Fuzzy Message Detection

Gabrielle Beck
becgabri@cs.jhu.edu
Johns Hopkins University

Julia Len
jlen@cs.cornell.edu
Cornell University

Ian Miers
imiers@cs.umd.edu
University of Maryland

Matthew Green
mgreen@cs.jhu.edu
Johns Hopkins University

Abstract

Many privacy-preserving protocols employ a primitive that allows a sender to "flag" a message to a recipient's public key, such that only the recipient (who possesses the corresponding secret key) can detect that the message is intended for their use. Examples of such protocols include anonymous messaging, privacy-preserving payments, and anonymous tracing. A limitation of the existing techniques is that recipients cannot easily outsource the detection of messages to a remote server, without revealing to the server the exact set of matching messages. In this work we propose a new class of cryptographic primitives called *fuzzy message detection schemes*. These schemes allow a recipient to derive a specialized message detection key that can identify correct messages, while also incorrectly identifying non-matching messages with a specific and chosen false positive rate p. This allows recipients to outsource detection work to an untrustworthy server, without revealing precisely which messages belong to the receiver. We show how to construct these schemes under a variety of assumptions; describe several applications of the new technique; and show that our schemes are efficient enough to use in real applications.

1 Introduction

Many privacy-preserving systems require a means to transmit confidential messages from one party to another without revealing the identity of the sender or receiver, or any other information about the communication pattern. Such mechanisms are an essential ingredient of anonymous messaging systems [WCFJ12], privacy-preserving analytics [BEM$^+$17], and private digital payment systems [BCG$^+$14, NM$^+$16, mon]. We refer to these generally as *anonymous message delivery systems*.

Anonymous message delivery systems can be roughly divided into two classes: those in which senders and recipients are both *online* when messages are exchanged, and those in which some participants may be *offline*. The former class comprises applications such as anonymous web browsing [DMS04, i2p]. The latter includes secure messaging and payment systems, where recipients may be intermittently-connected and too numerous to be core participants in an online network protocol. As in legacy systems such as email, SMS and cryptocurrency, many systems support these offline users via a "store-and-forward" approach: messages are collected by potentially untrusted parties, so that they can be retrieved by the intended recipient at some later point [BCG$^+$14, mon, vdHLZZ15].

Achieving privacy in store-and-forward systems is challenging if one wishes to hide recipient meta-data: in addition to hiding the path over which messages are forwarded, systems must also hide both stored metadata (e.g. recipient identity) and access patterns to stored data. Many systems, particularly those based on mix-nets, do not do this: they assume a trusted inbox provider who is online and stores anonymous and encrypted messages for the recipient to download (possibly including some cover traffic). However, failure to protect recipient metadata can harm privacy even when the recipient uses an anonymous communications network to access the store [DMS04]. An attacker who observes that a specific message is retrieved, even by an anonymous recipient, can correlate that message with subsequent responses the recipient may transmit or others may receive. Such correlations can harm privacy. For example, this access pattern leakage might reveal that a journalist has contacted their editor after receiving an important message from an (unknown) source. Leakage of this form is particularly relevant to anonymous payment systems, where the ability to link received funds to outbound payments can directly reveal information about the payment graph.

Deployed anonymous message delivery systems attempt to mitigate access pattern leakage in several ways. When sender and recipient are capable of some form of advanced coordination, messages can be stored at a known location or under a predictable (*e.g.,* pseudo random) identifier. Then various approaches can be used to hide access patterns, e.g. differential privacy [vdHLZZ15] or private information retrieval [CKGS98, AS16]. But all of these approaches require coordination. Moreover, they introduce considerable overheads either in additional traffic or computational requirements (e.g. for PIR).

Public-key message detection. The problem becomes more challenging when advanced coordination between sender and recipient is not possible. This specifically includes the *public key* setting, where messages are transmitted from sender to receiver using only the recipient's public key — as is the case with many payment and messaging systems. One common pattern is to employ a public-key *message detection* scheme. The folklore realization of this primitive uses public-key encryption to encrypt a "flag" ciphertext, which contains a recognizable plaintext that can be detected by a recipient who possesses the appropriate secret ("detection") key. To detect incoming messages, the recipient performs trial decryptions on *every* ciphertext retrieved from the mailbox. Variants of this approach are used by many protocols and deployed systems [BCG+14, mon, BCOP04, LZ16].[1]

The trial decryption approach is very costly in terms of bandwidth, since the recipient must download all ciphertexts. This has real-world impact on systems in which users have mobile data connections [Lee19]. To reduce bandwidth, recipients can "outsource" decryption work to the mailbox system by providing it with a copy of the detection key (as is done in some cryptocurrency light wallets [Mon19]). Unfortunately, revealing this key leaks critical data to the server: namely, the exact set of messages that are addressed to a given recipient.

Fuzzy Message Detection. To reduce the privacy leakage of these outsourced detection schemes, we propose a new cryptographic primitive: *fuzzy message detection* (FMD). Like a standard message detection scheme, a fuzzy message detection scheme allows senders to encrypt flag ciphertexts that recipients can detect using a secret key. But our schemes support a second, "fuzzy" detection key which is used by a server for outsourced message detection. Unlike the user's main secret, the fuzzy detection key will also identify *non-matching* flag ciphertexts with some false positive rate. Crucially, the false positive rate is set by the recipient and can be adapted dynamically even after messages are sent. Critically, when applied to ciphertexts generated by honest users, *the untrusted server must be unable to distinguish between a correct detection result and a false-positive.* As a result of this, the server will be unable to determine which results are true matches, and which are accidental false positives that serve as "cover traffic" to disguise the recipient's true messages. This approach enables a form of dynamic k-anonymity in the public-key setting.

Our contributions. In this work we formalize the definition of FMD, and propose several viable constructions with different properties. As a warm-up, we start by presenting a simple proposal which is capable of supporting a restricted set of false-positive probabilities that have the form $p = 2^{-n}$ for non-negative integers n. This scheme is general, in the sense that it can be built using any CPA-secure public-key encryption scheme with specific properties. We then consider the more challenging problem of whether FMD can achieve chosen-ciphertext security, and answer this question in the affirmative by proposing an efficient CCA-secure construction. As a building block of independent interest, we formalize a type of public-key encryption that has specific behavior when ciphertexts are decrypted under "wrong" secret keys; we call this notion *ambiguous encryption* (AMB-PKE). Our constructions of this primitive can be realized in standard cyclic groups where the CDH assumption holds.

We next turn to the much more challenging problem of designing FMD schemes that can operate on more fine-grained false positive rates of the form $p \equiv \frac{M}{N}$ for integers M, N. This requires us to investigate a new notion of encryption that we call *ambiguous functional encryption.* To show that our approach is

[1]Alpenhorn uses other features to hide public key lookups and sender's identity, but for metadata protection it simply buckets messages and does trial detection for the recipient.

practical, we instantiate such a fractional scheme using a form of bounded functional encryption realized from a customized garbled circuit construction.

We perform microbenchmarks of two of our constructions. For our CCA-secure FMD scheme with restricted probabilities of the form $p = 2^{-n}$ up to $n = 24$, flag ciphertexts are 68 bytes in size and require 1.927 ms to generate, while testing a match requires only 0.548 ms given a detection key with false positive probability of 3%. For our more powerful fractional FMD scheme, ciphertexts are much larger; generation takes 18.061 ms and testing a ciphertext requires 9.585 ms.

These benchmarks suggest that our FMD schemes are efficient enough for a number of applications. Most directly, it can be combined with an online anonymous networking protocol like Tor to provide anonymous store and forward messaging, which enables applications such as private payment detection in cryptocurrencies. It can also provide a service to bootstrap a shared secret necessary for communication in other protocols such as [AS16, vdHLZZ15] and be used in protocols such as Apple's FindMy for device tracking [fin].

To summarize, in this work we make the following contributions:

1. We introduce the notion of a *fuzzy message detection* (FMD) scheme to address the challenge of efficient message detection in privacy-preserving store-and-forward systems. We formalize its correctness and security under both CPA and CCA attacks.
2. We provide several constructions for FMD schemes, including an efficient scheme that supports restricted probability as well as one that supports fine-grained probabilities. To build these schemes, we introduce ambiguous encryption and ambiguous functional encryption, both of which could be of independent interest.
3. We implement our efficient restricted probability construction and demonstrate its feasibility by giving microbenchmarks.

2 Intuition

Intuitively, a fuzzy message detection (FMD) scheme is reminiscent of public key encryption but with some crucial modifications. Key generation works as expected, but encryption is replaced by a randomized Flag algorithm that, on input a public key (and no plaintext[2]) generates a *flag ciphertext.* Decryption is similarly replaced by a Test algorithm that, on input a ciphertext and *detection key*, outputs whether a match occurred. To generate the detection key, the scheme incorporates a new algorithm $\mathsf{Extract}$ that takes as input the scheme's "master" secret key *sk* and some intended false positive rate p.

The essential properties of an FMD scheme are threefold. First, given a detection key and any *matching ciphertext* (*i.e.,* one encrypted with the corresponding public key), the Test algorithm should always indicate a match. At the same time, for an honestly-generated ciphertext that was encrypted under a different public key, Test should indicate a false-positive match with probability approximately p, taken over the random coins that were used to generate the ciphertext but chosen dynamically by the recipient even after a message is encrypted. For privacy, we require a security property that we refer to as *detection ambiguity*: this holds that an adversarial server who is given honestly-generated ciphertext(s) and a detection key must be unable to distinguish between true and false-positive matches.

Detection ambiguity makes FMD more challenging to achieve than traditional security notions of public key encryption, in which adversaries obtain only *public* keys, but do not receive secret key material.[3] As a final note, we stress that in our formulation, we do not attempt to hide the value of p from the adversary: indeed, this would be challenging in the public-key setting, as the adversary can estimate this value statistically by testing its own ciphertexts against a given detection key.

Ambiguous encryption. A major consideration in our work is the behavior of public-key encryption when ciphertexts are decrypted with *incorrect secret keys.* While this has been considered, relatively little work has

[2]FMD schemes can also be modified to carry message data, but this can also be achieved by simply encrypting the necessary data into a separate public-key encryption ciphertext. For generality we leave this mode out of our description.

[3]In the literature, one of the closer related notions is that of *deniable encryption* [CDNO97] which is used to achieve adaptive security in protocols. *Blind public key encryption* [BLSV18] also achieves indistinguishability notions in the presence of secret key material and is known to imply - in conjunction with one way functions - blind Identity-Based Encryption.

been given to formalizing such behavior. As a building block for our techniques, we must therefore formalize a new cryptographic primitive: *ambiguous encryption* (AMB-PKE). In addition to the standard ciphertext indistinguishability and key-privacy notions, ambiguous encryption must satisfy an additional property that we call *key ambiguity*. A scheme achieves this notion with respect to some plaintext distribution $\mathcal{D}$ if the following condition holds: given a set of honestly generated key pairs (including both public *and* secret keys), as well as the encryption of an unknown plaintext sampled from $\mathcal{D}$, no adversary can determine which public key the ciphertext was encrypted with (except with negligible advantage). Our FMD constructions in this work make use of single-bit and multi-bit ambiguous encryption schemes for the specific case of the *uniform distribution* for bit encryption and larger plaintext domains, and we show how to construct these using standard assumptions such as CDH.

FMD with restricted false-positive rates. As a first result, we show how to construct efficient FMD that supports restricted values of p of the form $p = 2^{-n}$ for integers $0 \leq n \leq \gamma$ where γ is a constant shared by all participants. The main ingredient in our construction is a uniformly-ambiguous bit encryption scheme.

Our construction leverages a natural property of uniformly-ambiguous encryption: namely, that decrypting an honestly-generated ciphertext with the "wrong key" produces a random plaintext. In our simplest FMD construction, each recipient generates a collection of independent key pairs for the underlying ambiguous bit encryption scheme, and outputs the "public key" $pk^{\mathsf{FMD}} = (pk_1^{\mathsf{Amb}}, \ldots, pk_\gamma^{\mathsf{Amb}})$. To encrypt a flag ciphertext, the encryptor simply encrypts the plaintext bit "1" using each public key in the vector, and concatenates the resulting ciphertexts. The detection key for a specific probability $p = 2^{-n}$ comprises a subset of n underlying secret keys for the ambiguous encryption scheme: $dsk \leftarrow (sk_1^{\mathsf{Amb}}, \ldots, sk_n^{\mathsf{Amb}})$. For $i \in \{1, \cdots, n\}$, the Test algorithm attempts to decrypt each ciphertext C_i with the corresponding secret key at index i, and outputs 1 if all decrypt to 1. The core of our analysis in later sections is to show that both detection ambiguity and correctness flow directly from the security properties of ambiguous encryption.

While this basic construction is helpful as an intuition, using it as described above has some drawbacks. Implementing the scheme naively, even with an efficient underlying ambiguous encryption scheme, will produce relatively large flag ciphertexts. Moreover, the approach described above does not offer security against chosen ciphertext attacks.[4] In §5.2 we remedy these issues by presenting an efficient CCA-secure variant of our scheme that has a ciphertext size comparable to standard (hash) ElGamal encryption, with a security analysis in the random oracle model. As a further optimization, we discuss possible extensions that achieve time-based revocation of detection keys, using techniques from the field of Identity-Based Encryption [Sha84, BF01].

FMD with adaptive, fine-grained probabilities. While the restricted scheme above is quite efficient, it suffers from a limited available selection of false-positive rates $p = 2^{-n}$ for integer values of n. Moreover, these restrictions seem fundamentally challenging to remove. This raises a question: is it possible to define FMD in which the receiver can select more precise values for the rate p? Such adjustments might be necessary in order to adjust for variations in message traffic at the time of retrieval. As our second contribution, we answer this question in the affirmative. Concretely, we propose a construction that supports fine-grained values of p of the form $p \approx \frac{M}{N}$ for arbitrary integers $0 \leq M \leq N, N > 0$.

Supporting fine grained false-positive rates while retaining the security of an FMD scheme is non-trivial because the false-positive rate p is not known to the encryptor at the time it generates a flag ciphertext; indeed it may change after encryption.[5] To illustrate the challenge, consider a strawman FMD construction that works in the counterfactual situation where the rate p (and hence M, N) are fixed for all parties and are known to the encryptor. Our construction assumes a uniformly-ambiguous encryption scheme with the specific plaintext domain $\mathcal{M} = [N]$, and it works as follows:

1. To generate a flag ciphertext, the encryptor samples a plaintext m uniformly from $[M]$ and encrypts it under the receiver's public key.

[4]Solving the latter problem in particular is non-trivial, given that many common techniques for constructing CCA-secure public key encryption are incompatible with the ambiguity properties required from the underlying encryption. In particular, any CCA-secure public key encryption scheme that provides secret-key dependent checks is unlikely to be ambiguous, as decryption with the incorrect key will typically result in a decryption failure.

[5]Adaptive selection of p means that the receiver may select this value after seeing the ciphertext.

2. The receiver sets its detection key *dsk* equal to the secret key for the ambiguous encryption scheme.
3. To test whether a ciphertext matches, the server decrypts with *dsk* to obtain m', and outputs TRUE if $m' \in [M]$.

