $C_1 +_{na} C_2$. Note that letting C_2 ignore the output from C_1 amounts to assuming that Y_1 and Y_2 are conditionally independent given the knowledge of X:

$$p(y_1, y_2|x) = p(y_1|x)p(y_2|x).$$

Hence the formula for the combined channel is unchanged since this conditional independence also implies that

$$p(y_2|x) = p(y_2|y_1, x) = C_2[(y_1, x), y_2].$$

Therefore, we can still establish the same upper bound for the min-capacity of non-adaptive composition $C_1 +_{na} C_2$.

Corollary 5. *The min-capacity of the non-adaptive composition of channels $(\mathcal{X}, \mathcal{Y}_1, C_1)$ and $(\mathcal{Y}_1 \times \mathcal{X}, \mathcal{Y}_2, C_2)$ is at most the sum of the min-capacities of the channels:*

$$\mathcal{ML}(C_1 +_{na} C_2) \leq \mathcal{ML}(C_1) + \mathcal{ML}(C_2)$$

Of course, the upper bound from both the adaptive and non-adaptive cases can be extended to the scenario where n compositions are performed:

Corollary 6. $\mathcal{ML}(C_1 + C_2 + \cdots + C_n) \leq \sum_{i=1}^{n} \mathcal{ML}(C_i)$.

Corollary 7. $\mathcal{ML}(C_1 +_{na} C_2 +_{na} \cdots +_{na} C_n) \leq \sum_{i=1}^{n} \mathcal{ML}(C_i)$.

But we can wonder whether the upper bounds from corollaries 6 and 7 can be strengthened, particularly in the non-adaptive case. The following example shows that these bounds are actually tight.

Example 10. Let us consider some deterministic channels with two feasible outputs; the min-capacity of any such channel is 1 bit. First, consider a family of channels E_i that (like a password checker) output 1 exactly when a guess c_i matches the secret input:

E_i: `if (X == ` c_i`) Y = 1; else Y = 0;`

Composing either adaptively or non-adaptively n different instances of the family E_i yields a channel with $n+1$ distinct outputs, since at most one of the E_i's may produce the output 1. Hence, the min-capacity of the composition is logarithmic in n:

$$\mathcal{ML}(E_1 + E_2 + \cdots + E_n) = \log(n+1).$$

Consider now a variant family G_i, where the equality test is replaced with a greater-than-or-equal test:

G_i: `if (X >= ` c_i`) Y = 1; else Y = 0;`

Now it makes a difference whether we compose adaptively or non-adaptively. With a non-adaptive composition, we can get at most $n+1$ outputs from the combined channel. To understand why, assume that the c_is are chosen in increasing order. Then, the sequence of outputs must consist of k 1's (meaning that the value of X is greater than k of the c_i's) followed by $(n-k)$ 0's, for some k between 0 and n. Hence the combined channel has *logarithmic* capacity:

$$\mathcal{ML}(G_1 +_{na} G_2 +_{na} \cdots +_{na} G_n) = \log(n+1).$$

In contrast, with an adaptive composition, the resulting combined channel may yield a binary search where each G_i checks a different bit of the secret. For instance, if X is a 10-bit secret, we can build a combined channel that behaves as follows:

```
G1: if (X >= 512) Y1 = 1; else Y1 = 0;
G2: if (X >= 512*Y1 + 256) Y2 = 1; else Y2 = 0;
G3: if (X >= 512*Y1 + 256*Y2 + 128) Y3 = 1; else Y3 = 0;
...
```

This results in 2^n feasible outputs, giving *linear* capacity:

$$\mathcal{ML}(G_1 + G_2 + \cdots + G_n) = n.$$

Since each G_i has capacity 1, and n compositions of G_i yield a capacity of at most n, we conclude that the upper bound from Corollary 6 cannot be strengthened. But what if we restrict ourselves to non-adaptive composition? Curiously, that bound turns out to be tight as well.

Consider finally a family of programs A_i that use a bitwise "and" operation to test the ith bit of the secret:

A_i: `if (X &` 2^{i-1}`) Y = 1; else Y = 0;`

Clearly, composing non-adaptively n instances of A_i gives 2^n feasible outputs, again giving linear capacity:

$$\mathcal{ML}(A_1 +_{na} A_2 +_{na} \cdots +_{na} A_n) = n.$$

□

5. Related work

In this section, we briefly discuss some additional related work.

Worst-case posterior vulnerability, discussed in Section 2.3, is used by Mardziel et al. [17]. They consider a scenario where the confidentiality of a user's private information may be gradually consumed by a sequence of queries. After each query output y, the adversary's knowledge is updated dynamically, going from distribution p_X to posterior distribution $p_{X|y}$. The system wishes to ensure that the vulnerability never exceeds some threshold t. But (as we discussed in Section 2.2) aborting a query in the case when $V(p_{X|y}) > t$ would itself reveal information about X. For this reason, they decide whether or not to answer a query based on the worst-case posterior vulnerability, requiring in their Definition 3 that query C be answered only when $V^{worst}(p_X, C) \leq t$. In their implementation, they use abstract interpretation to compute a safe upper bound on $V^{worst}(p_X, C)$.

