Cryptogenography: Anonymity without trust

Sune K. Jakobsen

A thesis submitted in partial fulfillment of the requirements of the Degree of

Doctor of Philosophy

2016

Declaration

I, Sune K. Jakobsen, confirm that the research included within this thesis is my own work or that where it has been carried out in collaboration with, or supported by others, that this is duly acknowledged below and my contribution indicated. Previously published material is also acknowledged below.

I attest that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge break any UK law, infringe any third party's copyright or other Intellectual Property Right, or contain any confidential material.

I accept that the College has the right to use plagiarism detection software to check the electronic version of the thesis.

I confirm that this thesis has not been previously submitted for the award of a degree by this or any other university.

Signature: Date:

Details of collaboration and publications:

I published most of the results in Chapter 3, Section 5.4 and Chapter 6 (except Section 6.2) at ICALP 2014 [47]. A longer version of that paper has been published on arXiv [48] and has been submitted to a journal.

Sections 5.1-5.3 are based on work I published at ITCS 2014 together with Joshua Brody, Dominik Scheder and Peter Winkler [9]. I defined the cryptogenography game and invented all the protocols we used. I also found and proved the concavity characterization, which was used to prove the upper bound.

Finally, Chapter 7 is based on an ITCS 2016 paper [50] I published together with Claudio Orlandi. A longer version of that paper can be found on arXiv [49]. I had the idea that anonymous steganography might be possible and some idea about the kind of technique that was needed to do it. I mentioned the problem to Claudio Orlandi and together we found a construction that works. I proved the lower bound, and after the publication of the conference version I have strengthened the definition of anonymous steganography scheme.

Abstract

The usual methods of getting anonymity, such as using a VPN or the Tor network, requires some amount of trust: You have to either trust a particular server or trust that not too many servers in a network have been corrupted. In this thesis, we will explore how much we can do without this assumption.

Throughout the thesis we will assume that there is an adversary who can see all messages sent, that no two people have access to shared randomness and for much of the thesis we further assume that the adversary has unbounded computational power. In this case, it is impossible for one or more *leakers* to send any information without revealing some information about who they are. We define a measure of suspicion, which captures the anonymity loss of revealing information in this model: to reveal one bit of information, you will, in expectation, have to become one bit more suspicious. This measure is used to compute the exact amount of information a group of leakers can reveal if they want to keep reasonable doubt about who the leakers are. We also get exact results for the case where some people, censors, are trying to obstruct the leakage by sending misleading messages. The main result in these models is that (even without censors) the leakers can only reveal a very small amount of information. However, the protocols shown to exists might still be a useful alternative to warrant canaries.

We also consider the case where the adversary has bounded computational power. In this model, we show that it is still impossible for one leaker to reveal information without losing some anonymity. However, if we give the leaker access to a small anonymous channel, she can use this, combined with steganography, to reveal a large amount of information anonymously.

Acknowledgements

I would like to thank my supervisors, Peter Keevash and Søren Riis, for giving their invaluable support and advice, and for giving me the freedom and independence to pursue my own research interests, even when they brought me far away from the original topic of the PhD.

I would also like to thank my co-authors on the papers that became part of this thesis, Joshua Brody, Dominik Scheder, Peter Winkler and Claudio Orlandi, as well as my other co-authors, Jakob Grue Simonsen, Troels B. Sørensen and Vincent Conitzer. You have influenced both my thinking and my writing, and hence this thesis.

Peter Bro Miltersen deserves special thanks for hosting me for a visit at Aarhus University, where much of the work in Chapter 7 was done. I also want to thank all the people I met at Aarhus, for making it a particularly enjoyable month.

I am extremely grateful to my parents, who taught me everything from walking to solving systems of linear equations, who always encouraged me to think, and who did an incredible amount of work to help me pursue my interests. Without the great start to life they gave me, this thesis would not have been possible. I also want to thank my mother and my sisters for their unwavering support during my PhD.

I am thankful to my friends among the PhD students at Queen Mary, with whom I shared my PhD journey. You have all been great travel companions in the figurative sense, and several of you in the literal sense. I also want to thank all my other friends and family members for all the good times we have shared.

Finally, I would like to thank everyone who has ever played Mafia or Werewolf with me. You have helped fuel my desire to understand the theory behind hinting at information.

Contents

List of Figures

Chapter 1

Introduction

In 2013, when an NSA contractor, Edward Snowden, leaked thousands of documents revealing the scale of NSA's mass surveillance, it started an international debate about surveillance, privacy and anonymity. From the leaked files and from the following debate, we learned that NSA's goal is to "collect it all" [35]. It also became widely known that for some purposes, such as locating and justifying killing people[1], the US government considers metadata (data about who communicates with whom, when) to be sufficient. Such metadata cannot be hidden by just encrypting your messages, as encryption itself does not provide anonymity.

In the debate following the Snowden leakage, people have questioned whether we should be allowed to have secure communication. For example, during a speech last year, 2015, the current prime minister of the United Kingdom, David Cameron, asked rhetorically if we should allow people to communicate in ways that cannot be deciphered by the government, and answered "no, we must not" [73]. This raises the question: is it possible to prevent anonymous communication by banning it? In other words, if an adversary will punish anyone it believes is attempting to communicate anonymously or is helping others to do so, will it still be possible for people to communicate anonymously? In this thesis, we will explore various models, in an attempt to answer the question in the case where the adversary is able to punish anyone connected to the network.

[1]Former head of the NSA, Michael Hayden, said "We [US government] kill people based on metadata" [26].

1.1 Scope of the thesis

The word "cryptogenography" was coined by Peter Winkler, one of my co-authors on the paper "Cryptogenography" [9]. It is composed of three parts, crypto-geno-graphy, each originating from Greek, and translates to hidden-origin-writing.[2] The idea is that people send public messages, each of which can be traced back to a sender. The sequence of all the message is called the *transcript* and will be denoted T. When everyone has written their messages, the resulting transcript T can be interpreted as a message $G(T)$. We want to construct G in such a way that any small subset of the people can make it likely that $G(T)$ will give the message they intended to send, but in such a way that an observer cannot determine who these people are.

In this thesis we consider a much stronger adversary than what is typical in anonymity research. Throughout the thesis, we will assume that the adversary can see all messages people sent. Such an adversary is called a *global adversary*, and is common in anonymity research, but we will further assume the adversary knows any randomness that is shared between at least two people. In some chapters, we will also assume that the adversary has unbounded computational power and in some chapters we will assume that the non-leakers are not willing to help the leakers. We will now consider these assumptions in more detail.

1.1.1 Unbounded computational power

In Chapters 3-6 we will be assuming that the adversary has unbounded computational power. In practice, it is of course not possible to have unbounded computational power, but it could be used as a model of an adversary who has an extremely large amount of computational power or has an efficient way of solving problems in NP, possibly using quantum computers.[3]

Perhaps a more realistic scenario is one where the leakers cannot trust their own devices. Typically in anonymity research, we consider a person and their computer

[2]Contrary to what was suggested by Richard J. Lipton [56], the name was not inspired by the word "steganography". However, Lipton's comparison with steganography has inspired some of the later research in this thesis, in particular Chapter 6.

[3]The adversary having quantum computers is not in itself enough to justify the assumption of unbounded computational power, as you might still be able to use post-quantum cryptography [6].

to be one node, which can communicate with the rest of the world. However, if there is a risk that your computer has been hacked, or if you have limited control over what programs run on your computer or what information they reveal, the computer effectively becomes part of the "rest of the world". This means that any encryption would have to happen in the head of the sender. As computers have much more computational power than humans, at least for purposes related to cryptography, it is then reasonable to model the adversary's computational power as unbounded.

1.1.2 Non-leakers will not help leakers

In Chapter 6 and Chapter 7 we will assume that no non-leakers are willing to help the leakers. Furthermore, the results in Chapter 6 imply that the results in Chapters 3-5 also hold under this assumption. This assumption is why "anonymity without trust" is part of the title of this thesis: we are exploring how you can send information anonymously, if you cannot trust anyone.

At the time of writing, it seems that many people are willing to help others communicate anonymously. For example, the Tor network currently has around 7000 relays [63]. However, we will now argue, in three different ways, that if it was illegal to help leakers be anonymous, then it is reasonable to use a model that assumes that non-leakers will not help the leakers.

First the obvious argument: people tend not to do things they might be punished for, and if you help others to be anonymous, there will always be a risk that you will be discovered. This would probably stop many people from helping leakers send information anonymously, especially if the punishment was severe, but there might still be people willing to take the risk.

The second argument is by (informal) reduction. Suppose you have a protocol that ensures that l leakers can leak some information, with the help of h helpers while preserving the anonymity of both the leakers and helpers. Then $l + h$ leakers can use this protocol to reveal the same amount of information while all of them preserve anonymity: they simply choose h of them who will forget their information, and act as helpers in the original protocol. Thus, l leakers and h helpers can at most leak the same amount of information anonymously as $l + h$ leakers. This shows that upper bounds, in the case with any number of leakers but no helpers, translate to

upper bounds in the case where there is a bounded number of helpers.

Finally, we will consider what happens when the leakers cannot trust any of the helpers. This scenario might be plausible if it is illegal to be a helper and helpers risk losing their anonymity. In that case there might not be any helpers, but there might be many *censors* pretending to help the leakers, but who are just trying to reveal the leakers. Suppose you have a protocol that allows l leakers to reveal information with the help of h helpers. Assume further that the protocol ensures that leakers are going to keep their anonymity, even if all the helpers collaborate with the adversary. We will assume that the protocol gives a randomized algorithm that the helpers should follow if they are honest. We can now construct a protocol which allows l leakers to reveal some information without the use of helpers. The leakers will simply use some public randomness to simulate some imaginary helpers. The leakers would then send their messages as if the imaginary helpers existed and were sending the simulated messages. Now, anyone seeing the communication can simulate the helpers, so anyone can compute the transcript as it would have been if the helpers were really there. Hence the resulting protocol will be as good for revealing information as the original protocol. Furthermore, the leakers preserve anonymity, as we assumed that the adversary in the original protocol had access to all the helper's information.

1.2 Contributions of the thesis

1.2.1 Suspicion

In Chapter 3 we define a measure of suspicion, which exactly captures the price for revealing information, when observed by a computationally unbounded adversary: in order to reveal one bit of information, you will in expectation become at least one bit more suspicious, and you can always reveal one bit[4] for the price of getting one bit more suspicious in expectation. This measure will be useful in all the chapters where the adversary has unbounded computational power (all chapters except Chapter 7).

[4]In the sense of mutual information.

1.2.2 Information theoretic cryptogenography

In Chapter 3 we will consider a model where an adversary has unbounded computational power and can observe any message sent. Intuitively, it should be impossible to reveal any information in this model, without losing anonymity. However, we will see that many people can collaborate to reveal the information, by each making hints towards the information, in a way that ensures that no one will look guilty beyond reasonable doubt. If each person is a leaker with probability b_l and they want to ensure that an observer never says that there is probability above $b_m > b_l$ for the person being a leaker, then we can leak $\frac{-b_l \log(1-b_m)+b_m \log(1-b_l)}{b_m}$ bits of information per total number of people communicating. When there is a small number of leakers, this means that they can leak $\frac{-\log(1-b_m)}{b_m} - \log(e)$ bits per leaker. We will also prove matching upper bounds. In particular, this result proves the intuition that a single leaker cannot reveal much information, when always observed by an adversary with unbounded computational power, while keeping reasonable doubt. This holds even if non-leakers try to help the leaker. We will also consider various models where non-leakers can become leakers. Throughout this chapter we will assume that non-leakers collaborate with the leakers, but a result in Chapter 6 implies that this assumption is not necessary.

1.2.3 Resilient cryptogenography

In Chapter 4 we add censors to the model above. Censors are people who send misleading messages in an attempt to prevent the leakage of information.[5] We show that if each person is a censor with probability b_c, then the censors can prevent any leakage of information if and only if $b_l + b_c \geq b_m$. In the case $b_l + b_c < b_m$ the censors have two effects. First, they can lower the capacity, that is, they can ensure that each leaker can only reveal a smaller number of bits than if there were no censors. We find matching upper and lower bounds for the resulting capacity. Secondly, the censors can create some ambiguity about the information x that the leakers try to leak. An observer who watches the communication will not necessarily be able to deduce the correct secret x, but will only be able to write down a list

[5]They could also attempt to reveal the leakers, but for the protocols we consider, the censors' messages will not affect the leakers' anonymity.

of $1 + \lfloor \frac{b_c}{b_l} \rfloor$ elements which, with overwhelming probability, will contain x. Again, we will assume that people who are not leakers or censors will collaborate with the leakers, but a result in Chapter 6 shows that this assumption can be removed.

1.2.4 Cryptogenography games

In Chapter 5 we analyse a game where one or more leakers in a group of people want to reveal one or more bits of information. The leaker(s) win if an observer, who is on their team, can guess the information, but an adversary who has the same information cannot guess any of the leakers. For most of the chapter, we consider what happens when we have one leaker and one bit of information. We show that when there are only two people the probability of winning is at least $\frac{1}{3}$ and at most $\frac{3}{8}$,[6] and for a large number of people the winning probability is more than 0.5644 and at most $\frac{3}{4}$. In the case where the number of leakers and bits of information tends to infinity, we show that if bits per leaker tends to zero, the winning probability tends to 1 and if the ratio tends to infinity, the winning probability tends to 0.

1.2.5 Anonymous steganography

Finally, in Chapter 7 we consider the case where the adversary has bounded computational power, and where one leaker is trying to reveal information without help from non-leakers. Furthermore, we will assume that the leaker has access to a small or expensive anonymous channel. In this model we will see that the leaker can use steganography and the small anonymous channel to anonymously reveal a much larger amount of information than what she can send over the small channel. We will also see that in this model it is impossible to reveal information anonymously without using a small anonymous channel to bootstrap the communication.

1.3 Notation and preliminaries

We let $[n]$ denote the set $\{1, \ldots, n\}$. For a tuple or infinite sequence a, we let a_i denote the i'th element of a, and, unless stated otherwise, a^i denotes the tuple of the

[6] The lower bound has since been improved to 0.3384 and the upper bound to 0.3672 [5].

i first elements from a: $(a_1, \ldots, a_i)$. We use a_{-i} to mean $(a_1, \ldots, a_{i-1}, \perp, a_{i+1}, \ldots, a_n)$ and (a_{-i}, b) to mean $(a_1, \ldots, a_{i-1}, b, a_{i+1}, \ldots, a_n)$. Similar notation is used if A is a tuple or sequence of random variables. For tuples a and b of n_a and n_b elements we let $a \circ b$ denote the tuple $(a_1, \ldots, a_{n_a}, b_1, \ldots, b_{n_b})$. If a' is a single element, we abuse notation and write $a \circ a'$ instead of $a \circ (a')$.

A function is *negligible* if it goes to 0 faster than the inverse of any polynomial. We write $\mathsf{poly}(\cdot)$ and $\mathsf{negl}(\cdot)$ to mean some polynomial and negligible function respectively.

Unless stated otherwise, all random variables in this thesis are assumed to be discrete. Random variables are denoted by capital letters and they take values from the set denoted by the calligraphic version of the same letter (e.g. X takes values from $\mathcal{X}$). Specific values taken by a random variable are denoted by the lower case version of the same letter. If X and Y are random variables and $\Pr(Y = y) > 0$, we let $X|_{Y=y}$ denote the random variable X conditioned on $Y = y$. That is

$$\Pr(X|_{Y=y} = x) = \frac{\Pr(X = x, Y = y)}{\Pr(Y = y)}.$$

We typically write $\Pr(X = x|Y = y)$ instead of $\Pr(X|_{Y=y} = x)$.

In the rest of this section we will give some definitions and results from information theory. For an introduction to these concepts and for proofs, see Cover and Thomas [18]. For a random variable X and a value $x \in \mathcal{X}$ with $\Pr(X = x) > 0$ the *surprisal* or the *code-length*[7] of x is given by

$$-\log(\Pr(X = x)),$$

where log, as in the rest of this thesis, is the base-2 logarithm.

[7] If $-\log(\Pr(X = x))$ is an integer for all $x \in \mathcal{X}$, and we want to find an optimal prefix-free binary code for X, the length of the code for x should be $-\log(\Pr(X = x))$, thus the name code-length. If they are not integers, we can instead use $\lceil -\log(\Pr(X = x)) \rceil$ and waste at most one bit.

The *entropy of* X, $H(X)$, is the expected code-length of X

$$\begin{aligned} H(X) =& \mathbb{E}_x \left[-\log(\Pr(X = x)) \right] \\ =& -\sum_{x \in \mathcal{X}} \Pr(X = x) \log(\Pr(X = x)), \end{aligned}$$

where we define $0 \log(0) = 0$.

The entropy of a random variable X can be thought of as the uncertainty about X, or as the amount of information in X. For a tuple of random variables $(X_1, \ldots X_k)$ the entropy $H(X_1, \ldots X_k)$ is simply the entropy of the random variable $(X_1, \ldots, X_k)$. The *entropy of* X *given* Y, $H(X|Y)$ is

$$H(X|Y) = \sum_{y \in \mathcal{Y}} \Pr(Y = y) H(X|_{Y=y}). \tag{1.1}$$

A simple computation shows that

$$H(X|Y) = H(X, Y) - H(Y).$$

The *mutual information* $I(X;Y)$ of two random variables X, Y is given by

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

This is known to be non-negative.

The mutual information satisfies a data processing inequality

$$I(X; f(Y)) \leq I(X, Y) \tag{1.2}$$

for any function f and discrete random variables X, Y. This says that you cannot get more information about X out of Y by computing some function on Y.

The *mutual information* $I(X;Y|Z = z)$ *of* X *and* Y *given* $Z = z$ is given by

$$I(X;Y|Z = z) = I(X|_{Z=z}; Y|_{Z=z}),$$

where the joint distribution of $(X|_{Z=z}, Y|_{Z=z})$ is given by $(X, Y)|_{Z=z}$. The *mutual*

information $I(X;Y|Z)$ *of* X *and* Y *given* Z is

$$I(X;Y|Z) = \mathbb{E}_z I(X;Y|Z=z).$$

A simple computation shows that

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

We will need the chain rule for mutual information,

$$I(X;(T_1,\dots T_k)) = \sum_{i=1}^{k} I(X;T_i|(T_1,\dots,T_{i-1})).$$

Intuitively, this says that the information $(T_1,\dots,T_k)$ gives about X is the sum of the information each of the T_is reveals, given the previous T_is.

Let X and Y be jointly distributed random variables, and $f : \mathcal{Y} \to \mathcal{X}$ a function. We think of $f(Y)$ as a guess about what X is. The probability of error, P_e is now $\Pr(f(Y) \neq X)$. We will need (a weak version of) Fano's inequality,

$$P_e \geq \frac{H(X|Y) - 1}{\log(|\mathcal{X}|)}. \tag{1.3}$$

A *discrete memoryless channel* (or *channel* for short) q consists of a finite *input set* $\mathcal{Y}$, a finite *output set* $\mathcal{Z}$ and for each element $y \in \mathcal{Y}$ of the input set a probability distribution $q(z|y)$ on the output set. If Alice has some information X that she wants Bob to know, she can use a channel. To do that, Alice and Bob will have to both know a code. An *error correcting code*, or simply a *code*, $\mathfrak{c} : \mathcal{X} \to \mathcal{Y}^n$ is a function that for each $x \in \mathcal{X}$ specifies what Alice should give as input to the channel. Here n is the *length* of the code. Now the probability that Bob receives $Z_{\mathfrak{c}} = z_1 \dots z_n$ when $X = x$ is given by

$$\Pr(Z_{\mathfrak{c}} = z | X = x) = \prod_{i=1}^{n} q\left(z_i | \mathfrak{c}(x)_i\right).$$

Bob will then use a *decoding function* G which sends outputs z to guesses about X. A *rate* R is *achievable* if for all $\epsilon > 0$ there is a $n > 0$ such that for X uniformly

distributed on $\{1, \ldots, 2^{\lceil Rn \rceil}\}$ there is a code $\mathfrak{c}$ of length n for q and a decoding function G giving $\Pr(G(Z) = x | X = x) > 1 - \epsilon$ for all $x \in \mathcal{X}$.

For a distribution p on the input set $\mathcal{Y}$ we get a joint distribution of (Y, Z) given by $\Pr(Y = y, Z = z) = p(y)q(z|y)$. Now define the capacity C of q to be

$$C = \max_p I(Y; Z),$$

where max is over all distributions p of Y and the joint distribution of (Y, Z) is as above. Shannon's Noisy Coding Theorem says that any rate below C is achievable, and no rate above C is achievable [67].

If $p, q : \mathcal{X} \to [0, 1]$ are two probability distributions on $\mathcal{X}$ we have the inequality

$$-\sum_{x \in \mathcal{X}} p(x) \log(p(x)) \leq -\sum_{x \in \mathcal{X}} p(x) \log(q(x)), \tag{1.4}$$

with equality if and only if $p = q$. The interpretation is, if X's distribution is given by p, and you encode values of X using a code optimized to the distribution q, you get the shortest average code-length if and only if $p = q$.

For discrete probability distributions p and q over $\mathcal{X}$, the *Kullback-Leibler divergence* of q from p is defined by

$$D(p||q) = \sum_{x \in X} p(x) \log \left(\frac{p(x)}{q(x)} \right).$$

For numbers $p, q \in [0, 1]$, the *Kullback-Leibler divergence* $D(p||q)$ is the Kullback-Leibler divergence of the $\{0, 1\}$-distribution where 1 has probability q from the $\{0, 1\}$-distribution where 1 has probability p.

The Kullback-Leibler divergence is a measure of how far p is from q, but is not symmetrical in p and q, that is, in general $D(p||q) \neq D(q||p)$. We can also write the Kullback-Leibler divergence as

$$D(p||q) = \sum_{x \in \mathcal{X}} p(x) \log(p(x)) - \sum_{x \in \mathcal{X}} p(x) \log(q(x)).$$

Thus, inequality 1.4 implies that $D(p||q) \geq 0$ with equality if and only if $p = q$.

Chapter 2

Previous Work

2.1 Definition of anonymity

The word "anonymous" comes from the Greek word for "without a name", and intuitively we might think of a person as anonymous if we cannot name the person. We will now give a more precise definition of anonymity.

First of all, anonymity depends on an *adversary*, also sometimes called the *attacker* or, when passive, the *observer*. Anonymity is also with respect to a particular role $\mathcal{R}$ [52]. In anonymity research $\mathcal{R}$ is typically either the role of being the sender or the receiver of a message, but it could also be other things, such as attending a demonstration. In this thesis, we will only focus on *sender anonymity*, that is, $\mathcal{R}$ will always be the sender of a message.

A person who wants anonymity, Lea[1], does not mind that the adversary knows that she exists and knows her name. In fact we will assume that the adversary has a way to refer to any particular person. In the real world, this could be a name and address, a social security number or a physical body, but in our mathematical formulation, it will simply be a number. Lea also does not mind that the adversary realize that a particularly message m exists, and that someone must have played the role $\mathcal{R}$ of sending that message. When we say that Lea has anonymity, it means that the adversary cannot know if Lea played the role $\mathcal{R}$. This is the unlinkability

[1]We follow a naming convention suggested by Nadia Heninger in private communication with Claudio Orlandi [50]: the leaker is called Lea and the recipient is Joe the journalist.

definition from Pfitzmann and Hansen [62].

If the adversary has no information at all about who was playing role $\mathcal{R}$, the person has perfect anonymity, and conversely if the adversary can link $\mathcal{R}$ to a name/identifier with certainty, there is no anonymity at all. Typically, when a person is trying to keep anonymity, we will be somewhere in between these two extremes, and we need some way of measuring the amount of anonymity. As we will see, there are many possible measures we could use.

One measure is the size of the anonymity set. The *anonymity set* for $\mathcal{R}$ for a particular adversary $\mathcal{A}$ is the set of people who, given the adversary's knowledge, might be $\mathcal{R}$ [14, 62]. The disadvantage in using the size of the anonymity set as a measure of anonymity, is that it ignores some information, for example, the adversary might be 99.9% sure about who $\mathcal{R}$ is, and yet be unable to rule out any particular person from being $\mathcal{R}$. In this case there is almost no anonymity, yet the anonymity set is as large as possible.

Another possible measure, is to use the adversary's uncertainty, measured using the Shannon entropy, as a measure of the anonymity. This approach was suggested by Diaz, Seys, Claessens and Preneel [21] and independently by Serjantov and Danezis [66]. In each of these papers, we compute the distribution of the person, I, who had role $\mathcal{R}$ given the knowledge of the adversary, and compute the Shannon entropy, $H(I)$, of the resulting variable. Diaz et al. [21] suggested using $\frac{H(I)}{H_M}$, where H_M is the maximal possible entropy of I, as a measure of the amount of anonymity a system preserves. This measure is in the interval $[0, 1]$, and gives us a measure of how large a fraction of the possible anonymity a system preserves. Serjantov and Danezis suggested simply using $H(I)$ as a measure of the anonymity. This has the theoretical advantage that it is independent of which users are considered to be "part of the protocol" and it gives a measure of how much anonymity the sender has, rather than how large a fraction of the *possible* anonymity the sender has.

The entropy, $H(I)$, measures how much information the adversary would, on average, have to get from elsewhere to completely reveal who had role $\mathcal{R}$. However, as pointed out by Tóth, Hornák, and Vajda [69], this measure does not capture all the relevant information. For example, if the adversary estimates that one particular suspect send a message with probability 50%, and otherwise has no idea who the leaker is, the entropy is $1 + \frac{1}{2}\log(\text{world population} - 1) \approx 17.4$ bits, or approximately

the same as if the adversary has no idea who among $2^{17.4} \approx 170\,000$ people had sent the message. It is clear that in 50% of the cases, the leaker would strongly prefer the second scenario, while the leaker would be almost indifferent in the other 50% of the cases.

From the point of view of one particular leaker, it is probably more relevant how suspicious she might appear to the adversary. For this reason, Tóth, Hornák, and Vajda [69] suggested using a measure which we will call reasonable doubt. We say that the adversary has *reasonable doubt (at the b_m level)* about who had role $\mathcal{R}$ if, whenever the anonymity system is used, the adversary will never estimate anyone to have had role $\mathcal{R}$ with probability above b_m.

A similar looking definition is the *probability of error* as suggested by Chatzikokolakis, Palamidessi and Panangaden [12]. They imagine that the adversary will always make a guess about one person she thinks is a leaker[2], and they defined the probability of error to be the probability that the adversary is wrong about this guess. Reasonable doubt at the b_m level is a strictly stronger requirement than the probability of error being at least $1 - b_m$. For a proof of this and more discussion about the requirement of reasonable doubt, see Section 3.1.3

Clauß and Schiffner [16] suggested a measure similar to that of Serjantov and Danezis, but using Rényi-Entropy (a generalization of Shannon entropy) instead of Shannon entropy. At first, this seems to contain all the above as special cases: Shannon entropy is a special case of Rényi-Entropy, the size of the anonymity set is the exponential of the max-entropy, which is a special case of the Rényi-Entropy, and finally, the maximal probability b_m with which someone is estimated to have role $\mathcal{R}$, is exponential in the min-entropy. However, there is a subtle difference, as Clauß and Schiffner (as well as [21, 66]) only seem to suggest a way of measuring the anonymity given a particular run of the anonymity system, while the b_m level of reasonable doubt in Tóth et al. [69] is about the worst case run. Several other measures of the anonymity of communication have been suggested [3, 24, 32, 79].

Finally, Backes et al. [2] suggests measuring anonymity using the language from differential privacy [23]: we say that two random variables D_0 and D_1 are (ϵ, δ)-*differentially private* if, for any adversary $\mathcal{A}$ taking values of D_0 and D_1 as input and

[2] In [12] it is assumed that there is only one sender/leaker, but it can be generalised to a situation where the number of leakers in not fixed or known.

outputting 0 or 1 we have: $\Pr(\mathcal{A}(D_i) = 0) \leq e^{\epsilon} \Pr(\mathcal{A}(D_{1-i}) = 0) + \delta$ for $i \in \{0, 1\}$. Similarly, we say that D_0 and D_1 are computationally (ϵ, δ)-differentially private if the inequality holds for all PPTs (Probabilistic Polynomial Time algorithms) $\mathcal{A}$. Now, we can define an anonymous communication protocol to be (ϵ, δ)-anonymous if, for any two people PLR_0 and PLR_1, the adversarial view (the information available to the adversary) D_0 when PLR_0 sends a message and the adversarial view D_1 when PLR_1 sends the message are (ϵ, δ)-differentially private. The advantage in this definition is that it works no matter what prior information the adversary might have: no matter how suspicious or innocent Lea looks to the adversary before the communication, if Lea uses a (ϵ, δ)-anonymous communication system to send a message, the probability that the adversary will take a particular action against Lea, is only a multiplicative factor of e^{ϵ} plus a constant of δ higher, than the probability that the adversary would take the action against *Lea* if *another person* was sending the message.

In this thesis, we will mostly use the measure from Tóth et al. [69], which we call reasonable doubt, as well as a new measure of *suspicion*. The measure of suspicion might seem unnatural at first, and we do not think that people who want to stay anonymous will care about minimizing this particular measure. However, we will see that suspicion captures exactly the anonymity you lose – seen from an adversary with unbounded computational power who can see any message – when you send information: to send one bit of information, you will, on average, have to become one bit more suspicious. Knowing that suspicion is the currency with which you pay to reveal information, makes it possible for a leaker to make the optimal trade-off between sending information and keeping anonymity as measured by a measure she does care about, such as reasonable doubt. The measure of suspicion might also be useful for evaluating anonymous communication systems, which makes stronger assumptions about the adversary, as it tells us how much information we could have revealed without making such assumptions.

2.2 Ways of communicating anonymously

The simplest possible way to communicate anonymously, is to communicate through a trusted party. For example, if Lea has a story she wants the world to know about,

she might reveal it to a journalist, Joe, and ask him not to give away any information that might identify her. While this method is simple, it is not very secure: Joe will know who Lea is, and can always choose to reveal her identity. This would be a problem if Joe is not trustworthy, for example, he might use this information to blackmail Lea. Even if Joe is trustworthy, he might be forced, possibly by law, to reveal Lea.

The anon.penet.fi mail relay was essentially an email version of the journalist example above [19], except that it allowed communication to go both ways. This service, running from 1993 to 1996, would store email addresses and corresponding pseudonymous addresses. People could send an email through this system, and the recipient would only see a pseudonymous address. Conversely, emails sent to the pseudonymous address would be forwarded to a corresponding non-anonymous address. However, in 1996, anon.penet.fi was obligated by a Finnish court to reveal the email address of one of its users [39], and the service closed the same year, because it could no longer protect the identity of its users [40].

Similar systems, called virtual private networks or VPNs, exist for web browsing, for example Anonymizer [1]. Such services can choose not to keep logs in an attempt to avoid legal attacks. However, the user still needs to trust one particular company. We know that other types of companies have been forced to collaborate with intelligence services while being prohibited from revealing this (this happened to Lavabit, and caused the company to shut down [55] and it probably also happened to Microsoft, Google, Yahoo, Facebook and Apple as part of the Prism-program [36]), so even a user who trusts the intentions of the people behind the company, might not want to trust the company. Furthermore, even if the company can be trusted, an attacker who can observe all information going through the server, might be able to break the anonymity using traffic analysis.

Crowds [65] is another suggested method of achieving anonymity. To visit a webpage, a crowd-user would relay its request through another crowd-user. This second user will randomly choose either to send the request to the webpage, or to a third crowd-user. If the request is sent to a third crowd-user, this user will not know how many users have forwarded this request, so it will again either send the request to the web-page or to another crowd-member. The probability p_f that the request is forwarded is some constant. The hope is that a weak adversary – for

example an adversary who only controls a small number of crowd members, and who can only see messages sent through these crowd members – cannot be certain that any particular person is the sender of a message: she could just be relaying the message for someone else. However, even an adversary who only controls a single crowd member, might be able to use timing information to deduce that the person sending request to it, is the original sender for the message. This effect could be minimized by inserting time delays [65], but because the protocol corresponds to an untimeable game [51] this leakage cannot be completely avoided. A more serious problem is that crowds does not provide anonymity against a global adversary.

In the first paper in the field of anonymous communication [13], Chaum introduced the concept of a Mix. This is a server that takes as input a stream of messages, each of which has been encrypted, possibly with multiple layers of encryption. The output of the mix is the same messages with the outermost layer of encryption removed. The mix will output several messages at the same time, and all the messages in each batch are sorted lexicographically. If done correctly, this ensues that an outside observer cannot link any of the input messages to any of the output messages.

If many people send messages through a sequence of mixes, called a *cascade* of mixes, then an adversary would have to compromise each of the mixes to link an input message to an output message. In particular, it provides anonymity against an adversary who can observe all messages sent. One disadvantage is that if two people sent many messages to each other, a relatively weak adversary, who only observes the first and the last mix in the cascade, would be able to use traffic analysis. Another disadvantage is that each message has to go through all the mixes. This means that the system does not scale well, and that any one of the mixes can completely obstruct the system.

Another possibility is to use a *network* of mixes, where each person can choose which route their message takes through the network. This has the advantage that the first and last mix on the route will be different for different people, which prevents weak adversaries from doing a timing attack. Babel [37], Mixmaster [60] and Mixminion [20] all use a network of mixes to allow users to send emails anonymously.

Onion Routing [34] is superficially very similar to mix networks, but does typically not involve any mixing [68]. This allows the servers to forward a message

immediately after receiving it, instead of waiting for other messages to mix it with. This in turns lowers the latency (the time from when a message is sent by the original sender to when it is received by the intended recipient) which makes onion routing suitable for web browsing. Instead of basing its security on mixing, onion routing bases its security on the assumption that it is difficult for an adversary to observe a large part of the network. In particular, Tor [22], which is an implementation of onion routing, does not attempt to provide anonymity against a global adversary or even against an adversary who controls both the first and the last onion router on a path. These important differences between onion routing and mix networks were pointed out by Syverson [68].

Tor also allows for the use of hidden services [22]. These are provided by servers connected to the Tor network, in a way that does not reveal the server's IP-address. One particular kind of hidden service is SecureDrop [27], which is a whistleblower system that media can host. It makes it possible for whistleblowers to connect to the system through Tor, and upload documents and messages to journalists.

DC-nets, named after the dining cryptographers problem, gives a way of communicating information anonymously, even against a global adversary [14]. The only assumption needed is that any two people have access to shared randomness, which is not seen by the observer. This can be seen as a special case of secure multi-party computation. In the general case of secure multi-party computation, each player has an input to a function and we want to compute a function of these inputs, without revealing any further information. If this function simply computes the list of inputs in lexicographical order, we have a way of publishing information anonymously. Secure multi-party computation is possible against a global adversary if we assume either that the adversary has bounded computational power or that any two people have access to shared randomness unknown to the observer [4]. Furthermore, it works even if the attacker can corrupt some participants, as long as less than one third of the participants can be corrupted. Both DC-nets and multi-party computation have the disadvantage that the amount of communication required for each participant is at least linear in the size of the anonymity set, which makes them impractical for large scale use.

Several recent papers have attempted to construct practical protocols, that provide anonymity against traffic analysis. One such paper is Dissent [77]. In this

paper, the authors suggest having a relatively small number of servers, say 5, which will communicate in a way similar to DC-nets. The idea is that clients, who wish to browse the web, will generate a request, and secret share this request among the servers, that is, the client sends the request in such a way that all the servers have to collaborate to get the request. The servers now collaborate to decrypt the request in a way that does not reveal who made which request. Unlike for DC-nets, the communication per client no longer grows with the number of clients. To provide anonymity, we only need one of the servers to be honest.

Another such paper is Riposte [17], which works by using PIR (private information retrieval) [15] in reverse. In PIR, several servers contains shares of a database, and allow users to make requests to the servers, such that, assuming that the servers are not colluding, the servers cannot know which part of the database the user is looking at. Riposte turns this around to allow users to upload information to a database. The result is a sender-anonymous twitter-like system, where each user can upload small messages. Another recent idea is Vuvuzela [70], which uses a cascade of mixes and dummy messages, among other ideas, to ensure (ϵ, δ)-differential privacy against traffic attack.

For a survey of the anonymous communication literature until 2008, see Danezis and Diaz [19]. This survey contains a more detailed description of many of the systems mentioned above, as well as description of many other anonymous communication systems and attacks against such systems. Freehaven maintains a bibliography in anonymity [28], which currently contains more than 300 papers.

2.3 Other related ideas

While some of the above methods hide whether a particular person is sending information at any moment, they do not typically hide who is taking part in the protocol. Thus, an attacker who has the power to punish anyone it believes is attempting to communicate anonymously or is helping others to do so, can prevent the above methods from being used.

Steganography is a way of hiding communication, or at least making communication look like it is a different kind of communication [29, 64]. The idea is to embed the file you want to send, called the *payload*, into an "innocent" file, called

the *covertext*, to get an innocent looking file, called the *stegotext*. The recipient of the message can then run an extraction algorithm on the stegotext to get the payload. Steganography schemes that ensure that a random stegotext is indistinguishable to a random covertext have been suggested, both against computationally bounded [42] and against computationally unbounded adversaries [10]. Unfortunately, this requires that the distribution of the covertext is known, that you can make exponentially many samples from the covertext or that you can break down the covertext into smaller parts, such that for each part, you can sample from its distribution given the previous parts. It is not clear how to do this for e.g. an image, so in practice you have to use imperfect steganography.

While steganography itself does not provide anonymity, it can be combined with anonymity techniques from above. For example, it can be used to hide the fact that you are following the Tor protocol [59, 74], from a weak adversary. However, an adversary who knows which servers are connected to the Tor network, will still be able to see that you are connecting to one of those servers.

Another use of steganography is Message in a Bottle [45], which provides a way for Lea – who lives in a country, Tyria, controlled by censors – to send information, which the censors would want to censor, to the outside world. To do so, we need the help of someone, Joe, from outside Tyria. Joe will distribute his public key, and Lea will use this to encrypt her message. She will then embed this encrypted message into an innocent image, and upload it to her blog. Most blogs send out a blog ping to certain servers, and Joe goes through all such blog pings, and try to extract messages from all images uploaded to blogs. From most blog posts he would just get a random-looking sequence out, but from Lea's blog post he will get the intended message. This has been shown to be possible in practice [45]. The scheme ensures that the censors do not learn that Lea is communicating with Joe, but Joe will still learn who Lea is. In Chapter 7 we will see that a similar strategy can, in theory, be used to bootstrap an anonymous channel.

One use of anonymity, is to make people give more honest answers to survey questions. However, when interviewed, people might not trust the promise of anonymity they are given. One way to guarantee the respondent some amount of deniability is randomized response [71]: suppose you want to know how large a fraction π of people belong to a particular population A, but people are embarrassed to say that

they belong to this population. You then give the respondent a spinner, which gives "A" with some probability p and "not A" otherwise. Now, instead of asking people which population they belong to, you ask them to spin the spinner secretly, and ask if the result was the population they belong to. The closer p is to $\frac{1}{2}$ the less information the respondent reveals, but as long as $p \neq \frac{1}{2}$, we still get probabilistic information about which population the respondent belongs to. By asking sufficiently many people, we can, if they are honest, get an arbitrarily good estimate of the true fraction π of people who belong to population A.

This idea has been analyzed further by many papers in the area of privacy preserving data mining, including Evfimievski, Gehrke and Srikant [25]. In this paper, the authors noted a fact, which will also be useful in this thesis: let b_l be the prior probability $\Pr(L = 1)$ that the respondent is of a certain type called 1, and let r be an upper bound on the ratio between how likely the answer a is when the respondent is of type 1 and when she is not, that is $\frac{\Pr(A=a|L=1)}{\Pr(A=a|L\neq 1)} \leq r$. Then to ensure that the posterior probability $\Pr(L = 1|A = a)$ that the respondent is of type 1, is at most b_m, it is enough that $r \leq \frac{b_m(1-b_l)}{b_l(1-b_m)}$.

Finally, one situation where people might want to send information anonymously, is when they are under a gag order, that is, they have been ordered by court not to reveal some information. For example, a company might want to reveal whether they have been forced to give FBI access to their records, but would not be allowed to say so if they had. One solution to this problem is a so-called "warrant canary", often used against Section 215 of the USA Patriot Act [33,75]: as long as Section 215 has *not* been used to compel a company to give access to their records, the company can regularly publish a statement saying so. If the company stops publishing such statements, people might conclude that the company has been forced to give access to its records, but that it is not allowed to say so explicitly.

Most citizens would not notice if a company stopped sending out such a statement, so a webpage, Canary Watch [72] has been set up to keep track of them. One disadvantage in warrant canaries is that it is not always clear how to interpret the absence of a message: in 2014 Apple stopped publishing a statement it had previously been publishing regularly, saying that they had received no surveillance orders under Section 215 of the USA Patriot Act. It was not clear if that was because they had now received such an order, or simply because they changed their reporting for-

mat [75]. Furthermore, the legal issues around canary warrants are not settled. For example, we do not know if a company, which has received an order which should "kill" the canary, could be compelled to continue publishing (now false) statements saying that it has never received such an order [33, 75].

Chapter 3

Information Theoretic Cryptogenography

In this chapter we consider problems where one or more players might be trying to leak information about the outcome of a random variable X. In the main result of this chapter we assume that n players are communicating and each of them is a leaker with probability b_l. Each leaker wants to preserve reasonable doubt, that is, she wants to ensure that an observer who knows X and has unbounded computational power, will never assign probability above $b_m > b_l$ to the event that she is a leaker. At the same time the leakers want reliable leakage: they want to ensure that an observer who does not know X before the communication, will after the communication be able to guess X correctly which probability $1 - \epsilon$ on average. We let $n \to \infty$ and let X be uniformly distributed over $\{0,1\}^{\lceil nR \rceil}$ for some R. In Corollary 3.10 we will see that it is possible to achieve reliable leakage while keeping reasonable doubt for sufficiently large n if $R < D\left(\frac{b_l}{b_m}\middle\|\frac{b_l(1-b_m)}{b_m(1-b_l)}\right) = \frac{-b_l \log(1-b_m)+b_m \log(1-b_l)}{b_m}$ but not if $R > D\left(\frac{b_l}{b_m}\middle\|\frac{b_l(1-b_m)}{b_m(1-b_l)}\right)$.

We will also consider various related problems: in the *risky* model, we allow that the leakers lose reasonable doubt with some small probability ϵ, but we will see that this does not help the leakers reveal more information. We also consider a model where the number of leakers is fixed and known rather than binomially distributed. Finally, we consider some adaptive models, where players can become leakers as part of the protocol. All the main positive results in this chapter will rely on Shannon's

Noisy-Channel Coding Theorem, so they will be existential rather than constructive. The negative results will use a measure of suspicion which we will introduce in this chapter.

The number of players is denoted n and the players are called $\text{PLR}_1, \ldots, \text{PLR}_n$. Sometimes we will call PLR_1 Alice and PLR_2 Bob. We let L_i be the random variable that is 1 if player i knows the information and 0 otherwise. If there is only one player we write L instead of L_1. The joint distribution of $(X, L_1, \ldots, L_n)$ is known to everyone.

All messages are broadcast to all players and to two observers, Eve and Joe. Both observers see the transcript, but Eve also know the secret X. We want to reveal information about X to Joe, while at the same time make sure that for all i, Eve does not get too sure that $L_i = 1$. The random variable that is the transcript of a protocol will be denoted T, and specific transcripts t. This is a tuple of messages, so we can use the notation T^k, T_k, t^k, t_k as defined in the preliminaries. For example, T^k denotes the random variable that gives the tuple of the first k messages.

In this section we define the model for communication. Then in Section 3.1 we will define a measure of suspicion, and show how this can be used to measure the amount of information, $I(X;T)$, the leakers can reveal. Here the amount of information revealed is measured in mutual information, which means that Joe might only get probabilistic information. In Section 3.2 we show how we can ensure *reliable leakage*, that is, ensure that with probability $1 - \epsilon$ Joe's guess about what X is will be correct.

Throughout the chapter we will use the collaborating model, that is, we assume that non-leakers will follow any protocol we ask them to follow. In particular, the adversary is passive. In Chapter 6 we will define a model, where we do not need the non-leakers to collaborate, and we will see that any protocol in the collaborating model can be modified to work in the model where non-leakers are just following a known communication protocol with sufficient randomness. The model in Chapter 6 will be more useful in practice, however when showing the existence of protocols, it is easier first to do this in the collaborating model.

In the *collaborating model* we can tell all the players including the non-leaking players to follow some communication protocol, called a collaborating cryptogenography protocol. The messages sent by leaking player may depend on the value of

X, but the messages of non-leaking players have to be independent of X given the previous transcript. Formally, a *collaborating cryptogenography protocol* π specifies for any possible value t^k of the current transcript T^k:

- Should the communication stop or continue, and if it should continue,
- Who is next to send a message, say PLR_i, and
- A distribution $p_?$ and a set of distributions, $\{p_x\}_{x\in\mathcal{X}}$ (the distributions $p_?$ and $\{p_x\}_{x\in\mathcal{X}}$ depend on π and t^k). Now PLR_i should choose a message using $p_?$, if $L_i = 0$ and choose a message using p_x if $L_i = 1$ and $X = x$.

Furthermore, for any protocol π, there should a number $length(\pi)$ such that the protocol will always terminate after at most $length(\pi)$ messages. We assume that both Joe and Eve know the protocol. They also know the prior distribution of $(X, L_1, \ldots, L_n)$, and we assume that they have unbounded computational power, in particular they have enough computational power to compute $(X, L_1, \ldots, L_n)|_{T=t}$ for any transcript t.

Another way of stating the above definition of collaborating cryptogenography protocols is that the players follow a communication protocol[1], and the leakers are given x as input while the non-leakers are given a fixed input, say "you are not a leaker", which is not in $\mathcal{X}$.

While we think of different players as different people, two or more different players could be controlled by the same person. For example, they could be communicating in a chat room with perfect anonymity, except that a profile's identity will be revealed if the profile can be shown to be guilty in leaking with probability greater than 95%. Here each player would correspond to a profile, but the same person could have more profiles. However, we will use "player" and "person" as synonyms.

[1]These were first defined by Yao [78]. Unlike in [78] we allow more than two players, allow the protocol to specify who to send the next message, and allow each message to be more than one bit. All this is standard in communication complexity, see for example Kushilevitz and Nisan [54].

3.1 Bounds on $I(X;T)$

3.1.1 Suspicion

First we will look at the problem where only one player is communicating and she may or may not be trying to leak information. We will later use these results when we analyse the many-player problem.

In the one player case, Alice sends one message A. If she is not trying to leak information, she will choose this message in $\mathcal{A}$ randomly using a distribution $p_?$. If she is trying to leak information, and $X = x$, she will use a distribution p_x. For a random variable Y jointly distributed with L and a value $y \in \mathcal{Y}$ with $\Pr(Y = y) > 0$ we let $c_{Y=y} = \Pr(L = 1|Y = y)$. We usually suppress the random variable, and write c_y instead. Here Y could be a tuple of random variables, and y a tuple of values. If $y = (y_1, y_2)$ is a tuple, we write $c_{y_1 y_2}$ instead of $c_{(y_1,y_2)}$.

We want to see how much information Alice can leak to Joe (by choosing the p's), without being too suspicious to Eve. The following measure of suspicion turns out to be useful.

Definition 3.1. Let Y be a random variable jointly distributed with L. Then the *suspicion (of Alice) given* $Y = y$ is

$$\begin{aligned}\operatorname{susp}(Y = y) &= -\log(1 - c_y) \\ &= -\log(\Pr(L = 0|Y = y)).\end{aligned}$$

We see that $\operatorname{susp}(Y = y)$ depends on y and the joint distribution of L and Y, but to keep notation simple, we suppress the dependence on L. The suspicion of Alice measures how suspicious Alice is to someone who knows that $Y = y$ and knows nothing more. For example Y could be the tuple that consists of the secret information X and the current transcript. We can think of the suspicion as the surprisal of the event "Alice did not have the information".

Next we define the suspicion given a random variable Y, without setting it equal to something.

Definition 3.2. The *expected suspicion (of Alice) given* Y or just the *suspicion (of*

Alice) given Y is

$$\begin{aligned}\operatorname{susp}(Y) =& \mathbb{E}_y \operatorname{susp}(Y = y) \\ =& \sum_{y \in \mathcal{Y}} \Pr(Y = y) \operatorname{susp}(Y = y). \end{aligned} \tag{3.1}$$

In each of these definitions, Y can consist of more than one random variable, e.g. $Y = (X, A)$. Finally, we can also combine these two definitions, giving

$$\operatorname{susp}(X, A = a) = \sum_{x \in \mathcal{X}} \Pr(X = x | A = a) \operatorname{susp}((X, A) = (x, a)).$$

Where X and A can themselves be tuples of random variables.

From the definitions imply we also get

$$\begin{aligned}\operatorname{susp}(X, A) =& \sum_{a \in \mathcal{A}, x \in \mathcal{X}} \Pr(X = x, A = a) \operatorname{susp}(X = x, A = a) \\ =& \sum_{a \in \mathcal{A}} \Pr(A = a) \sum_{x \in \mathcal{X}} \Pr(X = x | A = a) \operatorname{susp}(X = x, A = a) \\ =& \sum_{a \in \mathcal{A}} \Pr(A = a) \operatorname{susp}(X, A = a),\end{aligned}$$

which can be thought of as (3.1) given X.

Example 1. As an example, we will consider how suspicion behaves in a simple protocol with only one player. We assume the secret X and the random variable L which indicates whether Alice is leaking are uniformly distributed on $\{0, 1\}$ independently from each other. Now

$$\operatorname{susp}(X = 0) = -\log(\Pr(L = 0 | X = 0)) = -\log(0.5) = 1.$$

Similarly, $\operatorname{susp}(X = 1) = 1$ so

$$\operatorname{susp}(X) = \Pr(X = 0) \operatorname{susp}(X = 0) + \Pr(X = 1) \operatorname{susp}(X = 0) = \frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 1 = 1.$$

That is, before any information has been revealed, we are 1 bit suspicious towards Alice. Now suppose Alice makes a guess about what X is. If $L = 1$ she knows

the true value and she will say this value, but otherwise she will guess uniformly at random from $\{0,1\}$. The resulting protocol is given more formally in Figure 3.1. We see that $\Pr(X = a|A = a) = \frac{3}{4}$, $\Pr(X = 1-a|A = a) = \frac{1}{4}$ and $\Pr(L = 0|X = A = a) = \frac{1}{3}$.

Input distribution: (X, L) uniformly distributed on $\{0,1\}^2$. Alice learns L and if $L = 1$ she also learns X.
Protocol:

1. If $L = 1$ then $A := X$, if $L = 0$, then $A \leftarrow \{0,1\}$.
2. Alice sends A.

Figure 3.1: Protocol from Example 1.

Now we get

$$\mathrm{susp}(X = 1-a, A = a) = -\log(\Pr(L = 0|X = 1-a, A = a)) = -\log(1) = 0.$$

This fits well with our interpretation of susp as how suspicious Eve should be towards Alice: if X is different from Alice's guess, Eve knows that Alice did not have the information, and hence, Eve should no longer be suspicious towards her. On the other hand we have

$$\mathrm{susp}(X = a, A = a) = \log(\Pr(L = 0|X = a, A = a)) = -\log\left(\frac{1}{3}\right) \approx 1.585,$$

that is, if Eve knows X and see that Alice's guess of X is correct, Eve should get more suspicious towards Alice. Next we will compute Eve's expected suspicion before X is revealed.

$$\begin{aligned}\mathrm{susp}(X, A = a) =& \Pr(X = a|A = a)\,\mathrm{susp}(X = a, A = a)\\ &+ \Pr(X = 1-a|A = a)\,\mathrm{susp}(X = 1-a|A = a)\\ =& \frac{3}{4}\log\left(\frac{1}{3}\right) + \frac{1}{4}\cdot 0\\ \approx& 1.189.\end{aligned}$$

The interpretation is this: suppose Eve knows that Alice's message is a and Eve knows that she will soon learn X. How suspicious does Eve expect she will be towards Alice once she learns X? Notice that this is different from $\text{susp}(A = a)$ which is how suspicious Eve currently is towards Alice: while an observer's probability that Alice is leaking is a martingale, the suspicion is not.

We can now compute the expected suspicion before anything is revealed.

$$\begin{aligned}\text{susp}(X, A) =& \sum_{a\in\{0,1\}} \Pr(A = a)\,\text{susp}(X, A = a)\\ \approx& \frac{1}{2}1.189 + \frac{1}{2}1.189\\ =& 1.189.\end{aligned}$$

If Alice did not send any messages, Eve's expected suspicion towards Alice would be $\text{susp}(X) = 1$, so this protocol causes Alice to look, on average, 0.189 bits more suspicious that if she did not send any messages. At the same time the amount of information an observer gets about X from A in this protocol is given by

$$\begin{aligned}I(X; A) =& H(X) - H(X|A)\\ =& 1 - \left(-\frac{3}{4}\log\left(\frac{3}{4}\right) - \frac{1}{4}\log\left(\frac{1}{4}\right)\right)\\ \approx& 0.189.\end{aligned}$$

As we will see in the Lemma below, this is not a coincidence.

The above example has some similarities to randomized response [71]: there is a truth X and Alice message depends probabilistically on X. However, there is also an important difference. In randomized response, Alice would spin a spinner that shows 0 with probability $\frac{1}{3}$ and 1 otherwise, and she would announce if the spinner showed X or $1 - X$. First of all this means that Alice never has to say "0" or "1" but instead says "same" or "different". This could make a difference in a survey where, instead of "1" we have "I have been unfaithful to my husband". Whether or not she has been unfaithful, she might be more comfortable saying "same" or "different", especially if the interview is recorded. Secondly, and more importantly, in randomized response Alice would always be actively collaborating. In contrast,

when Alice in the above example sends a message which improves the observer's belief about X, it will sometimes be by chance (when $L = 0$) and sometimes because she was deliberately sending information (when $L = 1$). This could potentially make both a psychological and a legal difference.

When Alice sends a message A this might reveal some information about X, but at the same time, she will also reveal some information about whether she is trying to leak X. We would like to bound $I(A; X)$ by the information A reveals about L. This is not possible. If, for example, we set $A = X$ whenever $L = 1$ and $A = a \notin \mathcal{X}$ when $L = 0$, then $I(A; X) = \Pr(L = 1)H(A)$ which can be arbitrarily large, but $I(A; L) \leq H(L) \leq 1$. Instead, we will generalize the example above, and show that $I(A; X)$ can be bounded by the expected increase in suspicion given X, and that this bound is tight.

Lemma 3.1. *If Alice sends a message A, we have*

$$I(X; A) \leq \operatorname{susp}(X, A) - \operatorname{susp}(X).$$

That is, the amount of information she sends about X is at most her expected increase in suspicion given X. There is equality if and only if the distribution of A is the same as $A|_{L=0}$.

Proof. With no information revealed, Alice's suspicion given X is

$$\operatorname{susp}(X) = -\sum_{x \in \mathcal{X}} \Pr(X = x) \log(1 - c_x).$$

We want to compute Alice's suspicion given X and her message A.

$$\begin{aligned}
\operatorname{susp}(X, A) &= \sum_{x,j} \Pr(X = x, A = a) \operatorname{susp}(X = x, A = a) \\
&= -\sum_{x,a} \Pr(X = x, A = a) \log(1 - c_{xa}) \\
&= -\sum_{x,a} \Pr(X = x, A = a) \left(\log(1 - c_x) + \log\left(\frac{1 - c_{xa}}{1 - c_x} \right) \right).
\end{aligned}$$

Now it follows that the cost in suspicion given X of sending A is

$$\mathrm{susp}(X, A) - \mathrm{susp}(X) = -\sum_{x,a} \Pr(X = x, A = a) \log\left(\frac{1 - c_{xa}}{1 - c_x}\right). \tag{3.2}$$

Next we want to see how much information A gives about X, that is $I(A; X) = H(A) - H(A|X)$. We claim that this is bounded by the cost in suspicion, or equivalently, $H(A) \leq \mathrm{susp}(X, A) - \mathrm{susp}(X) + H(A|X)$. First we compute $H(A|X)$ using (1.1):

$$\begin{aligned} H(A|X) &= \sum_x \Pr(X = x) H(A|X = x) \\ &= -\sum_x \Pr(X = x) \sum_a \Pr(A = a|X = x) \log(\Pr(A = a|X = x)) \\ &= -\sum_{x,a} \Pr(X = x, A = a) \log(\Pr(A = a|X = x)). \end{aligned} \tag{3.3}$$

We have

$$\begin{aligned} &\frac{1 - c_{xa}}{1 - c_x} \Pr(A = a|X = x) \\ &\quad = \frac{\Pr(L = 0|X = x, A = a)}{\Pr(L = 0|X = x)} \Pr(A = a|X = x) \\ &\quad = \frac{\Pr(L = 0, X = x, A = a)}{\Pr(X = x, A = a)} \frac{\Pr(X = x)}{\Pr(L = 0, X = x)} \frac{\Pr(X = x, A = a)}{\Pr(X = x)} \\ &\quad = \frac{\Pr(L = 0, X = x, A = a)}{\Pr(L = 0, X = x)} \\ &\quad = \Pr(A = a|X = x, L = 0) \\ &\quad = \Pr(A = a|L = 0) \end{aligned} \tag{3.4}$$

Here, the last equation follows from the assumption that A does not depend on X

when $L = 0$. From this we conclude

$$\begin{aligned}
&\mathrm{susp}(X, A) - \mathrm{susp}(X) + H(A|X) \\
&\qquad = -\sum_{x,a} \Pr(X = x, A = a) \log\left(\frac{1 - c_{xa}}{1 - c_x} \Pr(A = a|X = x)\right) \\
&\qquad = -\sum_{x,a} \Pr(X = x, A = a) \log\left(\Pr(A = a|L = 0)\right) \\
&\qquad = -\sum_{a} \Pr(A = a) \log\left(\Pr(A = a|L = 0)\right) \\
&\qquad \geq -\sum_{a} \Pr(A = a) \log(\Pr(A = a)) \\
&\qquad = H(A).
\end{aligned}$$

Here the first equality follows from (3.2) and (3.3), the second follows from (3.4) and the inequality follows from inequality (1.4). There is equality if and only if $\Pr(A = a) = \Pr(A = a|L = 0)$ for all a. □

We will now turn to the problem where many people are communicating. We assume that they send messages one at a time, so we can break the protocol into time periods where only one person is communicating, and see the entire protocol as a sequence of one player protocols. To make the notation simpler, we will assume that the protocol runs for a fixed number of messages, and the player to talk in round k only depends on k, not on which previous messages was sent. Any protocol π can be turned into such a protocol π' by adding dummy messages: In round k of π' we let $\mathrm{PLR}_{k \bmod n}$ talk. They follow protocol π in the sense that if it is not $\mathrm{PLR}_{k \bmod n}$ turn to talk according to π she sends some fixed message 1, and if it is her turn, she chooses her message as in π. The following corollary shows that a statement similar to Lemma 3.1 holds for each single message in a protocol with many players.

Corollary 3.2. *Let (L, T^{k-1}, X) have some joint distribution, where T^{k-1} denotes previous transcript. Let T_k be the next message sent by Alice. Then*

$$I(X; T_k|T^{k-1}) \leq \mathrm{susp}(X, T^k) - \mathrm{susp}(X, T^{k-1}).$$

Proof. For a particular value t^{k-1} of T^{k-1} we use Lemma 3.1 with $(X, T_k)|_{T^{k-1}=t^{k-1}}$

as (X, A). This gives us

$$I(X; T_k | T^{k-1} = t^{k-1}) \leq \text{susp}(X, T_k, T^{k-1} = t^{k-1}) - \text{susp}(X, T^{k-1} = t^{k-1}).$$

By multiplying each side by $\Pr(T^{k-1} = t^{k-1})$ and summing over all possible t^{k-1} we get the desired inequality. □

A protocol consists of a sequence of messages that each leaks some information and increases the suspicion of the sender. We can add up the increases in suspicion, and using the chain rule for mutual information we can also add up the amount of revealed information. However, Bob's message might not only affect his own suspicion, it might also affect Alice's suspicion. To show an upper bound on the amount of information the players can leak, we need to show that one person's message will, in expectation, never make another person's suspicion decrease. We get this from the following proposition by setting $Y = (X, T^{k-1})$ and $B = T_k$.

Proposition 3.3. *For any joint distribution on* (L, Y, B) *we have* $\text{susp}(Y) \leq \text{susp}(Y, B)$.

Proof. We have

$$\begin{aligned}
\text{susp}(Y = y) &= -\log(\Pr(L = 0 | Y = y)) \\
&= -\log\left(\sum_{b \in \mathcal{B}} \Pr(B = b | Y = y) \Pr(L = 0 | Y = y, B = b)\right) \\
\text{susp}(Y = y, B) &= -\sum_{b \in \mathcal{B}} \Pr(B = b | Y = y) \log\left(\Pr(L = 0 | Y = y, B = b)\right).
\end{aligned}$$

As $p \mapsto -\log(p)$ is convex, Jensen's inequality gives us

$$\text{susp}(Y = y, B) \geq \text{susp}(Y = y).$$

Multiplying each side by $\Pr(Y = y)$ and summing over all $y \in \mathcal{Y}$ gives us the desired inequality. □

Let susp_i denote the suspicion of PLR_i.[2]

[2]This is defined similar to the suspicion of Alice, except using L_i instead of L.