Clearly, for a *matched* ciphertext, the test condition will always be satisfied. For a non-matched ciphertext, it remains only to argue that the decrypted result must be approximately uniform in $[N]$ and hence will satisfy the test equation with probability $p \approx M/N$. This is a specific property guaranteed by any ambiguous encryption scheme, as we discuss in later sections. The problem with this approach is that it is difficult to adapt to our definition of FMD, where the probability p used in the detection key *can be chosen by the recipient*, quite possibly at a time subsequent to encryption. One can address part of this problem by fixing a large denominator N for all users in the system; however, the numerator M may still be different for each recipient. This prevents the encryptor from sampling a plaintext in the correct range and hence we cannot ensure that matched ciphertexts will always evaluate correctly. Solving this is challenging. Since M is not known at encryption time, plaintext sampling must somehow be deferred until after encryption has occurred.

A key observation of our work is that this problem can be addressed using functional encryption [BSW11] with novel properties. In a functional encryption scheme for a function f, the sender encrypts some input x under a master public key. Given a second value y, the holder of the corresponding trapdoor can extract a secret key sk_y. An untrusted third party can now combine the ciphertext and secret key to obtain $f(x, y)$. If we allow each receiver to generate its own public parameters for an instance of a functional encryption scheme, and define the function f to be a sampling function — where x represents some set of random coins chosen by the encryptor, and y represents the range M — then functional encryption achieves the correctness goal we desire. (Moreover, in a setting where each receiver generates a single detection key per epoch, we require functional encryption that survives only bounded collusion [SS10].) The remaining challenge is therefore to ensure that the resulting scheme is *ambiguous*, *i.e.,* to ensure that an adversary cannot distinguish a matched ciphertext from a mis-matched one that happens to satisfy the test conditions. This is not a property that is envisioned by traditional functional encryption definitions, and this requires us to develop new techniques.

In §6 we show how to achieve this goal. Our approach uses a new variant of Yao's classical garbling scheme to construct a special form of bounded function encryption. The novel contribution we offer is to show that this approach can be made to provide *ambiguity* when the garbling process is slightly modified, and combined with an ambiguous public-key encryption scheme. Achieving this is non-trivial: we must show, among other things, how to evaluate a garbled circuit with possibly-incorrect labels, without revealing this condition to the evaluator. While our scheme based on this construction yields larger ciphertexts than the restricted scheme above, it provides a demonstration that our approach is viable. We leave the construction of more efficient fractional FMD schemes as an open problem for future work.

3 Preliminaries

In this section we provide notation that will be used throughout the paper and recall primitives specified by prior work.

Notation. Let λ be an adjustable security parameter, let poly$(\cdot)$ be a polynomial function, and let $\nu(\cdot)$ be a negligible function. We will use $\mathcal{M}$ to represent the plaintext domain of an encryption scheme, and $\mathcal{C}$ for the ciphertext domain. For some probability distribution $\mathcal{D}$ over $\mathcal{M}$ we will use the notation $m \overset{\mathcal{D}}{\leftarrow} \mathcal{M}$ to denote sampling an element according to the distribution $\mathcal{D}$. If s is a bitstring, then we define $s[i]$ to be the i^{th} bit of s. Finally, we use $\overset{c}{\approx}$ to denote computational indistinguishability between two distributions.

Cyclic groups and group generation. Some constructions below require a cyclic group $\mathbb{G}$ where the computational Diffie-Hellman (CDH) problem is assumed to be hard. Our schemes will make use of a setup algorithm that generates a common cyclic group $\langle g \rangle = \mathbb{G}$ of prime order q. In practical implementations, this setup algorithm can be instantiated by employing standard cryptographic groups, or by sampling groups using a random oracle and a short seed. For this reason and for simplicity of notation we will omit this setup algorithm from many of our experiments.

3.1 Public Key Encryption

A public key encryption scheme PKE is a tuple of polynomial time algorithms $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ with the following interface:

$\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$: a probabilistic algorithm that takes in the security parameter λ and produces a public and private key.

$\mathsf{Enc}(pk, m) \to c$: a probabilistic algorithm that uses the public key pk to encrypt a message m and outputs a ciphertext c.

$\mathsf{Dec}(sk, c) \to m$ or $\perp$: a deterministic algorithm that takes in a secret key sk and a ciphertext c and outputs the plaintext message m or $\perp$ if decryption fails.

Correctness. We require perfect correctness for correct decryption such that the following holds for every message $m \in \mathcal{M}$ where $\mathcal{M}$ is the plaintext domain of the encryption scheme and security parameter λ:

$$\Pr\begin{bmatrix}(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda); \\ c \leftarrow \mathsf{Enc}(pk, m) : \mathsf{Dec}(sk, c) = m\end{bmatrix} = 1$$

Security. We use the common indistinguishability definition which requires that the adversary cannot, in laymen's terms, learn anything about the encrypted message beyond the length, provided they have access to only the public key and (potentially) an oracle $\mathcal{O}$.

Definition 1 (Indistinguishability)**.** Let $\mathsf{PKE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be a public key encryption scheme and let the random variable $\mathsf{IND\text{-}ATK}_b(\mathsf{PKE}, \mathcal{A}, \lambda)$ where $\mathcal{A} = \{\mathcal{A}_1, \mathcal{A}_2\}$, $b \in \{0, 1\}$, $\mathsf{ATK} \in \{\mathsf{CPA}, \mathsf{CCA}\}$, and $\lambda \in \mathbb{N}$ denote the result of the following probabilistic experiment:

```
$\underline{\mathsf{IND\text{-}ATK}_b(\mathsf{PKE}, \mathcal{A}, \lambda):}$
    $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$
    $(st, m_0, m_1) \leftarrow \mathcal{A}_1^{\mathcal{O}_{sk}(\cdot)}(pk)$
    $c^* \leftarrow \mathsf{Enc}(pk, m_b)$
    $B \leftarrow \mathcal{A}_2^{\mathcal{O}_{sk}(\cdot)}(st, c^*)$
    Output $B$
```

When $\mathsf{ATK} = \mathsf{CPA}$, $\mathcal{O}$ returns $\perp$ on all inputs. For $\mathsf{ATK} = \mathsf{CCA}$, the query $\mathcal{O}(c)$ returns $\perp$ if $c = c^*$. Otherwise, the oracle returns $m = \mathsf{Dec}(sk, c)$. The $\mathsf{IND\text{-}ATK}(\mathsf{PKE}, \mathcal{A}, \lambda)$ advantage of $\mathcal{A}$ is defined as

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{IND\text{-}ATK}}(\lambda) = \left|\mathbf{Pr}\left[\mathsf{IND\text{-}ATK}_0(\mathsf{PKE}, \mathcal{A}, \lambda) \Rightarrow 1\right] - \mathbf{Pr}\left[\mathsf{IND\text{-}ATK}_1(\mathsf{PKE}, \mathcal{A}, \lambda) \Rightarrow 1\right]\right|.$$

.

3.2 Key Privacy

Key privacy was first introduced by Bellare, Boldyreva, Desai and Pointcheval [BBDP01]. The definition covers a weak anonymity property that encryption schemes can have, whereby an adversary with only access to public keys and a decryption oracle cannot tell which key was used to generate a ciphertext.

Definition 2 (Key Privacy)**.** Let $\mathsf{PKE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be a public key encryption scheme and let the random variable $\mathsf{IK\text{-}ATK}_b(\mathsf{PKE}, \mathcal{A}, \lambda)$ where $\mathcal{A} = \{\mathcal{A}_1, \mathcal{A}_2\}$, $b \in \{0, 1\}$, $\mathsf{ATK} \in \{\mathsf{CPA}, \mathsf{CCA}\}$, and $\lambda \in \mathbb{N}$ denote the result of the following probabilistic experiment:

```
$\underline{\mathsf{IK\text{-}ATK}_b(\mathsf{PKE}, \mathcal{A}, \lambda):}$
    $(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(1^\lambda)$, $\forall i \in \{0, 1\}$
    $(st, m) \leftarrow \mathcal{A}_1^{\mathcal{O}_{sk_0, sk_1}(\cdot,\cdot)}(pk_0, pk_1)$
    $c^* \leftarrow \mathsf{Enc}(pk_b, m)$
    $B \leftarrow \mathcal{A}_2^{\mathcal{O}_{sk_0, sk_1}(\cdot,\cdot)}(st, c^*)$
    Output $B$
```

When $\mathsf{ATK} = \mathsf{CPA}$, $\mathcal{O}$ returns $\perp$ on all inputs. For $\mathsf{ATK} = \mathsf{CCA}$, the query $\mathcal{O}(id, c)$ returns $\perp$ if $c = c^*$. Otherwise, the oracle returns $m = \mathsf{Dec}(sk_{id}, c)$. The $\mathsf{IK\text{-}ATK}(\mathsf{PKE}, \mathcal{A}, \lambda)$ advantage of $\mathcal{A}$ is defined as

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{IK\text{-}ATK}}(\lambda) = \left| \mathbf{Pr}\left[\mathsf{IK\text{-}ATK}_0(\mathsf{PKE}, \mathcal{A}, \lambda) \Rightarrow 1\right] - \mathbf{Pr}\left[\mathsf{IK\text{-}ATK}_1(\mathsf{PKE}, \mathcal{A}, \lambda) \Rightarrow 1\right] \right|.$$

.

4 Definitions

We now formally define what a fuzzy message detection scheme is and introduce ambiguous encryption, another cryptographic primitive which will help us achieve FMD.

4.1 Fuzzy Message Detection

A *Fuzzy Message Detection* (FMD) scheme is a tuple of possibly probabilistic polynomial-time algorithms $(\mathsf{KeyGen}, \mathsf{Flag}, \mathsf{Extract}, \mathsf{Test})$ and an optional Setup algorithm, all associated with a set $\mathcal{P}$. These have the following profile:

$\mathsf{Setup}_{\mathsf{pp}}(1^\lambda) \rightarrow \mathsf{pp}$: This optional algorithm generates global parameters pp.

$\mathsf{KeyGen}_{\mathsf{pp}}(1^\lambda) \rightarrow (pk, sk)$: On input a security parameter λ and optionally a set of global parameters pp, outputs a public and secret key.

$\mathsf{Flag}(pk) \rightarrow C$: On input a public key, this randomized algorithm outputs a flag ciphertext C.

$\mathsf{Extract}(sk, p) \rightarrow dsk$: On input a secret key sk and a false positive rate $p \in \mathcal{P}$, this algorithm extracts a detection key dsk, or outputs $\perp$ if $p \notin \mathcal{P}$.

$\mathsf{Test}(dsk, C) \rightarrow \{0, 1\}$: The test routine, on input a detection key and a ciphertext, outputs a detection result.

Our constructions may require common public parameters, such as the description of a shared cyclic group. In this setting, we implicitly define an algorithm Setup that, on input a security parameter, generates these common parameters for use in the key generation algorithm. Finally, the finite set $\mathcal{P}$ is defined as the set of distinct false-positive rates $p \in [0, 1]$ that are supported by the scheme.

We require that an FMD scheme satisfy several properties, which we refer to as correctness, *fuzziness* and *detection ambiguity.* The latter property which ensures that an adversary cannot distinguish true matches from false positive matches.

Correctness. Intuitively, an FMD scheme is *correct* if valid matches always satisfy the Test algorithm. More formally, for all λ and for all $p \in \mathcal{P}$ the following equality must hold:

$$\Pr\left[\begin{matrix}(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda); dsk \leftarrow \mathsf{Extract}(sk, p); \\ C \leftarrow \mathsf{Flag}(pk) : \mathsf{Test}(dsk, C) = 1\end{matrix}\right] = 1$$

Fuzziness. This property, presented in the following definition, enforces that *invalid* matches should produce false positives with probability approximately p, taken over the random coins used in encryption and key generation. Critically, this holds under the assumption that both ciphertext and detection key were generated honestly.

Definition 3 (Fuzziness)**.** Let $\Pi = (\mathsf{KeyGen}, \mathsf{Flag}, \mathsf{Extract}, \mathsf{Test})$ be an FMD scheme with the allowable probability set $\mathcal{P}$. We say that Π is *fuzzy* if there exists some n and a negligible function $\nu(\cdot)$ such that for all $p \in \mathcal{P}$ and $\lambda > n$:

$$\left| \Pr\left[\begin{matrix}(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda); \\ (pk', sk') \leftarrow \mathsf{KeyGen}(1^\lambda); \\ dsk \leftarrow \mathsf{Extract}(sk, p); \\ C \leftarrow \mathsf{Flag}(pk') : \mathsf{Test}(dsk, C) = 1\end{matrix}\right] - p \right| < \nu(\lambda)$$

Security. Security for an FMD scheme requires that an adversarial server must be unable to distinguish between a true positive and a false positive, provided that ciphertexts and detection keys are honestly generated. We refer to this property as *detection ambiguity*, and we define a corresponding experiment under different attack variants, including a chosen plaintext and chosen-ciphertext variant.

Definition 4 (Detection ambiguity)**.** Let $\Pi = (\mathsf{KeyGen}, \mathsf{Flag}, \mathsf{Extract}, \mathsf{Test})$ be an FMD scheme supporting the allowable probability set $\mathcal{P}$, and let the random variable $\mathsf{DA\text{-}ATK}_b(\Pi, \mathcal{A}, \lambda)$ where $\mathcal{A} = \{\mathcal{A}_1, \mathcal{A}_2\}$, $b \in \{0, 1\}$, $\mathsf{ATK} \in \{\mathsf{CPA}, \mathsf{CCA}\}$, and $\lambda \in \mathbb{N}$ denote the result of the following probabilistic experiment:

```
$\underline{\mathsf{DA\text{-}ATK}_b(\Pi, \mathcal{A}, \lambda)}$:
  $(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(1^\lambda)$, $\forall i \in \{0, 1\}$
  $C^* \leftarrow \mathsf{Flag}(pk_b)$
  $(z, p) \leftarrow \mathcal{A}_1(pk_0, pk_1, C^*)$
  $dsk_i \leftarrow \mathsf{Extract}(sk_i, p)$, $\forall i \in \{0, 1\}$
  if $\mathsf{Test}(dsk_0, C^*) = \mathsf{Test}(dsk_1, C^*)$ and $p \in \mathcal{P}$, then:
    $B \leftarrow \mathcal{A}_2^{\mathcal{O}_{sk_0, sk_1}(\cdot,\cdot,\cdot)}(pk_0, pk_1, dsk_0, dsk_1, C^*, z)$
  else:
    $B \xleftarrow{\$} \{0, 1\}$
  Output $B$
```

When $\mathsf{ATK} = \mathsf{CPA}$, $\mathcal{O}$ returns $\perp$ on all inputs. For $\mathsf{ATK} = \mathsf{CCA}$, the query $\mathcal{O}(p', id, C)$ returns $\perp$ if $C = C^*$, $p' \notin \mathcal{P}$ or $id \notin \{0, 1\}$. Otherwise, the oracle computes $dsk \leftarrow \mathsf{Extract}(sk_{id}, p')$ and returns $\mathsf{Test}(dsk, C)$.

