Distribution-Matching Encryption

Ari Juels
Cornell Tech (Jacobs)
juels@cornell.edu

Thomas Ristenpart
University of Wisconsin
rist@cs.wisc.edu

Thomas Shrimpton
Portland State University
teshrim@pdx.edu

September 10, 2014
Version 1.2

Abstract

We detail a new cryptographic primitive, distributional encryption, that generalizes a number of previous ones, including format-transforming encryption, format-preserving encryption, stegonagraphy, and honey encryption. In so doing, we provide a framework by which one can compare these various primitives, and which surfaces new encryption mechanisms of potentially broad utility. In particular, we give the first password-based stegonagraphy primitive, and show how to use this to build censorship-resistant password-based authenticated key exchange.

1 Introduction

≪**TomR 1.1:** The below is taken from an email I sent on Jan 20, 2014.≫

Started trying to firm up my thoughts on this paper on generalizing HE/ stego/ FPE/ FTE to what I have been calling distributional encryption for lack of better term.

The basic idea is we have a primitive that is parametrized by two distributions, S_d (for domain) and S_r (for range). (We will want to extend to families of distributions for range and domain, but for now staying simple.) Encryption takes as input a key and message, and is randomized as usual. Decryption takes a key and ciphertext and is probably just deterministic.

To each of S_d and S_r we denote message and ciphertext spaces which are the support of the distributions. Correctness requires that for any $M \in \text{Supp}(S_d)$:

$$\Pr[\, D_K(E_K(M)) = M \,] > 1 - \epsilon$$

We say it is perfectly correct when $\epsilon = 0$.

For security, I suggest a framework of notions, all involving a basic concept of indistinguishability from the target distribution. So the basic deterministic CPA notion gives an adversary an oracle that is either implemented by E_K for a random, secret K, or by a sampler for S_r. We require that the adversary repeat no queries to its oracles (unachievable for deterministic algorithms).

A deterministic CCA variant adds a decryption oracle that either implements decryption in first world or sampling from S_d in ideal world. We will have to extend types of disallowed queries appropriately.

We can also extend definitions to randomized schemes.

This general formulation covers FTE, FPE, HE, Stego, (tweakable) PRPs, and other combinations that haven't been looked at before (such as HE with target ciphertext formats, which could be really useful for adding HE security to FPE-style applications).

Then we have an immediate question of channel capacity, which relates the requirement for correctness, efficiency, and security. It's clear to see there is some trade-off, since a target distribution with high min-entropy is going to be able to encrypt fewer bits (assuming encryption is efficient) compared to flat distributions. (This is orthogonal from the size of support of S_r.)

I suspect the stego literature may have something to say about this, though it is somewhat unclear in my memory as to whether they look at multi-bit encryption efficiency.

We also can look at further limitations along another dimension (this idea is partly due to Yevgeniy from a discussion with him. He thinks this is a cool direction, btw.) Namely, showing distributions for which constructions are

impossible even if the information capacity is sufficient. We could do this in general or via the DTE-encrypt-XXX construction where we compose a DTE with encryption and follow it by an distribution-aware embedding (XXX is placeholder until we have a name for this step). This would be similar, then, to the negative results about rank-encipher-unrank constructions for FPE from the 2009 paper.

Yev and I were discussing for example if you make the target distribution S_r the range of an PRNG, then decryption would seem to require inverting the PRNG which should be infeasible. This doesn't quite work, though, since obviously S_r is computationally close to uniform bit strings, so one could just pad out the intermediate encryption ciphertext with random bits. But something else may work, and this would be an interesting result.

Ok so also we need constructions. I'd like to understand what is a proper formalization of the XXX mentioned above, namely what enables a DTE-Encrypt-XXX type construction. Seemingly it's something like a randomness-recovering sampling algorithm. Namely for a given S_r, we need an efficient algorithm $D: \{0,1\}^n \rightarrow \text{Supp}(S_r)$ to (approximately) sample S_r, meaning if you feed D random coins r, then there exists another efficient algorithm D^{-1} such that

$$\Pr\left[D^{-1}(D(R)) = R \right] > 1 - \epsilon$$

By allowing error, this gives us some flexibility, for example we need not worry about being able to decrypt (asymptotically speaking for the moment) any constant number of subsets S of $\text{Supp}(S_r)$ for which the probability of that subset being hit when sampling from S_r is negligible.

Here's a potential example for the randomized case, based on discussions with Tom S from a while ago. Consider an S_r which is nearly flat, meaning that the difference between the area under $\text{Unif}(\text{Supp}(S_r))$ over the area under S_r is bounded by a small value. Then one can use rejection sampling to get an algorithm that works pretty well. First encrypt the intermediate plaintext (output by a DTE) to get a candidate ciphertext value C in $\text{Supp}(S_r)$ (perhaps using some kind of unranking scheme if needed to get into $\text{Supp}(S_r)$). Then choose a random value between 0 and 1, and check if that value is less than or equal to the probability of C under S_r. If it is not below the curve, then start over, encrypting the intermediate plaintext again with fresh coins, etc. Decryption works in the natural way.

This is only expected-time efficient (remember assumption above regarding ratio of areas), but one can also give a worst-case time just by absolutely bounding the number of iterations and outputing whatever was derived in the last step. Then the number of iterations will arise in the security bound.

Interestingly this is not randomness-recovering, since we never (need to) recover all the coins used by D. But it seems to work.

≪**TomR 1.2:** Some stuff from Tom S in response to above email:≫

– You mention that encryption for this new primitive is randomized, but then focus on deterministic notions. There's no problem with that, but I did want to add that if encryption is randomized, our general formulation also covers traditional IND$-secure encryption. Nonce-based versions should be straightforward to formalize, too, and may be useful in practice.

– Actually, thinking about IND$-secure encryption (and your PRG example, below) gave me an idea for an application. It might actually be interesting to have ciphertexts that appear to have been produced by a particular primitive, in the hopes of sending cryptanalytic efforts down the wrong path.

– I remind you of another idea we kicked around a while ago, for doing ciphertext-distribution-matching encryption via MCMC-style techniques. Using Metropolis-Hastings, a counter, and a PRF, you can perform a random walk over the ciphertext space that results in essentially any efficiently computable distribution. To decrypt, you'd need to know the sequence of counter values (and the key, of course) in order to recover the "coins" used for the random walk. This probably means you need stateful decryption, which is outside of the syntax you give, and an agreement on how many steps any particular encryption will take. (Sending counter values in the clear will break the distribution-matching property.) None of this is going to be efficient, but I think it's interesting.

– I don't understand your PRG example, below. Why should it matter if S_r covers the range of a PRG? If I take an IND$-secure encryption scheme that has only N-bit outputs, this will give the same distribution of outputs as does a secure PRG with N-bit outputs; both computationally indistinguishable from uniform. But I guess you mean something like, you can't *define* Pr(C) under S_r by sampling an X uniformly, and computing $C = f(X)$, where f is one-way?

– I think the Hopper et al. paper on stego formalisms does talk about stego-crypting multiple bits at once, because

their most secure scheme (meeting something like the CPA notion you describe) only sends a single bit at a time, and has a huge ctxt expansion. The multi-bit scheme meets a less stringent notion, and you can still only send log(security_params) bits, if I recall properly.

– You mention the rejection-sampling idea for doing encryption. That induces a one-sided performance loss, namely that encryption may have to resample many, many times. But we also kicked around some ideas for shifting some of the loss to decryption. In the case the sample from S_r does not allow you to uniquely recover the underlying randomness, but nearly so, then on the decryption side you can try multiple decryptions, and hope that only one of those "makes sense". (For example, if encryption is authenticated, but the ciphertext doesn't give you all of the coins, it ought to be that only a very small number of the possible decryptions will return something other than $\perp$.) This might not be an efficient tradeoff.

– It seems like the DTE-then-Encrypt paradigm needs lots of randomness, some for DTE and some for Encrypt (plus a key), and adding on ciphertext-distribution matching seems likely to require even more. Reducing the number of random bits needed might be an interesting direction, once we have some concrete constructions sorted out.

2 Basic Notions for Distribution-Matching Encryption

Distributions. A probability distribution over a set S is a map $p\colon S \to [0,1]$ that satisfies the requirement that $\sum_{s\in S} p(s) = 1$. The support of p is the set of all elements in S with non-zero probability, i.e., $\mathrm{Supp}(p) = \{s \in S \,|\, p(s) > 0\}$. Sampling from a set S according to a distribution p using an appropriate number of fresh coins is denoted by $s \leftarrow_p S$.

DME schemes. Like a conventional symmetric encryption scheme, a DME scheme $\mathsf{DME} = (\mathsf{enc}, \mathsf{dec})$ consists of two algorithms. Encryption enc takes input a key $K \in \{0,1\}^*$ and a message $M \in \{0,1\}^*$, and outputs a ciphertext $C \in \{0,1\}^*$ or a special error symbol $\perp$. Decryption dec takes input a key and a ciphertext in $\{0,1\}^*$ and outputs a message in $\mathcal{M}$ or $\perp$.

We fix three distinguished sets $\mathcal{K}, \mathcal{M}, \mathcal{C} \subset \{0,1\}^*$ called the key, message, and ciphertext space, respectively. We require that $\mathsf{enc}(K, M)$ outputs $\perp$ when $K \notin \mathcal{K}$ or $M \notin \mathcal{M}$. Similarly, $\mathsf{dec}(K, C)$ must output $\perp$ when either the key or ciphertext are not in the appropriate sets. Instead of having a key generation algorithm, we assume keys are drawn from $\mathcal{K}$ according to some distribution p_k. For conventional secret keys, for example, one sets $\mathcal{K} = \{0,1\}^k$ for some k and $p_k(K) = 1/2^k$ for all $K \in \mathcal{K}$.

We allow both encryption and decryption to be randomized or stateful.≪**TomR 2.1:** Need to discuss stateful in fuller detail , for example correctness is already a bit messed up below≫

We say that a DME scheme is *ϵ-everywhere correct* if for all messages M and all keys K

$$\Pr\left[\,\mathsf{dec}(K, \mathsf{enc}(K, M) = M\,\right] \geq 1 - \epsilon$$

where the probability is over the coins used by enc. The notion of "everywhere" because we quantify over all messages in the message space. A scheme is *perfectly everywhere correct* if it is ϵ-everywhere correct for $\epsilon = 0$.

So far, nothing about DME is different from conventional SE. The deviation begins with the fact that DME schemes will be designed to with target a distribution over messages and one over ciphertexts. The message distribution, denoted p_m, specifies the distribution from which the DME scheme designer expects plaintexts to be drawn. This is inspired directly from honey encryption (HE), and in terms of security the goal is that a ciphertext decrypted with the wrong key should result in a plaintext distributed closely to p_m. The ciphertext distribution, denoted p_c, is a target for what emitted ciphertexts should "look like". This is inspired by stegonagraphy, where p_c plays the role of the distribution of cover traffic or perhaps an innoucous-looking file.

As an example, consider the simple hash based encryption scheme in which one encrypts a message $M \in \mathcal{M} = \{0,1\}^\ell$ for some ℓ via $\mathsf{enc}(K, M) = (\mathsf{sa}, H(\mathsf{sa} \oplus K))$ for $\mathsf{sa} \leftarrow\!\!\$\ \{0,1\}^s$ and where $H\colon \{0,1\}^* \to \{0,1\}^\ell$ is a cryptographic hash function. Then, informally speaking, ciphertexts are distributed uniformly over $\{0,1\}^{s+\ell}$. Likewise, decrypting a fixed ciphertext sa, C using the incorrect key will give back a plaintext distributed uniformly over $\{0,1\}^\ell$. We will make these claims formal in Section **??**.

More challenging examples will be when we desire p_m and/or p_c to be non-uniform or uniform over strange sets.

Primitive	Deterministic	p_m	p_c	Security goal
FTE [**?**, **?**, **?**]	Sometimes	uniform over set	uniform over set	AE security
HE [**?**]	No	arbitrary dist	uniform bit strings	MR, IND$
Stegonography [**?**]	No	–	arbitrary dist	Real-or-dist
PW-based stego	No	arbitrary dist	arbitrary dist	Real-or-dist

≪**TomR 2.3:** Security field is pretty nebulous. Perhaps recast in terms of the notions we present?≫

Figure 1: Primitives captured by the DME formulation, along with the correctness and security goals sought. The bottom several primitives are novel, falling out of our DME framework. A dash '–' indicates that no explicit distributions are considered.

We will see some examples below of some existing encryption mechanisms that can be cast as DME schemes.

Prior schemes falling under the DME framework. The syntax and semantics of the DME primitive, being equivalent to that of existing SE, means that DME broadly encompasses a numer of previous schemes. As per the example above, many conventional SE schemes can be viewed as DME schemes for which the distributions p_m and p_c targeted are uniform over appropriate-length bit strings. More involved examples include stegosystems [**?**], honey encryption (HE) [**?**], and format-transforming encryption (FTE) [**?**]:

As first formalized by [**?**], a stegosystem encodes a message using a secret key in order to produce a covertext distributed indistinguishably from some target distribution over ciphertexts (what they call the channel distribution). We give more details about their formalization, and how it compares to ours, in Appendix **??**. Suffice to say, we can think of any stegosystem as a DME scheme with p_m uniform over bit strings, and where p_c models the covertext distribution. As we will argue later in the paper, a key benefit of viewing stegosystems as DME schemes is that we can provide stegonographic security even when keys are low entropy (e.g., passwords). This is not possible for existing schemes, and will allow us to build (among other primitives) stegonagraphic password-based authenticated encryption (steg-PAKE).

FTE [**?**] is a form of symmetric encryption in which ciphertexts must abide by a prescribed format, which technically means they belong to a particular set. Luchaup et al. [**?**] generalize FTE to also consider only allowing plaintexts from a certain set, making it a strict generalization of the earlier primitive format-preserving encryption (FPE) [**?**]. We can view FTE as a DME scheme in which p_m and p_c are uniform over specified sets $\mathcal{M}$ and $\mathcal{C}$. For example, in their application in anti-censorship, Dyer et al. set $\mathcal{C}$ to be the language accepted by a regular expression. FTE was originally envisioned to be used with high-entropy secret keys, and we will extend to allow use with low-entropy keys as well in some situations.

An HE scheme targets p_m to be some distribution over plaintexts and p_c to be uniform over appropriate-length bit strings. An example from [**?**] is an HE scheme for uniformly distributed prime numbers. This ensures that when encrypting a uniform prime number with a low-entropy password, an attacker that obtains a ciphertext cannot distinguish between decrypting ciphertext with the correct password and others. This can prevent offline-brute force attacks. To do so, Juels and Ristenpart suggest building HE schemes by composing a so-called distribution-transforming encoder (DTE) with one of several conventional SE encryption algorithm. We will also use DTEs, which "transform" a value from one distribution to a value distributed according to another. A detailed explanation of them appears in Section **??**.

At the simplest level, our main DME constructions will combine the techniques underlying HE with those underlying FTE and/or stegosystems in a relatively direct way. The DME formalization, however, provides a useful framework that places previous primitives into a unified design space for symmetric encryption. By seeing gaps in the DME design space not filled by prior primitives, we will obtain new encryption primitives that fill these gaps.

≪**TomR 2.2:** Can we add an example of a scheme that isn't from DTE-Encrypt-DTE? Perhaps go with the idea, I think originally may have been suggested by Yevgeniy, of giving distributions p_m and/or p_c for which one provably cannot provide DTEs or, even, DMEs?≫

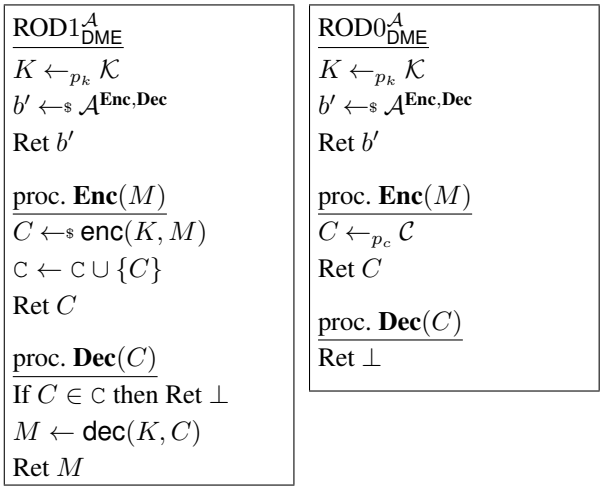

Figure 2: Games for real-or-distribution (ROD) security.