Theorem 3.4. *If T is the transcript of the entire protocol we have*

$$I(X;T) \leq \sum_{i=1}^{n} \left(\mathrm{susp}_i(X,T) - \mathrm{susp}_i(X)\right).$$

Proof. From the chain rule for mutual information, we know that

$$I(X;T) = \sum_{k=1}^{length(\pi)} I(X;T_k|T^{k-1}).$$

Now Corollary 3.2 shows that $I(X;T_k|T^{k-1}) \leq \mathrm{susp}_i(X,T^k) - \mathrm{susp}_i(X,T^{k-1})$ if $\textsc{plr}_i$ send the k'th message and Proposition 3.3 shows that $\mathrm{susp}_{i'}(X,T^k) \geq \mathrm{susp}_{i'}(X,T^{k-1})$ for all other i'. By summing over all players, we get

$$I(X;T_k|T^{k-1}) \leq \sum_{i=1}^{n} \left(\mathrm{susp}_i(X,T^k) - \mathrm{susp}_i(X,T^{k-1})\right).$$

By summing over all rounds in the protocol, we get the theorem. □

3.1.2 Keeping reasonable doubt

Until now we have bounded the amount of information the players can leak by the expected increase in some measure, suspicion, that we defined for the purpose. But there is no reason to think that someone who is leaking information cares about the expectation of this measure. A more likely scenario, is that each person leaking wants to ensure reasonable doubt, that is, they want to ensure that after the leakage, an observer who knows X will assign probability at most b_m to the event that she was leaking information: $\Pr(L_i = 1|X = x, T = t) \leq b_m$. If this is the case for all x

and after all possible transcripts t, we see that

$$\begin{aligned}\operatorname{susp}_i(X,T) &= \sum_{x,t} \Pr(X=x,T=t)\operatorname{susp}_i(X=x,T=t)\\ &= \sum_{x,t} \Pr(X=x,T=t)(-\log(\Pr(L_i=0|X=x,T=t)))\\ &\leq \sum_{x,t} \Pr(X=x,T=t)(-\log(1-b_m))\\ &= -\log(1-b_m).\end{aligned}$$

If we assume that each player before the protocol had probability $b_l < b_m$ of leaking independently of X, that is $\Pr(L_i = 1|X = x) = b_l$ for all x and i, we have $\operatorname{susp}_i(X) = -\log(1-b_l)$. Thus

$$I(X;T) \leq \sum_{i=1}^{n} (\operatorname{susp}_i(X,T) - \operatorname{susp}_i(X)) = (-\log(1-b_m) + \log(1-b_l))\, n. \quad (3.5)$$

This gives us an upper bound on how much information the leakers can reveal. However, it is not possible to reach this bound: to reach it, we would need to have $\Pr(L_i = 1|X = x, T = t) = b_m$ for all x, t, i. But the probability $\Pr(L_i = 1|X = x) = b_l$ can also be computed as $\mathbb{E}_{t\sim T} \Pr(L_i = 1|X = x, T = t)$, so $\Pr(L_i = 1|X = x, T = t)$ cannot be constantly $b_m > b_l$. The following theorem improves the upper bound from (3.5) by taking this into account.

Theorem 3.5. *Let π be a collaborating cryptogenography protocol, and T be its transcript. If for all players PLR_i and all $x \in \mathcal{X}$ and all transcripts t we have $\Pr(L_i = 1|X = x) = b_l$, and $\Pr(L_i = 1|T = t, X = x) \leq b_m$ then*

$$\begin{aligned}I(X;T) \leq& \frac{-b_l \log(1-b_m) + b_m \log(1-b_l)}{b_m} n\\ =& nD\left(\frac{b_l}{b_m}\middle|\middle|\frac{b_l(1-b_m)}{b_m(1-b_l)}\right).\end{aligned}$$

For an illustration of this theorem, see Figure 3.2.

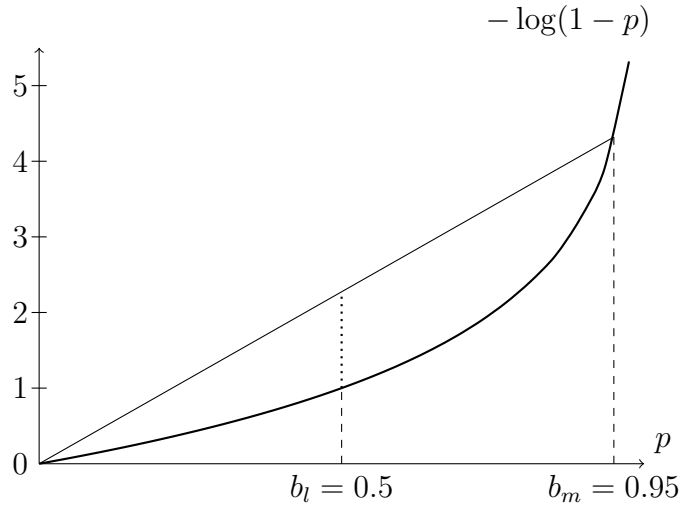

Figure 3.2: This figure illustrates Theorem 3.5. The curve shows the function $p \mapsto -\log(1-p)$, which is used for computing the suspicion. The line from $(0,0)$ to $(b_m, -\log(1-b_m))$ shows the maximum expected posterior suspicion PLR_i can have if she started with $\Pr(L_i = 1|X = x) = p$ and have $\Pr(L_i|X = x, T = t) \leq b_m$ for all transcripts t. The second coordinate of $(b_l, -\log(1-b_l))$ gives the prior suspicion towards PLR_i, so the dotted line gives the amount of information that player i can leak.

Proof. If $\Pr(L_i = 1|X = x, T = t) \leq b_m$ then

$$\begin{aligned}\text{susp}_i(X = x, T = t) &= -\log(1 - \Pr(L_i = 1|X = x, T = t)) \\ &\leq \frac{-\log(1-b_m)}{b_m} \Pr(L_i = 1|X = x, T = t). \qquad (3.6)\end{aligned}$$

This inequaility follows from the fact that we have equality when $\Pr(L_i = 1|X = x, T = t)$ is 0 or b_m, and the left hand side is convex in $\Pr(L_i = 1|X = x, T = t)$ while the right hand side is linear.

Let π and T be as in the assumptions. Now we get

$$\begin{aligned}
\operatorname{susp}_i(X,T) =& \sum_{x,t} \Pr(X=x,T=t)\operatorname{susp}_i(X=x,T=t)\\
\leq& \sum_{x,t} \Pr(X=x,T=t)\frac{-\log(1-b_m)}{b_m}\Pr(L_i=1|X=x,T=t)\\
=& \sum_{x,t} \frac{-\log(1-b_m)}{b_m}\Pr(L_i=1,X=x,T=t)\\
=& \frac{-\log(1-b_m)}{b_m}\Pr(L_i=1)\\
=& \frac{-b_l\log(1-b_m)}{b_m}.
\end{aligned}$$

Thus,

$$\begin{aligned}
I(X;T) \leq& \sum_{i=1}^{n}\left(\operatorname{susp}_i(X,T)-\operatorname{susp}_i(X)\right)\\
\leq& \left(\frac{-b_l\log(1-b_m)}{b_m}-(-\log(1-b_l))\right)n\\
=& \frac{-b_l\log(1-b_m)+b_m\log(1-b_l)}{b_m}n\\
=& nD\left(\frac{b_l}{b_m}\middle|\middle|\frac{b_l(1-b_m)}{b_m(1-b_l)}\right).
\end{aligned}$$

□

It is clear that the upper bound from Theorem 3.5 cannot be achieved for all distributions of $(X, L_1, \ldots, L_n)$. If for example $H(X) < nD\left(\frac{b_l}{b_m}\middle|\middle|\frac{b_l(1-b_m)}{b_m(1-b_l)}\right)$ we must also have $I(X;T) \leq H(X) < nD\left(\frac{b_l}{b_m}\middle|\middle|\frac{b_l(1-b_m)}{b_m(1-b_l)}\right)$, that is, the players do not have enough information to send to reach the upper bound. Even if $H(X)$ is high, we may not be able to reach the upper bound. If it is known that $L_1 = L_2 = \cdots = L_n$ the suspicion of the players will not depend on the player, only on the messages sent. So this problem will be equivalent to the case where only one person is sending messages.

We will now give an example where the upper bound from Theorem 3.5 is achievable. We will refer back to this example when we prove that reliable leakage is

possible.

Example 2. Assume that $X, L_1, \ldots, L_n$ are all independent, and $\Pr(L_i = 1) = b_l$ for all i. Furthermore, assume that $0 < b_l < b_m < 1$ and that $\frac{b_l(1-b_m)}{b_m(1-b_l)}$ is a rational number. Let $d, a \in \mathbb{N}$ be the smallest natural numbers such that $\frac{a}{d} = \frac{b_l(1-b_m)}{b_m(1-b_l)}$. We see that $\frac{b_l(1-b_m)}{b_m(1-b_l)} \in (0,1)$ so $0 < a < d$.

First we need a few implications of $\frac{a}{d} = \frac{b_l(1-b_m)}{b_m(1-b_l)}$. By multiplying both sides by $\frac{1-b_l}{a}$ we see that $\frac{1-b_l}{d} = \frac{b_l(1-b_m)}{b_m a} = \frac{b_l}{b_m a} - \frac{b_l}{a}$. By rearranging we get

$$\frac{b_l}{a} + \frac{1-b_l}{d} = \frac{b_l}{ab_m}. \tag{3.7}$$

If we instead multiply $\frac{a}{d} = \frac{b_l(1-b_m)}{b_m(1-b_l)}$ by $\frac{1-b_l}{(1-b_m)a}$ on both sides we get $\frac{1-b_l}{d(1-b_m)} = \frac{b_l}{ab_m}$. Together with equation (3.7) this implies

$$\frac{b_l}{a} + \frac{1-b_l}{d} = \frac{1-b_l}{d(1-b_m)}. \tag{3.8}$$

Finally, from $\frac{a}{d} = \frac{b_l(1-b_m)}{b_m(1-b_l)}$ we also get

$$\begin{aligned}
\frac{d-a}{d} =& 1 - \frac{a}{d} \\
=& 1 - \frac{b_l(1-b_m)}{b_m(1-b_l)} \\
=& \frac{b_m - b_m b_l - b_l + b_m b_l}{b_m(1-b_l)} \\
=& \frac{b_m - b_l}{b_m(1-b_l)}.
\end{aligned}$$

By multiplying through by $1 - b_l$ this gives us

$$(d-a)\frac{1-b_l}{d} = 1 - \frac{b_l}{b_m}. \tag{3.9}$$

We are now ready for the example. Assume that X is uniformly distributed on $\{1, \ldots, d\}^n$. Each player PLR_i now sends one message, independently of which messages the other players send. If $L_i = 0$, PLR_i chooses a message in $\{1, \ldots, d\}$

uniformly at random. If $L_i = 1$ and $X_i = x_i$, then PLR_i chooses a message in

$$\{x_i + 1, x_i + 2 \ldots, x_i + a\} \mod d$$

uniformly at random. Here we use $k \mod d$ to mean the number in $\{1, \ldots d\}$ that is equal to k modulo d. We can also write this set as $x_i + [a] \mod d$. The resulting protocol is defined more formally in Figure 3.3.

Parameters:

n : number of players

b_l : probability of each player being leaker

$a < d$: two natural numbers

Input distribution: $X, L_1, \ldots, L_n$ are independently distributed, X uniformly distributed on $[d]^n$ and for each i: $\Pr(L_i = 1) = b_l$. Each PLR_i learns L_i and if $L_i = 1$ she also learns X.

Protocol:

1. For i from 1 to n:
2. If $L_i = 1$ then $T_i \leftarrow X_i + [a] \mod d$, if $L_i = 0$, then $T_i \leftarrow [d]$
3. Player i sends T_i

Figure 3.3: Protocol from Example 2.

We see that over random choice of X, the message, T_i, that PLR_i sends, is uniformly distributed on $\{1, \ldots, d\}$, so $H(T_i) = \log(d)$. We want to compute $H(T_i|X)$. Given X, each of the $d - a$ elements not in $x_i + [a] \mod d$ can only be sent if $L = 0$, so they will be sent with probability $\frac{1-b_l}{d}$. Each of the a elements in the set $x_i + [a]$

mod d are sent with probability $\frac{b_l}{a} + \frac{1-b_l}{d}$. Thus,

$$\begin{aligned}
H(T_i|X=x) =& -\sum_{t_i \in \mathcal{T}_i} \Pr(T_i = t_i|X=x) \log(\Pr(T_i = t_i|X=x)) \\
=& -a\left(\frac{b_l}{a} + \frac{1-b_l}{d}\right) \log\left(\frac{b_l}{a} + \frac{1-b_l}{d}\right) - (d-a)\frac{1-b_l}{d} \log\left(\frac{1-b_l}{d}\right) \\
=& -\frac{b_l}{b_m} \log\left(\frac{1-b_l}{d(1-b_m)}\right) - \left(1 - \frac{b_l}{b_m}\right) \log\left(\frac{1-b_l}{d}\right).
\end{aligned}$$

The last equality follows from (3.7), (3.8) and (3.9). As this holds for all x, we get.

$$H(T_i|X) = -\frac{b_l}{b_m} \log\left(\frac{1-b_l}{d(1-b_m)}\right) - \left(1 - \frac{b_l}{b_m}\right) \log\left(\frac{1-b_l}{d}\right).$$

Now

$$\begin{aligned}
I(T_i; X) =& H(T_i) - H(T_i|X) \\
=& \log(d) + \frac{b_l}{b_m} \log\left(\frac{1-b_l}{d(1-b_m)}\right) + \left(1 - \frac{b_l}{b_m}\right) \log\left(\frac{1-b_l}{d}\right) \\
=& \log(1-b_l) - \frac{b_l}{b_m} \log(1-b_m) \\
=& \frac{-b_l \log(1-b_m) + b_m \log(1-b_l)}{b_m} \\
=& D\left(\frac{b_l}{b_m} \middle|\middle| \frac{b_l(1-b_m)}{b_m(1-b_l)}\right). \qquad (3.10)
\end{aligned}$$

The tuples (X_i, T_i, L_i) where i ranges over $\{1, \ldots n\}$ are independent from each other, so we have $I(T; X) = nD\left(\frac{b_l}{b_m} \middle|\middle| \frac{b_l(1-b_m)}{b_m(1-b_l)}\right)$ as wanted.

Finally, we need to show that the leakers have reasonable doubt, that is, $\Pr(L_i = 1|T = t, X = x) \leq b_m$. The probability is 0 if PLR_i sends a message not in $x_i + [a]$

mod d. Otherwise, we use independence and then Bayes' Theorem to get

$$\begin{aligned}
\Pr(L_i = 1 | T = t, X = x) =& \Pr(L_i = 1 | T_i = t_i, X_i = x_i) \\
=& \frac{\Pr(T_i = t_i | L_i = 1, X_i = x_i) \Pr(L_i = 1 | X_i = x_i)}{\Pr(T_i = t_i | X_i = x_i)} \\
=& \frac{\frac{1}{a} b_l}{\frac{b_l}{a} + \frac{1 - b_l}{d}} \\
=& \frac{\frac{b_l}{a}}{\frac{b_l}{a b_m}} \\
=& b_m. \qquad (3.11)
\end{aligned}$$

Here the penultimate equality follows from (3.7).

3.1.3 Why use reasonable doubt?

The requirement of reasonable doubt, which can be formally stated as

$$\forall i, x, t : \Pr(L_i = 1 | X = x, T = t) \leq b_m,$$

is easily confused with the requirement that an observer can only guess a leaker with probability at most b_m, that is

$$\forall d : \Pr(L_{d(X,T)} = 1) \leq b_m$$

where the for-all quantifier is over all *decision functions*: functions d that map a tuple (x, t) of a secret and a transcript to a guess of who is a leaker. This last requirement is equivalent to the probability of error, as defined by Chatzikokolakis, Palamidessi and Panangaden [12][3], being at most $1 - b_m$. However, we will see that the probability-of-error requirement is weaker than the corresponding requirement of reasonable doubt. To prove this, we will first give an alternative way of defining reasonable doubt.

Definition 3.3. An *incomplete decision function* d' is a function that for each tuple (x, t) of secret and transcript will either return the number i of a player or a special

[3]Note, however, that [12] only gave this definition in a model with only one sender.

symbol $\perp$, and such that there is at least one tuple (x,t) with $\Pr(X=x,T=t)>0$ such that $d'(x,t)\neq\perp$.

We let $\mathcal{D}'$ denote the set of all incomplete decision functions, and let $\mathcal{D}$ denote the set of *complete decision functions*, that is $d\in\mathcal{D}'$ that does not take the value $\perp$.

We can use incomplete decision functions to give an alternative characterization of reasonable doubt: we have reasonable doubt at the b_m level if and only if an adversary (modelled by an incomplete decision function) can guess a leaker with probability at most b_m *given that the adversary makes a guess.* This is formalised by the following theorem.

Theorem 3.6. *For any distribution of* (L,X,T) *we have*

$$\forall i,x,t:\ \Pr(L_i=1|X=x,T=t)\leq b_m$$

if and only if

$$\forall d'\in\mathcal{D}':\ \Pr(L_{d'(X,T)}=1|d'(X,T)\neq\perp)\leq b_m.$$

The intuition is that the incomplete decision function d' that maximizes its probability of guessing correctly given that it does not return $\perp$ is the function that only returns $\perp$ for the input (x,t) where the observer is most confident about who is a leaker. So both the characterizations give us the level of confidence the observer will have in the worst case.

Proof. "$\Rightarrow$" Suppose L,X,T satisfy $\forall i,x,t:\ \Pr(L_i=1|X=x,T=t)\leq b_m$. Then we have

$$\begin{aligned}
&\Pr(L_{d'(X,T)}=1|d'(X,T)\neq\perp)\\
&\quad=\frac{\sum_{(x,t):d'(x,t)\neq\perp}\Pr(X=x,T=t)\Pr(L_{d'(x,t)}=1|X=x,T=t)}{\sum_{(x,t):d'(x,t)\neq\perp}\Pr(X=x,T=t)}\\
&\quad\leq\frac{\sum_{(x,t):d'(x,t)\neq\perp}\Pr(X=x,T=t)b_m}{\sum_{(x,t):d'(x,t)\neq\perp}\Pr(X=x,T=t)}\\
&\quad=b_m.
\end{aligned}$$

In the inequality we are using that when x and t are fixed, $d'(x,t)$ is a constant. "$\Leftarrow$": We show this by contraposition: suppose for contradiction that there exist i_0, x_0, t_0 with $\Pr(L_{i_0} = 1 | X = x_0, T = t_0) > b_m$. Then we define the function $d'_{i_0,x_0,t_0} \in \mathcal{D}'$ which on input (x_0, t_0) returns i_0 and on all other inputs (x,t) returns $\perp$. Now

$$\Pr(L_{d'_{i_0,x_0,t_0}(X,T)} = 1 | d'_{i_0,x_0,t_0}(X,T) \neq \perp) = \Pr(L_{i_0} = 1 | X = x_0, T = t_0) > b_m.$$

□

This alternative characterization makes it easy to compare reasonable doubt with the probability of error.

Corollary 3.7. *If L, X, T give reasonable doubt at the b_m level, the probability of error is at least $1 - b_m$.*

This shows that the requirement of reasonable doubt is stronger that the requirement of large probability of error. Below we will see that it is strictly stronger, but first we will prove the corollary.

Proof. Assume that L, T, X gives reasonable doubt at the b_m level, then by Theorem 3.6 we have $\forall d' \in \mathcal{D}' : \Pr(L_{d'(X,T)} = 1 | d'(X,T) \neq \perp) \leq b_m$. As $\mathcal{D} \subset \mathcal{D}'$ we have in particular $\forall d \in \mathcal{D} : \Pr(L_{d(X,T)} = 1 | d(X,T) \neq \perp) = \Pr(L_{d(X,T)} = 1) \leq b_m$ which is equivalent to the probability of error being at least $1 - b_m$. □

To see that the opposite implication does not hold, suppose that $n = 100$, and L is such that only one person, chosen uniformly at random, is a leaker. Suppose further that with probability 10% the transcript T completely reveals the leaker, and otherwise gives no information about the leaker. Now the probability of error is $(1 - 10\%) \cdot \left(1 - \frac{1}{100}\right) = 0.891$. However, the transcript sometimes completely reveals the leaker, so $\Pr(L_i = 1 | X = x, T = t)$ is not bounded by any number $b_m < 1$. This shows that the requirement of reasonable doubt at the b_m level is a strictly stronger than the requirement of a probability of error of at least $1 - b_m$.

Theorem 3.6 shows that reasonable doubt provides a guarantee to the users of the anonymity system: we can interpret an incomplete decision function d' as an action taken against a player. If d' returns $\perp$ then no action is taken, if d' returns

a player then that player might be punished or investigated further. Theorem 3.6 shows that if an adversary takes action against someone, based only on this person's participation in an anonymity system which provides reasonable doubt at the b_m level, the adversary will, with probability at least $1 - b_m$ given that the adversary takes action, take action against someone who is not a leaker. Such a guarantee is particularly useful if the adversary is limited by legal rights, if the possible action is expensive for the adversary or if the adversary for some other reason is reluctant to take action against someone who could be innocent. In particular, let a *rational adversary* be an adversary who gets utility $r_l > 0$ for punishing a leaker and utility $r_i < 0$ for punishing an innocent and who chooses whether to punish in a way that optimizes this utility.[4] Such a rational adversary will use its punishment on a player for some transcripts if and only if the player does not have reasonable doubt at the b_m level for $b_m = \frac{r_i}{r_l - r_i}$.

Unfortunately, the requirement of reasonable doubt gives no guarantee that the leaker is not going to stand out. For example, suppose it is known that there is exactly one leaker and we want reasonable doubt at the 10% level. This could be achieved if, given the communication, 10 people are equally likely to be the leaker. However, it could also be that given the communication, one person is a leaker will probability 10% and 90 other people are each the leaker with probability 1%. Against a rational adversary this should not make any difference to the person who is a leaker with probability 10%. But if the adversary has other motives, such as not wanting to look weak or wanting to intimidate people, the person who appears guilty with probability 10% might be in more danger in the second scenario that in the first.

In the case where the number of leakers is not known, this effect can be even more extreme. Consider the protocol from Example 2 with $b_l = 0.1$, $b_m = 0.9$ (giving $a = 1$ and $d = 81$) and $n = 90$. In expectation, $n(b_l + (1 - b_l)\frac{a}{d}) = \frac{90}{9} = 10$ players should send a message that is consistent with being a leaker, and each of these would have $\Pr(L_i = 1 | X = x, T = t) = 0.9$. So in expectation, one of these would be innocent. However, suppose that a very unlikely event happens: only one

[4]The term "rational adversary" might be misleading, as we are also assuming that the utility is on a certain form. In more complicated scenarios it might be perfectly rational for an adversary to punish people just to intimidating them to not leak information.

player sends a message that is consistent with being a leaker. The computations in Example 2 shows that for this player, i, we still have $\Pr(L_i = 1|X = x, T = t) = 0.9$, but an irrational adversary might be more likely to punish one person, who is most likely a leaker, than to punish 10 people, which most likely has at least one innocent among them (even though each of them is most likely a leaker).

Notice, however, that if we allow for the possibility that the true value of b_l might be different from the 0.1 the protocol was designed for, then a transcript where only one player sends a message consistent with being a leaker, is evidence that $b_l < 0.1$. In fact, even if $b_l = 0$ we would expect $90 \cdot \frac{1}{81} = \frac{10}{9} = 1.11\ldots$ people to send messages consistent with being leakers. So a rational adversary, who is uncertain about b_l, should be less likely to punish people if only a small number of them might be leakers. Conversely, if everyone sends messages consistent with being a leaker, that is strong evidence that $b_l \approx 1$ and a rational adversary might punish everyone.

To summarize, if you want to prevent being punished by a rational adversary, reasonable doubt is the right measure of your anonymity: the level of reasonable doubt tells you exactly for which utility functions the rational adversary would, for some transcripts, punish you. However, if the level of reasonable doubt is not sufficient to prevent you from being punished in every scenario, the level of reasonable doubt does not tell you how likely you get a transcript where you will be punished. On the other hand, if you want to prevent the leaker from being punished by an adversary who always punishes exactly one person, then the probability of error is the relevant measure of anonymity. You can easily imagine other type of adversaries, for example an adversary that will punish people if they are sufficiently more suspicious than everyone else, but we will not attempt to construct measures of anonymity for use when faced with such adversaries.

3.2 Reliable leakage

In Example 2, Joe would receive some information about X in the sense of information theory: before he sees the transcript, any value of X would be as likely as any other value, and when he knows the transcript, he has a much better idea about what X is. However, his best guess about what X is, is still very unlikely to be correct. In this section, we want to show that we can have reliable leakage. That

is, no matter what value X is taking, we want Joe to be able to guess the correct value with high probability. We will see that this is possible, even when X has entropy close to $nD\left(\frac{b_l}{b_m}\middle\|\frac{b_l(1-b_m)}{b_m(1-b_l)}\right)$. Joe's guess would have to be a function G of the transcript t. Saying that Joe will guess X correctly with high probability when $X = x$ is the same as saying that $\Pr(G(T) = x|X = x)$ is close to one.

Definition 3.4. Let $L = (L_1, \ldots, L_n)$ be a tuple of random variables, where the L_i takes values in $\{0, 1\}$.

A *risky* (n, h, L, b_m, ϵ)*-protocol* is a collaborating cryptogenography protocol together with a function G from the set of possible transcripts to $\mathcal{X} = \{1, \ldots, 2^{\lceil h \rceil}\}$ such that when X and L are distributed independently and X is uniformly distributed on $\mathcal{X}$, then for any $x \in \mathcal{X}$, there is probability at least $1 - \epsilon$ that a random transcript t distributed as $T|_{X=x}$ satisfies

Reasonable doubt: $\forall i : \Pr(L_i = 1|T = t, X = x) \leq b_m$, and

Reliable leakage: $G(t) = x$

That is, no matter the value of X, with high probability Joe can guess the value of X, and with high probability no player will be estimated to have leaked the information with probability greater than b_m by Eve. However, there might be a small risk that someone will be estimated to have leaked the information with probability greater than b_m. This is the reason we call it a risky protocol. A safe protocol is a protocol where this never happens.

Definition 3.5. A *safe* (n, h, L, b_m, ϵ)*-protocol* is a risky (n, h, L, b_m, ϵ)-protocol where $\Pr(L_i = 1|T = t, X = x) \leq b_m$ for all i, t, x with $\Pr(T = t, X = x) > 0$.

First we will consider the case where $L_1, \ldots, L_n$ are independent, and the L_i's all have the same distribution. The following definitions of achievability and capacity are based on the similar definitions from information theory, as given by Shannon [67], but instead of measuring these in bits per time unit or per usage of a channel, we measure them in bits per player.

Definition 3.6. Let $\mathrm{Indep}_{b_l}(n)$ be the random variable $(L_1, \ldots, L_n)$ where $L_1, \ldots, L_n$ are independent, and each L_i is distributed on $\{0, 1\}$ and $\Pr(L_1 = 1) = b_l$.

A rate R is *safely/riskily* $(\text{Indep}_{b_l}, b_m)$*-achievable* if for all $\epsilon > 0$ and all n_0, there exists a safe/risky $(n, nR, \text{Indep}_{b_l}(n), b_m, \epsilon)$-protocol with $n \geq n_0$.

The *safe/risky* $(\text{Indep}_{b_l}, b_m)$*-capacity* is the supremum of all safely/riskily $(\text{Indep}_{b_l}, b_m)$-achievable rates.

It turns out that the safe and the risky $(\text{Indep}_{b_l}, b_m)$-capacities are the same, but at the moment we will only consider the safe capacity.

Proposition 3.8. *No rate* $R > D\left(\frac{b_l}{b_m}\middle|\middle|\frac{b_l(1-b_m)}{b_m(1-b_l)}\right)$ *is safely* $(\text{Indep}_{b_l}, b_m)$*-achievable.*

Proof. Assume for contradiction that $R > D\left(\frac{b_l}{b_m}\middle|\middle|\frac{b_l(1-b_m)}{b_m(1-b_l)}\right)$ is safely $(\text{Indep}_{b_l}, b_m)$-achievable, and let π be a safe $(n, Rn, \text{Indep}_{b_l}(n), b_m, \epsilon)$-protocol. Let $\delta = R - D\left(\frac{b_l}{b_m}\middle|\middle|\frac{b_l(1-b_m)}{b_m(1-b_l)}\right) > 0$. We know from Theorem 3.5 that

$$I(X;T) \leq nD\left(\frac{b_l}{b_m}\middle|\middle|\frac{b_l(1-b_m)}{b_m(1-b_l)}\right) = (R-\delta)n.$$

Now

$$H(X|T) = H(X) - I(X;T) \geq Rn - (R-\delta)n = \delta n.$$

By Fano's inequality (1.3) we get that the probability of error for Joe's guess is

$$P_e \geq \frac{\delta n - 1}{nR}.$$

Thus, for sufficiently large n_0 and sufficiently small ϵ we cannot have $P_e \leq \epsilon$. When $\Pr(G(T) \neq X) = P_e > \epsilon$ there must exist an $x \in \mathcal{X}$ such that $\Pr(G(T) \neq x | X = x) > \epsilon$, so R is not safely b_m-achievable. □

The idea in the proof of the following theorem is to consider each person to be one usage of a channel. As the proof relies on Shannon's Noisy-Channel Coding Theorem it is not constructive.

Theorem 3.9. *Any rate* $R < D\left(\frac{b_l}{b_m}\middle|\middle|\frac{b_l(1-b_m)}{b_m(1-b_l)}\right)$ *is safely* $(\text{Indep}_{b_l}, b_m)$*-achievable.*

Proof. Let $R < D\left(\frac{b_l}{b_m}\middle|\middle|\frac{b_l(1-b_m)}{b_m(1-b_l)}\right)$ and let $b'_m \leq b_m$ be a number such that $\frac{b_l(1-b'_m)}{b'_m(1-b_l)}$ is rational and $R < D\left(\frac{b_l}{b'_m}\middle|\middle|\frac{b_l(1-b'_m)}{b'_m(1-b_l)}\right)$. We need to show that given $\epsilon > 0$ and n_0 there exists a $(n, Rn, \text{Indep}_{b_l}(n), b_m, \epsilon)$-protocol with $n \geq n_0$.

Use b_l and b'_m to define a and d as in Example 2: $\frac{a}{d} = \frac{b_l(1-b'_m)}{b'_m(1-b_l)}$. We consider the channel C that on input j with probability b_l returns a random uniformly distributed element in $\{j+1, j+2 \ldots, j+a\} \mod d$, and with probability $1-b_l$ it returns a random and uniformly distributed element in $\{1, \ldots, d\}$. Thus, C is the channel given by $\Pr(C(j) = t) = \frac{1-b_l}{d} + \frac{b_l}{a}$ if $t \in (j + [a] \mod d)$ and $\Pr(C(j) = t) = \frac{1-b_l}{d}$ for all other $t \in [d]$.

We see that if leaker choose their message uniformly from some set on the form $(j + [a] \mod d)$ and non-leaker choose their message uniformly form $[d]$ then each person sending a message corresponds exactly to using this channel once. The computation (3.10) from Example 2 shows that when input of this channel is uniformly distributed, the mutual information between input and output is $D\left(\frac{b_l}{b'_m} \middle|\middle| \frac{b_l(1-b'_m)}{b'_m(1-b_l)}\right)$. Thus, the capacity of the channel is at least this value.[5] We now use Shannon's Noisy-Channel Coding Theorem to get an error correcting code $\mathfrak{c} : \mathcal{X} \to \{1, \ldots, d\}^n$ for this channel, that achieves rate R, has $n \geq n_0$ and for each x fails with probability less than ϵ. Formally that means that

$$\forall x : \Pr(G(C^n(\mathfrak{c}(X))) \neq X | X = x) \leq \epsilon \tag{3.12}$$

where $G : \{1, \ldots, d\}^n \to \mathcal{X}$ is a decoding function for $\mathfrak{c}$ and $C^n = (C_1, \ldots, C_n)$ is repeated use of the channel C defined above: $C^n(j_1, \ldots, j_n) = (C(j_1), \ldots, C(j_n))$ and for fixed $j = (j_1, \ldots, j_n)$ all the coordinates of $C^n(j)$ are independent.

When $X = x$, any player PLR_i that is not leaking will send a message t_i chosen uniformly at random from $[d]$ and any player PLR_i with $L_i = 1$ chooses a message t_i uniformly at random from $(j + [a] \mod d)$, where $j = \mathfrak{c}(x)_i$ is the i'th letter in the codeword for x. For a more formal describtion of the protocol, see Figure 3.4.

Since each person corresponds to one use of the channel C we have $t_i = C(\mathfrak{c}(X)_i)$ and hence $T = C^n(\mathfrak{c}(X))$. By (3.12) this ensures that $\forall x : \Pr(G(T) \neq X | X = x) \leq \epsilon$, so we have reliable leakage.

To finish the proof that there exists a $(n, Rn, \text{Indep}_{b_l}(n), b_m, \epsilon)$-protocol, we need to check that the protocol also gives reasonable doubt. Given X the random variable (T_i, L_i), is independent from $T_1, L_1, \ldots, T_{i-1}, L_{i-1}, T_{i+1}, L_{i+1}, \ldots, T_n, L_n$. Using the computation from (3.11) we now get that $\Pr(L_i = 1 | T = t, X = x)$ is either 0 or

[5] In fact, it is exactly this value because the channel is symmetric.

Parameters:

n : number of players

b_l : probability of each player being leaker

$a < d$: two natural numbers

$\mathfrak{c}$: an error correcting code

Input distribution: $X, L_1, \ldots, L_n$ are independently distributed, X uniformly distributed on $\{0,1\}^{\lceil Rn \rceil}$ and for each i: $\Pr(L_i = 1) = b_l$. Each PLR_i learns L_i and if $L_i = 1$ she also learns X.

Protocol:

1. For i from 1 to n:
2. If $L_i = 1$ then $T_i \leftarrow \mathfrak{c}(x)_i + [a] \mod d$, if $L_i = 0$, then $T_i \leftarrow [d]$
3. Player i sends T_i

Figure 3.4: Protocol from the proof of Theorem 3.9.

$b'_m \leq b_m$ so we also achieve reasonable doubt.

□

Corollary 3.10. *The safe* $(\text{Indep}_{b_l}, b_m)$*-capacity is* $D\left(\frac{b_l}{b_m} \middle|\middle| \frac{b_l(1-b_m)}{b_m(1-b_l)}\right)$.

Proof. Follows from Proposition 3.8 and Theorem 3.9. □

Corollary 3.10 shows that if you want information about something that some proportion b_l of the population knows, but no one wants other people to think that they know it with probability greater than b_m, you can still get information about the subject, and at a rate of $D\left(\frac{b_l}{b_m} \middle|\middle| \frac{b_l(1-b_m)}{b_m(1-b_l)}\right)$ bits per person you ask. What if only l people in the world have the information? They are allowed to blend into a group of any size n, and observers will think that any person in the larger group is as likely as anyone else to have the information. Only the number of people with the information is known to everyone. Can they reveal an arbitrarily large amount of information by blending into a sufficiently large group?

If they are part of a group of $n \to \infty$ people, then each person in the larger group would have the information with probability $b_l = \frac{l}{n}$. If we forget that exactly l people know the information, and instead assume that all the L_is are independent with $\Pr(L_i = 1) = b_l$ they would be able to leak

$$\begin{aligned} nD\left(\frac{b_l}{b_m}\middle\|\frac{b_l(1-b_m)}{b_m(1-b_l)}\right) &= \frac{-b_l\log(1-b_m)+b_m\log(1-b_l)}{b_m}n \\ &= \frac{-\frac{l}{n}\log(1-b_m)+b_m\log(1-\frac{l}{n})}{b_m}n \\ &\to \left(\frac{\log(1-b_m)}{b_m}-\log(e)\right)l \end{aligned}$$

bits of information, where e is the base of the natural logarithm. We will see that even in the case where the number of leakers is known and constant, we can still get this rate. First we define the distribution of $(L_1, \ldots, L_n)$ that we get in this case.

Definition 3.7. Let $\text{Fixed}(l, n)$ be the random variable $(L_1, \ldots, L_n)$ that is distributed such that the set of leakers $\{\text{PLR}_i | L_i = 1\}$ is uniformly distributed over all subsets of $\{\text{PLR}_1, \ldots, \text{PLR}_n\}$ of size l.

A rate R is *safely/riskily* (Fixed, b_m)*-achievable* if for all $\epsilon > 0$ and all l_0, there exists a safe/risky $(n, lR, \text{Fixed}(l, n), b_m, \epsilon)$-protocol for some $l \geq l_0$ and some n.

The *safe/risky* (Fixed, b_m)*-capacity* is the supremum of all safely/riskily (Fixed, b_m)-achievable rates.

Notice that in this definition, the rate is measured in bits per leaker rather than bits per person communicating. That is because in this setup we assume that the number of people with the information is the bounded resource, and that they can find an arbitrarily large group of person to hide in.

Again, it turns out that the safe and the risky (Fixed, b_m)-capacities are actually the same, but for the proofs it will be convenient to have both definitions.

Proposition 3.11. *No rate* $R > \frac{-\log(1-b_m)}{b_m} - \log(e)$*, where* e *is the base of the natural logarithm is safely* (Fixed, b_m)*-achievable.*

Proof. This proof is very similar to the proof of Proposition 3.8.

Assume for contradiction that $R > \frac{-\log(1-b_m)}{b_m} - \log(e)$ is safely b_m-achievable. Consider a safe $(n, lR, \text{Fixed}(l, n), b_m, \epsilon)$-protocol π. We know from Theorem 3.5

that

$$I(X;T) \leq \frac{-\frac{l}{n}\log(1-b_m) + b_m \log\left(1-\frac{l}{n}\right)}{b_m} n \leq l\left(\frac{-\log(1-b_m)}{b_m} - \log(e)\right).$$

Here the second inequality follows from $\ln(1+x) \leq x$ or equivalently $\log(1-x) \leq \frac{-x}{\ln(2)} = -x\log(e)$. Let $\delta := R - \left(\frac{-\log(1-b_m)}{b_m} - \log(e)\right)$. Now

$$H(X|T) = H(X) - I(X;T) \geq l\left(R - \left(\frac{-\log(1-b_m)}{b_m} - \log(e)\right)\right) = l\delta.$$

By Fano's inequality we get that the probability of error, $P_e = \Pr(G(t) \neq X)$ is

$$P_e \geq \frac{l\delta - 1}{lR}.$$

Thus, if we choose l_0 sufficiently large and ϵ sufficiently small we cannot have $l \geq l_0$ and $P_e \leq \epsilon$, so there must be some value x where the probability of error $\Pr(G(T) \neq x | X = x)$ is greater than ϵ. □

Theorem 3.12. *Any rate* $R < \frac{-\log(1-b_m)}{b_m} - \log(e)$ *is riskily* (Fixed, b_m)*-achievable.*

Before we give a proof of this Theorem, notice that there are two reasons that the proof of a lower bound in the Indep_{b_l} model given in Theorem 3.9 does not translate directly to a lower bound for the Fixed model. First, in the protocol given in the proof of Theorem 3.9, there is a very small risk that only the leakers send messages consistent with being leakers. This is fine when the L_i's are independent: even if all but one player are revealed as non-leakers, that does not change the probability of the last player being a leaker. However, when the total number of leakers is known, this would completely reveal who the leakers are. This is why Theorem 3.12 is about risky achievability rather than safe achievability. The second problem is that the different usages of the channel are no longer independent as the number of leakers is constant. Intuitively, this should not be a problem, it should only make the channel more reliable. However, to show that this works, we would have to go through the proof of Shannon's Noisy-Channel Coding Theorem, and show that it still works. Instead, we will give a shorter but less natural proof, where we divide the players into two groups and use Theorem 3.9 on each group. As we are using Theorem 3.9

the proof in not constructive.

Proof. Let $R < \frac{-\log(1-b_m)}{b_m} - \log(e)$, then we can find rational $b_l > 0$ and rational ${b_m}' < b_m$ and a $\delta > 0$ such that $R + \delta < D\left(\frac{b_l}{b'_m}\middle|\middle|\frac{b_l(1-b'_m)}{b'_m(1-b_l)}\right)$. Let $n_0, \epsilon > 0$ be given. By Theorem 3.9 for any $\epsilon' > 0$ and any n'_0 there exists a safe $(n, n(R+\delta), \mathrm{Indep}_{b_l}(n), {b_m}', \epsilon')$-protocol where $n > n'_0$. Take such a protocol, where $\epsilon' > 0$ is sufficiently small and n'_0 is sufficiently large. As b_l, and hence the denominator of b_l, is fixed and n can be sufficiently large, we can increase n a little to ensure that $b_l n$ is an integer, while still keeping the rate at least R. Thus we can assume that we have a $(n, nR, \mathrm{Indep}_{b_l}(n), {b_m}', \epsilon')$-protocol, where $l := b_l n$ is an integer.

Now we will use this to make a risky $(2n, 2\lceil nR\rceil, \mathrm{Fixed}(2nb_l), b_m, \epsilon)$-protocol. For such a protocol, X should be uniformly distributed on $\{1, \ldots, 2^{2\lceil nR\rceil}\}$, but instead we can also think of X as a tuple (X_1, X_2) where the X_i are independent and each X_i is uniformly distributed on $\{1, \ldots, 2^{\lceil nR\rceil}\}$. Now we split the $2n$ players into two groups of n, and let the first group use the protocol from the proof of Theorem 3.9 to leak X_1, and the second group use the same protocol to leak X_2. We let Joe's guess of the value of X_1 be a function G_1 depending only of the transcript of the communication of the first group, and his guess of X_2 be a function G_2 depending only on the transcript of the second group. These functions are the same as G in the proof of Theorem 3.9. The resulting protocol is defined more formally in Figure 3.5.

The total number of leakers is $2nb_l$, but the number of leakers in each half varies. Let S_{Indep} denote random variable that gives the number of leakers among n people, when each is leaking with probability $b_l = \frac{l}{n}$, independently of each other. So S_{Indep} is binomially distributed, $S_{\mathrm{Indep}} \sim \mathrm{B}\left(n, \frac{l}{n}\right)$. Let $S_{\mathrm{Fixed},1}$ denote the number of leakers in the first group as chosen above. Now we have.

Lemma 3.13. *For each k,*

$$\frac{\Pr(S_{\mathrm{Fixed},1} = k)}{\Pr(S_{\mathrm{Indep}} = k)} \leq 2.$$

Proof. We have

$$\Pr(S_{\mathrm{Fixed},1} = k) = \frac{\binom{2l}{k}\binom{2n-2l}{n-k}}{\binom{2n}{n}}$$

Parameters:

n : half the number of players

b_l : probability of each player being leaker

b_m : threshold of reasonable doubt

R : rate

$\epsilon' > 0$: a sufficiently small number

π' : an $(n, \lceil nR \rceil, \text{Indep}_{b_l}, b_m, \epsilon')$-protocol

Input distribution: $X, L_1, \ldots, L_n$ are independently distributed, X uniformly distributed on $\{0,1\}^{2\lceil Rn \rceil}$ and for each i: $\Pr(L_i = 1) = b_l$. Each PLR_i learns L_i and if $L_i = 1$ she also learns X.

Protocol:

1. Parse $X \in \{0,1\}^{2\lceil Rn \rceil}$ as (X_1, X_2) with $X_i \in \{0,1\}^{\lceil Rn \rceil}$
2. Let $\text{PLR}_1, \ldots, \text{PLR}_n$ use π' to reveal X_1
3. Let $\text{PLR}_{n+1}, \ldots, \text{PLR}_{2n}$ use π' to reveal X_2

Figure 3.5: Protocol from the proof of Theorem 3.12. The complications in the proof is in choosing a sufficiently small ϵ' and showing that the resulting protocol has reasonable doubt and reliable leakage. The protocol is not constructive because we only have an existence proof for π_{Indep}.

and

$$\Pr(S_{\text{Indep}} = k) = \binom{n}{k} \left(\frac{l}{n}\right)^k \cdot \left(\frac{n-l}{n}\right)^{n-k}.$$

A simple computation shows

$$\frac{\Pr(S_{\text{Fixed},1} = k)\Pr(S_{\text{Indep}} = k+1)}{\Pr(S_{\text{Indep}} = k)\Pr(S_{\text{Fixed},1} = k+1)} = \frac{n-2l+k+1}{2l-k}\frac{l}{n-l},$$

which is strictly greater than 1 for $k \geq l$ and < 1 for $k < l$. Thus, for fixed n and l

the ratio $\frac{\Pr(S_{\text{Fixed},1}=k)}{\Pr(S_{\text{Indep}}=k)}$ is maximized by $k = l$. Using Stirling's formula,

$$1 \leq \frac{n!}{\sqrt{2\pi n}\left(\frac{n}{e}\right)^n} \leq \frac{e}{\sqrt{2\pi}}$$

we get

$$\begin{aligned}
\frac{\Pr(S_{\text{Fixed},1} = l)}{\Pr(S_{\text{Indep}} = l)} =& \frac{\binom{2l}{l}\binom{2n-2l}{n-l}n^n}{\binom{2n}{n}\binom{n}{l}l^l(n-l)^{n-l}} \\
=& \frac{(2l)!(2n-2l)!n^n}{l!l!(n-l)!(n-l)!} \cdot \frac{n!n!(n-l)!l!}{(2n)!n!l^l(n-l)^{n-l}} \\
=& \frac{(2l)!(2n-2l)!n^n n!}{l!(n-l)!(2n)!l^l(n-l)^{n-l}} \\
\leq& \left(\frac{e}{\sqrt{2\pi}}\right)^3 \cdot \frac{\sqrt{(2\pi)^3(2l)(2n-2l)(n)}}{\sqrt{(2\pi)^3(l)(2n)(n-l)}} \cdot \frac{e^{l+2n+n-l}}{e^{2l+2n-2l+n}} \\
& \cdot \frac{(2l)^{2l}(2n-2l)^{2n-2l}n^{2n}}{l^{2l}(n-l)^{2(n-l)}(2n)^{2n}} \\
=& \sqrt{2}\left(\frac{e}{\sqrt{2\pi}}\right)^3 \frac{2^{2l+2n-2l}l^{2l}(n-l)^{2n-2l}n^{2n}}{2^{2n}l^{2l}(n-l)^{2(n-l)}n^{2n}} \\
\leq& \sqrt{2}\left(\frac{e}{\sqrt{2\pi}}\right)^3 \\
<& 2.
\end{aligned}$$

□

Given that $S_{\text{Indep}} = k = S_{\text{Fixed},1}$, the distribution on $(L_1, \ldots, L_n)$ and transcript is the same in the protocol for Indep_{b_l} as it is for the first group in the above protocol. As Joe's guessing function is the same in the two cases, the probability of error given $S_{\text{Indep}} = k = S_{\text{Fixed},1}$ is the same in the two protocols. Let E_k denote the probability of error in the protocol for Indep_{b_l} given $S_{\text{Indep}} = k$, and let $E_{\text{Fixed},1}$ denote the

probability that Joe's guess of X_1 is wrong.

$$\begin{aligned} E_{\text{Fixed},1} =& \sum_{k=1}^{n} \Pr(S_{\text{Fixed},1} = k) E_k \\ \leq& \sum_{k=1}^{n} 2 \Pr(S_{\text{Indep}} = k) E_k \\ \leq& 2\epsilon'. \end{aligned}$$

By the same argument, the probability that Joe guess X_2 wrong is at most $2\epsilon'$, so the probability that he guess $X = (X_1, X_2)$ is at most $4\epsilon'$. By choosing a sufficiently low ϵ' this is less than $\epsilon/2$

To compute the posterior probability $\Pr(L_i = 1 | X = x, T = t)$ that PLR_i was leaking, we have to take the entire transcript from both groups into account. Given T and X, let K denote the set of players who sent a message consistent with knowing X, and let $|K|$ denote the cardinality of K. Let S be the set of the $2l$ leaking players, and let s be a set of $2l$ players. Now

$$\Pr(S = s | X = x, T = t) = \frac{\Pr(T = t | S = s, X = x) \Pr(S = s | X = x)}{P(T = t | X = x)}.$$

This is 0 if s contains players who send a message not consistent with having the information, and is constant for all other s. Thus, any two players who send a message consistent with having the information, are equally likely to have known X given T and X, so they will have $\Pr(L_i = 1 | T = t, X = x) = \frac{2l}{|K|}$. So to ensure that $\Pr(L_i = 1 | T = t, X = x) \leq b_m$ with high probability (for each x and random t) we only need to ensure that with high probability, $|K| \geq \frac{2l}{b_m}$. We see that

$|K| = 2l + \mathrm{B}\left(2n - 2l, \frac{b_l(1-b_m{}')}{b_m{}'(1-b_l)}\right)$, which has expectation

$$\begin{aligned}
2l + (2n-2l)\frac{b_l(1-b_m{}')}{b_m{}'(1-b_l)} =& 2nb_l + 2n(1-b_l)\frac{b_l(1-b_m{}')}{b_m{}'(1-b_l)} \\
=& 2n\left(b_l + \frac{b_l(1-b'_m)}{b'_m}\right) \\
=& 2n\frac{b_l}{b'_m} \\
=& \frac{2l}{b_m{}'} \\
=& \frac{2l}{b_m} + 2l\frac{b_m - b_m{}'}{b_m b_m{}'}.
\end{aligned}$$

The variance of the binomial distribution $B(n,p)$ is $np(1-p)$, so the variance of $|K|$ is $(2n-2l)b_l(1-b_l)$. Hence, for sufficiently high n (and thus l) Chebyshev's inequality, shows that $|K| \geq \frac{2l}{b_m}$ with probability at least $1-\epsilon/2$. Thus, for sufficiently large n'_0 and sufficiently low ϵ', the resulting protocol is a risky $(2n, 2\lceil nR\rceil, \mathrm{Fixed}(2nb), b_m, \epsilon)$-protocol. □

3.2.1 General $\mathfrak{L}$-structures

We have shown that the safe (Fixed, b_m)-capacity is at most $\frac{-\log(1-b_m)}{b_m} - \log(e)$ which is at most the risky (Fixed, b_m)-capacity. To finish the proof that they are both $\frac{-\log(1-b_m)}{b_m} - \log(e)$, we only need to show that the safe capacity is not smaller than the risky. Notice that the corresponding claim is not true if we are only interested in the mutual information between X and transcript T. Here there exists a collaborating cryptography protocol where with probability $1 - 10^{-100}$ we have, $\Pr(L_i = 1|T = t) < b_l + 10^{-100}$, and yet $I(X;T) \geq 10^{100}$. To do this we need to take X to have extremely high entropy, and with a probability 10^{-100} a leaking player will send X in a message, and otherwise just send some fixed message. On the other hand, if we require that $\Pr(L_i = 1|T = t) < b_l + 10^{-100}$ holds for all transcripts, then $I(X;T)$ has to be small compared to total number of players. The point of this section is to show that you cannot do something similar for reliable leakage. We will do so in a much more general setting than the rest of the thesis using an abstraction we call $\mathfrak{L}$-structures. These $\mathfrak{L}$-structures will not be used in the rest of

the thesis. Readers who are willing to accept that safe and risky capacities are the same can skip this section.

The concept of $\mathfrak{L}$-structures generalises Indep_{b_l} and Fixed. Remember that the difference between Indep_{b_l} and Fixed capacity is not only in the distributions on $(L_1, \dots L_n)$, but also in what we are trying to minimize the use of. In Indep_{b_l} we want to have as few people communicating as possible, while in Fixed we only care about the number of people who are leaking. Our general definition has to capture this difference as well.

Definition 3.8. An $\mathfrak{L}$-*structure* $(\mathfrak{L}, C)$ is a set $\mathfrak{L}$ of joint distributions of $(L_1, \dots, L_n)$ (where n does not need to be the same for each element), where each L_i is distributed on $\{0, 1\}$, together with a *cost function* $C : \mathfrak{L} \to \mathbb{R}_{\geq 0}$.

Indep_{b_l} is the $\mathfrak{L}$-structure $(\mathfrak{L}_{\text{Indep}_{b_l}}, C_{\#})$, where $\mathfrak{L}_{\text{Indep}_{b_l}}$ is the set of distributions on $(L_1, \dots, L_n)$ (over $n \in \mathbb{N}$) where for all i, $\Pr(L_i = 1) = b_l$ and the L_i are independent, and $C_{\#}$ is the function that sends a distribution on $(L_1, \dots, L_n)$ to n.

Fixed is the $\mathfrak{L}$-structure $(\mathfrak{L}_{\text{Fixed}}, C_{\text{Fixed}})$ of distributions on $(L_1, \dots, L_n)$ such that for some number l the set $\{i | L_i = 1\}$ is uniformly distributed over all subsets of $\{1, \dots, n\}$ of size l, and C_{Fixed} sends such a distribution on $(L_1, \dots, L_n)$ to this number l.

For an $\mathfrak{L}$-structure $(\mathfrak{L}, C)$ a rate R is safely/riskily $(\mathfrak{L}, C, b_m)$-achievable if for all $\epsilon > 0$ and all $h_0 \geq 0$ there exists a safe/riskily (n, h, L, b_m, ϵ)-protocol with $h \geq h_0, h \geq C(L)R$ and $L \in \mathfrak{L}$.

The safe/risky $(\mathfrak{L}, C, b_m)$-capacity is the supremum of all safely/riskily $(\mathfrak{L}, C, b_m)$-achievable rates.

We see that Definition 3.8 agrees with Definition 3.6 and Definition 3.7,[6] and is much more general.

Proposition 3.14. *Let $(\mathfrak{L}, C)$ be an $\mathfrak{L}$-structure. The safe $(\mathfrak{L}, C, b_m)$-capacity and the risky $(\mathfrak{L}, C, b_m)$-capacity are non-decreasing functions of b_m.*

[6] In Definition 3.6 and Definition 3.7 the lower bounds which prevent solutions with small number of people and small entropy are given as $\forall n_0 \exists n : n \geq n_0$ respectively $\forall l_0 \exists l : l \geq l_0$ instead of $\forall h_0 \exists h : nR = h \geq h_0$ respectively $\forall h_0 \exists h : lR = h \geq h_0$ as in Definition 3.8. However, for fixed R, these requirements are equivalent.

Proof. Let ${b_m}' > b_m$. It is clear that any safe/risky (n, h, L, b_m, ϵ)-protocol is a safe/risky $(n, h, L, {b_m}', \epsilon)$-protocol, so any safe/riskily $(\mathfrak{L}, C, b_m)$-achievable rate is a safe/riskily $(\mathfrak{L}, C, {b_m}')$-achievable rate. □

Proposition 3.15. *Let $(\mathfrak{L}, C)$ be an $\mathfrak{L}$-structure. The safe $(\mathfrak{L}, C, b_m)$-capacity is at most the risky $(\mathfrak{L}, C, b_m)$-capacity.*

Proof. Any safe (n, h, L, b_m, ϵ)-protocol is a risky (n, h, L, b_m, ϵ)-protocol, so any safely $(\mathfrak{L}, C, b_m)$-achievable rate is riskily $(\mathfrak{L}, C, b_m)$-achievable. □

The opposite inequality almost holds. Before we show that, we need a lemma. We could easily get the lemma below from results in Chapter 6, but to avoid references to later chapters, we state and prove the result in this chapter. The proof of the lemma is constructive.

Lemma 3.16. *For any risky (n, h, L, b_m, ϵ)-protocol π, there is a risky (n, h, L, b_m, ϵ)-protocol π' where each message is either 0 or 1, and given previous transcript and given that the person sending the message is not leaking, there is at least probability $1/3$ of the message being 0 and at least $1/3$ of it being 1.*

Proof. To restrict to $\{0, 1\}$ we simply send one bit at a time, so now we only have to ensure that the probability of a message sent by a non-leaker being 0 is always in $[\frac{1}{3}, \frac{2}{3}]$. If the next message is 0 with probability $p < 1/3$, given that the sender is not leaking we modify the protocol (the case where $p > 2/3$ is similar). First, the player PLR_i sending the message decides if she would have sent 0 or 1 in the old protocol π. Call this message a. If $a = 0$ she chooses a number r in the interval $(0, p)$ uniformly at random, if $a = 1$ she chooses a number r in $(p, 1)$ uniformly at random. She then sends the bits of r one bit at a time until

- She says 1, or
- Given transcript until now, there is probability at least $\frac{1}{3}$ that $a = 0$

In the first case we know that $a = 1$, and we can go to the next round of π as if she had just sent the message 1 in π. Each time PLR_i says 0, it doubles the probability that $a = 0$ given the transcript, so if we are in the second case (and was not before

Parameters:

n : number of players

h : number of bits being leaked

L : distribution of leakers

b_m : threshold of reasonable doubt

ϵ : acceptable probability of error

π : a risky (n, h, L, b_m, ϵ)-protocol

Input distribution: X and L are independently distributed, X uniformly distributed on $\{0,1\}^h$, distribution of $L = (L_1, \ldots, L_n)$ is a parameter. Each PLR_i learns L_i and if $L_i = 1$ she also learns X.

Protocol:

1. $T := \lambda$
2. While T is not a complete transcript of π
3. Let i be the player to send the next bit in π when the transcript is T
4. If there is only one possible bit a that player i can send, set $T = T \circ a$, else
5. Let $p \in (0,1)$ be the probability that the next bit would be 1 given $L_i = 0$
6. Player i choose the next bit a to send according to π
7. If $a = 0$ player i chooses $r \leftarrow (0, p)$ otherwise she chooses $r \leftarrow (p, 1)$
8. While no message has been added to T
9. If $p \in [1/3, 2/3]$ then player i sends a, $T := T \circ a$, otherwise
10. Player i sends first bit r_1 of r
11. If $r_1 < p < 1/2$ or $r_1 > p > 1/2$ then $T := T \circ r_1$ otherwise
12. $p := \text{frac}(2p)$, $r := \text{frac}(2r)$ where $\text{frac}(x) = x - \lfloor x \rfloor$

Figure 3.6: Protocol from the proof of Lemma 3.16.

the last message), $\Pr(a = 0|T) < \frac{2}{3}$. In this case she will simply reveal a in the next message. For a more formal description of the protocol, see Figure 3.6.

Instead of choosing a real number uniformly from $(0, p)$ or $(p, 1)$, which would require access to randomness with infinite entropy, PLR_i can just in each step compute the probabilities of sending 0 or 1 given that she had chosen such a number. Thus, if for every probability p' every player has access to a coin that ends head up with probability p', they only need a finite number of coin flips to follow the above protocol. □

The following lemma "almost" says that the safe $(\mathfrak{L}, C, b_m)$-capacity is the same as the risky $(\mathfrak{L}, C, b_m)$-capacity. The proof is constructive, so if you can find good risky protocols, you can also find good safe protocols.

Lemma 3.17. *Let ${b_m}' > b_m$. The safe $(\mathfrak{L}, C, {b_m}')$-capacity is at least the same as the risky $(\mathfrak{L}, C, b_m)$-capacity.*

Proof. To show this, it is enough to show that if R is a riskily $(\mathfrak{L}, C, b_m)$-achievable rate, then R is safely $(\mathfrak{L}, C, {b_m}')$-achievable. Let R be a riskily $(\mathfrak{L}, C, b_m)$-achievable rate, and let $\epsilon' > 0$ and h'_0 be given. We want to show that there exists a safe $(n', h', L, {b_m}', \epsilon')$-protocol with $h' \geq h'_0$, $L \in \mathfrak{L}$ and $h' \geq C(L)R$.

As R is riskily $(\mathfrak{L}, C, b_m)$-achievable, there exists a risky (n, h, L, b_m, ϵ)-protocol for any $\epsilon > 0$ and some $L \in \mathfrak{L}$, $h \geq h'_0$, $h \geq C(L)R$ and n. Let π be such a protocol, where ϵ is a small number to be specified later.

We want to modify π to make it a safe protocol π'. First, by Lemma 3.16 we can assume that all messages send in π are in $\{0, 1\}$ and given that the sender is not leaking, it has probability at least $1/3$ of being 0 and at least probability $1/3$ of being 1.

To ensure that for no transcript t and player PLR_i we have $\Pr(L_i = 1|X = x, T = t) > {b_m}'$, we modify the protocol, such that everyone starts to pretend ignorance if the next message could result in $\Pr(L_i = 1|X = x, T^{k+1} = t^{k+1}) > {b_m}'$. Formally, we define a protocol π' that starts of as π but if at some point the transcript is t^k and for some i and $b \in \{0, 1\}$ we have $\Pr(L_i = 1|T^{k+1} = t^k \circ b, X = x) > {b_m}'$ all the players *pretends ignorance*, that is for the rest of the protocol they send messages as if they did not have the information and were following π. Notice that only the players who knows the information x can decide if they should pretend ignorance,

but this is not a problem as the players who do not have the information, are already sending messages as if they did not have the information. The protocol is also given in Figure 3.7.

Parameters:

n : number of players

h : number of bits being leaked

L : distribution of leakers

$b'_m > b_m$: thresholds of reasonable doubt for the safe respecitively the risky protocol

ϵ : acceptable probability of error

π : a risky (n, h, L, b_m, ϵ)-protocol where the players send one bit at a time and the probability a non-leaker sending 1 is always in $[1/3, 2/3]$

Input distribution: X and L are independently distributed, X uniformly distributed on $\{0,1\}^h$, distribution of $L = (L_1, \ldots, L_n)$ is a parameter. Each PLR_i learns L_i and if $L_i = 1$ she also learns X.

Protocol:

1. While the players have not reach the end of π and the players are not pretending ignorance:
2. Each leaker decides if for some i and some possible next messages t_{k+1} we have $\Pr(L_i = 1 | X = x, T^{k+1} = t^{k+1}) > {b_m}'$. If yes they start to *pretend ignorance*
3. If the leaker are not pretending ignorance, the next player in π sends a message as if following π.
4. If the leakers are pretending ignorance, everyone follows π but as if they where non-leakers.

Figure 3.7: Protocol from the proof of Lemma 3.17.

First we want to show that π' is ${b_m}'$-safe. As long as they do not pretend ignorance we know that $\Pr(L_i = 1 | T^k = t^k, X = x) \leq {b_m}'$ for the partial transcript t^k and all i. If at some point they start to pretend ignorance, we have $\Pr(L_i = 1 | T^k =$

$t^k, X = x) \leq b_m{}'$ before they start, and all messages will be chosen as if no one had the information. Eve, who knows X, can compute $\Pr(L_i = 1|T^{k+1} = t^k \circ b, X = x) > b_m{}'$ for each i and b, so she knows if everyone is pretending ignorance. Thus, Eve does not learn anything about L from listening to the rest of the communication, so we will still have $\Pr(L_i = 1|T = t, X = x) \leq b_m{}'$ when π' terminates.