The $\mathsf{DA\text{-}ATK}(\Pi, \mathcal{A}, \lambda)$ advantage of $\mathcal{A}$ is defined as

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{DA\text{-}ATK}}(\lambda) = \left| \mathbf{Pr}\left[\mathsf{DA\text{-}ATK}_0(\Pi, \mathcal{A}, \lambda) \Rightarrow 1\right] - \mathbf{Pr}\left[\mathsf{DA\text{-}ATK}_1(\Pi, \mathcal{A}, \lambda) \Rightarrow 1\right] \right|.$$

A fuzzy message detection scheme Π is $\mathsf{DA\text{-}ATK}$-secure if $\forall$ n.u.p.p.t. algorithms $\mathcal{A}$ and all $p \in \mathcal{P}$, $\mathbf{Adv}_{\mathcal{A}}^{\mathsf{DA\text{-}ATK}}(\lambda)$ is negligible.

Selective Security. Throughout the text when we talk of security against a *selective* variant, we limit the adversary to specifying p before receiving the public keys pk_0, pk_1. For schemes with a list of allowed probabilities $\mathcal{P}$ that is of size polynomial in λ, selective implies adaptive with a corresponding security reduction loss (see Appendix H).

Discussion and relationship to other security notions. The purpose of this definition is to ensure that no adversary can distinguish whether a given result of the test algorithm stems from a true positive or a false one. To prevent a trivial "win" condition for the adversary, we run the adversary only on keys and ciphertexts where the output of Test is identical on both keys.

At first glance, this definition seems similar to *key privacy* ($\mathsf{IK\text{-}ATK}$) [BBDP01], since both definitions assume an attacker who must determine which public key a ciphertext was encrypted with. Indeed, some form of key privacy will be necessary in our FMD schemes in order to ensure privacy in cases where the server has not (yet) received a detection key for a given recipient. Thus it is reasonable to ask whether detection ambiguity implies a form of key privacy for FMD ciphertexts.

The answer to this question is conditional. In standard key privacy definitions, the attacker receives a challenge ciphertext C^* encrypted under one of two public keys. However, those definitions do not include any notion of a detection secret key, and hence omit our restriction that $\mathsf{Test}(dsk_0, C^*) = \mathsf{Test}(dsk_1, C^*)$. Because our definition has this stronger restriction on the distribution of ciphertexts, it does not seem easy to prove that detection-ambiguity implies key privacy for all possible FMD schemes.

However, we note that in the specific case of schemes that support $1 \in \mathcal{P}$ (*i.e.,* the scheme supports a setting where the false-positive rate for detection keys can be increased to 100%), the fuzziness property of the scheme ensures that the the relation $\mathsf{Test}(dsk_0, C^*) = \mathsf{Test}(dsk_1, C^*)$ will be satisfied with probability negligibly close to 1. It is easy to see that if there exists an attacker who succeeds in distinguishing the two ciphertext distributions *without* access to the detection keys, then there must exist an attacker who succeeds when given access to those keys. This implies that in schemes where $1 \in \mathcal{P}$, the detection-ambiguity property implies key privacy. We formalize this below. For convenience in subsequent discussions, we will assume that all constructions must support $p = 1$ as an allowable probability, and thus we do need not consider key privacy as an additional security property.

Theorem 5. Let $\Pi = (\mathsf{KeyGen}, \mathsf{Flag}, \mathsf{Extract}, \mathsf{Test})$ be an FMD scheme that satisfies correctness, fuzziness, and $\mathsf{DA\text{-}ATK}$ for $\mathsf{ATK} \in \{\mathsf{CPA}, \mathsf{CCA}\}$ with the condition that $1 \in \mathcal{P}$. Then Π provides a modified variant of key privacy for FMD schemes ($\mathsf{IK\text{-}ATK\text{-}FMD}$).[6]

A proof of Theorem 5 is sketched in Appendix A.

4.2 Ambiguous Encryption

Looking ahead, in order to build secure FMD schemes it will be helpful for us to develop public key encryption schemes $\mathsf{PKE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ that satisfy a disassociation between a ciphertext and the public/private key pair it was created under. While related notions, such as key privacy ($\mathsf{IK\text{-}ATK}$) [BBDP01], consider attackers who have access to public keys, here we are interested in a different notion in which the adversary has access to secret key material. We refer to this new formulation as *ambiguous encryption* (AMB-PKE).

Correctness and Indistinguishability. An ambiguous encryption scheme must satisfy the standard correctness property of a public key encryption scheme. For a specific set of messages $\mathcal{M}$, all λ and all $m \in \mathcal{M}$:

[6]Specifically, we modify the $\mathsf{IK\text{-}ATK}$ game to replace encryption with the Flag algorithm, and decryption with the Test algorithm. The adversary does not specify a plaintext to encrypt. We present the details in Appendix A.

$$\Pr\begin{bmatrix}(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda); \\ C \leftarrow \mathsf{Enc}(pk, m) : \mathsf{Dec}(sk, C) = m\end{bmatrix} = 1$$

We further require that an ambiguous encryption scheme must satisfy (at least) the standard IND-CPA and IK-CPA [BBDP01] security notions.

The novel requirement is the additional property of *key ambiguity*, which is defined with respect to a specific probability distribution $\mathcal{D}$ over the plaintext space of the scheme. Intuitively, the ambiguity property requires that an adversary cannot distinguish which public key a ciphertext was generated under, *even when the adversary is given all secret keys.* Clearly this will not hold for all possible plaintext distributions. Our definitions therefore require that the plaintext is sampled according to the distribution $\mathcal{D}$. A scheme is considered ambiguous with respect to a specific distribution $\mathcal{D}$ if the following definition holds.

Definition 6 ($\mathcal{D}$-AMB)**.** Let $\mathsf{PKE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be a public-key encryption scheme for the message space $\mathcal{M}$ and let the random variable $\mathcal{D}\text{-}\mathsf{AMB}_b(\mathsf{PKE}, \mathcal{A}, \mathcal{D}, \lambda)$ where $b \in \{0,1\}$, $\mathcal{D}$ is a probability distribution over $\mathcal{M}$, and $\lambda \in \mathbb{N}$, denote the result of the following probabilistic experiment:

```
$\underline{\mathcal{D}\text{-}\mathsf{AMB}_b(\mathsf{PKE}, \mathcal{A}, \mathcal{D}, \lambda):}$
    $(pk_i, sk_i) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda), \forall i \in \{0,1\}$
    $b \xleftarrow{\$} \{0,1\}$
    $m \xleftarrow{\mathcal{D}} \mathcal{M}$
    $C^* \leftarrow \mathsf{PKE.Enc}(pk_b, m)$
    $B \leftarrow \mathcal{A}(\mathcal{D}, pk_0, pk_1, sk_0, sk_1, C^*)$
    Output $B$
```

The $\mathcal{D}\text{-}\mathsf{AMB}(\mathsf{PKE}, \mathcal{A}, \mathcal{D}, \lambda)$ advantage of $\mathcal{A}$ is defined as

$$\mathbf{Adv}_{\mathcal{A}}^{\mathcal{D}\text{-}\mathsf{AMB}}(\lambda) = \left| \mathbf{Pr}\left[\mathcal{D}\text{-}\mathsf{AMB}_0(\mathsf{PKE}, \mathcal{A}, \mathcal{D}, \lambda) \Rightarrow 1\right] - \mathbf{Pr}\left[\mathcal{D}\text{-}\mathsf{AMB}_1(\mathsf{PKE}, \mathcal{A}, \mathcal{D}, \lambda) \Rightarrow 1\right] \right|.$$

Encryption scheme PKE is $\mathcal{D}$-AMB secure if $\forall$ n.u.p.p.t. algorithms $\mathcal{A}$, $\mathbf{Adv}_{\mathcal{A}}^{\mathcal{D}\text{-}\mathsf{AMB}}(\lambda)$ is negligible.

Definition 7. A scheme PKE is a $\mathcal{D}$-ambiguous public key encryption scheme secure if it is (1) correct, (2) satisfies the IND-CPA property, (3) satisfies the IK-CPA property [BBDP01], and (4) is $\mathcal{D}$-AMB secure.

Discussion. Note that the exact properties of the scheme depend on the nature of the distribution $\mathcal{D}$. In the remainder of this work, we will focus primarily on the *uniform distribution* over schemes with various plaintext domains $\mathcal{M}$. We refer to these as *uniformly ambiguous* encryption schemes. We note that an exploration of how to achieve ambiguous encryption for alternative distributions may be valuable for other applications than ours, though we leave this to future work.

Finally, we present a very useful theorem that applies to encryption schemes which are $\mathcal{D}$-ambiguous. This theorem shows that, regardless of which message $m \in \mathcal{M}$ is encrypted to obtain a ciphertext C, decryption with an *incorrect* key produces a distribution that is computationally indistinguishable from the distribution $\mathcal{D}$. Note that this is *not* a theorem on correct decryption which obviously has the same distribution as the input plaintext but a general theorem on the output of incorrect decryption.

```
KeyGen($1^\lambda$) :
    $a \leftarrow \mathbb{Z}_q$
    $pk \leftarrow g^a$
    $sk \leftarrow a$
    Return $pk, sk$
```

```
Enc($pk, m$) :
    $k \leftarrow \mathbb{Z}_q$
    Return $(H_\ell(g^k, pk^k) \oplus m, g^k)$
```

```
Dec($sk, (c_1, c_2)$) :
    $x \leftarrow c_2^{sk}$
    Return $H_\ell(c_2, x) \oplus c_1$
```

Figure 1: H-ELG, a public-key encryption scheme that is uniformly-ambiguous. Here g is the generator of a cyclic group $\mathbb{G}$ of prime order q, which is generated by a common setup procedure (not shown). The hash function $H_\ell : \mathbb{G} \times \mathbb{G} \to \{0,1\}^\ell$ is a random oracle with an ℓ-bit output. (ℓ is variable and depends on the needed output size of the encryption scheme)

Theorem 8 (Decryption with an incorrect key)**.** Let PKE be a $\mathcal{D}$-ambiguous public key encryption scheme secure under CPA in the sense of Definition 7. Then $\forall m \in \mathcal{M}$ and sufficiently large λ, the output of the following experiment is computationally indistinguishable from the distribution $\mathcal{D}$ over $\mathcal{M}$:

```
ExpMsgIncorrect(PKE, $\lambda$):
    $(pk_i, sk_i) \leftarrow$ PKE.KeyGen($1^\lambda$), $\forall i \in \{0,1\}$
    $C \leftarrow$ PKE.Enc($pk_0, m$)
    Output PKE.Dec($sk_1, C$)
```

This theorem holds as a combined result of the IND-CPA security and ambiguity properties of the scheme. First, we show that the outputs of incorrect decryption on any two plaintext distributions must be indistinguishable, because if they were not an IND-CPA attacker would then be able to distinguish with non negligible advantage. Therefore, every plaintext distribution behaves like $\mathcal{D}$ when looking at the output of incorrect decryption and thus by ambiguity has a distribution according to $\mathcal{D}$. We give a proof of this theorem in Appendix I.

4.3 Realizing Ambiguous Encryption

Our constructions of FMD require the existence of secure encryption schemes that satisfy the ambiguity property for the *uniform* distribution over some plaintext domain $\mathcal{M}$. It is thus reasonable to ask whether or not such schemes actually exist.

Uniformly ambiguous encryption from CDH We answer the above question in the affirmative and give an example of a well-known construction that achieves chosen-plaintext security under the computational Diffie-Hellman assumption. Our basic CPA-secure construction H-ELG, which we present in Figure 1, is a straightforward "hashed ElGamal" construction [ABR01] that employs a shared cyclic group $\mathbb{G}$ for all keys generated at a specific security level, and a hash function $H_\ell : \mathbb{G} \times \mathbb{G} \to \{0,1\}^\ell$.[7]

Correctness and Security. Our scheme is a standard variant of hashed ElGamal, a scheme that has been widely studied in the literature. As a result, we omit a detailed proof of correctness and of the IND-CPA or IK-CPA properties. Instead we refer the reader to several existing proofs of security [ABR01, BS01, BBDP01], which employ the computational Diffie-Hellman assumptions as well various assumptions about the nature of the hash function H_ℓ.[8]

[7]Several approaches exist for implementing the hash function H in ElGamal in the literature and in folklore. The main difference between these is the structure and cryptographic assumption used to model H, which can be realized as a hard-core predicate of the Diffie-Hellman function [BS01, FGPS13], by employing standard hash functions and employing "hash" Diffie-Hellman assumptions [ABR01], or by modeling H as a random oracle.

[8]These range from modeling H_ℓ as a random oracle model, to "hash computational Diffie-Hellman" assumptions, and modeling H_ℓ as a hard-core predicate for the CDH function.

It remains, therefore, to prove that the scheme satisfies the property of $\mathcal{D}$-AMB where $\mathcal{D}$ is the uniform distribution over the plaintext space $\{0,1\}^\ell$. Notably, for this proof we require no additional cryptographic assumptions.

Theorem 9. The scheme $\mathsf{H\text{-}ELG}$ is secure in the $\mathcal{D}$-AMB sense, where $\mathcal{D}$ is the uniform distribution over $\{0,1\}^\ell$.

A proof of Theorem 9 is given in Appendix D. Intuitively this proof is simple: observe that in the $\mathcal{D}$-AMB experiment, the $\mathsf{H\text{-}ELG}$ challenge ciphertext comprises a random group element and a bitstring $c_1 = H_\ell(g^k, pk^k) \oplus m$ where m itself is a uniformly random string. We describe a simulation in which the string c_1 is constructed independently of the experiment bit b as is the remainder of the ciphertext, and therefore an adversary's advantage must be 0. More critically, this proof does not require any computational assumptions for the hash function.

CCA security. Achieving CCA-security in ambiguous PKE is slightly more challenging, due to the fragile nature of the ambiguity property. For example, many CCA techniques incorporate key-dependent ciphertext integrity checks within the decryption algorithm (such techniques are used in DHIES [ABR14] and constructions based on the Fujisaki-Okamoto transform [FO99].) The checks in these schemes will produce errors with high probability when the decryptor attempts to decrypt with the the incorrect secret key.

Addressing this requires a CCA-secure construction that does not rely on key-dependent integrity checks. One approach, which we use in this work, is to use a CCA-security construction based on tag-based public key encryption (TBE) [Sho01, MRY04, AGKS05, Kil06]. A useful feature of these constructions, which include the CHK transform [CHK04] and a similar proposal by Mackenzie *et al.* [MRY04] is that ciphertexts incorporate a publicly-verifiable and *key independent* integrity check that relies on one-time signatures. Using this technique eliminates the problems caused by key-dependent ciphertext integrity checks. Thus it remains only to show that the underlying TBE scheme is itself ambiguous, *i.e.,* that decryption using the wrong key does not produce any detectable condition.

Intuition for secure TBE in the random oracle model. In Section 5 we construct an appropriate TBE in the random oracle model for use in our larger FMD constructions. Our construction is a variant of the $\mathsf{H\text{-}ELG}$ scheme above, with the additional requirement that the "tag" for the scheme is fed into the input of the hash function H at both encryption and decryption time. Our approach here is inspired by a technique proposed by Abe and Okamoto [AKO09]. From a security perspective, the critical aspect of this scheme is that decryption with an incorrect key produces a pseudorandom output which preserves (uniform) ambiguity. We leave the problem of finding a standard-model ambiguous TBE to future work.

5 FMD with Restricted False-Positive Rates

In this section we first introduce several constructions that satisfy our definitions, for specific and restricted false positive rates. Specifically, our next constructions will require that each allowed $p \in \mathcal{P}$ takes the form $p = 2^{-n}$ for integers $0 \le n \le \gamma$, where γ is a constant.