3 Formal DME Security Notions

We give two classes of security targets. In the first class, we consider security when one has high-entropy keys, i.e., ones drawn uniformly from a sufficiently large set. Here we can achieve strong security goals akin to that targeted by Hopper et al. [**?**] for stegosystems, though we will extend these goals to provide authenticity properties as well. In the second class, we consider security when one has low-entropy keys such as passwords. Here we will explore quite non-standard security goals that, as in HE, take advantage of messages being drawn from a known (to the DME scheme designer as well as attackers) message distribution p_m.

Real-or-distribution security for high-entropy keys. We first give a simple embellishment of the classic security notion for symmetric encryption in which one asks that an attacker cannot distinguish between encryption under a key and a random bit string. Instead we ask that no attacker can distinguish between encryptions and a ciphertext drawn according to p_c. Advantage of an adversary $\mathcal{A}$ against a DME scheme $\mathsf{DME} = (\mathsf{kg}, \mathsf{enc}, \mathsf{dec})$ relative to a ciphertext distribution p_c is defined as

$$\mathbf{Adv}^{\mathrm{rod}}_{\mathsf{DME},p_c}(\mathcal{A}) = \left|\Pr\left[\,\mathrm{ROD1}^{\mathcal{A}}_{\mathsf{DME}} \Rightarrow 1\,\right] - \Pr\left[\,\mathrm{ROD0}^{\mathcal{A}}_{\mathsf{DME},p_c} \Rightarrow 1\,\right]\right| .$$

ROD security for chosen-plaintext-only attacks is defined by considering the above but for adversaries that make no decryption queries. We note that the security as specified will require randomized or stateful encryption (as we allow repeat queries to the encryption oracle) and ciphertext stretch (since we require the attacker not to be able to forge ciphertexts). This means it is unsuitable for FPE or FTE schemes, but one can give appropriate definitions for such settings should one desire.

Following a concrete security approach, we will give explicit statements that measure the resources used by $\mathcal{A}$ in terms of its worst-case running time (in some RAM model of computation) and the number of queries q_e to enc and q_d to dec.

It is easy to show ROD this security notion requires high entropy keys, as the attacker can obtain known plaintext-ciphertext outputs and, therefore, mount brute-force attacks to recover the key (in the low key entropy case). We also note that in these settings use of p_m is, essentially, superflous to security — use of this in a scheme will arise next. ≪**TomR 3.1:** We need to compare this notion to the Hopper et al. one. Intuitively we should be strengthening their thing, but for the setting of symmetric encryption.≫≪**TomS 3.2:** Similar to [HvAL] notion. No decryption oracle there, but the adversary does get to control the "history" that parameterizes encryption/channel sampling.≫

Real-or-distribution security for low-entropy keys. We now turn to a variant of the ROD notion for low-entropy keys. As just discussed, in this setting we cannot give the attacker known plaintext/ciphertext pairs. Rather, we first turn to a non-interactive setting.

Advantage of an adversary $\mathcal{A}$ against a DME scheme $\mathsf{DME} = (\mathsf{kg}, \mathsf{enc}, \mathsf{dec})$ relative to a distributions p_k, p_m,

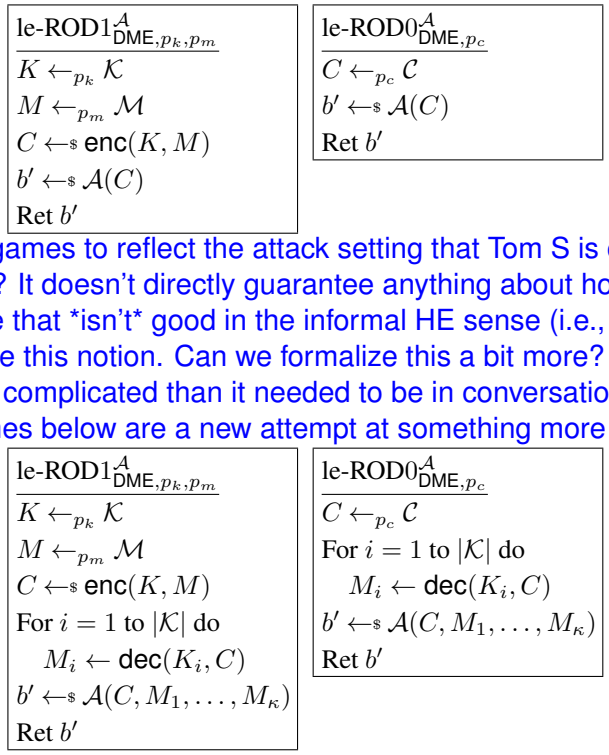

Figure 3: Games for low-entropy real-or-distribution (le-ROD) security.

and p_c is defined as

$$\mathbf{Adv}^{\text{le-rod}}_{\mathsf{DME},p_c}(\mathcal{A}) = \left|\Pr\left[\,\text{le-ROD1}^{\mathcal{A}}_{\mathsf{DME}} \Rightarrow 1\,\right] - \Pr\left[\,\text{le-ROD0}^{\mathcal{A}}_{\mathsf{DME},p_c} \Rightarrow 1\,\right]\right| .$$

Known-key and low-entropy key indistinguishability. Now we turn to the setting where we have low-entropy keys, such as passwords. In this context we have to deal with the fact that attackers can perform work sufficient to mount brute-force attacks. This rules out meeting conventional security goals such as semantic security, and by implication ROD security. It also rules out any form of authentication. ≪**TomR 3.5:** Do we want to formalize this somewhat obvious claim? Note that Qiang has been working on stuff related to this.≫ The goal is to obtain meaningful security even when using encryption with low-entropy keys. The approach is to take advantage of the fact that in many situations a scheme designer knows the distribution from which plaintexts will be drawn. We can tailor our schemes to these distributions, striving to ensure that ciphertexts, even when one can decrypt with a known key, are indistinguishable from values drawn according to the target ciphertext distribution p_c.

We start with the strongest notion, that we call known-key indistinguishability (KK-IND). The security games are shown in Figure 4, and are parameterized implicitly by a number $q \geq 1$. In the first game KK-ROD1 the adversary is given encryptions of q messages independently drawn according to p_m under a single key drawn according to a distribution p_k. In the second game KK-ROD0 the adversary is given the decryptions of q ciphertexts drawn independently according to p_c under a key distributed as per p_k. In the games we have extended our notation so that running $\mathsf{enc}(K, \vec{M})$ means running with independent coins $\mathsf{enc}(K, \vec{M}_1)$, $\mathsf{enc}(K, \vec{M}_2)$, and so on up to $\mathsf{enc}(K, \vec{M}_q)$. Likewise for dec. We define advantage of an adversary $\mathcal{A}$ against a DME scheme DME relative to distributions p_k, p_m, p_c by

$$\mathbf{Adv}^{\text{kk-rod}}_{\mathsf{DME},p_k,p_m,p_c}(\mathcal{A}) = \left|\Pr\left[\,\text{KK-ROD1}^{\mathcal{A}}_{\mathsf{DME},p_k,p_m} \Rightarrow 1\,\right] - \Pr\left[\,\text{KK-ROD0}^{\mathcal{A}}_{\mathsf{DME},p_k,p_c} \Rightarrow 1\,\right]\right| .$$

When $\mathcal{A}$ is computationally unbounded, this becomes equivalent to measuring the statistical distance between the triples $(K, \vec{M}, \vec{C})$ as distributed in each game. We say that a scheme DME is perfectly (p_k, p_m, p_c)-KK-ROD-secure if $\mathbf{Adv}^{\text{kk-rod}}_{\mathsf{DME},p_k,p_m,p_c}(\mathcal{A}) = 0$ for any $\mathcal{A}$.

The KK-ROD notion appears strange, and at first glance may seem unachievable. After all, we give the adversary the secret key, and one might expect then that all security goals are lost. But we will in fact show DME schemes

KK-ROD1$^{\mathcal{A}}_{\mathsf{DME},p_k,p_m}$	KK-ROD0$^{\mathcal{A}}_{\mathsf{DME},p_k,p_c}$
$K \leftarrow_{p_k} \mathcal{K}$	$K \leftarrow_{p_k} \mathcal{K}$
$\vec{M} \leftarrow_{p_m} \mathcal{M}^q$	$\vec{C} \leftarrow_{p_c} \mathcal{C}^q$
$\vec{C} \leftarrow\!\!\$\ \mathsf{enc}(K, \vec{M})$	$\vec{M} \leftarrow \mathsf{dec}(K, \vec{C})$
$b' \leftarrow\!\!\$\ \mathcal{A}(K, \vec{M}, \vec{C})$	$b' \leftarrow\!\!\$\ \mathcal{A}(K, \vec{M}, \vec{C})$
Ret b'	Ret b'

Figure 4: Games for known-key indistinguishability (KK-ROD) security.

can achieve this goal, and thereby provide non-cryptographic security properties that will prove useful in a variety of applications.

To see how this can be, we exercise the notion with the following positive result about the simple hash-based DME scheme described in Section 2. The theorem below shows that the scheme achieves KK-ROD security when considering p_m and p_c uniform over ℓ-bit strings.

Theorem 1 *Let* DME *be the hash-based DME scheme from Section 2, model* $H : \{0,1\}^* \to \{0,1\}^\ell$ *be a function, let* p_k *be arbitrary, and let* p_m *and* p_c *be uniform over* $\{0,1\}^\ell$*. Then* DME *is perfectly* (p_k, p_m, p_c)-KK-ROD *secure.*

Proof: The proof holds by the simple observation that xor is a permutation, meaning that $H(\mathsf{sa} \,\|\, K) \oplus M$ is uniformly distributed when M is, and similarly with $H(\mathsf{sa} \,\|\, K) \oplus C$. Thus $\Pr[(K, \vec{M}, \vec{C})] = \Pr[(K, \vec{M}', \vec{C}')]$ where the triples represent the event that the indicate values take on a fixed value in KK-ROD1 and KK-ROD0 respectively. ≪**TomR 3.6:** We should probably come upw ith a clean way of working directly with the random variables indicated by the games, since most of our arguments are likely going to reason directly about these distributions≫ ∎

≪**TomR 3.7:** We need more than KK-ROD for steg-PAKE. Well, it may be. Intuitively if you can modify the behavior of the server by flipping a bit of an encrypted key exchange value, then this might leak information helpful in speeding-up password recovery. Have to think this through.≫

≪**TomR 3.8:** Is there any way to get HE-style security plus authenticity? Functionality wise one might suspect so: instead of returning $\perp$... no that doesn't make sense because it would mean an attacker could force acceptance of a randomly chosen message from the message distribution.≫

4 le-ROD security of the [HvAL] `One-Bit` Scheme

Here we consider the `One-bit` scheme of Hopper et al. [**?**], and show that it is somewhat fragile with respect to our XXX-security notion.

Theorem 2 *(Informal.)*Assume that $\forall i \in [q]$ and h_{i-1}, the distributions $\mathcal{C}_{h_{i-1}}$ have min-entropy of at least one. When $H\colon \{0,1\}^* \to \{0,1\}$ is a random oracle, the `One-Bit` scheme $(\mathsf{enc}^H, \mathsf{dec}^H)$ in Figure 5 is not le-ROD-secure. In particular, setting $\delta = 1/8$, adversary $\mathcal{A}_1$ achieves advantage at least

$$1 - \left(e^{-\frac{\ell}{288}} p_m(0) + e^{-\frac{\ell}{320}} p_m(1) + 2e^{-\frac{\ell}{128}}\right) .$$

When $\ell \geq 256$, then, the advantage is at least $1 - 0.412 p_m(0) - 0.450 p_m(1) - 0.271$.

≪**TomS 4.1:** How does this value of ℓ compare to what you need for good correctness? Seems likely to be larger that what you need for the same level of correctness, since the distributions in the correctness setting are farther apart.≫

Proof: (*Still needs polishing.*) We begin by considering the case that $b = 0$, so that $c_1 c_2 \cdots c_\ell$ resulted from independent samples from the distribution determined by $\mathcal{C}_h$, where h takes on whatever is its initial value (and can be ignored for the attack). Let $X_i = H(K \,\|\, \langle i \rangle \,\|\, c_i)$, so that $\Pr[\, X_i = 1 \,] = \Pr[\, X_i = 0 \,] = 1/2$ for all $i \in [\ell]$, and $t = \sum_{i=1}^{\ell} X_i$

$\underline{\mathsf{enc}_K^H(m;(h,N))}$
for $i = 1$ to ℓ
  $s_i \leftarrow\!\!\$\ \mathcal{C}_h$
  $s_i' \leftarrow\!\!\$\ \mathcal{C}_h$
  if $H(K \,\|\, \langle N+i\rangle \,\|\, s_i) = m$ then $c_i \leftarrow s_i$
  else $c_i \leftarrow s_i'$
  $h \leftarrow h \,\|\, c_i$
$N \leftarrow N + \ell$
return $((h,N), c_1c_2\cdots c_\ell)$

$\underline{\mathsf{dec}_K^H(c_1c_2\cdots c_\ell;(h,N))}$
if $\sum_{i=1}^{\ell} H(K \,\|\, \langle N+i\rangle \,\|\, c_i) > \ell/2$ then $m \leftarrow 1$
else $m \leftarrow 0$
$N \leftarrow N + \ell$
return $((h,N), m)$

$\underline{\text{Adversary } \mathcal{A}_1^H(c_1c_2\cdots c_\ell, K)}$
$t \leftarrow \sum_{i=0}^{\ell} H(K \,\|\, \langle i\rangle \,\|\, c_i)$
if $\left|t - \frac{\ell}{2}\right| \geq \delta\frac{\ell}{2}$ then return 1
return 0

$\underline{\text{Adversary } \mathcal{A}_2^H(c_1c_2\cdots c_\ell)}$
for $K \in \mathcal{K}$
  if $\mathcal{A}_1(c_1c_2\cdots c_\ell, K) \Rightarrow 1$ then return 1
return 0

Figure 5: Hopper et al. `One-Bit` Scheme with the PRF $F_K(x) = H(K \,\|\, x)$, and the attacks. [Note: can probably extend this to arbitrary PRFs by changing the threshold for returning 1 in $\mathcal{A}_2$ appropriately... since if any non-negligible fraction α of keys result in "non-random" behavior, you won't have a good PRF to start with.] Adversary $\mathcal{A}_1$ is for known-key settings, while $\mathcal{A}_2$ is for (low-entropy) unknown-key settings. For simplicity, we assume that $N = 0$ initially, and that ℓ is even. The parameter δ is set to $1/8$ in the theorem statements. Alternative $\mathcal{A}_2$ is to mount a key-recovery attack, e.g. finding the key that maximizes the distance between t and $\ell/2$.

is a Binomial random varible with $\mu = \mathbb{E}[t] = \ell/2$. In this case, a standard Chernoff bound [[MU], Corrolaries 4.9 and 4.10] gives, for $0 < \delta < 1$

$$\Pr[\,\mathcal{A}_1 \Rightarrow 1 \,|\, b = 0\,] = \Pr\left[\left|t - \frac{\ell}{2}\right| \geq \delta\frac{\ell}{2}\right] \leq 2e^{-\frac{\ell\delta^2}{2}}$$

We will fix a value for δ in a moment.