Fix $x \in \mathcal{X}$. We want to compute the probability that they pretend ignorance given $X = x$. Let $E_{par,>b'_m}$ denote[7] the event that for transcript T from the execution of π, we can find some k and some i such that we have $\Pr(L_i = 1|T^k = t^k, X = x) > b_m{}'$. That is, at some point in the execution of π, an observer would say that PLR_i was leaking with probability greater than $b_m{}'$. Let $E_{tot,>b_m}$ be that event that for the complete transcript there is some i such that $\Pr(L_i = 1|T = t, X = x) > b_m$. For each transcript t where $\Pr(L_i = 1|T^k = t^k, X = x) > b_m{}'$ for some k, i, we consider that smallest k such that $\Pr(L_i = 1|T^k = t^k, X = x) > b_m{}'$ happens for some i. For this fixed t^k let $T^{-[k]}$ denote the random variable that is distributed as the rest of the transcript given that the transcript starts with t^k and $X = x$. Let S_{t^k} denote the random variable

$$S_{t^k} = \Pr(L_i = 1|T = t^k \circ T^{-[k]}, X = x).$$

That is, S_{t^k} is a function of $T^{-[k]}$. We see that S_{t^k} takes values in $[0, 1]$ and $\mathbb{E}S_{t^k} = \Pr(L_i = 1|T^k = t^k, X = x) > b'_m$ so by Markov's inequality on $1 - S_{t^k}$ we get

$$\Pr(1 - S_{t^k} \geq 1 - b_m|X = x) \leq \frac{\mathbb{E}(1 - S_{t^k})}{1 - b_m} < \frac{1 - b_m{}'}{1 - b_m}.$$

Thus, given that $E_{par,>b_m{}'}$ happens, $E_{tot,>b_m}$ will happen with probability at least $1 - \frac{1-b'_m}{1-b_m} = \frac{b_m{}'-b_m}{1-b_m} > 0$. So $\frac{b_m{}'-b_m}{1-b_m} \Pr(E_{par,>b_m{}'}|X = x) \leq \Pr(E_{tot,>b_m}|X = x) \leq \epsilon$, where the last inequality follows from the assumption about π.

Let E_{ig} be the event that in the evaluation of π' the players pretends ignorance. The players only pretends ignorance if they are one message away from making $E_{par,>b_m{}'}$ happen. We assumed that in π each possible message get sent with probability at least $1/3$ if the sender is not leaking. As there is probability at least $1 - b_m{}'$ that he is not leaking, each possible message gets sent with probability at least $\frac{1-b_m{}'}{3}$

[7]Here "par" is short for partial transcript.

so $\frac{1-b_m{}'}{3}\Pr(E_{ig}|X=x) \leq \Pr(E_{par,>b_m{}'}|X=x)$. Thus,

$$\Pr(E_{ig}|X=x) \leq \frac{3}{1-b_m{}'}\Pr(E_{par,>b_m{}'}|X=x) \leq 3\epsilon\frac{(1-b_m)}{(b_m{}'-b_m)(1-b_m{}')}.$$

Let T' denote the random variable you get from running π' and T the random variable you get from running π, with a joint distribution of (X, L, T, T') such that $(X, L, T) = (X, L, T')$ unless the players pretends ignorance. We need to show that there is a decoding function G' from the set of complete transcripts to possible values of X such that for each x, $\Pr(G'(T') = x|X = x) \geq 1 - \epsilon'$. From the assumptions about π we know that there is a function G from the set of possible transcripts to the support of X such that for each x, $\Pr(G(T) = x|X = x) \geq 1 - \epsilon$. We know that in π' and for fixed x, the players only pretends ignorance with probability at most $\frac{3\epsilon(1-b_m)}{(b_m{}'-b_m)(1-b_m{}')}$, so by setting $G' = G$ we get $\Pr(G'(T') = x|X = x) \geq 1 - \epsilon - \frac{3\epsilon(1-b_m)}{(b_m{}'-b_m)(1-b_m{}')}$. For sufficiently small ϵ (depending only on ϵ', b_m and $b_m{}'$) this is more than $1 - \epsilon'$ and we are done. □

If we add a continuity assumption, we get that the safe and the risky b_m capacity are the same.

Corollary 3.18. *Let $(\mathfrak{L}, C)$ be a $\mathcal{L}$-structure. If the safe $(\mathfrak{L}, C, b_m)$-capacity as a function of b_m is right-continuous at b_{m_0}, or if the risky $(\mathfrak{L}, C, b_m)$-capacity as a function of b_m is left-continuous at b_{m_0} then the safe $(\mathfrak{L}, C, b_{m_0})$-capacity and the risky $(\mathfrak{L}, C, b_{m_0})$-capacity are the same.*

Proof. Assume that the safe $(\mathfrak{L}, C, b_m)$-capacity as a function of b_m a right-continuous at b_{m_0}. Then Lemma 3.17 shows that the risky $(\mathfrak{L}, C, b_m)$-capacity is at most the safe $(\mathfrak{L}, C, b_m{}')$-capacity for all $b_m{}' > b_m$. By continuity assumption, this gives us that the risky $(\mathfrak{L}, C, b_m)$-capacity is at most the safe $(\mathfrak{L}, C, b_m)$-capacity. Proposition 3.15 shows the opposite inequality. The proof of the second part of the corollary is similar. □

Corollary 3.19. *Let $(\mathfrak{L}, C)$ be a $\mathcal{L}$-structure. The safe $(\mathfrak{L}, C, b_m)$-capacity and the risky $(\mathfrak{L}, C, b_m)$-capacity are the same for all but at most countably many values $b_m \in (0, 1)$.*

Proof. By Proposition 3.14, the safe $(\mathfrak{L}, C, b_m)$-capacity is a monotone function, so it is continuous in all but countably many points. Now Corollary 3.18 implies that it is the same as the risky $(\mathfrak{L}, C, b_m)$-capacity in all but countably many points. □

As promised, we can now show that the safe and the risky $(\text{Indep}_{b_l}, b_m)$-capacities are the same.

Corollary 3.20. *The safe* $(\text{Indep}_{b_l}, b_m)$*-capacity and the risky* $(\text{Indep}_{b_l}, b_m)$*-capacity are the same for all* $b_m \in (0, 1)$*.*

Proof. We know from Corollary 3.10 that the safe $(\text{Indep}_{b_l}, b_m)$-capacity is a continuous function of b_m. Now Corollary 3.18 implies that it is the same as the risky $(\text{Indep}_{b_l}, b_m)$-capacity. □

Corollary 3.21. *Let* $c \in (0, 1)$*. The safe* (Fixed, b_m)*-capacity and the risky* (Fixed, b_m)*-capacity are both* $\frac{-\log(1-b_m)}{b_m} - \log(e)$*.*

Proof. We know from Proposition 3.11 that the safe (Fixed, b_m)-capacity is at most $\frac{-\log(1-b_m)}{b_m} - \log(e)$, we know from Theorem 3.12 that the risky $(\text{Fixed}, b_m{}')$-capacity is at least $\frac{-\log(1-b_m)}{b_m} - \log(e)$, and from Corollary 3.19 that they are the same except on at most countably many values. Thus, they must both be $\frac{-\log(1-b_m)}{b_m} - \log(e)$ on all but countably many values. We know from 3.14 that both are monotone, so they must both be $\frac{-\log(1-b_m)}{b_m} - \log(e)$ without exceptions. □

3.3 Adaptive cryptogenographic protocols

Until now we have assumed that each player is either leaker or not a leaker. In this section we study some adaptive models where people start as non-leakers, but might start to leak at some point. Once a person is a leaker, that person will always be a leaker.

An *adaptive cryptogenography protocol* π is defined as follows: for each partial transcript t^k, each vector $L_{\cdot,k-1} = (L_{1,k-1}, \ldots, L_{n,k-1})$ describing the set of leaker when the k'th message was sent, and each secret x, π gives a probability distribution over vectors $L_{\cdot,k} \geq L_{\cdot,k-1}$ describing the set of leakers after the k'th message. We assume that no one is leaking from the beginning, that is, $L_{\cdot,0} = (0, \ldots, 0)$. Like a collaborative cryptogenography protocol, π specifies for each partial transcript t^k

- Should the communication stop or continue, and if it should continue,
- Who is next to send a message, say PLR_i, and
- A distribution $p_?$ and a set of distributions, $\{p_x\}_{x\in\mathcal{X}}$ (the distributions $p_?$ and $\{p_x\}_{x\in\mathcal{X}}$ depend on π and t^k). Now PLR_i should choose a message using $p_?$, if $L_{i,k} = 0$ and choose a message using p_x if $L_{i,k} = 1$ and $X = x$.

Here it is natural to put some restriction on how many leakers there can be, and on what can influence whether a person becomes a leaker. We suggest two ways of putting a limitation on the total number of leakers, and three different rules for what can affect the probability that a person becomes a leaker, giving a total of six different combinations. In this section we will find the capacities for two of them.

The two ways of restricting the total number of leakers are called "b_l-threshold" and "b_l-dormant". The *b_l-threshold* restriction requires that the expected number of leakers at the end of the protocol is at most $b_l n$. This is a slightly unnatural requirement, but is the easiest to analyse. A more natural requirement is the *b_l-dormant* restriction, which say that at the beginning each player is chosen to be a "dormant" leaker with probability b_l, and only dormant leakers can become leakers. We can think of dormant leakers as people with the personality or the capacity to become leakers. Clearly, the b_l-dormant model is more restrictive than the b_l-threshold model, but on the other hand, the leakers can do more in the b_l-dormant model than in Indep_{b_l} in the static model: If you take $b_l = b_m$, the $(\text{Indep}_{b_l}, b_m)$-capacity is 0, but in the b_l-dormant model you can leak information, for example by letting each dormant leaker become leaker with probability 1/2, and use a protocol for $\text{Indep}_{\frac{b_m}{2}}$.

The three models for how a player become a leaker are called "centrally organised", "informed choice" and "uninformed choice". In the *centrally organised* model we assume that there is someone organising who become leakers. We assume this person has all the relevant information, t^k, $L_{\cdot,k-1}$ and x, and hence there is no restriction on the distribution of $L_{\cdot,k}$ except $L_{\cdot,k} \geq L_{\cdot,k-1}$, that is leakers cannot turn into non-leakers. This model is probably unrealistic, but good for showing upper bounds. In the *informed choice* we assume that even the non-leakers know x, and they may use this when deciding whether to become a leaker, but each player makes

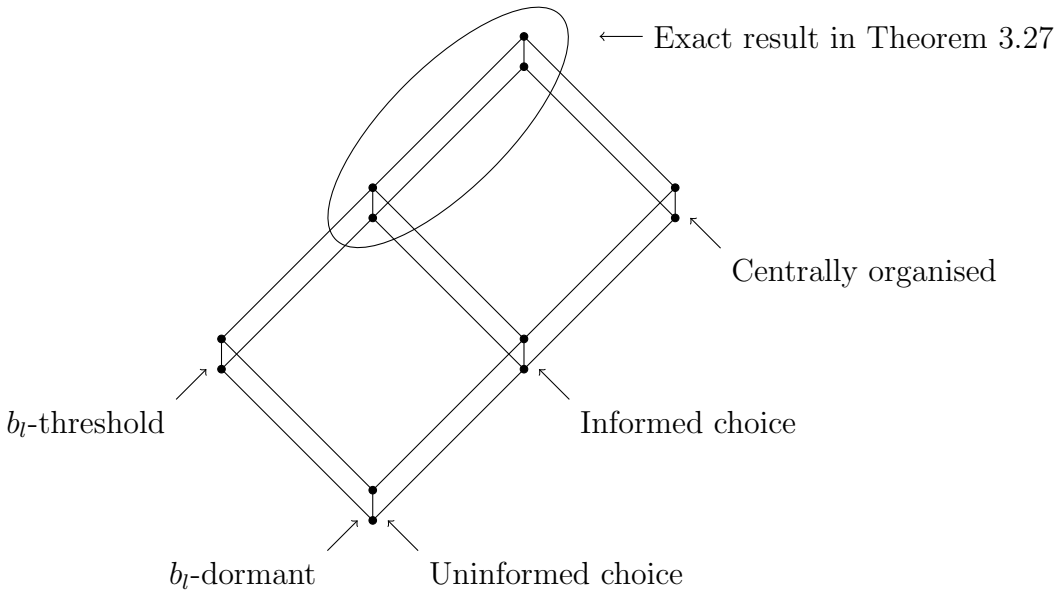

Figure 3.8: Diagram showing the different models for adaptive leakage. There are two different ways to restrict the total number of leakers, three different models for how a player can becomes a leaker, and for each of the resulting 6 models we can consider both the safe and the risky capacity. In each model the top dot represents the risky capacity and the bottom dot represents the safe capacity. The lines show trivial inequalities between the capacities, where the capacity represented by a higher point on the line is an upper bound for the capacity represented by the lower point. In Theorem 3.27 we see that all the capacities represented by points in the ellipse are the same.

the decision on whether to become a leaker on her own. That is, the distribution of $L_{i,k}$ depends on $L_{i,k-1}, x$ and t^k but given x and t^k it is independent from all the other $L_{j,k}$'s and $L_{j,k-1}$'s. Finally, there is the *uninformed choice* model where the players only learn x when they decide to become leakers. Here $L_{i,k}$ only depends on $L_{i,k-1}$ and t^k.

These give 6 different models that we call *adaptive models*. We use M_{b_l} to denote an adaptive model with parameter b_l. While "b_l-dormant informed choice" and "b_l-dormant uninformed choice" are probably the most realistic models, "b_l-threshold centrally organised" and "b_l-threshold informed choice" seem to be the easiest to analyse.

Definition 3.9. We let $\text{susp}_{i,k}$ denote the suspicion that $L_{i,k} = 1$, e.g.

$$\text{susp}_{i,k}(X, T^k) = -\sum_{x,t^k} \Pr(X = x, T^k = t^k) \log(\Pr(L_{i,k} = 0 | X = x, T^k, t^k)).$$

We define $L_i = L_{i,\text{length}(\pi)}$ and similarly $\text{susp}_i = \text{susp}_{i,\text{length}(\pi)}$.

Definition 3.10. A *risky* $(n, h, M_{b_l}, b_m, \epsilon)$*-protocol* is an adaptive cryptogenography protocol satisfying the requirements of model M_{b_l} together with a function G from the set of possible transcripts to $\mathcal{X} = \{1, \ldots, 2^{\lceil h \rceil}\}$ such that for any $x \in \mathcal{X}$, there is probability at least $1 - \epsilon$ that a random transcript t distributed as $T|_{X=x}$ satisfies

Reasonable doubt: $\forall i : \Pr(L_i = 1 | T = t, X = x) \leq c$, and

Reliable leakage: $G(t) = x$

A *safe* $(n, h, M_{b_l}, b_m, \epsilon)$*-protocol* is a risky $(n, h, M_{b_l}, b_m, \epsilon)$-protocol where $\Pr(L_i = 1 | T = t, X = x) \leq b_m$ for all i, t, x with $\Pr(T = t, X = x) > 0$.

A rate R is *safely/riskily* b_m*-achievable for* M_{b_l} if for all $\epsilon > 0$ and all n_0, there exists a safe/risky $(n, nR, M_{b_l}, b_m, \epsilon)$-protocol for some $n \geq n_0$.

The *safe/risky* b_m*-capacity for* M_{b_l} is the supremum of all safely/riskily b_m-achievable rates for M_{b_l}.

Theorem 3.22. *For $b_l \leq b_m$ and any model M_{b_l} the safe b_m-capacity for M_{b_l} is at most* $\frac{-b_l \log(1-b_m)}{b_m} - b_l \log(e)$.

Proof. As "b_l-threshold centrally organised" is the least restrictive model, we can assume that M_{b_l} is this model. Let π be an adaptive cryptogenography protocol for M_{b_l}. The function $\frac{-b_l \log(1-b_m)}{b_m} - b_l \log(e)$ is increasing in b_l, so we can assume that the expected number of leakers at the end of π is exactly $b_l n$, as the protocol would otherwise be an $M_{b'_l}$-protocol for some $b'_l < b_l$.

As when we proved Theorem 3.4 we can assume that the next player to send a message does not depend on the previous transcript t^{k-1}, but only on the number of messages sent. If PLR_j sends the k'th message Corollary 3.2 tells us that

$$I(X; T_k | T^{k-1}) \leq \text{susp}_{j,k-1}(X, T^k) - \text{susp}_{j,k-1}(X, T^{k-1}), \tag{3.13}$$

and Proposition 3.3 tells us that for $i \neq j$

$$\mathrm{susp}_{i,k-1}(X, T^k) \geq \mathrm{susp}_{i,k-1}(X, T^{k-1}). \tag{3.14}$$

If we move right hand side of (3.14) to the other side, sum over all $i \neq j$ and add the result to (3.13) we get

$$I(X; T_k | T^{k-1}) \leq \sum_{i=1}^{n} \left(\mathrm{susp}_{i,k-1}(X, T^k) - \mathrm{susp}_{i,k-1}(X, T^{k-1}) \right) \tag{3.15}$$

In this adaptive model we also need to consider how it affects the suspicion that players can turn into leakers. By a small abuse of notation we let c_{i,k',x,t^k} denote $\Pr(L_{i,k'} = 1 | X = x, T^k = t^k)$, and $c_{i,k}$ denote $\Pr(L_{i,k} = 1)$. As $\frac{d}{dx} \log(x) \geq \log(e)$ for $x \leq 1$ and $L_{i,k} \geq L_{i,k-1}$ we have for all i and k,

$$\begin{aligned}
&\mathrm{susp}_{i,k}(X, T^k) - \mathrm{susp}_{i,k-1}(X, T^k) \\
&\qquad = -\sum_{x,t^k} \Pr(X = x, T^k = t^k) \left(\log(1 - c_{i,k,x,t^k}) - \log(1 - c_{i,k-1,x,t^k}) \right) \\
&\qquad \geq -\sum_{x,t^k} \Pr(X = x, T^k = t^k) \log(e) \left((1 - c_{i,k,x,t^k}) - (1 - c_{i,k-1,x,t^k}) \right) \\
&\qquad = \sum_{x,t^k} \Pr(X = x, T^k = t^k) \log(e) \left(c_{i,k,x,t^k} - c_{i,k-1,x,t^k} \right) \\
&\qquad = (c_{i,k} - c_{i,k-1}) \log(e).
\end{aligned} \tag{3.16}$$

If we move right hand side of (3.16) to the other side, sum over all i and add the result to (3.15) we get

$$I(X; T_k | T^{k-1}) \leq \sum_{i=1}^{n} \left(\mathrm{susp}_{i,k}(X, T^k) - \mathrm{susp}_{i,k-1}(X, T^{k-1}) - \log(e) \left(c_{i,k} - c_{i,k-1} \right) \right).$$

Summing this over all rounds gives us

$$\begin{aligned} I(X;T) \leq & \sum_{i=1}^{n} \left(\mathrm{susp}_i(X,T) - \mathrm{susp}_{i,0}(X) - \log(e)\left(\Pr(L_i=1) - \Pr(L_{i,0}=1)\right)\right) \\ = & \sum_{i=1}^{n} \left(\mathrm{susp}_i(X,T) - \log(e)\Pr(L_i=1)\right). \end{aligned}$$

We have $\Pr(L_i = 1|X = x, T = t) \leq b_m$ so by the same argument as in Theorem 3.5 we have

$$\mathrm{susp}_i(X = x, T = t) \leq \frac{-\log(1-b_m)}{b_m} \Pr(L_i = 1|X = x, T = t).$$

Now we make a computation very similar to the one in Theorem 3.5.

$$\begin{aligned} \sum_i \mathrm{susp}_i(X,T) = & \sum_{i,x,t} \Pr(X=x,T=t)\,\mathrm{susp}_i(X=x,T=t) \\ \leq & \sum_{i,x,t} \Pr(X=x,T=t)\frac{-\log(1-b_m)}{b_m}\Pr(L_i=1|X=x,T=t) \\ = & \sum_{i,x,t} \frac{-\log(1-b_m)}{b_m}\Pr(L_i=1,X=x,T=t) \\ = & \sum_{i} \frac{-\log(1-b_m)}{b_m}\Pr(L_i=1) \\ \leq & n\frac{-b_l\log(1-b_m)}{b_m}. \end{aligned}$$

Here the last inequality follows from the assumption that $\mathbb{E}\sum_i L_i = nb_l$ By applying Fano's inequality as in the proof of Proposition 3.8, it follows that the safe b_m-capacity for M_{b_l} is at most $\frac{-b_l\log(1-b_m)}{b_m} - b_l\log(e)$. □

In the next two propositions we show that this upper bound also holds for risky protocols. It is proved in a constructive way, so if you can find good risky protocols you can also find good safe protocols.

Proposition 3.23. *Let $b_m{}' > b_m$. The safe $b_m{}'$-capacity for "b_l-threshold centrally organised" is at least the same as the risky b_m-capacity for "b_l-threshold centrally organised".*

Proof. Let M_{b_l} be the model "b_l-threshold centrally organised". We use that same strategy as in the proof of Lemma 3.17. Assume that R is riskily b_m-achievable for M_{b_l}. To show the statement, it is enough to show that R is then safely b'_m-achievable for M_{b_l}. Let $\epsilon' > 0$ and n'_0 be given. We need to show that there exists a safe $(n', Rn', M_{b_l}, {b_m}', \epsilon')$-protocol, where $n' \geq n'_0$. As R is riskily b_m-achievable for M_{b_l}, there exists a risky $(n, nR, M_{b_l}, b_m, \epsilon)$ protocol for any $\epsilon > 0$ and where $n \geq n'_0$. Let π be such a protocol for a small ϵ to be specified later. We will modify π to get a safe protocol π'. By Lemma 3.16 we can assume that all messages sent in π are in $\{0, 1\}$ and given that the sender is not leaking, it has probability at least $1/3$ of being 0 and at least probability $1/3$ of being 1.

As in the proof of Lemma 3.17, modify the π by forcing the players to pretend ignorance in some situations. To *pretend ignorance* means that all the players send messages as if they were non-leakers. If we want all the players to pretend ignorance from a certain point, it is important that all the leakers can decide whether they should pretend ignorance, but the non-leaker does not have to know, as they are already sending messages as if they were non-leaker. If Eve is able to decide if all the players pretend ignorance, it implies that once the players pretend ignorance, she does not get any further information.

We require the players to pretend ignorance from round $k+1$ and onwards, if the current transcript is t^k and $\Pr(L_{i,k} = 1 | T^k = t^k, X = x) \leq {b_m}'$ but $\Pr(L_{i,k} = 1 | T^{k+1} = t^k \circ t_{k+1}, X = x) > {b_m}'$ for some player i and some $t_{k+1} \in \{0, 1\}$. The leakers can all compute $\Pr(L_{i,k} = 1 | T^{k+1} = t^k \circ t_{k+1}, X = x)$, so the leakers know if they should start to pretend ignorance. Eve can also compute $\Pr(L_{i,k} = 1 | T^{k+1} = t^k \circ t_{k+1}, X = x)$, so once the players pretend ignorance she does not learn any further information. We also modify π such that when the players starts to pretend ignorance, no one turn into leakers. We can do this, because the model is centrally organised, so the probability of becoming a leaker can depend on X. Furthermore, we modify π such that if the partial transcript t^k satisfy $\Pr(L_{i,k-1} = 1 | T^k = t^k, X = x) \leq {b_m}'$ but $\Pr(L_{i,k} = 1 | T^k = t^k, X = x) > {b_m}'$ for some i, then no one becomes leakers at round k or any later rounds, and everyone starts to pretend ignorance. By induction on k, these modifications ensure that $\Pr(L_i = 1 | T = t, X = x) \leq {b_m}'$. The protocol is also defined in Figure 3.9.

Next we need to define the function G' that takes transcripts of π' to guesses

Parameters:

n : number of players

h : number of bits being leaked

L : distribution of leakers

$b'_m > b_m$: thresholds of reasonable doubt for the safe respectively the risky protocol

ϵ : acceptable probability of error

π : a risky $(n, h, M_{b_L}, b_m, \epsilon)$-protocol where the players send one bit at a time and the probability a non-leaker sending 1 is always in $[1/3, 2/3]$

Input distribution: X uniformly distributed on $\{0,1\}^h$, $L_{i,0} = 0$ for all i. If $L_{i,k} = 1$ then PLR_i learns X after message k.
Protocol:

1. While the players have not reach the end of π and the leakers are not pretending ignorance:
2. Each leaker decides if for some i and some possible next messages t_{k+1} we have $\Pr(L_{i,k} = 1 | X = x, T^{k+1} = t^{k+1}) > {b_m}'$. If yes they start to *pretend ignorance*
3. If the leaker are not pretending ignorance, the next player in π sends a message as if following π.
4. If the leakers are pretending ignorance, everyone follows π but as if they where non-leakers and no more players become leakers.

Figure 3.9: Protocol from the proof of Proposition 3.23.

of the value X. This is simply defined to be the same as the function G for π. To show that π' is a $(n', Rn', M_{b_l}, {b_m}', \epsilon')$-protocol, we need to show that for each x the probability $\Pr(G'(T) \neq x | X = x)$ is at most ϵ'. We define $E_{par,>{b_m}'}$ to be the event that for transcript T from the execution of π, we can find some k and some i such that we have $\Pr(L_{i,k-1} = 1 | T^k = t^k, X = x) > {b_m}'$ or $\Pr(L_{i,k} = 1 | T^k = t^k, X = x) > {b_m}'$, $E_{tot,>b_m}$ to be that event that for the total transcript there is some i such that $\Pr(L_i = 1 | T = t, X = x) > b_m$, and E_{ig} to be the event that the players starts to

pretend ignorance. The only situation where the players starts to pretend ignorance are when there is a possible message t_{k+1} that would give $\Pr(L_{i,k-1} = 1|T^k = t^k, X = x) > {b_m}'$ (as in the proof of Lemma 3.17) or if we would otherwise have increase some player i's probability of being a leaker $\Pr(L_{i,k-1} = 1|T^k = t^k, X = x)$ to a probability greater than ${b_m}'$. In the first case there is still probability at least $\frac{1-{b_m}'}{3}$ that $E_{par,>{b_m}'}$ would have happened if the players did not pretend ignorance, and in the second case there is probability 1 that $E_{par,>{b_m}'}$ would have happened. So we still have

$$\Pr(E_{ig}|X = x) \leq \frac{3}{1 - {b_m}'} \Pr(E_{par,>{b_m}'}|X = x).$$

All other computations and arguments are exactly as in the proof of Lemma 3.17. This gives us

$$\Pr(E_{ig}|X = x) \leq 3\epsilon \frac{(1 - b_m)}{({b_m}' - b_m)(1 - {b_m}')}.$$

Now we get

$$\begin{aligned}\Pr(G'(T) \neq x|X = x) \leq & \Pr(G(T) \neq x|X = x) + \Pr(G'(T') \neq G(T)|X = x) \\ \leq & \epsilon + \Pr(E_{ig}|X = x) \\ \leq & \epsilon + 3\epsilon \frac{(1 - b_m)}{({b_m}' - b_m)(1 - {b_m}')}.\end{aligned}$$

For sufficiently small ϵ, depending on ϵ', b_m and ${b_m}'$, this is less than ϵ'. □

Proposition 3.24. *For any $b_l \leq b_m$ and any model M_{b_l} the risky b_m-capacity for M_{b_l} is at most $\frac{-b_l \log(1-b_m)}{b_m} - b_l \log(e)$.*

Proof. As "b_l-threshold centrally organised" is the most general model, we can assume that M_{b_l} is this model. By continuity of $b_m \mapsto \frac{-b_l \log(1-b_m)}{b_m} - b_l \log(e)$ the result follows from Theorem 3.22 and Proposition 3.23. □

Proposition 3.25. *For any $b_l \geq b_m$ and any adaptive model M_{b_l} the risky b_m-capacity is at most $\frac{-b_m \log(1-b_m)}{b_m} - b_m \log(e) = \log(1 - b_m) - b_m \log(e)$.*

Proof. Let π be a risky $(n, h, M_{b_l}, b_m, \epsilon)$-protocol. Then we must have $\Pr(L_i = 1) \leq b_m$, so it is also a risky $(n, h, \text{"}b_l - \text{threshold centrally organised"}, b_m, \epsilon)$-protocol. The Proposition now follows from Proposition 3.24. □

We now show that the upper bounds are tight in two of the models. The proof relies on Theorem 3.9, so it is not constructive.

Proposition 3.26. *Let $b_l < b_m$. If M_{b_l} is "b_l-threshold centrally organised" or "b_l-threshold informed choice", the safe b_m-capacity for M_{b_l} is at least $\frac{-b_l \log(1-b_m)}{b_m} - b_l \log(e)$.*

Proof. As "b_l-threshold informed choice" is the most restrictive of the two models, we can assume that M_{b_l} is this model.

Let b_l and b_m be fixed, let n and $\epsilon > 0$ be given and choose some large integer m. We will define a protocol π for the model M_{b_l} that works in m stages. At the beginning everyone are *available* and after each stage some players be *unavailable*, meaning that they will not send any more messages. Before each stage starts, everyone, even an observer who does not know X will be able to compute who should be available and who should be unavailable in that stage. Define $n' = \lfloor \frac{b_m - b_l}{2b_m} n \rfloor$, ${b_l}' = \frac{2b_l b_m}{(b_m - b_l)m}$ and $h = \left\lfloor \left(D\left(\frac{b_l'}{b_m} \middle|\middle| \frac{b_l'(1-b_m)}{(b_m(1-b_l'))} \right) - m^{-2} \right) n' \right\rfloor$ and let X be uniformly distributed over $\{1, \ldots, 2^h\}^m$.

If there is less than n' players available at the beginning of stage j, the protocol halts. Otherwise, each of the first n' players who are available, choose whether to become leaker independently with probability ${b_l}'$. Assuming that n and hence n' is sufficiently big (given b_l, b_m, ϵ and m), we know from the proof of Theorem 3.9 that there exists a safe $(n', h, \mathrm{Indep}_{{b_l}'}(n'), b_m, \epsilon/(2m))$-protocol π. We let the n' players follow this protocol to leak X_j. According to Definition 3.4 there is a function G of the communication, that with high probability guesses the value the leakers tried to leak. Let $\hat{X}_j$ be G of the communication of the j'th stage. If $x_j \neq \hat{X}_j$ we let all the leakers pretend ignorance for the rest of the entire protocol, and the non-leakers stay non-leakers. At the end of the j'th stage some players will have $\Pr(L_{i,(j)} = 1 | T^{(j)} = t^{(j)}, X^j = \hat{X}^j) > 0$, where (j) denotes the round where stage j finishes. We let these players be unavailable for all the following stages, and all other available players stay available. In particular we see that all players who are leaking in a given stage, will be unavailable in all the following stages, unless $X^j \neq \hat{X}^j$, in which case they will pretend ignorance.

Eve can compute $\hat{X}_j$, so she can determine if the players are pretending ignorance. Hence, once they pretend ignorance, Eve will not get any further information, so we

Parameters:

n : number of players

b_l : probability of each player being leaker

b_m : threshold of reasonable doubt

ϵ : acceptable probability of error

m : number of stages

π : a safe $(n', h, \text{Indep}_{b_l'}(n'), b_m, \epsilon/(2m))$-protocol π, where $n' = \lfloor \frac{b_m - b_l}{2b_m} n \rfloor$, $b_l' = \frac{2b_l b_m}{(b_m - b_l)m}$ and $h = \left\lfloor \left(D\left(\frac{b_l'}{b_m} \middle|\middle| \frac{b_l'(1-b_m)}{(b_m(1-b_l'))} \right) - m^{-2} \right) n' \right\rfloor$

Input distribution: X uniformly distributed on $\{0,1\}^{hm}$, $L_{i,0} = 0$ for all i. If $L_{i,k} = 1$ then PLR_i learns X after message k.

Protocol:

1. Parse X as $(X_1, \ldots, X_m)$ with $\forall j : X_j \in \{0,1\}^h$
2. Set all n players to available
3. For j from 1 to m
4. If less than n' players are available, terminate the protocol
5. If the leakers are not pretending ignorance, each of the first n' available players become a leaker with probability b_l' independently of each other
6. The first n' available players follow π to leak X_j. Let $\hat{X}_j$ denote the output
7. If $\hat{X}_j \neq X_j$ all leakers start to pretend ignorance
8. Every player i with $\Pr(L_{i,(j)} = 1 | T^{(j)} = t^{(j)}, X^j = \hat{X}^j) > 0$ becomes unavailable

Figure 3.10: Protocol from the proof of Proposition 3.26.

only need to prove that until they pretend ignorance, they have reasonable doubt.

Let j be the last stage in which player i send a message where he did not pretend ignorance. As he did not pretend ignorance, we must have $X^{j-1} = \hat{X}^{j-1}$, and he must have been available in stage j, otherwise he would not have sent a message in

that stage. That means that $\Pr(L_{i,(j-1)} = 1 | T^{(j-1)} = t^{(j-1)}, X^{j-1} = x^{j-1}) = 0$ so Eve would know that he did not leak in earlier rounds. As he had probability ${b_l}'$ of becoming a leaker at stage j and the players used a safe $(n', h, \text{Indep}_{b_l'}(n'), b_m, \epsilon/(2m))$-protocol in round j, we must have $\Pr(L_{i,(j)} = 1 | T^{(j)} = t^{(j)}, X = x) \leq b_m$, and no further message will change this probability. Thus, the protocol ensures reasonable doubt.

Next we want to compute the rate for the protocol we have defined. In the limit, when $n \to \infty$ much faster than $m \to \infty$ the rate is

$$
\begin{aligned}
&\lim_{m\to\infty}\lim_{n\to\infty}\frac{hm}{n}\\
&=\lim_{m\to\infty}\lim_{n\to\infty}\frac{\left\lfloor\left(D\left(\frac{b_l'}{b_m}\middle\|\frac{b_l'(1-b_m)}{(b_m(1-b_l'))}\right)-m^{-2}\right)n'\right\rfloor m}{n}\\
&=\lim_{m\to\infty}\lim_{n\to\infty}\frac{\left\lfloor\left(\frac{-b_l'\log(1-b_m)+b_m\log(1-b_l')}{b_m}-m^{-2}\right)n'\right\rfloor m}{n}\\
&\geq\lim_{m\to\infty}\lim_{n\to\infty}\frac{\frac{-b_l'\log(1-b_m)+b_m\log(1-b_l')}{b_m}n'm-(m^{-2}n'+1)m}{n}\\
&\geq\lim_{m\to\infty}\lim_{n\to\infty}\frac{\frac{-\frac{2b_lb_m}{(b_m-b_l)m}\log(1-b_m)+b_m\log\left(1-\frac{2b_lb_m}{(b_m-b_l)m}\right)}{b_m}\left(\frac{b_m-b_l}{2b_m}n-1\right)m-\left(m^{-2}\frac{b_m-b_l}{2b_m}n+1\right)m}{n}\\
&=\lim_{m\to\infty}\frac{-\frac{2b_lb_m}{(b_m-b_l)m}\log(1-b_m)+b_m\log\left(1-\frac{2b_lb_m}{(b_m-b_l)m}\right)}{b_m}\left(\frac{b_m-b_l}{2b_m}\right)m-\left(m^{-2}\frac{b_m-b_l}{2b_m}\right)m\\
&=\lim_{m\to\infty}\frac{-b_l\log(1-b_m)+m\frac{b_m-b_l}{2}\log\left(1-\frac{2b_lb_m}{(b_m-b_l)m}\right)}{b_m}-\left(m^{-2}\frac{b_m-b_l}{2b_m}\right)m\\
&=\frac{-b_l\log(1-b_m)-b_lb_m\log(e)}{b_m},
\end{aligned}
$$

as we wanted.

Finally, we want to compute the probability of error. We divide the errors into two types. A type one error is an error where $\hat{X}_j \neq X_j$ for some j. A type two error is the case where the protocol halts because there are less than n' available players left.

By construction, the probability of getting a type one error in stage j is at most $\frac{\epsilon}{2m}$. From the proof of Theorem 3.9, we see that if the players never pretend igno-

rance the number of players who would become unavailable in stage j is binomially distributed with parameters n' and ${b_l}' + (1 - {b_l}')\frac{a}{d}$ where a and d are parameters from that proof. For any $\delta > 0$ we can choose a and d such that $\frac{a}{d} \leq \frac{{b_l}'(1-b_m)}{b_m(1-{b_l}')} + \delta$. Then

$$\begin{aligned}
{b_l}' + (1 - {b_l}')\frac{a}{d} \leq & b_l' + (1 - b_l')\left(\frac{b_l'(1-b_m)}{b_m(1-b_l')} + \delta\right) \\
\leq & b_l' + \frac{b_l'(1-b_m)}{b_m} + \delta \\
= & \frac{{b_l}'}{b_m} + \delta.
\end{aligned}$$

Thus, the total number of players who would become unavailable if there were enough players and they never pretended ignorance would be binomially distributed with parameters mn' and $p \leq \frac{{b_l}'}{b_m} + \delta = \frac{2b_l}{(b_m - b_l)m} + \delta$ and thus have expectation

$$\begin{aligned}
mn'p \leq & m\left\lfloor \frac{b_m - b_l}{2b_m} n \right\rfloor \left(\frac{2b_l}{(b_m - b_l)m} + \delta\right) \\
\leq & m\frac{b_m - b_l}{2b_m} n \left(\frac{2b_l}{(b_m - b_l)m} + \delta\right) \\
\leq & \frac{b_l}{b_m} n + \delta nm
\end{aligned}$$

By choosing δ to be sufficiently small depending on m, and n sufficiently big Chebyshev's inequality shows that with probability greater than $1 - \frac{\epsilon}{2}$ the total number of player who become unavailable is at most $\frac{b_m + b_l}{2b_m} n \leq n - \lfloor \frac{b_m - b_l}{2b_m} n \rfloor$ and hence there will be n' available players left for the last stage. Thus, the probability of a type two error would be less than $\frac{\epsilon}{2}$ so the total probability of error is less than ϵ. □

Theorem 3.27. *If M_{b_l} is "b_l-threshold centrally organised" or "b_l-threshold informed choice" both the safe and the risky b_m-capacity for M_{b_l} is $\frac{-\min(b_l, b_m)\log(1-b_m)}{b_m} - \min(b_l, b_m)\log(e)$.*

Proof. For $b_l < b_m$ is follows from Proposition 3.24, Proposition 3.26 and the fact that the risky capacity must be at least the same as the safe. For $b_l \geq b_m$ is follows from the case $b_l < b_m$, Proposition 3.25 and the fact that the b_m-capacity must be non-decreasing in b_m. □

For an illustration of this theorem, see Figure 3.11.

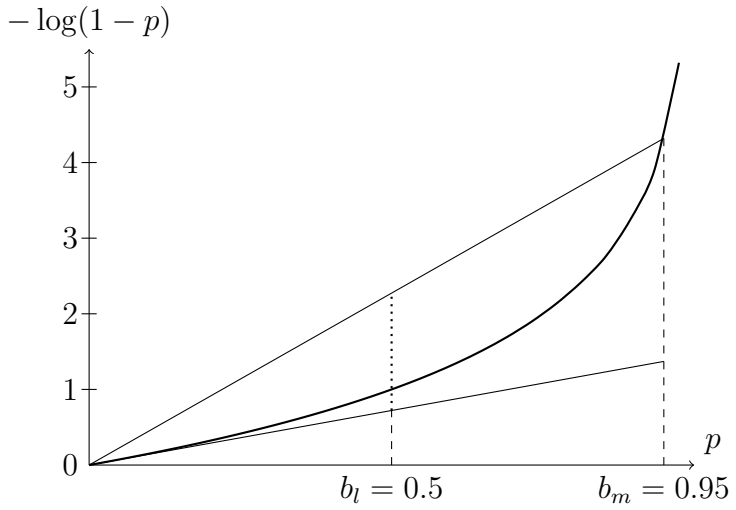

Figure 3.11: This figure illustrates the advantage of the adaptive models "b_l-threshold centrally organised" and "b_l-threshold informed choice" compared to the non-adaptive model. Most of the figure is as Figure 3.2. The new line is the tangent to $p \mapsto -\log(1-p)$ at $p = 0$. For $b_l \leq b_m$ the safe/risky b_m-capacity for M_{b_l} is given as the length of the dotted line. In this case, the advantage in using these two adaptive models over the static one, is given by the difference between the lower line and the curve. When $b_l > b_m$ in the static model, there is not even reasonable doubt from the beginning, and the capacity is $-\infty$. In these two adaptive models, the capacity is the same as for $b_l = b_m$.

Notice that while our upper bounds for the six models are the same and are the same for the safe and risky case, the b_m-capacities might be different between the models, and the safe b_m-capacity might even be different from the risky b_m-capacity for some models. Similarly, even though our upper bound for all the models does not depend on b_l as long as $b_l \geq b_m$, we conjecture that in the b_l-dormant models the b_m-capacity for M_{b_m} is less than the b_m-capacity for M_1.

Chapter 4

Resilient Cryptogenography

In the previous chapter we considered a model for cryptogenography where all the communicating parties were collaborating in revealing some information: some players were sending messages that were correlated with a secret, and some players were sending messages that were not, but everyone were following the same protocol. In this chapter we will see what happens when some players, censors, try to obstruct a protocol by sending misleading messages.

We will see that if the probability, b_c, that each player is a censor is at least $b_m - b_l$, then the censors can completely prevent the leakers from sending any information. When $b_l + b_c < b_m$, the censors can have two effects on the leakers ability to leak information. First, if $b_c \geq b_l$ the censors can "spread false stories", that is, they pretend that they are leakers and that X is some value x'. Similarly, if there are more censors than leakers, they can split into $\lfloor \frac{b_c}{b_l} \rfloor$ groups that each pretend that they are the true leakers, and that $X = x_i$. Hence, the best the leakers can hope for is that after the communication an observer can write down a list of $1 + \lfloor \frac{b_c}{b_l} \rfloor$ elements that contains the true value x. To capture this, we need to redefine reliable leakage, to allow the observer output a list of fixed length of guesses about x and only require that x is on the list with probability $1 - \epsilon$. Corollary 4.17 shows that the leakers can achieve this for list of length $1 + \lfloor \frac{b_c}{b_l} \rfloor$ but no shorter. Secondly, as we will see in Corollary 4.23, the censors can lower the number of bits that the leaker can reveal to $D\left(\frac{b_l}{b_m} || \frac{b_l(1-b_m)}{b_m(1-b_c-b_l)}\right)$ per player. The main theorem of this chapter is Theorem 4.26, which shows that asymptotically as number of people tend to infinity,

the leakers can get the best of both: they can reveal $D\left(\frac{b_l}{b_m}||\frac{b_l(1-b_m)}{b_m(1-b_c-b_l)}\right)$ bits per person, and still get down to a list of size $1+\lfloor\frac{b_c}{b_l}\rfloor$.

We will also consider the case where both the fraction of leakers and the fraction of censors are close to 0. In this case we will see that even if there are many more censors than leakers, say $b_c = 10^{-3}$ and $b_l = 10^{-6}$, the censors have almost no effect on the rate. All the positive results in this chapter are existential, so we do not construct good leaker protocols. However, the negative results are constructive: we give algorithms that given a leaker protocol construct good censor protocols.

In a censor/leaker game each player can have one of three different alignments. She can be *leaker* (denoted by 1), *censor* (-1) or *neutral* (0). We let L_i be the random variable that gives the alignment of player i and let $L = (L_1, \ldots, L_n)$. A *leaker protocol* π is a communication protocol that for each possible value t^k of previous transcript specifies

1. Does the protocol terminate here, and if not:
2. who should send the next message, say player i, and
3. if $L_i = 0$ what distribution with support U_{t^k}, should player i use to send the next message, and
4. for each $x \in \mathcal{X}$, if $L_i = 1$, what distribution over U_{t^k} should player L_i use to send her next message.

Notice that requirements 3 and 4 together imply that the protocol is *non-revealing*, that is, any message that can be sent by someone, when the partial transcript is t^k, can be sent by a neutral, when the transcript is t^k. In this definition we are assuming that the neutral people are willing to follow any leaker protocol π that we might specify. This is of course not realistic, but if the neutral players send out some randomness, for example by the exact time they post something online, we can model this as an innocent communication protocol ι. In Chapter 6 we will see that for any ι that contains sufficiently much randomness and any protocol π as defined above, we can build a protocol ι^π that implements π but where the neutral people simply follow ι (Theorem 6.3). Hence, to *not* be a neutral player in ι^π you would have to change your behavior.

We assume that the censors follow a secret censor protocol σ, which is chosen after they learn the protocol π. The *censor protocol* specifies for each possible partial transcript t^k, secret value x and set of censors, which probability distribution over U_{t^k} a censor should use to choose her next message, if she is next to send a message. If the distribution never depends on who the other censors are, we say that the protocol is *autonomous*. We assume that the censors only send messages when π says that they are the next person to send a message, and we assume that they can only send messages from U_{t^k}, that is, messages that could be sent according to π. This assumption will make the formal setup simpler, and it does not change the power of the censors: if a censor did send a message at a time when they are not supposed to send a message, or if they sent a message that a neutral player or a leaker would never send, everyone would know that this person was a censor. Everyone could then ignore the messages sent by that person. If a player does not send a message, when it is his turn, everyone would also know that he is a censor, and we could simply pretend that he sent the lexicographically first message in U_{t^k}. The only advantage a censor could have in not respecting when to send messages or not choosing his messages from U_{t^k} is to communicate with other censors or an observer. However, all the censors have the same information available when sending messages, and Eve also has the same information, so this would not be useful for the censors.[1] We assume that π is known to everyone, but that only the censors know σ, that is, G will not depend on σ.

We have a parameter $b = (b_l, b_c, b_m)$ that determines the distribution of the alignments L, and how suspicious they are willing to look to Eve: We assume that the players' alignments are chosen independently at random, that each player is a leaker with probability b_l, a censor with probability b_c and that no leaker wants to look like they are a leaker with probability greater than b_m.[2]

Can the leakers still reveal some information? If $b_l = b_c$, the censors can simply pretend to be leakers and pretend that the secret is some x' rather than the true

[1]The censors could use communication to get some shared randomness, however this will not give them any advantage: See the proof of Theorem 4.14 and the remarks after the proof.

[2]If the leakers follow a protocol that ensures that they never look like they are a leaker with probability above b_m, then no one will ever look like they are a leaker with probability above b_m: Suppose for contradiction that Alice looks like she is a leaker with probability above b_m, after running such a protocol. Then by assumption Alice cannot be a leaker, and hence her probability of being a leaker is 0.

value x. Thus, no matter what protocol π we use, it will always be possible for the censors to create some doubt about what the secret is. However, if the set of possible secrets $\mathcal{X}$ is huge, reducing the set of possible secret to just two values, or even some constant number, $\mathcal{L}$, of values, might be useful. Thus, we have two different parameters in how effectively we can leak information: How small a set can an observer confidently say that X belongs to, and the number $2^{\lceil h \rceil}$ of elements in the prior set of possible secrets $\mathcal{X}$. We also allow for some probability of at most ϵ, that x is not on the list of $\mathcal{L}$ elements.

Definition 4.1. Let $h \in \mathbb{R}^+, b_l, b_m, b_c \in [0,1)$, $\epsilon \in (0,1)$, $n, \mathcal{L} \in \mathbb{N}$ with $\frac{b_l}{1-b_c} < b_m$ be given. Let $b = (b_l, b_c, b_m)$. We say that *player i's alignment is given by b* and write $L_i \sim b$ if L_i is a random variable that is 1 with probability b_l, is -1 with probability b_c and is 0 otherwise. We say that *the alignments are given by b* and write $L \sim b$ if $L_i \sim b$ for all i and all the L_i are independent. Let $L_- = (\min(L_1, 0), \ldots, \min(L_n, 0))$, that is, L_- specifies the set of censors. A $(n, h, \mathcal{L}, \epsilon, b)$-*protocol* π is a protocol for n players such that when $L \sim b$, and when the secret is distributed uniformly on $[2^{\lceil h \rceil}]$ independently from L then no matter what protocol σ the censors use, we have:

Reasonable doubt: For all players i, secrets x, transcript t and set of censors l_-, such that t occurs with positive probability for $X = x$ and $L_- = l_-$ when the censors follow σ, we have $\Pr(L_i = 1 | X = x, L_- = l_-, T = t) \leq b_m$.

Reliable leakage: There is a function G sending transcripts T to subsets of $\mathcal{X}$ of size at most $\mathcal{L}$, such that for any strategy for the censors and any $x \in \mathcal{X}$ we have $\Pr(x \in G(T) | X = x) \geq 1 - \epsilon$.

For $b = (b_l, b_c, b_m)$ a *b-leaker protocol* is a leaker protocol, π, such that for any censor protocol, σ, the requirement of reasonable doubt is satisfied.

The inequality $\frac{b_l}{1-b_c} < b_m$ together with $b_m < 1$ implies that $b_l + b_c < 1$, so the described distribution on L_i is possible, and it also implies $\Pr(L_i = 1 | L_- = l_-) < b_m$, so there is reasonable doubt about each player, before they start communicating. In the following we will always assume that $b_l, b_m, b_c \in [0,1)$ and that they satisfy $\frac{b_l}{1-b_c} < b_m$. Because we are assuming the L_is to be independent, there is no reason to consider risky protocols.

In the "reasonable doubt" requirement, the probability $\Pr(L_i = 1|X = x, L_- = l_-, T = t)$ refers to the joint distribution of (X, L, T). This distribution depends on b, π and σ. However, we will see that the probability $\Pr(L_i = 1|X = x, L_- = l_-, T = t)$ does not depend on σ. Notice that the probability is given L_-, that is, we assume that Eve knows who the censors are. If we did not have this assumption and allowed $\Pr(L_i = 1|X = x, L_- = l_-, T = t) > b_m$, the censors could just reveal themselves after the execution of the protocol. Then Eve would know L_- and there would no longer be reasonable doubt about player i. The assumption that Eve knows L_- also implies that the reasonable doubt requirement is always satisfied for the censors.

Definition 4.2. For $b = (b_l, b_c, b_m)$ we define

$$r(b) = \frac{b_m(1 - b_l - b_c)}{b_l(1 - b_m)}.$$

We also define $b' = (b'_l, b'_c, b'_m)$, where $b'_l = \max\left(b_l - \frac{b_c}{r(b)-1}, 0\right)$, $b'_c = 0$ and $b'_m = \max\left(b_m\left(1 - \frac{b_c}{b_l(r(b)-1)}\right), 0\right)$.

The following theorem will be useful for proving upper bounds on the amount of information the leakers can reveal. Intuitively, it says that the censors can choose a strategy that "neutralizes" some of the leakers, so for an observer who knows π but does not know that some of the players are censors, it will look as if there are fewer leakers and that the leakers are only willing to look less suspicious than they really are. When the alignments and requirement of reasonable doubt are given by b the censors can make it look like they are given by b' as defined above.

Theorem 4.1. *Let $b = (b_l, b_c, b_m)$ and b' be as given by Definition 4.2. If π is a b-leaker protocol it is also a b'-leaker protocol, and there is a censor protocol σ such that the distribution of (X, T) is the same when $L \sim b$ as when $L \sim b'$.*

The rest of this section will build up to the proof of this theorem.

Proposition 4.2. *Let $b = (b_l, b_c, b_m)$. If $b'_l > 0$, then $r(b) = r(b')$.*

Proof. Assume $b'_l > 0$ and define $r = r(b)$. Then we must have $0 < b'_l = b_l - \frac{b_c}{r-1} = b_l\left(1 - \frac{b_c}{b_l(r-1)}\right)$ so $1 - \frac{b_c}{b_l(r-1)} > 0$ and hence $b_m\left(1 - \frac{b_c}{b_l(r-1)}\right) > 0$ so $b'_m = b_m\left(1 - \frac{b_c}{b_l(r-1)}\right)$.

We want to show that $r(b) = r(b')$. That is

$$\frac{b_m(1-b_l-b_c)}{b_l(1-b_m)} = \frac{b_m\left(1-b_l+\frac{b_c}{r-1}\right)}{b_l\left(1-b_m\left(1-\frac{b_c}{b_l(r-1)}\right)\right)}.$$

By assumption $b_m \geq b_l \geq b'_l > 0$, so we can divide through by $\frac{b_m}{b_l}$, giving us

$$\frac{1-b_l-b_c}{1-b_m} = \frac{1-b_l+\frac{b_c}{r-1}}{1-b_m\left(1-\frac{b_c}{b_l(r-1)}\right)}.$$

Then we multiply by the denominators

$$(1-b_m)\left(1-b_l+\frac{b_c}{r-1}\right) = (1-b_l-b_c)\left(1-b_m\left(1-\frac{b_c}{b_l(r-1)}\right)\right),$$

expand

$$\begin{aligned}1-b_l+\frac{b_c}{r-1}-b_m+b_mb_l-\frac{b_cb_m}{r-1}&\\ =1-b_l-b_c-b_m+b_mb_l+b_mb_c+\frac{b_cb_m}{b_l(r-1)}-\frac{b_mb_c}{r-1}-\frac{b_mb_c^2}{b_l(r-1)},&\end{aligned}$$

and simplify

$$\frac{b_c}{r-1} = -b_c+b_mb_c+\frac{b_cb_m}{b_l(r-1)}-\frac{b_mb_c^2}{b_l(r-1)}.$$

If $b_c = 0$ it is clearly true. Otherwise, we can divide through by b_c and rearrange to get

$$\frac{b_l-b_m+b_cb_m}{b_l(r-1)} = -1+b_m.$$

We assume $b_m < 1$ so we can divide through by $-\frac{1-b_m}{r-1}$ to get

$$\frac{b_m-b_l-b_cb_m}{b_l(1-b_m)} = r-1.$$

All the operations we used can be inverted, so this equation is equivalent to $r(b) =$

$r(b')$. We see that indeed

$$r - 1 = \frac{b_m(1 - b_l - b_c)}{b_l(1 - b_m)} - 1 = \frac{b_m - b_l - b_c b_m}{b_l(1 - b_m)}.$$

Thus, $r(b) = r(b')$. □

Just after Definition 4.1 we decided to always assume $\frac{b_l}{1-b_c} < b_m$. The following shows that this assumption is equivalent to $r(b) > 1$.

Proposition 4.3. *For $b_l \in (0,1), b_c, b_m \in [0,1)$ we have $\frac{b_l}{1-b_c} < b_m \Leftrightarrow r(b) > 1$.*

Proof. Notice that

$$\begin{aligned} r(b) =& \frac{b_m(1 - b_l - b_c)}{b_l(1 - b_m)} \\ =& \frac{b_m(1 - b_c)}{b_l} \frac{1 - b_l - b_c}{(1 - b_c)(1 - b_m)} \\ =& \frac{b_m(1 - b_c)}{b_l} \frac{1 - b_l - b_c}{1 - b_l - b_c - (1 - b_c)\left(b_m - \frac{b_l}{1-b_c}\right)}. \end{aligned}$$

For $b_l, b_c, b_m \in [0,1)$ the assumption $\frac{b_l}{1-b_c} < b_m$ is equivalent to $b_m - \frac{b_l}{1-b_c} > 0$ and to $\frac{b_m(1-b_c)}{b_l} > 1$, so $\frac{b_l}{1-b_c} < b_m$ implies that each of the two factors in the last line, and hence $r(b)$, is greater than 1. Conversely, if $r(b) > 1$ and $b_l \in (0,1), b_c, b_m \in [0,1)$ then at least one of the two factors must be greater than 1 and hence $\frac{b_l}{1-b_c} < b_m$. □

The following concept turns out to be useful when analyzing whether a protocol satisfies the requirement of reasonable doubt. Specifically, we will see that a leaker protocol π is a b-protocol if and only if $r_{i,t} \leq r(b)$ for all i, t.

Definition 4.3. For a leaker protocol π, censor protocol σ, a player i and a transcript t the *likelihood ratio* $r_{i,t}$ is given by

$$r_{i,t} = \frac{\Pr(T = t | X = x, L_i = 1)}{\Pr(T = t | X = x, L_i = 0)}.$$

The *likelihood ratio for player i* is $r_i = \max_t r_{i,t}$, the *likelihood ratio* for protocols π and σ is $\max_{i,t} r_{i,t}$, and the *likelihood ratio* for π is the maximum of likelihood ratios for π and σ over all leaker protocols σ.

The likelihood ratio corresponds to what Evfimievski, Gehrke and Srikant call amplification [25]. It also plays the same role as e^ϵ in differential privacy. However, unlike for differential privacy, the guarantee we give in this thesis is one-sided: we assume that players do not want to be revealed as leakers but also that they do not mind being revealed as non-leakers.

We will now see that only player is message can affect $r_{i,t}$: it does not matter who the censors are or what protocol they follow.

Proposition 4.4. *Let π be a leaker protocol, σ a censor protocol, t a transcript and let $K(i,t)$ be the set of rounds where player i sent a message in t. Let l_- be an n-tuple taking values in $\{0,-1\}$, with $(l_-)_i = 0$. The likelihood ratio for player i and t is*

$$\begin{aligned} r_{i,t} &= \prod_{k\in K(i,t)} \frac{\Pr(T_k = t_k | X = x, L_i = 1, T^{k-1} = t^{k-1})}{\Pr(T_k = t_k | X = x, L_i = 0, T^{k-1} = t^{k-1})} \\ &= \prod_{k\in K(i,t)} \frac{\Pr(T_k = t_k | X = x, L_i = 1, L_- = l_-, T^{k-1} = t^{k-1})}{\Pr(T_k = t_k | X = x, L_i = 0, L_- = l_-, T^{k-1} = t^{k-1})} \\ &= \frac{\Pr(T = t | X = x, L_i = 1, L_- = l_-)}{\Pr(T = t | X = x, L_i = 0, L_- = l_-)}. \end{aligned}$$

In particular the likelihood ratio $r_{i,t}$ of π and σ does not depend on σ.

In the proof of this and later proposition, we will sometime shorten the notation for probabilities, by omitting the random variables and just write the value. For example we write $\Pr(l_i | x, l_-, t^k)$ instead of

$$\Pr(L_i = l_i | X = x, L_- = l_-, T^k = t^k).$$

We will only do this when the random variable is the capital version of the letter used to represent the values it takes, so the notation is well-defined.

In order to prove Proposition 4.4 we need the following proposition, which will also be useful later. Recall that a censor protocol is said to be autonomous if a censor's message never depends on who the other censors are.

Proposition 4.5. *Let the alignments $L_1, \ldots, L_n$ and the secret X be independent random variables. For fixed π, σ and integer k, the alignments $L_1, \ldots, L_n$ are inde-*

pendent given X, L_- and T^k. If σ is an autonomous censor protocol, then $L_1, \ldots, L_n$ are independent given X and T^k.

Proof. First we show that $L_1, \ldots, L_n$ are independent given X, L_- and T^k. Let π and σ be fixed. We prove the statement by induction on k. For $k = 0$ it is true by assumption.

For the induction step, assume that the statement holds for k. That is, for all n-tuples l of alignments, and all x and t such that $\Pr(X = x, L_- = l_-, T^k = t^k) > 0$ we have

$$\Pr(L = l | X = x, L_- = l_-, T^k = t^k) = \prod_i \Pr(L_i = l_i | X = x, L_- = l_-, T^k = t^k).$$

By multiplying by $\Pr(X = x, L_- = l_-, T^k = t^k)$ (or dividing to go in the other direction) and using that L_- is a function of L, we see that this is equivalent to saying that for all x, l and t we have

$$\Pr(x, l, t^k) = \Pr(x, l_-, t^k) \prod_i \Pr(l_i | x, l_-, t^k). \tag{4.1}$$

We want to show that if this is true, then the same statement holds for $k + 1$.

Assume that after transcript t^k, the next person to send a message is player j. We have

$$\begin{aligned}
&\Pr(X = x, L = l, T^{k+1} = t^{k+1}) \\
&\quad = \Pr(T_{k+1} = t_{k+1} | X = x, L = l, T^k = t^k) \Pr(X = x, L = l, T^k = t^k) \\
&\quad = \Pr(T_{k+1} = t_{k+1} | x, l_j, l_-, t^k) \Pr(X = x, L = l, T^k = t^k).
\end{aligned} \tag{4.2}$$

Here the second equality follows from the fact that a message only depends on the secret X, the alignment L_j of the sender, the previous transcript T^k and (if j is a censor) on L_-.

If $l_i = -1$ then this information follows from l_-, so $\Pr(T_{k+1} = t_{k+1} | X = x, L_i = l_i, L_- = l_-, T^k = t^k) = \Pr(T_{k+1} = t_{k+1} | X = x, L_- = l_-, T^k = t^k)$. If $l_i \neq -1$, and

player $j \neq i$ sends the $k+1$'th message we also have

$$\begin{aligned}
&\Pr(T_{k+1} = t_{k+1} | X = x, L_i = l_i, L_- = l_-, T^k = t^k) \\
&\quad = \sum_{a \in \{-1,0,1\}} \Pr(T_{k+1} = t_{k+1} | x, L_j = a, l_-, t^k) \Pr(L_j = a | x, l_i, l_-, t^k) \\
&\quad = \sum_{a \in \{-1,0,1\}} \Pr(T_{k+1} = t_{k+1} | x, L_j = a, l_-, t^k) \Pr(L_j = a | x, l_-, t^k) \\
&\quad = \Pr(T_{k+1} = t_{k+1} | X = x, L_- = l_-, T^k = t^k). \qquad (4.3)
\end{aligned}$$

In the first equality we are using the assumption that a message only depends on the secret X, alignment of the sender, previous transcript, and possibly L_- (if the sender is censor). In the second equality we use the induction hypothesis which says that the L_i's are independent given X, L_- and T^k. Now for $i \neq j$ we get

$$\begin{aligned}
&\Pr(L_i = l_i | X = x, L_- = l_-, T^{k+1} = t^{k+1}) \\
&= \frac{\Pr(X = x, L_i = l_i, L_- = l_-, T^{k+1} = t^{k+1})}{\Pr(X = x, L_- = l_-, T^{k+1} = t^{k+1})} \\
&= \frac{\Pr(T_{k+1} = t_{k+1} | x, l_i, l_-, t^k) \Pr(X = x, L_i = l_i, L_- = l_-, T^k = t^k)}{\Pr(T_{k+1} = t_{k+1} | x, l_-, t^k) \Pr(X = x, L_- = l_-, T^k = t^k)} \\
&= \frac{\Pr(X = x, L_i = l_i, L_- = l_-, T^k = t^k)}{\Pr(X = x, L_- = l_-, T^k = t^k)} \\
&= \Pr(L_i = l_i | X = x, L_- = l_-, T^k = t^k), \qquad (4.4)
\end{aligned}$$

Here we use (4.3) in the third equality.