5.1 A basic FMD construction from Ambiguous Bit Encryption

In this section we formalize the simple FMD scheme we introduced in §2, and show that it achieves security under chosen plaintext attack. This scheme is built from any public key bit encryption scheme that is both uniformly ambiguous and $\mathsf{IK\text{-}CPA}$ secure, such as the $\mathsf{H\text{-}ELG}$ scheme from the previous section.

This construction, which we call $\mathsf{FMD1}$, is presented in Figure 2.

Intuitively, the scheme is quite simple: each public key for the scheme comprises γ independent public keys for the underlying ambiguous bit encryption scheme PKE, and each ciphertext consists of γ separate PKE encryptions of the plaintext "1", using each corresponding public key. To extract a detection key for some

```
KeyGen($1^\lambda$) :
for $i \in [\gamma]$ :
    $pk_i, sk_i \leftarrow$ PKE.KeyGen($\lambda$)
$pk \leftarrow (pk_1, \ldots, pk_\gamma)$
$sk \leftarrow (sk_1, \ldots, sk_\gamma)$

Extract($sk, p = 2^{-n}$) :
$(sk_1, \ldots, sk_\gamma) \leftarrow sk$
Return $(sk_1, \ldots, sk_n)$

Flag($pk$) :
$(pk_1, \ldots, pk_\gamma) \leftarrow pk$
$C \leftarrow$ PKE.Enc($pk_1, 1$) $\cdots$ PKE.Enc($pk_\gamma, 1$)
Return $C$

Test($dsk, C$) :
$(sk_1, \ldots, sk_n) \leftarrow dsk$
$(C_1, \ldots, C_\gamma) \leftarrow C$
$s \leftarrow$ PKE.Dec($sk_1, C_1$) $\cdots$ PKE.Dec($sk_n, C_n$)
if $s = 1^n$ :
    Return 1
Return 0
```

Figure 2: A CPA-secure FMD scheme FMD1 supporting restricted values of p ($p \approx 2^{-n}$). PKE = (KeyGen, Enc, Dec) is a uniformly-ambiguous *bit encryption* scheme, which can be realized using H-ELG from §4.3.

probability $p = 2^{-n}$, the key holder simply reveals n of the corresponding secret decryption keys. The Test algorithm operates as follows: a decryptor simply attempts to decrypt n of the sub-component ciphertexts, and verifies that each decryption outputs 1. (In the special case where $n = 0$, then this test is deemed to always succeed.)

Correctness, fuzziness and security for the scheme in Figure 2 depends on the properties of the underlying ambiguous bit encryption scheme. We now prove that they hold.

Theorem 10. Let FMD1 = (KeyGen, Flag, Extract, Test) be the scheme given in Figure 2, and let PKE be a uniformly-ambiguous public key encryption scheme (in the sense of Definition 7) with plaintext domain $\{0, 1\}$. Then FMD1 is correct and "fuzzy" (in the sense of Definition 3.)

For a proof of Theorem 10 see Appendix B.

Theorem 11. Let PKE = (KeyGen, Enc, Dec) be a uniformly-ambiguous bit encryption scheme in the sense of Definition 7. Then FMD1 is DA-CPA secure for the set $\mathcal{P} = \{\frac{1}{2^n} \mid n \in [\gamma]\}$. Moreover, let $\mathcal{A}$ be an adversary in the *selective* version of the DA-CPA definition for the scheme in Figure 2. Then for any particular $p = \frac{1}{2^n}$, there exists an adversary $\mathcal{B}$ against the IK-CPA security of PKE, and an adversary $\mathcal{E}$ against the $\mathcal{D}$-AMB security of PKE such that

$$\mathbf{Adv}^{\mathsf{DA\text{-}CPA}}_{\mathcal{A}}(\lambda) \leq \ (\gamma - n)\mathbf{Adv}^{\mathsf{IK\text{-}CPA}}_{\mathcal{B}}(\lambda) + \ 4n \cdot \mathbf{Adv}^{\mathcal{D}\text{-}\mathsf{AMB}}_{\mathcal{E}}(\lambda).$$

For a proof of Theorem 11 see Appendix C.

5.2 A CCA-secure restricted FMD construction

The previous construction suffers from two chief limitations. First, each FMD ciphertext comprises up to γ full ciphertexts for the underlying ambiguous PKE scheme, which leads to poor bandwidth efficiency. More critically, ciphertexts are insecure when an attacker has access to a Test oracle that can be used to decrypt chosen ciphertexts. The reason for this is straightforward: not only is the underlying PKE scheme potentially vulnerable to CCA attacks in its own right, but an attacker may be able to maul a challenge FMD ciphertext in other ways, *e.g.,* by simply changing the order of the component ciphertexts. To address these issues, we construct a more efficient direct scheme that achieves CCA-security in the random oracle model for restricted values $p = 2^{-n}$.

We take as a starting point our hashed ElGamal construction H-ELG, and we modify it first to support encryption under up to γ public keys in a single ciphertext, using a single random element $u = g^r$. We next produce a tag-based encryption scheme by incorporating a tag (u, w) into the input to the hash function H within the encryption algorithm. If H is modeled as a random oracle, then any change to this tag will result

```
KeyGen($1^\lambda$) :
for $i \in [\gamma]$ :
    $x_i \leftarrow \mathbb{Z}_q$
    $h_i \leftarrow g^{x_i}$
$sk \leftarrow (x_1 \ldots x_\gamma)$
$pk \leftarrow (g, h_1, \ldots h_\gamma)$
Return $sk, pk$

Extract($sk, p = 2^{-n}$) :
$(x_1, \ldots, x_\gamma) \leftarrow sk$
Return $(x_1, \ldots, x_n)$
```

```
Flag($pk$) :
$h_1, \ldots h_\gamma \leftarrow pk$
$r \xleftarrow{\$} \mathbb{Z}_q$
$u \leftarrow g^r$
$z \xleftarrow{\$} \mathbb{Z}_q$
$w \leftarrow g^z$
for $i \in [\gamma]$ :
    $k_i \leftarrow H(u \| {h_i}^r \| w)$
    $c_i \leftarrow k_i \oplus 1$
$m \leftarrow G(u \| c_1 \| \ldots \| c_\gamma)$
$y \leftarrow \left[(z - m)r^{-1}\right] \pmod q$
Return $(u, y, \{c_i\}_{i \in [\gamma]})$
```

```
Test($dsk, (u, y, \{c_i\}_{i \in [\gamma]})$) :
$x_1, \ldots x_n \leftarrow dsk$
$m \leftarrow G(u \| c_1 \| \ldots \| c_\gamma)$
$w \leftarrow g^m \cdot u^y$
for $i \in [n]$:
    $k_i \leftarrow H(u \| u^{x_i} \| w)$
    $b_i \leftarrow k_i \oplus c_i$
if $b_i = 1, \forall i \in [n]$ :
    Return 1
else:
    Return 0
```

Figure 3: A CCA-secure efficient FMD scheme, FMD2 supporting restricted values of p ($p \approx 2^{-n}, n \leq \gamma$). The constant γ is a global parameter and $\mathbb{G}$ is a common group of prime order q, generated by a setup procedure that is not explicitly described. $H : \mathbb{G}^3 \rightarrow \{0,1\}$, $G : \mathbb{G} \times \{0,1\}^\gamma \rightarrow \mathbb{Z}_q$ are hash functions.

in a random bit upon decryption. Recall that u is the first (shared) element of the ElGamal ciphertext. The value w corresponds to a chameleon hash [KR00] computed on the message $(0, z)$, where z is chosen at random. Once the ciphertext has been computed, we use a master trapdoor for the chameleon hash (which is part of the scheme's secret key) in order to compute a collision (y, m) where m is a hash of the remaining components of the ciphertext. In this scheme w can be viewed as a public key for a one-time signature, and the resulting hash collision can be viewed as a signature, which fits within the CHK paradigm [CHK04]. We present the resulting construction in Figure 3.

Theorem 12. Let FMD2 = (KeyGen, Flag, Extract, Test) be the scheme given in Figure 3. Then FMD2 is correct and "fuzzy" (in the sense of Definition 3.)

A proof of Theorem 5.2 is given in Appendix E.

Theorem 13. Let FMD2 = (KeyGen, Flag, Extract, Test) be the FMD scheme in Figure 3. Then FMD2 is secure under DA-CCA where $\mathcal{P} = \{\frac{1}{2^s} \mid s \in [\gamma]\}$. More specifically, suppose $\mathcal{A}$ is an adversary who defeats the *selective* version of the DA-CCA game using q_H queries to hash oracle H, q_G queries to oracle G, and q_D decryption queries running in time τ. Then there exists a discrete log adversary B running in time at most $\tau + 3 \cdot g_{OP}$ and a Gap-DH adversary C running in time at most $\tau + O(q_d(q_H^2 + q_H \cdot n))$, where:

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{DA\text{-}CCA}}(\lambda) \leq \frac{q_G}{q} + \mathbf{Adv}_{C}^{\mathrm{DLOG}}(\lambda) + q_H \cdot \mathbf{Adv}_{\mathcal{B}}^{\mathrm{Gap\text{-}DH}}(\lambda).$$

For a proof of Theorem 13 see Appendix F.

5.3 Revocation and compact public keys

A practical limitation of the above schemes is that detection keys, once issued, cannot be revoked; nor can the probability p be changed over time. A simple approach to addressing this problem is for recipients to generate and distribute new public keys on a routine basis, *e.g.,* once every time epoch. Unfortunately, this dramatically increases the size of the public key material that must be distributed. An alternative approach is to replace the underlying PKE scheme with an appropriate *ambiguous* Identity Based Encryption (IBE) scheme, where each "identity" used by the encryptor represents a time period. An additional advantage of this approach is that we can reduce the size of each individual public key by a factor of γ, replacing the vector of public keys with a single set of IBE master parameters. The ElGamal-based scheme of the previous section can be modified by replacing the ElGamal primitive with an IBE key encapsulation mechanism based on the Boneh-Franklin IBE scheme [BF01]. We present the full construction in Appendix G.

6 Constructions with fractional false-positive rates

Our previous schemes are limited to allowable false-positive rates p such that $p = 2^{-n}$. However, applications exist where such limitations on p are undesirable. In this section, we present a feasibility result, constructing a scheme that supports more granular rates of the form $p = \frac{M}{N}$. As described in section 2, the key idea in our construction is to leverage a form of bounded functional encryption [SS10] for a sampling function $f : \{0,1\}^{\kappa+\gamma} \times \{0,1\}^{\gamma} \to \{0,1\}^{\gamma}$, where the first input corresponds to a set of random coins selected by the encryptor, and the second input encodes an integer M chosen by the receiver. The function must output a (nearly) uniform value in the set $[M]$.

Intuition. We can characterize all of our previous constructions as the concatenation of many instantiations of a single bit encryption scheme where for each ciphertext bit, the probability of decrypting incorrectly and receiving the sentinel value used for flagging is $\frac{1}{2}$ due to the ambiguity properties of the underlying encryption scheme. By providing a subset of n secret keys to the decryptor, we can reduce the match probability to 2^{-n}. The challenge with these schemes is that some false-positive rates are difficult to reach. Probabilities such as $\frac{1}{3}$ could be accomplished by simply increasing the plaintext domain of the underlying ambiguous encryption scheme from $|\mathcal{M}| = 2$ to $|\mathcal{M}| = 3$. But probabilities like $\frac{3}{4}$ seem fundamentally challenging to achieve using this approach.

Suppose we relax the requirement that the sender not know the probability p. How could we achieve an arbitrary probability of $p = \frac{M}{N}$ (for $0 \leq M \leq N$) in this setting? If we assume both parties have access to a uniformly-ambiguous PKE scheme with a plaintext domain $\mathcal{M}$ of size N, at least one answer seems straightforward: fix a subset $\mathcal{M}' \subset \mathcal{M}$ that contains only M elements. The sender samples uniformly from $\mathcal{M}'$ and the test algorithm checks if the plaintext resulting from decryption is within $\mathcal{M}'$. For matched ciphertexts, this test will always succeed. For mismatched keys and ciphertexts, decryption should produce a result that is (nearly) uniformly distributed in the full plaintext space $\mathcal{M}$. Thus, provided there is an efficient algorithm for checking membership in $\mathcal{M}$, this solution achieves an arbitrary probability of decryption equal to $\frac{M}{N}$.

The problem with this approach is that, while we can reasonably fix the value N as a constant within an FMD construction, the sender will not know M or the set $\mathcal{M}'$ at the time of encryption, and thus cannot sample a plaintext from this subset. Indeed, M may not be selected until *after* a ciphertext has been constructed. Addressing this requires a way to allow the sender to delay the sampling process until after the receiver has chosen M. One way we can accomplish this is by using a functional encryption scheme with *ambiguity* properties, wherein the sampling process is itself embedded within the decryption procedure. Such schemes do not, to our knowledge exist, and thus we must derive the necessary properties directly.

Scheme Overview. Our scheme operates as follows. Each recipient generates a set of 2γ public keys for a (uniformly) ambiguous public-key encryption scheme with plaintext space $\{0,1\}^{\lambda}$. The encryptor now constructs a boolean circuit $\mathbf{C}$ that implements the function $f(R, M)$, where one set of input wires corresponds to the sender's random coins R, and a second set corresponds to the receiver's input M. To produce a flag ciphertext, the encryptor now garbles $\mathbf{C}$ using a modified garbling scheme that we describe below. This scheme produces a pair of wire labels for each input wire, with each label in the domain $\{0,1\}^{\lambda}$. The encryptor samples $R \in \{0,1\}^{\kappa+\gamma}$ and outputs the garbled circuit, along with the subset of labels that encode R. For each of the m input wires that correspond to the receiver's input, the encryptor encrypts each of the two labels using a distinct public key produced by the recipient, and outputs the full set of ciphertexts. To extract a detection key for some specific value M, the recipient simply determines which bit on each input wire of the circuit encodes the integer M that it chooses, and outputs one secret key for each wire to the server. The server operator can now decrypt exactly one encrypted label for each input wire, such that the full collection corresponds to the value M chosen by the recipient. It evaluates the garbled circuit on input labels corresponding to R and M to obtain a (plaintext) output O. The Test algorithm is deemed to output 1 iff $O < M$.