Now consider the case that $b = 1$, with message $m = 0$, so that $c_1c_2\cdots c_\ell$ are the result of $\mathsf{enc}_K^H(0)$. Recall that, at the time that c_i was assigned, it was to one of two independent samples s_i, s_i' from the channel distribution. In particular, it was assigned the value of s iff $H(K \,\|\, \langle i\rangle \,\|\, s) = 0$. Thus, letting $X_i = H(K \,\|\, \langle i\rangle \,\|\, c_i)$, we have

$$\begin{aligned}
\Pr[\,X_i = 0\,] &= \Pr[\,X_i = 0 \cap c_i = s_i\,] + \Pr\left[\,X_i = 0 \cap c_i = s_i' \cap (s_i = s_i')\,\right] + \Pr\left[\,X_i = 0 \cap c_i = s_i' \cap (s_i \neq s_i')\,\right] \\
&= \Pr[\,c_i = s_i\,] + \Pr\left[\,X_i = 0 \cap c_i = s_i' \cap (s_i \neq s_i')\,\right] \\
&= \Pr[\,H(K \,\|\, \langle i\rangle \,\|\, s) = 0\,] + \Pr\left[\,X_i = 0 \cap c_i = s_i' \,|\, (s_i \neq s_i')\,\right]\Pr\left[\,s_i \neq s_i'\,\right] \\
&= \frac{1}{2} + \Pr\left[\,H(K \,\|\, \langle i\rangle \,\|\, s_i) = 1 \cap H(K \,\|\, \langle i\rangle \,\|\, s_i') = 0\,\right]\Pr\left[\,s_i \neq s_i'\,\right] \\
&= \frac{1}{2} + \frac{1}{4}\Pr\left[\,s_i \neq s_i'\,\right] \\
&= \frac{1}{2} + \frac{1}{4}(1 - \kappa(\mathcal{C}_{h_{i-1}}))
\end{aligned}$$

where h_{i-1} is the history at the time that s_i, s_i' are sampled, and $\kappa(\mathcal{C}_h)$ is the collision probability of the indicated distribution. Thus, when the message was $m = 0$, we have $\mu_0 = \sum_{i=1}^{\ell} \mathbb{E}[X_i] = \mu - \frac{1}{4}\sum_{i=1}^{\ell}(1 - \kappa(\mathcal{C}_{h_{i-1}}))$. By a symmetrical argument, when $b = 1$ and the message was $m = 1$, we have $\Pr[\,X_i = 1\,] = \frac{1}{2} + \frac{1}{4}(1 - \kappa(\mathcal{C}_{h_{i-1}}))$, and $\mu_1 = \sum_{i=1}^{\ell} \mathbb{E}[X_i] = \mu + \frac{1}{4}\sum_{i=1}^{\ell}(1 - \kappa(\mathcal{C}_{h_{i-1}}))$.

Let $\gamma = \frac{1}{4}\sum_{i=1}^{\ell}(1-\kappa(\mathcal{C}_{h_{i-1}}))$, and let $\beta_{i-1} = -\log \max_s \Pr\left[S \leftarrow\!\!\$\, \mathcal{C}_{h_{i-1}} : S = s\right]$. By standard results relating the min-entropy and the Rényi-entropy, it follows that $2^{-2\beta_{i-1}} \leq \kappa(C_{h_{i-1}}) \leq 2^{-\beta_{i-1}}$ for all $i \in [\ell]$. Thus,

$$\frac{1}{4}\sum_{i=1}^{\ell}\left(1-\frac{1}{2^{\beta_{i-1}}}\right) \leq \gamma \leq \frac{1}{4}\sum_{i=1}^{\ell}\left(1-\frac{1}{2^{2\beta_{i-1}}}\right)$$

which leads to upper- and lowerbounds on μ_0 and μ_1 in terms of μ and the min-entropies

$$\mu - \frac{1}{4}\sum_{i=1}^{\ell}\left(1-\frac{1}{2^{2\beta_{i-1}}}\right) \leq \mu_0 \leq \mu - \frac{1}{4}\sum_{i=1}^{\ell}\left(1-\frac{1}{2^{\beta_{i-1}}}\right)$$
$$\mu + \frac{1}{4}\sum_{i=1}^{\ell}\left(1-\frac{1}{2^{\beta_{i-1}}}\right) \leq \mu_1 \leq \mu + \frac{1}{4}\sum_{i=1}^{\ell}\left(1-\frac{1}{2^{2\beta_{i-1}}}\right)$$

Recall that $\mu = \ell/2$, and an assumption of the theorem is that $\forall i \in [q], \beta_{i-1} \geq 1$. Since the β_{i-1} are also finite, we have $\frac{\ell}{4} < \mu_0 \leq \frac{3\ell}{8}$, and $\frac{5\ell}{8} \leq \mu_1 < \frac{3\ell}{4}$. Thus, in the case that $b = 1$, we have

$$\begin{aligned}
\Pr[\mathcal{A}_1 \Rightarrow 0] &= \Pr[\mathcal{A}_1 \Rightarrow 0 | m=0]\Pr[m=0] + \Pr[\mathcal{A}_1 \Rightarrow 0 | m=1]\Pr[m=1] \\
&< \Pr\left[\sum_{i=1}^{\ell} X_i > (1-\delta)\frac{\ell}{2} \,\middle|\, m=0\right] p_m(0) + \Pr\left[\sum_{i=1}^{\ell} X_i < (1+\delta)\frac{\ell}{2} \,\middle|\, m=1\right] p_m(1) \\
&< \Pr\left[\sum_{i=1}^{\ell} X_i > \frac{7\ell}{16} \,\middle|\, m=0\right] p_m(0) + \Pr\left[\sum_{i=1}^{\ell} X_i < \frac{9\ell}{16} \,\middle|\, m=1\right] p_m(1)
\end{aligned}$$

where the last line is obtained by setting $\delta = 1/8$. (We choose $\delta = 1/8$ because $7\ell/16$ is the midway point between $3\ell/8$, the upperbound on μ_0, and $\ell/2$; likewise $9\ell/16$ is midway between $\ell/2$ and $5\ell/8$.) Since $\mathcal{A}_1$ is more likely to output 0 when μ_0, μ_1 are closer to μ, we rewrite $7\ell/16$ and $9\ell/16$ in the following multiplicative Chernoff bound

$$\begin{aligned}
\Pr[\mathcal{A}_1 \Rightarrow 0] &< \Pr\left[\sum_{i=1}^{\ell} X_i > \left(1+\frac{1}{6}\right)\frac{3\ell}{8} \,\middle|\, m=0\right] p_m(0) + \Pr\left[\sum_{i=1}^{\ell} X_i < \left(1-\frac{1}{10}\right)\frac{5\ell}{8} \,\middle|\, m=1\right] p_m(1) \\
&\leq e^{-\frac{(3\ell/8)(1/6)^2}{3}} p_m(0) + e^{-\frac{(5\ell/8)(1/10)^2}{2}} p_m(1) \\
&= e^{-\frac{\ell}{288}} p_m(0) + e^{-\frac{\ell}{320}} p_m(1)
\end{aligned}$$

where ≪**TomS 4.2:** If the lowerbound on the min-entropies β_{i-1} is assumed to be larger, the bounds become better. This would give a small improvement, here.≫ This allows us to conclude that $\Pr[\mathcal{A}_1 \Rightarrow 1 \,|\, b=1] - \Pr[\mathcal{A}_1 \Rightarrow 1 \,|\, b=0]$ is at least

$$1 - \left(e^{-\frac{\ell}{288}} p_m(0) + e^{-\frac{\ell}{320}} p_m(1) + 2e^{-\frac{\ell}{128}}\right)$$

when $\delta = 1/8$. For example, when $\ell = 256$, the advantage of $\mathcal{A}_1$ is at least $1 - 0.412 p_m(0) - 0.450 p_m(1) - 0.271$, or roughly 0.3 when $p_m(0) = p_m(1) = 0.5$. ≪**TomS 4.3:** This bound is improved if you assume knowledge of m and use it to check just one "side", mostly because the factor of 2 in $2\exp(\ell/128)$ disappears. But asymptotically, the results are the same.≫

Corollary 1 *(Informal.)*Assume that $\forall i \in [q]$ and h_{i-1}, the distributions $\mathcal{C}_{h_{i-1}}$ have min-entropy of at least one. When $H\colon \{0,1\}^* \to \{0,1\}$ is a random oracle, the `One-Bit` scheme $(\mathsf{enc}^H, \mathsf{dec}^H)$ in Figure 5 is not XXX-secure. In particular, setting $\delta = 1/8$, adversary $\mathcal{A}_2$ achieves advantage at least $1 - \left(e^{-\frac{\ell}{288}} p_m(0) + e^{-\frac{\ell}{320}} p_m(1)\right)^{|\mathcal{K}|} - 1/c$ when $c > 1$ is a constant such that $\ell \geq 128 \ln(2c|\mathcal{K}|)$.

Proof: In the case that $b = 0$, we have $\Pr[\mathcal{A}_2 \Rightarrow 1] \leq |\mathcal{K}|2e^{-\frac{\ell\delta^2}{2}}$ by a union bound and the proof of the previous theorem. In the case that $b = 1$, let Key_i be the event that key K_i causes $\mathcal{A}_1(c_1, c_2, \ldots, c_\ell, K_i)$ to output 1, for some enumeration $K_1, K_2, \ldots, K_{|\mathcal{K}|}$ of $\mathcal{K}$. ≪**TomS 4.4:** The running time of the attack can be improved by ordering keys from most-to-least likely.≫ Consider $\Pr[\mathcal{A}_2 \Rightarrow 1] = 1 - \Pr[\mathcal{A}_2 \Rightarrow 0] = 1 - \Pr[\bigwedge_i \neg\mathrm{Key_i}] = 1 - \prod_i \Pr[\neg\mathrm{Key_i}]$, where independence follows because the keys are distinct and H is modeled as a random oracle. Thus, again taking $\delta = 1/8$ we have

$$\Pr[\mathcal{A}_2 \Rightarrow 1] - \Pr[\mathcal{A}_2 \Rightarrow 1] \geq 1 - \left(e^{-\frac{\ell}{288}}p_m(0) + e^{-\frac{\ell}{320}}p_m(1)\right)^{|\mathcal{K}|} - |\mathcal{K}|2e^{-\frac{\ell}{128}}$$

To ensure that the final term $|\mathcal{K}|2e^{-\frac{\ell}{128}} \leq 1/c$ for some constant $c > 1$, we require $\ell \geq 128\ln(2c|\mathcal{K}|)$. Moreover, the previous theorem tells us that when $\ln(2c|\mathcal{K}|) > 2$, so that $\ell > 256$, the parenthesized term vanishes as $|\mathcal{K}|$ grows.

5 DTE-Encrypt-DTE Constructions

We investigate a natural methodology for building DME schemes, which can be seen as generalization of the Rank-Encipher-Unrank approach introduced for FPE [**?**]. We start by providing a more general treatment of distribution-transforming encoders (DTE), originally introduced by Juels and Ristenpart [**?**] for use in building HE schemes.

DTE schemes. A DTE scheme $\mathsf{DTE} = (\mathsf{encode}, \mathsf{decode}, p_x, p_y)$ is a pair of algorithms and pair of distributions. The possibly randomized encode algorithm encode takes as input a value $X \in \mathcal{X} \equiv \mathrm{Supp}(p_x)$ and outputs a value $Y \in \mathcal{Y} \equiv \mathrm{Supp}(p_y)$. The possibly randomized decode algorithm decode takes as input a value $Y \in \mathcal{Y}$ and outputs a value $X \in \mathcal{X}$. We require that for all $X \in \mathcal{X}$

$$\Pr[\mathsf{decode}(\mathsf{encode}(X)) = X] = 1$$

where the probability is over the random coins used by the two algorithms. ≪**TomR 5.1:** We may want to generalize this notion of correctness to match the DME ones later.≫

In their use by [**?**] for HE, a DTE was used with some distribution p_x over plaintext messages and p_y was the uniform distribution over bit strings of an appropriate length. Here we will also be interested in DTEs that convert from uniform distributions over bit strings to values that should appear to be non-uniformly distributed.

The construction. Let p_m be a message distribution, p_c be a ciphertext distribution, and p_x and p_y be the uniform distribution over bit strings of lengths that will be defined appropriately below. Let $\mathsf{DTE}_m = (\mathsf{encode}_m, \mathsf{decode}_m, p_m, p_x)$ and $\mathsf{DTE}_c = (\mathsf{encode}_c, \mathsf{decode}_c, p_y, p_c)$ be DTE schemes. Let $\mathsf{SE} = (\mathsf{enc}, \mathsf{dec})$ be a conventional symmetric encryption scheme. Then the DTE-Encrypt-DTE scheme $\mathsf{DME} = (\mathsf{Enc}, \mathsf{Dec})$ has encryption and decryption defined by

$$\mathsf{Enc}(K, M) = \mathsf{encode}_c(\mathsf{enc}(K, \mathsf{encode}_m(M))) \qquad \text{and} \qquad \mathsf{Dec}(K, C) = \mathsf{decode}_m(\mathsf{dec}(K, \mathsf{decode}_c(C)))\ .$$

If we set p_m and p_c to be uniform, then building a DTE-Encrypt-DTE scheme is equivalent to building a Rank-Encrypt-Unrank style scheme [**?**, **?**]. ≪**TomS 5.2:** If I take Rank as encode_m and Unrank as encode_c, so that I follow your DTE-Encrypt-DTE construction, then I don't see what can be asserted about p_m (although $\mathrm{Supp}(p_m)$ is strings of a particular format). However, p_y should be uniform for good Encrypt stages, so likewise p_c uniform over the support of p_c. Note: even if $\mathrm{Supp}(p_m) = \mathrm{Supp}(p_c)$, i.e. FPE, we don't want to demand $p_m = p_c$ if we want "typical" kinds of security.≫ When p_c is uniform over bit strings, and hence DTE_c can be the identity, then the resulting DTE-Encrypt scheme is equivalent to an HE scheme. ≪**TomS 5.3:** Isn't this under the assumption that $\mathrm{dec}_{K'}(Y)$ is uniform over bitstrings when K' is the wrong key and Y is uniform? That's probably true for natural schemes, even when $K' = K$, although not for CBC (or other schemes) with padding/redundancy.≫ When p_m is arbitrary ≪**TomR 5.4:** need to define this≫ then Encrypt-DTE gives rise to a stegonagraphy scheme with covertext distribution p_c. ≪**TomS 5.5:** A couple of observations: (1) p_x need not be uniform if one uses typical encryption schemes, which are provably secure against adversarially chosen p_x; (2) if we want to target "typical" kinds of security notions for this DME construction, we'll need p_m to be adversarially chosen; this fights against (3) if we want HE-type properties, then p_x and p_m are pretty restricted, based on current knowledge; (4) for typical enc, we'll have p_y uniform-ish.≫

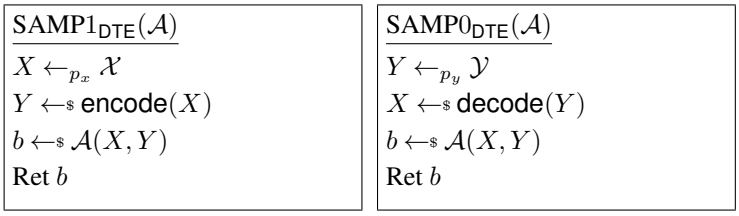

Figure 6: Games used to define security of DTE scheme $\mathsf{DTE} = (\mathsf{encode}, \mathsf{decode}, p_x, p_y)$.

In the above we have glossed over the lengths of strings, implicitly assuming that the domains and ranges of DTE_m, DTE_c, and SE align appropriately. We will see how this plays out in detail for particular constructions below.

The main challenge towards using the DTE-Encrypt-DTE construction is that of building DTEs that transform uniform bit strings to other distributions. The DTE schemes explored thus far, including all those in [**?**], work in the other direction. One approach for doing so is to leverage the large literature on random variable sampling algorithms that use uniform random variables to generate, for example, normal or exponentially distributed random variables. Unfortunately, these algorithms do not easily provide DTEs since they are not invertible (being typically many-to-one).