We now look at the right hand side of (4.1) for $k+1$.

$$\begin{aligned}
&\Pr(x, l_-, t^{k+1}) \prod_i \Pr(l_i | x, l_-, t^{k+1}) \\
&\qquad = \Pr(x, l_j, l_-, t^{k+1}) \prod_{i \neq j} \Pr(l_i | x, l_-, t^{k+1}) \\
&\qquad = \Pr(t_{k+1} | x, l_j, l_-, t^k) \Pr(x, l_j, l_-, t^k) \cdot \prod_{i \neq j} \Pr(l_i | x, l_-, t^k) \\
&\qquad = \Pr(t_{k+1} | x, l_j, l_-, t^k) \Pr(x, l_-, t^k) \cdot \prod_i \Pr(l_i | x, l_-, t^k). \qquad (4.5)
\end{aligned}$$

In the second equality we use (4.4) to get from $\Pr(l_i|x, l_-, t^{k+1})$ to $\Pr(l_i|x, l_-, t^k)$, and in the last equality we use $\Pr(x, l_j, l_-, t^k) = \Pr(x, l_-, t^k)\Pr(l_j|x, l_-, t^k)$.

Now equations (4.2) and (4.5) show that if we multiply both sides of (4.1) by $\Pr(t_{k+1}|x, l_j, l_-, t^k)$ we get (4.1) with $k+1$ instead of k. Hence, the result follows by induction.

To prove that if σ is autonomous then $L_1, \dots L_n$ are independent given X and T^k simply remove all occurrences of $L_- = l_-$ in the above proof and note that then (4.3) also holds in the case $l_i = -1$. □

We are now ready to prove Proposition 4.4.

Proposition 4.4 (repeated). *Let π be a leaker protocol, σ a censor protocol, t a transcript and let $K(i,t)$ be the set of rounds where player i sent a message in t. Let l_- be an n-tuple taking values in $\{0, -1\}$, with $(l_-)_i = 0$. The likelihood ratio for player i and t is*

$$\begin{aligned} r_{i,t} &= \prod_{k \in K(i,t)} \frac{\Pr(T_k = t_k | X = x, L_i = 1, T^{k-1} = t^{k-1})}{\Pr(T_k = t_k | X = x, L_i = 0, T^{k-1} = t^{k-1})} \\ &= \prod_{k \in K(i,t)} \frac{\Pr(T_k = t_k | X = x, L_i = 1, L_- = l_-, T^{k-1} = t^{k-1})}{\Pr(T_k = t_k | X = x, L_i = 0, L_- = l_-, T^{k-1} = t^{k-1})} \\ &= \frac{\Pr(T = t | X = x, L_i = 1, L_- = l_-)}{\Pr(T = t | X = x, L_i = 0, L_- = l_-)}. \end{aligned}$$

In particular the likelihood ratio $r_{i,t}$ of π and σ does not depend on σ.

Proof. By repeated use of the definition of conditional probability we have

$$\Pr(T = t | x, L_i = a_i, l_-) = \prod_k \Pr(T_k = t_k | x, L_i = a_i, l_-, t^{k-1}). \tag{4.6}$$

If player $j \neq i$ sends the k'th message, we have

$$\begin{aligned}
&\Pr(T_k = t_k | X = x, L_i = 1, L_- = l_-, T^{k-1} = t^{k-1}) \\
&\quad = \sum_{a \in \{-1,0,1\}} \Pr(T_k = t_k | x, L_j = a, l_-, t^{k-1}) \Pr(L_j = a | x, L_i = 1, l_-, t^{k-1}) \\
&\quad = \sum_{a \in \{-1,0,1\}} \Pr(T_k = t_k | x, L_j = a, l_-, t^{k-1}) \Pr(L_j = a | x, L_i = 0, l_-, t^{k-1}) \\
&\quad = \Pr(T_k = t_k | X = x, L_i = 0, L_- = l_-, T^{k-1} = t^{k-1}). \qquad (4.7)
\end{aligned}$$

In the first and the last inequality we are using the assumption that a message only depends on the secret X, alignment of the sender, previous transcript and possibly L_- (if the sender is censor). In the second equality we use Proposition 4.5.

By combining (4.6) and (4.7) we get

$$\begin{aligned}
&\frac{\Pr(T = t | X = x, L_i = 1, L_- = l_-)}{\Pr(T = t | X = x, L_i = 0, L_- = l_-)} \\
&= \prod_k \frac{\Pr(T_k = t_k | X = x, L_i = 1, L_- = l_-, T^{k-1} = t^{k-1})}{\Pr(T_k = t_k | X = x, L_i = 0, L_- = l_-, T^{k-1} = t^{k-1})} \\
&= \prod_{k \in K(i,t)} \frac{\Pr(T_k = t_k | X = x, L_i = 1, L_- = l_-, T^{k-1} = t^{k-1})}{\Pr(T_k = t_k | X = x, L_i = 0, L_- = l_-, T^{k-1} = t^{k-1})} \\
&= \prod_{k \in K(i,t)} \frac{\Pr(T_k = t_k | X = x, L_i = 1, T^{k-1} = t^{k-1})}{\Pr(T_k = t_k | X = x, L_i = 0, T^{k-1} = t^{k-1})}. \qquad (4.8)
\end{aligned}$$

First equality follows from (4.6) and the second from (4.7). In the last equation we used the fact that when a message is sent by a non-censor, it only depends on X,

alignment of the sender and on previous transcript. Now

$$\begin{aligned} r_{i,t} =& \frac{\Pr(T=t|X=x, L_i=1)}{\Pr(T=t|X=x, L_i=0)} \\ =& \frac{\sum_{l_-} \Pr(T=t|X=x, L_i=1, L_-=l_-)\Pr(L_-=l_-|X=x, L_i=1)}{\sum_{l_-} \Pr(T=t|X=x, L_i=0, L_-=l_-)\Pr(L_-=l_-|X=x, L_i=0)} \\ =& \frac{\sum_{l_-} \Pr(T=t|X=x, L_i=1, L_-=l_-)\Pr(L_-=l_-|X=x, L_i=1)}{\sum_{l_-} \Pr(T=t|X=x, L_i=0, L_-=l_-)\Pr(L_-=l_-|X=x, L_i=1)} \\ =& \prod_{k\in K(i,t)} \frac{\Pr(T_k=t_k|X=x, L_i=1, T^{k-1}=t^{k-1})}{\Pr(T_k=t_k|X=x, L_i=0, T^{k-1}=t^{k-1})} \end{aligned}$$

Here the sums are over all possible l_-. The third equality follows from the fact that the L_i's are independent given X. To see the fourth equality, notice that the third right hand side is of the form $\frac{\sum_{l_-} a_{l_-} p_{l_-}}{\sum_{l_-} b_{l_-} p_{l_-}}$ where, by equation (4.8), we have $\frac{a_{l_-}}{b_{l_-}} = c$ which does not depend on l_-. Thus, $\frac{\sum_{l_-} a_{l_-} p_{l_-}}{\sum_{l_-} b_{l_-} p_{l_-}} = \frac{\sum_{l_-} cb_{l_-} p_{l_-}}{\sum_{l_-} b_{l_-} p_{l_-}} = c$. Now the theorem follows from equation (4.8). □

Proposition 4.6. *The likelihood ratio for π and σ does not depend on σ.*

Proof. The likelihood ratio for π and σ is $\max_{i,t} r_{i,t}$, where i ranges over all the players and t over all transcripts that are possible when leakers and neutrals follow π and censors follow σ. By Proposition 4.4 $r_{i,t}$ does not depend on σ and by assumption, all messages sent in σ can also be sent by non-leakers who follow π, and it is always possible that all players are non-leakers. Thus, the set of transcripts t that can occur when following π and σ is the same as the set of transcripts that can occur when following π and some other censor protocol σ'. This shows that $\max_{i,t} r_{i,t}$ does not depend on σ. □

We are now ready to prove the previously mentioned characterization of b-leaker protocols.

Proposition 4.7. *Let $b = (b_l, b_c, b_m)$ with $b_l > 0$. A protocol π is a b-leaker protocol if and only if its likelihood ratio r satisfies*

$$r \leq r(b).$$

Proof. When $l_i = -1$ we have $\Pr(L_i = 1|X = x, L_- = l_-, T = t) = 0 \leq b_m$. Otherwise,

$$\begin{aligned}
&\Pr(L_i = 1|X = x, L_- = l_-, T = t) \\
&\quad = \frac{\Pr(T = t|X = x, L_i = 1, L_- = l_-)\Pr(L_i = 1|X = x, L_- = l_-)}{\Pr(T = t|X = x, L_- = l_-)} \\
&\quad = \frac{\Pr(T = t|X = x, L_i = 1, L_- = l_-)\frac{b_l}{1-b_c}}{\Pr(T = t|x, L_i = 1, l_-)\frac{b_l}{1-b_c} + \Pr(T = t|x, L_i = 0, l_-)\frac{1-b_c-b_l}{1-b_c}} \\
&\quad = \frac{r_{i,t}b_l}{r_{i,t}b_l + 1 - b_c - b_l}.
\end{aligned}$$

Here the first equality is Bayes' formula given X and L_-. In the last equality we multiply through by $\frac{1-b_c}{\Pr(T=t|x,L_i=0,l_-)}$, and use Proposition 4.4. We see that the resulting formula is increasing in $r_{i,t}$ and for $r_{i,t} = \frac{b_m(1-b_l-b_c)}{b_l(1-b_m)}$ we get b_m. Thus, $r_{i,t} \leq \frac{b_m(1-b_l-b_c)}{b_l(1-b_m)}$ is equivalent to $\Pr(L_i = 1|X = x, L_- = l_-, T = t) \leq b_m$. □

Intuitively the following lemma says that the probability that player i is a censor can be used to "neutralize" some of the probability that he could be a leaker: if player i is a leaker, his distribution would differ from the neutral distribution, and if he is a censor it would differ. But when the observer does not know what he is, the resulting distribution would be as if there was only probability b'_l (as defined in Definition 4.2) that he was leaking, and that he was otherwise neutral. This will be used to prove Theorem 4.1, which is similar, but about all players rather than just a single player. Both proofs are constructive.

Lemma 4.8. *Let $b = (b_l, b_c, b_m)$ and let π be a b-leaker protocol. Let i be a player and let L have a distribution where all the L_j are independent and $L_i \sim b$. If σ is an autonomous censor protocol then there is an autonomous censor protocol σ', which is identical to σ for all players except i, such that*

$$\Pr(T = t|x) = b'_l \Pr(T = t|x, L_i = 1) + (1 - b'_l)\Pr(T = t|x, L_i = 0) \tag{4.9}$$

for all transcripts t. Furthermore, σ' does not depend on the distribution of L_{-i}, and for two different censor protocols σ_0 and σ_1 the resulting protocols σ'_0 and σ'_1 give the same distributions for player i.

Proof. First we argue that the case $b_l(r(b)-1) < b_c$ follows from the case $b_l(r(b)-1) = b_c$. To see this, notice that when $b_l(r(b)-1) < b_c$ we have $b'_m = b'_l = 0$. As r is continuous in b_c and $r(b_l, 0, b_m) \geq 1$ (because $b_l \leq b_m$), we can find a value β_c with $0 \leq \beta_c < b_c$ such that $b_l(r(b_l, \beta_c, b_m) - 1) = \beta_c$. Now let $\beta = (b_l, \beta_c, b_m)$ and define β' by the same formula as b'. We see that $\beta'_l = \beta'_m = 0$. Let π, σ and i satisfy the assumptions in the lemma. As π is a b-leaker protocol and $\beta_c < b_c$ it must also be a β-leaker protocol. Assuming the lemma is true in the case $b_l(r(b)-1) = b_c$ we will use the lemma for β to get an autonomous censor protocol σ' such that when $L_i \sim \beta$ and the players follow π and σ' we get $\Pr(T^k = t^k|x) = \Pr(T^k = t^k|x, L_i = 0)$. Now we define a protocol σ'' that achieves the same when $L_i \sim b$: When player i is censor he will with probability $\frac{\beta_c}{b_c}$ follow σ' as if he is censor and otherwise he will follow π as if he is neutral. As σ' is autonomous, the distribution of (X, T) when $L_i \sim \beta$ and the players follow π and σ' is the same as the distribution of (X, T) when $L_i \sim b$ and the players follow π and σ''. It is clear that if σ' does not depend on the distribution of L_{-i} and on what the other censors do in σ, then the same holds for σ''. This show that the case $b_l(r(b)-1) < b_c$ follows from the case $b_l(r(b)-1) = b_c$

Thus, in the following we will assume that $b_l(r(b)-1) \geq b_c$ and hence $b'_l = b_l - \frac{b_c}{r(b)-1} \geq 0$. We can now rewrite the left hand side of (4.9).

$$\begin{aligned}
&\Pr(T = t|X = x) \\
&= b_l \Pr(T = t|X = x, L_i = 1) \\
&\quad + (1 - b_l - b_c) \Pr(T = t|X = x, L_i = 0) \\
&\quad + b_c \Pr(T = t|X = x, L_i = -1) \\
&= b'_l \Pr(T = t|X = x, L_i = 1) \\
&\quad + (1 - b_l - b_c) \Pr(T = t|X = x, L_i = 0) \\
&\quad + b_c \frac{\Pr(T = t|X = x, L_i = 1) + (r(b) - 1) \Pr(T = t|X = x, L_i = -1)}{r(b) - 1}. \qquad (4.10)
\end{aligned}$$

Thus, equation (4.9) is equivalent to the right hand side of (4.9) being equal to the right hand side of (4.10). By subtracting the first two terms of right hand side of (4.10) from both of these, then use the fact that $(1 - b'_l) - (1 - b_l - b_c) = \frac{b_c r(b)}{r(b)-1}$, and

multiply by $\frac{r(b)-1}{b_c}$ we get,

$$\begin{aligned} & r(b)\Pr(T=t|X=x,L_i=0) \\ & \quad =\Pr(T=t|x,L_i=1)+(r(b)-1)\Pr(T=t|x,L_i=-1). \end{aligned} \tag{4.11}$$

Thus, (4.9) is equivalent to (4.11), so instead of showing that we can obtain (4.9) we will show that we can obtain (4.11).

For each secret value x and each partial transcript t^k, where player i is next to send a message, we want to define a probability distribution that player i should use to send his next message if he is a censor. We want to ensure that for each partial transcript t^k and each i we have

$$\begin{aligned} & r(b)\Pr(T^k=t^k|X=x,L_i=0) \\ & =\Pr(T^k=t^k|X=x,L_i=1)+(r(b)-1)\Pr(T^k=t^k|X=x,L_i=-1). \end{aligned} \tag{4.12}$$

We will show this by induction on k for fixed t. The equation clearly holds for $k=0$. Furthermore, if it holds for $k-1$ and the k'th message is sent by player $j\neq i$ we know from Proposition 4.5 that L_i and L_j are independent given X and T^{k-1} and hence equation (4.12) also holds for k. Finally, assume that the equality (4.12) holds for $k-1$ and that player i is next to send a message. By substituting $\Pr(T^k=t^k|X=x,L_i=-1)=\Pr(T_k=t_k|X=x,L_i=-1,T^{k-1}=t^{k-1})\Pr(T^{k-1}=t^{k-1}|X=x,L_i=-1)$ into equation (4.12) and isolating $\Pr(T_k=t_k|X=x,L_i=-1,T^{k-1}=t^{k-1})$, we see that equation (4.12) is equivalent to

$$\begin{aligned} & \Pr(T_k=t_k|X=x,L_i=-1,T^{k-1}=t^{k-1}) \\ & \quad =\frac{r(b)\Pr(T^k=t^k|X=x,L_i=0)-\Pr(T^k=t^k|X=x,L_i=1)}{(r(b)-1)\Pr(T^{k-1}=t^{k-1}|X=x,L_i=-1)}. \end{aligned} \tag{4.13}$$

We know from Proposition 4.7 that $r(b)$ is greater than or equal to the likelihood ratio of π, so it follows that the right hand side is non-negative. If we keep t^{k-1}

Parameters:

$b = (b_l, b_c, b_m)$: probability of being leaker respectively censor and threshold of reasonable doubt

π : a b-leaker protocol

σ : a censor protocol

Input distribution: $X, L_1, \ldots, L_n$ are independently distributed and for each j: $\Pr(L_j = 1) = b_l$ and $\Pr(L_j = -1) = b_c$. Each PLR_j learns L_j and if $|L_j| = 1$ she also learns X and if $L_j = -1$ she also learns $L_- = (\min(L_1, 0), \ldots, \min(L_n, 0))$. She will follow π if $L_j \neq -1$ and otherwise she will follow the censor protocol given here.
Protocol:

1. All censors PLR_j for $j \neq i$ follow σ.
2. If the transcript is t^{k-1} and PLR_i is a censor who, accordion to π, should send a message, she chooses message t_k according to the distribution

$$\begin{aligned}&\Pr(T_k = t_k | X = x, L_i = -1, T^{k-1} = t^{k-1})\\ &\quad = \frac{r(b)\Pr(T^k = t^k | X = x, L_i = 0) - \Pr(T^k = t^k | X = x, L_i = 1)}{(r(b) - 1)\Pr(T^{k-1} = t^{k-1} | X = x, L_i = -1)}.\end{aligned}$$

Figure 4.1: Censor protocol from proof of Lemma 4.8.

fixed and sum the right hand side over all t_k we get

$$\begin{aligned}&\frac{r(b)\Pr(T^{k-1} = t^{k-1} | X = x, L_i = 0) - \Pr(T^{k-1} = t^{k-1} | X = x, L_i = 1)}{(r(b) - 1)\Pr(T^{k-1} = t^{k-1} | X = x, L_i = -1)}\\ &= \frac{(r(b) - 1)\Pr(T^{k-1} = t^{k-1} | X = x, L_i = -1)}{(r(b) - 1)\Pr(T^{k-1} = t^{k-1} | X = x, L_i = -1)}\\ &= 1.\end{aligned}$$

Here the first equality follows from equation (4.12) for $k - 1$. Thus, for fixed t^{k-1}, equation (4.13) defines a probability distribution over the possible t_k, and if player i follows this distribution when he is a censor, we get (4.12) and hence (4.9). The protocol is also defined in Figure 4.1. □

Finally, we are ready to prove the main theorem of this section.

Theorem 4.1 (repeated). *Let $b = (b_l, b_c, b_m)$ and b' be as given by Definition 4.2. If π is a b-leaker protocol it is also a b'-leaker protocol, and there is a censor protocol σ such that the distribution of (X, T) is the same when $L \sim b$ as when $L \sim b'$.*

Proof of Theorem 4.1. If $b'_l = 0$ then clearly any leaker protocol satisfy the reasonable doubt requirement when $L \sim b'$, and hence is a b'-leaker protocol. If $b'_l > 0$ we know from Proposition 4.2 that $r(b) = r(b')$ so by Proposition 4.7 we know that if π is a b-leaker protocol, then π is also a b'-leaker protocol.

For the second part of the theorem, let σ_0 be the autonomous censor protocol where any censor always sends the lexicographically first possible message, and let σ_i be the censor protocol we get when using Lemma 4.8 to change player i's part of σ_{i-1}. Because this construction only affects the probability distributions for player i we see that in σ_n player i chooses his messages using the same distributions as in σ_i. Furthermore, as the player i part of the protocol you get out of Lemma 4.8 does not depend on the protocol σ you put in, we see that if we put σ_n into the Lemma, the resulting protocol for player i must agree with the output we get when input is σ_{i-1}. Hence, when we give Lemma 4.8 input σ_n and any i we get the protocol σ_n out. Thus, when $L \sim b$ and the players follow π and σ_n we must have

$$\Pr(T^k = t^k | x) = b'_l \Pr(T^k = t^k | x, L_i = 1) + (1 - b'_l) \Pr(T^k = t^k | x, L_i = 0) \quad (4.14)$$

for all i.

Let σ denote the resulting protocol σ_n. It is also define explicitly in Figure 4.2. We show by induction on k that when the players follow π and σ then (X, T^k) have the same distribution when $L \sim b$ as when $L \sim b'$. In the following we write $\Pr_b$ to denote probabilities when $L \sim b$ and $\Pr_{b'}$ to denote probabilities when $L \sim b'$. That is, we want to show $\Pr_b(X = x, T^k = t^k) = \Pr_{b'}(X = x, T^k = t^k)$.

This is clearly true for $k = 0$. It is now enough to show that

$$\Pr_b(T_k = t_k | x, t^{k-1}) = \Pr_{b'}(T_k = t_k | x, t^{k-1}).$$

Let player i be the next player to send a message when the transcript is t^{k-1}. Then

Parameters:

$b = (b_l, b_c, b_m)$: probability of being leaker respectively censor and threshold of reasonable doubt

π : a b-leaker protocol

Input distribution: $X, L_1, \ldots, L_n$ are independently distributed and for each i: $\Pr(L_i = 1) = b_l$ and $\Pr(L_i = -1) = b_c$. Each PLR_i learns L_i and if $|L_i| = 1$ she also learns X and if $L_i = -1$ she also learns $L_- = (\min(L_1, 0), \ldots, \min(L_n, 0))$. She will follow π if $L_i \neq -1$ and otherwise she will follow the censor protocol given here.
Protocol:

1. If the transcript is t^{k-1} and PLR_i is a censor who, according to π should send a message, she chooses message t_k according to the distribution

$$\begin{aligned}&\Pr(T_k = t_k | X = x, L_i = -1, T^{k-1} = t^{k-1})\\ &\quad = \frac{r(b)\Pr(T^k = t^k | X = x, L_i = 0) - \Pr(T^k = t^k | X = x, L_i = 1)}{(r(b) - 1)\Pr(T^{k-1} = t^{k-1} | X = x, L_i = -1)}.\end{aligned}$$

Figure 4.2: Censor protocol from proof of Theorem 4.1.

we have

$$\begin{aligned}\Pr_b(T_k = t_k | X = x, T^{k-1} = t^{k-1})&\\ &= \frac{\Pr_b(T^k = t^k | X = x)}{\Pr_b(T^{k-1} = t^{k-1} | X = x)}\\ &= \frac{b'_l \Pr_b(t^k | x, L_i = 1) + (1 - b'_l)\Pr_b(t^k | x, L_i = 0)}{b'_l \Pr_b(t^{k-1} | x, L_i = 1) + (1 - b'_l)\Pr_b(t^{k-1} | x, L_i = 0)}\\ &= \frac{\Pr(T^k = t^k | X = x, L_i \sim b', L_{-i} \sim b)}{\Pr(T^{k-1} = t^{k-1} | X = x, L_i \sim b', L_{-i} \sim b)}\\ &= \Pr(T^k = t^k | X = x, L_i \sim b', L_{-i} \sim b, T^{k-1} = t^{k-1})\\ &= \Pr_{b'}(T^k = t^k | X = x, T^{k-1} = t^{k-1}).\end{aligned}$$

Here the second equality comes from (4.14) and the last equality follows from the fact that the distribution player i uses to send t_k does not depend on the alignment of the other players. □

4.1 Generalized list decoding

In order to study the censor/leaker game, we first need to generalize a result about list decoding. In list decoding we usually consider a channel where input set and output set are the same, say $[m]$, and input i often gives output i, and when the output is something else, it is considered to be a channel error. We then want to find a code $\mathfrak{c} \subset [m]^n$ and a decoding function G that sends received strings y to subsets of $\mathfrak{c}$ of size at most $\mathcal{L}$, such that if $x \in [m]^n$ and y can be the output of the channel on input x with less than $(1-q)n$ errors of the channel then $x \in G(y)$. A generalization of list decoding is *list recoverability*, which we can think of in the following way. We have a channel with m possible inputs and $\binom{m}{k}$ possible outputs, one for each subset of size k of $[m]$. When the channel does not make an error, it will on input i return an output corresponding to a set that contains i. Furthermore, we still allow the channel to have up to $(1-q)n$ errors. For an introduction to list decoding and list recoverability, see Guruswami [38].

One disadvantage of list recoverable codes, is that they only exist for $\mathcal{L} \geq k$. To see this, let $x^1, \ldots, x^k$ be k different codewords, and consider the output where the i'th set contains $x_i^1, \ldots, x_i^k$. For this output, each of the k inputs $x^1, \ldots, x^k$ are possible inputs, so we must have $\mathcal{L} \geq k$. This is a problem for us, because will need $\mathcal{L} \ll k$.

We want to generalize list decoding in a similar way to list recovery, but instead of allowing the channel to output any of the $\binom{m}{k}$ subsets of size k of $[m]$, we put some restriction on which subsets the channel can output, even when it has an error. More precisely, we have a family $\mathcal{S} \subset \binom{[m]}{k}$ consisting of m subsets of $[m]$. Each $S \in \mathcal{S}$ has size k and each $i \in [m]$ is contained in k such sets S. On input i the channel can either output an S with $i \in S \in \mathcal{S}$, or it can have an error and output an S with $x \notin S \in \mathcal{S}$. We call such a channel a *k-regular pre-Bayesian channel*, where "pre-Bayesian" refers to the fact that it does not have a probability distribution on its output given the input, only a set of non-error outputs. Again, we allow the channel to make errors in up to $(1-q)n$ usages of the channel. Restricting outputs to $\mathcal{S}$ helps us, because when $x^1, \ldots, x^k$ are k different randomly chosen codewords, then there will typically not be any $S \in \mathcal{S}$ which contains $x_i^1, \ldots x_i^k$.

We do not need any further assumption about the channel and all the proofs

in this section would work for any k-regular pre-Bayesian channel. However, in the following we will work with the pre-Bayesian channel given by

$$\mathcal{S} = \{\{i, i+1, \ldots i+k-1\} \mod m : i \in [n]\}.$$

The following defines a channel (including probabilities) that is related to this pre-Bayesian channel.[3]

Definition 4.4. For $m, k \in \mathbb{N}, q \in (0,1)$ with $q > \frac{k}{m}$, we let $C_{m,k,q}$ be the channel with input set and output set $[m]$, where on input i there is probability q that the output is chosen uniformly from the set $\{i, i+1, \ldots i+k-1\} \mod m$, and otherwise it is chosen uniformly from the other $m-k$ possible values.

The assumption that $q > \frac{k}{m}$ implies that on input i each of the values in $\{i, i+1, \ldots, i+k-1\} \mod m$ are more likely outputs than the other possible outputs. Similarly, for any k-regular pre-Bayesian channel, we could define a channel where with probability q is does not have an error, and chose uniformly among the k allowed outputs, and with probability $1-q$ it choses among the other $m-k$ outputs. We call the resulting channel a *k-regular channel*.

The following definition is from Cover and Thomas [18].

Definition 4.5. For a channel C the *transmission matrix* is the matrix where rows correspond to inputs and columns to outputs and the entry (i,j) gives the probability of getting output j on input i.

A channel is *symmetric* if all the rows of its transmission matrix are permutations of each other, and all the columns are permutations of each other.

We see that the channel $C_{m,k,q}$ is symmetric. More generally, all regular channels are symmetric[4] In the following, any theorem stated for $C_{m,k,q}$ holds in the more

[3]Another k-regular pre-Bayesian channel, which might be more relevant for coding theory, is the channel that sends bits in blocks each consisting of n' bits and is said to have an error each time more than r bits in a block change. Formally, this channel has input alphabet $\{0,1\}^{n'}$ and

$$\mathcal{S} = \{B(x,r) : x \in \{0,1\}^{n'}\}$$

where $B(x,r)$ denote the set of strings in $\{0,1\}^{n'}$ that differ from x in at most r positions. This channel is k-regular for $k = \sum_{a=0}^{r} \binom{n'}{a}$.

[4]Notice that the definition of symmetric is weaker than what you might expect. In particular,

general setting of k-regular channels. The next proposition is taken from Cover and Thomas [18, Theorem 8.2.1].

Proposition 4.9. *For a symmetric channel, the capacity is achieved by a uniform distribution on the input.*

In the following $D(\cdot||\cdot)$ denotes the Kullback-Leibler divergence.

Proposition 4.10. *The capacity of $C_{m,k,q}$ is $D(q||\frac{k}{m})$.*

Proof. As $C_{m,k,q}$ is symmetric, its capacity attained for uniformly distributed input. Let X denote the uniform input and Y the corresponding output. Then the capacity is given by

$$\begin{aligned} I(X;Y) =& H(Y) - H(Y|X) \\ =& \log(m) + q\log\left(\frac{q}{k}\right) + (1-q)\log\left(\frac{1-q}{m-k}\right) \\ =& q\log\left(\frac{mq}{k}\right) + (1-q)\log\left(\frac{m(1-q)}{m-k}\right) \\ =& D\left(q\middle|\middle|\frac{k}{m}\right). \end{aligned}$$

□

We want to show the existence of a code $\mathfrak{c} \subset [m]^n$ for the channel $C_{m,k,q}$, such that for each possible output y, the set of possible inputs $x \in \mathfrak{c}$ for which y is a likely output has size at most $\mathcal{L}$. To make this precise, we use the following definition.

Definition 4.6. For $m, k, n \in \mathbb{N}$, $r \in \mathbb{N}_0$ with $m > k$, $n \geq r$ and $y \in [m]^n$ we define

$$\tilde{B}_{m,k}(y,r) = \{x \in [m]^n : |\{i : y_i \in \{x_i, x_i+1, \dots x_i+k-1\} \mod m\}| \geq n-r\}.$$

That is, if we assume that in at least $n-r$ of the usages of the channel, the output was among the most likely outputs given the input, then $\tilde{B}_{m,k}(y,r)$ denotes the set of possible inputs, when the output is y. In the case where $k=1$, $\tilde{B}_{m,k}(y,r)$ is a ball in the Hamming metric of $[m]^n$, hence the notation $\tilde{B}$. Now the goal is to show the existence a code $\mathfrak{c}$ such that $\mathfrak{c} \cap \tilde{B}_{m,k}(y, n(1-q))$ is small for all outputs y. First we will get an estimate of the size of $|\tilde{B}_{m,k}(0^n, n(1-q))|$.

a symmetric channel can have inputs i_1 and i_2 such that no "isomorphism" acting on both inputs and outputs sends i_1 to i_2.

Proposition 4.11. *For $q > \frac{k}{m}$,*

$$\lim_{n\to\infty} \frac{\log|\tilde{B}_{m,k}(0^n, n(1-q))|}{n\left(\log(m) - D\left(q||\frac{k}{m}\right)\right)} = 1.$$

Notice that $|\tilde{B}_{m,k}(y, n(1-q))|$ does not depend on y as long as $y \in [m]^n$.

Proof. Let $\tilde{B}'_{m,k}(y, r)$ be the set of strings where *exactly* r positions are from the unlikely set:

$$\tilde{B}'_{m,k}(y, r) = \{x \in [m]^n : |\{i : y_i \in \{x_i, x_i + 1, \ldots x_i + k - 1\} \mod m\}| = n - r\}.$$

Clearly

$$\tilde{B}_{m,k}(0^n, n(1-q)) = \bigcup_{r=0}^{\lfloor n(1-q)\rfloor} \tilde{B}'_{m,k}(0^n, r),$$

so

$$\left|\tilde{B}_{m,k}(0^n, n(1-q))\right| = \sum_{r=0}^{\lfloor n(1-q)\rfloor} \left|\tilde{B}'_{m,k}(0^n, r)\right|. \tag{4.15}$$

To specify an element $x \in \tilde{B}'_{m,k}(0^n, r)$ you need to specify the r positions i where x_i is not the likely set, for each of these r you need to specify which of the $n - k$ possible values x_i takes, and for each of the $n - r$ other positions i, you need to specify which of the k possible values x_i takes. This gives us

$$|\tilde{B}'_{m,k}(0^n, r)| = \binom{n}{r}(m-k)^r k^{n-r}.$$

Using Stirling's formula, $\log(n!) = n\log(n) - n\log(e) + O(\log(n))$ we get

$$\begin{aligned}
&\log|\tilde{B}'_{m,k}(0^n, r)| \\
&= \log(n!) - \log(r!) - \log((n-r)!) + r\log(m-k) + (n-r)\log(k) \\
&= n\log(n) - r\log(r) - (n-r)\log(n-r) - \log(e)(n-r-(n-r)) \\
&\quad + O(\log(n)) + r\log(m-k) + (n-r)\log(k) \\
&= -r\log\left(\frac{r}{n}\right) - (n-r)\log\left(\frac{n-r}{n}\right) \\
&\quad + O(\log(n)) + r\log\left(\frac{m-k}{m}\right) + (n-r)\log\left(\frac{k}{m}\right) + n\log(m) \\
&= n\left(\log(m) + \frac{r}{n}\log\left(\frac{(m-k)n}{mr}\right) + \frac{n-r}{n}\log\left(\frac{kn}{m(n-r)}\right)\right) + O(\log(n)) \\
&= n\left(\log(m) - D\left(\frac{n-r}{n}\middle\|\frac{k}{m}\right)\right) + O(\log(n)).
\end{aligned}$$

If we set $r = \lfloor n(1-q)\rfloor$ we see that $\frac{n-r}{n} = \frac{n-\lfloor n(1-q)\rfloor}{n} = \frac{\lceil nq\rceil}{n} \le q + \frac{1}{n}$. As $D\left(\cdot\|\frac{k}{m}\right)$ has bounded derivative around q, we get that $nD\left(\frac{n-r}{n}\middle\|\frac{k}{m}\right) = nD\left(q\middle\|\frac{k}{m}\right) + O(1)$. This shows that $\lim_{n\to\infty} \frac{\log|\tilde{B}'_{m,k}(0^n,\lfloor n(1-q)\rfloor)|}{n\left(\log(m) - D\left(q\|\frac{k}{m}\right)\right)} = 1$ and as $\tilde{B}'_{m,k}(0^n, \lfloor n(1-q)\rfloor) \subset \tilde{B}_{m,k}(0^n, n(1-q))$ we have

$$\liminf_{n\to\infty} \frac{\log|\tilde{B}_{m,k}(0^n, n(1-q))|}{n\left(\log(m) - D\left(q\|\frac{k}{m}\right)\right)} \ge 1.$$

To show the upper bound we observe that

$$\begin{aligned}
\frac{\tilde{B}'_{m,k}(0^n, r)}{\tilde{B}'_{m,k}(0^n, r-1)} &= \frac{\binom{n}{r}(m-k)^r k^{n-r}}{\binom{n}{r-1}(m-k)^{r-1}k^{n-r+1}} \\
&= \frac{(n-r+1)(m-k)}{rk}.
\end{aligned}$$

For r with $n - r \ge qn$ we have $\frac{n-r}{n} \ge q > \frac{k}{m}$, so $\frac{m-k}{k} > \frac{1}{q} - 1 \ge \frac{r}{n-r} > \frac{r}{n-r+1}$, showing that the above ratio is > 1. Thus, the biggest contribution to $\tilde{B}_{m,k}(0^n, n(1-q))$ in

equation (4.15) comes from $\tilde{B}'_{m,k}(0^n, \lfloor n(1-q) \rfloor)$. Hence,

$$\begin{aligned}\limsup_{n\to\infty} \frac{\log|\tilde{B}_{m,k}(0^n, n(1-q))|}{n\left(\log(m) - D\left(q||\frac{k}{m}\right)\right)} \leq & \limsup_{n\to\infty} \frac{\log|\tilde{B}'_{m,k}(0^n, \lfloor n(1-q) \rfloor)| + \log(n)}{n\left(\log(m) - D\left(q||\frac{k}{m}\right)\right)} \\ =& 1.\end{aligned}$$

□

Until now, we have thought of a code $\mathfrak{c}$ as a subset of some $[m]^n$. However, in the proof of the following theorem we need to take a random code, and it will be more useful to define a code to be a list $\mathfrak{c}^1, \ldots, \mathfrak{c}^{2^{\lceil Rn \rceil}}$ of codewords. We will allow elements in the list to be identical, so the codewords do not form a set but a multi-set. However, this distinction between lists and sets is not important for us, and we will sometimes think of codes as lists, sometimes as functions $i \mapsto \mathfrak{c}^i$ and sometimes as (multi-)sets.

Theorem 4.12. *If $D(q||\frac{k}{m}) - R > \frac{\log(m)}{\mathcal{L}+1}$, then for sufficiently large n there exists a code $\mathfrak{c} \subset [m]^n$ of size $2^{\lceil Rn \rceil}$ such that for all possible outputs $y \in [m]^k$ we have $|\mathfrak{c} \cap \tilde{B}_{m,k}(y, n(1-q))| \leq \mathcal{L}$.*

Proof. We prove this using the probabilistic method, in particular the proof is not constructive. Let $\mathfrak{C}$ be the random variable where each instance $\mathfrak{c}$ of $\mathfrak{C}$ is a list of $2^{\lceil nR \rceil}$ elements from $[m]^n$, and each of these elements is chosen uniformly and independently. We use $\mathfrak{C}^i$ to denote the i'th element in $\mathfrak{C}$. The probability that for some specific $y \in [m]^n$ the $\mathcal{L}+1$ elements $\mathfrak{C}^{i_0}, \ldots \mathfrak{C}^{i_\mathcal{L}}$ all lie in $\tilde{B}_{m,k}(y, n(1-q))$ is $\left(\frac{|\tilde{B}_{m,k}(y,n(1-q))|}{m^n}\right)^{\mathcal{L}+1}$. There are m^n choices of y and less that $2^{\lceil nR \rceil(\mathcal{L}+1)}$ choices of $i_0, \ldots i_l$. By the union bound, the probability P_e that there exists $y, i_0, \ldots, i_\mathcal{L}$ with $\mathfrak{C}^{i_0}, \ldots \mathfrak{C}^{i_\mathcal{L}} \in \tilde{B}_{m,k}(y, n(1-q))$ is less than

$$\left(\frac{|\tilde{B}_{m,k}(y, n(1-q))|}{m^n}\right)^{\mathcal{L}+1} m^n 2^{\lceil nR \rceil(\mathcal{L}+1)}.$$

By taking logarithm and using Proposition 4.11 we get

$$\begin{aligned}\log(P_e) \leq & (\mathcal{L}+1)\left(\log(|\tilde{B}_{m,k}(y,n(1-q))|)+\lceil nR\rceil\right)-n\mathcal{L}\log(m)\\ \leq & (1+o(1))(\mathcal{L}+1)n\left(\log(m)-D\left(q\middle|\middle|\frac{k}{m}\right)+R\right)-n\mathcal{L}\log(m)\\ \leq & (1+o(1))n\left(\log(m)-(\mathcal{L}+1)\left(D\left(q\middle|\middle|\frac{k}{m}\right)-R\right)\right).\end{aligned}$$

By assumption, $\log(m)-(\mathcal{L}+1)(D(q||\frac{k}{m})-R)<0$, so this tends to $-\infty$. Thus, P_e tends to 0. That is, with high probability $\mathfrak{C}$ does not have any $\tilde{B}_{m,k}(y,n(1-q))$ with more than $\mathcal{L}$ elements. In particular, we can choose a $\mathfrak{c}$ such that no $\mathfrak{c}\cap\tilde{B}_{m,k}(y,n(1-q))$ has more than $\mathcal{L}$ elements.

Here $\mathfrak{c}$ might contain the same codeword more than once, but as no $\tilde{B}_{m,k}(y,n(1-q))$ contains more that $\mathcal{L}$ elements from $\mathfrak{c}$, it contains each element at most $\mathcal{L}$ times. If we do not want to allow a code to contain the same element more than once, we simply get a code for a slightly higher rate $R'>R$ to get a code $\mathfrak{c}'$ and then let $\mathfrak{c}$ be the set of all codewords that occur in $\mathfrak{c}'$ (now without multiplicity). □

The above theorem says that even if the channel's errors are chosen by an adversary who will know our code, there is a code $\mathfrak{c}$ that ensures that we can decode to get a list of $\mathcal{L}$ elements containing x. But even if the code is designed to work against an adversary who knows the code, it is useful to know what happens when there is no such adversary. To get the most general result, we will assume that there is an adversary who completely controls the channel, but does not know our code. Here the adversary is not meant to model an actual intelligent adversary - such an adversary is modelled by the censors - but instead to model the worst possible thing that could happen if our model is wrong - if non-leakers do not follow π, if the fraction of leakers is not b_l or if there is some pattern in who becomes leakers - but no one is actively trying to make the leakage fail. For this purpose, it is natural to assume that the adversary does not know the code, even if we are not trying to keep the code a secret.

We would like to ensure that if we input $x\in[m]^n$ to some channel controlled by an adversary who does not know our code $\mathfrak{c}$ and we get output $y\in[m]^n$, then there is only small probability that there exists $x'\neq x$ with $x'\in\mathfrak{c}\cap\tilde{B}_{m,k}(y,n(1-q))$. That

is, we do not require that the output list will contain the input x, but we want to ensure that there is only small probability that it contains anything else. Of course for any fixed code $\mathfrak{c}$ the adversary could choose to always output some fixed $y \in \mathfrak{c}$, so we need to use a randomly generated code. The only information the adversary learns about $\mathfrak{c}$ is one input value $x = \mathfrak{c}^j$. We still want any code we use to satisfy $|\mathfrak{c} \cap \tilde{B}_{m,k}(y, n(1-q))| \leq \mathcal{L}$ as in Theorem 4.12.

Theorem 4.13. *Let q, k, m, R be as in Theorem 4.12 and let $\epsilon \in (0,1)$ be given. Then for sufficiently large n there exists a random variable $\mathfrak{C}$ which as values takes codes $\mathfrak{c}$ satisfying the requirements of Theorem 4.12 such that for each $j \in \left[2^{\lceil Rn \rceil}\right]$ if Y is a random variable independent from all $\mathfrak{C}^i$ given $\mathfrak{C}^j$ then*

$$\Pr(\exists i \neq j : \mathfrak{C}^i \in \tilde{B}_{m,k}(Y, n(1-q))) \leq \epsilon$$

where the probability is over $\mathfrak{C}$ and Y.

Proof. First we choose a random variable $\mathfrak{C}'$ where each $\mathfrak{C}'^i$ is chosen randomly and independently as in the proof of Theorem 4.12. Let $E = 1$ if $\forall y : |\mathfrak{C}' \cap \tilde{B}_{m,k}(y, n(1-q))| \leq \mathcal{L}$ and $E = 0$ otherwise. The proof of Theorem 4.12 shows that for sufficiently large n, we have $\Pr(E = 1) \geq 1 - \epsilon$.

As each $\mathfrak{C}'^i$ with $i \neq j$ is chosen uniformly from $[m]^n$ and independent of Y we get

$$\Pr(\mathfrak{C}'^i \in \tilde{B}_{m,k}(Y, n(1-q))) = \frac{|\tilde{B}_{m,k}(y, n(1-q))|}{m^n},$$

for each $i \neq j$ and hence

$$\Pr(\exists i \neq j : \mathfrak{C}'^i \in \tilde{B}_{m,k}(Y, n(1-q))) \leq \frac{|\tilde{B}_{m,k}(y, n(1-q))| \cdot (2^{nR} - 1)}{m^n}.$$

Using Proposition 4.11 we get

$$\begin{aligned}
&\log(\Pr(\exists i \neq j : \mathfrak{C}'^i \in \tilde{B}_{m,k}(Y, n(1-q)))) \\
&\quad \leq \log(|\tilde{B}_{m,k}(Y, n(1-q))|) + n(R - \log(m)) \\
&\quad = n(1+o(n))\left(\log(m) - D\left(q\middle|\middle|\frac{k}{m}\right) + R - \log(m)\right) \\
&\quad = n(1+o(n))\left(R - D\left(q\middle|\middle|\frac{k}{m}\right)\right).
\end{aligned}$$

By assumption $R < D(q||\frac{k}{m})$, so this expression tends to $-\infty$ and hence for sufficiently large n the probability is at most $\epsilon(1-\epsilon)$. We now let $\mathfrak{C} = \mathfrak{C}'|_{E=1}$. This ensures

$$\Pr(\exists i \neq j : \mathfrak{C}^i \in \tilde{B}_{m,k}(Y, n(1-q))) \leq \frac{\epsilon(1-\epsilon)}{1-\epsilon} \leq \epsilon. \qquad \square$$

4.2 Minimal list size

For specific values of $b = (b_l, b_c, b_m)$ we want to measure how much information a censor/leaker protocol can reveal. However, just defining this is more complicated than defining the capacity of a channel, because we have two parameters: the rate R and the list size $\mathcal{L}$. As we will see in Theorem 4.26, for fixed b the closure of the set of combinations $(\mathcal{L}, R)$ with $R > 0$ that can be achieved is a product set, and hence can be described by two numbers: the maximal rate and the minimal list size. However, in order to prove Theorem 4.26 we first need a definition that does not assume anything about the set of achievable $(\mathcal{L}, R)$. Recall from Definition 4.1 that a $(n, h, \mathcal{L}, \epsilon, b)$-protocol is a b-protocol where the number of people is n, the secret is chosen from $[2^{\lceil h \rceil}]$ and given transcript an observer is able to compute a list of length $\mathcal{L}$ that for a random transcript contains the secret with probability at least $1 - \epsilon$.

Definition 4.7. The *rate* of an $(n, h, \mathcal{L}, \epsilon, b)$-protocol π is $\frac{h}{n}$, and $\mathcal{L}$ is its *list size*.

For $\mathcal{L} \in \mathbb{N}$ and $R > 0$ we say that $(\mathcal{L}, R)$ is b-*achievable* if for all $\epsilon > 0$ and all n_0 there exists $n > n_0$ and an $(n, nR, \mathcal{L}, \epsilon, b)$-protocol.

The $(\mathcal{L}, b)$-*capacity* is the supremum over all rates R such that $(\mathcal{L}, R)$ is b-achievable.

The *b-capacity* is the supremum over all rates R such that $(\mathcal{L}_R, R)$ is b-achievable for some $\mathcal{L}_R$.

The *b-minimal list size* is the smallest $\mathcal{L}$ such that for all $h, \epsilon > 0$ there is an $(n, h, \mathcal{L}, \epsilon, b)$-protocol for some n.

In this section we will focus on minimal list size and ignore the rate. First we give a lower bound and later a matching upper bound on the minimal list size.

Theorem 4.14. *The b-minimal list size is at least* $\left\lfloor \frac{b_c}{b_l} \right\rfloor + 1$.

Proof. Let b be given, and assume that we have an $(n, h, \mathcal{L}, \epsilon, b)$-protocol π where $\mathcal{L} = \left\lfloor \frac{b_c}{b_l} \right\rfloor$, $h > \log(\mathcal{L} + 1)$ and $\epsilon > 0$. We now define a censor protocol σ' where the censors have access to shared randomness, which is unknown to G and the leakers. This shared randomness is not allowed according to our definition of a censor protocol, but a protocol with shared randomness can be considered to be a probability distribution over protocols without randomness. Now if the censors can ensure $\Pr(X \in G(T)) \leq 1 - \epsilon$ using a random protocol, there must exists a protocol σ without shared randomness that ensures $\Pr(X \in G(T)) \leq 1 - \epsilon$. Thus, we can allow the censors to have shared randomness.

Each censor chooses a number from $\{0, 1, \ldots, \mathcal{L}\}$ independently at random. Any number $i > 0$ is chosen with probability $\frac{b_l}{b_c}$, and otherwise 0 is chosen. This way, the probability that a person is censor and choose a particular $i > 0$, is the same as the probability that he is leaker. Now the shared randomness is used to choose $x_1, \ldots, x_{\mathcal{L}} \in \mathcal{X}$, uniformly under the condition that they are all different and different from the true value x of X. Then any censor who chose $i > 0$ will follow protocol π as if he was leaker and the true value of X is x_i. The censors who chose 0 behaves as if they were neutral. Given that the $\mathcal{L}+1$ values $x, x_1, \ldots, x_{\mathcal{L}}$ form the set $\mathcal{X}' = \{x, x_1, \ldots, x_{\mathcal{L}}\}$ the transcript T is independent of which of the $\mathcal{L}+1$ elements in $\mathcal{X}'$ that is the true value x. Furthermore, given that $\{x, x_1, \ldots, x_{\mathcal{L}}\} = \mathcal{X}'$, each of the $\mathcal{L}+1$ values in $\mathcal{X}'$ are equally likely to be the true value x. As $G(T)$ contains at most $\mathcal{L}$ elements, and $x, x_1, \ldots x_{\mathcal{L}}$ are $\mathcal{L}+1$ different elements, $\Pr(X \in G(T)) \leq \frac{\mathcal{L}}{\mathcal{L}+1}$. Thus, $\epsilon \geq \frac{1}{\mathcal{L}+1}$, so the b-minimal list size is $> \left\lfloor \frac{b_c}{b_l} \right\rfloor$. □

Notice that the above impossibility result holds much more generally. All we need is that the censors can pretend to be leakers. In particular, as long as each

leaker do not know who the other leakers are, each censor do not need to know who the other censors are, they only need to know $x_1, \dots x_{\mathcal{L}}$. Similarly, the censors will not need any more computational power than the leakers.

In the above proof, the censor protocol σ without shared randomness, will have to depend on G. In other words, if the recipient knows the censors' strategy, including $x_1, \dots, x_l$, the recipient might still be able to find x. The easiest way to avoid this, is to modify the model to allow the censor to use shared randomness. This would arguably make the model more natural.

Alternatively, we can use a different way of getting a protocol σ without shared randomness from a protocol σ' with shared randomness: in σ the censors pretend to follow σ'. Each time a censor is about to send a message he computes the distribution of the shared randomness given previous messages, samples from this distribution and send his next message accordingly. This way, following σ gives the same distribution as following σ'. The method works in our model, but not as generally as the above proof. This method uses the assumptions that messages are sent one by one and that the censors know who each other are. Furthermore, in general the censors would not be able to follow this strategy if they have only polynomial computational power.

Theorem 4.15. *If $b_l + b_c \geq b_m$ the b-minimal list size is ∞.*

Proof. When $b_l + b_c \geq b_m$ we have $r(b) = \frac{b_m(1-b_l-b_c)}{b_l(1-b_m)} \leq \frac{b_m}{b_l}$. Thus,

$$\begin{aligned} b_l' &= \max\left(b_l - \frac{b_c}{r(b)-1}, 0\right) \\ &\leq \max\left(b_l - \frac{b_c}{\frac{b_m}{b_l}-1}, 0\right) \\ &= \max\left(b_l - \frac{b_c}{\frac{b_m-b_l}{b_l}}, 0\right) \\ &\leq \max\left(b_l - \frac{b_c}{\frac{b_c}{b_l}}, 0\right) \\ &= 0. \end{aligned}$$

Here the first inequality follows from $r(b) \leq \frac{b_m}{b_l}$ and the second from $b_l + b_c \geq b_m$.

By Theorem 4.1 for any b-leaker protocol π there is a censor protocol σ, such that the distribution of (X, T) is the same when $L \sim b$ and $L \sim (0, 0, b'_m)$. But for $L \sim (0, 0, b'_m)$ everyone are sending messages independent of X, so X and T are independent. Thus, it is possible for the censors to follow a strategy that makes X and T independent, so any $(n, h, \mathcal{L}, \epsilon, b)$-protocol with $h \geq \log(\mathcal{L}) + 1$ must have $\epsilon \geq \frac{1}{2}$. □

The following theorem shows an upper bound on the b-minimal list size. The proof is by contradiction: if you can get down to a list size of $\mathcal{L} > \left\lfloor \frac{b_c}{b_l} \right\rfloor + 1$ you can get down to a list size of $\mathcal{L} - 1$. This part of the proof is constructive, and could be combined with induction to get an extremely inefficient constructive proof of the theorem. It is possible to construct more efficient protocols that achieve a minimal list size, but in the proof we prioritized simplicity of the proof over efficiency of the resulting protocol.

Theorem 4.16. *If $b_l + b_c < b_m$ the b-minimal list size is at most $\left\lfloor \frac{b_c}{b_l} \right\rfloor + 1$.*

Proof. Let b be given. Fix a value h, and let $\mathcal{L}$ be the smallest value such that for any $\epsilon > 0$ there is an n' and an $(n', h, \mathcal{L}, \epsilon, b)$-protocol (there exists a number $\mathcal{L}$ that satisfy this, namely $2^{\lceil h \rceil}$, so there must be a smallest integer satisfying it). Assume for contradiction that $\mathcal{L} > \left\lfloor \frac{b_c}{b_l} \right\rfloor + 1$. As $\mathcal{L}$ is an integer, this implies $\mathcal{L} > \frac{b_c}{b_l} + 1$ and hence $(\mathcal{L} - 1)b_l > b_c$.

Let $\epsilon > 0$ be given. We want to show that there exists an $(n, h, \mathcal{L}-1, \epsilon, b)$-protocol for some value n. By assumption there exists an $(n', h, \mathcal{L}, \epsilon/2, b)$-protocol π'. Now π starts by letting the first n' players simulate π', so we get a set $G'(T')$ of size $\mathcal{L}$. Then we let each of the $n-n'$ remaining players send a message from $G'(T')$. Each of the neutral players chose their message uniformly and independently. If $x \notin G'(T')$ the leakers also chose their message uniformly, otherwise, if $\mathcal{L} \leq \frac{b_m(1-b_c-b_l)}{b_l(1-b_m)}$, all the leakers send the message x. If $\mathcal{L} > \frac{b_m(1-b_c-b_l)}{b_l(1-b_m)}$, each leaker sends the message x with probability $\frac{b_m(1-b_c-b_l)}{\mathcal{L}b_l(1-b_m)}$, and otherwise he chooses a message $G'(T') \setminus \{x\}$ uniformly at random. For a more formal definition of this protocol see Figure 4.3.

Let E_j denote the events "player j is either leaker or neutral and player j sent the message x". If $\mathcal{L} \leq \frac{b_m(1-b_c-b_l)}{b_l(1-b_m)}$ then given $x \in G'(T')$ the event E_j happens with

Parameters:

n : number of players

$b = (b_l, b_c, b_m)$: probability of being leaker respectively censor and threshold of reasonable doubt

h : number of bits to leak

$\mathcal{L} > \left\lfloor \frac{b_c}{b_l} \right\rfloor + 1$: list size known to be achievable for this value of h

ϵ : acceptable probability of error

n' : number of players in previous protocol

π' : an $(n', h, \mathcal{L}, \epsilon/2, b)$-protocol

Input distribution: $X, L_1, \ldots, L_n$ are independently distributed, X is uniformly distributed on $\{0,1\}^h$ and for each i: $\Pr(L_i = 1) = b_l$ and $\Pr(L_i = -1) = b_c$. Each PLR_i learns L_i and if $|L_i| = 1$ she also learns X and if $L_i = -1$ she also learns $L_- = (\min(L_1, 0), \ldots, \min(L_n, 0))$. She will follow the protocol given here if $L_i \neq -1$ and otherwise she might follow any censor protocol σ.

Protocol:

1. Let $\text{PLR}_1, \ldots, \text{PLR}_{n'}$ follow π' to reveal X. Let $G'(T')$ denote the output: an list of length $\mathcal{L}$ that will typically contain x.
2. For each i from $n' + 1$ to n
3. If PLR_i is a non-leaker or $x \notin G'(T')$ choose T_i uniformly from $G'(T')$, otherwise
4. Choose T_i to be x with probability $\min\left(1, \frac{b_m(1-b_c-b_l)}{\mathcal{L}b_l(1-b_m)}\right)$ and otherwise choose uniformly from $G'(T') \setminus \{x\}$
5. PLR_i sends T_i

Figure 4.3: Protocol from proof of Theorem 4.16.

probability

$$b_l + \frac{1 - b_l - b_c}{\mathcal{L}} = \frac{1 + (\mathcal{L} - 1)b_l - b_c}{\mathcal{L}} > \frac{1}{\mathcal{L}}.$$

Here the inequality follows from $(\mathcal{L}-1)b_l > b_c$. If $\mathcal{L} > \frac{b_m(1-b_c-b_l)}{b_l(1-b_m)}$ then E_j happens with probability

$$\begin{aligned} b_l\frac{b_m(1-b_c-b_l)}{\mathcal{L}b_l(1-b_m)} + \frac{1-b_l-b_c}{\mathcal{L}} =& \frac{1}{\mathcal{L}}\left(1+\frac{b_m(1-b_c-b_l)}{1-b_m}-b_l-b_c\right) \\ >& \frac{1}{\mathcal{L}}\left(1+\frac{b_m(1-b_m)}{1-b_m}-b_l-b_c\right) \\ =& \frac{1}{\mathcal{L}}\left(1+b_m-b_l-b_c\right) \\ >& \frac{1}{\mathcal{L}}. \end{aligned}$$

Here each of the two inequalities follows from $b_m > b_l + b_c$. We see that the E_j's are independent. Hence, for sufficiently large value of n, the probability that more than $\frac{n-n'}{\mathcal{L}}$ of the $n-n'$ events E_j happens is at least $1-\epsilon/2$. In particular, the probability that more than $\frac{n-n'}{\mathcal{L}}$ of the $n-n'$ messages is i is at least $1-\epsilon/2$. Now we take $G(T)$ to be the set of all messages sent more than $\frac{n-n'}{\mathcal{L}}$ times in the last $n-n'$ messages. Clearly, this set contains at most $\mathcal{L}-1$ elements. Furthermore, the only ways we can have $x \notin G(T)$ are if $x \notin G'(T')$ or if $x \in G'(T')$ but $x \notin G(T)$. For each $x \in X$ each of these happens with probability at most $\frac{\epsilon}{2}$, so by the union bound the error probability is at most ϵ. If $\mathcal{L} \leq \frac{b_m(1-b_c-b_l)}{b_l(1-b_m)}$ the likelihood ratio for each of the $n-n'$ players is $\frac{1}{\mathcal{L}^{-1}} = \mathcal{L} \leq \frac{b_m(1-b_c-b_l)}{b_l(1-b_m)} = r(b)$, and if $\mathcal{L} > \frac{b_m(1-b_c-b_l)}{b_l(1-b_m)}$ their likelihood ratio is $\frac{\frac{b_m(1-b_c-b_l)}{\mathcal{L}b_l(1-b_m)}}{\mathcal{L}} = \frac{b_m(1-b_c-b_l)}{b_l(1-b_m)} = r(b)$. As the n' first players only sent the messages from a b-leaker protocol, by Proposition 4.4 and Proposition 4.7 they most have likelihood ratio at most $r(b)$. Thus, all players in π have likelihood ratio at most $r(b)$, so Proposition 4.7 shows that π preserves reasonable doubt. Thus, π is an $(n, h, \mathcal{L}, \epsilon, b)$-protocol. □

Corollary 4.17. *If $b_l + b_c < b_m$ the b-minimal list size is $\left\lfloor \frac{b_c}{b_l} \right\rfloor + 1$.*

Proof. Follows immediately from the above theorems. □

4.3 Capacity

In the previous section we focused on optimizing one parameter, the list size, without thinking about how many people were needed. In this section we do the opposite:

we ask how many bits can we leak per person, when the list size is only required to be bounded. In the next section we will show that asymptotically we can get the optimal value of both parameters at the same time.

Theorem 4.18. *Let b be given. If $b_m \leq b_c + b_l$ no $(\mathcal{L}, R)$ with $R > 0$ is b-achievable.*

Proof. This follows from Theorem 4.15. □

In order to show an upper bound on the capacity for b, we will need a generalization of Fano's inequality.

Lemma 4.19 (Generalisation of Fano's inequality)**.** *Let X be a random variable taking values in $\mathcal{X}$ and let Y be a random variable that takes subsets of $\mathcal{X}$ of size at most $\mathcal{L}$ as values. The probability $P_e = \Pr(X \notin Y)$ satisfies*

$$P_e \geq \frac{H(X|Y) - 1 - \log(\mathcal{L})}{\log(|\mathcal{X}|)}.$$

Proof. Let X, Y and P_e be as above. We have a joint distribution of (X, Y), and we use this to define a joint distribution of $(X, Y, (Z_1, Z_2))$: If $X \in Y$ we set $Z_1 = 1$ otherwise $Z_1 = 0$. If $X \in Y$ we let Z_2 be a number such that X is the Z_2'th smallest element in Y (according to some fixed order on $\mathcal{X}$) otherwise we let Z_2 be the number such that X is the Z_2th smallest element in $\mathcal{X}$. We have $H(Z_2|Z_1 = 1) \leq \log(\mathcal{L})$, $H(Z_2|Z_1 = 0) \leq \log(|\mathcal{X}|)$ and hence

$$\begin{aligned} H(Z_2|Z_1) &= \Pr(Z_1 = 1)H(Z_2|Z_1 = 1) + \Pr(Z_1 = 0)H(Z_2|Z_1 = 0) \\ &\leq \log(\mathcal{L}) + P_e \log(|\mathcal{X}|). \end{aligned}$$

Furthermore, $H(Z_1) \leq 1$ and X can be written as a function of Y and (Z_1, Z_2) so we have

$$H(X|Y) \leq H(Z_1, Z_2) = H(Z_1) + H(Z_2|Z_1) \leq 1 + \log(\mathcal{L}) + P_e \log(|\mathcal{X}|).$$

Rewriting this inequality gives us the lemma. □

We now show an upper bound on how much information the leakers can reveal. Such a bound correspond to a good censor protocol. We use the protocol from Theorem 4.1, so the proof is constructive.