The channels we have considered here are known in information theory as *discrete memoryless channels*; they are appropriate for modeling non-interactive scenarios. But it is also interesting to consider interactive scenarios in which secret inputs and observable outputs *alternate*. Min-entropy leakage in interactive scenarios is briefly explored by Andrés et al. [24]. A fuller treatment, though considering only Shannon leakage, is given by Alvim et al. [25], making use of channels with *memory* and *feedback*.

In this paper, we have assumed that the prior π is known to the adversary. It is also interesting to consider the case where the adversary has possibly incorrect *beliefs* about the prior, a scenario explored by Clarkson et al. [26] and by Hamadou et al. [27].

Algorithmic techniques for calculating or bounding min-entropy leakage or min-capacity have seen considerable interest, including work on probabilistic automata by Andrés at al. [24] and work on deterministic imperative programs by Backes et al. [28], Newsome et al. [29], Köpf and Rybalchenko [30], Heusser and Malacaria [31], and Meng and Smith [32]. Also, negative complexity results have been given by Yasuoka and Terauchi [33].

The relationship between min-entropy leakage and *differential privacy* (see Dwork [34]) has been studied by Alvim et al. [35] and Barthe and Köpf [23].

Finally, min-entropy leakage assumes implicitly that the adversary gains only by guessing the secret *exactly*, in one try. Alvim et al. [18] introduce *g-leakage*, a generalization of min-entropy leakage that uses *gain functions* to model other operational scenarios, including those where the adversary gains by guessing the secret only *approximately* or *partially*.

6. Conclusion

In this paper, we have explored the viewpoint of secrecy as a *resource* that may be gradually *consumed* by a system. For measuring consumption, we have considered dynamic, worst-case, and average-case models, and we have concluded that (average) *min-entropy leakage* is particularly useful. Finally, we have shown that min-entropy leakage satisfies a number of compositionality results that allow the leakage of a complex system to be bounded by the leakage of its constituents.

Acknowledgments

We are grateful to Miguel E. Andrés, Mário S. Alvim, and the anonymous referees for their insightful comments and suggestions. This work was partially supported by the National Science Foundation under grants CNS-0831114 and CNS-1116318.

References

[1] M. K. Reiter, A. D. Rubin, Crowds: Anonymity for Web Transactions, ACM Transactions on Information Systems Security 1 (1) (1998) 66–92.

[2] P. Kocher, Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems, in: Proc. Advances in Cryptology (CRYPTO 1996), vol. 1109 of *Lecture Notes in Computer Science*, Springer-Verlag, 104–113, 1996.

[3] D. Clark, S. Hunt, P. Malacaria, Quantitative Analysis of the Leakage of Confidential Data, in: Proc. Workshop on Quantitative Aspects of Programming Languages, vol. 59 (3) of *Electr. Notes Theor. Comput. Sci*, 238–251, 2001.

[4] M. Boreale, Quantifying Information Leakage in Process Calculi, in: Proc. ICALP '06, 119–131, 2006.

[5] P. Malacaria, Assessing Security Threats of Looping Constructs, in: Proc. 34th Symposium on Principles of Programming Languages (POPL '07), 225–235, 2007.

[6] B. Köpf, D. Basin, An Information-Theoretic Model for Adaptive Side-Channel Attacks, in: Proc. 14th ACM Conference on Computer and Communications Security (CCS '07), 286–296, 2007.

[7] K. Chatzikokolakis, C. Palamidessi, P. Panangaden, Anonymity Protocols as Noisy Channels, Information and Computation 206 (2008) 378–401.

[8] K. Chatzikokolakis, C. Palamidessi, P. Panangaden, On the Bayes Risk in Information-Hiding Protocols, Journal of Computer Security 16 (5) (2008) 531–571.

[9] G. Smith, On the Foundations of Quantitative Information Flow, in: L. de Alfaro (Ed.), Proc. 12th International Conference on Foundations of Software Science and Computational Structures (FoSSaCS '09), vol. 5504 of *Lecture Notes in Computer Science*, 288–302, 2009.

[10] B. Köpf, G. Smith, Vulnerability Bounds and Leakage Resilience of Blinded Cryptography under Timing Attacks, in: Proc. 23nd IEEE Computer Security Foundations Symposium (CSF '10), 44–56, 2010.

[11] C. Braun, K. Chatzikokolakis, C. Palamidessi, Quantitative Notions of Leakage for One-try Attacks, in: Proc. 25th Conference on Mathematical Foundations of Programming Semantics (MFPS 2009), vol. 249 of *ENTCS*, 75–91, 2009.