6 DTE constructions

6.1 Pseudorandom sampling: Improved DTE for RSA private keys

In [], an HE scheme is introduced for encryption of RSA private keys. An RSA private key consists of a pair of ℓ-bit primes (p, q), where ℓ is a security parameter. The target distribution in [], one closely approximating key generation for real-world applications, such as SSL, is such that p and q are uniformly and independently distributed over ℓ-bit primes. It is convenient in our exploration here to focus on encryption of a single prime π, i.e., $\mathcal{M}$ is the set of ℓ-bit primes.

The HE scheme proposed in [] relies on a DTE based on rejection sampling. A seed s consists of a sequence of t independent random integers $\pi_1, \pi_2, \ldots, \pi_t \in [2^{\ell-1}, 2^{\ell})$. The DTE maps a seed s to a corresponding prime by applying a primality test (e.g., Miller-Rabin) to each integer in the sequence until a prime π is identified.[1] The prime number theorem states that the probability that a randomly selected ℓ-bit integer is prime is $O(1/\ell)$. Unfortunately, this means that a randomly generated seed s contains a prime with high probability only if $t = \omega(\ell)$. Thus the DTE proposed in [] results in an HE with superlinear ciphertext expansion.[2]

We propose a DTE with small constant (a factor of 2) ciphertext expansion.

Preliminaries: Let $\mathsf{prng}_\ell(\sigma, i)$ denote a PRNG that takes as input a PRNG seed $\sigma \in \{0,1\}^\ell$ (distinct from a DTE seed) and index $i \in \mathbb{Z}$ and yields an ℓ-bit output r_i. We drop the subscript ℓ where it is clear from context. Let $[z]_{\uparrow 1}$ denote the result of setting the leading bit of a binary representation of integer z to 1. Let $\mathsf{primetest}(z) \rightarrow \{\mathtt{false}, \mathtt{true}\}$ denote a primality testing algorithm, such as Miller-Rabin. Finally, let $\perp$ be a distinguished error symbol.

Construction:

Our full DTE construction is specified in Figure 7.

In this construction, a DTE seed $s = (\sigma, \nu)$, where σ is a PRNG seed and ν is a "mask." In the decode algorithm, a sequence of integers $r_1 \oplus \nu, r_2 \oplus \nu, \ldots$ is generated, where $r_j = \mathsf{prng}(\sigma, j)$ until a prime $\pi = r_i \oplus \nu$ is identified, i.e., $\mathsf{primetest}(r_i \oplus \nu) = \mathtt{true}$. We refer to this as the *seed sequence*.

The mask ν allows a seed s to be "cooked" for a given input π to the encode algorithm. Let $\gamma(i)$ denote the probability that the first prime in a sequence of random ℓ-bit integers appears in the i^{th} position. By selecting the ν

[1] If no prime is identified, then a prime is generated at random. Thus the DTE is probabilistic.

[2] It might seem that a more efficient construction is possible in which s is a seed for a PRNG yielding as output the sequence of ℓ-bit integers $\pi_1, \ldots, \pi_t$. For this DTE construction, however, there exists no efficient encoding, i.e., way to select s such that $\pi \in \{pi_1, \ldots, \pi_t\}$.

appropriately, we can meet two requirements: (1) π is assigned randomly to a position i sampled from γ and (2) π is the first prime in the seed sequence.

In the algorithm encode, an index i is sampled from γ by selecting trial random integers until a prime is found; counting the number of such trials yields i. A seed $s = (\sigma, \nu)$ is then generated in which π is set to position i according to requirement (1); this is done by selecting σ at random and letting $\nu = r_i \oplus \pi$. Of course, it is possible then that π is not the first prime in the seed sequence for s, i.e., that requirement (2) is violated. In this case, the seed is rejected and another generated. Such rejection sampling continues until requirement (2) is met.

Security:

Let us assume for simplicity that primetest is ideal, i.e, always correct. We will prove security information theoretically, while modeling prng as a random oracle. This means that we compute probabilities over coins ξ specifying the oracle.

Let $\Gamma(t) = \sum_{j=t+1}^{\infty} \gamma(i)$. In other words, $\Gamma(t)$ is the probability that a seed sequence of length t contains no prime. Let Π_ℓ denote the set of ℓ-bit primes. Let $\mathsf{index}(s)$ denote the position of the first prime in the seed sequence of s, i.e., the smallest index (position) i such that $\mathsf{primetest}([\mathsf{prng}(\sigma, i) \oplus \nu]_{\uparrow 1}) = \mathtt{true}$.

Let p_s denote the probability distribution defined by $\mathsf{pr}[s = s' : \pi \stackrel{\$}{\leftarrow} \Pi_\ell; s \leftarrow \mathsf{enc}(\pi)]$. In other words, it is the distribution over seeds induced by encoding a prime randomly selected from Π.

Define $p(p_s, U) = \sum_{s \in \mathcal{S}} |p_s - 1/|\mathcal{S}||$, i.e., $p(p_s, U)$ is the L_1 distance between p_s and the uniform distribution over seeds.

Lemma 1 *Suppose $\gamma(i) \geq 2^{-\ell/4}$ for $i \in [1, t]$ and $\Gamma(t) \leq 2^{-z}$ for some $0 \leq z < 5\ell/4$. Let prng be a random oracle instantiated with coins ξ. With probability at least $1 - 2(t+1)|\Pi|e^{-(2^{\ell/4}-1)}$ over ξ, it is the case that $p(p_s, U) \leq 2^{-z} + 2^{-(\ell/4)+1}$.*

Proof: For a given set of coins ξ, partition the seed space $\mathcal{S}$ into sets $\{S_{\pi,i}\}\}_{\pi \in \Pi, i \in [1,t]} \bigcup S_\perp$. Here, $S_{\pi,i}$ is the set of seeds $s \in \mathcal{S}$ such that $\mathsf{decode}(s) = (\pi, i)$, and and $S_\perp$ is the set of seeds such that $\mathsf{index}(s) > t$ and thus $\mathsf{decode}(s) = \perp$. Finally, let $X_{\pi,i} = |S_{\pi,i}|$.

The function $\mathsf{encode}(\pi)$ selects an index i from distribution γ. Then it selects an ℓ-bit σ uniformly at random and tests whether $(\sigma, \nu) \in S_{\pi,i}$ for $\nu = \mathsf{prng}(\sigma, i) \oplus \pi$. Thus, $\mathsf{encode}(\pi)$ may be viewed as selecting a prime π, then an index i, and then a seed s uniformly at random from $S_{\pi,i}$. Hence

$$p_s(s) = \gamma(i)/(|\Pi| X_{\pi,i}). \tag{1}$$

Let $X_{\pi,i}(s)$ be a Bernoulli random variable such that $X_{\pi,i}(s) = 1$ iff $\mathsf{decode}(s) = (\pi, i)$. (Similarly $s \in X_\perp$ iff $\mathsf{decode}(s) = \perp$.) Thus $X_{\pi,i} = \sum_{s \in \mathcal{S}} X_{\pi,i}(s)$.

Now, $X_{\pi,i}(s) = \mathsf{pr}_\xi[s \in S_{\pi,i}] = \mathsf{pr}[[r_i \oplus \nu]_{\uparrow 1} = \pi | \mathsf{index}(s) = i] \mathsf{pr}[\mathsf{index}(s) = i] = \gamma(i)/|\Pi|$. Define $\mu_{\pi,i} = E[X_{\pi,i}]$. We see that $\mu_{\pi,i} = \gamma(i) 2^{2\ell}/|\Pi|$.

By assumption, $\gamma(i) \geq 2^{-\ell/4}$, implying that $\gamma(i)/|\Pi| \geq 2^{-5\ell/4}$. Taking the standard Chernoff Bound $\mathsf{pr}[X \notin [(1-\delta)\mu, (1+\delta)\mu] \leq 2e^{-(\delta^2\mu)/2}$ for $\delta \in [0, 1]$ and setting $\delta = 2^{-\ell/4}$, therefore, we obtain:

$$\mathsf{pr}[X_{\pi,i} \notin [(1-\delta)\mu_{\pi,i}, (1+\delta)\mu_{\pi,i}]] \leq 2e^{-2^{-\ell/4-1}}. \tag{2}$$

Now $\mu_\perp = \Gamma(t) 2^{2\ell}$. Taking the standard Chernoff bound $\mathsf{pr}[X > (1+\delta)\mu] \leq 2e^{-(\delta^2\mu)/3}$, noting that $\Gamma(t) < 2^{-z}$ by assumption, and letting $\delta = 2^{-\ell/4}$, we observe that

$$\mathsf{pr}[X_\perp > (1+\delta)\mu_\perp] \leq e^{-2^{-\ell/4-1}}/2 \tag{3}$$

for $z \leq 5/4\ell$. ∎

$\underline{\mathsf{decode}(\sigma,\nu)[\ell,t]}$
if $\sigma \notin \{0,1\}^\ell$ or $\nu \notin \{0,1\}^\ell$ then ret $\perp$
$i \leftarrow 0$
do
  $\pi \leftarrow [\mathsf{prng}(\sigma,i) \oplus \nu]_{\uparrow 1}$
  $i \leftarrow i+1$
until ($\mathsf{primetest}(\pi) = \mathtt{true}$ or $i = t+1$)
if $i = t+1$ ret $\perp$
ret $(\pi, [i])$

$\underline{\mathsf{encode}(\pi)[\ell,t]}$
if $\pi \notin \{0,1\}^\ell$ or $\mathsf{primetest}(\pi) = \mathtt{false}$ then ret $\perp$
$i \leftarrow 0$
do
  $z \xleftarrow{\$} \{0,1\}^\ell$
  $i \leftarrow i+1$
until $\mathsf{primetest}(z) = \mathtt{true}$
do
  $\sigma \xleftarrow{\$} \{0,1\}^\ell$
  $\nu \leftarrow \pi \oplus \mathsf{prng}(\sigma,i)$
until $\mathsf{decode}(\sigma,\nu) = (\pi,[i])$
ret (σ,ν)

Figure 7: DTE for primes.

7 DME from Walk Schemes

7.1 Walk Schemes and Random Walks

Fix a finite, non-empty set $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$ of states. Let $\mathbf{P}$ be a (row) stochastic matrix where the entry in row s and column t (written $[P]_{s,t}$) is defined to be $\Pr[\, t \mid s \,]$, where $s, t \in \mathcal{S}$. Together $(\mathcal{S}, \mathbf{P})$ define a (discrete, finite, time-homogeneous) Markov process. Note that row s of $\mathbf{P}$, written $[P]_s$, is a distribution; namely, the distribution over next states $t \in \mathcal{S}$ given that the current state is s. We will assume that $(\mathcal{S}, \mathbf{P})$ defines an ergodic Markov process, so that $\mathbf{P}$ has a unique stationary distribution. As a reminder, this means that $\mathbf{Q} = \lim_{i\to\infty} \mathbf{P}^i$ exists, and that every row of Q is this stationary distribution.

Now, let $\boldsymbol{\pi} = [\pi(s_1)\ \pi(s_2)\ \cdots\ \pi(s_n)]$ be a distribution over $\mathcal{S}$, written as a row vector, and let $\Sigma \subseteq \{0,1\}^*$ be a set. A *walk scheme* is a tuple $(\mathcal{S}, \mathbf{P}, \boldsymbol{\pi}, \mathsf{propose}, \mathsf{decide})$, and in this context we refer to π as the *target distribution* of the walk scheme. The *proposal algorithm* $\mathsf{propose}\colon \Sigma \times \mathcal{S} \to \mathcal{S}$ is a (potentially) randomized algorithm that, on input a state-string $\sigma \in \Sigma$ and an element $s \in \mathcal{S}$, samples a $t \in \mathcal{S}$ according to the *proposal distribution* $[P]_s$, and outputs t. We allow the proposal algorithm to take a state-strimg in order to support instantiations that may, for example, make use of a key or a counter. When $\Sigma = \emptyset$, we omit the state-string σ from the notation. The (potentially) randomized *decision algorithm* $\mathsf{decide}\colon \Sigma \times \mathcal{S} \times \mathcal{S} \to \mathcal{S}$ takes as input a triple (σ, s, t) and returns a state s'. As a correctness condition, we insist that $\mathsf{decide}(\sigma, s, t) \in \{s, t\}$.

Markov Walks induced by Walk Schemes. Intuitively, a walk scheme $(\mathcal{S}, \mathbf{P}, \boldsymbol{\pi}, \mathsf{propose}, \mathsf{decide})$ will be used to specify a particular stochastic "walk" process. In particular, a walk (starting in state $s_0 \in \mathcal{S}$, with initial state-string $\sigma_0 \in \Sigma$) is a sequence of random variables $X_0 = s_0, X_1, X_2, \ldots$ where, for each $i > 0$, the value of X_i is defined by the following steps:

$$\begin{aligned} t &\leftarrow\!\!\$\ \mathsf{propose}(\sigma_i, s_{i-1}) \\ s_i &\leftarrow\!\!\$\ \mathsf{decide}(\sigma_i, s_{i-1}, t) \\ X_i &\leftarrow s_i \end{aligned}$$

where the σ_i are determined by some implicit external process. As a simple example, let $\Sigma = \emptyset$, let $\boldsymbol{\pi}$ be the stationary distribution of $\mathbf{P}$, and define $\mathsf{decide}(s,t) = t$ for all $s, t \in S$. Then the resulting sequence $X_0 = s_0, X_1, X_2, \ldots$ gives a classical "random walk" on the state space S and, for any T that is at least the mixing time of the walk, X_T will be a sample from the distribution $\boldsymbol{\pi}$.

Walks will usually specify a termination condition stop, at which point something is done with the final X_i (e.g. this is provided to some other algorithm). For the moment, we leave the termination condition implicit.

Metropolis-Hastings walks. Set $\Sigma = \emptyset$ and define decide as follows. On input (s,t), compute $\alpha = \min\{1, \frac{\boldsymbol{\pi}(t)}{\boldsymbol{\pi}(s)} \frac{[\mathbf{P}]_{t,s}}{[\mathbf{P}]_{s,t}}\}$. If $\alpha = 1$ then $\mathsf{decide}(s,t) = t$. If $\alpha < 1$ then $\mathsf{decide}(s,t) = t$ with probability α, and $\mathsf{decide}(s,t) = s$ with probability

$1-\alpha$.[3] The last step is easily implemented by picking a uniform real $u \in [0,1]$, and setting $\mathsf{decide}(s,t) = t$ iff $u \le \alpha$.

Rejection-sampling walks. We note that the walk scheme formalism also allows one to captures standard rejection sampling. First, fix $\Sigma = \emptyset$. Instead of calling $t \leftarrow\!\!\$\ \mathsf{propose}(s_{i-1})$, we call $t \leftarrow\!\!\$\ \mathsf{propose}(\mathtt{s})$ for some fixed $\mathtt{s}$ and, correspondingly, we call $s' \leftarrow \mathsf{decide}(\mathtt{s}, t)$. The sequence $X_0, X_1, \ldots$ is terminated as soon as some $X_i \neq \mathtt{s}$. Here, the "matrix" $\mathbf{P}$ need consist of a single row $[\mathbf{P}]_{\mathtt{s}}$ that is the fixed proposal distribution. Thus, one samples repeatedly from $[\mathbf{P}]_{\mathtt{s}}$ until the proposed t is such that $\mathsf{decide}(\mathtt{s}, t) = t$.