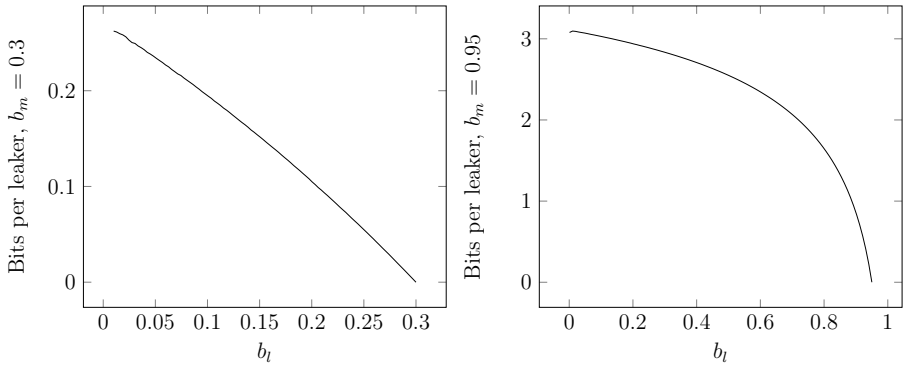

Figure 4.4: These two plots show the number of bits that the leakers can leak per expected leaker when there are no censors, that is, the $(b_l, 0, b_m)$-capacity divided by b_l. The left plot is for $b_m = 0.3$ and the right is for $b_m = 0.95$

Proposition 4.20. *Let b be given. If $b_m > b_c + b_l$ and π is a b-leaker protocol for n players, there is a censor protocol σ, such that when $L \sim b$ and leakers and neutrals are following π and censors are following σ, we have*

$$I(X;T) \leq nD\left(\frac{b_l}{b_m}\middle|\middle|\frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}\right)$$

Proof. Let b with $b_m > b_c + b_l$ be given, and let b' be as in Defintion 4.2. Then by Theorem 4.1 π is also a b'-protocol, and there is a censor protocol σ such that the distribution of (X,T) when following π and σ on $L \sim b$ is the same as following π on distribution $L \sim b'$ (here no censor protocol is needed as $b'_c = 0$). A simple computation shows that if $b_m = b_l + b_c$ then $b_l = \frac{b_c}{r(b)-1}$. As r is increasing in b_m this implies that for $b_m > b_l + b_c$ we have $b'_l = \max\left(b_l - \frac{b_c}{r(b)-1}, 0\right) > 0$, and similarly $b'_m > 0$. Now

$$\begin{aligned} I(X;T) \leq & nD\left(\frac{b'_l}{b'_m}\middle|\middle|\frac{b'_l(1-b'_m)}{b'_m(1-b'_l)}\right) \\ = & nD\left(\frac{b_l}{b_m}\middle|\middle|\frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}\right). \end{aligned}$$

The inequality follows from Theorem 3.5. To get the equality we use the definition of b' to simplify the first input to $D(\cdot||\cdot)$ and $r(b) = r(b')$ which follows from Proposition 4.2 to simplify the second input. □

Now we can prove an upper bound on the capacity. This proof is similar to that of Proposition 3.8 and Proposition 3.11.

Theorem 4.21. *Let b be given. If $b_m > b_c + b_l$ then the b-capacity is at most $D\left(\frac{b_l}{b_m}\middle\|\frac{b_l(1-b_m)}{b_m(1-b_c-b_l)}\right)$.*

Proof. Let $b_m > b_c + b_l$ be given and assume for contradiction that some $(\mathcal{L}, R)$ with $R > D\left(\frac{b_l}{b_m}\middle\|\frac{b_l(1-b_m)}{b_m(1-b_c-b_l)}\right)$ is b-achievable. Then for arbitrarily large n and arbitrarily small ϵ we can find an $(n, nR, \mathcal{L}, \epsilon, b)$-protocol π. Define

$$\delta = R - D\left(\frac{b_l}{b_m}\middle\|\frac{b_l(1-b_m)}{b_m(1-b_c-b_l)}\right) > 0.$$

By Proposition 4.20 there is a censor protocol that ensures

$$I(X;T) \leq nD\left(\frac{b_l}{b_m}\middle\|\frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}\right).$$

By the definition of a $(n, nR, \mathcal{L}, \epsilon, b)$-protocol, even if the censors follow this protocol we should have $\Pr(X \in G(T)) \geq 1 - \epsilon$. Using the data processing inequality (1.2) for mutual information we get

$$\begin{aligned} H(X|G(T)) &= H(X) - I(X;G(T)) \\ &\geq H(X) - I(X;T) \\ &\geq n\left(R - D\left(\frac{b_l}{b_m}\middle\|\frac{b_l(1-b_m)}{b_m(1-b_c-b_l)}\right)\right) \\ &= n\delta. \end{aligned}$$

Now Lemma 4.19 gives us

$$\begin{aligned} \epsilon &\geq \frac{H(X|G(T)) - 1 - \log(\mathcal{L})}{\log(|\mathcal{X}|)} \\ &\geq \frac{n\delta - 1 - \log(\mathcal{L})}{nR}. \end{aligned}$$

For fixed $(\mathcal{L}, R)$ and suffciently large n this is bounded away from 0. Hence, such $(\mathcal{L}, R)$ is not b-achievable. □

Next we want to find matching lower bounds on the capacity, that is, show that there exists good leaker protocols. The proofs will not be constructive.

Definition 4.8. For a code $\mathfrak{c} \subset [m]^n$ let $\pi_{\mathfrak{c}}$ be the protocol where each player sends one message: player i sends her message in round i. If player i is neutral, she sends a message from $[m]$ chosen uniformly at random. If player i is a leaker and $X = x$, she chooses a message from $A_i = \{\mathfrak{c}_i(x), \mathfrak{c}_i(x)+1, \ldots, \mathfrak{c}_i(x)+k-1\} \bmod m$ uniformly at random, where $\mathfrak{c}_i(x)$ denotes the i'th element in the codeword for x. The protocol is also defined in Figure 4.5.

Parameters:

n : number of players

$m > k$: natural numbers

$\mathfrak{c}$: an error correcting code

Input distribution: $X, L_1, \ldots, L_n$ are independently distributed, X is uniformly distributed on $\{0,1\}^h$ are for each i: $\Pr(L_i = 1) = b_l$ and $\Pr(L_i = -1) = b_c$. Each PLR_i learns L_i and if $|L_i| = 1$ she also learns X, and if $L_i = -1$ she also learns $L_- = (\min(L_1, 0), \ldots, \min(L_n, 0))$. She will follow the protocol given here if $L_i \neq -1$ and otherwise she might follow any censor protocol σ.

Protocol:

1. For i from 1 to n:
2. If $L_i = 1$ then $T_i \leftarrow \mathfrak{c}(x)_i + [k] - 1 \mod m$, if $L_i = 0$, then $T_i \leftarrow [m]$
3. Player i sends T_i

Figure 4.5: Protocol $\pi_{\mathfrak{c}}$ defined in Definition 4.8

Theorem 4.22. *Let b satisfy $b_m > b_c + b_l$. Then for any $R < D\left(\frac{b_l}{b_m} \middle|\middle| \frac{b_l(1-b_m)}{b_m(1-b_c-b_l)}\right)$, there is an $\mathcal{L}$ such that $(\mathcal{L}, R)$ is achievable.*

Proof. Let b and R satisfy the assumptions in the statement of the theorem. As $D(\cdot||\cdot)$ is continuous, we can choose integers $k < m$, such that $\frac{b_l(1-b_m)}{b_m(1-b_c-b_l)} \leq \frac{k}{m} < \frac{b_l}{b_m}$ and $R < D\left(\frac{b_l}{b_m} \middle|\middle| \frac{k}{m}\right)$. For the same reason, we can then choose q such that $\frac{k}{m} < q <$

$\frac{b_l}{b_m}$ and $R < D\left(q\middle|\middle|\frac{k}{m}\right)$. Let $\mathcal{L}$ be an integer such that $D\left(q\middle|\middle|\frac{k}{m}\right) - R > \frac{\log(m)}{\mathcal{L}+1}$. From Theorem 4.12 we know that for sufficiently large n there is a code $\mathfrak{c} \subset [m]^n$ with $\lceil 2^{nR} \rceil$ elements and such that for all $y \in [m]^n$ we have $|\mathfrak{c} \cap \tilde{B}_{m,k}(y, n(1-q))| \leq \mathcal{L}$. We now let the players use $\pi_{\mathfrak{c}}$.

Let T_i be the message sent by player i, and E_i be the event "player i is either leaker or neutral and player i sent a message from A_i". We have

$$\begin{aligned}
\Pr(E_i) =& b_l \cdot \Pr(T_i \in A_i | L_i = 1) + (1 - b_c - b_l) \cdot \Pr(T_i \in A_i | L_i = 0) \\
=& b_l + (1 - b_c - b_l)\frac{k}{m} \\
\geq& b_l + (1 - b_c - b_l)\frac{b_l(1 - b_m)}{b_m(1 - b_c - b_l)} \\
\geq& \frac{b_l}{b_m} \\
>& q.
\end{aligned}$$

We now define $G(T) = \mathfrak{c}^{-1}(\mathfrak{c} \cap \tilde{B}_{m,k}(T, n(1-q)))$, that is, $G(T)$ gives the set of messages whose codeword belongs to $\mathfrak{c} \cap \tilde{B}_{m,k}(T, n(1-q))$. By assumption about $\mathfrak{c}$ this set will have size at most $\mathcal{L}$ for all possible transcripts T.

As the E_i's are independent given X and each happens with probability $\frac{b_l}{b_m} > q$, the probability that at least nq of them happen is at least $1 - \epsilon$ for sufficiently large n. In particular, the probability that we have $T_i \in A_i$ for at least nq values of i is at least $1 - \epsilon$, no matter what the censors do. This means that $\Pr(\mathfrak{C}(x) \in \tilde{B}_{m,k}(T, n(1-q)) | X = x) \geq 1 - \epsilon$, so $\Pr(X \in G(T) | X = x) \geq 1 - \epsilon$.

Finally, we need to check that there is reasonable doubt. Each leaker chooses her message uniformly from a set of size k and each neutral from superset of size m. This gives a likelihood ratio of $\frac{m}{k} \leq \frac{b_m(1-b_l-b_c)}{b_l(1-b_m)} = r(b)$ so by Proposition 4.7 we have reasonable doubt. □

Corollary 4.23. *If $b_l + b_c < b_m$ the b-capacity is $D\left(\frac{b_l}{b_m}||\frac{b_l(1-b_m)}{b_m(1-b_c-b_l)}\right)$.*

Proof. Follows immediately from the above theorems. □

In Figure 4.6 we see a plot of the (b_l, b_c, b_m)-capacities divided by b_l for some values of b_c and b_m (we divide by b_l to get capacity measured in bit per expected leaker instead of bits per player). For $b_m = 0.95$ and small values of b_l a large

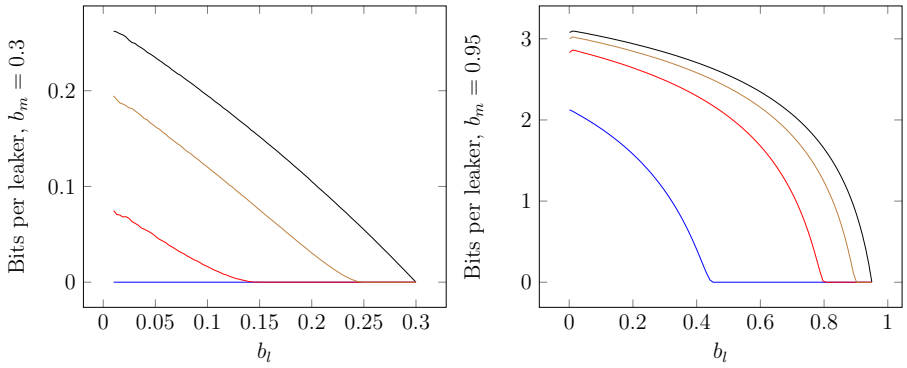

Figure 4.6: These two figures show the effect the censors have on the rate. The left figure is for $b_m = 0.3$ and the right for $b_m = 0.95$. The black (top) lines are the same as in Figure 4.4 and show the number of bits the leakers can leak per expected leaker when there are no censors. The brown (second from above) lines shown the same for $b_c = 0.05$, the red (second from below) for $b_c = 0.15$ and blue (bottom) for $b_c = 0.5$.

number of censors does not have much effect on the capacity. Even for $b_c = 0.5$ the censors lowers the rate with less than 32%. For $b_m = 0.3$ and $b_c = 0.15$ the censors have a larger effect on the capacity. In Section 4.5 we will consider the case where b_l is small in more detail.

Theorem 4.14 tells us that in order to make cryptogenography resilient to censors, we will have to accept that an observer cannot with high probability identify x, but only find a list with bounded length containing x. However, the following theorem shows that if we apply our randomly generated censor-resilient leaker protocol designed for some distribution b with censors to a situation where there are no censors, the list will most likely not contain any false secrets. This holds even if the fraction of leakers is not close to b_l. For example, if there are no censors and no leakers the list will most likely be empty. The theorem also holds, even if the neutral players and the leakers are not following π! Our only assumption about the communication is that it does not depend on the codewords of any false inputs $x' \neq x$.

Theorem 4.24. *Fix $b = (b_l, b_c, b_m)$ with $b_m > b_l + b_c$ and $R < D\left(\frac{b_l}{b_m} \middle|\middle| \frac{b_l(1-b_m)}{b_m(1-b_c-b_l)}\right)$ and $\epsilon > 0$. For sufficiently large n there exists a random variable $\mathfrak{C}$ that takes codes $\mathfrak{c}$ as values such that $\pi_{\mathfrak{c}}$ is always an $(n, nR, \mathcal{L}, \epsilon, b)$-protocol and furthermore*

if $x \in \{0,1\}^{\lceil nR \rceil}$ and π' is any communication protocol where each of the n players only get the codeword $\mathfrak{C}(x)$ for x as input then

$$\Pr(\exists x' \neq x : x' \in G_{\mathfrak{C}}(T')) < \epsilon,$$

where T' is the transcript of π' on input $\mathfrak{C}(x)$, and $G_{\mathfrak{C}}$ is the decoding function corresponding to $\pi_{\mathfrak{C}}$. The probability is over $\mathfrak{C}$ and randomness in π'.

Proof. Choose q, k, m as in the proof of Theorem 4.22 and let $\mathfrak{C}$ be the random variable obtained in Theorem 4.13. As each particular code $\mathfrak{c}$ satisfies the requirements of Theorem 4.12, it follows from the proof of Theorem 4.22 that if n is sufficiently large $\pi_{\mathfrak{c}}$ is always an $(n, nR, \mathcal{L}, \epsilon, b)$-protocol.

To show the second part, we can consider T' to be the output Y of a channel that takes $\mathfrak{C}(x)$ as input. Now the statement follows from Theorem 4.13. □

This could be generalized from just one value x to any set of values. This tells us that even without any assumptions on b_l and b_c or on the distribution on the messages send by neutral players, if a value x' is on the list, we can conclude that a group of people knew $\mathfrak{c}^{x'}$ and actively tried to put x' on the list (or a very unlikely event happened). Of course we have no way of knowing if these people were truthful (in which case we would call them leakers) or not (in which case we would call them censors).

4.4 Getting the best of both

Now that we have determined how small a list size we can have, and how high a rate we can achieve, we will see that asymptotically we can get the best of both parameters at the same time. The proof that you can get the best of both is constructive: given a family of protocols with small list size and a family of protocols with large rate, we give a construction that combines the two families to a family that has both a small list size and a large rate. However, as we do not have a construction that gives the optimal rate, the proof of the theorem below is not constructive.

Theorem 4.25. *Let $b = (b_l, b_m, b_c)$ with $b_l + b_c < b_m$ be given, and let $\mathcal{L}_0 = \left\lfloor \frac{b_c}{b_l} \right\rfloor + 1$. Then the $(\mathcal{L}_0, b)$-capacity is $D\left(\frac{b_l}{b_m} || \frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}\right)$.*

Proof. The upper bound follows directly from Theorem 4.21. To show the lower bound, let $R < D\left(\frac{b_l}{b_m} || \frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}\right)$ and let $\epsilon > 0$ and n_0 be given. We only need to find an $(n, nR, \mathcal{L}, \epsilon, b)$-protocol with $n \geq n_0$.

Let $R < R' < D\left(\frac{b_l}{b_m} || \frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}\right)$. By Theorem 4.22 there is an $\mathcal{L}'$ such that $(R', \mathcal{L}')$ is b-achievable, and by Theorem 4.16 the b-minimal list size is at most $\mathcal{L}_0$. This implies that for some n'' there is an $(n'', \log(\mathcal{L}'), \mathcal{L}, \epsilon/2, b)$-protocol π''. Let $n_0' = \max\left(\frac{Rn''}{R'-R}, n_0\right)$. As $(R', \mathcal{L}')$ is b-achievable we can find an $(n', n'R', \mathcal{L}', \epsilon/2, b)$-protocol π' for some $n' \geq n_0'$. Define $n = n' + n''$. We will now give an $(n, nR, \mathcal{L}_0, \epsilon, b)$-protocol π.

π starts by running π' on the first n' players, giving a set $G'(T')$ of size $\mathcal{L}'$ such that for each x, we have $\Pr(x \in G'(T') | X = x) \geq 1 - \epsilon/2$. Next π uses the last n'' players to simulate π'' as if X could only take values in $G'(T')$. If $x \notin G'(T')$ the leakers will just behave as if they were non-leakers. The protocol is defined more formally in Figure 4.7.

As both π' and π'' are b-leaker protocols, they have likelihood ratio at most $r(b)$. In π each player only participates in one of the two subprotocols, so the likelihood ratio of π is also at most $r(b)$, so π is a b-leaker protocol. It only fails if either π' or π'' fails, so it fails with probability at most ϵ. Thus, it is an $(n, n'R', \mathcal{L}, \epsilon, b)$-protocol. Its rate is

$$\frac{n'R'}{n} = \frac{n'R'}{n' + n''} \geq \frac{\frac{Rn''}{R'-R}R'}{\frac{Rn''}{R'-R} + n''} = \frac{Rn''R'}{Rn'' + R'n'' - Rn''} = R.$$

Here the inequality comes from the fact that $\frac{n'R'}{n'+n''}$ is increasing in n' and $n' \geq n_0' \geq \frac{Rn''}{R'-R}$. □

Putting the above results together, we get our main theorem which (except for the null-set where $R = C$) determines which $(\mathcal{L}, R)$ are b-achievable.

Theorem 4.26. *Let $b = (b_l, b_m, b_c)$ with $b_l + b_c < b_m$ be given, and let $\mathcal{L}_0 = \lfloor \frac{b_c}{b_l} \rfloor + 1$ and $C = D\left(\frac{b_l}{b_m} || \frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}\right)$. If $0 < R \neq C$ then $(\mathcal{L}, R)$ is b-achievable if and only if both $\mathcal{L} \geq \mathcal{L}_0$ and $R < C$.*

Parameters:

b : probability of being leaker respectively censor and threshold of reasonable doubt

R' : a rate between R, and $D\left(\frac{b_l}{b_m}||\frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}\right)$

$\mathcal{L}'$: a list length such that $(R', \mathcal{L}')$ is b-achievable

ϵ : acceptable probability of error

π', n' : an $(n', n'R', \mathcal{L}', \epsilon/2, b)$-protocol for some n'

π'', n'' : an $(n'', \log(\mathcal{L}'), \mathcal{L}, \epsilon/2, b)$-protocol for some n''

$n = n' + n''$: total number of players needed

Input distribution: $X, L_1, \ldots, L_n$ are independently distributed, X is uniformly distributed on $\{0,1\}^h$ and for each i: $\Pr(L_i = 1) = b_l$ and $\Pr(L_i = -1) = b_c$. Each PLR_i learns L_i and if $|L_i| = 1$ she also learns X, and if $L_i = -1$ she also learns $L_- = (\min(L_1, 0), \ldots, \min(L_n, 0))$. She will follow the protocol given here if $L_i \neq -1$ and otherwise she might follow any censor protocol σ.

Protocol:

1. Let $\text{PLR}_1, \ldots, \text{PLR}_{n'}$ follow protocol π' to reveal X, and let $G'(T')$ denote the resulting list
2. Let player $\text{PLR}_{n'} + 1, \ldots, \text{PLR}_n$ follow π''. If $X \in G'(T')$, the secret to reveal is X's lexicographical position in $G'(T')$. If $X \notin G'(T')$, the leakers pretend ignorance (behave as neutrals), so we do not need to specify the secret to be leaked.

Figure 4.7: Protocol from proof of Theorem 4.25.

Proof. The "if" part follows from Theorem 4.25 and the fact that increasing $\mathcal{L}$ and decreasing R only makes it easier for the leakers. The "only if" part follows from Corollary 4.17 and 4.23. □

This means that asymptotically, we can get the best rate and the best list size at the same time. However, this only holds asymptotically (and the speed of convergence is not uniform in (b_l, b_m, b_c)): If $b_c = b_l(2 - 10^{-100})$, we can get down to a set of 3 possible secrets with a reasonable n, but to get down to a set of 2 possible

secrets, n would have to be big enough that the -10^{-100} has an effect.

4.5 Few leakers and censors

In the previous sections we have considered a situation where the probability of being a leaker and the probability of being a censor are both constant, while the total number of players tends to infinity. In this section we want to see what happens when the total number of players goes to infinity faster than the expected number of leakers. We keep both b_m and b_c fixed with $b_c < b_m$, and allow b_l to be arbitrarily small.

Here we will see that while the list size tends to infinity, the rate measured in bits per expected leaker is bounded away from 0. In other words, as long as $b_c < b_m$ a small minority can reveal information without revealing themselves, and the number of bits they can reveal, will for small b_l be approximately proportional to the expected number of leakers, and the resulting list size will be inversely proportional to the expected number of leakers.

Definition 4.9. Let b_c and b_m be given. For $\mathcal{L} \in \mathbb{N}$ and $R > 0$ we say that $(\mathcal{L}, R)$ is (b_c, b_m)-*achievable* if for all $\epsilon > 0$ and all n_0 there exist b_l, n and an $(n, Rnb_l, \mathcal{L}, \epsilon, (b_l, b_c, b_m))$-protocol with $nb_l > n_0$.

We say R is (b_c, b_m)-*achievable* if $(\mathcal{L}, R)$ is (b_c, b_m)-achievable for some $\mathcal{L}$.

The (b_c, b_m)-*capacity* is the supremum over all rates R that are (b_c, b_m)-achievable.

Here the rate is measured in bits per expected leaker rather than bits per person. We do not require that b_l is small, but this must be the case for any protocol that achieves a rate close to the capacity.

In order to determine the (b_c, b_m)-capacity, we need the following proposition.

Proposition 4.27. *For fixed $b_c < b_m$ we have*

$$\lim_{b_l \to 0^+} \frac{1}{b_l} D\left(\frac{b_l}{b_m} \middle|\middle| \frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}\right) = \frac{\log(\frac{1-b_c}{1-b_m})}{b_m} + \frac{b_c - b_m}{b_m(1-b_c)} \log(e).$$

Furthermore, for any b_l, b_c, b_m with $b_l + b_c < b_m$ we have

$$\frac{1}{b_l} D\left(\frac{b_l}{b_m}\middle\|\frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}\right) \leq \frac{\log(\frac{1-b_c}{1-b_m})}{b_m} + \frac{b_c - b_m}{b_m(1-b_c)}\log(e). \tag{4.16}$$

Proof. From the definition of $D(\cdot||\cdot)$ we get

$$\begin{aligned}
&\frac{1}{b_l} D\left(\frac{b_l}{b_m}\middle\|\frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}\right) \\
&= \frac{1}{b_l}\left(\frac{b_l}{b_m}\log\left(\frac{1-b_l-b_c}{1-b_m}\right) + \left(1-\frac{b_l}{b_m}\right)\log\left(\frac{1-\frac{b_l}{b_m}}{1-\frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}}\right)\right) \\
&= \frac{\log\left(\frac{1-b_l-b_c}{1-b_m}\right)}{b_m} + \left(\frac{1}{b_l}-\frac{1}{b_m}\right)\left(\log\left(1-\frac{b_l}{b_m}\right) - \log\left(1-\frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}\right)\right) \\
&\xrightarrow{b_l\to 0} \frac{\log\left(\frac{1-b_c}{1-b_m}\right)}{b_m} + \left(\frac{-1}{b_m} + \frac{1-b_m}{b_m(1-b_c)}\right)\log(e) \\
&= \frac{\log\left(\frac{1-b_c}{1-b_m}\right)}{b_m} + \frac{b_c-b_m}{b_m(1-b_c)}\log(e).
\end{aligned}$$

This proves the first part of the proposition. To prove the second part, we differentiate the left hand side of (4.16) (divided by $\log(e)$ to simplify the computations):

$$\begin{aligned}
&\frac{\partial}{\partial b_l}\left(\frac{1}{b_l\log(e)} D\left(\frac{b_l}{b_m}\middle\|\frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}\right)\right) \\
&= -\frac{(1-b_c)b_l(b_m-b_c-b_l) + (1-b_c-b_l)((1-b_c)b_m-b_l)\ln\left(\frac{(1-b_c-b_l)(b_m-b_l)}{(1-b_c)b_m-b_l}\right)}{b_l^2(1-b_c-b_l)((1-b_c)b_m-b_l)}.
\end{aligned} \tag{4.17}$$

To show the desired inequality, it is enough to show that this is non-positive for $b_c + b_l \leq b_m < 1$. First we show this for the case $b_c > 0$ and $b_m = b_c + b_l < 1$. In

this case denominator is positive, $b_m - b_c - b_l = 0$, and

$$\begin{aligned}(1-b_c)b_m - b_l =& (1-b_c-b_l+b_l)b_m - b_l \\ =& (1-b_m)b_m + b_l b_m - b_l \\ =& (1-b_m)(b_m - b_l) \\ =& (1-b_c-b_l)(b_m-b_l),\end{aligned}$$

hence $\ln\left(\frac{(1-b_c-b_l)(b_m-b_l)}{(1-b_c)b_m-b_l}\right) = 0$ so the entire numerator is 0, which is non-positive. To generalize to $b_m > b_c + b_l$ we differentiate (4.17) with respect to b_m:

$$\begin{aligned}&\frac{\partial^2}{\partial b_l \partial b_m}\left(\frac{1}{b_l \log(e)} D\left(\frac{b_l}{b_m}||\frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}\right)\right) \\ =&\frac{-b_c^2}{(b_m-b_l)((1-b_c)b_m-b_l)^2}.\end{aligned}$$

This is clearly non-positive, hence both sides of equality (4.17) are non-positive in the case $b_m > b_c + b_l$ and $b_c > 0$. This shows inequality (4.16) in the case $b_c + b_l \leq b_m$ and $b_c > 0$. To get the case $b_c = 0$, observe that both sides of (4.16) are continuous in b_c. □

Theorem 4.28. *Let $b_c < b_m$ be given. Then no rate above*

$$\frac{\log\left(\frac{1-b_c}{1-b_m}\right)}{b_m} - \frac{(b_m-b_c)\log(e)}{b_m(1-b_c)}$$

is (b_c, b_m)-achievable.

Proof. Assume for contradiction that there is a $\mathcal{L}$ and a rate $R > \frac{\log\left(\frac{1-b_c}{1-b_m}\right)}{b_m} - \frac{(b_m-b_c)\log(e)}{b_m(1-b_c)}$ such that $(\mathcal{L}, R)$ is (b_c, b_m)-achievable, and let

$$\delta = R - \left(\frac{\log\left(\frac{1-b_c}{1-b_m}\right)}{b_m} - \frac{(b_m-b_c)\log(e)}{b_m(1-b_c)}\right).$$

Let π be a $(n, Rnb_l, \mathcal{L}, \epsilon, (b_l, b_c, b_m))$-protocol. If $b_m \leq b_c + b_l$, it follows from Theorem 4.1 that the censors can ensure $I(X;T) = 0$ and otherwise we know from Proposition

4.20 there is a censor protocol that ensures

$$\begin{aligned} I(X;T) \leq & nD\left(\frac{b_l}{b_m}\middle|\middle|\frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}\right) \\ \leq & nb_l\left(\frac{\log(\frac{1-b_c}{1-b_m})}{b_m}+\frac{b_c-b_m}{b_m(1-b_c)}\log(e)\right) \\ = & nb_l(R-\delta). \end{aligned}$$

Here the second inequality follows from second part of Proposition 4.27.

Now as $G(T)$ returns sets of size at most $\mathcal{L}$ and contains x with probability $1-\epsilon$, Lemma 4.19 implies

$$\begin{aligned} \epsilon \geq & \frac{H(X|G(T))-1-\log(\mathcal{L})}{Rnb_l} \\ = & \frac{H(X)-I(X;G(T))-1-\log(\mathcal{L}))}{Rnb_l} \\ \geq & \frac{H(X)-I(X;T)-1-\log(\mathcal{L}))}{Rnb_l} \\ \geq & \frac{nb_lR-nb_l(R-\delta)-1-\log(\mathcal{L}))}{Rnb_l} \\ \geq & \frac{\delta nb_l-1-\log(\mathcal{L})}{Rnb_l}. \end{aligned}$$

Here the first equality follows from rules for mutual information, and the second inequality from the data processing inequality. Thus, for fixed $\mathcal{L}$ and $R > \frac{\log\left(\frac{1-b_c}{1-b_m}\right)}{b_m} - \frac{(b_m-b_c)\log(e)}{b_m(1-b_c)}$ and for sufficiently large nb_l, we have a lower bound on ϵ. Thus, $(\mathcal{L}, R)$ is not (b_c, b_m)-achievable. □

We are now ready for the main theorem of this section. Notice that the proof of the lower bound uses Theorem 4.26, so it is not constructive.

Theorem 4.29. *Let (b_c, b_m) be given with $b_c < b_m$. Then the (b_c, b_m)-capacity is*

$$\frac{\log\left(\frac{1-b_c}{1-b_m}\right)}{b_m} - \frac{(b_m-b_c)\log(e)}{b_m(1-b_c)}.$$

Proof. The upper bound follows from Theorem 4.28.

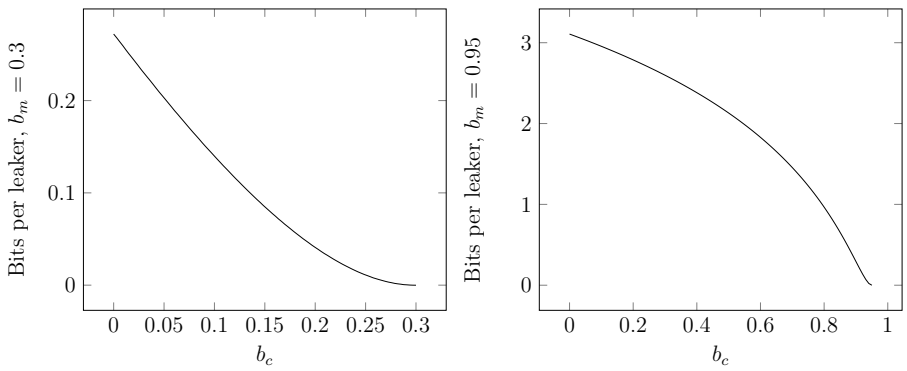

Figure 4.8: The plot on the left shows the $(b_c, 0.3)$-capacity and the plot on the right shows the $(b_c, 0.95)$-capacity as a function of b_c.

To show the lower bound, let $R < \frac{\log\left(\frac{1-b_c}{1-b_m}\right)}{b_m} - \frac{(b_m - b_c)\log(e)}{b_m(1-b_c)}$. Then by Proposition 4.27 there exists a $b_l > 0$ such that $R < \frac{1}{b_l} D\left(\frac{b_l}{b_m} || \frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}\right)$. Choose such a b_l and let $\mathcal{L} = \lfloor \frac{b_c}{b_l} \rfloor + 1$. We will show that $(\mathcal{L}, R)$ is b-achievable. To do that, let $\epsilon > 0$ and n_0 be given. We need to show that there exists an $(n, Rnb_l, \mathcal{L}, \epsilon, (b_l, b_c, b_m))$-protocol π with $n \geq n_0$. This follows from Theorem 4.26 and $Rb_l < D\left(\frac{b_l}{b_m} || \frac{b_l(1-b_m)}{b_m(1-b_l-b_c)}\right)$. $\square$

The (b_c, b_m)-capacity is plotted for $b_m = 0.95$ and $b_m = 0.3$ in Figure 4.8. We see that especially for $b_m = 0.95$ a large fraction of the people must be censors in order to get a substantial reduction in the rate. From Theorem 4.29 we see that the (b_c, b_m)-capacity is continuous in $b_c \in [0, 1)$. This implies that in the case where both b_c and b_l are small the censors only have a vanishingly small effect on the rate at which the leakers leak information, even if b_c is much larger than b_l. For example, if $b_l = 10^{-6}$ and $b_c = 10^{-3}$, the censors will only have a small effect on the rate, although there will typically be a thousand times more censors than leakers.

Chapter 5

Cryptogenography Games

The general theme of this thesis is to figure out which trade-offs between sending information and keeping anonymity are possible. In Chapter 3 we first found a measure of suspicion which exactly captures the anonymity you lose when sending information. This measure has the disadvantage of being artificial: it is unlikely that anyone is going to care about their expected suspicion. We also considered a more natural way of measuring the trade-off: how much information can you reveal if you want to ensure that an observer will assign probability at most b_m to the event that you are revealing information. In this chapter, we will consider a different way of measuring such trade-offs: if you play a game where you have to correctly send information *and* ensure that an adversary does not guess a leaker in her first guess, what is the maximal probability of winning?

This chapter has two parts. The first part is based on the first publication about cryptogenography [9] and is using a different method than the rest of the thesis. In this part there is only one leaker, and the secret is only one bit. The resulting problem is very similar to problems in the area of information complexity, and we will use a method which was independently discovered in this area. For two players, including one leaker, we show that the leaker's probability of winning is at least $\frac{1}{3} = 0.33\ldots$ and at most $\frac{3}{8} = 0.375$. These bounds have later been improved to 0.3384 and 0.3672 [5]. For a large number of players we show that the leaker's probability of winning is at least 0.5644 and at most $\frac{3}{4} = 0.75$. Both the lower bounds are proved by constructing protocols.

In the second part of the chapter we consider the same problem when there are many leakers and many bits to leak. For this problem the measure of suspicion once again turns out to be useful. Here the main results are Theorem 5.26, which says that the players have probability at least p of winning if there is $\left(\frac{-\log(p)}{1-p} - \log(e)\right)$ bits per leaker, and Theorem 5.33, which says that if there is $r\log(e)$ bits per leaker, their probability of winning is at most $\frac{\log(r+1)}{r\log(e)}$. See Figure 5.12 for an illustration of these bounds. The lower bound is proved by a non-constructive argument that shows the existence of a good protocol.

5.1 Model

The (n-player) cryptogenography game is formally defined as follows. There are n players, denoted $\text{PLR}_1, \ldots, \text{PLR}_n$. Inputs consist of $(X, L) \sim \mu$, where μ is the uniform distribution over $\{0,1\} \times [n]$. We refer to X as the *secret* and say that PLR_L is the *leaker*, or that PLR_L knows the secret. Both X and L are given to PLR_L; other players receive no input. Players communicate using a protocol π, after which an observer can compute a guess $\text{G} : \{0,1\}^* \to \{0,1\}$ for the secret. We will sometimes refer to this observer as Joe. Let $\text{Eve} : \{0,1\}^* \to [n]$ be the function that maximizes $\Pr[\text{Eve}(T) = L \mid \text{G}(T) = X]$ for each possible value of the protocol transcript. Notice that Eve depends on G. This function represents the best possible guess of an adversary (whom we call Eve), who sees all communication between the players and wants to determine the identity of the leaker. Note that $\text{G}(T)$ and $\text{Eve}(T)$ are functions of the messages sent in π. We define the success of a protocol as

$$\text{succ}(\pi) := \Pr[\text{G}(T) = X \text{ and } \text{Eve}(T) \neq L] \ .$$

The *communication cost* of π, denoted $\text{CC}(\pi)$, is the maximum number of bits sent during π, taken over all possible inputs (x, l) and all choices of randomness. We will focus on understanding the maximum possible $\text{succ}(\pi)$ of a protocol, not on the communication cost.

The following lemma shows that one can assume without loss of generality that players learn the secret with certainty.

Lemma 5.1. *For all protocols π there exists a cryptogenography protocol π' with*

transcript T' *such that* $\mathrm{succ}(\pi') = \mathrm{succ}(\pi)$, $\mathrm{CC}(\pi') = \mathrm{CC}(\pi) + n$, *and such that* $\Pr[\mathrm{G}(T') = X] = 1$.

Proof. The players first execute π. Then each player send one bit, which is used to indicate whether they know $\mathrm{G}(T)$ to be wrong. If a player do not know X or if $\mathrm{G}(T) = X$ the player will send the message 0. If a player knows X and $\mathrm{G}(T) \neq X$ the player will send 1. The protocol is formally defined in Figure 5.1

Parameters:

n : number of players

π : a cryptogenography protocol

Input distribution: (X, L) uniformly distributed on $\{0,1\} \times [n]$. Each $\textsc{plr}_i$ learns if $L = i$ and if it is, she also learns X.

Protocol:

1. Let all the players follow π and let $\mathrm{G}(T)$ denote the output
2. For i for 1 to n
3. If $L = i$ and $X \notin \mathrm{G}(T)$ then $\textsc{plr}_i$ sends 1 otherwise $\textsc{plr}_i$ sends 0

Figure 5.1: Protocol from proof of Lemma 5.1.

Let T' be the transcript of the resulting protocol π' and define $\mathrm{G}'(T')$ to equal $\mathrm{G}(T)$ if all players communicate a 0 in the extra round of communication; otherwise, set $\mathrm{G}'(T') = 1 - \mathrm{G}(T)$. It is easy to see that $\mathrm{G}'(T') = X$ with certainty: either π correctly computes X already, or the leaker announces that $\mathrm{G}(T) \neq X$. It is also trivial to verify that $\mathrm{CC}(\pi') = \mathrm{CC}(\pi) + n$. Thus, it remains to show that

$\text{succ}(\pi') = \text{succ}(\pi)$. This can be seen through the following chain of equalities.

$$\begin{aligned}
\text{succ}(\pi') &= \Pr[\text{G}'(T') = X \wedge \text{Eve}'(T') \neq L] \\
&= \Pr[\text{Eve}'(T') \neq L] \\
&= \Pr[\text{Eve}'(T') \neq L \mid \text{G}(T) = X] \cdot \Pr[\text{G}(T) = X] \\
&\quad + \Pr[\text{Eve}'(T') \neq L \mid \text{G}(T) \neq X] \cdot \Pr[\text{G}(T) \neq X] \\
&= \Pr[\text{Eve}'(T') \neq L \mid \text{G}(T) = X] \cdot \Pr[\text{G}(T) = X] \\
&= \Pr[\text{Eve}(T) \neq L \mid \text{G}(T) = X] \cdot \Pr[\text{G}(T) = X] \\
&= \text{succ}(\pi),
\end{aligned}$$

where the second equality holds because players always learn X in π', the third equality holds by conditioning on $\text{G}(T)$, the fourth equality holds because if $\text{G}(T) \neq X$ then the leaker is going to reveal herself in π', and the penultimate equality holds because, conditioned on π correctly computing X, the eavesdropper in π' learns nothing new about L. □

5.2 Cryptogenography game protocols

In this section, we present a series of protocols that demonstrate what is possible for the players to achieve.

5.2.1 Two player cryptogenography game

When $n = 2$, we refer to players as Alice and Bob instead of PLR_1 and PLR_2.

Theorem 5.2. *There is a two-player cryptogenography protocol* π *with* $\text{succ}(\pi) = 1/3$ *and* $\text{CC}(\pi) = 2$.

Proof. This protocol proceeds in two rounds. In the first round of communication, Alice decides whether to "pass" or "speak". If she passes, then Bob speaks in the second round; otherwise she speaks. In the second round of communication, whoever speaks will (i) send the secret if she knows it and (ii) send a random bit otherwise. G now outputs the second bit of communication as the guess for the secret.

All that remains is to complete the protocol is to specify how Alice chooses to pass or speak in the first round. If Alice knows the secret, she passes with probability 2/3 and speaks with probability 1/3; otherwise, she passes with probability 1/3 and speaks with probability 2/3. The protocol is also defined in Figure 5.2.

Input distribution: (X, L) uniformly distributed on $\{0,1\} \times [2]$. Each PLR_i learns if $L = i$ and if it is, she also learns X.
Protocol:

1. If $L = 1$, Alice sends "speak" with probability $\frac{1}{3}$ and if $L \neq 1$ she sends "speak" with probability $\frac{2}{3}$. If she does not send "speak" she sends "pass".
2. The Alice sent "speak" she will send the next message, if she sent "pass" Bob will send the next message: If the sender knows X he or she sends X otherwise he or she chooses a message uniformly at random from $\{0,1\}$.

Figure 5.2: Protocol from proof of Theorem 5.2.

Note that Alice is more likely to speak in round 2 when she *does not* know the secret. This is perhaps counterintuitive—the players output the second bit of communication, so intuitively Alice should speak more often when she actually knows the bit. Is there an a priori reason why Alice shouldn't just announce the secret if she knows it and pass otherwise? Unfortunately in this case, Eve will learn with certainty who knows the bit. Alice's probabilities of passing are chosen to give Eve no information about the leaker *conditioned on players successfully outputting the secret.*

Claim 5.3. $\Pr[\text{G}(T) = X] = 2/3$.

Proof. The leaker speaks in the second round with probability 1/3. In this case, players output correctly with certainty. Otherwise, players output a random bit and are correct with probability 1/2. Overall, they output the correct bit with probability $1/3 + (2/3) \cdot (1/2) = 2/3$. □

Claim 5.4. $\Pr[\text{Eve}(T) = L \mid \text{G}(T) = X] = 1/2$.

Proof. Without loss of generality, assume Alice speaks in the second round and that she sends the message 0. Call the resulting transcript t. From Eve's point of view,

there are three cases:

(i) Alice is the leaker and therefore outputs the correct bit in round 2. A priori, Alice is the leaker with probability $\frac{1}{2}$; given that she is the leaker she will speak with probability $\frac{1}{3}$; given this, there is probability $\frac{1}{2}$ that the secret is 0. Thus, the a priori probability of this case is $\frac{1}{2}\frac{1}{3}\frac{1}{2} = \frac{1}{12}$.

(ii) Alice is not the leaker but outputs the correct bit anyway. The priori probability that Alice is not the leaker is $\frac{1}{2}$; given that she is not the leaker, there is probability $\frac{2}{3}$ that she speaks; given that, there is probability $\frac{1}{2}$ that $X = 0$ and independently probability $\frac{1}{2}$ that she says 0. This gives an a priori probability of $\frac{1}{2}\frac{2}{3}\frac{1}{2}\frac{1}{2} = \frac{1}{12}$ for this case happening.

(iii) Alice is not the leaker and outputs incorrectly in round 2. Again, the a priori probability that Alice is not the leaker but speaks is $\frac{1}{2}\frac{2}{3} = \frac{1}{3}$. Given this, there is probability $\frac{1}{2}\frac{1}{2}$ that she says 0 and $X = 1$. This gives probability $\frac{1}{12}$ of this case happening.

Thus, conditioned on the transcript being t, each of these cases has probability $\frac{1}{3}$. However, in the third case, the players have already failed. Thus, *conditioned on players correctly outputting the secret*, Alice and Bob are equally likely to be the leaker, and Eve can only guess at random. □

In this protocol, players output the secret with probability 2/3 and given this, Eve guesses the leaker with probability 1/2. Thus, the overall success probability is 1/3. □

This lower bound on the winning probability in the two-player game was improved by Doerr and Künnemann from the $\frac{1}{3}$ proved here to 0.3384, by using computer search to find a better protocol [5]. The protocol which achieves this winning probability involves 18248 different game states.

5.2.2 Cryptogenography game protocols with many players

Next, we present a series of protocols for the general case.

The Majority-Votes Protocol. In this protocol π_{MAJ}, there is a single round of communication, with each player sending a single bit. PLR_i sends X if she knows

the secret; otherwise, PLR_i sends a random bit. This protocol is also defined in Figure 5.3. Let t_i denote the bit communicated by PLR_i, and define $\text{G}_{\text{MAJ}}(t) := \text{MAJ}(t_1, \ldots, t_n)$. When n is odd, the distribution of $\text{MAJ}(t_1, \ldots, t_n)$ is biased slightly towards X. This enables players to achieve success probability somewhat larger than $1/2$.

Parameter:

n : number of players

Input distribution: (X, L) uniformly distributed on $\{0,1\} \times [n]$. Each PLR_i learns if $L = i$ and if it is, she also learns X.

Protocol:

1. For i from 1 to n
2. If $L = i$ then PLR_i outputs X, otherwise PLR_i chooses a message uniformly from $\{0,1\}$

Figure 5.3: Majority-Votes Protocol, π_{MAJ}

Lemma 5.5. *If n is odd, then π_{MAJ} succeeds with probability $1/2 + \Theta(1/\sqrt{n})$.*

Proof. The communication $t_1, \ldots, t_n$ consists of $n-1$ random bits, along with the secret X. It will be helpful to be more explicit about the success probability. Let z_i be an indicator variable for the event $t_i = X$. Note that since the t_i's are uniform and independent for $i \neq L$, so are the z_i's. In π_{MAJ}, players output $\text{MAJ}(t_1, \ldots, t_n)$; therefore, $\text{G}_{\text{MAJ}}(t) = X$ if and only if $\sum_{i \neq L} z_i \geq \frac{n-1}{2}$. Thus, we have

$$\Pr[\text{G}_{\text{MAJ}}(T) = X] = \Pr\left[\sum_{i \neq L} z_i \geq \frac{n-1}{2}\right] = \frac{1}{2} + 2^{-n}\binom{n-1}{(n-1)/2} = \frac{1}{2} + \Theta\left(\frac{1}{\sqrt{n}}\right) .$$

It is easy to see that the best choice for Eve is to guess a random player i whose bit agrees with the majority. There are at least $n/2$ such bits; therefore, $\Pr[\text{Eve}(T) = L \mid \text{G}_{\text{MAJ}}(T) = X] = 1/2 + \Theta(1/\sqrt{n}) - O(1/n) = 1/2 + \Theta(1/\sqrt{n})$. □

We can achieve a more precise analysis by conditioning on $\sum_i z_i$. We have

$$\begin{aligned}\mathrm{succ}(\pi_{\mathrm{MAJ}}) &= \sum_{k=(n-1)/2}^{n-1} \Pr\left[\sum_i z_i = k\right] \Pr\left[\mathrm{Eve}(T) \neq L \mid \sum z_i = k\right] \\ &= \sum_{k=(n-1)/2}^{n-1} 2^{-(n-1)} \cdot \binom{n-1}{k} \cdot \left(1 - \frac{1}{k+1}\right) \ .\end{aligned}$$

A straightforward calculation shows that the success probability of π_{MAJ} is maximized at $\mathrm{succ}(\pi_{\mathrm{MAJ}}) \approx 0.5406$ when $n = 23$. However, for large n, the success probability decreases and approaches $1/2$. Our next protocol handles both cases by emulating a protocol for a smaller number of players.

A Continuous Protocol for large n. Let $n > n'$ be given, and fix an n'-player protocol π'. We will now construct a protocol π for n players that achieves the same success probability as π'. The idea is to let $n - n'$ people drop out of the game, but if the protocol specifies these $n - n'$ players, the leaker might be among them.

To remove the $n - n'$ players without removing the leaker from the game, we use what we call a *continuous protocol*. That is, we assume the existence of a real-valued "clock" that all players see, or more formally, the protocol assumes that all players have access to some $\eta \in \mathbb{R}_{\geq 0}$. When the protocol begins, $\eta = 0$, and η increases as the protocol progresses. While this does not fit the usual definition of a communication protocol, such protocols have been used with the name "protocol with a clock" in information complexity by Braverman, Garg, Pankratov and Weinstein [8]. Later we will see how we can turn a continuous protocol into a real communication protocol.

Each player generates a real number $r_i \in [0, 1]$. The leaker $\textsc{plr}_L$ sets $r_L := 1$; for $i \neq L$, $\textsc{plr}_i$ sets r_i uniformly in $[0, 1]$. As η increases, each player announces when $\eta = t_i$. When all but n' players have spoken, the remaining players run π'. See Figure 5.4 for a more formal definition of the protocol. We call the communication before emulating π' the *continuous phase* of the communication. It is easy to see that at the end of this continuous phase, L is uniformly distributed over the n' remaining players. Thus, π has precisely the same success probability as π'. This gives us the following lemma.

Parameter:

n : number of players we wish to use

$n' < n$: number of players in existing protocol

π' : a protocol for n' players

Input distribution: (X, L) uniformly distributed on $\{0,1\} \times [n]$. Each PLR_i learns if $L = i$ and if it is, she also learns X.
Protocol:

1. Each player PLR_i choose a number r_i. If $L \neq i$, r_i is chosen uniformly at random from $[0,1]$, if $L = i$ then $r_i = 1$
2. For time t increasing continuously from 0 to 1
3. If $r_i = t$ for some i and less than $n - n'$ messages have been send, player PLR_i sends the message "I am not a leaker"
4. If more than $n - n'$ players have send a message the protocol aborts, otherwise
5. The n' players who have not sent a message now follow π'

Figure 5.4: Continuous protocol from a protocol with fewer players. The probability that the protocol aborts because too many players revealed themself as non-leakers is 0.

Lemma 5.6. *Given any n' player protocol π' and any $n > n'$, there exists a n-player* continuous *protocol π achieving* $\text{succ}(\pi) = \text{succ}(\pi')$.

Together with Lemma 5.5, we get an efficient protocol for all large n.

Corollary 5.7. *For all $n \geq 23$, there is a continuous protocol π achieving* $\text{succ}(\pi) \geq 0.5406$.

Proof. The corollary follows from Lemma 5.5 and Lemma 5.6. The resulting protocol is given in Figure 5.5. □

The assumption that all players have shared access to a continuous clock is perhaps unnatural, and it is unclear how players can emulate such a protocol without access to this clock. Nevertheless, it is a useful abstraction, and while it is hard to

Parameter:

$n \geq 23$: number of players we wish to use

Input distribution: (X, L) uniformly distributed on $\{0,1\} \times [n]$. Each PLR_i learns if $L = i$ and if it is, she also learns X.
Protocol:

1. Each player PLR_i choose a number r_i. If $L \neq i$, r_i is chosen uniformly at random from $[0,1]$, if $L = i$ then $r_i = 1$
2. For time t increasing continuously from 0 to 1
3. If $r_i = t$ for some i and less than $n - 23$ messages have been send, player PLR_i sends the message "I am not a leaker"
4. If more than $n-23$ players have send a message the protocol aborts, otherwise
5. For i from 1 to 23
6. If the i'th of the 23 players who have not sent a message is a leaker, she sends X otherwise she sends a message chosen uniformly at random from $\{0,1\}$

Figure 5.5: Continuous protocol proving Corollary 5.7.

see how such protocols can be emulated, it is easy to construct a protocol that *approximates* them. Our next protocol is just such a construction.

Lemma 5.8. *Fix n, n' with $n > n'$, and let $\epsilon > 0$. For any n'-player protocol π', there exists an n-player protocol π with* $\text{succ}(\pi) \geq \text{succ}(\pi') - \epsilon$ *and* $\text{CC}(\pi) = \text{CC}(\pi') + O(n^3/\epsilon)$.

Proof. Given ϵ, let $m = \lceil \frac{n^2}{\epsilon} \rceil$. Similar to the continuous protocol, each PLR_i with $i \neq L$ generates $t_i \in [m]$ uniformly. The leaker then sets $r_L := m + 1$. In the first phase of communication, players proceed in rounds $k = 1, 2, \ldots$. In the kth round, each PLR_i announces whether $r_i \leq k$. Call PLR_i *alive* if $r_i > k$. Communication in the first phase continues until $k = m$ or until at most n' players remain alive. In the second phase, the remaining alive players execute π' if exactly n' players remain; otherwise, they output something arbitrary. The protocol is defined more formally

in Figure 5.6.

Parameter:

n : number of players we wish to use

$n' \leq n$: number of players in existing protocol

m : number of rounds. The higher it is, the better is the resulting protocol

π : protocol for n' players

Input distribution: (X, L) uniformly distributed on $\{0,1\} \times [n]$. Each PLR_i learns if $L = i$ and if it is, she also learns X.
Protocol:

1. Each player PLR_i choose a number r_i. If $L \neq i$, r_i is chosen uniformly at random from $[m]$, if $L = i$ then $r_i = m + 1$
2. For time t from 1 to m
3. For i from 1 to n
4. If $r_i = t$ and less than $n - n'$ messages have been send, player PLR_i sends the message "I am not a leaker"
5. If more than $n - n'$ players have send a message the protocol aborts, otherwise
6. The n' players who have not sent a message now follow π

Figure 5.6: Protocol from proof of Lemma 5.8.

There are $O\left(\frac{n^2}{\epsilon}\right)$ rounds of communication in the first phase of π, and each player sends a single bit in each of these rounds. Thus, π uses $O\left(\frac{n^3}{\epsilon}\right)$ additional communication over π'.

It is easy to see that conditioned on the communication in the first phase of π, L is uniformly distributed over the remaining alive players. The probability that two particular players choose the same r_i is $m^{-1} \leq \frac{\epsilon}{n^2}$. As there are $\binom{n}{2} \leq n^2$ pairs of players, the union bound implies that the probability that two players have the same r_i is at most $n^2 \frac{\epsilon}{n^2} = \epsilon$. Thus, the probability that players do not execute π' is at most ϵ. □

The above Lemma is not optimal in terms of communication complexity. We could easily reduce the number of rounds by improving the analysis: it does not matter if more than one player drop out in the same round, as long as that does not bring the number of alive players below n'. We could further decrease the number of rounds by increasing the probability that a player drop out in the first round.

Taking the majority-votes protocol and fixing ϵ to be a suitably small constant yields the following corollary.

Corollary 5.9. *For all $n \geq 23$, there exists a protocol π with* $\mathrm{succ}(\pi) > 0.5406$ *and* $\mathrm{CC}(\pi) = O(n^3)$.

Proof. The corollary follows from Lemma 5.5 and Lemma 5.8. The resulting protocol is given in Figure 5.7. □

Beating Majority-Votes. For our final protocol we show that, perhaps surprisingly, one can boost success by reversing the above operations. Specifically, we consider an n-player protocol with two phases of communication. In the first phase, each player votes, as in π_{MAJ}. In the second phase of communication, players communicate to decide one-by-one who will *not* participate in the vote. Call a player *dead* if he has been chosen to no longer participate. Eventually, players decide to end the second phase of communication and compute the majority of the remaining votes. By voting first, and eliminating players from the vote one-by-one, the protocol can *adaptively* decide when to stop the protocol. At a high level, the protocol ends when the votes of the remaining players form a *super-majority*. Say that $t_1, \ldots t_n$ form a τ-super-majority if τ of the n bits agree.

Fix a function $\tau : \mathbb{N} \to \mathbb{N}$. For each τ, we define a protocol π_τ as follows. First, the n players vote. Then, while there is no $\tau(n')$-super-majority among the remaining n' live players, they communicate to decide on a player to bow out of the protocol. The protocol ends when a super-majority of the remaining votes is achieved. The protocol is defined more formally in Figure 5.8.

Finding a closed-form expression for the optimal τ appears to be nontrivial; however, for small n (we used $n = 1200$), we can compute τ and the resulting $\mathrm{succ}(\pi_\tau)$ easily using dynamic programming: let $v(i, j)$ denote the probability of winning if there are currently i players voting 0 and j players voting 1 left and we

Parameter:

$n \geq 23$: number of players we wish to use

m : number of rounds. The higher it is, the better is the resulting protocol

Input distribution: (X, L) uniformly distributed on $\{0,1\} \times [n]$. Each PLR_i learns if $L = i$ and if it is, she also learns X.
Protocol:

1. Each player PLR_i choose a number r_i. If $L \neq i$, r_i is chosen uniformly at random from $[m]$, if $L = i$ then $r_i = m + 1$
2. For time t from 1 to m
3. For i from 1 to n
4. If $r_i = t$ and less than $n - 23$ messages have been send, player PLR_i sends the message "I am not a leaker"
5. If more than $n-23$ players have send a message the protocol aborts, otherwise
6. For i from 1 to 23
7. If the i'th of the 23 players who have not sent a message is a leaker, she sends X otherwise she sends a message chosen uniformly at random from $\{0,1\}$

Figure 5.7: Protocol proving Corollary 5.9.

stop at the optimal time. Clearly $v(i,0) = v(0,i) = 1 - \frac{1}{i}$. Furthermore, for $i, j > 0$ we can choose to stop or continue by eliminating one more person. If we stop, we should guess that the majority is correct. This guess will be correct with probability $\frac{\max(i,j)}{i+j}$, and when it is, Eve will guess the leaker with probability $\max(i,j)^{-1}$. Thus, the probability of winning given that we stop is $\frac{\max(i,j)\left(1-\frac{1}{\max(i,j)}\right)}{i+j}$. If we eliminate one more player, there is probability $\frac{i}{i+j}$ that we eliminate a player voting 0, in which case we win with probability $v(i-1,j)$ and there is probability $\frac{j}{i+j}$ that we eliminate a player voting 1, in which case we win with probability $v(i,j-1)$. This

Parameter:

n : number of players

$\tau : \mathbb{N} \to \mathbb{N}$: function that determines when we stop eliminating players

m : number of rounds. The higher it is, the better is the resulting protocol

Input distribution: (X, L) uniformly distributed on $\{0,1\} \times [n]$. Each PLR_i learns if $L = i$ and if it is, she also learns X.
Protocol:

1. For i from 1 to n
2. If $L = i$, PLR_i sends X, otherwise PLR_i sends a message chosen uniformly from $\{0,1\}$
3. Each player PLR_i choose a number r_i. If $L \neq i$, r_i is chosen uniformly at random from $[m]$, if $L = i$ then $r_i = m + 1$
4. For time t from 1 to m
5. Let n' denote the number of people who have not send the message "I am not a leaker". For $b \in \{0,1\}$ let n'_b denote the number of these player who send b in their first message
6. If $\max(n'_0, n'_1) < \tau(n')$
7. For i from 1 to n
8. If $r_i = t$ player PLR_i sends the message "I am not a leaker"

Figure 5.8: Protocol proving Theorem 5.10.

gives us a recurrence relation:

$$v(i,j) = \max\left(\frac{\max(i,j)\left(1 - \frac{1}{\max(i,j)}\right)}{i+j}, \frac{iv(i-1,j) + jv(i,j-1)}{i+j}\right), \text{for } i,j > 0.$$

Together with the values for $i = 0$ and for $j = 0$ this formula can be used to compute $v(i,j)$ for any i,j. Once we have $v(i,j)$ for all i,j with $i + j = n$ we can compute

$\text{succ}(\pi_\tau)$ for this protocol:

$$\text{succ}(\pi_\tau) = \frac{\sum_{k=0}^{n} \binom{n}{k} v(k, n-k)}{2^n}.$$

Along with Lemma 5.8, this gives a protocol with success probability greater than 0.5644.

Theorem 5.10. *For all $n \geq 1200$, there exists an n-player cryptogenography protocol π with* $\text{succ}(\pi) > 0.5644$.

5.3 Hardness results

In this section, we give upper bounds on the best possible success probability both in the two-player and general case. We start with a high-level description of our approach.

In Section 5.2 we gave several protocols achieving high success probability under the uniform distribution on inputs (X, J). In this section, it will be helpful to consider the space of all possible input distributions. Let $\Delta(\{0,1\} \times [n])$ denote the set of all possible distributions on (X, L). Given a (partial) communication transcript $t \in \{0,1\}^k$ of a known protocol π, define μ_t to be the input distribution μ, conditioned on the first k bits of communication equaling t. Our motivation here is two-fold: first, examining general distributions allows us to appeal to the geometry of $\Delta(\{0,1\} \times [n])$. In particular, we show that the success of a protocol satisfies certain concavity conditions when viewed as a function $s : \Delta(\{0,1\} \times [n]) \to [0,1]$ over the distribution space. Second, our arguments will examine how a protocol π affects μ_t. We show that μ is a convex combination of $\{\mu_t\}$. We are particularly interested in how μ "splits" into distributions μ_0 and μ_1; i.e., we look at convex combinations on conditional distributions one bit at a time. Importantly, we show that for each player PLR_i, the set of all possible distributions obtainable by splitting μ forms a plane in $\Delta(\{0,1\} \times [n])$; we call this the PLR_i*-allowed plane through* μ. Any plane, that is an allowed plane through μ' for some distribution μ' is called an *allowed plane*. Our first lemma characterizes the possible distribution splits made by a cryptogenography protocol.

Lemma 5.11. *Let π be a protocol where only one message gets sent, this message is in $\{0,1\}$, and this message is sent by* PLR_i*. If π is used with prior distribution μ, let $\nu(t)$ denote the probability that* PLR_i *sends message t and let μ_t be the distribution given that* PLR_i *sent message t. Then*

1. $\mu = \nu(0)\mu_0 + \nu(1)\mu_1$.
2. *Each μ_t is proportional to μ on $\{0,1\} \times ([n] \setminus \{i\})$.*

Proof. 1: We have

$$\begin{aligned}
\mu(x,i) &= \Pr(X = x, L = i) \\
&= \sum_{t=0}^{1} \Pr(X = x, L = i, T = t) \\
&= \sum_{t=0}^{1} \Pr(T = t) \Pr(X = x, L = i | T = t) \\
&= \sum_{t=0}^{1} \nu(t)\mu_t(x,i).
\end{aligned}$$

2: Let $x' \in \{0,1\}$ and $i' \in [n] \setminus \{i\}$. Then by Bayes' theorem

$$\begin{aligned}
\mu_t(x',i') &= \Pr(X = x', L = i' | T = t) \\
&= \frac{\Pr(T = t | X = x', L = i') \Pr(X = x', L = i')}{\Pr(T = t)} \\
&= \frac{\Pr(T = t | X = x', L = i')}{\Pr(T = t)} \mu(x',i')
\end{aligned}$$

The probability distribution PLR_i used to choose his message only depends on his information, and thus does not depend on (x', i') as long as $i' \neq i$. So $\frac{\Pr(T=t|X=x',L=i')}{\Pr(T=t)}$ is a constant, and μ_t is indeed proportional to μ on $\{0,1\} \times ([n] \setminus \{i\})$ □

The next lemma is the converse of Lemma 5.11. It says that every possible split conforming to the restrictions of Lemma 5.11 are possible in a communication protocol.