[12] G. Smith, Quantifying Information Flow Using Min-Entropy, in: Proc. QEST 2011: 8th International Conference on Quantitative Evaluation of SysTems, 159–167, 2011.

[13] C. E. Shannon, A Mathematical Theory of Communication, Bell System Technical Journal 27 (1948) 379–423, 623–656.

[14] A. Rényi, On Measures of Entropy and Information, in: Proc. 4th Berkeley Symposium on Mathematics, Statistics and Probability 1960, 547–561, 1961.

[15] T. M. Cover, J. A. Thomas, Elements of Information Theory, John Wiley & Sons, Inc., second edn., 2006.

[16] B. Espinoza, G. Smith, Min-Entropy Leakage of Channels in Cascade, in: G. Barthe, A. Datta, S. Etalle (Eds.), Proc. Formal Aspects of Security and Trust (FAST 2011), Lecture Notes in Computer Science, 70–84, 2012.

[17] P. Mardziel, S. Magill, M. Hicks, M. Srivatsa, Dynamic Enforcement of Knowledge-based Security Policies, in: Proceedings of the Computer Security Foundations Symposium (CSF '11), 114–128, 2011.

[18] M. S. Alvim, K. Chatzikokolakis, C. Palamidessi, G. Smith, Measuring Information Leakage using Generalized Gain Functions, in: Proc. 25th IEEE Computer Security Foundations Symposium (CSF 2012), 265–279, 2012.

[19] U. Maurer, J. Massey, Cascade ciphers: The importance of being first, Journal of Cryptology 6 (1) (1993) 55–61.

[20] R. G. Gallager, Information Theory and Reliable Communication, John Wiley & Sons, Inc., 1968.

[21] M. Boreale, F. Pampaloni, M. Paolini, Asymptotic Information Leakage under One-Try Attacks, in: Proc. FOSSACS '11, 396–410, 2011.

[22] W. Feller, An Introduction to Probability Theory and Its Applications, vol. I, John Wiley & Sons, Inc., third edn., 1968.

[23] G. Barthe, B. Köpf, Information-theoretic Bounds for Differentially Private Mechanisms, in: Proc. 24th IEEE Computer Security Foundations Symposium (CSF 2011), 191–204, 2011.

[24] M. Andrés, C. Palamidessi, P. van Rossum, G. Smith, Computing the Leakage of Information-Hiding Systems, in: J. Esparza, R. Majumdar (Eds.), Tools and Algorithms for the Construction and Analysis of Systems (TACAS '10), vol. 6015 of *Lecture Notes in Computer Science*, 373–389, 2010.

[25] M. Alvim, M. Andrés, C. Palamidessi, Information Flow in Interactive Systems, in: Proc. 21st International Conference on Concurrency Theory (CONCUR 2010), 102–116, 2010.

[26] M. Clarkson, A. Myers, F. Schneider, Belief in Information Flow, in: Proc. 18th IEEE Computer Security Foundations Workshop (CSFW '05), 31–45, 2005.

[27] S. Hamadou, V. Sassone, C. Palamidessi, Reconciling Belief and Vulnerability in Information Flow, in: Proc. 31st IEEE Symposium on Security and Privacy, 79–92, 2010.

[28] M. Backes, B. Köpf, A. Rybalchenko, Automatic Discovery and Quantification of Information Leaks, in: Proc. 30th IEEE Symposium on Security and Privacy, 141–153, 2009.

[29] J. Newsome, S. McCamant, D. Song, Measuring Channel Capacity to Distinguish Undue Influence, in: Proc. Fourth Workshop on Programming Languages and Analysis for Security (PLAS '09), 73–85, 2009.

[30] B. Köpf, A. Rybalchenko, Approximation and Randomization for Quantitative Information-Flow Analysis, in: Proc. 23nd IEEE Computer Security Foundations Symposium (CSF '10), 3–14, 2010.

[31] J. Heusser, P. Malacaria, Quantifying Information Leaks in Software, in: Proc. ACSAC '10, 261–269, 2010.

[32] Z. Meng, G. Smith, Calculating Bounds on Information Leakage Using Two-Bit Patterns, in: Proc. Sixth Workshop on Programming Languages and Analysis for Security (PLAS '11), 1:1–1:12, 2011.

[33] H. Yasuoka, T. Terauchi, Quantitative Information Flow — Verification Hardness and Possibilities, in: Proc. 23nd IEEE Computer Security Foundations Symposium (CSF '10), 15–27, 2010.

[34] C. Dwork, A Firm Foundation for Private Data Analysis, Communications of the ACM 54 (1).

[35] M. Alvim, M. Andrés, K. Chatzikokolakis, P. Degano, C. Palamidessi, Differential Privacy: on the trade-off between Utility and Information Leakage, in: G. Barthe, A. Datta, S. Etalle (Eds.), Proc. Formal Aspects of Security and Trust (FAST 2011), Lecture Notes in Computer Science, 39–54, 2012.