Reversible walk schemes. Fix a walk scheme $\mathcal{W}$. For each $s' \in \mathcal{S}$ and $\sigma \in \{0,1\}^*$, let $\mathcal{A}(s' \,|\, \sigma) = \{s \in \mathcal{S} \mid \Pr[\, s' \leftarrow\!\!\$\ \mathsf{propose}(\sigma, s) \wedge s' \leftarrow\!\!\$\ \mathsf{decide}(\sigma, s, s')\,] > 0\}$ be the set of possible s'-ancestors (given state string σ), where the probability is over the coins c_p, c_d of propose and decide, respectively. If T is a number such that $|\mathcal{A}(s' \,|\, \sigma)| \le T$ always, then we say that $\mathcal{W}$ is T-*ambiguous*. For our applications, we desire walk schemes with small T, and for which $\mathcal{A}(s' \,|\, \sigma)$ is efficiently computable when given (σ, c_p, c_d). When $T = 1$ we can uniquely reverse a given walk sequence, hopefully efficiently. ≪**TomS 7.1:** Seems to require deterministic decide, and invertible propose.≫ ≪**TomS 7.2:** From the FPE paper, eprint version, page 12: "In fact, we rather expect that most computationally interesting Markov processes can be recast so as to make them computationally reversible." No further discussion.≫

7.2 The Feistel walk

There is a very natural way to view a Feistel enciphering of a bitstring $M \in \{0,1\}^{2n}$ as a Markov process, since the distribution of the input to round $i+1$ depends on previous rounds only through the input to round i. Moreover, balanced Feistel enciphering, using a random function ρ as the round function, is easily seen as a random walk on state space $\mathcal{S} = \{0,1\}^{2n}$: from any state $s_i = M_\ell \,\|\, M_r$ (the input to Feistel round i), select uniformly among the 2^n possible states $M_r \,\|\, X$ where $X \in \{0,1\}^n$. Uniform selection occurs because one computes the next state $s_{i+1} = M_r \,\|\, (M_\ell \oplus \rho(i+i, M_r))$. The graphical representation of this Markov chain is a 2^n-regular graph with vertex set $\mathcal{S}$. Note that since the graph is regular, the stationary distribution is uniform.[4] ≪**TomS 7.3:** The relationship between Feistel (more generally, substitution-permutation networks) and Markov chains seems to have been known for a long time. There are a bunch of papers from the 80s and 90s that use this viewpoint to do differential/linear cryptanalysis.≫

Casting this in our walk scheme syntax, we write $\mathcal{F} = (\{0,1\}^{2n}, \mathbf{P}, \boldsymbol{\pi}, \mathsf{propose}, \mathsf{decide})$ where $[\mathbf{P}]_{s,t} = 1/2^n$ if $s = X \,\|\, Y$ and $t = Y \,\|\, Z$ for $X, Y, Z \in \{0,1\}^n$, and $[\mathbf{P}]_{s,t} = 0$ otherwise. (Note that $\mathbf{P}$ is not symmetric, although it is doubly stochastic.) The proposal algorithm is as follows: on input $\sigma = (\rho, i)$ and $s_{i-1} = X \,\|\, Y$, return $Y \,\|\, (X \oplus \rho(i, Y))$. (In the computational setting, $\sigma = (K, i)$ and $\rho(i, Y)$ is replaced by $F_K(i, Y)$ for PRF F with the appropriate domain and range.) The decision algorithm is trivial, namely on input any (σ, s_{i-1}, t), it returns t. As we just noted, the target stationary distribution $\boldsymbol{\pi}$ for this walk scheme must be the uniform distribution.

Observe that the Feistel walk is uniquely reversible, if one knows the description of ρ, and the number of steps that were taken. Equivalently, if one knows the coins that were used at each step to sample the proposal. Note that decide is deterministic, so each state has a uniquely determined ancestor.

It is straightforward to cast unbalanced Feistel, and ciphers derived from shuffling (e.g. Thorpe, swap-or-not) as walk processes over appropriately specified walk schemes. ≪**TomS 7.4:** It's curious that the intro to the "Swap-or-not" paper makes a big deal of the fact that everything in practice is either Feistel or a SP-network, and that swap-or-not is an entirely new kind of beast. From our perspective, these are all examples of enciphering via a walk scheme.≫

Feistel with non-uniform output distributions. By altering the decision algorithm, one can affect other-than-uniform distributions $\boldsymbol{\pi}$. For example, say one wished to bias the output towards bit strings that have many 0-bits. Then the target stationary distribution might be $\boldsymbol{\pi}(s) = 2^{z(s)}/N$ where $z(s)$ is the number of 0-bits in string

[3] For completeness, we must addresss the corner cases of $\boldsymbol{\pi}(s) = 0$ or $[\mathbf{P}]_{s,t} = 0$. If $[\mathbf{P}]_{s,t} = 0$, then $\mathsf{propose}(\sigma, s)$ never returns t, so this is not an issue. If you want $\boldsymbol{\pi}(s) = 0$ then you should **never** accept s as a proposal. That's intuitive, but I don't know if it is correct w.r.t. the analysis of the algorithm

[4] In general, for a connected, non-bipartite graph, a random walk has stationary distribution $\boldsymbol{\pi}(s) = \deg(s)/2|E|$. Both connectedness and non-bipartiteness are necessary for a unique stationary distribution.

(state) s, and N is the appropriate normalizing constant need to make $\boldsymbol{\pi}(s)$ a distribution.[5] To achieve this $\boldsymbol{\pi}(s)$, the decide algorithm is as described for Metropolis-Hastings walks: on input $(s,t) \in \{0,1\}^{2n} \times \{0,1\}^{2n}$, compute $\alpha = \min\left\{1, 2^{(z(t)-z(s))}\frac{[\mathbf{P}]_{t,s}}{[\mathbf{P}]_{s,t}}\right\}$, and return t with probability α (i.e. "accept" the Feistel round proposal t), and return s with probability $1-\alpha$ (i.e. "reject" the proposal t).≪**TomS 7.5:** BTW, this "proposal", "propose", "accept" language is standard in MCMC literature.≫ The main algorithmic issue is to make sure the walk is reversible; see the following discussion of general MH-walk encryption, and the pseudocode in Figures 8,9.

≪**TomS 7.6:** An interesting application for, say, format-preserving encrpytion of passwords might work like this. The state space is ASCII strings pw of length n. Transition matrix $\mathbf{P}$ is whatever. The target stationary distribution $\boldsymbol{\pi}$ is something like $\boldsymbol{\pi}(pw) = 2^{\sum_i T_i(pw)}/N$ where $T_1(pw) = 1$ iff pw contains a number, $T_2(pw) = 1$ iff pw contains an uppercase, $T_3(pw) = 1$ iff pw contains a non-alphanumeric character, etc. This would give ciphertexts (of plaintext passwords) that are biased towards being "good" passwords.≫

7.3 Encryption via MH Walks

Here we describe how to implement stateful encryption from a walk scheme and a suitable PRF. In particular, fix $\mathcal{S}, \mathbf{P}, \boldsymbol{\pi}$. Assume that there is a deterministic algorithm $\mathsf{sample}_{\mathbf{P}}\colon \mathcal{S} \times \{0,1\}^* \to \mathcal{S}$ that faithfully samples the distribution $[\mathbf{P}]_s$ when provided a random coin-string $c \in \{0,1\}^*$ as input, this holding for all $s \in \mathcal{S}$. Moreover, assume that sample is invertible in the following sense: for every c there is exactly one pair (t,s) such that $t = \mathsf{sample}_{\mathbf{P}}(s,c)$. We write $s \leftarrow \mathsf{sample}_{\mathbf{P}}^{-1}(t,c)$ for computing the inverse. (In a moment, we'll see how to implement such a $\mathsf{sample}_{\mathbf{P}}$ in specific cases.)

Let $F\colon \{0,1\}^k \times (\{0,1\} \times \mathbb{Z} \times \mathbb{Z}) \to \{0,1\}^*$ be a function family. Then we can define a stateful encryption algorithm $\mathsf{MHenc}\colon \{0,1\}^k \times \mathbb{Z} \times \mathbb{Z} \times \mathcal{S} \to \mathbb{Z} \times \mathcal{S}$ as shown in Figure 8. Encryption is relatively simple. Given a message counter j and an agreed upon parameter rnds, we take a MH walk, staring from message $M \in \mathcal{S}$, using the PRF to generate coins for sampling proposals and deciding whether or not proposals are accepted.

Figure 8 also gives the code for decryption $\mathsf{MHdec}\colon \{0,1\}^k \times \mathbb{Z} \times (\mathbb{Z} \times \mathbb{R}) \times \mathcal{S} \to \mathbb{Z} \times \mathcal{S}$, which is signficantly trickier. Consider the first step in the reverse-walk, which needs to find the message $M \in \mathcal{S}$ that corresponds to the input ciphertext $C \in \mathcal{S}$. The state C could have been reached in one of two ways: either from some other state $s \neq C$, in which case it must have been that $\mathsf{sample}_{\mathbf{P}}(s, c_\ell)$ returned C *and* the coins d_ℓ resulted in C being accepted; or C was reached from C, as a result of $\mathsf{sample}_{\mathbf{P}}(C, c_\ell)$ returning some state $t \neq C$ that was *not* accepted. Unfortunately, we cannot always determine which of the two cases is the correct one, and so we must keep both possibilities alive, and recurse. Notice that one could have as many as 2^{rnds} "live" paths at the end of this process, so we prune away paths based on a parameter δ. Namely, if the probability of a potential path is less than δ, it is dropped from further consideration. Alternatively, we could keep the N_p most probable paths alive at each step; see Figure 9. ≪**TomS 7.7:** This is probably the better option.≫ In either case, at the end of the process, decryption returns the most likely path endpoint (i.e. message) from among the paths that remain alive.

Implementing $\mathsf{sample}_{\mathbf{P}}$. The choice of the proposal matrix $\mathbf{P}$ will have a large effect on the implementation of $\mathsf{sample}_{\mathbf{P}}$, and (likely) the performance of encryption and decryption. On one hand, consider $[\mathbf{P}]_s$ as the uniform distribution for all $s \in \mathcal{S}$. Then $\mathsf{sample}_{\mathbf{P}}(s,c) = s \oplus c$ appropriately samples $[\mathbf{P}]_s$, assuming s is a bitstring and $|s| = |c|$. Clearly, this is invertible in the required sense. (Note that, in this case, $[\mathbf{P}]_{s,t} = [\mathbf{P}]_{t,s}$ and so the computation of the acceptance probability α is simplified.) However, if the target distribution $\boldsymbol{\pi}$ is far from uniform –say it places most of its probability mass on a small subset of $\mathcal{S}$– then one expects to need a much longer MH walk to approach the target. ≪**TomS 7.8:** More research required to understand this tradeoff.≫

Sampling proposals from $[\mathbf{P}]_s$ that is the "Feistel distribution" (given current state s) is likewise straightforward, even though this is a highly non-uniform distribuion. In particular, $\mathsf{sample}_{\mathbf{P}}(s = X \,\|\, Y, c = \rho(i,Y)) = Y \,\|\, (X \oplus c)$.

Sampling proposals from more general non-uniform distributions requires additional thought; in particular because decryption must be able invert it. Certain distributions, for example the Normal distribution, can be approximately sampled by invoking the central limit theorem. For example $\mathsf{sample}_{\mathbf{P}}(s, c = (c_1, c_2, \ldots, c_m)) = s + (1/Z)\left(\sum_{1 \le i \le m}(c_i - \mathbb{E}[c_i])\right)$ where Z normalizes to give the sum unit variance, and arithmetic operations are

[5] In this case, one could compute N analyticallly, but a selling point of Metropolis-Hastings walks is that one does not need to.

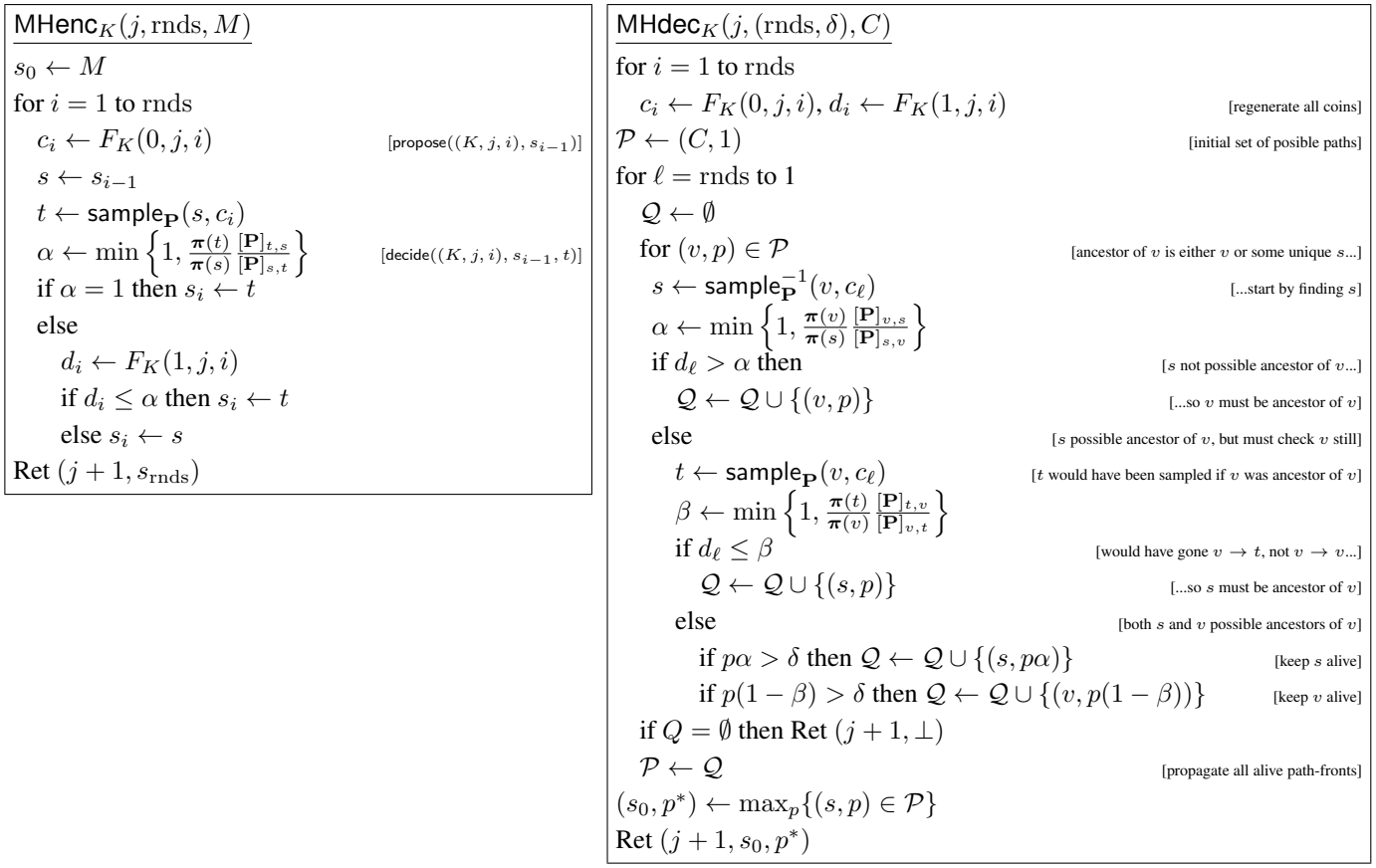

Figure 8: **Left:** Encryption using the MH walk. In the check "$d_i \leq \alpha$" we assume that d_i is treated as a real number in $[0, 1)$. **Right:** Corresponding decryption algorithm. Parameter δ is the probability threshold for keeping candidate decryption paths (i.e., reverse walks) alive. Decryption returns the most likely path endpoint from among those remaining after rnds reverse-steps. [NOTE: to avoid underflow, use the log-trick instead of multiplying probabilities together]

defined appropriately for the type of s and c. (Note that arithmetic operations must also allow invertibility, so modular addition may be the wrong choice.)

Acknowledgements

The authors thank Yevgeniy Dodis for insightful comments.