The novel contribution in our construction is a garbled circuit scheme that is ambiguous. Since we are

```
$\underline{\mathsf{KeyGen}(1^\lambda):}$
For $i \in |\mathcal{I}_2|, v \in [0,1]$
    $(pk_i^v, sk_i^v) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$
$mpk \leftarrow \{pk_i^v\}_{i \in |\mathcal{I}_2|, v \in [0,1]}$
$msk \leftarrow \{sk_i^v\}_{i \in |\mathcal{I}_2|, v \in [0,1]}$
Return $(mpk, msk)$

$\underline{\mathsf{Extract}(msk, p):}$
$\{sk_j^i\}_{i \in [0,1], j \in |\mathcal{I}_2|} \leftarrow msk$
$\frac{M}{2^\gamma} \leftarrow p$
$z \leftarrow \mathsf{bin}(M)$
Return $(z, \{sk_j^{z_j}\}_{j \in |\mathcal{I}_2|})$
```

```
$\underline{\mathsf{Flag}(mpk):}$
$\{pk_j^i\}_{i \in [0,1], j \in |\mathcal{I}_2|} \leftarrow mpk$
$\tilde{C}, \{L_j^i\}_{i \in [0,1], j \in |\mathcal{I}_1 \cup \mathcal{I}_2|} \leftarrow \mathsf{AGarble}(\lambda, \mathbf{C})$
$R \xleftarrow{\$} \{0,1\}^{|\mathcal{I}_1|}$
For $i \in [0,1], j \in [|\mathcal{I}_2|]$ do
    $c_j^i \leftarrow \mathsf{PKE.Enc}(pk_j^i, L^i_{|\mathcal{I}_1|+j})$
Return $\tilde{C}, \{c_j^i\}_{i \in [0,1], j \in |\mathcal{I}_2|}, \{L_i^{R_i}\}_{i \in |\mathcal{I}_1|}$
```

```
$\underline{\mathsf{Test}(dsk, C):}$
$(z, \{dsk_i\}_{i \in |\mathcal{I}_2|}) \leftarrow dsk$
$\tilde{C}, \{c_j^i\}_{i \in [0,1], j \in |\mathcal{I}_2|}, \{L_j\}_{j \in |\mathcal{I}_1|} \leftarrow C$
for $j$ in $[|\mathcal{I}_2|]$:
    $L_{|\mathcal{I}_1|+j} \leftarrow \mathsf{PKE.Dec}(dsk_j, c_j^{z_j})$
$t \leftarrow \mathsf{AEval}(\mathbf{C}, \tilde{C}, \{L_j\}_{j \in |\mathcal{I}_1 \cup \mathcal{I}_2|})$
if $\mathsf{integer}(t) < M$:
    Return 1
else:
    Return 0
```

Figure 4: $\mathsf{FracFMD}$, an FMD scheme that uses a uniformly-ambiguous public key encryption scheme PKE with plaintext space $\{0,1\}^\lambda$ and our garbled circuit scheme ($\mathsf{AGarble}, \mathsf{AEval}$) from §6.1. The scheme embeds the description of a circuit $\mathbf{C} : \{0,1\}^{\kappa+\gamma} \times \{0,1\}^\gamma \to \{0,1\}^\gamma$ that implements the function $f(R, M) = R \bmod M$ where R, M are treated as non-negative integers, $|\mathcal{I}_1| = \kappa + \gamma$, and $|\mathcal{I}_2| = \gamma$. The function $\mathsf{integer}(\cdot)$ converts a bitstring into an integer. The function $\mathsf{bin}(\cdot)$ converts an integer to a γ bitstring.

concerned only with cases where Test outputs 1, the key is to ensure that an adversary cannot distinguish between the function output $f(R, M)$ and a uniform output O that happens to satisfy $O < M$. To do this, we design a garbling scheme in which incorrect evaluation (with random labels) cannot be distinguished from correct execution of a function that has an output with a (nearly) uniform distribution. While Yao's original scheme does not provide this property, we are able to achieve it by using a variant of the point-and-permute technique [BMR90]. Our final observation is when the server decrypts the ambiguous public-key ciphertexts using an incorrect collection of secret keys, the ambiguity property of the encryption scheme ensures that the distribution of the resulting labels will be close to uniform.

Our construction is presented in its entirety in Figure 4. For simplicity, we specify a double key encryption scheme defined as $\overline{\mathsf{Enc}}(k_0, k_1, m)$ where each of k_0, k_1 and m is of length λ bits. For the sampling function we employ the circuit $f(R, M) = R \pmod M$ with the assumption that the bit length of R is sufficiently long enough to minimize bias following the modular reduction.

6.1 An "ambiguous" garbling scheme

Our approach requires a variant of Yao's garbling scheme [Yao86] that provides an ambiguity property. Specifically, we require that an adversary cannot distinguish between a correct evaluation of a garbled circuit, and an "incorrect" evaluation in which the receiver's input labels have been replaced with random strings. Naturally this definition is highly dependent on the properties of the specific circuit being evaluated. To sidestep this issue, we define our security property in terms of a simulation-based definition which we present further below.

Our garbling scheme provides two algorithms: $\mathsf{AGarble}$ and AEval. Our implementation of $\mathsf{AGarble}$ takes as input a circuit C and a security parameter λ. It produces a garbled circuit $\tilde{C}$ along with a set of labels $\{L_{i,v}\}_{i \in \mathcal{I}, v \in \{0,1\}}$ one for each input wire and each value the wire can take. The algorithm AEval takes a garbled circuit $\tilde{C}$ as input along with a set of labels $\{L_{i,x_i}\}_{i \in \mathcal{I}}$ and produces some output $\{z_i\}_{i \in \mathcal{O}}$. The garbling scheme differs from the classical scheme of Yao in one significant way: Yao's scheme (and variants) incorporate decryption checks, which will cause evaluation to fail with high probability when conducted using the wrong input labels. We note that this condition can be eliminated by employing the "point and permute" technique first introduced in [BMR90]. In this approach, each gate evaluation will always succeed, even if the wrong input labels are used.
As with a classical garbling scheme, we require double key encryption scheme $\overline{\mathsf{Enc}}(k_0, k_1, m)$. For our purposes we define $\overline{\mathsf{Enc}}(k_0, k_1, m) = \mathsf{PRG}(k_0, k_1, |m|) \oplus m$ where $\mathsf{PRG}(k_0, k_1, \ell)$ is a two-key pseudorandom generator

with output length ℓ bits.

- ▶ **AGarble(λ, C) :** Garbling a circuit $\mathbf{C}$ produces $|\mathbf{C}|$ garbled tables and $|\mathcal{I}|$ "keys" by first generating three values for every single wire $i \in \mathcal{I} \cup \mathcal{W}$: $L_{i,0} \overset{\$}{\leftarrow} \{0,1\}^\lambda, L_{i,1} \overset{\$}{\leftarrow} \{0,1\}^\lambda,\ r_i \overset{\$}{\leftarrow} \{0,1\}$. The first two are labels corresponding to the wire values 0 and 1 respectively while the last is a *permute bit* (also known as a mask).

  Let G be a gate in $\mathbf{C}$ with input wires α, β, output wire δ, and $g : \{0,1\} \times \{0,1\} \to \{0,1\}$ the function this gate computes. Let $\hat{a} = a \oplus r_\alpha$, $\hat{b} = b \oplus r_\beta, L_{\alpha,\hat{a}}$, and $L_{\beta,\hat{b}}$ be the labels and masked bits the evaluator holds when evaluating the table where a is the true value of wire α and b the true value of wire β. We construct the garbled table of G by enumerating through the possible values of $\hat{a}$ and $\hat{b}$ as follows:

Garbled Output
$\overline{\mathsf{Enc}}(L_{\alpha,0}, L_{\beta,0}, (L_{\delta,g(r_\alpha,r_\beta)\oplus r_\gamma} \| g(r_\alpha, r_\beta) \oplus r_\delta)$
$\overline{\mathsf{Enc}}(L_{\alpha,0}, L_{\beta,1}, (L_{\delta,g(r_\alpha,\overline{r_\beta})\oplus r_\delta} \| g(r_\alpha, \overline{r_\beta}) \oplus r_\delta)$
$\overline{\mathsf{Enc}}(L_{\alpha,1}, L_{\beta,0}, (L_{\delta,g(\overline{r_\alpha},r_\beta)\oplus r_\delta} \| g(\overline{r_\alpha}, r_\beta) \oplus r_\delta)$
$\overline{\mathsf{Enc}}(L_{\alpha,1}, L_{\beta,1}, (L_{\delta,g(\overline{r_\alpha},\overline{r_\beta})\oplus r_\delta} \| g(\overline{r_\alpha}, \overline{r_\beta}) \oplus r_\delta)$

  Equivalently, at each row $2 \cdot \hat{a} + \hat{b}$ of this table we store the value $\overline{\mathsf{Enc}}(L_{\alpha,\hat{a}}, L_{\beta,\hat{b}}, L_{\delta,g(a,b)\oplus r_\delta} \| g(a,b) \oplus r_\delta)$. This ensures that when the evaluator decrypts using $L_{\alpha,\hat{a}}$ and $L_{\beta,\hat{b}}$ they receive the evaluation of the gate on the unmasked inputs hidden by the mask bit of the next wire.

  Every internal gate in the circuit is garbled in this fashion. Those gates connected to the output wires of the circuit are garbled differently: for the i^{th} output gate, instead of encrypting labels, the table directly encrypts the output gate value $g(a,b)$.

- ▶ **AEval($\mathbf{C}, \tilde{C}, \{L_\mathcal{I}\}$)** : The AEval algorithm is relatively straightforward. Using labels and masked bits $\{L_\mathcal{I}\}$, the evaluator evaluates the circuit top down, decrypting the rows in each table according to the permute bits and the given labels before outputting the values it gets from the final output gate tables.

6.2 Fractional FMD construction

We present our complete FMD construction FracFMD for fractional false-positive rates in Figure 4. The scheme makes use of an ambiguous garbling scheme (AGarble, AEval), as well as a uniformly-ambiguous CPA-secure public-key encryption scheme PKE with plaintext domain $\mathcal{M} = \{0,1\}^\lambda$.[9]

Correctness of the scheme is based on the correctness of the underlying ambiguous garbling scheme and public-key encryption. We now consider the fuzziness and security properties.

Theorem 14. Let (AGarble, AEval) be the garbling scheme specified in section 6.1 and let PKE be a uniformly-ambiguous public key encryption scheme secure in the sense of Definition 7. Then FracFMD = (KeyGen, Extract, Flag, Test) is fuzzy in the sense of Definition 3.

Proof. Fuzziness is proven directly by the fact that PKE is uniformly ambiguous and by $\overline{\mathsf{Enc}}$ being pseudorandom. From Theorem 8, the input that is given to $\tilde{C}$ are uniformly random strings. Because encryption is XOR with a pseudorandom string, decrypting table rows will never result in $\perp$ and since the adversary never learns the actual labels used to garble the circuit, each evaluated row is indistinguishable from the uniform distribution over $\{0,1\}^{\lambda+1}$. Now consider the output gates denoted as $h_0 \dots h_\gamma$ where $h_i = \mathsf{PRG}(L^a_\alpha, L^b_\beta) \oplus o_i$.

[9]The latter scheme can be realized using a multi-bit variant of the H-ELG construction of §4.3.

Again, the decryptor in this case does not know L^a_α or L^b_β so by the security of the PRG, h_i is uniform for all i and the output $\overline{o} \in \{0,1\}^\gamma$ of the circuit is uniform. Let probability $p = \frac{M}{N}$ so that $Test(dsk, C) = 1$ occurs when the output of the circuit is less than M. The probability that $\overline{o}$ is less than M is simply $\frac{M}{2^\gamma} = \frac{M}{N}$ and thus the scheme is fuzzy. □

Theorem 15. FracFMD *is* DA-CPA *secure.*

A proof of this theorem is included in Appendix J for details. Here we provide some intuition for why our garbled circuit scheme can achieve security against a DA-CPA adversary.

For simplicity we begin by considering a strawman attacker that is *selective.* We will then modify our approach to the adaptive definition. Our overall approach is to show that we can construct a ciphertext independent of the public key chosen by the bit b, and that with some probability this ciphertext will produce correct outputs satisfying the Test condition for both detection keys.

There are two things that depend on b: the ciphertexts encrypting the input labels and the evaluation of the garbled circuit. By the ambiguity of the public key encryption scheme, decrypting the labels with the wrong keys should produce the uniform distribution, which is the same distribution used for label sampling. We now turn our attention to the garbled circuit. If we replace the encrypted tables with uniform values, the evaluator can distinguish either by discrepancies in decrypting internal gate or final circuit output. Because the internal gates encrypt uniformly random values, the output of incorrect decryption appears uniformly random to the adversary if they do not have access to the real secret key. For similar reasons, the output of incorrect decryption is also uniform in the range specified by the number of output wires. Since correct functionality outputs a value uniformly in $[0, M)$, incorrect evaluation that falls into this range is indistinguishable from correct.

If the adversary was selective, we would be done since we can simulate by generating a random ciphertext, testing it against both detection keys, and then retrying as needed. This approach would require that we generate an expected $\frac{1}{p^2}$ ciphertexts. The challenge in our proof is that the adversary is adaptive, and can select a new p each time we rewind. If the number of allowed probabilities $\mathcal{P}$ was polynomial in the security parameter, we could simply guess the probability p that will be selected by the adversary, and abort when our guess is incorrect. Unfortunately the cardinality of $\mathcal{P}$ could be exponential in the security parameter. Instead, we take advantage of the fact that the probability of constructing a garbled circuit C which evaluates to a false positive depends entirely on the adversary chosen probability M. Take the smallest probability p_{min} that is non-negligible. This must exist and probabilities higher or equal to this must be asked for sufficiently often or else the adversary has no advantage. Then we know the lower bound on the expected number of retries is $\frac{1}{p^2_{min}}$ and the normal selective analysis actually still holds.

7 Implementation and Evaluation

We implemented two of the schemes from the previous sections and conducted a set of experiments to determine the efficiency of these constructions in practical settings.

Table 1: Microbenchmarks for FMD2 (as described in Figure 3) with a maximum false positive rate of 2^{-15} ($\gamma = 15$). Note that KeyGen and Flag execution time is independent of p.

ciphertext size	KeyGen (time)	Flag (time)	p	Test (time)	Extract (time)
68 bytes	0.447 ms	1.927 ms	2^{-5}	0.548 ms	96.8 ns
			2^{-10}	0.933 ms	108 ns
			2^{-15}	1.311 ms	123 ns

Table 2: Microbenchmarks for FracFMD (as described in Figure 4). All experiments use the statistical parameter $\kappa = 40$. We give the complexity of the sampling circuit in terms of the number of non-XOR gates because XOR gates do not have any impact on the garbled circuit size.

FracFMD denominator size (2^γ)	# non-XOR gates	Ciphertext (size)	KeyGen (time)	Flag (time)	Test (time)	Extract (time)
2^8	5,757 gates	318,530 bytes	0.298 ms	18.061 ms	9.585 ms	196 ns
2^{24}	21,925 gates	1,230,914 bytes	0.894 ms	66.672 ms	35.653 ms	355 ns

7.1 Implementation

We implemented both the FMD2 and FracFMD schemes in Go [10]. For FMD2 we realized the group $\mathbb{G}$ using the curve `secp256r1` provided in Go's `elliptic` library [Goo], using point compression for ciphertext serialization. The hash functions G, H use Go's SHA-256 implementation with unique prefixes. For FracFMD we realized the uniformly ambiguous encryption scheme using an implementation of the H-ELG scheme with plaintext size $\ell = 128$ bits over the same elliptic curve. To implement the sampling functions for the garbled circuit, we wrote two short C programs computing $a \ mod \ B$ with hardcoded $B \in \{2^8, 2^{24}\}$ and a an integer of size 48 bits and 64 respectively (this encodes a $\kappa = 40$ bit statistical security parameter for the sampling function.) We compiled each circuit using the CBMC-GC compiler [FHK+14] (as packaged by Hastings *et al.* [HHNZ19].) Finally, we wrote a garbled circuit implementation in Go using the BLAKE hash function for encryption, and employing both the point-and-permute and Free-XOR optimization of Kolesnikov and Schneider [KS08].