Lemma 5.12. *Let* PLR_i *be a player, let* μ, μ_0, μ_1 *be distributions over* $\{0,1\} \times [n]$*, and* ν *a distribution with support* $\{0,1\}$ *such that*

1. $\mu = \nu(0)\mu_0 + \nu(1)\mu_1$.
2. *Each* μ_t *is proportional to* μ *on* $\{0,1\} \times ([n] \setminus \{i\})$.

Then there is a protocol π *where only player* PLR_i *sends messages, he only sends one message, he sends message* $t \in \{0,1\}$ *with probability* $\nu(t)$*, and the posterior probability distribution given that he sends the message* t *is* μ_t*.*

Proof. If PLR_i has the information and it is 0, he should send the message $t \in \{0,1\}$ with probability $\frac{\nu(t)\mu_t(0,i)}{\mu(0,i)}$, if he has the information and it is 1 he should send the message $t \in \{0,1\}$ with probability $\frac{\nu(t)\mu_t(1,i)}{\mu(1,i)}$ and if he does not have the information, he should send message t with probability $\frac{\nu(t)\mu_t(\{0,1\}\times([n]\setminus\{i\}))}{\mu(\{0,1\}\times([n]\setminus\{i\}))}$. By requirement 1, this gives well-defined probability distributions. The following computation shows that the probability of sending t is $\nu(t)$

$$\begin{aligned}
\Pr(T=t) =& \Pr(T=t|(X,L)=(0,i))\mu(0,i) + \Pr(T=t|(X,L)=(1,i))\mu(1,i) \\
&+ \Pr(T=t|L\neq i)\mu(\{0,1\}\times([n]\setminus\{i\})) \\
=& \frac{\nu(t)\mu_t(0,i)}{\mu(0,i)}\mu(0,i) + \frac{\nu(t)\mu_t(1,i)}{\mu(1,i)}\mu(1,i) \\
&+ \frac{\nu(t)\mu_t(\{0,1\}\times([n]\setminus\{i\}))}{\mu(\{0,1\}\times([n]\setminus\{i\}))}\mu(\{0,1\}\times([n]\setminus\{i\})) \\
=& \nu(t)\mu_t(0,i) + \nu(t)\mu_t(1,i) + \nu(t)\mu_t(\{0,1\}\times([n]\setminus\{i\})) \\
=& \nu(t).
\end{aligned}$$

To finish the proof we need to check that the posterior distribution given transcript t is indeed μ_t. Used Bayes' theorem we get

$$\begin{aligned}
\Pr((X,L)=(x,i)|T=t) =& \frac{\Pr(T=t|(X,L)=(x,i))\Pr((X,L)=(x,i))}{\Pr(T=t)} \\
=& \frac{\frac{\nu(t)\mu_t(x,i)}{\mu(x,i)}\mu(x,i)}{\nu(t)} \\
=& \mu_t(x,i)
\end{aligned}$$

as wanted. Similarly, for $l \neq i$ we have

$$\begin{aligned}\Pr((X,L)=(x,l)|T=t) =& \frac{\Pr(T=t|(X,L)=(x,l))\Pr((X,L)=(x,l))}{\Pr(T=t)} \\ =& \frac{\frac{\nu(t)\mu_t(\{0,1\}\times([n]\setminus\{i\}))}{\mu(\{0,1\}\times([n]\setminus\{i\}))}\mu(x,l)}{\nu(t)} \\ =& \frac{\mu_t(\{0,1\}\times([n]\setminus\{i\}))}{\mu(\{0,1\}\times([n]\setminus\{i\}))}\mu(x,l) \\ =& \mu_t(x,l).\end{aligned}$$

Here the last equality follows from requirement 2. □

Instead of playing the cryptogenography game starting from the uniform distribution over $\{0,1\}\times[n]$, we could start from any other distribution μ (and let all the players know that we are starting from distribution μ). Let $\mathrm{succ}(\mu,\pi)$ denote the probability of winning, when using protocol π starting from distribution μ. Let $\mathrm{succ}(\mu) = \sup_\pi \mathrm{succ}(\mu,\pi)$ where the supremum is over all protocols π, and let $\mathrm{succ}_n(\mu) = \sup_{CC(\pi)\leq n} \mathrm{succ}(\mu,\pi)$.

For a distribution μ we now know that the PLR_i-allowed plane through μ, as defined previously, is the set of all distributions μ' that are proportional to μ on $\{0,1\}\times([n]\setminus\{i\})$. We see that this is indeed a plane in the set $\Delta(\{0,1\}\times[n])$ of distributions over $\{0,1\}\times[n]$.

Lemma 5.13. *The function* $\mathrm{succ}:\Delta(\{0,1\}\times[n])\to[0,1]$ *satisfies:*

1. $\mathrm{succ}(\mu)\geq\mathrm{succ}(\mu,\pi_0)$ *where* π_0 *is the protocol where they do not communicate at all.*

2. *For any allowed plane,* succ *restricted to that plane is concave.*

Proof. 1: $\mathrm{succ}(\mu) = \sup_\pi \mathrm{succ}(\mu,\pi) \geq \mathrm{succ}(\mu,\pi_0)$.

2: Whenever μ_0,μ_1 are distributions in the PLR_i-allowed plane, and ν is a distribution with support $\{0,1\}$ such that $\mu = \sum_{t=0}^{1}\nu(t)\mu_t$, Lemma 5.12 says that we can find a protocol where PLR_i sends one message, sends message t with probability $\nu(t)$, and the distribution given that PLR_i sends t is μ_t. For every $\epsilon>0$ we can now construct a protocol π_ϵ such that $\mathrm{succ}(\mu,\pi_\epsilon)\geq\sum_{t=0}^{1}\nu(t)\,\mathrm{succ}(\mu_t)-\epsilon$. The protocol π_ϵ

starts with the one-message protocol we obtain from Lemma 5.12. If the message t is sent, they continue from there, using a protocol π_t with $\text{succ}(\mu_t, \pi_t) \geq \text{succ}(\mu_t) - \epsilon$. The existence of such a protocol follows from the definition of $\text{succ}(\mu_t)$. It is clear that the resulting π_ϵ satisfies the required inequality. As we can do this for all $\epsilon > 0$ we get $\text{succ}(\mu) \geq \sum_{t=0}^{1} \nu(t)\,\text{succ}(\mu_t)$. It follows from the converse of Jensen's inequality that succ is concave in the PLR_i-allowed plane. □

We are now ready for a characterization of succ. This is very similar to a characterization of the information cost discovered independently in the area of information complexity [8, 57, 58].

Theorem 5.14. *The function* $\text{succ} : \Delta(\{0,1\} \times [n]) \to [0,1]$ *is the point-wise smallest function* $s : \Delta(\{0,1\} \times [n]) \to [0,1]$ *that satisfies*

1. $s(\mu) \geq \text{succ}(\mu, \pi_0)$ *where* π_0 *is the protocol where they do not communicate at all.*

2. *For any allowed plane,* s *restricted to that plane is concave.*

Proof. We know from Lemma 5.13 that succ satisfies the two requirements. It is clear that the point-wise infimum of a family of functions satisfying requirement 1 will itself satisfy requirement 1, and similar for requirement 2. Thus, there is a smallest function s^* satisfying both requirements.

Requirement 1 simply says that $s^*(\mu) \geq \text{succ}_0(\mu)$. Assume for induction that $s^*(\mu) \geq \text{succ}_k(\mu)$, and consider a protocol π with $CC(\pi) \leq k+1$. We can view the protocol π as first sending one message $t_1 \in \{0,1\}$ sent by PLR_i (if he can send more than two messages in the first round, we simply let him send one bit of the message at a time), and for each possible message t_1 calling some subsequent protocol π_{t_1} with $CC(\pi_{t_1}) \leq k$. If we let $\nu(t_1)$ denote the probability that PLR_i sends t_1 and let μ_{t_1} denote probability distribution given the PLR_i sends t_1, we know from Lemma 5.11 that all the μ_{t_1}s are in the i-allowed plane through μ and that $\mu = \sum_{t_1=0}^{1} \nu(t_1)\mu_{t_1}$.

So

$$\begin{aligned}\operatorname{succ}(\mu,\pi) &= \sum_{t_1=0}^{1} \nu(t_1)\operatorname{succ}(\mu_{t_1},\pi_{t_1}) \\ &\leq \sum_{t_1=0}^{1} \nu(t_1)\operatorname{succ}_k(\mu_{t_1}) \\ &\leq \sum_{t_1=0}^{1} \nu(t_1)s^*(\mu_{t_1}) \\ &\leq s^*(\mu)\end{aligned}$$

Here the second inequality follows from induction hypothesis, and the last follows from the fact that s^* is concave in the i-allowed plane. As this holds for all π with $CC(\pi) \leq k+1$ we get $\operatorname{succ}_{k+1}(\mu) \leq s^*(\mu)$, and by induction we have $\operatorname{succ}_k \leq s^*$ for all k.

Now $s^*(\mu) \geq \lim_{k\to\infty} \operatorname{succ}_k(\mu) = \operatorname{succ}(\mu)$ but succ satisfies the two requirement in the theorem, and s^* is the smallest function satisfying the two requirements. Thus, $s^* = \operatorname{succ}$. □

This theorem gives us a way to show upper bounds on $\operatorname{succ}(\mu)$: whenever we have a function s satisfying the two requirements, we have $s(\mu) \geq \operatorname{succ}(\mu)$. In the rest of this section we will show upper bounds on succ by guessing such functions s. A function similar to the function s we will use below was suggested by "fedja"[1] and this function was then improved to s by Wadim Zudilin, both on Mathoverflow [46].

Theorem 5.15. *Let μ_2 denote the uniform distribution on $\{0,1\} \times [2]$. Then $\operatorname{succ}(\mu_2) \leq \frac{3}{8}$.*

This bound was published in [9]: later Doerr and Künnemann [5] strengthened the bound from $\frac{3}{8} = \frac{48}{128} = 0.375$ to $\frac{47}{128} < 0.3672$.

Proof. For brevity, write $x_i := \mu(0,i), y_i := \mu(1,i)$ for $i \in \{1,2\}$ being one of the

[1] fedja wishes to stay pseudonymous.

players. Define

$$f(x_1, x_2, y_1, y_2) := x_1^2 + x_2^2 + y_1^2 + y_2^2 - 6(x_1x_2 + y_1y_2) \quad \text{and}$$
$$s(x_1, x_2, y_1, y_2) := \frac{1 - f(x_1, x_2, y_1, y_2)}{4} \ .$$

Proposition 5.16. *Let μ_2 be the uniform distribution on $\{0,1\} \times [2]$. Then $s(\mu_2) = \frac{3}{8}$.*

The proof is a simple calculation.

Lemma 5.17. *The function s is concave on all allowed planes.*

Proof. By the symmetry of s, it is enough to show that s is concave on all PLR_1-allowed planes. Let μ be a distribution and let $(\mu_v)_{v\in\mathbb{R}}$ be a line in a PLR_1-allowed plane through μ (let us say we get μ at $v = 0$). We show that f is convex (and thus s is concave) along this line. Since $(\mu_v)_{v\in\mathbb{R}}$ is an allowed line, the values $(x_2(v), y_2(v))$ will be proportional to (x_2, y_2) throughout.

First we handle the case that $(x_2(v), y_2(v)) = (x_2, y_2)$. That is, PLR_1's message does not change the probabilities involving PLR_2. In words, she talks only about the value of her bit, not about whether she knows it or not. In this case we can assume that

$$\mu_v = (x_1 + v, x_2, y_1 - v, y_2) \ .$$

Now $f(\mu_v)$ is a quadratic polynomial in v with leading monomial $2v^2$, and thus is convex.

From now on, we assume that $(x_2(v), y_2(v)) \neq (x_2, y_2)$ unless $v = 0$. Let $b := x_2 + y_2$ be the probability that PLR_2 has the bit. Note that $b > 0$, because the case $b = 0$ would mean $(x_2(t) = y_2(t)) = (0, 0)$ throughout, and we have handled this case already above. Now μ_v is of the form

	0	1
plr_1	$x_1 + cvb$	$y_1 + \bar{c}vb$
plr_2	$x_2(1-v)$	$y_2(1-v)$

where c is a parameter that describes the "slope" of the line $(\mu_v)_{v\in\mathbb{R}}$, and $\bar{c} := 1 - c$. Again, $v = 0$ recovers the original distribution μ. Again, $f(\mu_v)$ is quadratic in v,

and the leading monomial is

$$(c^2 + \bar{c}^2)b^2 + x_2^2 + y_2^2 + 6b(cx_2 + \bar{c}y_2). \tag{5.1}$$

We want to show that this is non-negative. It is quadratic in c^2 with leading monomial $2b^2c^2$ (note that $\bar{c}^2 = 1 - 2c + c^2$). Thus, (5.1) is minimized when the derivative with respect to c is 0:

$$\begin{aligned}
\frac{\partial(5.1)}{\partial c} = 2cb^2 - 2\bar{c}b^2 + 6b(x_2 - y_2) = 0 \;\Leftrightarrow \\
cb - (1-c)b + 3(x_2 - y_2) = 0 \Leftrightarrow \\
2cb - b + 3(x_2 - y_2) = 0 \Leftrightarrow \\
2cb + 2x_2 - 4y_2 = 0 \Leftrightarrow \\
cb = 2y_2 - x_2. \qquad (5.2)
\end{aligned}$$

To get to the second line we use the assumption that $b > 0$ and divide by $2b$, and to get to the fourth line we use $b = x_2 + y_2$. From $cb + \bar{c}b = b = x_2 + y_2$ and (5.2) we get $\bar{c}b = 2x_2 - y_2$. Plugging these values of cb and $\bar{c}b$ into (5.1), we obtain

$$\begin{aligned}
(c^2 + \bar{c}^2)b^2 + x_2^2 + y_2^2 + 6b(cx_2 + \bar{c}y_2) = \\
(cb)^2 + (\bar{c}b)^2 + x_2^2 + y_2^2 + 6x_2(bc) + 6y_2(\bar{c}b) = \\
(2y_2 - x_2)^2 + (2x_2 - y_2)^2 + x_2^2 + y_2^2 + 6x_2(2y_2 - x_2) + 6y_2(2x_2 - y_2) = 16x_2y_2 \geq 0.
\end{aligned}$$

This shows that f, and hence s, is convex on all allowed planes. □

Lemma 5.18. *Let $\mu \in \Delta(\{0,1\} \times [2])$ be a distribution and let π_0 be the empty protocol, i.e., the one without any communication. Then $s(\mu) \geq \mathrm{succ}(\mu, \pi_0)$.*

Proof. First, let us compute $\mathrm{succ}(\mu, \pi_0)$. Since there is no communication, G only depends on μ. If G = 0, then Eve guesses the player j that maximizes x_j. If G = 1, she maximizes y_j. Thus, $\mathrm{succ}(\mu, \pi_0) = \max(\min(x_1, x_2), \min(y_1, y_2))$. Let us show that $s(\mu) \geq \min(x_1, x_2)$. The proof that $s(\mu) \geq \min(y_1, y_2)$ will be symmetric. We introduce the shorthand $s_x := x_1 + x_2$ and $m_x := \max(x_1, x_2)$, and similarly for y.

So $\min(x_1, x_2) = s_x - m_x$.

$$s \geq \min(x_1, x_2) \Leftrightarrow 1 - f - 4\min(x_1, x_2) \geq 0 \Leftrightarrow 1 - 4s_x + 4m_x - f \geq 0 \ . \quad (5.3)$$

Let us bound f from above:

$$\begin{aligned} f(x_1, x_2, y_1, y_2) &= x_1^2 + x_2^2 + y_1^2 + y_2^2 - 6(x_1x_2 + y_1y_2) \\ &= 4(x_1^2 + x_2^2) + 4(y_1^2 + y_2^2) - 3(x_1 + x_2)^2 - 3(y_1 + y_2)^2 \\ &= 4(x_1^2 + x_2^2) + 4(y_1^2 + y_2^2) - 3s_x^2 - 3s_y^2 \\ &\leq 4s_xm_x + 4s_ym_y - 3s_x^2 - 3s_y^2 \ . \end{aligned}$$

Let us combine this with (5.3):

$$\begin{aligned} 1 - 4s_x + 4m_x - f &\geq 1 - 4s_x + 4m_x - 4m_xs_x - 4s_ym_y + 3s_x^2 + 3s_y^2 \\ &= (1 - s_x)(1 - 3s_x) + 4m_x(1 - s_x) - 4m_ys_y + 3s_y^2 \\ &= s_y(1 - 3s_x + 4m_x - 4m_y + 3s_y) \qquad \text{(note that } 1 - s_x = s_y\text{)} \\ &\geq s_y(1 - 3s_x + 2s_x - 4s_y + 3s_y) \quad \text{(since } m_x \geq \tfrac{s_x}{2} \text{ and } m_y \leq s_y\text{)} \\ &= s_y(1 - s_x - s_y) = 0 \ . \end{aligned}$$

This shows that $s(\mu) \geq \min(x_1, x_2)$. Together with $s(\mu) \geq \min(y_1, y_2)$, which can be show in a similar way, this implies $s(\mu) \geq \max(\min(x_1, x_2), \min(y_1, y_2)) = \mathrm{succ}(\mu, \pi_0)$ and proves the lemma. □

By Theorem 5.14 this implies that $\mathrm{succ}(\mu_2) \leq s(\mu_2) = \frac{3}{8}$. □

This upper bound in the two player case was later improved by Doerr and Künnemann to 0.3672 [5].

Our final theorem generalizes the above argument to n players.

Theorem 5.19. *Let μ_n denote the uniform distribution on $\{0, 1\} \times [n]$. Then $\mathrm{succ}(\mu_n) \leq \frac{3}{4} - \frac{1}{2n}$.*

Proof. For brevity, we denote by x_i the probability that player i has the bit, and it

is 0, that is, $x_i := \mu(0, i)$. Similarly, $y_i := \mu(1, i)$. We define

$$f(\vec{x}, \vec{y}) := 2 \|\vec{x}\|_2^2 + 2 \|\vec{y}\|_2^2 - \|x\|_1^2 - \|y\|_1^2 \ .$$

where $\|\vec{x}\|_p := (\sum_{i=1}^n x_i^p)^{1/p}$ and define

$$s_n(\vec{x}, \vec{y}) := \frac{1 - f(\vec{x}, \vec{y})}{2} \ .$$

We will prove three things. First, $s_n(\mu_n) = \frac{3}{4} - \frac{1}{2n}$. Second, $s_n(\mu) \geq \mathrm{succ}(\mu, \pi_0)$, where π_0 is the "empty" protocol without any communication. Third, s_n is concave along allowed planes. This will conclude the proof.

Proposition 5.20. *Let μ_n be the uniform distribution on $\{0, 1\} \times [n]$. Then $s_n(\mu_n) = \frac{3}{4} - \frac{1}{2n}$.*

Proof. Every (x, l) has probability $\frac{1}{2n}$. Therefore, $\|\vec{x}\|_2^2 = \|\vec{y}\|_2^2 = n \cdot \left(\frac{1}{2n}\right)^2$ and $\|\vec{x}\|_1^2 = \|\vec{y}\|_1^2 = \left(\frac{1}{2}\right)^2 = \frac{1}{4}$. Thus, $f(\mu_n) = 4n \cdot \left(\frac{1}{2n}\right)^2 - 2 \cdot \left(\frac{1}{2}\right)^2 = \frac{1}{n} - \frac{1}{2}$, and $s_n(\mu_n) = \frac{3}{4} - \frac{1}{2n}$. □

Proposition 5.21. *Let $\mu \in \Delta(\{0, 1\} \times [n])$ be a distribution and let π_0 be the empty protocol, i.e., the one without any communication. Then $s_n(\mu) \geq \mathrm{succ}(\mu, \pi_0)$.*

Proof. What is $\mathrm{succ}(\mu, \pi_0)$? The transcript of π_0 is empty, thus $\mathrm{G}(T)$ only depends on μ. If $\mathrm{G} = 0$, then Eve optimally guesses the player i that maximizes x_i, and the success probability for the players is $\mu(0, [n]) - \max_i \mu(0, i)$. Similarly, if $\mathrm{G} = 1$, she chooses the i maximizing y_i, and the success probability is $\mu(1, [n]) - \max_i \mu(1, i)$. Thus, the success probability of π_0 is

$$\max\Big(\mu(0, [n]) - \max_i \mu(0, i), \quad \mu(1, [n]) - \max_i \mu(1, i)\Big) \ .$$

For brevity, we define $m_x := \max_i x_i = \max_i \mu(0, i)$, $m_y := \max_i y_i = \max_i \mu(1, i)$, $s_x := \sum_i x_i = \mu(0, [n])$, and $s_y := \sum_i y_i = \mu(1, [n])$. We want to show that $s_n(\mu) \geq \mathrm{succ}(\mu, \pi_0) = \max(s_x - m_x, s_y - m_y)$. We will show that $s_n(\mu) \geq s_x - m_x$. The

inequality $s_m(\mu) \geq s_y - m_y$ will follow analogously.

$$s_n(\mu) \geq s_x - m_x \iff \frac{1 - f(\vec{x}, \vec{y})}{2} \geq s_x - m_x \iff 1 - 2s_x + 2m_x - f(\vec{x}, \vec{y}) \geq 0 \ . \tag{5.4}$$

Let us bound $f(\vec{x}, \vec{y})$ from above:

$$f(\vec{x}, \vec{y}) = 2 \left\|\vec{x}\right\|_2^2 + 2 \left\|\vec{y}\right\|_2^2 - \|x\|_1^2 - \|y\|_1^2 \leq 2m_x s_x + 2m_y s_y - s_x^2 - s_y^2 \ .$$

Thus, we evaluate (5.4):

$$\begin{aligned} 1 - 2s_x + 2m_x - f(\vec{x}, \vec{y}) &\geq 1 - 2s_x + 2m_x - 2m_x s_x - 2m_y s_y + s_x^2 + s_y^2 \\ &= 1 - 2s_x + s_x^2 + s_y^2 + 2m_x(1 - s_x) - 2m_y s_y \\ &= (1 - s_x)^2 + s_y^2 + 2m_x(1 - s_x) - 2m_y s_y \\ &= 2s_y^2 + 2m_x s_y - 2m_y s_y \\ &= 2s_y(s_y + m_x - m_y) \geq 0 \ . \end{aligned}$$

In the penultimate equality we use $1 - s_x = s_y$ and the last inequality follows from $s_y - m_y \geq 0$. Replacing the roles of x and y, a similar calculation shows that $s_n(\mu) \geq s_y - m_y$, and thus $s_n(\mu) \geq \max(s_x - m_x, s_y - m_y) = \mathrm{succ}(\mu, \pi_0)$. □

Proposition 5.22. *For any allowed plane, s_n restricted to that plane is concave.*

Proof. By symmetry, we can restrict ourselves to PLR_1-allowed planes. That is, all distributions μ' that are proportional to μ on $\{0, 1\} \times ([n] \setminus \{1\})$. Let μ be any distribution and let $(\mu_v)_{v \in \mathbb{R}}$ be a line through μ that is contained in a PLR_1-allowed plane. It suffices to show that s_n is concave along all such lines.

First suppose that in our line, each μ_v is not only proportional to μ on $\{0, 1\} \times$

$([n] \setminus \{1\})$, but actually identical to it. Then μ_v looks as follows:

	0	1
plr_1	$x_1 + v$	$y_1 - v$
plr_2	x_2	y_2
$\vdots$	$\vdots$	$\vdots$
plr_n	x_n	y_n

(5.5)

and $f(\mu_v)$ is quadratic in v with leading monomial $2v^2$. Therefore, it is convex, and s_n is concave, along $(\mu_v)_{v\in\mathbb{R}}$.

Suppose from now on that μ_v is not identical to μ on $\{0,1\} \times ([n] \setminus \{1\})$. How does a line $(\mu_v)_{v\in\mathbb{R}}$ through μ in a PLR_1-allowed plane look? The probabilities $x_2, \ldots, x_n$ and $y_2, \ldots, y_n$ get multiplied by a factor $(1 - v)$. Let $b_0 := x_2 + \cdots + x_n$, $b_1 := y_2 + \cdots + y_n$, and $b = b_0 + b_1$. Note that $b > 0$, otherwise all μ_v are 0 on $\{0,1\} \times ([n] \setminus \{1\})$, and this belongs to the above case. The distribution μ_v on the PLR_1-allowed plane containing μ has the form

	0	1
plr_1	$x_1 + cvb$	$y_1 + \bar{c}vb$
plr_2	$x_2(1 - v)$	$y_2(1 - v)$
$\vdots$	$\vdots$	$\vdots$
plr_n	$x_n(1 - v)$	$y_n(1 - v)$

(5.6)

where $c \in \mathbb{R}$ is some parameter specific to the line $(\mu_v)_{v\in\mathbb{R}}$, and $\bar{c} := 1 - c$. For fixed $\vec{x}, \vec{y}, c$, all μ_v lie on a line. It remains to show that f is convex along this line. We evaluate $f(\mu_v)$, which is a quadratic polynomial in v, and analyze the coefficient of the monomial v^2: In the terms $\|\vec{x}\|_2^2$, $\|\vec{y}\|_2^2$, $\|\vec{x}\|_1^2$, $\|\vec{y}\|_1^2$, evaluated at μ_t, the monomial

v^2 has the following coefficients:

$$\begin{aligned}
\|\vec{x}\|_2^2 &\longrightarrow c^2b^2 + x_2^2 + \cdots + x_n^2 \geq c^2b^2 \\
\|\vec{y}\|_2^2 &\longrightarrow \bar{c}^2b^2 + y_2^2 + \cdots + y_n^2 \geq \bar{c}^2b^2 \\
\|\vec{x}\|_1^2 &\longrightarrow (cb - x_2 - \cdots - x_n)^2 = (cb - b_0)^2 = c^2b^2 - 2cbb_0 + b_0^2 \\
\|\vec{y}\|_1^2 &\longrightarrow (\bar{c}b - y_2 - \cdots - y_n)^2 = (\bar{c}b - b_1)^2 = \bar{c}^2b^2 - 2\bar{c}bb_1 + b_1^2
\end{aligned}$$

Thus, the coefficient of v^2 of $f(\vec{x}, \vec{y}) = 2\|\vec{x}\|_2^2 + 2\|\vec{y}\|_2^2 - \|x\|_1^2 - \|y\|_1^2$ is at least

$$\begin{aligned}
2c^2b^2 + 2\bar{c}^2b^2 - (c^2b^2 - 2cbb_0 + b_0^2) - (\bar{c}^2b^2 - 2\bar{c}bb_1 + b_1^2) \\
= b^2(c^2 + \bar{c}^2) - b_0^2 - b_1^2 + 2b(cb_0 + \bar{c}b_1) \ . \qquad (5.7)
\end{aligned}$$

It remains to show that this is non-negative. Recall that b is the probability that PLR_1 does not know the bit. Since we assume $b > 0$, the expression in (5.7) is quadratic in c with leading monomial $2b^2c^2$ (note that $\bar{c}^2 = (1-c)^2 = 1 - 2c + c^2$). Thus, (5.7) is minimized if its derivate with respect to c is 0:

$$\frac{\partial(5.7)}{\partial c} = 2b^2(c - \bar{c}) + 2b(b_0 - b_1) = 2b^2(2c - 1) + 2b(b - 2b_1) = 4b^2c - 4bb_1 \ .$$

This is 0 if and only if $c = \frac{b_1}{b}$. At that point, $\bar{c} = \frac{b_0}{b}$. In particular, $c, \bar{c} \geq 0$. This is not a priori clear, since c is a parameter of the line (μ_t), not a probability. Let us evaluate (5.7) at $c = \frac{b_1}{b}$:

$$\begin{aligned}
(5.7) &= b^2(c^2 + \bar{c}^2) - b_0^2 - b_1^2 + 2b(cb_0 + \bar{c}b_1) \\
&\geq b^2(c^2 + \bar{c}^2) - b_0^2 - b_1^2 \\
&= b^2\left(\left(\frac{b_1}{b}\right)^2 + \left(\frac{b_0}{b}\right)^2\right) - b_0^2 - b_1^2 = 0 \ .
\end{aligned}$$

This shows that f is convex along the line $(\mu_t)_{t\in\mathbb{R}}$, and thus on whole PLR_1-allowed plane containing μ. Thus, s_n is concave along those planes, which proves the proposition. □

□

5.4 Multiple leakers

We now move on to look at the problem with more than one leaker and more than one bit of information X to be leaked. It is obvious how to generalize X to more information, we simply take X to be uniformly distributed on $\{1, \ldots, 2^{\lceil h \rceil}\}$. It is less obvious how to generalize to more leakers. When more people are leaking, it would be unreasonable to require Eve to guess all the leakers. If this was the rule, one of the leaking players could just reveal himself as a leaker and say what X is, while the rest of the leakers and all the non-leakers send empty messages. Instead, we let Eve guess at one person and if that person is leaking, she wins.

Definition 5.1. For fixed values of h, number of leakers l and number of communicating players $n > l$ and a collaborating cryptogenography protocol π, we let $\mathrm{Succ}(h, l, n, \pi)$ denote the probability that after the players communicate using protocol π, Joe will guess the correct value of $X \in [2^{\lceil h \rceil}]$ but Eve's guess will not be a leaker, assuming that Joe and Eve each guess using the strategy that maximizes their own chance of winning and that Eve learns Joe's guess before guessing herself. We define

$$\mathrm{Succ}(h, l, n) = \sup_{\pi}(\mathrm{Succ}(h, l, n, \pi)),$$

where the supremum is over all collaborating cryptogenography protocols π. Finally, we define

$$\mathrm{Succ}(h, l) = \lim_{n \to \infty} \mathrm{Succ}(h, l, n).$$

In this section we will investigate the asymptotic behavior of $\mathrm{Succ}(h, l)$ when at least one of l and h tends to infinity. First we need some propositions. The first proposition implies that the limit, which defines $\mathrm{Succ}(h, l)$, really exists.

Proposition 5.23. $\mathrm{Succ}(h, l, n)$ *is non-decreasing in* n.

Proof. We use the elimination strategy used in the proof of Lemma 5.8. Let $n' > n$ and let π be a protocol for parameters h, l, n. We now construct a sequence of protocols π'_m for parameters h, l, n'. In the protocol π'_m each non-leaking player

thinks of a uniformly chosen number in $\{1, \ldots, m\}$. First everyone who thought of the number 1 announce that and they are out, then everyone who thought of the number 2 and so on, until only n players are left. If two or more players thought of the same number, we might end up with less than n players left. In that case the leakers just announce themselves. If we are left with exactly n players, we know that the l leakers are still among them, and we have no further information about who they are. They then use protocol π, and win with probability $\mathrm{Succ}(h, l, n)$. For a formal definition of the protocol, see Figure 5.9. As $m \to \infty$, the probability that two players thought of the same number tends to 0, so $\mathrm{Succ}(h, l, n', \pi'_m) \to \mathrm{Succ}(h, l, n, \pi)$. □

Proposition 5.24. $\mathrm{Succ}(h, l, n)$ *and* $\mathrm{Succ}(h, l)$ *are non-increasing in* h*.*

Proof. Let $h > h'$ and let π be a protocol for parameters h, l, n and let the secret be denoted X. We construct a protocol π' with parameters h', l, n and secret denoted by X'. In the first round of π', PLR_1 announces $h - h'$ independent and uniformly chosen bits Y, and from then on, everyone follows protocol π for $X = X' \circ Y$. This protocol is more formally defined in Figure 5.10. It is clear that $\mathrm{Succ}(h, l, n, \pi) \leq \mathrm{Succ}(h', l, n, \pi')$. □

Proposition 5.25. *The probability that the communicating players wins the game does not change if Eve is told the value of* X *before they start to communicate.*

Proof. If Joe guesses the correct value of X, Eve was going to assume that that was the correct value anyway (as she wants to maximize the probability that she is correct given that Joe was correct), and if Joe guesses wrong, she would win anyway. □

In the rest of this section, we will assume that Eve knows the value of X. We can now prove a lower bound on Succ as l and h tends to infinity. The proof uses Corollary 3.21, so it is not constructive.

Theorem 5.26. *For all* $p \in (0, 1)$*,*

$$\liminf_{l \to \infty} \mathrm{Succ}\left(\left\lceil \left(\frac{-\log(p)}{1-p} - \log(e)\right) l \right\rceil, l\right) \geq p.$$

Parameter:

n : number of players we wish to use

$n' \leq n$: number of players in existing protocol

$l \leq n'$: number of leakers

m : number of rounds. The higher it is, the better is the resulting protocol

π : protocol that allows l leakers among a total of n' players to reveal h bits

Input distribution: X, L are independently distributed, X is uniformly distributed on $\{1, \ldots 2^{\lceil h \rceil}\}$ and L is uniformly distributed on all vectors $\{0,1\}^n$ with exactly l 1's. Each PLR_i learns if L_i and if it $L_i = 1$ she also learns X.
Protocol:

1. Each player PLR_i choose a number r_i. If $L_i = 0$, r_i is chosen uniformly at random from $[m]$, if $L_i = 1$ then $r_i = m + 1$
2. For time t from 1 to m
3. For i from 1 to n
4. If $r_i = t$ and less than $n - n'$ messages have been send, player PLR_i sends the message "I am not a leaker"
5. If more than $n - n'$ players have send a message the protocol aborts, otherwise
6. The n' players who have not sent a message now follow π

Figure 5.9: Protocols from proof of Proposition 5.23.

Proof. We know from Corollary 3.21 that the safe (Fixed, b_m)-capacity is $\frac{-\log(1-c)}{c} - \log(e)$. If we let $\epsilon > 0$, and use this corollary for $b_m = 1 - p + \epsilon/2$ we get that for sufficiently high l, n and $h = \left\lceil \left(\frac{-\log(p)}{1-p} - \log(e) \right) l \right\rceil$ there is a protocol π that will ensure that Joe's probability of guessing wrong is at most $\epsilon/2$, and seen from Eve's perspective, no one is leaking with probability greater than $1 - p + \epsilon/2$. By the union bound, the probability that Joe is wrong or Eve is correct[2] is at most

[2]Here we assume that Joe guesses on the most likely value of X, and we allow Eve to use any strategy. It could be that Joe could do better by taking Eve's guess into account, but he is guaranteed at least this probability of winning.

Parameter:

n : number of players

l : number of leakers

h : number of bits revealed in original protocol

h' : number of bits we wish to reveal

π : protocol that allows l leakers among a total of n' players to reveal h bits

Input distribution: X', L are independently distributed, X' is uniformly distributed on $\{1, \dots 2^{\lceil h \rceil}\}$ and L is uniformly distributed on all vectors $\{0,1\}^n$ with exactly l 1's. Each PLR_i learns if L_i and if it $L_i = 1$ she also learns X'.
Protocol:

1. PLR_1 sends a message Y chosen uniformly from $\{0,1\}^{h-h'}$.
2. Everyone follows π as if the secret to be revealed is $X = X' \circ Y$

Figure 5.10: Protocols from proof of Proposition 5.24.

$\epsilon/2 + 1 - p + \epsilon/2$, thus the communicating players win with probability at least $p - \epsilon$. □

In particular we have the following corollary.

Corollary 5.27. *Let $l \to \infty$ and $h = h(l)$ be a function of l with $h = o(l)$. Then* $\text{Succ}(h, l) \to 1$.

Proof. Let $h(l) = o(l)$ be a function. For each l, we have $\text{Succ}(h(l), l) \in [0,1]$, so we only need to show that for any $\epsilon > 0$ there exists l_0 such that for all $l \geq l_0$ we have $\text{Succ}(h(l), l) \geq 1 - 2\epsilon$. If we put $p = 1 - \epsilon$ in Theorem 5.26 we get

$$\liminf_{l \to \infty} \text{Succ}\left(\left\lceil \left(\frac{-\log(1-\epsilon)}{\epsilon} - \log(e) \right) l \right\rceil, l\right) \geq 1 - \epsilon.$$

This means that there is some l_1 such that for all $l \geq l_1$ we have

$$\text{Succ}\left(\left\lceil \left(\frac{-\log(1-\epsilon)}{\epsilon} - \log(e) \right) l \right\rceil, l\right) \geq 1 - 2\epsilon.$$

As $h(l) = o(l)$, there must be some l_2 such that $h(l) \leq \left(\frac{-\log(1-\epsilon)}{\epsilon} - \log(e)\right) l$ for all $l \geq l_2$. Now define $l_0 = \max(l_1, l_2)$. For all $l \geq l_0$ we have

$$\begin{aligned}\mathrm{Succ}(h(l), l) \geq & \mathrm{Succ}\left(\left\lceil\left(\frac{-\log(1-\epsilon)}{\epsilon} - \log(e)\right) l\right\rceil, l\right) \\ \geq & 1 - 2\epsilon.\end{aligned}$$

Here the first inequality uses Proposition 5.24 and $l \geq l_0 \geq l_2$ and the second inequality uses that $l \geq l_0 \geq l_1$. □

Next we want to show upper bounds on $\mathrm{Succ}(h, l)$. In order to do that, we will need to be able to modify a protocol, and say that the resulting protocol is *equivalent*. To do that, we will use the following definition of equivalence of protocols, which will also be useful in the next chapter.

Definition 5.2. Let the distribution of $(X, L_1, \ldots, L_n)$ be given and let π be a protocol with transcript T and $\bar{\pi}$ a protocol with transcript $\bar{T}$. For a transcript t of π let μ_t denote the distribution $(X, L_1, \ldots, L_n)|_{T=t}$, and similar for transcripts $\bar{t}$ of $\bar{\pi}$. We say that π and $\bar{\pi}$ are *equivalent for* $(X, L_1, \ldots, L_n)$ (or just *equivalent* when it is clear what the distribution of $(X, L_1, \ldots, L_n)$ is) if the distribution of μ_T is the same as the distribution of $\mu_{\bar{T}}$.

Notice that for fixed t, μ_t is a distribution of $(X, L_1, \ldots, L_n)$, so μ_T is a random variable those values are themselves distributions over $(X, L_1, \ldots, L_n)$. For π and π' to be equivalent, we require the probability that the posterior distribution of $(X, L_1, \ldots, L_n)$ is μ to be the same for both π and π'. We have the following proposition.

Proposition 5.28. *If μ and μ' are two distributions of $(X, L_1, \ldots, L_n)$ with the same support, then π and $\bar{\pi}$ are equivalent for μ if and only if they are equivalent for μ'.*

Proof. Let $(X, L) = (X, L_1, \ldots, L_n)$ be random variables with distribution given by μ, and (X', L') have the distribution given by μ'. Furthermore, let $\bar{T}$ be the distribution of the transcript when X, L are given by μ following protocol $\bar{\pi}$. Similarly, we

define T, T' and $\bar{T}'$ the obvious way. By symmetry it is enough to prove one of the implications in the proposition. Assume that π and $\bar{\pi}$ are equivalent for μ. That means that μ_T and $\mu_{\bar{T}}$ have the same distribution. As these distributions are over (X, L) that is equivalent to the distributions (X, L, μ_T) and $(X, L, \mu_{\bar{T}})$ being the same. In particular, for any (x, l) in the domain of μ we have $\mu_T|_{(X,L)=(x,l)} \sim \mu_{\bar{T}}|_{(X,L)=(x,l)}$. Thus, we have $(X', L', \mu_{T'}) \sim (X', L', \mu_{\bar{T}'})$, where $\mu_{T'}$ denote the posterior distribution we would have over (X, L) given transcript T' if the prior distribution was still given by μ. Let μ'_T denote the posterior distribution given transcript T if the prior distribution was given by μ'. What we need to show is $(X', L', \mu'_{T'}) \sim (X', L', \mu'_{\bar{T}'})$. We already know $(X', L', \mu_{T'}) \sim (X', L', \mu_{\bar{T}'})$, so to finish the proof, we only need to show that μ'_t is a function of μ_t and that this function does not depend on which protocol we use to produce T.

Using Bayes' Theorem we have

$$\mu_t(x, l) = \Pr((X, L) = (x, l)|T = t) = \frac{\Pr(T = t|(X, L) = (x, l))}{\Pr(T = t)}\mu(x, l) \tag{5.8}$$

and

$$\begin{aligned}
\mu'_t(x, l) =& \Pr((X', L') = (x, l)|T' = t) \\
=& \frac{\Pr(T' = t|(X', L') = (x, l))}{\Pr(T' = t)} \Pr((X', L') = (x, l)) \\
=& \frac{\Pr(T = t|(X, L) = (x, l))}{\Pr(T' = t)}\mu'(x, l) \\
=& \frac{\mu_t(x, l)\mu'(x, l)}{\mu(x, l)} \frac{\Pr(T = t)}{\Pr(T' = t)}
\end{aligned}$$

The last equality we use equation 5.8. This shows that given μ, μ' and μ_T we can compute μ'_T up to the multiplicative constant $\frac{\Pr(T=t')}{\Pr(T'=t')}$. But as μ'_T is a probability measure, it sums to 1, so it is a function of μ, μ' and μ_T. □

Thus, when the support of $(X, L_1, \ldots, L_n)$ is clear, we can simply say equivalent.

Proposition 5.29. *If π and π' are equivalent collaborating cryptogenography protocols, then* $\mathrm{Succ}(h, l, n, \pi) = \mathrm{Succ}(h, l, n, \pi')$.

Proof. The probability of winning given $T = t$ only depends on μ_T. Thus, when

$\mu_T \sim \mu_{T'}$ the probability of winning is the same for π and π'. □

The next lemma shows that we can ensure that before any player crosses probability c of having the bit, seen from Eve's perspective, that player lands on this probability.

Lemma 5.30. *Let π be any collaborating cryptogenography protocol, let $(X, L_1, \ldots, L_n)$ have any distribution and let $c \in (0,1)$. If $\Pr(L_i = 1|X = x) < c$ for all i, x then there exists an equivalent collaborating cryptogenography protocol π' such that when we use it on $(X, L_1, \ldots L_n)$ and let T' denote its transcript, it satisfies: for all $x \in \mathcal{X}$, all PLR_i and all non-empty partial transcripts t'^k, if*

$$\Pr(L_i = 1|T'^k = t'^k, X = x) > c.$$

then there is a $k' < k$ such that

$$\Pr(L_i = 1|T'^{k'} = t'^{k'}, X = x) = c$$

Proof. Let π, $(X, L_1, \ldots, L_n)$ and c be given, and assume that $(x, i) = (x_0, i_0)$ is a counterexample to the requirement from the lemma. We will then construct a protocol π' such that (x_0, i_0) is not a counterexample for π', and any (x, i) that satisfied the requirement for π also satisfy it for π'. By induction, this is enough to prove the lemma.

We can assume that the messages in π are sent one bit at a time. We say a partial transcript t^k is problematic if

$$\Pr(L_{i_0} = 1|T^k = t^k, X = x_0) < c$$

but

$$\Pr(L_{i_0} = 1|T^{k+1} = t^k \circ m, X = x_0) > c.$$

for some bit value m. Without loss of generality, assume that $m = 1$. Let $p = \Pr(T_{k+1} = 1|T^k = t'^k)$.

We will use the c-notation from Section 3.1, so for example

$$c_{t^k, x_0} = \Pr(L_i = 1|T^k = t^k, X = x_0).$$

Now

$$c > c_{t^k,x_0} = pc_{t^k\circ 1,x_0} + (1-p)c_{t^k\circ 0,x_0}$$

so $c_{t^k\circ 0,x_0} < c$. Let $q \in (p,1)$ be the number such that

$$c = qc_{t^k\circ 1,x_0} + (1-q)c_{t^k\circ 0,x_0}.$$

Now we modify π. First, the player PLR_j, who is going to send to $k+1$'th message in π, decides if she would have sent 0 or 1 in π. If she would have sent 1 she sends the bits 11. If she would have sent 0 she sends 10 with probability $\frac{p(1-q)}{q(1-p)} \in (0,1)$, and otherwise she sends 00. In all cases she sends the bits one at a time. They then continue the protocol π as if only the last of the two bits had been sent. The protocol is defined more formally in Figure 5.11.

If we let T' denote the transcript of the protocol with this modification, we get

$$c_{T'^{k+1}=t^k\circ 0,x_0} = c_{T^{k+1}=t^k\circ 0,x_0} < c$$

and

$$\begin{aligned} c_{T'^{k+1}=t^k\circ 1,x_0} =& \frac{pc_{T^{k+1}=t^k\circ 1,x_0} + (1-p)\frac{p(1-q)}{q(1-p)}c_{T^{k+1}=t^k\circ 0,x_0}}{p + (1-p)\frac{p(1-q)}{q(1-p)}} \\ =& qc_{T^{k+1}=t^k\circ 1,x_0} + (1-q)c_{T^{k+1}=t^k\circ 0,x_0} \\ =& c. \end{aligned}$$

So if PLR_j sends 11 or 10 in the modified protocol, we land on probability c after the first bit. Let π' be the protocol we get from π by doing this modification for each problematic partial transcript t^k in π. It is clear that π and π' are equivalent, and that any (x,i) that satisfied the requirement before also does so afterwards.

□

Parameters:

n : number of players

h : number of bits being leaked

(X, L) : joint distribution of secret and leakers

c : a probability

(x_0, i_0) : a counterexample to the requirement

π : a protocol where each message is one bit

Input distribution: (X, L) has some distribution over $\mathcal{X} \times \{0,1\}^n$. Each PLR_i learns L_i and if $L_i = 1$ she also learns X.

Protocol:

1. $t := \lambda, k := 0$
2. While t is not a complete transcript of π
3. Let PLR_j be the player to send the next bit in π when the transcript is t
4. If $\Pr(L_{i_0} = 1 | T^k = t, X = x_0) \geq c$ or $\Pr(L_{i_0} = 1 | T^{k+1} = t \circ b, X = x_0) \leq c$ for both $b \in \{0,1\}$
5. Player i choose and send her next message b as she would in π
6. $t := t \circ b$, $k := k + 1$
7. else,
8. let b be the bit such that $\Pr(L_{i_0} = 1 | T^{k+1} = t \circ b, X = x_0) > c$
9. $p := \Pr(T_{k+1} = b | T^k = t, X = x_0)$
10. let q be the number such that $c = qc_{t^k \circ 1, x_0} + (1-q)c_{t^k \circ 0, x_0}$
11. let PLR_j choose her next message b' as she would in π when the transcript is t
12. if $b' = b$ she sends $b' \circ b'$
13. if $b' \neq b$ she sends $b \circ b'$ with probability $\frac{p(1-q)}{q(1-p)}$ and otherwise she sends $b' \circ b'$
14. $t := t \circ b'$, $k := k + 1$ (here everyone knows b' because is always the last bit PLR_j sent)

Figure 5.11: Protocol from the proof of Lemma 5.30.

We are now ready to upper bound Succ.

Lemma 5.31. *For any $c \in (0,1)$ and any h, l, n, π, we have* $\mathrm{Succ}(h,l,n,\pi) \leq 1 - \frac{ch + l\log(1-c) + lc\log(e) - c}{h}$.

Proof. As $\mathrm{Succ}(h,l,n)$ is non-decreasing in n, we can assume that $n > \frac{l}{c}$, so that $\Pr(L_i = 1 | X = x) < c$ at the beginning. By Lemma 5.30 and Proposition 5.29 we can assume that π satisfies the requirement for π' in 5.30.

Let π' be the protocol that starts of as π, but where all the players starts to pretend ignorance (as in the proof of Lemma 3.17) if $\Pr(L_i = 1 | T^k = t^k, X = x) = c$ for some i, current transcript t^k and the true value x of X. This ensures that $\Pr(L_i = 1 | T' = t, X = x) \leq c$ for all i and t. Let T' be the transcript of π'. From Theorem 3.5 we get

$$I(X;T') \leq \left(-\frac{\log(1-c)}{c} - \log(e) \right) l$$

We let Joe guess as he would if we used protocol π. By Fano's inequality, (1.3), Joe's probability of being wrong when he only sees the transcript of π' is

$$\begin{aligned} P_e \geq & \frac{H(X|T') - 1}{\log(|\mathcal{X}|)} \\ = & \frac{H(X) - I(X;T') - 1}{\log(|\mathcal{X}|)} \\ \geq & \frac{h - l\left(\frac{-\log(1-c)}{c} - \log(e)\right) - 1}{h} \end{aligned}$$

In the cases where Joe is wrong in π' there are two possibilities: Either the players did not pretend ignorance, in which case Joe would also be wrong if they used protocol π, or they did pretend ignorance so $\Pr(L_i = 1 | T^k = t^k, X = x) = c$ for some i and some smallest k. When this first happens Eve can just ignore all further messages in π and guess that $\textsc{plr}_i$ is leaking. This way she is wins with probability at least c. Thus, all the situations in π' where Joe guesses wrong, correspond to situations in π where Eve would win with probability at least c. So Eve's probability

of winning when the players are using protocol π is at least

$$cP_e \geq \frac{ch + l\log(1-c) + lc\log(e) - c}{h}.$$

□

The following corollary will be strengthened in Corollary 5.34.

Corollary 5.32. *For fixed l we have*

$$\lim_{h\to\infty} \mathrm{Succ}(h, l) = 0.$$

Proof. By Lemma 5.31 we have $\mathrm{Succ}(h, l) \leq 1 - \frac{ch+l\log(1-c)+lc\log(e)-c}{h}$ for each $c \in (0, 1)$. Setting $c = 1 - \epsilon$ we get

$$\limsup_{h\to\infty} \mathrm{Succ}(h, l) \leq \limsup_{h\to\infty} 1 - \frac{ch + l\log(1-c) + lc\log(e) - c}{h} = \epsilon.$$

As $\mathrm{Succ}(h, l) \in [0, 1]$ and the above holds for all $\epsilon > 0$ we have $\lim_{n\to\infty} \mathrm{Succ}(h, l) = 0$. □

We can now show an upper bound on Succ even when both l and h tend to infinity linearly. This upper bound is illustrated in Figure 5.12

Theorem 5.33. *Let $r > 0$ be a real number. Now*

$$\limsup_{l\to\infty} \mathrm{Succ}(\lfloor r\log(e)l\rfloor, l) \leq \frac{\log(r+1)}{r\log(e)}$$

Proof. Set $c = \frac{r}{r+1}$ and $h = \lfloor r\log(e)l\rfloor$ in Lemma 5.31. Then Eve's probability of winning is at least

$$\frac{r\lfloor r\log(e)l\rfloor - l(r+1)\log(r+1) + lr\log(e) - r}{\lfloor r\log(e)l\rfloor(r+1)}$$

As l tends to infinity, this tends to

$$\frac{r^2\log(e) - (r+1)\log(r+1) + r\log(e)}{r\log(e)(r+1)} = 1 - \frac{\log(r+1)}{r\log(e)}$$

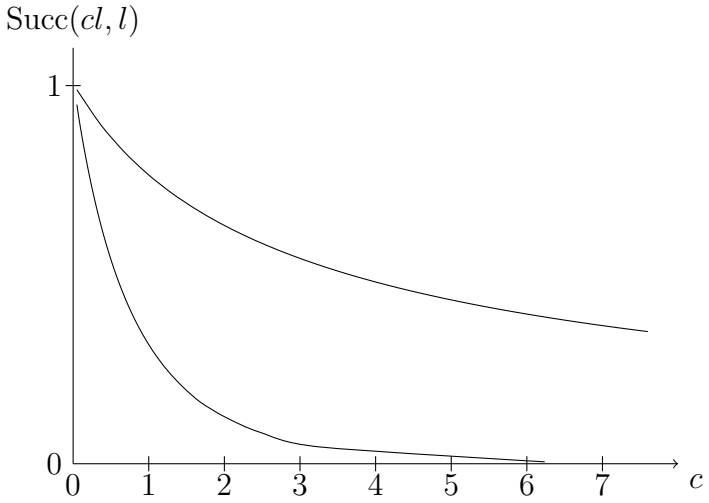

Figure 5.12: The above graph shows our upper and lower bounds (Theorem 5.33 and Theorem 5.26) on $\mathrm{Succ}(cl, l)$ as a function of c for sufficiently large values of l. We have not proved that the limit exists. Both upper and lower bounds are 1 at $c = 0$ and tends to 0 for $c \to \infty$.

as wanted. □

In particular we have the following corollary.

Corollary 5.34. *Let $h \to \infty$ and let $l = l(h)$ be a function of h with $l(h) = o(h)$. Then* $\mathrm{Succ}(h, l) \to 0$.

Proof. Let $l(h) = o(h)$ be a function. As $\mathrm{Succ}(h, l(h)) \in [0, 1]$ for all h, we only need to show that for all $\epsilon > 0$ there exists a h_0 such that for all $h \geq h_0$ we have $\mathrm{Succ}(h, l(h)) \leq 2\epsilon$. We see that $\frac{\log(r+1)}{r \log(e)} \to 0$ as $r \to \infty$, so we can find a number r such that $\frac{\log(r+1)}{r \log(e)} \leq \epsilon$. By Theorem 5.33 we have

$$\limsup_{l \to \infty} \mathrm{Succ}(\lfloor r \log(e) l \rfloor, l) \leq \frac{\log(r+1)}{r \log(e)} \leq \epsilon$$

So there exists a number l_1 such that for all $l \geq l_1$ we have

$$\mathrm{Succ}(\lfloor r \log(e) l \rfloor, l) \leq 2\epsilon.$$

As $l(h) = o(h)$ there is a h_1 such that for all $h \geq h_1$ we have $h \geq r \log(e) l(h)$. By Corollary 5.32 $\lim_{h \to \infty} \mathrm{Succ}(h, l) = 0$ for each value l. So for each value l there is a

$h_2(l)$ such that for all $h \geq h_2(l)$ we have $\mathrm{Succ}(h, l) \leq 2\epsilon$. Define $h_2 = \max_{l<l_1} h_2(l)$. Now define $h_0 = \max(h_1, h_2)$. We want to show that for any $h \geq h_0$ we have $\mathrm{Succ}(h, l(h)) \leq 2\epsilon$. If $l(h) < l_1$, then $h \geq h_0 \geq h_2 \geq h_2(l(h))$, so $\mathrm{Succ}(h, l(h)) \leq 2\epsilon$. If $l(h) \geq l_1$ then

$$\begin{aligned}\mathrm{Succ}(h, l(h)) \leq & \mathrm{Succ}(\lfloor r \log(e) l \rfloor, l) \\ \leq & 2\epsilon.\end{aligned}$$

Here the first inequality follows from $h \geq h_0 \geq h_1$ and Proposition 5.24, and the second inequality follows from $l \geq l_1$. □

Chapter 6

Hiding Among Innocents

Until now, we have assumed that even the players who are not trying to leak information will collaborate. In this chapter we will show that we do not need the non-leakers to collaborate. As long as some people are communicating innocently in a sufficiently non-deterministic way, we can use these people as if they were collaborating.

Formally, we model the innocent communication by an innocent communication protocol. While protocols usually are designed to compute some function, innocent communication protocols is a way of describing what is already going on. An *innocent communication protocol* ι is a protocol that for each possible partial transcript s^k and each player j gives a finite set $\mathcal{M}_{s^k,j}$ of possible messages that that person can send in the next round, and a probability distribution on that set. In innocent communication protocols every person sends a message in each round. This assumption is not a restriction: if we have a protocol where only one player sends messages at a time, we can turn it into an innocent communication protocol, by requiring that all the other players send the empty message with probability 1. Given an innocent communication protocol ι and a cryptogenography protocol π, we will construct a protocol ι^π which achieves the same as π, but where the non-leaker follow ι. We will see that this is possible both for collaborative cryptogenographic protocols (Theorem 6.1) and for the case where there are censors (Theorem 6.3), under slightly different assumptions about ι.

To keep the theorems simple, we will only consider innocent communication

protocols that continue for infinitely many rounds. This assumption is of course unrealistic, but we will see that for any particular protocols ι and π and number $\epsilon > 0$ there is an integer k' such that with probability $1 - \epsilon$ the first k' messages of ι will be enough.

In order to find the protocol ι^π you need have a description of the protocol ι. This is a strong assumption: even if you are able to communicate innocently, it does not mean that you are aware of the distribution you use to pick your random messages. In steganography, the weaker assumption that you have a random oracle that takes history and player index as input and gives a message following the innocent distribution as output, is sometimes enough [41]. However, it is not clear if this weaker assumption is enough for doing cryptogenography. While it may not be possible to find ι for all kinds of innocent communications, there are situations where we can approximate ι very well. For example, if a person posts blog posts, we can consider the message to be only the parity of the minutes in the sending time. This value will probably, for most people, be close to uniformly distributed on $\{0, 1\}$.

6.1 Hiding among innocents without censors

Let S denote the random variable that is the infinite transcript we get from running ι, and let S^k denote the partial transcript of the first k rounds. We say that ι is *informative* if for a random transcript S and for each player j we have $\prod_{k\in\mathbb{N}} \Pr(S_{k,j} = s_{k,j}|S^{k-1} = s^{k-1}) = 0$ with probability 1. Here $S_{k,j}$ is the message sent by player j in round k. In other words, if at each round in the protocol you try to guess what message player j will send in the next round, then with probability 1 you will eventually fail. Notice that the model for innocent communication here is almost the same as the definition of *always informative* in Hopper's PhD thesis [41] when one player is communicating.[1]

We say that a collaborating cryptogenography protocol π is *revealing* if there is a partial transcript t^k and a player PLR_j that is to send the next message A when the transcript is t^k and a message a such that PLR_j will send message a with positive

[1]The differences are that in [41], the communication is only from one player to another and $\prod_{k\in\mathbb{N}} \Pr(S_{k,j} = s_{k,j}|S^{k-1} = s^{k-1})$ has to go to 0 exponentially fast.

probability if $L_j = 1$ but not if $L_j = 0$. If PLR_j did send such a message a, it would reveal her as a leaker. If π is not revealing, we say that it is *non-revealing*.[2] The point in cryptogenography is to hide who is sending the information, so we are only interested in non-revealing protocols.

The following is the main theorem of this section. Notice that in this theorem we will assume that π is collaborative. Thus, the result can be used to strengthen the results in Chapter 3 and Chapter 5[3] to a model where non-leakers are not actively collaborating, but it cannot be used in the case where some players are actively trying to prevent the leakage. In Theorem 6.3 we will show that a similar result holds in that model.

Theorem 6.1. *Let π be a non-revealing collaborating cryptogenography protocol, and let ι be an informative innocent communication protocol. Then there exists a protocol ι^π that is equivalent to π, but where the non-leakers follow the protocol ι.*

Proof. We will define an algorithm that given π and ι constructs a protocol ι^π and at the same time an interpretation function i that sends transcripts s of ι^π to transcripts t of π. We want them to satisfy the following.

1. For each partial transcript s^k of ι^π and each player PLR_j, the protocol ι^π gives a probability distribution, depending only on X, L_j, s^k and j that PLR_j will use to choose his next message.

2. If $L_j = 0$ then PLR_j chooses her messages in ι^π using the same distributions as in ι.

3. The interpretation function i sends (infinite) transcripts s of ι^π to either transcripts t of π or to "error". The probability of error is 0.

[2] A non-revealing protocol can also reveal who the leakers are. For example, if it is known that exactly one person is leaking and all but one person sends a message that could not have been sent by a leaker. However, if $\Pr(L = (0, \ldots, 0)) > 0$ then a non-revealing protocol will never reveal anyone as a leaker.

[3] The construction suggested in Lemma 5.1 for achieving certainty about the secret results in revealing protocol. However, assuming that it is possible to send message that are never send in ι, you can use Lemma 5.1 on ι^π to get a protocol where you learn X with certainty. All other protocols in Chapter 3 and Chapter 5 are non-revealing.

4. If T denotes the transcript of π and S denotes the transcript of ι^π, then given that $i(S)$ is not error, $(X, L_1, \ldots, L_n, i(S))$ is distributed as $(X, L_1, \ldots, L_n, T)$.

5. For each transcript t of π, the random variable $(X, L_1, \ldots, L_n)$ is independent from S given $i(S) = t$.

Here the second requirement ensures that non-leakers can follow the protocol without knowing X or π. In fact, unlike in the collaborating communication protocol, they might be thinking that everyone is just having an innocent conversation. Thus, in ι^π we often refer to the non-leakers as *innocents*. Notice the important assumption that first the innocent communication protocol ι is defined and *then* we create a protocol ι^π for leaking information on top of that. This corresponds to assuming that the non-leaking players either do not care about the leak, or that they are oblivious to the protocol. If ι was allowed to depend on what the leakers do, the non-leaking players could try to prevent the leak. The model where some of the non-leakers are motivated to prevent the leakage has been analysed in Chapter 4.

The fourth of the above requirements tells us that ι^π reveals at least as much about $(X, L_1, \ldots, L_n)$ as π and the last requirement says that we do not learn anything more. This ensures that Joe and Eve, who both know ι^π, learn exactly as much from the transcript of ι^π as they would from the transcript of π. Recall that in Definition 5.2, we defined two protocols π and π' to be equivalent for a distribution μ of (X, L) if, when used on input (X, L), the distributions of posterior distributions μ_T and $\mu_{T'}$ are the same.