A Message Indistinguishability

≪**TomR A.1:** The below is basically a scratchpad, you should probably ignore it for now...≫

Returning to the intuition given above, our goal for HE security is to ensure that trial decryptions result in plausible honeymessages. This should hold *even* for low-entropy keys and messages, setting HE security apart from more traditional goals. We start with the following security game, which we call message indistinguishability (MI). It asks that an HE scheme ensure that, relative to some message distribution p_m and key distribution p_k, an attacker cannot distinguish between a challenge message M^* drawn from p_m and a message produced by decrypting an honest encryption of M^* using an adversarially-specified key. See Figure 10. The advantage of an adversary $\mathcal{A}$ in winning the MI game is defined as

$$\mathbf{Adv}^{\text{mi}}_{\mathsf{HE},p_m,p_k}(\mathcal{A}) = 2 \cdot \Pr\left[\text{MI}^{\mathcal{A}}_{\mathsf{HE},p_m,p_k} \Rightarrow \mathsf{true}\right] - 1$$

$\underline{\mathsf{MHenc}_K(j, \text{rnds}, M)}$

$s_0 \leftarrow M$

for $i = 1$ to rnds

  $c_i \leftarrow F_K(0, j, i)$ [propose$((K, j, i), s_{i-1})$]

  $s \leftarrow s_{i-1}$

  $t \leftarrow \mathsf{sample}_{\mathbf{P}}(s, c_i)$

  $\alpha \leftarrow \min\left\{1, \frac{\boldsymbol{\pi}(t)}{\boldsymbol{\pi}(s)} \frac{[\mathbf{P}]_{t,s}}{[\mathbf{P}]_{s,t}}\right\}$ [decide$((K, j, i), s_{i-1}, t)$]

  if $\alpha = 1$ then $s_i \leftarrow t$

  else

    $d_i \leftarrow F_K(1, j, i)$

    if $d_i \leq \alpha$ then $s_i \leftarrow t$

    else $s_i \leftarrow s$

Ret $(j + 1, s_{\text{rnds}})$

$\underline{\mathsf{MHdec}_K(j, (\text{rnds}, N_p, \delta), C)}$

for $i = 1$ to rnds

  $c_i \leftarrow F_K(0, j, i)$, $d_i \leftarrow F_K(1, j, i)$ [regenerate all coins]

$\mathcal{P} \leftarrow (C, 1)$ [initial set of posible paths]

for $\ell = \text{rnds}$ to 1

  $\mathcal{Q} \leftarrow \emptyset$

  for $(v, p) \in \mathcal{P}$ [ancestor of v is either v or some unique s...]

    $s \leftarrow \mathsf{sample}_{\mathbf{P}}^{-1}(v, c_\ell)$ [...start by finding s]

    $\alpha \leftarrow \min\left\{1, \frac{\boldsymbol{\pi}(v)}{\boldsymbol{\pi}(s)} \frac{[\mathbf{P}]_{v,s}}{[\mathbf{P}]_{s,v}}\right\}$

    if $d_\ell > \alpha$ then [s not possible ancestor of v...]

      $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(v, p)\}$ [...so v must be ancestor of v]

    else [s possible ancestor of v, but must check v still]

      $t \leftarrow \mathsf{sample}_{\mathbf{P}}(v, c_\ell)$ [t would have been sampled if v was ancestor of v]

      $\beta \leftarrow \min\left\{1, \frac{\boldsymbol{\pi}(t)}{\boldsymbol{\pi}(v)} \frac{[\mathbf{P}]_{t,v}}{[\mathbf{P}]_{v,t}}\right\}$

      if $d_\ell \leq \beta$ [would have gone $v \to t$, not $v \to v$...]

        $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(s, p)\}$ [...so s must be ancestor of v]

      else [both s and v possible ancestors of v]

        $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(s, p\alpha), (v, p(1 - \beta))\}$ [keep s and v alive]

  sort $(\cdot, \cdot) \in \mathcal{Q}$ by second component

  $\mathcal{P} \leftarrow$ top N_p pairs in sorted $\mathcal{Q}$

  if $\max_p\{(s, p) \in \mathcal{P}\} < \delta$ then Ret $(j + 1, \perp)$

$(s_0, p^*) \leftarrow \max_p\{(s, p) \in \mathcal{P}\}$

Ret $(j + 1, s_0, p^*)$

Figure 9: (Maximum Likelihood with threshold) **Left:** Encryption using the MH walk. In the check "$d_i \leq \alpha$" we assume that d_i is treated as a real number in $[0, 1)$. **Right:** Corresponding decryption algorithm. At each step in the reverse walk, the most probable N_p states are kept. Decryption aborts if, at any time, the most probable current state has probability $< \delta$ of being correct. (Note: setting $\delta = 0$ and $N_p = 1$ gives the (one-step) maximum-likelihood decryption.) [NOTE: to avoid underflow, use the log-trick instead of multiplying probabilities together]

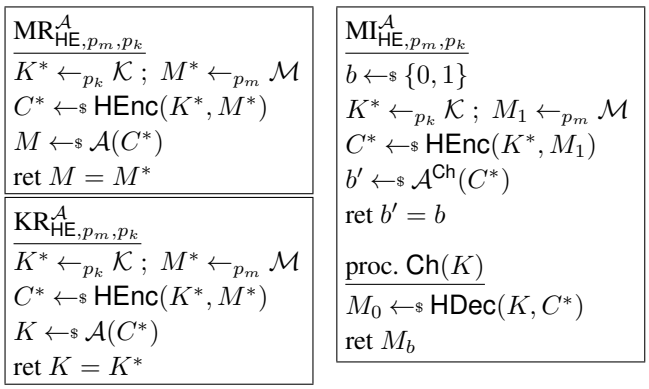

Figure 10: Security game for message recovery (top left), key recovery (bottom left), and message indistinguishability (right).

where the probability is over the coins used by the security game and the event corresponds to the game returning true.

We also define in Figure 10 games for message recovery and key recovery, and define advantage in the normal way for each: $\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{HE},p_m,p_k}(\mathcal{A}) = \Pr[\mathrm{MR}^{\mathcal{A}}_{\mathsf{HE},p_m,p_k} \Rightarrow \mathsf{true}]$ and $\mathbf{Adv}^{\mathrm{kr}}_{\mathsf{HE},p_m,p_k}(\mathcal{A}) = \Pr[\mathrm{KR}^{\mathcal{A}}_{\mathsf{HE},p_m,p_k} \Rightarrow \mathsf{true}]$. Note that unless p_m specifies a single message, these are unknown message attacks, which is weaker than normal mechanisms which allow known or chosen message attacks.

We start with the following proposition, which establishes that MI security implies MR security.

Theorem 3 *Fix distributions p_m, p_k, and HE scheme HE. Let $\mathcal{A}$ be an MR adversary running in time t. Then for the MI adversary $\mathcal{B}$ given in the proof below it holds that*

$$\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{HE},p_m,p_k}(\mathcal{A}) \leq 2 \cdot \mathbf{Adv}^{\mathrm{mi}}_{\mathsf{HE},p_m,p_k}(\mathcal{B})$$

$\mathcal{B}$ runs in time that of $\mathcal{A}$ and makes a single query.

Proof: Adversary $\mathcal{B}$ operates as follows. On input C^*, it runs $\mathcal{A}(C^*)$, which in turn outputs a message guess M. Then $\mathcal{B}$ queries $\mathsf{Ch}(K)$ for an arbitrary K and is given back a message M'. If $M = M'$ then $\mathcal{B}$ outputs 1 and otherwise outputs a random bit. In the case that the MI challenge bit b is 1, we have that $\mathcal{B}$ correctly outputs 1 at least as often as when $\mathcal{A}$ succeeds, meaning $\Pr[\mathrm{MI1}^{\mathcal{B}}_{\mathsf{HE},p_m,p_k} \Rightarrow \mathsf{true}] \geq \Pr[\mathrm{MR}^{\mathcal{A}}_{\mathsf{HE},p_m,p_k} \Rightarrow \mathsf{true}]$ where MI1 is the MI game except with bit set to 1.

$\underline{\mathsf{steg}(\kappa, C)}$
Parse C as $C = C_1 \parallel \ldots, \parallel C_n$, where $C_i \in \{0,1\}^{\ell - r}$
$R_0 \leftarrow 0^r$
for $j = 1$ to n do
    $\tilde{R}_j \leftarrow \mathsf{enc}_\kappa(R_{j-1}; j)$ [deterministic encrypt R_{j-1} under IV j]
    $S_j = \tilde{R}_j \parallel C_j$
    $M_j \leftarrow \mathsf{decode}(S_j)$ [deterministic decode]
    $R_j \leftarrow\!\!\$\ \mathsf{coinextr}(S_j, M_j)$ [probabilistic recovery (whp) of coins]
$\tilde{R}_{n+1} \leftarrow \mathsf{enc}_\kappa(R_n; i)$ [determinsitic encrypt R_{j-1} under IV j]
Encode $\tilde{R}_{n+1}$ under CMU scheme as covertext M
Ret $\left(\{M_j\}_{j=1}^n, M\right)$

Figure 11: Chained steganographic encoding

B AJ's workspace

For a given $\mathsf{DTE} = (\mathsf{encode}, \mathsf{decode})$, let encode^* represent the deterministic function over coins that computes encode. Specifically, for ρ a probability distribution over bitstrings $\{0,1\}^r$, define encode as the algorithm $R \xleftarrow{\rho} \{0,1\}^r; S \leftarrow \mathsf{encode}^*(M; R)$.

We define a DTE as *ϵ-coin-extractable* if there exists a (probabilistic) function $\mathsf{coinextr}$ such that

$$\mathsf{pr}[S = S^* \mid S \leftarrow\!\!\$\ \mathcal{S}; M \leftarrow \mathsf{decode}(S); R \leftarrow \mathsf{coinextr}(S, M); S^* \leftarrow \mathsf{encode}^*(M; R)] \geq 1 - \epsilon. \tag{4}$$

Suppose that C is a (long) message (typically a ciphertext) for which we wish to create a sequence of corresponding covertexts. Assume that $\mathcal{S} = \{0,1\}^\ell$, for $\ell > r$. Given input a secret key κ, and a ciphertext C of length $n(\ell - r)$, our encoding scheme is sketched in Figure 11.

Note that DTE decoding on a seed selected uniformly at random achieves the target distribution sampling required by the CMU scheme.

≪**TomS B.1:** Should it be $R_j \leftarrow \mathsf{coinextr}(S_j, M_j)$, above in Figure 11? And $\tilde{R}_{n+1} \leftarrow \mathsf{enc}_\kappa(R_n; n)$, rather than $\mathsf{enc}_\kappa(R_n; i)$?≫

≪**TomS B.2:** Also, does $\mathsf{enc}_\kappa(R_{j-1}; j)$ mean to encrypt R_{j-1} under "randomness" j (which would be consistent with your notatation, above, for encode^*), or to encrypt j under randomness R_{j-1} (which is more intuitive)? If the former, is enc randomized with its own internal coins (in which case $|\tilde{R}_j| > |R_{j-1}|$) or deterministic?≫

A natural question: Can we characterize modifications to an image as a DTE in order to formalize steganography via embedding in images? Probably not.

B.1 DTE-sampling separation

Given a DTE $(\mathsf{encode}, \mathsf{decode} = G_1, p_x, p_y)$, there is an efficient sampling algorithm over p_x: $Y \leftarrow_{p_y} \mathcal{Y}; X \leftarrow \mathsf{decode}(Y)$. We wish to show, however, that the converse does not hold, i.e., a poly-time sampling algorithm for p_x does not imply a poly-time DTE.

Referring to the definition of a DTE in Section 5, let $\mathcal{X}^* = \{0,1\}^{2\ell}$ and $\mathcal{Y} = \{0,1\}^\ell$. Let $G_1 : \mathcal{Y} \to \mathcal{X}^*$ be a PRG and $\mathcal{X} = G_1(\mathcal{Y})$. Let p_y be the uniform distribution over $\mathcal{Y}$ and $p_x(X) = \Pr\left[X = G_1(Y) \mid Y \leftarrow_{p_y} \mathcal{Y}\right]$.

Observe that there is a trivial efficient sampling algorithm for $\mathcal{X}$ under p_x. But...

Observation 1 *There does not exist a polynomial-time computable DTE* $(\mathsf{encode}, \mathsf{decode} = G_1, p_x, p_y)$.

Proof: [sketch] Suppose such a DTE exists. Then for any $X \in \mathcal{X}$, by definition, $\mathsf{decode}(\mathsf{encode}(X)) = X$. Suppose $\tilde{X} \in \mathcal{X}^* - \mathcal{X}$. Then either: (1) $\mathsf{encode}(\tilde{X}) \notin \mathcal{Y}$ or (2) $\mathsf{decode}(\mathsf{encode}(\tilde{(X)})) \neq X$. Therefore there is a poly-time algorithm that distinguishes $\mathcal{X}$ from $\mathcal{X}^* - \mathcal{X}$ and thus breaks the PRG.

One objection to this example is that p_x is an uninteresting distribution in practice: There is no efficient membership test for $\mathcal{X}$, so it is not a natural message distribution in practice.

Consider, therefore, a variant scheme $G_2 : \mathcal{Y} \to \mathcal{X} * \times \mathcal{P}$ that outputs $X = G_1(Y)$ along with a NIZK proof $P \in calP$ that $X \in \mathcal{X}$. In this case, there is a trivial membership test for $\mathcal{X}$, verification of the proof P. Nonetheless, there remains no poly-time computable DTE. A simulator can be constructed for proof P that enables any such DTE to be used to break the PRG G_1.

B.2 DTE composition theorem

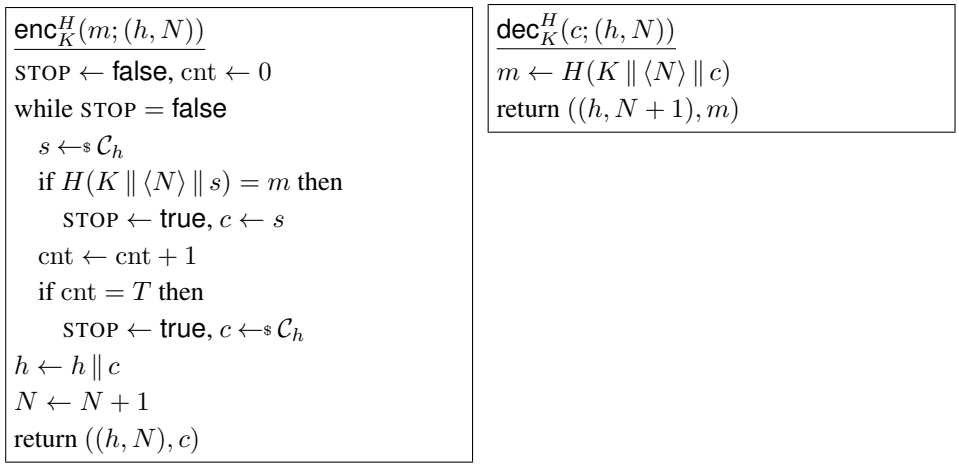

Figure 12: Rejection-sampling encryption of a one-bit message. Stopping threshold T is a system parameter.

B.3 Security analysis of (Truncated) Rejection Sampling

≪**TomS B.3:** Following lemma is for a single-bit message, and a single ciphertext. But I think it extends to multiple-bit and multiple-ciphertexts pretty directly (inclusion of N in the RO-input would allow domain separation across ciphertexts).≫

Lemma 2 *(Informal.)(Warm-up.)* If not provided access to the RO, no adversary can distinguish between an ouput of rejection-sampling encryption and a direct channel sample.