7.2 Evaluation

Our evaluation section covers the following: a very preliminary exploration of the effect of parameter choices on anonymity properties, potential applications of FMD, microbenchmarks for both FMD2 and FracFMD with varying values of p and γ, and lastly an analysis of the efficiency of utilizing FMD2 in a simple end-to-end deployment scenario. Our experiments show that FMD is not the bottleneck in at least some real world scenarios. The end to end evaluation was conducted on a laptop with an i7-8550U processor at 1.80GHz using 16 GB RAM and running Ubuntu 18.04. For the microbenchmarks evaluation was done on a laptop with an i5-1038NG7 processor at 2.00GHz using 32 GB RAM and running macOS Big Sur. Microbenchmarks were computed using Go's benchmarking tool.

Selecting the parameters p, γ. One critical decision that must be made when using fuzzy message detection is selecting the appropriate parameters for a specific application scenario. Parameter choice will depend on the number of total messages at the server, the number of users, and the recipient's expected traffic.

Lewis's Attacks After seeing an unpublished manuscript of our work, an independent researcher Lewis [Sar21a] implemented FMD2 and examined FMD at a conceptual level to determine how resilient it is to de-anonymization attacks. This work observes that the privacy received is dependent on parameter selection, which is application specific. Lewis explores *differential attacks* in which a curious server learns that a subset of messages are intended for one recipient and uses this to identify the receiver. She also investigates different statistical attacks, including ones which identify if a recipient has received any message in a given time-frame and even an attack which uncovers the recipients for a small subset of messages with a low false positive rate. Importantly, the most egregious of these (i.e. identification of the receiver) can hopefully be mitigated by attempting to prevent the leakage of meta-data to the server. For example, having senders connect via an anonymous network to send messages will help prevent link identification from statistical attacks and having some users download all messages will help prevent differential attacks. For a more in-depth exploration

[10]Available at `https://github.com/becgabri/fuzzycrypto`

p	Time Period		
	1 day	1 month	1 year
5×10^{-4}	3.951×10^{-82}	1.185×10^{-80}	1.422×10^{-79}
3.3×10^{-4}	3.650×10^{-53}	1.095×10^{-51}	1.314×10^{-50}
1.7×10^{-4}	3.325×10^{-24}	9.976×10^{-23}	1.197×10^{-21}

Table 3: Upper bounding probability of trivial attribution in potential cryptocurrency application. Values of p correspond to 150, 100, and 50 expected false positives, assuming no messages addressed to the receiver.

with an existing simulator for exploring parameter choices, and preliminary heuristic guidelines see the other work at [Sar21b].

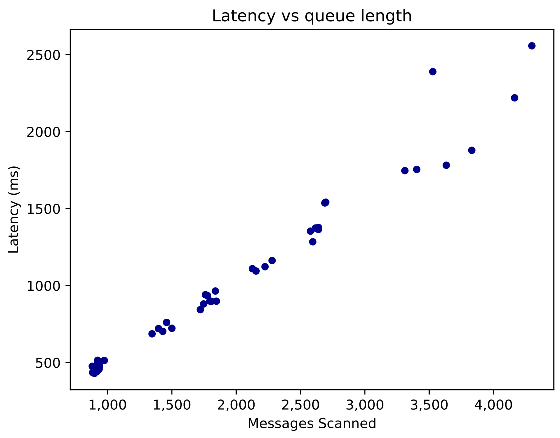
Case Study: Cryptocurrency To better understand how to select the false positive rate, we present a simple case study of a cryptocurrency application where every transaction has only one recipient. We solely consider *trivial attributions* as defined by Lewis [Sar21b], meaning a transaction is sent to only one recipient. We assume there are 400,000 unique recipients and that the average number of transactions per day is 300,000, which approximately represents the daily level of traffic seen by Bitcoin [bit]. In Table 3, we calculate an upper bound on the probability of at least one trivial attribution over the course of a day, a month, and a year, assuming all users use the same probability p. As p decreases, the probability of a trivial attribution increases drastically. However, we can tune p to a more reasonable value that minimizes both the number of incorrectly addressed messages received and the probability for trivial attribution.

Potential Applications One of the main avenues we see for FMD is as a supplemental tool in anonymous communication systems for providing probabilistic receiver anonymity with low client overhead. Another direction for FMD is in providing a variant of differential privacy. On its own, FMD does not provide DP guarantees. However, we believe we can achieve something close to randomized response by augmenting our scheme with a false negative rate p'. If we set $p = p'$ and change our setting, we can likely achieve $(\ln\frac{1-p}{p}, 0)$ bounded differential privacy. We conjecture this false negative rate can be obtained by modifying our circuit in the FracFMD scheme to use extra random coins from the sender to decide whether to output something uniform in the range $[0, M)$ or $[M, 2^\gamma)$. It is less clear how our other schemes could be modified to support this capability: just as with the false positive rate, a false negative and a true negative must be indistinguishable from one another. This disqualifies trivial attempts to modify the detection key for our bit-wise schemes by lying about the secret key at a particular index. Adding a false negative rate would affect both usability and security. We leave a full exploration of this and other application scenarios to future work.

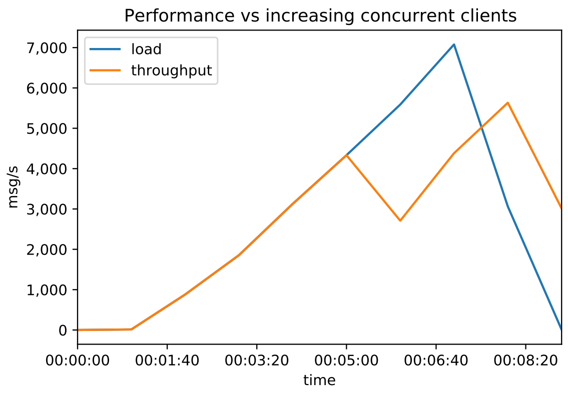
Microbenchmarks for restricted FMD Table 1 gives baseline performance for the restricted FMD2 scheme at $\gamma = 15$ with a minimum fuzziness of 2^{-15} (i.e. ciphertexts encoded 15 flag bits). The ciphertext size is reasonably small and decryption time grows only slightly as p gets smaller.

Microbenchmarks for Fractional FMD We evaluated the FracFMD scheme with parameter $\gamma \in \{8, 24\}$, assuming optimizations for Free-XOR and point-and-permute with a 120-bit encryption key size.[11] We observe that encryption and test times depend on the circuit size, which is a function of the *denominator*. Hence we ran our benchmarks on random numerator values. For a setting that supports fractional values $x/2^8$ we calculated a ciphertext size of 318,530 bytes and for fractional values $x/2^{24}$ the size was 1,230,914 bytes. Fractional FMD scheme appears to have small overhead in terms of encryption and test operations; the

[11]We store labels in a 16-byte field, using 15 bytes to store labels and 1 byte to store the selector bit for point-and-permute (padded with 7 random bits.)



(a) Latency that a client experiences as a result of a GET request where the server must check x ciphertext values. Done with $n = 5$, $\gamma = 80$ for the scheme presented in Figure 3.



(b) Throughput and load as a function of time in experiment 2. Load is calculated as the sum of submitted flag messages and submitted requests to test those messages. Throughput is calculated as the number of messages processed in a given response when it completes. Measurements are for a 60s rolling average.

Figure 5: Performance in an end-to-end scenario.

major cost is the large ciphertext size, due to the use of garbled circuits. This motivates the development of improved ambiguous functional encryption, which could produce more efficient fractional FMD constructions.

End-to-End Evaluation We expect system performance to be governed primarily by two parameters: (1) the frequency with which recipients query for messages relative to how frequently messages are sent (this affects queue size), and (2) load on the server as a function of the the number of concurrent clients.

Experimental setup For our experimental setup, we script clients on a separate server to both send and receive messages. We arbitrarily detection probability at 1 in 32. I.e., testing is done with the first 5 keys out of a total of 60. For our experiments, recipients query the server directly instead of via Tor. This eliminates Tor's large latency variability in our experiment.

Results Results are given in Figure 5. Experiment 1 confirms our intuition: as the number of messages being processed per recipient rises, the time to handle each request increases linearly. Experiment 2 is more surprising. We expected to see the server swamped under load at which point throughput would degrade. Indeed we observe this around six minutes into the experiment. Shortly thereafter, however, the server fails to handle message submissions. This is not due to the overhead of processing test requests, but due to the MySQL connection limit triggering on sender message submission. This operation does not involve our FMD scheme, however; rather, it indicates that our application reaches the limit of its ability to process submitted messages.

8 Related Work

A number of systems provide metadata privacy for store-and-forward networks. Nearly all approaches rely on an out of band shared secret. Fuzzy Message Detection can be seen as a way to bootstrap such a secret. Using a shared-secret, Pung [AS16] constructs fully anonymous messaging from a generic keyword PIR [CGN98] scheme. The keyword PIR serves as a message store with messages located at a random identifier derived from a secret shared between the sender and recipient. Similarly DP5 [BDG15] uses PIR to provide presence notification again assuming shared secrets. Signal's Sealed Sender [Lun17] feature hides the sender of a message, but not the recipient, from the server.

The notable exception is Alpenhorn [LZ16]. which is explicitly meant to bootstrap anonymous communication given no prior interaction. At its core, Alpenhorn uses a mixnet to deliver an encrypted message to a bucket $\mathsf{H}(\mathit{username@email.com}) \bmod k$ where k is the number of messages in the current round. Clients download all messages in that bucket and trial decrypt to locate their friend requests. It is in essence the same "download everything" and trial decrypt approach we wish to refine. Alpenhorn's approach fundamentally requires at least one honest server in the mixnet to achieve privacy. When used with the Vuvuzela Mixnet [vdHLZZ15], it provides differential privacy. In contrast, fuzzy message detection natively provides only probabilistic privacy guarantees without a trusted server.

Our techniques are superficially similar to the "blind Bernoulli trials" technique of Connor and Schuchard [CS19]. However, this work solves the opposite problem: they allow a sender to pick the probability which a server will select their message while keeping the probability private. This requires a trusted third party. In contrast, our protocols let the recipient decide the false positive rate with which a server selects messages while hiding which messages are true positives. Our techniques are very similar to those used by [BLSV18] to construct blind IBE, a stronger variant of anonymous IBE. In that work, the authors define blind public key encryption and blind garbled circuits. Our notion of ambiguous garbling is a re-phrasing of their blind garbled circuit construction and can be built from the same tools. Their notion of blind public key encryption also describes loss of knowledge on the encrypted plaintext even in the presence of a secret key: a distinguisher with a secret key cannot distinguish between the encrypted message and a random value from the co-domain of the encryption scheme. They show this notion implies high leakage resilience and KDM security for linear functions of the secret key. It is likely that these two notions are non-trivially related to one another, although they may not be equivalent.

9 Conclusion

In this work we investigated the problem of developing fuzzy message detection schemes to support privacy-preserving retrieval of messages from store-and-forward delivery systems. We showed that these schemes can be realized efficiently using standard cryptographic techniques, with various trade-offs regarding the false-positive rate p and ciphertext size. This work leaves some open questions. First, we realized a construction of fractional FMD from a garbled circuit scheme, and noted that this scheme can be viewed as a form of *ambiguous* functional encryption; however, we leave the problem of formally defining this notion and finding alternative constructions to future work. More generally, this work raises broader questions about how to develop encryption schemes that exhibit well-defined behavior under decryption with incorrect secret keys. We believe that developing this area could enable new and interesting applications.

Acknowledgements. We would like to acknowledge support from the DARPA SIEVE program, the National Science Foundation under awards CNS-1653110, CNS-1801479, and DGE-1650441, the Tezos Foundation, and a Google Security & Privacy Award. The first author would also like to thank Henry de Valence for his comments.

References

[ABR01] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. The Oracle Diffie-Hellman Assumptions and an Analysis of DHIES. In *CT-RSA '01*, 2001.

[ABR14] M Abdalla, M Bellare, and P Rogaway. DHIES: An encryption scheme based on the Diffie-Hellman problem, contribution to IEEE P1363a, 1998, 2014.

[AGKS05] Masayuki Abe, Rosario Gennaro, Kaoru Kurosawa, and Victor Shoup. Tag-KEM/DEM: A new framework for hybrid encryption and a new analysis of kurosawa-desmedt kem: A new framework for hybrid encryption and a new analysis of Kurosawa-Desmedt KEM. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, pages 128–146, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[AKO09] Masayuki Abe, Eike Kiltz, and Tatsuaki Okamoto. Compact CCA-secure encryption for messages of arbitrary length. In *International Workshop on Public Key Cryptography*, pages 377–392. Springer, 2009.

[AS16] Sebastian Angel and Srinath T. V. Setty. Unobservable communication over fully untrusted infrastructure. In Kimberly Keeton and Timothy Roscoe, editors, *12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2016, Savannah, GA, USA, November 2-4, 2016*, pages 551–569. USENIX Association, 2016.

[BBDP01] Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-privacy in public-key encryption. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 566–582. Springer, 2001.

[BCG$^+$14] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy, SP 2014, Berkeley, CA, USA, May 18-21, 2014*, pages 459–474. IEEE Computer Society, 2014.

[BCOP04] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In *EUROCRYPT*, pages 506–522, 2004.

[BDG15] Nikita Borisov, George Danezis, and Ian Goldberg. DP5: A private presence service. *PoPETs*, 2015(2):4–24, 2015.

[BEM$^+$17] Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnés, and Bernhard Seefeld. Prochlo: Strong privacy for analytics in the crowd. In *Proceedings of the 26th Symposium on Operating Systems Principles, Shanghai, China, October 28-31, 2017*, pages 441–459. ACM, 2017.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In *CRYPTO*, pages 213–229, 2001.

[bit] How Many Bitcoin Users Are There? `https://www.buybitcoinworldwide.com/how-many-bitcoin-users/`. Accessed: 2021-04-16.

[BLSV18] Zvika Brakerski, Alex Lombardi, Gil Segev, and Vinod Vaikuntanathan. Anonymous ibe, leakage resilience and circular security from new assumptions. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 535–564. Springer, 2018.

[BMR90] Donald Beaver, Silvio Micali, and Phillip Rogaway. The round complexity of secure protocols. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 503–513, 1990.

[BS01] Dan Boneh and Igor Shparlinski. On the unpredictability of bits of the elliptic curve Diffie-Hellman scheme. In *CRYPTO '01*, 07 2001.

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *Theory of Cryptography*, pages 253–273, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[CDNO97] Ran Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable encryption. In Burton S. Kaliski Jr., editor, *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings*, volume 1294 of *Lecture Notes in Computer Science*, pages 90–104. Springer, 1997.

[CGN98] Benny Chor, Niv Gilboa, and Moni Naor. Private information retrieval by keywords. *IACR Cryptol. ePrint Arch.*, 1998:3, 1998.

[CHK04] Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In *EUROCRYPT*, pages 207–222, 2004.

[CKGS98] Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. Private information retrieval. *J. ACM*, 45(6):965–981, November 1998.

[CS19] R. Joseph Connor and Max Schuchard. Blind Bernoulli Trials: A noninteractive protocol for hidden-weight coin flips. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 1483–1500, Santa Clara, CA, August 2019. USENIX Association.

[DMS04] Roger Dingledine, Nick Mathewson, and Paul F. Syverson. Tor: The second-generation onion router. In Matt Blaze, editor, *Proceedings of the 13th USENIX Security Symposium, August 9-13, 2004, San Diego, CA, USA*, pages 303–320. USENIX, 2004.