Proposition 6.2. *If ι^π satisfies the above requirements, then ι^π and π are equivalent.*

Proof. By assumption, i gives error with probability 0, so we can ignore all those cases. By requirement 4, $i(S)$ has the same distribution as T, and by requirement 4 and 5 the distribution μ_s of $(X, L_1, \ldots, L_n)$ given $S = s$ equals the distribution $\mu_{i(s)}$. Thus, μ_S, $\mu_{i(S)}$ and μ_T have the same distribution. □

Before we construct the protocol ι^π we will define a function i' that sends partial transcripts $s^{k'}$ of ι^π to tuples $(t^k, [y, z))$ where t^k is a partial transcript of π, and $[y, z) \subset [0, 1)$ is a half-open interval. When $i'(s^{k'}) = (t^k, [y, z))$, we refer to t^k as

the interpretation of $s^{k'}$. Loosely speaking, the point of the interval is that not all messages in ι are sufficiently unlikely that they can correspond to a message in π, so instead of interpreting them to a message in π, we store the information by remembering an interval. This is very similar to arithmetic coding [76], however it is complicated by the fact that each player's messages depend on other players' messages. For an infinite transcript s, the function i' will satisfy

1. $i'(\lambda) = (\lambda, [0, 1))$, where λ is the empty transcript
2. If $i'(s^{k'}) = (t^k, [y, z))$ then either
   - $i'(s^{k'+1}) = (t^k \circ m, [0, 1))$ for some message m in π, or
   - $i'(s^{k'+1}) = (t^k, [y', z'))$, where $[y', z') \subseteq [y, z)$
3. If $i'(s^{k'}) = (t^k, [y, z))$ and t^k is a complete transcript for π, then $y = 0$, $z = 1$ and $i'(s^{k''}) = (t^k, [0, 1))$ for all $k'' \geq k'$

Thus, every time we reveal one more round from the transcript s, we will either learn one message in π from the interpretation of s, or the interval gets smaller or stays the same. If $i'(s^{k'}) = (t^k, [y, z))$ we say that t^k is *the interpretation of* $s^{k'}$. We let $j(s^{k'})$ and $j(t^k)$ denote the index of the player to send the next message in π when the current transcript is t^k. When it is clear what $s^{k'}$ is, we write j instead of $j(s^{k'})$. We will sometimes abuse language, and talk about what the players are doing in π even if they are following ι^π. For example if $j = j(s^{k'})$ we might say that "j is sending the message m in π" instead of "j is sending messages in ι^π which will be interpreted as m by i'". If $i'(s^{k'}) = (t^k, [y, z))$ and $i'(s^{k'+1}) = (t^k \circ m, [0, 1))$ we say that at time k' player $j(s^{k'})$ finished sending the message m in π and at time $k' + 1$ player $j(s^{k'+1})$ starts sending a new message in π.

For each partial transcript t^k of π, we let $\mathcal{A}_{t^k}$ denote the set of possible next messages. We assume that all sets of messages, both in π and ι, have an ordering, for example the lexicographical order. Algorithm 1 gives a pseudo code for i', but we will also define it in the main text.

Define a function $f : [0, 1) \to \mathcal{A}_{t^k, j}$ such that

$$f^{-1}(a) = [\Pr(T_{k+1} < a | T^k = t^k, L_j = 0), \Pr(T_{k+1} \leq a | T^k = t^k, L_j = 0)).$$

Algorithm 1 i'.

procedure I'($s^{k'}$)
  $t \leftarrow \lambda$ ▷ λ denotes the empty string, t a *partial* transcript
  $k \leftarrow 0$
  $y \leftarrow 0$
  $z \leftarrow 1$
  for r from 1 to k' **do**
    $y' \leftarrow y + (z - y) \Pr(S_{r,j(t)} < s_{r,j(t)} | S^{r-1} = s^{r-1})$
    $z' \leftarrow y + (z - y) \Pr(S_{r,j(t)} \leq s_{r,j(t)} | S^{r-1} = s^{r-1})$
    $y \leftarrow y'$
    $z \leftarrow z'$
    if $\exists a \in \mathcal{A}_t : \Pr(T_{k+1} < a | T^k = t, L_{j(t)} = 0) \leq y$ **and**
        $z \leq \Pr(T_{k+1} \leq a | T^k = t, L_{j(t)} = 0)$ **then**
      $t \leftarrow t \circ a$
      $k \leftarrow k + 1$
      $y \leftarrow 0$
      $z \leftarrow 1$
      if t is a complete transcript **then**
        return $(t, [0, 1))$
      end if
    end if
  end for
  return $(t, [y, z))$
end procedure

By definition of innocent communication protocol, each message in ι is chosen from a finite set, but to explain the point of the function f, imagine for now that ι said that in the next round PLR_j should send a real number chosen uniformly from $[0, 1)$. We could now interpret that as the message $f(x) \in \mathcal{A}_{t^k}$ in π. Then ι^π would say that if PLR_j was innocent he should send a number uniformly from $[0, 1)$ and if he was leaking, he should first choose $a \in \mathcal{A}_{t^k}$ using the distribution specified by π, and then send a number chosen uniformly at random from $f^{-1}(a)$. More generally, if ι said that PLR_j should choose his next message M from some continuous distribution on $\mathbb{R}$, we could take the quantile function given $L_j = 0$ of the message, that is

$$m \mapsto \Pr(M < m | L_j = 0),$$

to turn it into a message that is uniform on $[0, 1)$ given $L_j = 0$. Unfortunately, there is only finitely many possible messages for PLR_j to sent in each round, so instead of getting a number out of the quantile function, we define a similar function to get an interval. Let $i'(s^{k'}) = (t^k, [y, z))$ and choose some ordering on $\mathcal{M}_{s^{k'},j}$. Define $g : [y, z) \to \mathcal{M}_{s^{k'},j}$ by

$$g^{-1}(m) = \{y + (z - y)t | t \in [\Pr(M < m | L_j = 0), \Pr(M \leq m | L_j = 0))\}.$$

Here when we write M we are implicitly assuming that the current transcript is $s^{k'}$. Instead of getting a number in $[0, 1)$ out of $m \in \mathcal{M}_{j,s^{k'}}$, as we did above, we now get an interval $g^{-1}(m) \subset [y', z')$, whose length is proportional to the probability that an innocent player would send that message. If $g^{-1}(m) \subset f^{-1}(a)$ for some $a \in \mathcal{A}_{t^k}$ we say that PLR_j send a in π and define $i'(s^{k'+1}) = (t^k \circ a, [0, 1))$. Otherwise, PLR_j is not done sending his message and we define $i'(s^{k'+1}) = (t^k, g^{-1}(m))$. Algorithm 1 gives a pseudo code for computing i'. Here $s_{r,j}$ denotes the message in ι sent by player j in round r.

If for some k' we have $i'(s^{k'}) = (t, [0, 1))$ where t is a complete transcript of π we define $i'(s^{k''}) = (t, [0, 1))$ for all $k'' > k'$ and $i(s) = t$. If for some s no such k' exists, we define $i(s)$ to give "error".

Next we define the protocol ι^π. Any non-leaking player chooses his messages as given by ι and when the current transcript is $s^{k'}$ all players except $\text{PLR}_{j(s^{k'})}$ also

choose their messages as in ι. When a leaking player, $\text{PLR}_{j(s^{k'})}$, starts sending a message in π, he first choose the message $a \in \mathcal{A}_{t^k}$ using the distribution given by π (this distribution depends on $X = x$). Next he chooses a number α randomly and uniform in $f^{-1}(a)$. Until he has sent his message in π, he will now send messages m such that $\alpha \in g^{-1}(m)$. This uniquely specifies which messages m to send (notice that g will depend on current transcript in ι^π, so m is not necessarily the same for every round, and can depend on messages sent by other players). When we get to a transcript $s^{k'}$ that is interpreted as a complete transcript t of π, all the players will just follow ι. Figure 6.1 defines ι^π more formally, and Figure 6.2 gives an example of how one message in π is send by using ι^π.

We see that if in π a leaking player's distribution of a is exactly the same as a non-leaking players, then the distribution of the number α chosen by the leaking player is uniform on $[0, 1)$. By the definition of g, the probability that a leaking player sends a particular message m in ι^π is exactly the probability given by ι, and thus the same as a non-leaking player. Using this reasoning in the opposite direction, this tells us that we can assume that even the innocents, when starting sending a message in π, chooses a uniformly distributed $\alpha \in [0, 1)$ and sends the message m such that $\alpha \in g(m)$, until they have sent the message in π. They probably do not do that, but the probability of any transcript is the same as if they did.

Finally, we need to check that ι^π satisfies the 5 requirements. The first two follow from the construction. To show the third, we need to show that for a random transcript s of ι^π there will with probability 1 exists a k' such that $i'(s^{k'}) = (t, [0, 1))$ where t is a complete transcript for π. As π only has finitely many rounds, it is enough to show that for each message of π we start sending in ι^π, there is probability 1 that we will finish sending it. Assume that $i'(s^{k'}) = (t^k, [0, 1))$ for some k', where t^k is an incomplete transcript of π, but for all $k'' > k'$ the interpretation of $s^{k''}$ is still t^k. If $\text{PLR}_{j(s^{k'})}$ is innocent, everyone will be following ι, so by the assumption that ι is informative, the set of transcripts where the length of the interval does not go to 0 has probability 0. As stated earlier we can assume that when sending a message in π, even the innocents starts by choosing a random number α uniformly from $[0, 1)$. As f only jumps in finitely many points, there is probability 0 that $\text{PLR}_{j(s^{k'})}$ chooses one of these points. If he does not, and the length of the interval goes to 0, he will eventually send his message in π. Thus, there is probability 0 that a non-leaker

does not send his message. A leaker chooses his random $\alpha \in [0,1)$ using a different distribution, but we can divide $[0,1)$ into a finite set of intervals (given by $f^{-1}(a)$) such that it is uniform on each of these intervals. This tells us that given $s^{k'}$ there is a constant C such that, as long as $\text{PLR}_{j(s^{k'})}$ is still sending the same message in π, any continuation of the transcript is at most C times more likely when $\text{PLR}_{j(s^{k'})}$ is leaking as when he is not leaking. Thus, there is still probability $C \cdot 0 = 0$ that he will not finish his message in π.

For the fourth requirement, we observe that any leaking player is actually choosing messages in π following the distribution given by π, and then making sure that the message send in ι^π will be interpreted as the message he wanted to send in π. The innocent players are not doing this, but we have seen that the distribution on the message they send in ι^π are the same as if they did. Thus, requirement 4 holds. Finally, we see that given $i(S) = t$ a player not sending a message in π always follows ι and a player sending a message in π can be thought of as haven chosen an α uniformly from $f^{-1}(a)$ where a is the next message in transcript t. This is independent from $(X, L_1, \ldots, L_n)$ and thus the last requirement follows. □

To implement the protocol ι^π the leaking players do not have to choose all the infinitely many digits in a random number α. Instead, they can just for each message compute the probability that they would send each message if they had chosen an α. We also see that if $i(S)$ does not give an error, then there is some k such that S^k determines $i(S)$. If we let K be the random variable that is ∞ when we have error and otherwise gives the smallest value k such that S^k determines $i(S)$, then we know that $\Pr(K = \infty) = 0$. So all the probability mass of K is on the integers, hence for any $\epsilon > 0$ there must exists some k_0 such that $\Pr(K \geq k_0) < \epsilon$. That is, $i(S^{k_0})$ gives a total transcript with probability greater than $1 - \epsilon$.

Parameters:

π : A collaborative cryptogenography protocol

ι : An informative innocent communication protocol

Input distribution: (X, L) has some distribution over $\mathcal{X} \times \{0,1\}^n$. Each PLR_i learns L_i and if $L_i = 1$ she also learns X.
Protocol:

1. $t := \lambda, k := 0, s' := \lambda, k' := 0$
2. While t is not a complete transcript of π
3. Let PLR_j be the player to send the next bit in π when the transcript is t
4. If $L_j = 1$, PLR_j chooses the message t_{k+1} according to π and she chooses an α uniformly at random from the interval $[\Pr(T_{k+1} < a | T^k = t^k, L_j = 0), \Pr(T_{k+1} \leq a | T^k = t^k, L_j = 0))$
5. $y := 0, z := 1$
6. While no new message have been added to t:
7. If $L_j = 1$ let $s_{k'+1}$ be the unique message such that $y + (z-y)\Pr(S_{k'+1,j} < s_{k'+1}) \leq \alpha < y + (z-y)\Pr(S_{k'+1,j} < s_{k'+1})$ where the probabilities are according to ι.
8. Leakers PLR_i with $i \neq j$ and non-leakers all sends a message according to ι, if PLR_j is a leakers, she now sends the message $s_{k'+1,j}$ chosen in the previous line
9. $s' := s' \circ s_{k'+1}, k' := k' + 1$ where $s_{k'+1}$ is the list of messages sent in the previous line
10. $y' = y + (z-y)\Pr(S_{k'+1,j} < s_{k'+1}), z' = y + (z-y)\Pr(S_{k'+1,j} < s_{k'+1})$
11. $y := y', z := z'$
12. If there exists a t_{k+1} with $[y, z) \subset [\Pr(T_{k+1} < a | T^k = t^k, L_j = 0), \Pr(T_{k+1} \leq a | T^k = t^k, L_j = 0))$ then $t := t \circ t_{k+1}$ and $k := k + 1$
13. Everyone follows ι

Figure 6.1: Protocol from the proof of Theorem 6.1.

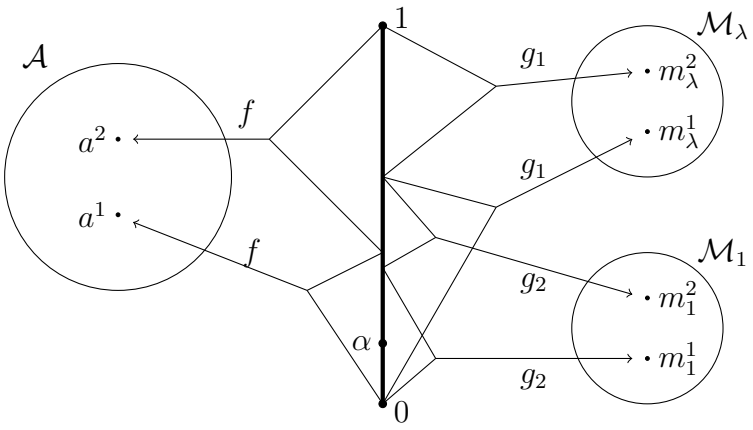

Figure 6.2: Example of how to construct a part of ι^π.
In this figure we see an example of how construct a part of ι^π. The line in the middle represent the interval $[0, 1)$, which contain subsets $[y, z)$ for i'. In π, the next player to send a message is PLR_j. The message A_1 should come from $\mathcal{A} = \{a^1, a^2\}$, which is represented on the left of the figure. We have $\Pr(A_1 = a^1 | L_j = 0) = 0.4$, so $f : [0, 1) \to \mathcal{A}$ sends $x \in [0, 0.4)$ to a^1, and $x \in [0.4, 1)$ to a^2 as indicated. Now $L_j = 1$, so PLR_j first chooses a message from $\mathcal{A}$ to send, this happens to be a^1, and then a number α chosen randomly and uniformly from $f^{-1}(a^1)$.
In ι, the next message M_λ that PLR_j sends should be from $\mathcal{M}_\lambda = \{m_\lambda^1, m_\lambda^2\}$. If PLR_j was innocent and was following the protocol ι, we would have $\Pr(M_\lambda = m_\lambda^1) = 0.6$, so $g_1 : [0, 1) \to \mathcal{M}_\lambda$ sends $x \in [0, 0.6)$ to m_λ^1 and the rest to m_λ^2. As $\alpha \in [0, 0.6)$, PLR_j now sends the message m_λ^1. We see that $g_\lambda^{-1}(m_\lambda^1)$ overlaps with both $f^{-1}(a^1)$ and $f^{-1}(a^2)$, so an observer cannot yet determine which message in π PLR_i was sending, so PLR_j has not sent his message yet. His next message, M_1, should be send from $\mathcal{M}_1 = \{m_1^1, m_1^2\}$, and again it happens that if he was following ι then $\Pr(M_1 = m_1^1) = 0.6$, so $g_1 : [0, 0.6) \to \mathcal{M}_1$ sends $x \in [0, 0.36)$ to m_1^1 and the rest to m_1^2. As $\alpha \in [0, 0.36)$, PLR_j sends the message m_1^1, and now $g_1^{-1}(m_1^1) \subset f^{-1}(a^1)$, so now an observer can see that PLR_j was sending the message a^1 in π, and PLR_j is done sending his message in π.

6.2 Hiding among innocents with censors

In this section we will show that even when there are censors, the leakers can still hide among players who are following a fixed innocent strategy ι. We cannot simply reuse the protocol ι^π from above, because the censors might obstruct the protocol. This can be done in two ways: either by obstructing when $j(s^k)$ is a censor, or when it is not. When $j(s^k)$ is a censor, that censor can keep sending message which makes the interval in $i'(s^k)$ smaller: the censor could simply send messages as if he had chosen an α which is on the boundary between two intervals on the form $f^{-1}(a)$. When the $j(s^k)$ is not a censor, the censors could possibly obstruct the protocol by making player $j(s^k)$s messages uninformative. For example, suppose that ι says that only one player should send a non-empty message in each round, and no players can start sending non-empty messages before the player who currently sends non-empty messages sends the message "over". Then a censor can ensure that $j(s^k)$ cannot send her message in π by never sending the message "over". Such a protocol ι could still satisfy our definition of informative, if the probability that a player following ι does not send "over" tends to 0.

To avoid this problem, we need a stronger definition than informative. We say that an innocent communication protocol ι is *strongly informative* for all players PLR_j, we have $\prod_{k\in\mathbb{N}} \Pr(S_{k,j} = s_{k,j} | S^{k-1} = s^{k-1}) = 0$ for all possible transcripts s. While "informative" means that we cannot predict all of player js messages with positive probability when all the players follow ι, "strongly informative" means that we cannot do so with positive probabilities, even if we can control the messages send by all the other players.

Even if we assume that ι is strongly informative, censors could still obstruct the protocol ι^π when they are $j(s^k)$ as described above. To avoid this, we need to eventually give up on a player sending a message in π. That is, if the interval in $i'(s^{k'}) = (t^k, [y, z))$ becomes too small, we will define $i'(s^{k+1}) = (t^k \circ t_{k+1,0}, [0, 1))$ where $t_{k+1,0}$ is the lexicographically first element in $\mathcal{A}_{t^k}$. When this happens, we say that player $j = j(s^{k'})$ *timed out*. From here, the players (and the function i) will continue as if player j had sent $t_{k+1,0}$ in π, except that if player j is a leaker, she will now follow ι, i.e. pretend ignorance, for the rest of the protocol.

The following is a resilient version of Theorem 6.1.

Theorem 6.3. *Let $n, \mathcal{L} \in \mathbb{N}, h, \epsilon, \epsilon' > 0, b = (b_l, b_c, b_m)$ with $b_m < 1$ be given. Let π be an $(n, h, \mathcal{L}, \epsilon, b)$-protocol and ι a strongly informative innocent communication protocol. Then there exists an $(n, h, \mathcal{L}, \epsilon + \epsilon', b)$-protocol $\iota^{\pi,res}$ where the neutral players follow ι.*

Proof. Let π and ι be as in the statement of the theorem.

We define i' just as in the proof of Theorem 6.1, except that if the interval $z - y$ becomes very small, i' will just say that the current player $j(t^k)$ sent the lexicographically first message, $t_{k+1,0}$, in $\mathcal{A}_{t^k}$. When this happens, we say that player $j(t^k)$ *timed out*. More precisely, for any partial transcript t^k we define

$$C_{t^k} = \frac{\min_{t^{k+1} \in \mathcal{A}_{t^k}} \Pr(T_{k+1} = t_{k+1} | T^k = t^k, L_{j(t^k)} = 0)}{2^{k+2} \cdot |\mathcal{A}_{t^k}| \cdot \epsilon'}$$

Notice that C_{t^k} is strictly positive: as π is a $(n, h, \mathcal{L}, \epsilon, b)$-protocol, it is a b-leaker protocol, hence it is non-revealing, so any message send by a leaker could be send by a non-leaker. Algorithm 2 computes this new i'.

Next we need to specify $\iota^{\pi,\text{res}}$. In this protocol, the non-leakers will follow ι. Any leaker j will do as in the proof of Theorem 6.1 (of course using the new i' to interpret other player's messages), unless j has previously timed out. Once a leaker has timed out, that leaker will follow ι for the rest of the communication. The protocol $\iota^{\pi,\text{res}}$ is defined more formally in Figure 6.3.

Before we show that this $\iota^{\pi,\text{res}}$ is a $(n, h, \mathcal{L}, \epsilon + \epsilon', b)$-protocol, we will show two propositions.

Proposition 6.4. *For any infinite transcript S where leaker and innocent are following $\iota^{\pi,res}$ and censors are following σ', there exists some k' such that $i'(S^{k'}) = (t, [0, 1))$ where t is a complete transcript using protocol π and some σ.*

Proof. As π only has finitely many rounds, it is enough to show that any message started will be finished. Assume the $i'(s^{k'}) = (t^k, [0, 1))$ for some k'. For $k'' > k'$ with $i'(s^{k''}) = (t^k, [y, z])$ we have

$$z - y = \prod_{l=k'}^{k''-1} \Pr\left(S_{l,j(t^k)} = s_{l,j(t^k)} | S^{l-1} = s^{l-1}\right)$$

Algorithm 2 i'.

procedure I'($s^{k'}$)
  $t \leftarrow \lambda$ ▷ λ denotes the empty string, t a *partial* transcript
  $k \leftarrow 0$
  $y \leftarrow 0$
  $z \leftarrow 1$
  for r from 1 to k' **do**
    $y' \leftarrow y + (z - y) \Pr(S_{r,j(t)} < s_{r,j(t)} | S^{r-1} = s^{r-1})$
    $z' \leftarrow y + (z - y) \Pr(S_{r,j(t)} \leq s_{r,j(t)} | S^{r-1} = s^{r-1})$
    $y \leftarrow y'$
    $z \leftarrow z'$
    if $\exists a \in \mathcal{A}_t : \Pr(T_{k+1} < a | T^k = t, L_{j(t)} = 0) \leq y$ **and**
        $z \leq \Pr(T_{k+1} \leq a | T^k = t, L_{j(t)} = 0)$ **then**
      $t \leftarrow t \circ a$
      $k \leftarrow k + 1$
      $y \leftarrow 0$
      $z \leftarrow 1$
    else
      if $z - y < C_{t^k}$ **then**
        $t \leftarrow t \circ t_{k+1,0}$ ▷ If this line is evaluated we say $j(t^k)$ *timed out*
        $k \leftarrow k + 1$
        $y \leftarrow 0$
        $z \leftarrow 1$
      end if
    end if
    if t is a complete transcript **then**
      return $(t, [0, 1))$
    end if
  end for
  return $(t, [y, z))$
end procedure

By the assumption that ι is strongly informative, we have

$$\prod_{l\in\mathbb{N}} \Pr\left(S_{l,j(t^k)} = s_{l,j(t^k)} | S^{l-1} = s^{l-1}\right) = 0.$$

We know that $\prod_{l=1}^{k'-1} \Pr\left(S_{l,j(t^k)} = s_{l,j(t^k)} | S^{l-1} = s^{l-1}\right) > 0$, so[4] there must exists a k'' such that $\prod_{l=k'}^{k''} \Pr\left(S_{l,j(t^k)} = s_{l,j(t^k)} | S^{l-1} = s^{l-1}\right) < C_{t^k}$. This shows that $i'(s^{k''+1})$ cannot be on the form $(t^k, [y, z))$, so the message must be finished. □

Thus, we can define $i(S)$ to be the complete transcript t with $i'(S^{k'}) = (t, [0, 1))$ just as we did in the proof of Theorem 6.1.

Proposition 6.5. *For any x and any l, if the leakers and neutral are following $\iota^{\pi,res}$, and the censors are following some σ', the probability that at least one leaker or neutral will time out during the execution of the protocol given $X = x$ and $L = l$ is at most ϵ'.*

Proof. Let us bound the probability that a leaker or neutral will time out while in round k of π given t^{k-1}. As stated in the proof of Theorem 6.1, we can assume that leakers and neutrals first decide on a message $a \in \mathcal{A}_{t^{k-1}}$ and then choose a number $\alpha \in f^{-1}(a)$. We want to bound the probability that α is within a distance of $C_{t^{k-1}}$ of the boundary between two intervals on the form $f^{-1}(a')$. As there are $|\mathcal{A}_{t^{k-1}}|$ such intervals, there are $|\mathcal{A}_{t^{k-1}}| - 1 < |\mathcal{A}_{t^{k-1}}|$ such boundaries. Thus, the set of numbers that are within $C_{t^{k-1}}$ of such a boundary has measure at most $2C_{t^{k-1}}|\mathcal{A}_{t^{k-1}}|$. When choosing α, uniformly from $f^{-1}(a)$, the probability density is taking the value

$$\frac{1}{|f^{-1}(a)|} \leq \frac{1}{\min_{a'\in\mathcal{A}_{t^{k-1}}} |f^{-1}(a')|} = \frac{1}{\min_{a'\in\mathcal{A}_{t^{k-1}}} \Pr(T^k = a' | T^{k-1} = t^{k-1}, L_j = 0)}.$$

Here $|\cdot|$ denotes the length of an interval. Thus, the probability given a that α is within $C_{t^{k-1}}$ of a boundary is at most $\frac{2C_{t^{k-1}}|\mathcal{A}_{t^{k-1}}|}{\min_{a'\in\mathcal{A}_{t^{k-1}}} \Pr(T^k=a'|T^{k-1}=t^{k-1},L_j=0)} = 2^{-k}\epsilon'$. As this is true for all a, the probability is this at most $2^{-k}\epsilon'$ when taking the weighted average over a. We see that when α is at least $C_{t^{k-1}}$ away from a boundary point,

[4]Here we are assuming, just as in Chapter 4, that the censors will only send messages that a neutral would send with positive probability. If we allow censors to send other messages we would have to modify i', but just as we argued in Chapter 4, this can be done by pretending that such players send a default message $t_{k,0}$ whenever it is their turn.

we cannot have $z - y < C_{t^{k-1}}$ and $\alpha \in [y, z)$ without $[y, z)$ being contained in an interval on the form $f^{-1}(a)$. Thus, in round k of π there is probability at most $2^{-k}\epsilon'$ that a leaker or a neutral will time out, so in total there is probability at most ϵ' that a non-censor times out. □

We are now ready to show that $\iota^{\pi,\text{res}}$ is a $(n, h, \mathcal{L}, \epsilon + \epsilon', b)$-protocol. To do so, we need to show two properties of $\iota^{\pi,\text{res}}$: reasonable doubt and reliable leakage.

Proposition 6.6. *The protocol $\iota^{\pi,res}$ keeps reasonable doubt, that is, it is a b-leaker protocol.*

Proof. By assumption, π is a b-leaker protocol. This implies that the likelihood ratio of π is at most $r(b)$. From the proof of Theorem 6.1 we know that if it was not for the possibility of timing out, the likelihood ratio of $\iota^{\pi,\text{res}}$ would also be at most $r(b)$. For any particular player, j, we see that the fact that other players can time out does not affect the upper bound on player js likelihood ratio, and if player j times out, she will just start pretending ignorance. Thus, the likelihood ratio of $\iota^{\pi,\text{res}}$ is at most $r(b)$, so $\iota^{\pi,\text{res}}$ is a b-leaker protocol. □

Proposition 6.7. *Let X be uniformly distributed on $\mathcal{X} = \left[2^{\lceil h \rceil}\right]$ and independently $L = (L_1, \dots L_n) \sim b$. Assume the leakers and neutrals use $\iota^{\pi,res}$ to communicate. Then there is a function G' taking as values subsets of $\mathcal{X} = [2^{\lceil h \rceil}]$ of size at most $\mathcal{L}$, such that no matter what protocol σ' the censors use, we have $\Pr(X \in G' | X = x) \geq 1 - \epsilon - \epsilon'$.*

Proof. Define $G'(S) = G(i(S))$, where G is the function which shows π to be an $(n, h, \mathcal{L}, \epsilon, b)$-protocol. For any censor protocol σ' used against $\iota^{\pi,\text{res}}$, we define a joint distribution of (X, L, T, S) as follows. We let (X, L) be distributed as in the statement of this theorem. For simplicity, we first consider the case without censor. When the leakers and neutral sends a message in $\iota^{\pi,\text{res}}$ we can, as we argued in the proof of Theorem 6.1, assume that they first think of a message in π they want to send. By using the messages they were thinking of to get a distribution of T, and the message they send to get the distribution of S, we get a distribution (X, L, T, S) such that $i(S) = T$ unless someone timed out. Next we add the censors. For these, we simply assume that they follow σ' to construct our distribution of S, and we use i' to interpret this as a distribution of message t_k. This way we still have a

distribution of (X, L, T, S) where $i(S) = T$ unless *either a leaker or a neutral* timed out. The probability that such a player times out is at most ϵ' given $X = x$. By the union bound,

$$\begin{aligned}\Pr(X \notin G'(S)|X = x) =& \Pr(X \notin G(i(S))|X = x) \\ \leq& \Pr(X \notin G(T)|X = x) + \Pr(i(S) \neq T|X = x) \\ \leq& \epsilon + \epsilon'.\end{aligned}$$

□

□

Parameters:

π : A resilient cryptogenography protocol

ι : A strongly informative innocent communication protocol

ϵ' : Acceptable increase in probability of error

Input distribution: (X, L) has some distribution over $\mathcal{X} \times \{-1, 0, 1\}^n$. Each PLR_i learns L_i and if $|L_i| = 1$ she also learns X, and if $L_i = -1$ she also learns $L_- = (\min(L_1, 0), \ldots, \min(L_n, 0))$. Players PLR_i with $L_i \neq -1$ will follow the protocol we define here, but and players with $L_i = -1$ might follow any censor protocol σ.
Protocol:

1. $t := \lambda, k := 0, s' := 0, k' := 0$
2. While t is not a complete transcript of π
3. Let PLR_j be the player to send the next bit in π when the transcript is t
4. If $L_j = 1$, PLR_j chooses the message t_{k+1} according to π and then chooses an α uniformly at random from the interval $[\Pr(T_{k+1} < a | T^k = t^k, L_j = 0), \Pr(T_{k+1} \leq a | T^k = t^k, L_j = 0))$
5. $y := 0, z := 1$
6. While no new message have been added to t:
7. If $L_j = 1$, let $s_{k'+1}$ be the unique message such that $y + (z-y)\Pr(S_{k'+1,j} < s_{k'+1}) \leq \alpha < y + (z-y)\Pr(S_{k'+1,j} < s_{k'+1})$ where $S_{k'+1,j} \sim \iota$.
8. Leakers PLR_i with $i \neq j$ and non-leakers all sends a message according to ι, if PLR_j is a leakers, she now sends the message $s_{k'+1,j}$ chosen in line 7
9. $s' := s' \circ s_{k'+1}, k' := k' + 1$ where $s_{k'+1}$ is the list of messages sent in line 8 , $y' := y + (z-y)\Pr(S_{k'+1,j} < s_{k'+1}), z' := y + (z-y)\Pr(S_{k'+1,j} < s_{k'+1})$, $y := y', z := z'$
10. If there exists a t_{k+1} with $[y, z) \subset [\Pr(T_{k+1} < a | T^k = t^k, L_j = 0), \Pr(T_{k+1} \leq a | T^k = t^k, L_j = 0))$ then $t := t \circ t_{k+1}$ and $k := k + 1$
11. Else, if $z - y < C_{t^k}$ as defined in the proof, then $t := t \circ t_{k+1,0}, k := k+1$
12. Everyone follows ι

Figure 6.3: Protocol from the proof of Theorem 6.3.

Chapter 7

Anonymous Steganography

In this chapter, we consider a model where there is only one leaker and we want to ensure that an observer can guess a secret, but does not get much advantage in guessing who the leaker was. However, unlike in the rest of this thesis, we will in this chapter assume that the adversary has bounded computational power. We will see that even with this weaker adversary, the leaker can only reveal a very small amount of information. However, the leaker can do something she cannot do against an unbounded adversary: she can bootstrap an anonymous channel. That is, if she has access to an expensive or small anonymous channel as well as a large public non-anonymous channel, she can use this to send a large message anonymously.

One idea for having anonymous communication might be to use steganography. The goal of traditional steganography is to hide that a certain communication is taking place, by embedding sensitive content in innocent looking traffic (such as pictures, videos, or other documents). There is no doubt that steganography is a useful tool for Lea the leaker: using steganography she could send sensitive documents to Joe the journalist in such a way that even someone monitoring all internet traffic would not be able to notice that this sensitive communication is taking place. However, steganography alone cannot help Lea if she wants to make sure that *even Joe* does not learn her identity.

To solve this problem, we introduce a novel cryptographic primitive, which we call *anonymous steganography*. Very informally, anonymous steganography works in the following way: Lea wants to communicate a sensitive large message x to

Joe. To do so, she embeds x in some large innocent looking document c which she uploads to a popular website, not necessarily in an anonymous way. Then Lea produces some short decoding key dk which is a function of c and all other documents on the website – or at least a set large enough so that her identity is hidden in a large group of users, such as "all videos uploaded last week". She then sends dk to Joe using an anonymous channel. Now Joe is able to recover the original message x from the website using the decoding key dk, but at the same time Joe has no way of telling which document contains the message and therefore which of the website's users is the leaker. Intuitively, it is crucial for Lea's anonymity that Joe can only decode *the entire website*: if Joe had a way of decoding single documents or portions he would easily be able to pinpoint which document actually contains the leaked message. To construct an anonymous steganography scheme we will be using an indistinguishablity obfuscator. These are conjectured to exist and a candidate construction exists [30], but they have not been proved secure under standard cryptographic assumptions. Furthermore, the candidate construction is so slow that our scheme is currently infeasible in practice.

In the scheme described above, Lea sends Joe a decoding key dk using a pre-existing anonymous channel. It is a natural question to ask whether this is necessary, or if we can construct a scheme where all communication between Lea and Joe takes place over regular channels. Unfortunately this is too good to be true, and in Section 7.4 we prove that it is impossible to construct an anonymous steganography scheme unless Lea sends a key of super-logarithmic size to Joe. The idea behind the proof is: if the scheme is correct at some point the probability that Joe outputs x has to increase from exponentially small to 1. Once the probability has increased to only polynomially small, Joe can estimate how each message sent by any of the users over the non-anonymous channel affects this probability and concludes that the messages which increases this probability the most, must come from the leaker. Hence, the message that causes this increase has to be sent over an anonymous channel.

In this chapter we will assume that the innocents sample their documents independently from the uniform distribution. As we are assuming the adversary has bounded computational power, we could weaken this to the assumption that innocents are uploading documents which are computationally indistinguishable from

uniformly distributed. This is a good model if Lea is uploading her message to a webpage where people upload encrypted documents or if she is taking part in a cryptographic protocol where each person provides a large amount of randomness.

If she is uploading to a webpage where the innocent communication is not computational indistinguishable from uniform, we could imagine using the results from Chapter 6. However, this requires knowing the distribution of innocent messages. This might be a reasonable assumption when the domain is small, for example it is realistic to find a distribution which is very close to computationally indistinguishable from the minute number a blogpost is posted. However, for larger messages, this is much harder, for example it will be extremely difficult to define a distribution over videos which is indistinguishable from videos uploaded to youtube. Instead, if Lea is uploading to a website where the documents do not look uniformly distributed, she should use a steganography protocol to transform her uniformly looking message t_i to an innocent looking message t'_i. If the transcript of innocent looking documents is $(t'_1, \ldots, t'_n)$ we define t_j to be what you get when applying the steganographic extraction algorithm on t'_j. For a good steganography protocol this t_j should be computationally indistinguishable from uniform, when t'_j is chosen from the innocent distribution. The problem of finding such steganography protocols is well known (see Section 2.3).

7.1 Definitions

We define an *anonymous steganography* scheme to be a tuple of four algorithms, $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{KeyEx}, \mathsf{Dec})$. All algorithms, even when not specified, take as input the security parameter λ, and the length parameters s, ℓ, ℓ' (s is *short*, ℓ is *long*). The syntax of the algorithm is as follows:

- $ek \leftarrow \mathsf{Gen}(1^\lambda)$ is a randomized algorithm which generates an encoding key ek.
- $c \leftarrow \mathsf{Enc}_{ek}(x)$ is a randomized algorithm which encodes a secret message $x \in \{0,1\}^{\ell'}$ into a (random looking) document $c \in \{0,1\}^{\ell}$.
- $dk \leftarrow \mathsf{KeyEx}_{ek}(t, i)$ takes as input a public vector of documents $t \in (\{0,1\}^{\ell})^d$, an index $i \in [d]$ such that $t_i = c$ and extracts a (short) decoding key $dk \in$

$\{0,1\}^s$.

- $x' = \mathsf{Dec}_{dk}(t)$ recovers a message x' using the decoding key dk and the public vector of documents t in a deterministic way.

We chose to keep Gen separated from Enc since a single key could be used to encode multiple messages – in a natural extension of the scheme Lea hides her secret(s) in a subset of documents $I \subset [d]$. Finally, KeyEx is a separated algorithm since it takes as input documents which are generated from honest users *after* c is published.

7.1.1 How to use the scheme

To use anonymous steganography, Lea generates the encoding key ek using Gen and encodes her secret x using Enc_{ek} to get the c. Lea then uploads c to this website as if she was a normal user of the website, and waits until more honest content is published. Then she chooses the set of documents she is hiding among, for example, all files uploaded to this website during that day or that week. Lea then downloads all these documents t and finds the index i of her own document in this set. Finally, she computes $dk \leftarrow \mathsf{KeyEx}_{ek}(t, i)$, and uses the small anonymous channel to send dk to Joe together with a pointer to t.

Lea can use this scheme, even if Joe have never heard about anonymous steganography beforehand. In this case, Lea can just send him a short message over the anonymous channel, asking him to run a program that finds t on the internet and computes Dec_{dk} on them.

7.1.2 Properties

We require the following properties: *correctness* (meaning that $x' = x$ with overwhelming probability), *compactness* (meaning that $s < \ell'$) and *anonymity* (meaning that the receiver does not learn any information about i). Another natural requirement is *confidentiality* (meaning that one should not be able to learn the message without the decoding key dk), but we will see that this follows from *anonymity*. Formal definitions follow.

Definition 7.1 (Correctness)**.** We say an anonymous steganography scheme is q-*correct* if for all $\lambda \in \mathbb{N}, x \in \{0,1\}^{\ell'}, i \in [d], t_{-i} \in (\{0,1\}^{\ell})^{d-1}$ over $\{0,1\}^{\ell}$ the following holds:

$$\Pr\left[\mathsf{Dec}_{dk}\left((t_{-i}, c)\right) = x\right] \geq q,$$

where $ek \leftarrow \mathsf{Gen}(1^{\lambda})$, $c \leftarrow \mathsf{Enc}_{ek}(x)$, $dk \leftarrow \mathsf{KeyEx}_{ek}((t_{-i}, c), i)$ and the probabilities are taken over all the randomness in these algorithms. We simply say that a scheme is *correct* when $q \geq 1 - \mathsf{negl}(\lambda)$.

Definition 7.2 (Anonymity)**.** We define a game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$:

1. The adversary $\mathcal{A}$ outputs a message $x \in \{0,1\}^{\ell'}$;
2. The challenger $\mathcal{C}$:
   (a) generates a key $ek \leftarrow \mathsf{Gen}(1^{\lambda})$;
   (b) samples a bit $b \leftarrow \{0,1\}$;
   (c) computes $c_b \leftarrow \mathsf{Enc}_{ek}(x)$ and samples $c_{1-b} \leftarrow \{0,1\}^{\ell}$;
   (d) outputs c_0 and c_1;
3. The adversary $\mathcal{A}$ outputs a vector t such that there exists i_0 and i_1 with $t_{i_0} = c_0$ and $t_{i_1} = c_1$;
4. The challenger $\mathcal{C}$ finds a possible value of i_b and outputs $dk \leftarrow \mathsf{KeyEx}_{ek}(t, i_b)$;
5. $\mathcal{A}$ outputs a guess bit g;

We say π satisfies *anonymity* if for all PPT $\mathcal{A}$ we have $\left|\Pr[g = b] - \frac{1}{2}\right| = \mathsf{negl}(\lambda)$.

This definition says that even if all but two of the players are censors, Eve will not be able to get an advantage in guessing which of the two non-censors is the leaker. This definition implies that if there are more than two non-leakers, Eve still do not get any advantage when guessing who the leakers are. The definition given here is much stronger than the definition in the published version of the work [50]: in the original definition, the adversary had to send i_0, i_1 and $t_{-(i_0,i_1)}$ before receiving t_{i_0} and t_{i_1}. Unlike the original definition, the above definition provides anonymity

against an adversary who controls the database to which Lea is uploading files, and who can reorder, insert and delete documents in an attempt to break the anonymity. The anonymity set will be the set of users who uploaded documents that have not been modified. Thus, an adversary with control over the website would be able to confirm that Lea was the leaker by modifying all the other documents. However, this is unlikely, as the adversary would have to do this before the leak has been completed, and would have to suspect that Lea was about to leak information anonymously using this particular database.

It might also seem natural to add the requirement that an adversary who does not know dk cannot get any information about the message. However, as we will see below, this follows from the anonymity requirement.

Definition 7.3 (Confidentiality)**.** Consider the following game:

1. $\mathcal{A}$ outputs $x_0, x_1 \in \{0,1\}^\ell, i, t_{-i}$;
2. $\mathcal{C}$outputs $t_i \leftarrow \mathsf{Enc}_{\mathsf{Gen}(1^\lambda)}(m_b)$ with $b \leftarrow \{0,1\}$;
3. $\mathcal{A}$ outputs a guess bit g;

We say π satisfies *confidentiality* if for all PPT $\mathcal{A}$ we have $\left|\Pr[g = b] - \frac{1}{2}\right| = \mathsf{negl}(\lambda)$.

Given an adversary B which breaks *confidentiality* we construct an adversary $\mathcal{A}$ for anonymity. First, assume for contradiction that for any message x_0 adversary $\mathcal{B}$ has only negligible advantage when guessing if a string is $t_0 \leftarrow \mathsf{Enc}_{\mathsf{Gen}(1^\lambda)}(x_0)$ or $r \leftarrow \{0,1\}^\ell$. By a hybrid argument, $\mathcal{B}$ would then only have negligible advantage when guessing if a string is $t_0 \leftarrow \mathsf{Enc}_{\mathsf{Gen}(1^\lambda)}(x_0)$ or $t_1 \leftarrow \mathsf{Enc}_{\mathsf{Gen}(1^\lambda)}(x_1)$. This contradicts the assumption that $\mathcal{B}$ breaks confidentiality.

Thus, we can choose a message x_0 where $\mathcal{B}$ has non-negligible advantage in distinguishing it from $r \leftarrow \{0,1\}^\ell$. Now we let $\mathcal{A}$ be an adversary that sends this message x_0 in the first round of the anonymity game, and later chooses i_0, i_1 and t in any legal way. Then $\mathcal{A}$ runs $\mathcal{B}$ on c_0. Because $\mathcal{B}$ has a non-negligible advantage at telling the difference between $c_0 \leftarrow \mathsf{Enc}_{\mathsf{Gen}(1^\lambda)}(x_0)$ and a random sample $c_0 \leftarrow \{0,1\}^\ell$, $\mathcal{A}$ will have a non-negligible advantage at the anonymity game.

7.2 Building blocks

In this section, we will describe the building blocks we will use to construct an anonymous steganography scheme.

7.2.1 Indistinguishability obfuscation.

An *indistinguishability obfuscator* is a randomized function that takes a circuit as input and gives a circuit as output. The idea is that the output circuit should compute the same function as the input circuit, but at the same time hide the information, such as a secret key, which was used to build the input circuit. We use an obfuscator $\mathcal{O}$ as proposed in [30] which takes any polynomial size circuit C and outputs an obfuscated version $\mathcal{O}(C)$ that satisfies the following property.

Definition 7.4 (Indistinguishability Obfuscation)**.** We say $\mathcal{O}$ is an *indistinguishability obfuscator* for a circuit class $\mathcal{C}$ if for all $C_0, C_1 \in \mathcal{C}$ such that $\forall x : C_0(x) = C_1(x)$ and $|C_0| = |C_1|$ it holds that:

1. $\forall C \in \mathcal{C}, \forall x \in \{0,1\}^n, \mathcal{O}(C)(x) = C(x)$;
2. $|\mathcal{O}(C)| = \mathsf{poly}(\lambda|C|)$
3. for all PPT $\mathcal{A}$:

$$|\Pr[\mathcal{A}(\mathcal{O}(C_0)) = 1] - \Pr[\mathcal{A}(\mathcal{O}(C_1)) = 1]| < \mathsf{negl}(\lambda)$$

It is not known if indistinguishable obfuscators exist, but a candidate has been proposed [30]. Even if they exist, they are far from being efficient enough to be practical.

7.2.2 IND-CPA public-key encryption scheme

A public-key encryption scheme is a tuple $(\mathsf{E.G}, \mathsf{E.E}, \mathsf{E.D})$ where, $\mathsf{E.G}$ takes the security parameter as input and outputs (pk, sk) where pk is called the public key and sk is the (secret) private key. The encryption algorithm $\mathsf{E.E}$ takes the public key

and a message as input, and outputs a ciphertext $c \leftarrow \mathsf{E.E}_{pk}(x)$, and the decryption algorithm $\mathsf{E.D}$ takes the private key and a ciphertext as input, and outputs a message $x' \leftarrow \mathsf{E.D}_{sk}(c)$. We make the following definitions

Definition 7.5. A public-key encryption scheme $(\mathsf{E.G}, \mathsf{E.E}, \mathsf{E.D})$ is *correct* if, for all λ and all $x \in \{0,1\}^\ell$ we have $x = \mathsf{E.D}_{sk}(\mathsf{E.E}_{pk}(x))$ whenever $(pk, sk) \leftarrow \mathsf{E.G}(\lambda)$.

Definition 7.6. We define the *indistinguishability game for chosen plaintext attack*, or *IND-CPA* game as follows.

1. $\mathcal{C}$ choose $b \leftarrow \{0,1\}$.
2. $\mathcal{C}$ compute $(pk, sk) \leftarrow \mathsf{E.G}(\lambda)$ and sends pk to the attacker.
3. $\mathcal{A}$ outputs two messages m_0, m_1.
4. $\mathcal{C}$ outputs $c \leftarrow \mathsf{E.E}_{pk}(m_b)$
5. $\mathcal{A}$ outputs a guess g

We say that $(\mathsf{E.G}, \mathsf{E.E}, \mathsf{E.D})$ is IND-CPA secure if for all PPT $\mathcal{A}$ we have $\left|\Pr[g = b] - \frac{1}{2}\right| < \mathsf{negl}(\lambda)$.

7.2.3 Homomorphic encryption

Informally, a homomorphic encryption scheme is a way to encrypt data in such a way that someone can make computations on the data, even if they cannot decrypt it.

Formally, let $(\mathsf{HE.G}, \mathsf{HE.E}, \mathsf{HE.D})$ be an IND-CPA public-key encryption scheme with an additional algorithm $\mathsf{HE.Eval}$ which on input the public key pk, n ciphertexts $c_1, \ldots, c_n$ and a circuit $C : \{0,1\}^n \to \{0,1\}$ outputs a ciphertext c^*, then we say that:

Definition 7.7 (Correctness – HE)**.** An HE scheme $(\mathsf{HE.G}, \mathsf{HE.E}, \mathsf{HE.D}, \mathsf{HE.Eval})$ is *correct for a circuit class* $\mathcal{C}$ if for all $C \in \mathcal{C}$

$$\mathsf{HE.D}_{sk}(\mathsf{HE.Eval}_{pk}(C, \mathsf{HE.E}_{pk}(x_1), \ldots, \mathsf{HE.E}_{pk}(x_n)) = C(x_1, \ldots, x_n)$$

Definition 7.8 (Compactness – HE)**.** An HE scheme $(\mathsf{HE.G}, \mathsf{HE.E}, \mathsf{HE.D}, \mathsf{HE.Eval})$ is called *compact* if there exists a polynomial $s \in \mathsf{poly}(\lambda)$ such that the output of $\mathsf{HE.Eval}(C, c_1, \ldots, c_n)$ is at most s bits long (regardless of the size of the circuit $|C|$ or the number of inputs n).

The first candidate homomorphic encryption for all circuits was introduced by Gentry [31]. Later Brakerski and Vaikuntanathan [7] showed that it is possible to build homomorphic encryption based only on the (reasonable) assumption that the learning with error problem (LWE) is computationally hard.

7.2.4 Pseudorandom functions

A *pseudorandom function family* (or a *PRF*) is a function $f : \{0,1\}^\lambda \times \{0,1\}^\lambda \to \{0,1\}$, such that for random $k \in \{0,1\}^\lambda$, the function $x \to f(k,x)$ is indistinguishable from random. More formally, we consider the following game.

- Challenger chooses $b \leftarrow \{0,1\}, k \leftarrow \{0,1\}^\lambda$.
- If $b = 1$ then $h(x) = f(k,x)$ if $b = 0$ then $h(x)$ is chosen uniformly and independently from $\{0,1\}$ for each $x \in \{0,1\}^\lambda$.
- The adversary $\mathcal{A}$ gets oracle access to h.
- The adversary $\mathcal{A}$ outputs a guess bit g.

We say that f is a PRF if for all PPT $\mathcal{A}$ we have $\left|\Pr[g = b] - \frac{1}{2}\right| < \mathsf{negl}(\lambda)$.

It is well known that the existence of one way function (implied by the existence of homomorphic encryption) implies the existence of PRFs.

7.2.5 Somewhere statistically binding vector commitment scheme

This primitive was introduced by Hubáček and Wichs [43] under the name *somewhere statistically binding hash*, but we think that the term *vector commitment scheme* is better at communicating the goal of this primitive.

The idea behind commitment schemes in general is that you can commit to some value without revealing it. For example, suppose you made some prediction,

represented with a string β, which you do not want to reveal for now. However, you also want to ensure that in the future, you can prove that you made this particular prediction. To do this, you could compute $\gamma = H(\beta \circ r)$ for some cryptography hash function H and randomness r and reveal the result, or you could encrypt β and reveal the result γ. Later you can reveal that you had chosen β, by revealing β and the randomness used. This is called decommiting. In the first case, γ will typically be much shorter than β, and the commitment will only be computational. That is, there exists a $\beta' \neq \beta$ with $H(\beta' \circ r') = \gamma$, but the commitment scheme still works, because such $\beta'$and r' are hard to find. In the second case, the commitment will be statistically binding[1], which means that there are no $\beta' \neq \beta$ which encrypt to γ, however, γ will have to be at least as long as β.

To construct an anonymous steganography scheme, we will need a commitment scheme. The problem is, we need the best from both types of commitments: we need the γ to be much shorter than β to make our dk shorter than the message we send, but we also need the commitment to be statistically binding to make our proofs work. This is impossible, for the simple reason that if γ is much shorter than β, there are more possible values of β than γ, so each β cannot have their own γ. Instead, we will be using what we call a vector commitment scheme. In this scheme, any commitment will be computationally binding, which mean that we can make γ shorter than β, but it will also be statistically binding in one particular entry i. That is, if you can open γ as both β and β' we must have $\beta_i = \beta'_i$. Furthermore, the value of i will not be known to the person who makes the commitment.

More formally a SSB vector commitment scheme is composed of the following algorithms:

Key Generation: The key generation algorithm $ck \leftarrow \mathsf{VC.G}(1^\lambda, L, i)$ takes as input an integer $L \leq 2^\lambda$ and index $i \in [L]$ and outputs a public key ck.

Commit: The commit algorithm $\mathsf{VC.C}_{ck} : (\{0,1\}^{\ell_\mathsf{b}})^L \to \{0,1\}^{\ell_\mathsf{c}}$ is a deterministic polynomial time algorithm which takes as input a string $x = (x_1, \ldots, x_L) \in (\{0,1\}^{\ell_\mathsf{b}})^L$ and outputs $\mathsf{VC.C}_{ck}(x) \in \{0,1\}^{\ell_\mathsf{c}}$.

[1]Assuming the encryption scheme is correct will probability 1, not just probability $1 - \mathsf{negl}(\lambda)$. This is not the case for *deniable encryption*, which is designed specifically to ensure that the encryption *can* be opened to something else. [11]

Decommit: The decommit algorithm $\pi \leftarrow \mathsf{VC.D}_{ck}(x, j)$ given the commitment key ck, the input $x \in (\{0,1\}^{\ell_\mathsf{b}})^L$ and an index $j \in [L]$, creates a proof of correct decommitment $\pi \in \{0,1\}^{\ell_\mathsf{d}}$.

Verify: The verify algorithm $\mathsf{VC.V}_{ck}(y, j, u, \pi)$ given the key ck and $y \in \{0,1\}^{\ell_\mathsf{c}}$ an integer index $j \in [L]$, a value $u \in \{0,1\}^{\ell_\mathsf{b}}$ and a proof $\pi \in \{0,1\}^{\ell_\mathsf{d}}$, outputs 1 for accept (that $y = \mathsf{VC.C}_{ck}(x)$ and $x_j = u$) or 0 for reject.

Definition 7.9 (Vector Commitment Scheme – Correctness)**.** A vector commitment scheme is *correct* if for any $L \leq 2^\lambda$ and $i, j \in [L]$, any $ck \leftarrow \mathsf{VC.G}(1^\lambda, L, i)$, $x \in (\{0,1\}^{\ell_\mathsf{b}})^L$, $\pi \leftarrow \mathsf{VC.D}_{ck}(x, j)$ it holds that $\mathsf{VC.V}_{ck}(\mathsf{VC.C}_{ck}(x), j, x_j, \pi) = 1$.

Definition 7.10 (Vector Commitment Scheme – Index Hiding)**.** We consider the following game between an attacker $\mathcal{A}$ and a challenger $\mathcal{C}$:

- The attacker $\mathcal{A}(1^\lambda)$ chooses an integer L and two indices $i_0 \neq i_1 \in [L]$;
- The challenger $\mathcal{C}$ chooses a bit $b \leftarrow \{0,1\}$ and sets $ck \leftarrow \mathsf{VC.G}(1^\lambda, L, i_b)$.
- The attacker $\mathcal{A}$ gets ck and outputs a guess bit g.

We say a vector commitment scheme is *index hiding* if for all PPT $\mathcal{A}$ we have

$$\left| \Pr[g = b] - \frac{1}{2} \right| < \mathsf{negl}(\lambda)$$

Definition 7.11 (Vector Commitment Scheme – Somewhere Statistically Binding)**.** We say ck is *statistically binding for index i* if there are no y, u, u', π, π' such that $u \neq u'$ and

$$\mathsf{VC.V}_{ck}(y, i, u, \pi) = \mathsf{VC.V}_{ck}(y, i, u', \pi') = 1$$

In Hubáček and Wichs [43] it is shown how to construct SSB vector commitments using homomorphic encryption. Notice that we do not require the scheme to be computationally binding, because this property follows from the properties of being somewhere statistically binding and index hiding: an adversary who can break the binding at some coordinate will learn that that coordinate was not statistically bound.

7.3 A protocol for anonymous steganography

We start with a high-level description of our protocol (in steps) before presenting the actual construction and proving that it satisfies our notion of anonymity.

First attempt: Let the encoding key ek be a key for a PRF f, and let the encoding procedure simply be a random looking symmetric encryption of x using this PRF. That is, $c^j = x^j \oplus f_{ek}(j)$ where x^j is the jth bits in x and c^j the jth bit in c. Clearly now the resulting document c is indistinguishable from other elements sampled from the uniform distribution over $\{0,1\}^\ell$.

In this first attempt we let the decoding key dk be the obfuscation of a circuit $C[i, ek, \gamma](t)$. The circuit contains two hard-wired secrets, the index of Lea's document $i \in [d]$ and the key for the PRF ek. It also contains the hash of the entire set of documents $\gamma = H(t)$. On input a database t the circuit checks if $\gamma = H(t)$ and if this is the case outputs x by decrypting t_i with ek.

Clearly this first attempt fails miserably since the size of the circuit is now proportional to the size of the entire database $t = d\ell$, which is even larger than the size of the secret message $|x| = \ell' \leq \ell$.

Second attempt:[2] To remove the dependency on the number of documents d, we include in the decoding key an encryption $\alpha = \mathsf{HE.E}_{pk}(i)$ of the index i (using the homomorphic encryption scheme), and an obfuscation of a (new) circuit $C[ek, sk, \gamma](\beta)$, which contains hardwired secrets ek and sk (the secret key for the homomorphic encryption scheme), as well as a hash $\gamma = H(\mathsf{HE.Eval}(\mathsf{mux}[t], \alpha))$, where the circuit $\mathsf{mux}[t](i)$ outputs t_i. The circuit C now checks that $\gamma = H(\beta)$ and if this is the case computes $t_i \leftarrow \mathsf{HE.D}_{sk}(\beta)$ using the secret key of the HE scheme, then decrypts t_i using the PRF key ek and outputs the secret message x. When Joe receives the decoding key dk, Joe constructs the circuit $\mathsf{mux}[t]$ (using the public t) and computes $\beta = \mathsf{HE.Eval}(\mathsf{mux}[t], \alpha)$. To learn the secret, he runs the obfuscated circuit on β.

In other words, we are now exploiting the compactness of the homomorphic encryption scheme to let Joe compute an encryption of the document $c = t_i$ from the public database t and the encryption of i. Since Lea the leaker can predict this

[2] A different approach at this stage could be to use $i\mathcal{O}$ for Turing machines [53]. Unfortunately, [53] uses *complexity-leveraging* and therefore must assume *sub-exponentially hard* $i\mathcal{O}$ for circuits, while the solution described next will be secure using only standard hardness.

ciphertext[3], she could construct a circuit which only decrypts when this particular ciphertext is provided as input. However, the size of β is proportional to $\mathsf{poly}(\lambda)+\ell$, so the obfuscation of C is still too long.[4]

Third attempt: To remove the dependency from the length of the document ℓ, we construct a circuit which takes as input an encryption of a single bit j of t_i instead of the whole ciphertext. However, we also need to make sure that the circuit only decrypts these particular ciphertexts, and does not help Joe in decrypting anything else. Moreover, the circuit must perform this check efficiently (meaning, independent of the size of ℓ), so we cannot simply "precompute" these ℓ ciphertexts and hardwire them into C.

This is where we use the vector commitment scheme: we let the decoding key include a commitment key ck. We include in the obfuscated circuit a commitment $\gamma = \mathsf{VC.C}_{ck}(\beta)$, where $\beta = (\beta^1, \ldots, \beta^\ell)$ is a vector of encryptions of bits, and we make sure that the circuit only helps Joe in decrypting these ℓ ciphertexts. In other words, we obfuscate the circuit $C[ek, sk, ck, \gamma](\beta', \pi', j)$ which first checks if $\mathsf{VC.V}_{ck}(\gamma, j, \beta', \pi') = 1$ and if this is the case it outputs the jth bit of x from the jth bit of the ciphertext $t_i^j \leftarrow \mathsf{HE.D}_{sk}(\beta')$. We have now almost achieved our goal, since the size of the decoding key is $\mathsf{poly}(\lambda \log(d\ell))$.

Final attempt: We now have to argue that our scheme is secure. Intuitively, while it is true that the index i is only sent in encrypted form, we have a problem since the obfuscated circuit contains the secret key for the homomorphic encryption scheme, and we therefore need a final fix to be able to argue that the adversary does not learn any information about i.

The final modification to our construction is to encrypt the index i twice under two independent public keys. From these encryptions Joe computes two independent encryptions of the bit t_i^j which he inputs to the obfuscated circuits together with proofs of decommitment. The circuit now outputs $\perp$ if any of the two decommitment proofs are incorrect, otherwise the circuit computes and outputs x^j from one of the two encryptions and ignores the second ciphertext.

Anonymity: Very informally, we can now prove that Joe cannot distinguish

[3]The evaluation algorithm $\mathsf{HE.Eval}$ can always be made deterministic since we do not need circuit privacy.

[4]Note that the decryption key also contains an encryption of i which depends logarithmically on d, but we are going to ignore all logarithmic factors.

between the decoding keys computed using indices i_0 and i_1 in the following way: we start with the case where the decoding key contains two encryptions of i_0 (this corresponds to the game in the definition with $b = 0$). Then we define a hybrid game where we change one of the two ciphertext from being an encryption of i_0 with an encryption of i_1. In particular, since we change the ciphertext which is ignored by the obfuscated circuit, this does not change the output of the circuit at all (and we can argue indistinguishability since the obfuscated circuit does not contain the secret key for this ciphertext). We also replace the random document c_{i_1} with an encryption of x with a new key for the PRF. Finally, we change the obfuscated circuit and let it recover the message x from the second ciphertext. Thanks to the SSB property of the commitment scheme it is possible to prove, in a series of hybrids, that the adversary cannot notice this change. To conclude the proof we repeat the hybrids (in inverse order) to reach a game which is identical to the definition of anonymity when $b = 1$.

The Actual Construction: A complete specification of our anonymous steganography scheme follows. Note that in our construction $\ell' = \ell$.

Key Generation: On input the security parameter λ the algorithm Gen samples a random key $ek \in \{0,1\}^\lambda$ for the PRF and outputs ek.

Encoding: On input a message $x \in \{0,1\}^\ell$ and an encoding key ek the algorithm Enc outputs an encoded message $c \in \{0,1\}^\ell$ where for each bit $j \in [\ell]$, $c^j = x^j \oplus f_{ek}(j)$.

Key Extraction: On input the encoding key ek, the database of documents t, and index i such that $t_i = c$ the algorithm KeyEx outputs a decoding key dk generated as follows:

1. For all $u \in \{0,1\}$ run $(pk_u, sk_u) \leftarrow \mathsf{HE.G}(1^\lambda)$ and $\alpha_u \leftarrow \mathsf{HE.E}_{pk_u}(i)$.
2. For all $j \in [\ell], u \in \{0,1\}$ run $\beta_u^j = \mathsf{HE.Eval}_{pk_u}(\mathsf{mux}[t,j], \alpha_u)$[5] where the circuit $\mathsf{mux}[t,j](i)$ outputs the j-th bit of the i-th document t_i^j;
3. For all $u \in \{0,1\}$ run $ck_u \leftarrow \mathsf{VC.G}(1^\lambda, \ell, 1)$ and $\gamma_u \leftarrow \mathsf{VC.C}_{ck_u}(\beta_u^1, \ldots, \beta_u^\ell)$.

[5]Note that we consider $\mathsf{HE.Eval}$ to be a deterministic algorithm. This can always be achieved by fixing the random tape of $\mathsf{HE.Eval}$ to some constant value.