Proof: Fix a one-bit message m. Let $s_0, s_1, \ldots$ be the samples used in the process of encrypting m, and let G_i be the event that the while-loop halts with sample s_i, either because the resampling threshold was reached ($\mathrm{cnt} = T$), or because the i-th sample was the first to satisfy $H(K \,\|\, \langle N \rangle \,\|\, s_i) = m$. Then we can write

$$\begin{aligned}
&\Pr\left[\, c \leftarrow\!\!\$\ \mathsf{enc}_K^H(m;(h,N)) : c = z \,\right] \\
&\quad = \Pr\left[\, s_0 = z \wedge G_0 \,\right] + \Pr\left[\, s_1 = z \wedge G_1 \wedge \neg G_0 \,\right] + \cdots \\
&\qquad + \Pr\left[\, s_{T-1} = z \wedge G_{T-1} \wedge \neg G_{T-2} \wedge \cdots \wedge \neg G_1 \wedge \neg G_0 \,\right] + \Pr\left[\, s_T = z \wedge \neg G_{T-1} \wedge \neg G_{T-2} \wedge \cdots \wedge \neg G_1 \wedge \neg G_0 \,\right]
\end{aligned}$$

Now, note that $\Pr\left[\, s_1 = z \wedge G_1 \wedge \neg G_0 \,\right] = \Pr\left[\, G_1 \wedge \neg G_0 \,|\, s_1 = z \,\right] \Pr\left[\, s_1 = z \,\right]$, and that if the event $\neg G_0 \wedge G_1$ holds given $s_1 = z$, it cannot be the case that $s_0 = s_1 = z$. (We do not care about $s_0 = s_1 \neq z$, obviously.) The same argument can be made for the other terms $\Pr\left[\, s_i = z \wedge G_{i-1} \wedge \neg G_{i-2} \wedge \cdots \wedge \neg G_0 \,\right]$. Thus we do not have to additionally consider collisions among channel samples in the analysis. Continuing, let $\mathcal{C}_h(z)$ be the probability that

$\mathcal{C}_z$ assigns to z, so that

$$\begin{aligned}
&\Pr\left[\, c \leftarrow\!\!\$\ \mathsf{enc}_K^H(m;(h,N)) \colon c = z \,\right] \\
&\quad = \mathcal{C}_h(z)(1/2) + \mathcal{C}_h(z)(1/2)(1-(1/2)) + \cdots + \mathcal{C}_h(z)(1/2)(1-(1/2))^{T-1} + \mathcal{C}_h(z)(1-(1/2))^T \\
&\quad = \mathcal{C}_h(z)(1/2)\left(\sum_{i=0}^{T-1}(1/2)^i\right) + \mathcal{C}_h(z)(1/2)^T \\
&\quad = \mathcal{C}_h(z)(1/2)\frac{1-(1/2)^T}{1-1/2} + \mathcal{C}_h(z)(1/2)^T \\
&\quad = \mathcal{C}_h(z)
\end{aligned}$$

∎

Thus we see that the rejection sampling scheme (for one-bit messages) yields ciphertexts that perfectly mimic samples from the channel distribution. Notice that this is independent of the underlying message m, hence of the message distribution p_m.

Using the previous lemma as a warm-up, consider now the case that the adversary is told the entire RO table "for free". The structure of the proof of the next lemma is very similar to that of the previous one.

Lemma 3 *(Informal.)* Fix a channel history h, thereby fixing a channel distribution $\mathcal{C}_h$, and fix an integer N. Assume the random oracle H in rejection-sampling encryption takes on a particular instantiation $H = f$. For any message $m \in \mathcal{M}$ and key K, let $f_K^{-1}(m)$ be the set of points c such that $f(K \,\|\, \langle N \rangle \,\|\, c) = m$. Then, for any particular $z \in \mathcal{C}_h$,

$$\Pr\left[\, c \leftarrow\!\!\$\ \mathsf{enc}_K^H(m;(h,N)) \colon c = z \,|\, H = f \,\right] = \begin{cases} \mathcal{C}_h(z)\left(\frac{1-(1-\mathcal{C}_h(f_K^{-1}(m)))^T}{\mathcal{C}_h(f_K^{-1}(m))} + (1-\mathcal{C}_h(f_K^{-1}(m)))^T\right) & \text{if } z \in f_K^{-1}(m) \\ \mathcal{C}_h(z)\left(1-\mathcal{C}_h(f_K^{-1}(m))\right)^T & \text{otherwise} \end{cases}$$

where $\mathcal{C}_h(\cdot)$ is the total probability assigned by $\mathcal{C}_h$ to the indicated point or set. Thus, as $T \to \infty$,

$$\Pr\left[\, c \leftarrow\!\!\$\ \mathsf{enc}_K^H(m;(h,N)) \colon c = z \,|\, H = f \,\right] = \begin{cases} \frac{\mathcal{C}_h(z)}{\mathcal{C}_h(f_K^{-1}(m))} & \text{if } z \in f_K^{-1}(m) \\ 0 & \text{otherwise} \end{cases}$$

The full expression in the lemma statement is complex, but the asymptotic version is intutive. Essentially, it says that the probability of m encrypting to z goes to the probability of sampling the the point z given that we are sampling from the set $f_k^{-1}(m)$. As a sanity check, we note that $\sum_{z \in \text{Supp}(\mathcal{C}_h)} \mathcal{C}_h(z)/\mathcal{C}_h(f_K^{-1}(m)) = \mathcal{C}_h(f_K^{-1}(m))/\mathcal{C}_h(f_K^{-1}(m)) = 1$.

Proof: Using the same notation and events from the previous proof

$$\begin{aligned}
&\Pr\left[\, c \leftarrow\!\!\$\ \mathsf{enc}_K^H(m;(h,N)) \colon c = z \,|\, H = f \,\right] \\
&\quad = \Pr\left[\, s_0 = z \wedge G_0 \,|\, H = f \,\right] + \Pr\left[\, s_1 = z \wedge G_1 \wedge \neg G_0 \,|\, H = f \,\right] + \cdots \\
&\qquad + \Pr\left[\, s_{T-1} = z \wedge G_{T-1} \wedge \neg G_{T-2} \wedge \cdots \wedge \neg G_1 \wedge \neg G_0 \,|\, H = f \,\right] \\
&\qquad + \Pr\left[\, s_T = z \wedge \neg G_{T-1} \wedge \neg G_{T-2} \wedge \cdots \wedge \neg G_1 \wedge \neg G_0 \,|\, H = f \,\right] \qquad (5)
\end{aligned}$$

Consider the first term in this sum. Notice that

$$\begin{aligned}
\Pr\left[\, s_0 = z \wedge G_0 \,|\, H = f \,\right] &= \Pr\left[\, s_0 = z \wedge s_0 \in f_K^{-1}(m) \,|\, H = f \,\right] \\
&= \Pr\left[\, s_0 = z \,\right] \mathbb{I}[z \in f_K^{-1}(m)] \\
&= \mathcal{C}_h(z) \mathbb{I}[z \in f_K^{-1}(m)]
\end{aligned}$$

where $\mathbb{I}[z \in f_K^{-1}(m)]$ where $\mathbb{I}[x] = 1$ iff x is true. More generally, for $0 \le i < T$

$$\begin{aligned}
\Pr\left[\, s_i = z \wedge G_i \wedge \neg G_{i-1} \wedge \cdots \wedge \neg G_0 \,\right] &= \Pr\left[\, s_i = z \wedge G_i \,\middle|\, \bigwedge_{j=0}^{j-1} \neg G_j \,\right] \Pr\left[\, \bigwedge_{j=0}^{j-1} \neg G_j \,\right] \\
&= \Pr\left[\, s_i = z \wedge G_i \,\middle|\, \bigwedge_{j=0}^{j-1} \neg G_j \,\right] \prod_{j=0}^{i-1} \Pr\left[\, \neg G_j \,\right]
\end{aligned}$$

where the second line follows because the s_i are sampled independently. We notice immediately that $\Pr[\neg G_j] = 1 - \Pr[G_j] = 1 - \Pr\left[s_j \in f_K^{-1}(m)\right] = 1 - \mathcal{C}_h(f_K^{-1}(m))$. Now, as above,

$$\begin{aligned}\Pr\left[s_i = z \wedge G_i \;\middle|\; \bigwedge_{j=0}^{i-1} \neg G_j\right] &= \Pr\left[s_i = z \wedge s_i \in f_K^{-1}(m) \;\middle|\; \bigwedge_{j=0}^{i-1} \neg G_j\right] \\ &= \Pr\left[s_i = z \wedge s_i \in f_K^{-1}(m)\right] \\ &= \mathcal{C}_h(z)\mathbb{I}[z \in f_K^{-1}(m)]\end{aligned}$$

where we have again used the fact that s_i is independently sampled. Thus $\Pr[s_i = z \wedge G_i \wedge \neg G_{i-1} \wedge \cdots \wedge \neg G_0] = \mathcal{C}_h(z)\mathbb{I}[z \in f_K^{-1}(m)]\left(1 - \mathcal{C}_h(f_K^{-1}(m))\right)^i$. For the case that $i = T$, the last term on the righthand side of Equation 5, we have

$$\begin{aligned}\Pr[s_T = z \wedge \neg G_{T-1} \wedge \neg G_{T-2} \wedge \cdots \wedge \neg G_1 \wedge \neg G_0 \mid H = f] &= \Pr\left[s_T = z \;\middle|\; \bigwedge_{j=0}^{T-1} \neg G_j\right] \Pr[\neg G_j] \\ &= \mathcal{C}_h(z)\left(1 - \mathcal{C}_h(f_K^{-1}(m))\right)^T\end{aligned}$$

and so, overall, we have

$$\begin{aligned}\Pr\left[c \leftarrow\!\!\$\ \mathsf{enc}_K^H(m;(h,N)) \colon c = z \mid H = f\right] &= \mathcal{C}_h(z)\mathbb{I}[z \in f_K^{-1}(m)]\left(\sum_{i=0}^{T-1}\left(1 - \mathcal{C}_h(f_K^{-1}(m))\right)^i\right) \\ &\quad + \mathcal{C}_h(z)\left(1 - \mathcal{C}_h(f_K^{-1}(m))\right)^T\end{aligned}$$

So, when $z \notin f_K^{-1}(m)$, this evaluates to $\mathcal{C}_h(z)\left(1 - \mathcal{C}_h(f_K^{-1}(m))\right)^T$, as the lemma states. On the other hand, when $z \in f_K^{-1}(m)$, we have

$$\begin{aligned}\Pr\left[c \leftarrow\!\!\$\ \mathsf{enc}_K^H(m;(h,N)) \colon c = z \mid H = f\right] &= \mathcal{C}_h(z)\left(\sum_{i=0}^{T-1}\left(1 - \mathcal{C}_h(f_K^{-1}(m))\right)^i\right) + \mathcal{C}_h(z)\left(1 - \mathcal{C}_h(f_K^{-1}(m))\right)^T \\ &= \mathcal{C}_h(z)\frac{1 - (1 - \mathcal{C}_h(f_K^{-1}(m)))^T}{1 - (1 - \mathcal{C}_h(f_K^{-1}(m)))} + \mathcal{C}_h(z)(1 - \mathcal{C}_h(f_K^{-1}(m)))^T \\ &= \mathcal{C}_h(z)\left(\frac{1 - (1 - \mathcal{C}_h(f_K^{-1}(m)))^T}{\mathcal{C}_h(f_K^{-1}(m))} + (1 - \mathcal{C}_h(f_K^{-1}(m)))^T\right)\end{aligned}$$

as claimed in the lemma statement. ∎

We would like to establish what would make a function f "good" for a given $\mathcal{C}_h$ and message distribution p_m. There are at least two things to consider, corresponding to the two possible values of $\Pr\left[c \leftarrow\!\!\$\ \mathsf{enc}_K^H(m;(h,N)) \colon c = z \mid H = f\right]$ given by the lemma. [...] ≪**TomS B.4:** I had written stuff up for the old, incorrect version of Lemma 3. It's commented out now; I'll try to revisit it tonight, before I leave.≫

Attack and proofs workspace (old stuff). ≪**TomS B.5:** The rest of this section should be considered "intuition building". It may disappear as we move forward.≫ However, when the adversary is provided access to the random oracle H, there is a simple, efficient attack that distinguishes with good probability, when both keys and plaintexts are drawn according to low min-entropy distributions, and the message space is sufficiently large.

Theorem 4 Fix a channel history h. Let μ be the min-entropy of p_m, κ be the min-entropy of kg, and σ be the min-entropy of the channel distribution $\mathcal{C}_h$. Let encryption succeed (i.e., halts before reaching the threshold T) with probability $\epsilon = \epsilon(T,\sigma)$ for all $m \in \mathcal{M}$. (Note that in the ROM for H, this is a function of T and σ.) There exists an adversary $\mathcal{A}$ such that

$$\mathbf{Adv}_{\text{rej}}^{\text{le-rod}}(\mathcal{A}) \geq \left(\frac{1}{2^{\kappa+\mu}} + \frac{1 - 2^{-\mu}}{|\mathcal{M}|}\right)\epsilon - \frac{1}{|\mathcal{M}|}\,.$$

Thus, as $\epsilon \to 1$, the advantage bound goes to

$$\mathbf{Adv}^{\text{le-rod}}_{\text{rej}}(\mathcal{A}) \geq \frac{1}{2^{\mu}} \left(\frac{1}{2^{\kappa}} - \frac{1}{|\mathcal{M}|} \right) .$$

Proof: Assume that the adversary knows the channel history h and the plaintext counter N. ≪**TomS B.6:** I think this will likely be our convention, anyway.≫ On input c, adversary $\mathcal{A}$ sets L to be the most likely key, X to be the most likely plaintext, and outputs 1 iff query $H(L \,\|\, \langle N \rangle \,\|\, c) = X$.

If c was produced by direct sampling from $\mathcal{C}_h$, as in experiment le-ROD0, then $\Pr[\,\mathcal{A} \Rightarrow 1\,] = 1/|\mathcal{M}|$, as no queries to H were made to produce c.

Now, consider experiment le-ROD1, so that c was produced via $\mathsf{enc}^H_L(M;(h,N))$, where $M \leftarrow_{p_m} \mathcal{M}$ and $K \leftarrow\!\!\$\ \mathsf{kg}$. Let E be the event that c was output because $H(K \,\|\, \langle N \rangle \,\|\, c) = M$, rather than because the sampling counter had reached the threshold T. Then

$$\begin{aligned}
\Pr[\,\mathcal{A} \Rightarrow 1 \,|\, E\,] &= \Pr[\,\mathcal{A} \Rightarrow 1 \,|\, E \wedge (K,M) = (L,X)\,]\Pr[\,(K,M) = (L,X)\,] \\
&\quad + \Pr[\,\mathcal{A} \Rightarrow 1 \,|\, E \wedge (K,M) \neq (L,X)\,]\Pr[\,(K,M) \neq (L,X)\,] \\
&= (1)\Pr[\,(K,M) = (L,X)\,] + \Pr[\,\mathcal{A} \Rightarrow 1 \wedge (K,M) \neq (L,X) \,|\, E\,] \\
&= \frac{1}{2^{\kappa+\mu}} + \Pr[\,\mathcal{A} \Rightarrow 1 \wedge (K,M) \neq (L,X) \,|\, E\,]
\end{aligned}$$

where $\Pr[\,(K,M) = (L,X)\,] = \Pr[\,K = L\,]\Pr[\,M = X\,] = 2^{-\kappa}2^{-\mu}$ because of how L and M were fixed. Let us consider the second term in the last line,

$$\begin{aligned}
\Pr[\,\mathcal{A} \Rightarrow 1 \wedge (K,M) \neq (L,X) \,|\, E\,] =& \Pr[\,\mathcal{A} \Rightarrow 1 \wedge (K \neq L \wedge M = X) \,|\, E\,] \\
&+ \Pr[\,\mathcal{A} \Rightarrow 1 \wedge (K \neq L \wedge M \neq X) \,|\, E\,] \\
&+ \Pr[\,\mathcal{A} \Rightarrow 1 \wedge (K = L \wedge M \neq X) \,|\, E\,] \\
=& \Pr[\,\mathcal{A} \Rightarrow 1 \wedge (K \neq L \wedge M = X) \,|\, E\,] \\
&+ \Pr[\,\mathcal{A} \Rightarrow 1 \wedge (K \neq L \wedge M \neq X) \,|\, E\,] \\
=& \Pr[\,\mathcal{A} \Rightarrow 1 \,|\, (K \neq L \wedge M = X) \wedge E\,] \cdot \Pr[\,(K \neq L \wedge M = X) \,|\, E\,] \\
&+ \Pr[\,\mathcal{A} \Rightarrow 1 \,|\, (K \neq L \wedge M \neq X) \wedge E\,] \cdot \Pr[\,(K \neq L \wedge M \neq X) \,|\, E\,] \\
=& \frac{1}{|\mathcal{M}|}\left(\Pr[\,(K \neq L \wedge M = X) \,|\, E\,] + \Pr[\,(K \neq L \wedge M \neq X) \,|\, E\,]\right) \\
=& \frac{1}{|\mathcal{M}|}\Pr[\,K \neq L \,|\, E\,] \\
=& \frac{1 - 2^{-\mu}}{|\mathcal{M}|}
\end{aligned}$$

and so, combining with our earlier derivation, we have, $\Pr[\,\mathcal{A} \Rightarrow 1 \,|\, E\,] = \frac{1}{2^{\kappa+\mu}} + \frac{1-2^{-\mu}}{|\mathcal{M}|}$.