[FGPS13] Nelly Fazio, Rosario Gennaro, Irippuge Milinda Perera, and William E. Skeith. Hard-core predicates for a Diffie-Hellman problem over finite fields. In Ran Canetti and Juan A. Garay, editors, *CRYPTO '13*, pages 148–165, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[FHK$^+$14] Martin Franz, Andreas Holzer, Stefan Katzenbeisser, Christian Schallhart, and Helmut Veith. CBMC-GC: An ANSI C Compiler for Secure Two-Party Computations. In Albert Cohen, editor, *Compiler Construction - 23rd International Conference, CC 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014. Proceedings*, volume 8409 of *Lecture Notes in Computer Science*, pages 244–249. Springer, 2014.

[fin] End-to-end encryption in Find My. `https://support.apple.com/guide/security/end-to-end-encryption-sec60fd770ba/web`. Accessed: 2020-01-20.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *CRYPTO '99*, volume 1666, pages 537–554, 1999.

[Goo] Google. Golang elliptic curve library.

[HHNZ19] M. Hastings, B. Hemenway, D. Noble, and S. Zdancewic. SoK: General purpose compilers for secure multi-party computation. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 1220–1237, 2019.

[i2p] i2p. Available at `https://geti2p.net`.

[Kil06] Eike Kiltz. Chosen-ciphertext security from tag-based encryption. In Shai Halevi and Tal Rabin, editors, *TCC '06*, volume 3876, pages 581–600. Springer, 2006.

[KR00] Hugo Krawczyk and Tal Rabin. Chameleon signatures. In *NDSS '00*, 2000.

[KS08] Vladimir Kolesnikov and Thomas Schneider. Improved garbled circuit: Free xor gates and applications. In *International Colloquium on Automata, Languages, and Programming*, pages 486–498. Springer, 2008.

[Lee19] Linda Naeun Lee. Zcash Reference Wallet Light Client Protocol. Available at `https://electriccoin.co/blog/zcash-reference-wallet-light-client-protocol/`, January 2019.

[Lun17] Joshua Lund. Technology preview: Sealed sender for Signal. `https://signal.org/blog/sealed-sender/`, 2017.

[LZ16] David Lazar and Nickolai Zeldovich. Alpenhorn: Bootstrapping secure communication without leaking metadata. In Kimberly Keeton and Timothy Roscoe, editors, *12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2016, Savannah, GA, USA, November 2-4, 2016*, pages 571–586. USENIX Association, 2016.

[mon] The Monero cryptocurrency. Available at `https://www.getmonero.org/`.

[Mon19] MoneroOutreach. Monero Wallet Quickstart. Available at `https://www.moneroutreach.org/stories/monero_wallet_quickstart.html`, April 2019.

[MRY04] Philip MacKenzie, Michael K. Reiter, and Ke Yang. Alternatives to non-malleability: Definitions, constructions, and applications. In Moni Naor, editor, *TCC '04*, volume 2951, pages 171–190. Springer, 2004.

[NM$^+$16] Shen Noether, Adam Mackenzie, et al. Ring confidential transactions. *Ledger*, 1:1–18, 2016.

[Sar21a] Sarah Jamie Lewis. Discreet log 1: Anonymity, bandwidth, and fuzzytags, Feb 2021.

[Sar21b] Sarah Jamie Lewis. fuzzy-tags sim git repository. Available at `https://git.openprivacy.ca/openprivacy/fuzzytags-sim`, 2021.

[Sha84] Adi Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, pages 47–53, 1984.

[Sho01] Victor Shoup. A proposal for an ISO standard for public key encryption. *IACR Crypto ePrint*, 2001:112, 2001.

[SS10] Amit Sahai and Hakan Seyalioglu. Worry-free encryption: Functional encryption with public keys. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, CCS '10, pages 463–472, New York, NY, USA, 2010. Association for Computing Machinery.

[vdHLZZ15] Jelle van den Hooff, David Lazar, Matei Zaharia, and Nickolai Zeldovich. Vuvuzela: scalable private messaging resistant to traffic analysis. In Ethan L. Miller and Steven Hand, editors, *Proceedings of the 25th Symposium on Operating Systems Principles, SOSP 2015, Monterey, CA, USA, October 4-7, 2015*, pages 137–152. ACM, 2015.

[WCFJ12] David Isaac Wolinsky, Henry Corrigan-Gibbs, Bryan Ford, and Aaron Johnson. Dissent in numbers: Making strong anonymity scale. In Chandu Thekkath and Amin Vahdat, editors, *10th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2012, Hollywood, CA, USA, October 8-10, 2012*, pages 179–182. USENIX Association, 2012.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets. In *Proceedings of the 27th Annual Symposium on Foundations of Computer Science*, SFCS '86, pages 162–167, Washington, DC, USA, 1986. IEEE Computer Society.

A Proof of Theorem 5 (Key privacy for FMD)

Here we briefly sketch an informal argument that any FMD scheme Π that satisfies correctness, fuzziness, and $\mathsf{DA\text{-}ATK}$ under the specific condition that $1 \in \mathcal{P}$ also satisfies a variant of key privacy for FMD.

We first define a variant of the key privacy experiment that is adapted to the specific setting of FMD. Here we replace the encryption algorithm with Flag, and decryption with Test (computed on any arbitrary false positive rate p') in a straightforward way.

Definition 16 ($\mathsf{IK\text{-}ATK\text{-}FMD}$)**.** Let $\Pi = (\mathsf{KeyGen}, \mathsf{Flag}, \mathsf{Extract}, \mathsf{Test})$ be an FMD scheme supporting the allowable probability set $\mathcal{P}$, and let the random variable
$\mathsf{IK\text{-}ATK\text{-}FMD}_b(\Pi, \mathcal{A}, \lambda)$ where $\mathcal{A}$, $b \in \{0,1\}$,
$\mathsf{ATK} \in \{\mathsf{CPA}, \mathsf{CCA}\}$, and $\lambda \in \mathbb{N}$ denote the result of the following probabilistic experiment:

```
$\underline{\mathsf{IK\text{-}ATK\text{-}FMD}_b(\Pi, \mathcal{A}, \lambda)\text{:}}$
    $(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(1^\lambda)$, $\forall i \in \{0,1\}$
    $C^* \leftarrow \mathsf{Flag}(pk_b)$
    $B \leftarrow \mathcal{A}^{\mathcal{O}_{sk_0,sk_1}(\cdot,\cdot,\cdot)}(pk_0, pk_1, C^*)$
    Output $B$
```

When $\mathsf{ATK} = \mathsf{CPA}$, $\mathcal{O}$ returns $\perp$ on all inputs. For $\mathsf{ATK} = \mathsf{CCA}$, the query $\mathcal{O}(id, C, p')$ returns $\perp$ if $C = C^*$ or $id \notin \{0,1\}$. Otherwise, the oracle computes $dsk \leftarrow \mathsf{Extract}(sk_{id}, p')$ and returns $\mathsf{Test}(dsk, C)$.

The $\mathsf{IK\text{-}ATK\text{-}FMD}(\Pi, \mathcal{A}, \lambda)$ advantage of $\mathcal{A}$ is defined as

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{IK\text{-}ATK\text{-}FMD}}(\lambda) = \left| \begin{array}{l} \mathbf{Pr}\left[\,\mathsf{IK\text{-}ATK\text{-}FMD}_0(\Pi, \mathcal{A}, \lambda) \Rightarrow 1\,\right] \\ -\,\mathbf{Pr}\left[\,\mathsf{IK\text{-}ATK\text{-}FMD}_1(\Pi, \mathcal{A}, \lambda) \Rightarrow 1\,\right] \end{array} \right|.$$

FMD scheme Π is $\mathsf{IK\text{-}ATK\text{-}FMD}$-secure if $\forall$ n.u.p.p.t. algorithms $\mathcal{A}$ and all $p \in \mathcal{P}$, $\mathbf{Adv}_{\mathcal{A}}^{\mathsf{DA\text{-}ATK}}(\lambda)$ is negligible.

We now sketch a proof of Theorem 5.

Proof sketch. Let $\mathcal{A}$ be some adversary with non-negligible advantage $\mathbf{Adv}_{\mathcal{A}}^{\mathsf{IK\text{-}ATK\text{-}FMD}}(\lambda)$ against FMD scheme Π. We will show there exists an adversary $\mathcal{B}$ against Π such that

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{IK\text{-}ATK\text{-}FMD}}(\lambda) \leq \mathbf{Adv}_{\mathcal{B}}^{\mathsf{DA\text{-}ATK}}(\lambda).$$

Briefly, our reduction works as follows: $\mathcal{B}$ interacts with the $\mathsf{DA\text{-}ATK}$ challenger with $p = 1$, and upon receiving pk_0, pk_1, C^* invokes $\mathcal{A}$ on input (pk_0, pk_1, C^*). If $\mathsf{ATK} = \mathsf{CCA}$, $\mathcal{B}$ forwards $\mathcal{A}$'s decryption queries to the $\mathsf{DA\text{-}ATK}$ oracle. When $\mathcal{A}$ outputs B, $\mathcal{B}$ outputs B as its result.

Note that because $p = 1$ then (by the *fuzziness* property of the FMD scheme) it must hold that the condition $\mathsf{Test}(dsk_0, C^*) = \mathsf{Test}(dsk_1, C^*)$ in the $\mathsf{DA\text{-}ATK}$ experiment is satisfied with probability (negligibly close to) 1. Thus the distribution that $\mathcal{A}$ receives is indistinguishable from the real $\mathsf{IK\text{-}ATK\text{-}FMD}$ experiment. Therefore if $\mathcal{A}$ distinguishes with advantage ϵ, then $\mathcal{B}$ must distinguish with advantage negligibly close to ϵ. $\square$

B The basic FMD construction is correct and fuzzy

Proof. Correctness for FMD derives from the correctness of the underlying scheme PKE. Note that if $n = 0$, then the test condition always succeeds. For $n > 0$, observe that for any valid flag ciphertext $C \leftarrow \mathsf{Flag}(pk)$, $C = C_1 \| \ldots \| C_n \| \ldots \| C_\gamma$ where $\forall i \in \{1, \ldots, n\}$ each $C_i = \mathsf{Encrypt}_{pk_i}(1)$. By the correctness property of PKE

it must hold that for each i, $\mathsf{Decrypt}_{sk_i}(C_i) = 1$ with probability 1, and hence Test will always output 1 on input a correct flag ciphertext.

Fuzziness requires a different approach. Let us assume by contradiction that for some $p = 2^{-n}$ where $n \in \{0, \ldots, \gamma\}$, the scheme FMD does not satisfy Definition 3. This implies that in the experiment of Definition 3, the Test algorithm's condition $s = 1^n$ succeeds with probability non-negligibly different from 2^{-n}. This has two implications:

1. Clearly $n > 0$, since the test condition succeeds with probability 1 when $n = 0$.
2. If $\forall i \in \{1, \ldots, n\}$ it holds that
$$(|\mathbf{Pr}\left[\mathsf{PKE.Dec}_{sk_i}(C_i) = 1\right]| - \frac{1}{2}) < \nu(\lambda),$$
then the test condition $s = 1^n$ will succeed with probability negligibly close to 2^{-n}.

Hence if we assume that the test condition $s = 1^n$ succeeds with probability non-negligibly *different* from 2^{-n}, it must be the case that at least one index i, $\mathbf{Pr}\left[\mathsf{PKE.Dec}_{sk_i}(C_i) = 1\right]$ is non-negligibly different from $1/2$. We will call this the *error condition.* Recalling that $C_i \leftarrow \mathsf{Enc}_{pk'_i}(1)$, the error condition is equivalent to
$$\Pr\left[\mathsf{PKE.Dec}_{sk}(C) = 1 \;\middle|\; \begin{array}{l} (pk, sk) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda), \\ (pk', sk') \leftarrow \mathsf{PKE.KeyGen}(1^\lambda), \\ C \leftarrow \mathsf{PKE.Enc}(pk', 1) \end{array}\right]$$
is non negligibly different from $\frac{1}{2}$. By Theorem 8, this probability must be negligibly close to $\frac{1}{2}$ and this completes our proof.

□

C Proof of Theorem 11

Proof. Let $pk_{\mathbf{0}} = \{pk_0^i\}_{i \in [\gamma]}$ and $pk_{\mathbf{1}} = \{pk_1^i\}_{i \in [\gamma]}$ be the public keys handed to the adversary. We give a proof for the *selective* case where $\mathcal{A}$ does not define p adaptively. Because the number of supported probabilities is polynomial in λ, by the lemma in Appendix H we can achieve full adaptive security. Our argument will now proceed by a series of hybrids, leveraging $\mathsf{IK\text{-}CPA}$ security for the unknown bit positions and uniform ambiguity for those in which the adversary knows the secret key. Consider the following sequence:

$$\mathcal{H}_0 = \left\{ \begin{array}{r} pk_0, pk_1, dsk_0, dsk_1, \mathsf{PKE.Enc}_{pk_0^1}(1), \ldots, \\ \mathsf{PKE.Enc}_{pk_0^\gamma}(1) \end{array} \right\}$$
$$\mathcal{H}_1 = \left\{ \begin{array}{r} pk_0, pk_1, dsk_0, dsk_1, \mathsf{PKE.Enc}_{pk_0^1}(1), \ldots, \\ \mathsf{PKE.Enc}_{pk_0^{\gamma-1}}(1), \mathsf{PKE.Enc}_{pk_1^\gamma}(1) \end{array} \right\}$$
$$\vdots$$
$$\mathcal{H}_{\gamma-n} = \left\{ \begin{array}{r} pk_0, pk_1, dsk_0, dsk_1, \mathsf{PKE.Enc}_{pk_0^1}(1), \ldots, \\ \mathsf{PKE.Enc}_{pk_0^{n-1}}(1), \mathsf{PKE.Enc}_{pk_1^n}(1), \ldots, \\ \mathsf{PKE.Enc}_{pk_1^\gamma}(1) \end{array} \right\}$$
$$\mathcal{H}_{\gamma-n+1} = \left\{ \begin{array}{r} pk_0, pk_1, dsk_0, dsk_1, \mathsf{PKE.Enc}_{pk_0^1}(1), \ldots, \\ \mathsf{PKE.Enc}_{pk_1^{n-1}}(1), \mathsf{PKE.Enc}_{pk_1^n}(1), \ldots, \\ \mathsf{PKE.Enc}_{pk_1^\gamma}(1) \end{array} \right\}$$

$$\vdots$$

$$\mathcal{H}_{\gamma-1} = \left\{ \begin{matrix} pk_0, pk_1, dsk_0, dsk_1, \mathsf{PKE.Enc}_{pk_1^1}(1), \ldots, \\ \mathsf{PKE.Enc}_{pk_1^\gamma}(1) \end{matrix} \right\}$$

If $\mathcal{H}_0 \approx \mathcal{H}_{\gamma-1}$, then Figure 2 is secure in the DA-CPA game.