4. Pick a random bit $\sigma \in \{0,1\}$.
5. Construct the circuit $C[ek, \sigma, sk_\sigma, ck_0, ck_1, \gamma_0, \gamma_1](\beta'_0, \beta'_1, \pi'_0, \pi'_1, j)$ as follows:
   (a) if$(\forall u \in \{0,1\} : \mathsf{VC.V}_{ck_u}(\gamma_u, j, \beta'_u, \pi'_u))$ output $\mathsf{HE.D}_{sk_\sigma}(\beta'_\sigma) \oplus f_{ek}(j)$;
   (b) else output $\perp$;
6. Compute an obfuscation $\bar{C} \leftarrow \mathcal{O}(C_\sigma)$ where C_σ is a shorthand for the circuit defined before, padded to length equal to $\max_{\tau,\rho}(C, C'_{\tau,\rho})$, where the circuits $C'_{\tau,\rho}$ are defined in the proof of security.
7. Output $dk = (pk_0, pk_1, \alpha_0, \alpha_1, ck_0, ck_1, \bar{C})$

Decoding: On input a decoding key dk and a database of documents t the algorithm Dec outputs a message x' in the following way:

1. Parse $dk = (pk_0, pk_1, \alpha_0, \alpha_1, ck_0, ck_1, \bar{C})$;
2. For all $j \in [\ell], u \in \{0,1\}$ run $\beta_u^j = \mathsf{HE.Eval}_{pk_u}(\mathsf{mux}[t, j], \alpha_u)$;
3. For all $u \in \{0,1\}$ run $\gamma_u \leftarrow \mathsf{VC.C}_{ck_u}(\beta_u^1, \ldots, \beta_u^\ell)$.
4. For all $j \in [\ell], u \in \{0,1\}$ compute $\pi_u^j \leftarrow \mathsf{VC.D}_{ck_u}((\beta_u^1, \ldots, \beta_u^\ell), j)$;
5. For all $j \in [\ell]$ output $(x')^j \leftarrow \bar{C}(\beta_0^j, \beta_1^j, \pi_0^j, \pi_1^j, j)$;

Theorem 7.1. *If a) f is PRF b)* $(\mathsf{VC.G}, \mathsf{VC.C}, \mathsf{VC.D}, \mathsf{VC.V})$ *is a vector commitment scheme satisfying Definitions 7.9-7.11 c)* $(\mathsf{HE.G}, \mathsf{HE.E}, \mathsf{HE.D}, \mathsf{HE.Eval})$ *is a homomorphic encryption scheme satisfying Definition 7.5-7.8 and d) $\mathcal{O}$ is an obfuscator for all polynomial size circuits satisfying Definition 7.4 then the anonymous steganography scheme* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{KeyEx}, \mathsf{Dec})$ *constructed above satisfies Definition 7.1 and 7.2 and has* $|dk| = \mathsf{poly}(\lambda \log(d\ell))$.

Proof. Correctness follows from inspection of the protocol. In particular, for each bit $j \in [\ell]$ Definition 7.4 Bullet 1 implies that

$$\bar{C}(\beta_0^j, \beta_1^j, \pi_0^j, \pi_1^j, j) = C[ek, \sigma, sk_\sigma, ck_0, ck_1, \gamma_0, \gamma_1](\beta_0^j, \beta_1^j, \pi_0^j, \pi_1^j, j)$$

By Definition 7.7, $\forall u \in \{0,1\}$ the ciphertext β_u^j is such that

$$\mathsf{HE.D}_{sk_u}(\beta_u^j) = \mathsf{mux}[t, j](\mathsf{HE.D}_{sk_u}(\alpha_u)) = \mathsf{mux}[t, j](i) = t_i^j.$$

Now, since $t_i^j = x^j \oplus f_{ek}(j)$ it follows that the output z of $\bar{C}$ is either $\perp$ or x^j. Finally, the circuit only outputs $\perp$ if $\exists u \in \{0,1\}$ such that $\mathsf{VC.V}_{ck_u}(\gamma_u, j, \beta_u^j, \pi_u^j) = 0$. But since

$$ck_u \leftarrow \mathsf{VC.G}(1^\lambda, \ell, 1), \gamma_u \leftarrow \mathsf{VC.C}_{ck_u}(\beta_u^1, \ldots, \beta_u^\ell), \pi_u^j \leftarrow \mathsf{VC.D}_{ck_u}((\beta_u^1, \ldots, \beta_u^\ell), j),$$

Definition 7.9 shows that $\mathsf{VC.V}_{ck_u}(\gamma_u, j, \beta_u^j, \pi_u^j) = 1$ so $\bar{C}$, and therefore Dec, outputs x^j.

We prove anonymity using a series of hybrid games. We start with a game which is equivalent to the definition when $b = 0$ and we end with a game which is equivalent to the definition when $b = 1$. We prove at each step that the next hybrid is indistinguishable from the previous. Therefore, at the end we conclude that the adversary cannot distinguish whether $b = 0$ or $b = 1$.

Hybrid 0: This is the same as the definition when $b = 0$. In particular, here it holds that

$$(\alpha_0, \alpha_1) \leftarrow (\mathsf{HE.E}_{pk_0}(i_0), \mathsf{HE.E}_{pk_1}(i_0)).$$

Hybrid 1: In the first hybrid we replace $\alpha_{1-\sigma}$ with $\alpha_{1-\sigma} \leftarrow \mathsf{HE.E}_{pk_{1-\sigma}}(i_1)$. Note that the circuit $C[ek, \sigma, sk_\sigma, ck_0, ck_1, \gamma_0, \gamma_1](\cdot)$ does *not* contain the secret key $sk_{1-\sigma}$, therefore any adversary that can distinguish between Hybrid 0 and 1 can be turned into an adversary which breaks the IND-CPA property of the HE scheme.

Hybrid 2: In the previous hybrids t_{i_1} is a random string from $\{0,1\}^\ell$. In this hybrid we replace t_{i_1} with an encryption of x using a new PRF key ek'. That is, for each bit $j \in [\ell]$ we set $t_{i_1}^j = x^j \oplus f_{ek'}(j)$. Clearly, any adversary that can distinguish between Hybrid 1 and Hybrid 2 can be used to break the PRF.

Hybrid 3.(τ, ρ): We now define a series of 2ℓ hybrids indexed by $\tau \in [\ell], \rho \in \{0,1\}$. In Hybrid 3.$(\tau, \rho)$ we replace the obfuscated circuit with the circuit:

$C'[\tau, \rho, ek, ek', \sigma, sk_0, sk_1, ck_0, ck_1, \gamma_0, \gamma_1](\beta_0', \beta_1', \pi_0', \pi_1', j)$:

1. if$(\exists u \in \{0,1\} : \mathsf{VC.V}_{ck_u}(\gamma_u, j, \beta_u', \pi_u') = 0)$ output $\perp$
2. else if$(j \geq \tau + \rho)$ output $\mathsf{HE.D}_{sk_\sigma}(\beta_\sigma') \oplus f_{ek}(j)$;
3. else output $\mathsf{HE.D}_{sk_{1-\sigma}}(\beta_{1-\sigma}') \oplus f_{ek'}(j)$;

We use $C'_{\tau,\rho}$ as a shorthand for a circuit defined as above, and which, if necessary, is padded to make C_σ and all the $C'_{\tau,\rho}$'s equally long.

In addition, we also replace the way the keys for the vector commitment schemes are generated. Remember that in the previous hybrids

$$\forall u \in \{0,1\} \quad ck_u \leftarrow \mathsf{VC.G}(1^\lambda, \ell, 1),$$

which are now replaced with

$$\forall u \in \{0,1\} \quad ck_u \leftarrow \mathsf{VC.G}(1^\lambda, \ell, \tau).$$

From inspection it is clear that in Hybrid $3.(1,0)$ is computationally indistinguishable from Hybrid 2 thanks to Definition 7.4 (Bullet 3) since 1) the keys ck_0, ck_1 are identically distributed and 2) the circuit $C'_{1,0}$ computes the same function as the circuit C obfuscated in Hybrid 2: since j is indexed starting from 1 we always have $j \geq 1 + 0$ and the branch (3) is never taken.

Next, we argue that Hybrid $3.(\tau,0)$ is indistinguishable from Hybrid $3.(\tau,1)$. First we note that the commitment keys ck_0, ck_1 are identically distributed in these two hybrids i.e., in both hybrids

$$\forall u \in \{0,1\} \quad ck_u \leftarrow \mathsf{VC.G}(1^\lambda, \ell, \tau).$$

The only difference between the two hybrids is what circuits are being obfuscated: in Hybrid $3.(\tau,0)$ we obfuscate $C'_{\tau,0}$ and in Hybrid $3.(\tau,1)$ we obfuscate $C'_{\tau,1}$. We will now argue that these two circuits give the same output on every input, and therefore an adversary that can distinguish between Hybrid $3.(\tau,0)$ and Hybrid $3.(\tau,1)$ can be used to break the indistinguishability obfuscator.

It follows from inspection that the two circuits behave differently only on inputs of the form $(\beta'_0, \beta'_1, \pi'_0, \pi'_1, \tau)$. On input of this form:

- $C'_{\tau,0}$ (since $j = \tau \geq \tau$) chooses branch (2) and outputs

$$x_0^j \leftarrow \mathsf{HE.D}_{sk_\sigma}(\beta'_\sigma) \oplus f_{ek}(j),$$

- $C'_{\tau,1}$ (since $j = \tau \not\geq \tau + 1$) chooses branch (3) and outputs

$$x_1^j \leftarrow \mathsf{HE.D}_{sk_{1-\sigma}}(\beta'_{1-\sigma}) \oplus f_{ek'}(j).$$

Now, the statistically binding property of the vector commitment scheme (Definition 7.11) allows us to conclude that there exists only one single pair (β'_0, β'_1) for which $C'_{\tau,0}$ and $C'_{\tau,1}$ do not output $\perp$ (remember that in both hybrids the commitment keys ck_0, ck_1 are statistically binding on index τ), namely the pair

$$\forall u \in \{0,1\} \quad \beta_u^j = \mathsf{HE.Eval}_{pk_u}(\mathsf{mux}[t,\tau], \alpha_u)$$

which decrypts to the pair $(t_{i_0}^j, t_{i_1}^j)$ (since we changed $\alpha_{1-\sigma}$ in Hybrid 1), which in turns were defined as (since we changed $t_{i_1}^j$ in Hybrid 2)

$$(t_{i_0}^j, t_{i_1}^j) = (x^j \oplus f_{ek}(j), x^j \oplus f_{ek'}(j))$$

which implies that $x_0^j = x_1^j$ and therefore the two circuits have the exact same input/output behavior.

Finally, we argue that Hybrid $3.(\tau, 1)$ is indistinguishable from Hybrid $3.(\tau+1, 0)$ for all $\tau \in [\ell]$ since by definition the circuits $C'_{\tau,1}$ and $C'_{\tau+1,0}$ are identical and the only difference between these hybrids is in the way the commitment keys ck_0, ck_1 are generated. In particular, the only difference is the index on which the keys are statistically binding. Therefore, any adversary who can distinguish between $3.(\tau, 1)$ and Hybrid $3.(\tau + 1, 0)$ can be used to break the index hiding property (Definition 7.10) of the vector commitment scheme.

This concludes the technical core of our proof, what is left now is to make few simple changes to go from Hybrid $3.(\ell, 0)$ to the game from Definition 7.2 when $b = 1$.

Hybrid 4: In this hybrid we replace the obfuscated circuit with

$$C[ek', \sigma', sk_{\sigma'}, ck_0, ck_1, \gamma_0, \gamma_1](\cdot)$$

where $\sigma' = 1 - \sigma$. It is easy to see that the input/output behavior of this circuit is exactly the same as $C'_{\ell,1}$: since $\forall j \in [\ell] : j \not\geq \ell + 1$ the circuit $C'_{\ell,1}$ always executes

branch 3) and therefore an adversary that can distinguish between Hybrid 4 and Hybrid 3.$(\ell, 0)$ can be used to break the indistinguishability obfuscator.

Hybrids 5, 6, 7: In Hybrid 5 we change the distribution of both commitment keys ck_0, ck_1 to $\mathsf{VC.G}(1^\lambda, \ell, 1)$ whereas in Hybrid 4 they were both sampled as $\mathsf{VC.G}(1^\lambda, \ell, \ell)$. Indistinguishability follows from the index hiding property. In Hybrids 6 we replace t_{i_0} with a uniformly random string in $\{0,1\}^\ell$ whereas in the previous hybrid it was an encryption of x using the PRF f with key ek. Since the obfuscated circuit no longer contains ek we can use an adversary which distinguishes between Hybrids 5 and 6 to break the PRF. In Hybrid 7 we replace $\alpha_{1-\sigma'}$ (which in the previous hybrid is an encryption of i_0) with an encryption of i_1. Since the obfuscated circuit no longer contains $sk_{1-\sigma'} = sk_\sigma$ we can use an adversary which distinguishes between Hybrids 6 and 7 to break the IND-CPA property of the encryption scheme. Now Hybrid 7 is exactly as the definition of anonymity with $b = 1$ with a random bit $\sigma' = 1 - \sigma$ (which is distributed uniformly at random) and a random encoding key ek'. This concludes the proof. □

Our theorem, together with the results of [43] implies the following.

Corollary 7.2. *Assuming the existence of homomorphic encryption and indistinguishability obfuscators for all polynomially sized circuits, there exists an anonymous steganography scheme.*

7.4 Lower bound

In this section we show that no correct anonymous steganography scheme can have a decoding key of size $O(\log(\lambda))$. Since the decoding key must be sent over an anonymous channel, this gives a lower bound on the number of bits which are necessary to bootstrap anonymous communication.

To show this, we find a strategy for Joe that gives him a higher probability of guessing the leaker than if he guessed uniformly at random.

Our lower bound applies to a more general class of anonymous steganography schemes than defined earlier, in particular it also applies to *reactive* schemes where the leaker can post multiple documents to the website, as a function of the documents

posted by other users. We define a *reactive anonymous steganography* scheme as a tuple of algorithms $\pi = (\mathsf{Enc}, \mathsf{KeyEx}, \mathsf{Dec})$ where:

- $(t_k, state_j) \leftarrow \mathsf{Enc}_{ek}(x, t^{k-1}, state_{j-1})$ is an algorithm which takes as input a message $x \in \{0,1\}^{\ell'}$, a sequence of documents t^{k-1} (which represents the set of documents previously sent) and a state of the leaker, and outputs a new document $t_k \in \{0,1\}^{\ell}$, together with a new state.
- $dk \leftarrow \mathsf{KeyEx}_{ek}(t^d, state)$ is an algorithm which takes as input a transcript of all documents sent and the current state of the leaker and outputs a decryption key $dk \in \{0,1\}^s$.
- $x' = \mathsf{Dec}_{dk}(t^d)$ is an algorithm that given transcript t^d returns a guess x of what the secret is in a deterministic way.

To use a reactive anonymous steganography scheme, the leaker's index i is chosen uniformly at random from $\{1, \ldots, n\}$ where n is the number of players. For each k from 1 to d we generate a document t_k. If $k \not\equiv i \mod n$ we let $t_k \leftarrow \{0,1\}^{\ell}$. This corresponds to the non-leakers sending a message. When $k \equiv i \mod n$ we define $(t_k, state_j) \leftarrow \mathsf{Enc}_{ek}(x, t^{k-1}, state_{j-1})$. Then we define $dk \leftarrow \mathsf{KeyEx}_{ek}(t^d, state)$ and $x' = \mathsf{Dec}_{dk}(t^d)$. Here dk is the message that Lea would send over the small anonymous channel.[6]

The definition of q-correctness for reactive schemes is the same as for standard schemes, but our definition of anonymity is weaker because we do not allow the adversary to choose the documents for the honest users. By using a weaker definition of anonymity, we get a stronger lower bound.

Definition 7.12 (Correctness)**.** A reactive anonymous steganography scheme is *q-correct* if for all λ and $x \in \{0,1\}^{\ell'(\lambda)}$ we have

$$\Pr\left[\mathsf{Dec}_{dk}\left(t^d\right) = x\right] \geq q.$$

where t and dk is chosen as above.

[6]Note that an anonymous steganography scheme can easily be turned into a reactive anonymous steganography scheme by combining Gen and Enc into one algorithm and storing ek in the *state*.

Definition 7.13 (Weak Anonymity)**.** Consider the following game between an adversary A and a challenger C

1. The adversary A outputs a message $x \in \{0,1\}^{\ell'}$;
2. The challenger C samples random $i \in [n]$, and generates t^d, dk as described above
3. The challenger C outputs t^d, dk
4. A outputs a guess g;

We say that an adversary has advantage $\epsilon(\lambda)$ if $\left|\Pr[g = i] - \frac{1}{n}\right| \geq \epsilon(\lambda)$. We say a reactive anonymous steganography scheme provides *anonymity* if, for any adversary, the advantage is negligible.

In this model we assume that the non-leakers' documents are chosen uniformly at random. This is realistic in the case where we use steganography, so that each t_k is the result of extracting information from a larger file. We could also define a more general model where the distribution of each non-leaker's documents t_k depends on the previous transcript. The proof of our impossibility results works as long as the adversary can sample from $T_k|_{T^{k-1}=t^{k-1}, i \not\equiv k \mod n}$ in polynomial time. An assumption like this is necessary to ensure that running a cryptographic protocol, for example multi-party computation, is not considered to be innocent communication. Using this general model, we can also model the more realistic situation where the players do not take turns in sending documents, but at each step only send a document with some small probability. To do this, we just consider "no document" to be a possible value of t_k.

We could also generalize the model to let the leaker use the anonymous channel at any time, not just after all the documents have been sent. However, in such a model, the anonymous channel transmits more information than just the number of bits send over the channel: the times at which the bits are sent can be used to transmit information [44]. For the number of bits sent to be a fair measure of how much information is transferred over the channel, we should only allow the leaker

to use the channel when Joe knows she would use the anonymous channel[7], and the leaker should only be allowed to send messages from a prefix-free code, which might depend on the transcript, but should be computable in polynomial time for Joe. This ensures that he receives another bit if and only if he expects to receive a bit, thus, he only gets information from the *values* of the bits he receives, not from whether he receives a bit. Our impossibility result also holds for this more general model, however, to keep the notation simple, we will assume that the anonymous channel is only used at the end.

Also note that we assume that Eve knows the secret. If we assume that Eve does not know the message X, but its distribution is known to everyone, there is a protocol π that reveals the secret with probability $\frac{1}{n}$ but reveals no information about the leaker: Simply let the first player say a random number i between 1 and n. Then player i sends a message. If she is the leaker she sends x otherwise she just send a random message from the same distribution as X. Then Dec is just the message she sent.

Finally, we could generalize the model by allowing access to public randomness. However, this does not help the players: as none of the players are controlled by the adversary, the players can generate trusted randomness themselves.

We let $T' = (T'_1, \ldots, T'_d)$ denote the random variable where each T'_i is uniformly distributed on $\{0,1\}^\ell$. In particular $T'|_{T'^k=t^k}$ is the distribution the transcript would follow if the first k documents are given by t^k and all the players were non-leakers. We let dk' be uniformly distributed on $\{0,1\}^s$. Joe can sample from both $T'|_{T'^k=t^k}$ and dk' and he can compute Dec. His strategy to guess the leaker, given a transcript t, will be to estimate $\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^k = t^k)$ for each $k \leq d$. That is, given that the transcript of the first k documents is t^k and all later documents are chosen as if the sender was not a leaker and the anonymous channel just sends random bits, what is the probability that the result is x? He can estimate this by sampling: given t^k he randomly generates t^d and dk', and then he computes Dec of this extended transcript.

If we assume that the protocol π is symmetric[8] in the messages x, then before

[7]That is, there should be a polynomial time algorithm that given previous transcript t^k and previous messages over the anonymous channel decides if the leaker sends a message over the anonymous channel.

[8]By this we mean that for random transcript T' and random dk' the result $\mathsf{Dec}_{dk'}(T')$ is uni-

any documents are sent, we have $\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^0 = t^0) = 2^{-\ell'}$. If, after all the documents are sent, there exists a key $dk' \in \{0,1\}^s$ such that $\mathsf{Dec}_{dk'}(t^d) = x$, then for a random dk' we must have $\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^d = t^d) \geq 2^{-s}$. As $s < \ell'$ the documents in t^d must have increased the probability of decoding to x. The non-leakers' documents affect this probability, but *in expectation* they do not, so in most cases most of this increase will have to come from the leaker, that is, these probabilities would tend to be higher just after the leaker's documents than just before. Of course, a leaking player might send some documents that lowers $\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^k = t^k)$ to confuse Joe, so we need a way to add up all the changes a players does to $\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^k = t^k)$. The simplest idea would be to compute the additive difference

$$\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^k = t^k) - \Pr(\mathsf{Dec}_{dk'}(T') = x | T'^{k-1} = t^{k-1})$$

and add these for each player. However, the following example shows that if Joe uses this strategy for determining who the leakers is, there is a protocol for two people which ensures that $\Pr(\mathsf{Dec}_{dk}(T) = X) > 1 - \epsilon$ but Joe will almost always guess that the non-leaker is the leaker.

Example 3. Consider this protocol for two players, where one of them wants to leak one bit. We have $s = 0$, that is dk is the empty string and will be omitted from the notation. First we define the function Dec. This function looks at the two first documents. If none of these are 0^ℓ, it returns the first bit of the third document. Otherwise, it defines the *leader* to be the first player who send 0^ℓ. Next Dec looks at the first time the leader sent a document different from 0^ℓ. If this number represents a binary number less than $\frac{9}{10} \cdot 2^\ell$, then Dec returns the last bit of the document before, otherwise it outputs the opposite value of that bit. If the leader only sends the document 0^ℓ, then the output of Dec is just the last bit sent by the other player.

The leaker's strategy is to become the leader. There is extremely small probability that the non-leaker sends 0^ℓ in his first document, so we will ignore this case. Otherwise, the leaker sends 0^ℓ in her first document and becomes the leader. When sending her next document, she looks at the last document from the non-leaker. If

formly distributed. In the formal proof we will show why we can make this assumption.

it ended in 0, Joe will think there is 90% chance that 0 it is output and 10% chance that the output will be 1, and if it ended in 1 it is the other way around. If the last bit in the non-leakers document is the bit the leakers wants to leak, she just sends the document $0^{\ell-1}1$. To Joe, this will look like the non-leaker raised the probability of this outcome from 50% to 90% and then the leaker raised it to 100%. Thus, Joe will guess that the non-leaker was the leaker.

If the last bit of the previous document was the opposite of what the leaker wanted to reveal, she will "reset" by sending 0^{ℓ}. This brings Joe's estimate that the result will be 1 back to 50%. The leaker will continue "resetting" until the non-leaker have sent a document ending in the correct bit more times than he has sent a document ending in the wrong bit. For sufficiently high d, this will happen with high probability, and then the leaker sends $0^{\ell}1$. This ensures that $\mathsf{Dec}(T)$ gives the correct value and that Joe will guess that the non-leaker was the leaker.

If the leaker wants to send many bits, the players can just repeat this protocol.

Obviously, the above protocol for revealing information is not a good protocol: it should be clear to Joe that the leader is not sending random documents.

As the additive difference does not work, Joe will instead look at the multiplicative factor

$$\frac{\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^k = t^k)}{\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^{k-1} = t^{k-1})}.$$

Definition 7.14. For a transcript t the *multiplicative factor* $mf_{j,[k_0,k_1]}$ of player j over the time interval $[k_0, k_1]$ is given by

$$mf_{j,[k_0,k_1]}(t) = \prod_{k \in [k_0,k_1] \cap (j+n\mathbb{N})} \frac{\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^k = t^k)}{\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^{k-1} = t^{k-1})},$$

We also define

$$mf_{-j,[k_0,k_1]}(t) = \prod_{k \in [k_0,k_1] \setminus (j+n\mathbb{N})} \frac{\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^k = t^k)}{\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^{k-1} = t^{k-1})},$$

If we use the multiplicative factor on the non-leaker in the protocol in Example 3 we see that for each document sent by the non-leaker there is probability 0.5 that his multiplicative factor increases by a factor 1.8 and probability 0.5 that it is multiplied by a factor 0.2. Thus, if the non-leaker first sends a document which

decreases $\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^k = t^k)$ from 0.5 to 0.1 and later a document that increases it from 0.5 to 0.9, the two documents no longer cancel each other out: they result in multiplying the multiplicative factor by 0.36.

For fixed k_0 and non-leaking player j the sequence

$$mf_{j,[k_0,k_0]}(T), mf_{j,[k_0,k_0+1]}(T), \ldots$$

is a martingale. This implies that if we consider the first k_1-2 documents to be fixed and player 1 sends a document at time $k_1 - 1$ and player 2 at time k_1, then player 1's document can affect the distribution of $mf_{2,[k_0,k_1]}(T')|_{T'^{k_1-1}=t^{k_1-1}}$ but no matter what document t_{k_1-1} player 1 sends, $mf_{2,[k_0,k_1]}(T')|_{T'^{k_1-1}=t^{k_1-1}}$ will have expectation $mf_{2,[k_0,k_1-1]}(t^{k_1-1})$. Similar statements holds for the sum of additive differences, but the advantage of the multiplicative factor is that it is non-negative. For example, as the multiplicative factor starts at 1 there is probability at most 0.1 that it will ever be at least 10. Thus, while the leaker's multiplicative factor has to be large in most cases, all the non-leakers will with high probability have small multiplicative factors. The same does not hold for the sum of additive differences, because as Example 3 shows, you can have a probability arbitrarily close to 1 that a non-leaker's sum of additive differences increases to 0.4 (or any other positive number) as long as there is a small probability that it decreases to negative values of large absolute value.

Proposition 7.3. *For j and k_0, k_1 we have:*

$$\mathbb{E}_{t' \sim T'|_{T^{k_1-1}=t^{k_1-1}}} mf_{j,[k_0,k_1]}(t') = mf_{j,[k_0,k_1-1]}(t)$$

Proof. For $k_1 \not\equiv j \mod n$ we have $mf_{j,[k_0,k_1]}(t) = mf_{j,[k_0,k_1-1]}(t)$ for any t so the

statement is trivially true. For $k_1 \equiv j \mod n$ and fixed t we have

$$\begin{aligned}
&\mathbb{E}_{t' \sim T'|_{T'^{k_1-1}=t^{k_1-1}}} mf_{j,[k_0,k_1]}(t') \\
&= \sum_{t'} \Pr(T' = t' | T'^{k_1-1} = t^{k_1-1}) \prod_{k \in [k_0,k_1] \cap (j+n\mathbb{N})} \frac{\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^{k_1} = t'^{k_1})}{\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^{k_1-1} = t'^{k_1-1})} \\
&= \sum_{t'} \frac{\Pr(T' = t' | T'^{k_1-1} = t^{k_1-1}) \Pr(\mathsf{Dec}_{dk'}(T') = x | T'^{k_1} = t'^{k_1})}{\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^{k_1-1} = t'^{k_1-1})} mf_{j,[k_0,k_1-1]}(t) \\
&= \sum_{t'} \frac{\Pr(T' = t' | T'^{k_1} = t'^{k_1}) \Pr(T'^{k_1} = t'^{k_1} | T'^{k_1-1} = t^{k_1-1}) \Pr(\mathsf{Dec}_{dk'}(T') = x | T'^{k_1} = t'^{k_1})}{\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^{k_1-1} = t'^{k_1-1})} \\
&\quad \cdot mf_{j,[k_0,k_1-1]}(t) \\
&= \sum_{t'^{k_1}} \frac{\Pr(T'^{k_1} = t'^{k_1} | T'^{k_1-1} = t^{k_1-1}) \Pr(\mathsf{Dec}_{dk'}(T') = x | T'^{k_1} = t'^{k_1})}{\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^{k_1-1} = t'^{k_1-1})} mf_{j,[k_0,k_1-1]}(t) \\
&= \sum_{t'^{k_1}} \frac{\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^{k_1-1} = t'^{k_1-1})}{\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^{k_1-1} = t'^{k_1-1})} mf_{j,[k_0,k_1-1]}(t) \\
&= mf_{j,[k_0,k_1-1]}(t)
\end{aligned}$$

Here the second equality is obtained by pulling the $k = k_1$ term out of the product, and recognising the resulting product as $mf_{j,[k_0,k_1-1]}(t)$. □

By sampling $T'^d|_{T'^k=t^k}$ and dk' Joe can estimate $\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^k = t^k)$ with a small *additive* error, but when the probability is small, there might still be a large *multiplicative* error. In particular, Joe can only do polynomially many samples, so when $\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^k = t^k)$ is less than polynomially small Joe will most likely estimate it to be 0.[9] Instead, the idea is to estimate the multiplicative factor starting from some time k_0 such that $\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^k = t^k)$ is not too small for any $k \geq k_0$. The following proposition is useful when choosing k_0 and choosing how many samples we make.

Definition 7.15. In the following we say that Joe's estimate of $\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^k = t^k)$ is *bad* if $\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^k = t^k) \geq \frac{\epsilon^2}{2^{s+7}d^2}$ but his estimate is not in

[9] This is the reason that anonymous steganography with small anonymous channel works at all: we keep $\Pr(\mathsf{Dec}_{dk'}(T') = x | T'^k = t^k)$ exponentially small until Lea uses the anonymous channel. When Lea then uses the anonymous channel to send dk, the probability of x being the output increases from exponentially small to 1.

the interval

$$\left[\left(1-\frac{1}{2d}\right)\Pr(\mathsf{Dec}_{dk'}(T')=x|T'^k=t^k),\left(1+\frac{1}{2d}\right)\Pr(\mathsf{Dec}_{dk'}(T')=x|T'^k=t^k)\right].$$

Proposition 7.4. *Assume that Joe makes* $\frac{3\cdot 2^{s+9}d^4}{\epsilon^2}\ln\left(\frac{4d}{\epsilon}\right)$ *samples of* $\mathsf{Dec}_{dk'}(T')|_{T'^k=t^k}$ *to estimate* $\Pr(\mathsf{Dec}_{dk'}(T')=x|T'^k=t^k)$.

No matter the true value of $\Pr(\mathsf{Dec}_{dk'}(T')=x|T'^k=t^k)$*, there is probability at most* $\frac{\epsilon}{2d}$ *that his estimate is bad.*

Proof. By definition, the estimate cannot be bad unless $\Pr(\mathsf{Dec}_{dk'}(T')=x|T'^k=t^k)\geq\frac{\epsilon^2}{2^{s+7}d^2}$, so in the following we assume this inequality holds.

Let Y_i be the random variable that is 1 if the ith sample of T', dk' satisfy $\mathsf{Dec}_{dk'}(T')=x$ and is 0 otherwise. Let Y be the sum of all the Y_is. We see that

$$\mu:=\mathbb{E}Y\geq\frac{3\cdot 2^{s+9}d^4}{\epsilon^2}\ln\left(\frac{4d}{\epsilon}\right)\frac{\epsilon^2}{2^{s+7}d^2}=12d^2\ln\left(\frac{4d}{\epsilon}\right).$$

Joe's estimate of $\Pr(\mathsf{Dec}_{dk'}(T')=x|T'^k=t^k)$ is going to be divided by number of samples. Thus, Joe's estimate is going to be within a factor $1\pm\delta$ of the correct probability if Y is within a factor $1\pm\delta$ of μ.

As the Y_is are independent and only take the values 0 and 1, we can use the multiplicative Chernoff bound [61, Theorem 4.1 and Theorem 4.2]:

$$\begin{aligned}\Pr(Y\geq(1+\delta)\mu)&\leq\left(\frac{e^\delta}{(1+\delta)^{1+\delta}}\right)^\mu\leq e^{\frac{-\delta^2\mu}{3}},\text{ for }\delta<1\\ \Pr(Y\leq(1-\delta)\mu)&\leq e^{\frac{-\delta^2\mu}{2}}\leq e^{\frac{-\delta^2\mu}{3}}.\end{aligned}$$

So the probability of a bad estimate is at most $2e^{\frac{-\delta^2\mu}{3}}$, where $\delta=\frac{1}{2d}$ and $\mu\geq 12d^2\ln(\frac{4d}{\epsilon})$. We get

$$2e^{\frac{-\delta^2\mu}{3}}\leq 2e^{-\frac{\left(\frac{1}{2d}\right)^2 12d^2\ln\left(\frac{4d}{\epsilon}\right)}{3}}=2e^{-\ln\left(\frac{4d}{\epsilon}\right)}=\frac{\epsilon}{2d}.$$

□

Proposition 7.5. *Fix a leaker protocol and a number* $m_0>2$*. Let* i *denote the*

(random) index of the leaker. For a random transcript T given $X = x$ there is probability at most $\frac{4d}{m_0}$ that there exists $j \neq i$ and k_0 such that $mf_{j,[k_0,d]}(T) \geq \frac{m_0}{2}$ or $mf_{-i,[k_0,d]}(T) \geq \frac{m_0}{2}$.

Proof. For fixed k_0, and non-leaker j we have $\mathbb{E}_{T|X=x}\left(mf_{j,[k_0,d]}(T)\right) = 1$ no matter the leakers strategy. As

$$mf_{j,[k_0,d]}(t) \geq 0$$

this implies that

$$\Pr\left(mf_{j,[k_0,d]}(T) \geq \frac{m_0}{2} \middle| X = x\right) \leq \frac{2}{m_0}.$$

Similarly for $mf_{-i,[k_0,d]}$. We have

$$mf_{j,[k_0,d]}(t) = mf_{j,[k_0-1,d]}(t)$$

if player j does not send the k_0'th document, so for fixed t there are only d different values (not counting 1) of $mf_{j,[k_0,d]}(t)$ with $j \neq i$ and $k_0 \leq d$. By the union bound, the probability that one of the $mf_{j,[k_0,d]}(t)$'s or one of the $mf_{-i,[k_0,d]}(t)$'s are above $\frac{m_0}{2}$ is at most $2d\frac{2}{m_0} = \frac{4d}{m_0}$. □

Now we are ready to prove the impossibility result.

Theorem 7.6. *Let ϵ be a function in λ such that $\frac{1}{\epsilon}$ is bounded by a polynomial, and let π be a $q(\lambda)$-correct reactive anonymous steganography scheme with $s(\lambda) = O(\log(\lambda))$ and $\ell' \geq s+7+2\log(d)-2\log(\epsilon)$. Now there is a probabilistic polynomial time Turing machine $\mathcal{A}$ that takes input t and x and outputs the leaker identity with probability*

$$q(\lambda) + \frac{1-q(\lambda)}{n(\lambda)} - \epsilon(\lambda).$$

Notice that we cannot do better than $q + \frac{1-q}{n}$. The players could use a protocol where with probability q the leaker reveals herself and the information and otherwise no-one reveals any information. This protocol succeeds with probability q, and when it does, Joe will guess the leaker. With probability $1-q$ it does not succeed, and Joe has probability $\frac{1}{n}$ of guessing the leaker. In total Joe will guess the leaker with probability $q + \frac{1-q}{n}$.

Proof. Let π be a reactive anonymous steganography scheme. We assume that for random T' and dk' the random variable $\mathsf{Dec}_{dk'}(T')$ is uniformly distributed[10] on $\{0,1\}^{\ell'}$ and we will just let the adversary send $0^{\ell'}$ in the anonymity game.

Let $m_0 = \frac{8d}{\epsilon}$. Consider a random transcript t. If for some k_0 and some non-leaker j we have $mf_{j,[k_0,d]} \geq \frac{m_0}{2}$ or if we have $mf_{-i,[k_0,d]} \geq \frac{m_0}{2}$ we set $E = 1$.

First Joe will estimate $\Pr(\mathsf{Dec}_{dk'}(T') = 0^{\ell'} | T'^k = t^k)$ for all k using

$$\frac{3 \cdot 2^{s+9} d^4}{\epsilon^2} \ln\left(\frac{4d}{\epsilon}\right)$$

samples for each k. Set $E = 1$ if at least one of these estimates is bad. In all cases where E has not been defined yet we set $E = 0$. By the above propositions and the union bound, $\Pr(E = 1) \leq \frac{4d}{\frac{8d}{\epsilon(\lambda)}} + d\frac{\epsilon(\lambda)}{2d} = \epsilon(\lambda)$.

Now let k_0 be the smallest number such that for all $k \geq k_0$ Joe's estimate of $\Pr(\mathsf{Dec}_{dk'}(T') = 0^{\ell'} | T'^k = t^k)$ is at least $\frac{\epsilon^2}{2^{s+7}d^2}$. The idea would be to estimate the multiplication factors $mf_{j,[k_0+1,d]}$, but the problem is that $\Pr(\mathsf{Dec}_{dk'}(T') = 0^{\ell'} | T'^{k_0} = t^{k_0})$ could be large (even 1) even though $\Pr(\mathsf{Dec}_{dk'}(T') = 0^{\ell'} | T'^{k_0-1} = t^{k_0-1})$ is small, so the players might not reveal any information after the $k_0 - 1$'th document. Thus, Joe needs to include the $k_0 - 1$'th document in his estimate of the multiplication factors, but his estimate of $\Pr(\mathsf{Dec}_{dk'}(T') = 0^{\ell'} | T'^{k_0-1} = t^{k_0-1})$ might be off by a large factor. To solve this problem, we define

$$mf_j = \begin{cases} mf_{j,[k_0+1,d]} & \text{if } j \not\equiv k_0 - 1 \mod n \\ mf_{j,[k_0+1,d]} \frac{\Pr(\mathsf{Dec}_{dk'}(T') = 0^{\ell'} | T'^{k_0} = t^{k_0})}{(1-\frac{1}{2d})^{-1}\frac{\epsilon^2}{2^{s+7}d^2}} & \text{if } j \equiv k_0 - 1 \mod n \end{cases}$$

that is, we pretend that $\Pr(\mathsf{Dec}_{dk'}(T') = 0^{\ell'} | T'^{k_0} = t^{k_0}) = (1 - \frac{1}{2d})^{-1}\frac{\epsilon^2}{2^{s+7}d^2}$ and then use $mf_{j,[k_0,d]}$. We define mf_{-i} the similar way. Joe's estimate of $\Pr(\mathsf{Dec}(T) = X | T^{k_0-1} = t^{k_0-1})$ is less than $\frac{\epsilon^2}{2^{s+7}d^2}$, otherwise k_0 would have been lower (here we are using the assumption $h \geq s + 7 + 2\log(d) - 2\log(\epsilon)$. Without this, k_0 could be

[10] If this is not the case, we can define a reactive anonymous scheme $\widetilde{\pi}$ where this is the case: just let X' be uniformly distributed on $\{0,1\}^{\ell'}$, let $\widetilde{\mathsf{Enc}}(x, t^k, state) = \mathsf{Enc}(x \oplus X', t^k, state)$ and $\widetilde{\mathsf{Dec}_{dk}}(t) = X' \oplus \mathsf{Dec}_{dk}(t)$, where $\oplus$ is bitwise addition modulo 2. To use $\widetilde{\pi}$ we would need ℓ' bits of public randomness to give us X'. To get this, we can just increase ℓ by ℓ' and let X' be the last ℓ' bits of the first document.

1). Thus, if this estimate it not bad we must have

$$\Pr(\mathsf{Dec}_{dk'}(T') = 0^{\ell'} | T'^{k_0-1} = t^{k_0-1}) \leq \left(1 - \frac{1}{2d}\right)^{-1} \frac{\epsilon^2}{2^{s+7}d^2}$$

So if $E = 0$ then $mf_j \leq mf_{j,[k_0,d]} \leq \frac{m_0}{2}$. Similar for mf_{-i}.

If $E = 0$ then $mf_j \leq \frac{m_0}{2}$ for all $j \neq i$ and $mf_{-i} \leq \frac{m_0}{2}$. Furthermore, as all of Joe's estimate are good, his estimate of mf_j is off by at most a factor $\left(1 - \frac{1}{2d}\right)^{-d} < 2$. Now we define Joe's guess: if exactly one of his estimated mf_j's are above m_0 he guesses that this player j is the leaker. Otherwise, he chooses his guess uniformly at random from all the players. There are two ways $\Pr(\mathsf{Dec}_{dk'}(T') = 0^{\ell'} | T'^k = t^k)$ can increase as k increases[11]: by the leaker sending documents or by a non-leaker sending documents. In the cases where $E = 0$ and Joe's estimate of mf_i is less than m_0 we know that the contribution from the leaker's documents is a factor less than $2m_0$. As $E = 0$ we also know that the total contribution from all the non-leakers is at most a factor $\frac{m_0}{2}$. So when only dk' has not been revealed to Joe we have

$$\Pr(\mathsf{Dec}_{dk'}(T) = X | T = t^d) < \frac{\epsilon^2}{2^{s+7}d^2} \left(1 - \frac{1}{2d}\right)^{-1} 2m_0 \frac{m_0}{2} \leq \frac{\epsilon^2}{2^{s+6}d^2} m_0^2 = 2^{-s}$$

As the only randomness left to be revealed[12] is dk' which is uniformly distributed on a set of size 2^{-s}, we know that

$$\Pr(\mathsf{Dec}_{dk'}(T) = 0^{\ell'} | T = t^d)$$

is a multiple of 2^{-s}. This implies

$$\Pr(\mathsf{Dec}_{dk'}(T) = 0^{\ell'} | T = t^d) = 0.$$

[11] If we allow the leaker to send anonymous bits before the end of the open communication, this is a third way $\Pr(\mathsf{Dec}_{dk'}(T') = 0^{\ell'} | T'^k = t^k)$ can increase. However, if the times where the anonymous channel is used are predictable by Joe, he can still sample as if the anonymous bits where random. This way, each anonymous bits makes $\Pr(\mathsf{Dec}_{dk'}(T') = 0^{\ell'} | T'^k = t^k)$ increase by at most a factor 2. If the leaker can only send s anonymous bit in total this only moves a factor 2^s increase in $\Pr(\mathsf{Dec}_{dk'}(T') = 0^{\ell'} | T'^k = t^k)$ from a later point in the proof to here.

[12] Here we are using that Dec is deterministic. However, allowing it to be non-deterministic does not help: we could just increase ℓ and let Dec use the extra bits in each document as randomness instead of using a random tape.

In other words, if $\mathsf{Dec}_{dk}(T) = 0^{\ell'}$ and $E = 0$ then A must output i. Furthermore, in all other cases where $E = 0$ Joe will either guess the leaker correctly (because Joe's estimate of mf_i is sufficiently high) or guess uniformly among all the players. The probability that Joe is correct is now

$$\begin{aligned}\Pr(g = i) \geq & \Pr(\mathsf{Dec}_{dk}(T) = 0^{\ell'}, E = 0) + \frac{\Pr(\mathsf{Dec}_{dk}(T) \neq 0^{\ell'}, E = 0)}{n} \\ = & \Pr(\mathsf{Dec}_{dk}(T) = 0^{\ell'}) - \Pr(\mathsf{Dec}_{dk}(T) = 0^{\ell'}, E = 1) \\ & + \frac{\Pr(\mathsf{Dec}_{dk}(T) \neq 0^{\ell'})}{n} - \frac{\Pr(\mathsf{Dec}_{dk}(T) \neq 0^{\ell'}, E = 1)}{n} \\ \geq & q + \frac{1-q}{n} - \Pr(E = 1) \geq q + \frac{1-q}{n} - \epsilon.\end{aligned}$$

□

Finally we can conclude that:

Corollary 7.7. *If π is a reactive anonymous steganography scheme with $s = O(\log(\lambda))$, d polynomial in λ and $\frac{\ell'}{\log(\lambda)} \to \infty$ that ensures weak anonymity, then the probability of correctness q tends to 0 as $\lambda \to \infty$.*

Proof. Let π be as in the assumption and define

$$\epsilon = \max\left(\lambda^{-1}, 2^{-\frac{s+7+2\log(d)-\ell'}{2}}\right)$$

By assumption, $s = O(\log(\lambda))$, $\log(d) = O(\log(\lambda))$, and $\frac{\ell'}{\log(\lambda)} \to \infty$, so $\epsilon \to 0$. The parameters satisfy the assumptions in Theorem 7.6 so there is an adversary that can guess the leaker with probability

$$q + \frac{1-q}{n} - \epsilon = \frac{1}{n} + \frac{n-1}{n}q - \epsilon \geq \frac{1}{n} + \frac{q(n-1) - n\epsilon}{n}.$$

As π ensures anonymity, $\frac{q(n-1)-n\epsilon}{n}$ must be negligible and as $\epsilon \to 0$ we must have $q \to 0$. □

Chapter 8

Summary and Conclusions

In Chapter 3 we studied the problem of sending information anonymously in the presence of what might be the strongest possible passive adversary who cannot read people's minds: an adversary who has unbounded computational power, who can observe all messages and (this assumption was implicit) knows all of people's shared randomness. Intuitively, it should be impossible to send information anonymously, when such an adversary exists: any hint towards a secret should increase the adversary's suspicion towards the leaker. We formalized this intuition by defining a measure of suspicion and showed that it exactly captures the price of revealing information: when you reveal one bit of information, your suspicion must increase by one in expectation. Conversely, it is always possible to reveal one bit of information while only increasing your expected suspicion by one.

We used this measure to show that if leakers just want to preserve reasonable doubt, they can reveal some amount of information. However, even in the best case models[1], each leaker can reveal at most $\frac{-\log(1-b_m)}{b_m} - \log(e)$, where b_m is the threshold of reasonable doubt. For $b_m = 0.3$ this is only $0.27\ldots$ bits and even for $b_m = 0.95$ it is only $3.1\ldots$ bits.

Perhaps the biggest obstacle to using cryptogenography, is to get people to follow a protocol. Part of this problem is that many people might not care to follow a protocol. This issue was addressed in Chapter 6, and it turned out not to be a problem at all: as long as people send out a sufficient amount of randomness, we

[1]That is, in the model where the total number of people is much larger than the number of leakers, and in two of the adaptive models: "b_l-threshold centrally organised" and "b_l- threshold informed choice".

can take a protocol π, and adapt it to the distribution of innocent communication. To do so, we need to know the distribution of the innocent communication used. This should be possible by, for example, using the last digit of time stamps, or by using the randomness that people send out when participating in cryptographic protocols.

Another part of the problem of getting people to follow a protocol, is that some people might be actively against the purpose of the protocol, and want to obstruct it. This problem was addressed in Chapter 4. If b_l denotes the probability that any given person is a leaker and b_c denotes the probability that they are a censor (that is, they are trying to obstruct the protocol), then the censors can prevent any information from being revealed if $b_l + b_c \geq b_m$. However, we also saw that if $b_l + b_c$ is much smaller than b_m, the censors only have a small effect on the number of leakers it takes to leak a bit of information. In the case where b_l and b_c are both small, the main effect of the censors is that they can spread false stories: they can ensure that the observer of the communication can only produce a list of length $1 + \lfloor \frac{b_c}{b_l} \rfloor$ which, unless something unlikely happens, contains the truth x.

Probably the largest part of the problem of getting people to follow a cryptogenography protocol, is to choose a protocol and communicate it to people who want to reveal information, without being stopped or punished by the adversary. One way of doing this would be to prepare the protocol before the adversary is "active". Warrant canaries is an example of a communication method which has to be prepared before an adversary is "activated". Besides providing anonymity to the leakers, cryptogenography has the advantage over warrant canaries that there is a much smaller risk of false positives: while a warrant canary can die (disappear) by mistake, in cryptogenography it is possible to ensure that $G(T)$ returns an empty message with very high probability, unless some people try to send a message. However, there would be many legal questions about such a use of cryptogenography: is it "speech", if you send less than one bit of information? Is it legal to prepare to use cryptogenography to leak information you will not be allowed to leak? Can a company, whose employees have, beyond reasonable doubt, leaked information, be punished for leaking information, if no particular employee, no particular action and no particular message is likely to have been part of the leakage?

In the model in Chapter 3 and Chapter 4 we assumed the leakers would never

be willing to lose reasonable doubt, and as long as they had reasonable doubt, they did not care how suspicious they were. Perhaps it would be more realistic if the leakers were willing to make trade-offs between the amount of information revealed and how suspicious they are. There is no canonical model for such a situation, as different people might prefer different outcomes, but in Chapter 5 we considered a model where the players had to make such trade-offs. We saw that in the case with many leakers, the measure of suspicion was still useful, although in this model we did not manage to get matching upper and lower bounds. We proved a concavity characterization, which is useful for proving upper bounds on the utility in the case of only one leaker.

Finally, in Chapter 7 we considered a model where the adversary has bounded computational power. We showed that this is still not enough for a single leaker to be able to reveal information anonymously, but it can in theory be used for bootstrapping a small or expensive anonymous channel. One of the building blocks used for doing this bootstrapping is indistinguishability obfuscation, so the scheme is currently infeasible in practice. It is still an open problem if you can bootstrap anonymous communication in a computationally cheaper way. Another disadvantage in the scheme is that it requires the leaker to download a large amount of information. Perhaps this could be improved by finding a protocol, where the leaker only needs to get a hash of the transcript.

Throughout the thesis we have considered the problem of how to get anonymity in the presence of an extremely strong adversary. In particular, we have only considered models where leakers cannot get any meaningful help from non-leakers. While we do have positive results, we have also shown very strong impossibility results. Most anonymity research assumes that there exist helpers and at least one of them can be trusted. Our impossibility results justify this assumption, by showing that only very little can be done without it.

References

[1] Anonymizer. anonymizer. `https://www.anonymizer.com/`. 27

[2] Michael Backes, Aniket Kate, Praveen Manoharan, Sebastian Meiser, and Esmaeil Mohammadi. AnoA: A framework for analyzing anonymous communication protocols. In *Computer Security Foundations Symposium (CSF), 2013 IEEE 26th*, pages 163–178. IEEE, 2013. 25

[3] Rajiv Bagai, Huabo Lu, Rong Li, and Bin Tang. An accurate system-wide anonymity metric for probabilistic attacks. In *Proceedings of the 11th Privacy Enhancing Technologies Symposium (PETS 2011)*, July 2011. 25

[4] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 1–10. ACM, 1988. 29

[5] Marvin Künnemann Benjamin Doerr. Improved protocols and hardness results for the two-player cryptogenography problem. *arXiv preprint arXiv:1603.06113*, 2016. 18, 136, 141, 155, 158

[6] Daniel J Bernstein, Johannes Buchmann, and Erik Dahmen. *Post-quantum cryptography*. Springer Science & Business Media, 2009. 14

[7] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 97–106, 2011. 202

[8] Mark Braverman, Ankit Garg, Denis Pankratov, and Omri Weinstein. From information to exact communication. In *Proc. 45th Annual ACM Symposium on the Theory of Computing*, 2013. 143, 154

[9] Joshua Brody, Sune Jakobsen, Dominik Scheder, and Peter Winkler. Cryptogenography. In *ITCS*, 2014. 2, 14, 136, 155

[10] Christian Cachin. An information-theoretic model for steganography. In *Information Hiding, Second International Workshop, Portland, Oregon, USA, April 14-17, 1998, Proceedings*, volume 1525 of *Lecture Notes in Computer Science*, pages 306–318. Springer, 1998. 31

[11] Rein Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable encryption. In *Advances in Cryptology?CRYPTO'97*, pages 90–104. Springer, 1997. 203

[12] Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Prakash Panangaden. Probability of error in information-hiding protocols. In *20th IEEE Computer Security Foundations Symposium (CSF'07)*, pages 341–354. IEEE, 2007. 25, 52

[13] David Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2), February 1981. 28

[14] David Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of Cryptology*, 1:65–75, 1988. 24, 29

[15] Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. Private information retrieval. *Journal of the ACM (JACM)*, 45(6):965–981, 1998. 30

[16] Sebastian Claußand Stefan Schiffner. Structuring anonymity metrics. In *Proceedings of the Second ACM Workshop on Digital Identity Management*, DIM '06, pages 55–62, New York, NY, USA, 2006. ACM. 25

[17] Henry Corrigan-Gibbs, Dan Boneh, and David Mazières. Riposte: An anonymous messaging system handling millions of users. In *Security and Privacy (SP), 2015 IEEE Symposium on*, pages 321–338. IEEE, 2015. 30

[18] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, New York, NY, 1991. 19, 109, 110

[19] George Danezis and Claudia Diaz. A survey of anonymous communication channels. Technical Report MSR-TR-2008-35, Microsoft Research, January 2008. 27, 30

[20] George Danezis, Roger Dingledine, and Nick Mathewson. Mixminion: Design of a type iii anonymous remailer protocol. In *Security and Privacy, 2003. Proceedings. 2003 Symposium on*, pages 2–15. IEEE, 2003. 28

[21] Claudia Diaz, Stefaan Seys, Joris Claessens, and Bart Preneel. Towards measuring anonymity. In *Proceedings of Privacy Enhancing Technologies Workshop (PET 2002)*. Springer-Verlag, LNCS 2482, April 2002. 24, 25

[22] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th USENIX Security Symposium*, August 2004. 29

[23] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014. 25

[24] M. Edman, F. Sivrikaya, and B. Yener. A combinatorial approach to measuring anonymity. *Intelligence and Security Informatics, 2007 IEEE*, pages 356–363, May 2007. 25

[25] Alexandre Evfimievski, Johannes Gehrke, and Ramakrishnan Srikant. Limiting privacy breaches in privacy preserving data mining. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 211–222. ACM, 2003. 32, 96

[26] Lee Ferran. Ex-NSA chief: 'we kill people based on metadata'. *abcnews*, 2014. `http://abcnews.go.com/blogs/headlines/2014/05/ex-nsa-chief-we-kill-people-based-on-metadata/`. 13

[27] Freedom of the Press Foundation. Securedrop. `https://freedom.press/securedrop`. 29

[28] Freehaven. Anonymity bibliography. `http://www.freehaven.net/anonbib/`. 30

[29] Jessica Fridrich. *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge University Press, New York, NY, USA, 1st edition, 2009. 30

[30] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 40–49, 2013. 195, 200

[31] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 169–178, 2009. 202

[32] Benedikt Gierlichs, Carmela Troncoso, Claudia Diaz, Bart Preneel, and Ingrid Verbauwhede. Revisiting A Combinatorial Approach Toward Measuring Anonymity. In *WPES '08: Proceedings of the 7th ACM workshop on Privacy in the electronic society*, volume ACM, pages 111–116, Alexandria,VA,USA, 2008. ACM. 25

[33] Naomi Gilens. The NSA has not been here: Warrant canaries as tools for transparency in the wake of the snowden disclosures. *Harv. J. Law & Tec*, 28:525–593, 2015. 32, 33

[34] David M Goldschlag, Michael G Reed, and Paul F Syverson. Hiding routing information. In *Information Hiding*, pages 137–150. Springer, 1996. 28

[35] Glenn Greenwald. The crux of the NSA story in one phrase: 'collect it all'. *The Guardian*, 2013. `http://www.theguardian.com/commentisfree/2013/jul/15/crux-nsa-collect-it-all`. 13

[36] Glenn Greenwald and Ewen MacAskill. NSA prism program taps in to user data of apple, google and others. *The Guardian*, 7(6):1–43, 2013. 27

[37] Ceki Gülcü and Gene Tsudik. Mixing e-mail with babel. In *Network and Distributed System Security, 1996., Proceedings of the Symposium on*, pages 2–16. IEEE, 1996. 28

[38] Venkatesan Guruswami. Algorithmic results in list decoding. *Foundations and Trends in Theoretical Computer Science*, 2(2):107–195, 2006. 108

[39] Temporary injunction in the anonymous remailer case. `http://web.archive.org/web/19970414065743/http://www.penet.fi/injuncl.html`. 27

[40] Johan Helsingius. Johan helsingius closes his internet remailer. `http://web.archive.org/web/19970414065812/http://www.penet.fi/press-english.html`. 27

[41] Nicholas J. Hopper. *Toward a theory of Steganography*. PhD thesis, Carnegie Mellon University, 2004. 177

[42] Nicholas J. Hopper, John Langford, and Luis von Ahn. Provably secure steganography. In *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*, pages 77–92, 2002. 31

[43] Pavel Hubacek and Daniel Wichs. On the communication complexity of secure function evaluation with long output. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, pages 163–172. ACM, 2015. 202, 204, 212

[44] Russell Impagliazzo and Ryan Williams. Communication complexity with synchronized clocks. In *Computational Complexity (CCC), 2010 IEEE 25th Annual Conference on*, pages 259–269. IEEE, 2010. 214

[45] Luca Invernizzi, Christopher Kruegel, and Giovanni Vigna. Message in a bottle: Sailing past censorship. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*. ACM, 2013. 31

[46] Sune K. Jakobsen. Greatest function satisfying some convexity requirements. mathoverflow.net/questions/81753. (2011). 155

[47] Sune K. Jakobsen. Information theoretical cryptogenography. In *ICALP (1)*, pages 676–688, 2014. 2

[48] Sune K Jakobsen. Information theoretical cryptogenography. *arXiv preprint arXiv:1402.3125*, 2014. 2

[49] Sune K Jakobsen and Claudio Orlandi. How to bootstrap anonymous communication. *arXiv preprint arXiv:1502.05273*, 2015. 3

[50] Sune K Jakobsen and Claudio Orlandi. How to bootstrap anonymous communication. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*, pages 333–344. ACM, 2016. 3, 23, 198

[51] Sune K Jakobsen, Troels B Sørensen, and Vincent Conitzer. Timeability of extensive-form games. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*, pages 191–199. ACM, 2016. 28

[52] Dogan Kesdogan, Jan Egner, and Roland Bschkes. Stop-and-go-mixes providing probabilistic anonymity in an open system. In *In Proceedings of Information Hiding workshop*, pages 83–98. Springer-Verlag, 1998. 23

[53] Venkata Koppula, Allison Bishop Lewko, and Brent Waters. Indistinguishability obfuscation for turing machines with unbounded memory. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 419–428, 2015. 205

[54] Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, New York, NY, USA, 1997. 36

[55] Ladar Levison. Secrets, lies and snowden's email: why I was forced to shut down lavabit. *The Guardian*, 2014. `http://www.theguardian.com/commentisfree/2014/may/20/why-did-lavabit-shut-down-snowden-email`. 27

[56] Richard J. Lipton. Who knew the secret? `https://rjlipton.wordpress.com/2013/12/13/who-knew-the-secret/`, 2013. 14

[57] N. Ma and P. Ishwar. Interactive source coding for function computation in collocated networks. *IEEE Trans. Inf. Theory*, 58(7):4289–4305, 2012. 154

[58] N. Ma and P. Ishwar. The infinite-message limit of two-terminal interactive source coding. *IEEE Trans. Inf. Theory*, 59(7):4071–4094, 2013. 154

[59] Hooman Mohajeri Moghaddam, Baiyu Li, Mohammad Derakhshani, and Ian Goldberg. Skypemorph: Protocol obfuscation for tor bridges. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 97–108. ACM, 2012. 31

[60] Ulf Möller, Lance Cottrell, Peter Palfrader, and Len Sassaman. Mixmaster protocol?version 2. *Draft, July*, 154, 2003. 28

[61] Rajeev Motwani and Prabhakar Raghavan. *Randomized algorithms*. Chapman & Hall/CRC, 2010. 220

[62] Andreas Pfitzmann and Marit Hansen. A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management, 2010. 24

[63] The Tor project. Tor metrics – relays with exit, fast, guard, stable, and hsdir flags. `https://metrics.torproject.org/relayflags.html`. 15

[64] Niels Provos and Peter Honeyman. Hide and seek: An introduction to steganography. *Security & Privacy, IEEE*, 1(3):32–44, 2003. 30

[65] Michael K Reiter and Aviel D Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security (TISSEC)*, 1(1):66–92, 1998. 27, 28

[66] Andrei Serjantov and George Danezis. Towards an information theoretic metric for anonymity. In *Proceedings of Privacy Enhancing Technologies Workshop (PET 2002)*. Springer-Verlag, LNCS 2482, April 2002. 24, 25

[67] Claude E. Shannon. A mathematical theory of communication. *Bell system technical journal*, 27, 1948. 22, 57

[68] Paul Syverson. Why I'm not an entropist. In *Security Protocols XVII*, pages 213–230. Springer, 2009. 28, 29

[69] Gergely Tóth, Zoltán Hornák, and Ferenc Vajda. Measuring anonymity revisited. In *Proceedings of the Ninth Nordic Workshop on Secure IT Systems*, pages 85–90, November 2004. 24, 25, 26

[70] Jelle Van Den Hooff, David Lazar, Matei Zaharia, and Nickolai Zeldovich. Vuvuzela: Scalable private messaging resistant to traffic analysis. In *Proceedings of the 25th Symposium on Operating Systems Principles*, pages 137–152. ACM, 2015. 30

[71] Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965. 31, 40

[72] Canary Watch. Canary watch. `https://www.canarywatch.org/`. 32

[73] Nicholas Watt, Rowena Mason, and Ian Traynor. David Cameron pledges anti-terror law for internet after paris attacks. *The Guardian*, 2015. `http://www.theguardian.com/uk-news/2015/jan/12/david-cameron-pledges-anti-terror-law-internet-paris-attacks-nick-clegg`. 13

[74] Zachary Weinberg, Jeffrey Wang, Vinod Yegneswaran, Linda Briesemeister, Steven Cheung, Frank Wang, and Dan Boneh. Stegotorus: a camouflage proxy for the tor anonymity system. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 109–120. ACM, 2012. 31

[75] Rebecca Wexler. Warrant canaries and disclosure by design: The real threat to national security letter gag orders. *Yale LJF*, 124:158–349, 2014. 32, 33

[76] Ian H. Witten, Radford M. Neal, and John G. Cleary. Arithmetic coding for data compression. *Commun. ACM*, 30(6):520–540, June 1987. 180

[77] David Isaac Wolinsky, Henry Corrigan-Gibbs, Bryan Ford, and Aaron Johnson. Dissent in numbers: Making strong anonymity scale. In *Presented as part of the 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI 12)*, pages 179–182, 2012. 29

[78] Andrew Chi-Chih Yao. Some complexity questions related to distributive computing(preliminary report). In *Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing*, STOC '79, pages 209–213, New York, NY, USA, 1979. ACM. 36

[79] Ye Zhu and Riccardo Bettati. Anonymity vs. information leakage in anonymity systems. In *ICDCS'05*, pages 514–524, 2005. 25