On the other hand, if $\neg E$ holds then c was a direct sample from $\mathcal{C}_h$, without a corresponding H query. Given that $\neg E$ holds, there were previous samples $s_0, s_1, \ldots, s_{T-1}$, some of which may have taken on the same value. Let S be the (multi)set of these samples. Then we can write

$$\begin{aligned}
\Pr[\,\mathcal{A} \Rightarrow 1 \,|\, \neg E\,] &= \Pr[\,H(L \,\|\, \langle N \rangle \,\|\, c) = X \,|\, c \notin S \wedge L = K\,]\Pr[\,c \notin S \wedge L = K\,] \\
&\quad + \Pr[\,H(L \,\|\, \langle N \rangle \,\|\, c) = X \,|\, c \notin S \wedge L \neq K\,]\Pr[\,c \notin S \wedge L \neq K\,] \\
&\quad + \Pr[\,H(L \,\|\, \langle N \rangle \,\|\, c) = X \,|\, c \in S \wedge L = K\,]\Pr[\,c \in S \wedge L = K\,] \\
&\quad + \Pr[\,H(L \,\|\, \langle N \rangle \,\|\, c) = X \,|\, c \in S \wedge L \neq K\,]\Pr[\,c \in S \wedge L \neq K\,] \\
&= \frac{1}{|\mathcal{M}|}\Pr[\,c \notin S\,] + \Pr[\,H(L \,\|\, \langle N \rangle \,\|\, c) = X \,|\, c \in S \wedge L = K\,]\Pr[\,c \in S \wedge L = K\,] \\
&\quad + \Pr[\,H(L \,\|\, \langle N \rangle \,\|\, c) = X \,|\, c \in S \wedge L \neq K\,]\Pr[\,c \in S \wedge L \neq K\,]
\end{aligned}$$

$$
\begin{aligned}
&\geq \frac{1}{|\mathcal{M}|}\Pr\left[\, c \notin S \,\right] + \Pr\left[\, H(L \,\|\, \langle N \rangle \,\|\, c) = X \,|\, c \in S \wedge L \neq K \,\right]\Pr\left[\, c \in S \wedge L \neq K \,\right] \\
&= \frac{1}{|\mathcal{M}|}(1 - \Pr\left[\, c \in S \,\right]) + \Pr\left[\, H(L \,\|\, \langle N \rangle \,\|\, c) = X \,|\, c \in S \wedge L \neq K \,\right]\Pr\left[\, c \in S \wedge L \neq K \,\right] \\
&= \frac{1}{|\mathcal{M}|}(1 - \Pr\left[\, c \in S \,\right]) + \frac{1}{|\mathcal{M}|}\Pr\left[\, c \in S \wedge L \neq K \,\right] \\
&= \frac{1}{|\mathcal{M}|}(1 - \Pr\left[\, c \in S \,\right]) + \frac{1}{|\mathcal{M}|}\Pr\left[\, c \in S \,\right](1 - \Pr\left[\, L = K \,\right]) \\
&\geq \frac{1}{|\mathcal{M}|}(1 - \Pr\left[\, c \in S \,\right]) + \frac{1}{|\mathcal{M}|}\Pr\left[\, c \in S \,\right]\left(1 - 2^{-\kappa}\right) \\
&= \frac{1}{|\mathcal{M}|}\left(1 - \Pr\left[\, c \in S \,\right] + \Pr\left[\, c \in S \,\right]\left(1 - 2^{-\kappa}\right)\right) \\
&= \frac{1}{|\mathcal{M}|}\left(1 - (2 - 2^{-\kappa})\Pr\left[\, c \in S \,\right]\right) \\
&\geq \frac{1}{|\mathcal{M}|}\left(1 - (2 - 2^{-\kappa})\left(1 - \frac{T}{2^{\sigma}}\right)\right)
\end{aligned}
$$

where, recall, σ is the min-entropy of the channel. Hence, in total for the le-ROD1 experiment, we have

$$
\Pr\left[\, \mathcal{A}^{H} \Rightarrow 1 \,\right] \geq \left(\frac{1}{2^{\kappa+\mu}} + \frac{1 - 2^{-\mu}}{|\mathcal{M}|}\right)\Pr\left[\, E \,\right] + \left(\frac{1}{|\mathcal{M}|}\left(1 - (2 - 2^{-\kappa})\left(1 - \frac{T}{2^{\sigma}}\right)\right)\right)\Pr\left[\, \neg E \,\right]
$$

and, using the fact that $\epsilon \leq \Pr\left[\, E \,\right] \leq 1$ ≪**TomS B.7:** Sadly, I don't know what to do with the $\Pr\left[\, \neg E \,\right]$ term, since I have to *lower*bound it. I'm forced to lowerbound it by zero, for lack of better ideas.≫

$$
\Pr\left[\, \mathcal{A}^{H} \Rightarrow 1 \,\right] \geq \left(\frac{1}{2^{\kappa+\mu}} + \frac{1 - 2^{-\mu}}{|\mathcal{M}|}\right)\epsilon
$$

Finally, using our observation above that in the ideal setting $\Pr\left[\, \mathcal{A}^{H} \Rightarrow 1 \,\right] = 1/|\mathcal{M}|$, we reach the conclusion that

$$
\mathbf{Adv}^{\text{le-rod}}(\mathcal{A}) \geq \left(\frac{1}{2^{\kappa+\mu}} + \frac{1 - 2^{-\mu}}{|\mathcal{M}|}\right)\epsilon - \frac{1}{|\mathcal{M}|}
$$

so as $\epsilon \to 1$, the bound goes to

$$
\mathbf{Adv}^{\text{le-rod}}(\mathcal{A}) \geq \frac{1}{2^{\mu}}\left(\frac{1}{2^{\kappa}} - \frac{1}{|\mathcal{M}|}\right)
$$

∎

$G0(\mathcal{A})$, $\boxed{G1(\mathcal{A})}$:

$K \leftarrow\!\!\$\ \mathsf{kg}$
$c \leftarrow\!\!\$\ \mathcal{C}_h$
$m \leftarrow\!\!\$\ \mathcal{M}$
$b \leftarrow\!\!\$\ \mathcal{A}^H(c)$
return b

Oracle $H(Z)$:
$L \,\|\, \langle N \rangle \,\|\, X \leftarrow Z$, where $|L| = |K|$
$Y \leftarrow\!\!\$\ \mathcal{M}$
$Y' \leftarrow\!\!\$\ \mathcal{M} \setminus \{Y\}$
if $L = K$ and $X = c$ then
  $Y \leftarrow m$
if $L = K$ and $X \neq c$ then
  if $Y = m$ then
    BAD $\leftarrow$ true
    $\boxed{Y \leftarrow Y'}$
return Y

$\boxed{G4(\mathcal{A})}$, $G5(\mathcal{A})$:

$K \leftarrow\!\!\$\ \mathsf{kg}$
$m \leftarrow\!\!\$\ \mathcal{M}$
$c \leftarrow\!\!\$\ \mathsf{enc}_K^G(m; (h, N))$
$b \leftarrow\!\!\$\ \mathcal{A}^H(c)$
return b

Oracle $G(Z)$:
$Y \leftarrow\!\!\$\ \mathcal{M}$
$K \,\|\, \langle N \rangle \,\|\, X \leftarrow Z$
if $\mathtt{H}[K, X] \neq \perp$ then
  $Y \leftarrow \mathtt{H}[L, X]$
$\mathtt{H}[L, X] \leftarrow Y$
return Y

Oracle $H(Z)$:
$L \,\|\, \langle N \rangle \,\|\, X \leftarrow Z$, where $|L| = |K|$
$Y \leftarrow\!\!\$\ \mathcal{M}$
$Y' \leftarrow\!\!\$\ \mathcal{M} \setminus \{Y\}$
if $L = K$ and $X = c$ then
  return $\mathtt{H}[K, c]$
if $L = K$ and $X \neq c$ then
  if $\mathtt{H}[K, X] \neq \perp$ then return $\mathtt{H}[K, X]$
  if $Y = m$ then
    BAD $\leftarrow$ true
    $\boxed{Y \leftarrow Y'}$
return Y

$G6(\mathcal{A})$:

Figure 13: Games for the proof of this theorem

Theorem 5 (Rejection-sampling encryption is le-ROD-secure for uniform messages.) If the keyspace has min-entropy κ and p_m is uniform over $\mathcal{M}$, then $\mathbf{Adv}^{\text{le-rod}}(\mathcal{A}) \leq 2q/2^{\kappa}|M|$

Proof: (Sketch.) Consider an adversary $\mathcal{A}^H(c)$, provided input c that was either produced by $\mathsf{enc}_K^H(m; (h, N))$ or by directly sampling $\mathcal{C}_h$. The preceding lemma tells us that without access to H, no adversary can distinguish the two cases. It remains to argue that oracle access to H does not significantly increase the ability to distinguish between these two possibilities. For simplicity, we silently assume that $\mathcal{A}$ also knows (h, N).

Consider the following experiment: $K \leftarrow\!\!\$\ \mathsf{kg}$, $c \leftarrow\!\!\$\ \mathcal{C}_h$, let m be sampled from $\mathcal{M}$ according to an arbitrary distribution, and then the adversary is given c. We ignore N when discussing $\mathcal{A}$'s queries, since we assume this is known by the adversary, and can therefore assume that it is implicitly part of each query it makes. Let $(L_1, X_1), \ldots, (L_q, X_q)$ be the sequence of the adversary's queries, these returning $Y_1, \ldots, Y_q$, respectively. Let $\mathcal{L} = \{(L_i, X_i, Y_i) \,|\, L_i = K\}$, i.e., the set of queries that were for the correct key. We say that $\mathcal{L}$ is *inconsistent with* c if there exists $(K, X, Y) \in \mathcal{L}$ such that $X \neq c$ and $Y = M$. Otherise, $\mathcal{L}$ is consistent with c.

Note that if $\mathcal{L}$ is consistent, then none of the adversaries queries are inconsistent with an encryption of m under key K. This is because either a query (L_i, X_i) was for $L_i \neq K$, in which case $H(L_i \,\|\, \langle N \rangle \,\|\, X_i)$ would never be called to encrypt m under K; or $H(L_i, X_i) = H(K, X_i) \neq m$, in which case X_i (if sampled) would not cause encryption to halt and return $X_i \neq c$. Thus, we say the set of queries/responses $\mathcal{Q} = \{(L_i, X_i, Y_i)\}_{i \in [q]}$ is consistent with c if $\mathcal{L} \subseteq \mathcal{Q}$ is consistent with c. ≪**TomS B.8:** I want to conclude that if $\mathcal{Q}$ is consistent with c, then the RO-queries give no help to $\mathcal{A}$ in distinguishing. Intuitively, it seems right. See the lemma, below.≫

Still cleaning up.

Consider the games in Figure 13. By inpsection $\Pr[\,G0(\mathcal{A}) = 1\,] = \Pr[\,\text{le-ROD0}(\mathcal{A}) = 1\,]$. Game $G1$, which

contains the boxed statement, differs from $G0$ in that it forces consistent query-response tuples, just after the setting of the BAD-flag. We have

$$\begin{aligned}\Pr[\,G1(\mathcal{A}) \Rightarrow 1\,] &\le \Pr[\,G0(\mathcal{A}) \Rightarrow 1\,] + \Pr[\,G0(\mathcal{A}) :\ \text{BAD}\,] \\ &= \Pr[\,\text{le-ROD0}(\mathcal{A}) = 1\,] + \Pr[\,G0(\mathcal{A}) :\ \text{BAD}\,]\end{aligned}$$

since $G0$ and $G1$ are identical-until-BAD. In $G0$ we observe that all H queries are responded to by uniformly sampled values from $\mathcal{M}$. (The values Y' are never used.) So, by a simple union bound, we have $\Pr[\,G0(\mathcal{A}) :\ \text{BAD}\,] \le q/2^{\kappa}|\mathcal{M}|$. ≪**TomS B.9:** I'd feel a bit better about this if we could push sampling of key until after $\mathcal{A}$ runs, but I don't see how to do that. Should be fine as is.≫

In game $G1$ it is clear that the set query-response tuples $\mathcal{Q}$ is consistent with c, as the game is constructed to force consistency. To continue, we appeal to the following lemma.

Lemma 4 *Let c and let $\mathcal{Q}$ be any set of query-response tuples that is consistent with c, in the way described at the beginning of this proof. Then, given $\mathcal{Q}$ and c as input (and no further oracle access), no adversary can distinguish between the cases $c \xleftarrow{\$} \mathcal{C}_h$ and $c \xleftarrow{\$} \mathsf{enc}_K(m;(h,N))$ for any message m.*

Proof: [lemma](Sketch.) Consider a ciphertext $c \xleftarrow{\$} \mathsf{enc}_K(m;(h,N))$. During encryption, zero or more queries of the form $K \,\|\, \langle N\rangle \,\|\, s_i$ were made, where the $s_0, s_1, \ldots, s_t$ (for some $t < T-1$) were independent channel samples. Note that $H(K \,\|\, \langle N\rangle \,\|\, s_i) \ne m$ for all of these s_i, because c was the first channel sample such that $H(K \,\|\, \langle N\rangle \,\|\, s_i) = m$. In fact, notice that *any* sequence of channel samples $s_0, s_1, \ldots, s_t, c$ (for $t < T-1$) such that $\forall i$: (1) $H(K \,\|\, \langle N\rangle \,\|\, s_i) \ne m$, and (2) $s_i \ne c$ would result in c being the ciphertext for m. Crucially, this is true no matter what is m.

Now, say that c was actually the result of sampling just once from the channel, as in the le-ROD0 game. If the adversary has asked queries that are consistent with c, then those queries explain equally well the case that c is a ciphertext of some message.≪**TomS B.10:** getting handwavy in here, but it's intuitive.≫ So these queries offer no help in distinguishing the le-ROD1 and le-ROD0 games. □

With this lemma, we move to game $G4$, which is a rewriting of $G1$ to allow for separate oracle "interfaces" H and G to the underlying random-oracle table $\mathtt{H}[]$. Thus, $\Pr[\,G1(\mathcal{A}) \Rightarrow 1\,] = \Pr[\,G4(\mathcal{A}) \Rightarrow 1\,]$. We note that in $G4$ the introduction of new code to check for $L = K, X \ne c$ and $\mathtt{H}[K,c]$ does not violate consistency. This is true because, if $\mathtt{H}[K,c] \ne \perp$, it must be that it was set during encryption by one of the rejected samples.

In $G4$, we also reintroduce a BAD-flag, to allow us to give an identical-until-BAD game, $G5$. As usual,

$$\Pr[\,G4(\mathcal{A}) \Rightarrow 1\,] \le \Pr[\,G5(\mathcal{A}) \Rightarrow 1\,] + \Pr[\,G5(A) :\ \text{BAD}\,]$$

and by the same argument used earlier, $\Pr[\,G5(A) :\ \text{BAD}\,] \le q/2^{\kappa}|\mathcal{M}|$.

By $G5$, we notice that the distribution of the plaintext m no longer matters, since the simulation of the random oracle H for $\mathcal{A}$ no longer depends on the distribution of m, only its particular value. (In fact, we could use a fixed string, independent of m for the test "if $Y = m$" that determines the setting of BAD.) Thus in $G6$ we simply change the distribution of m from uniform to that described by p_m. [...] At this point, we have $\Pr[\,\text{le-ROD1}(\mathcal{A}) = 1\,] = \Pr[\,G6(\mathcal{A}) \Rightarrow 1\,]$. Thus we conclude that

$$\mathbf{Adv}(\mathcal{A}) \le 2q/2^{\kappa}|\mathcal{M}|$$

given the previous lemma, which remains to be proved.