Claim 17. *For $0 \le i < \gamma - n$, $\mathcal{H}_i \approx \mathcal{H}_{i+1}$.*

Proof. Suppose there exists a distinguisher D who can differentiate between $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$ with non negligible advantage. This distinguisher violates the IK-CPA security of PKE: we can use it to construct an adversary $\mathcal{B}$ that wins the IK-CPA game with non negligible advantage. An algorithm for such an adversary follows:

Description of adversary $\mathcal{B}$:

1. Receive pk_0^* and pk_1^* from the IK-CPA challenger
2. Challenge on the message $m = 1$ and receive c^*
3. $\forall j \in [\gamma - i - 1]$ correctly generate public private key pairs $(pk_0^j, sk_0^j), (pk_1^j, sk_1^j)$ using $\mathsf{PKE.KeyGen}(1^\lambda)$. $\forall j \in [\gamma - i + 1, \gamma]$ generate (pk_1^j, sk_1^j) using $\mathsf{PKE.KeyGen}(1^\lambda)$. Then set the following values:
   $pk_0 = (pk_0^1, \ldots, pk_0^{\gamma-i-1}, pk_0^*, {pk_1}^{\gamma-i+1} \ldots pk_1^\gamma)$
   $pk_1 = (pk_1^1, \ldots, pk_1^{\gamma-i-1}, pk_1^*, pk_1^{\gamma-i+1} \ldots pk_1^\gamma)$
   $dsk_0 = (sk_0^1, \ldots, sk_0^n)$
   $dsk_1 = (sk_1^1, \ldots, sk_1^n)$
   $C \leftarrow \mathsf{PKE.Enc}_{pk_0^1}(1) \| \mathsf{PKE.Enc}_{pk_0^2}(1) \| \ldots$
   $\| \mathsf{PKE.Enc}_{pk_0^{\gamma-i-1}}(1) \| c^* \| \mathsf{PKE.Enc}_{pk_1^{\gamma-i+1}}(1) \|$
   $\ldots \| \mathsf{PKE.Enc}_{pk_1^\gamma}(1)$
   give $pk_0, pk_1, dsk_0, dsk_1, C$ to the distinguisher D.
4. Receive b' from D and return b'.

Since the distinguisher D succeeds with non-negligible advantage and $\mathcal{B}$ succeeds when D succeeds because D's response corresponds precisely to which key the message has been encrypted under, $\mathcal{B}$ wins with non negligible advantage in the IK-CPA game. Therefore, $\mathcal{H}_i \approx \mathcal{H}_{i+1}$ and more specifically,

$$|\mathbf{Adv}_D^{\mathcal{H}_{i+1}}(\lambda) - \mathbf{Adv}_D^{\mathcal{H}_i}(\lambda)| \le \mathbf{Adv}_{\mathcal{B}}^{\mathsf{IK\text{-}CPA}}(\lambda)$$

□

Claim 18. *For $\gamma - n \le i < \gamma - 1$, $\mathcal{H}_i \approx \mathcal{H}_{i+1}$*

Proof. Suppose there exists a distinguisher D who distinguishes between $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$ with more than non-negligible advantage. Then there is an adversary $\mathcal{E}$ who can use D to win the $\mathcal{D}$-AMB game where $\mathcal{D}$ is the uniform distribution over $\{0, 1\}$.

Description of adversary $\mathcal{E}$:

1. Receive $(pk_0^*, pk_1^*, sk_0^*, sk_1^*, c)$. If $\mathsf{PKE.Dec}(sk_0^*, c) = 0$ or $\mathsf{PKE.Dec}(sk_1^*, c) = 0$, guess bit b' at random.
2. $\forall j \in [\gamma - i - 1]$, generate (pk_0^j, sk_0^j),, (pk_1^j, sk_1^j) by calling $\mathsf{PKE.KeyGen}(1^\lambda)$. For $j \in [\gamma - i + 1, \gamma]$ generate (pk_1^j, sk_1^j). Then set the following values:

$$\begin{aligned}
pk_0 &= (pk_0^1 \dots pk_0^{\gamma-i-1}, pk_0^*, pk_1^{\gamma-i+1}, \dots pk_1^{\gamma})\\
pk_1 &= (pk_1^1 \dots pk_1^{\gamma-i-1}, pk_1^*, pk_1^{\gamma-i+1}, \dots pk_1^{\gamma})\\
dsk_0 &= (sk_0^1, \dots sk_0^{\gamma-i-1}, sk_0^*, sk_0^{\gamma-i+1}, \dots sk_0^{n})\\
dsk_1 &= (sk_1^1, \dots, sk_1^{\gamma-i-1}, sk_1^*, sk_1^{\gamma-i+1} \dots sk_1^{n})\\
C &\leftarrow \mathsf{PKE.Enc}_{pk_0^1}(1)\| \dots \|\mathsf{PKE.Enc}_{pk_0^{\gamma-i-1}}(1)\|c^*\|\\
&\quad \mathsf{PKE.Enc}_{pk_1^{\gamma-i+1}}(1)\| \dots \|\mathsf{PKE.Enc}_{pk_1^{\gamma}}(1)
\end{aligned}$$

$\mathcal{E}$ gives $pk_0, pk_1, dsk_0, dsk_1, C$ to D.

3. Upon receiving b' from D forward onto the challenger b'.

Since $\mathsf{Test}(dsk_0, c) = \mathsf{Test}(dsk_1, c)$ implies that $\mathsf{PKE.Dec}(sk_0^{\gamma-i}, c_i) = \mathsf{PKE.Dec}(sk_1^{\gamma-i}, c_i)$ the distribution we are giving to the distinguisher is correct. Step 2 of $\mathcal{B}$ is executed with probability $\frac{1}{4}$: the encryption scheme is uniformly ambiguous so with probability $\frac{1}{2}$ the message is 1 and from the fuzziness proof the probability that $\mathsf{PKE.Dec}(sk_{\bar{b}}, C) = 1$ for a uniformly ambiguous, $\mathsf{IND}-\mathsf{CPA}$ secure bitwise PKE scheme is $\frac{1}{2}$. Therefore, $\mathcal{H}_i \approx \mathcal{H}_{i+1}$ and

$$|\mathbf{Adv}_D^{\mathcal{H}_{i+1}}(\lambda) - \mathbf{Adv}_D^{\mathcal{H}_i}(\lambda)| \leq 4\mathbf{Adv}_{\mathcal{E}}^{\mathcal{D}\text{-}\mathsf{AMB}}$$

□

Since the sequence $\mathcal{H}_0$, ..., $\mathcal{H}_{\gamma-1}$ is polynomial in λ by assumption, $\mathcal{H}_0 \approx \mathcal{H}_{\gamma-1}$. Therefore the scheme presented in 2 is secure under DA-CPA.

□

D Proof of Theorem 9 (Ambiguity of H-ELG)

Our main result in this section is an analysis of the *ambiguity* property of the H-ELG. This proof does not require any specific assumptions about the nature of the function H_ℓ, provided that it is efficient and produces a deterministic output.

Proof sketch. Let $\mathcal{A}$ be an adversary who has a non-negligible advantage in the $\mathcal{D}$-AMB-CPA experiment instantiated with $\mathcal{M} = \{0,1\}^\ell$. Let the challenge ciphertext $C^* = (c_0, c_1)$. We will now consider a hybrid experiment in which we replace $c_1 = H_\ell(g^k \| pk_b^k) \oplus m$ with a random string $c_1 \overset{\$}{\leftarrow} \{0,1\}^\ell$. It is clear that in this hybrid, $\mathcal{A}$ must have zero advantage: this holds because the adversary's view is now independent of the bit b chosen by the experiment. It remains only to demonstrate that, in $\mathcal{A}$'s view, this modified hybrid is distributed identically to the real experiment. We stress that the arguments in this proof do not depend on the nature of the hash function.

Notice that in the real experiment, $\mathcal{A}$ receives from the challenger a tuple $(pk_0, pk_1, sk_0, sk_1, C)$ with $C = (c_0, c_1)$. In the hybrid experiment, the only difference is the construction of c_1, which is changed from $c_1 = H_\ell(g^k \| pk_b^k) \oplus m$ where m is a uniformly-random string, to a random string. Clearly these two distributions are identical. Hence, since $\mathcal{A}$ must have zero advantage in the hybrid experiment, its advantage in the real experiment must also be 0. □

E Proof of Theorem 5.2

Proof. Observe that for any valid flag ciphertext $(u, y, \{c_i\}_{i\in[n]}) \leftarrow \mathsf{Flag}(pk)$, it holds that $\forall i \in \{1, \dots, n\}$ each $c_i = H(u\|{h_i}^r\|w) \oplus 1$. During Test we then have that for each $i \in [s]$, k_i is computed as $H(u\|u^{x_i}\|w) = H(u\|g^{r\cdot x_i}\|w) = H(u\|{h_i}^r\|w) = c_i$. Thus, $k_i \oplus c_i = 1, \forall i \in [s]$, which implies that Test returns 1 on input a correct flag ciphertext.

For fuzziness, let (pk, sk) and (pk', sk') be two key pairs generated by KeyGen. Let $dsk \leftarrow \mathsf{Extract}(sk, p)$, such that $x_1, \dots, x_s \leftarrow dsk$, $dsk' \leftarrow \mathsf{Extract}(sk', p)$, such that $x'_1, \dots, x'_s \leftarrow dsk'$, and $(u, y, \{c_i\}_{i\in[n]}) \leftarrow \mathsf{Flag}(pk')$. Then during the execution of $\mathsf{Test}(dsk, C)$, for each $i \in [s]$, we have that $k_i \leftarrow H(u\|g^{r\cdot x_i}\|w)$. We have that $x_i \neq x'_i$ with probability $(q-1)/q$. In this case since H is a random oracle, we have that k_i is a random bit, so that b_i is also a random bit, meaning it is 1 with probability $1/2$. The event that $b_i = 1$ for all $i \in [s]$ then happens with probability 2^{-s}. □

F Proof of Theorem 13 (Security of CCA-secure scheme)

Proof. This proof is done against the selective variant of the DA-CCA game. Combining this proof with the lemma presented in Appendix H allows us to achieve full adaptive security. We will prove selective security by constructing a series of hybrid games. For simplicity, we associate variables v^* with the challenge ciphertext for all variables v shown in Figure 3. The notation $\mathbf{DDH}_g(g^a, g^b, Z)$ refers to a call to a $\mathbf{DDH}$ oracle which returns 1 if $Z = g^{ab}$ and 0 otherwise. We let $sk_0 = x_1 \dots x_\gamma$, $sk_1 = y_1 \dots y_\gamma$, $dsk_0 = x_1 \dots x_n$, and $dsk_1 = y_1 \dots y_n$.

$\mathcal{H}_0 =$ This is the DA-CCA game, encrypting using key pk_b. Let $pk_0 = \{pk_0^1, \dots pk_0^\gamma\}$ and $pk_1 = \{pk_1^1, \dots, pk_1^\gamma\}$. Instead of generating public keys $pk_0^i = g^{x_i}$, $pk_1^i = g^{y_i}$ $\forall i \in [\gamma]$ with $x_i, y_i \xleftarrow{\$} \mathbb{Z}_q$, do the following:

```
$a \xleftarrow{\$} \mathbb{Z}_q \setminus \{0\}$
for $i \in [\gamma]$ :
  $x'_i, y'_i \xleftarrow{\$} \mathbb{Z}_q$
  $pk_0^i \leftarrow g^{a\cdot x'_i}$
  $pk_1^i \leftarrow g^{a\cdot y'_i}$
```

This implicitly defines $\forall i \in [\gamma]$, $sk_0^i = a \cdot x'_i$ and $sk_1^i = a \cdot y'_i$. The flag C^* is defined below according to the Flag algorithm:

$$C^* = (u^*, y^*, \{H(u^*\|{u^*}^{sk_i}\|w^*) \oplus 1\}_{i\in[\gamma]})$$

<u>Oracles H, G</u>: on input Z if Z has not been queried, query a random value, Q, from the respective co-domain. Return Q while recording Q as being the output of H/G on Z. If Z has been queried, return the value associated with Z.
<u>Test Oracle $\mathcal{O}$</u>: on input $(p' = \frac{1}{2^s}, \mathrm{id}, (u, y, \{c_i\}_{i\in[\gamma]}))$, if p' is in the appropriate range, query G on $u\|c_1\|\dots\|c_\gamma$ for m, then $\forall i \in [s]$ honestly query H on $u\|u^{sk_i}\|(g^m \cdot u^y)$ to get k_i. Return $\bigwedge_{i=1}^{s} \big(k_i \oplus c_i\big)$.

$\mathcal{H}_0$ is an equivalent reformulation of the original DA-CCA game. Let $x_i = a \cdot x'_i$ and $y_i = a \cdot y'_i$, then this is precisely the original security game

$\mathcal{H}_1 =$ If $G(\cdot)$ returns m^* for some other query Q made by $\mathcal{A}$, return $\perp$.

Claim 19. $\mathcal{H}_0 \approx \mathcal{H}_1$.

Since G is a random oracle, the probability that an adversary queries another such input which gives m^* is at most $\frac{q_G}{q}$ where $\mathcal{A}$ makes q_G queries to oracle G.

$$|\mathbf{Adv}_{\mathcal{A}}^{\mathcal{H}_1}(\lambda) - \mathbf{Adv}_{\mathcal{A}}^{\mathcal{H}_0}(\lambda)| \leq \frac{q_G}{q}$$

$\mathcal{H}_2 =$ Change Flag for the challenge ciphertext to be as follows:

```
$r \xleftarrow{\$} \mathbb{Z}_q$
$u \leftarrow g^r$
$m \xleftarrow{\$} \mathbb{Z}_q$
$y \xleftarrow{\$} \mathbb{Z}_q$
$w \leftarrow g^m \cdot u^y$
for $i \in [\gamma]$ :
    $k_i \leftarrow H(u \| u^{sk_b^i} \| w)$
    $c_i \leftarrow k_i \oplus 1$
Program $G$ to give $m$ on $(u \| c_1 \| \dots \| c_\gamma)$
Return $C \leftarrow (u, y, \{c_i\}_{i \in [\gamma]})$
```

Claim 20. $\mathcal{H}_1 = \mathcal{H}_2$.

In $\mathcal{H}_1$, the adversary sees $(u, y, \{c_i\}_{i \in [\gamma]})$ where $y = [(z - m)r^{-1}] \pmod q$ for $r \xleftarrow{\$} \mathbb{Z}_q$. Since r is is distributed uniformly at random, r^{-1} is also distributed uniformly at random and thus y is as well. $\{(u, \{c_i\}_{i \in [\gamma]})\}$ is the same in both $\mathcal{H}_1$ and $\mathcal{H}_2$: u and the input to $\mathcal{H}$ are unchanged. Therefore,

$$\mathbf{Adv}_{\mathcal{A}}^{\mathcal{H}_2}(\lambda) = \mathbf{Adv}_{\mathcal{A}}^{\mathcal{H}_1}(\lambda)$$

$\mathcal{H}_3 =$ The test oracle $\mathcal{O}$ is modified in the following way:
Test oracle $\mathcal{O}$: on input $(p', \text{id}, C = (u, y, \{c_i\}_{i \in [\gamma]}))$ if $u = u^*$, $g^{G(u \| c_1 \| \dots \| c_\gamma)} \cdot u^y = w^*$, and $C \neq C^*$ output $\perp$.

Claim 21. $\mathcal{H}_2 \approx \mathcal{H}_3$.

Assuming the DLOG assumption holds in $\mathbb{G}$ and given Lemma 1, the difference of the advantage of $\mathcal{A}$ in $\mathcal{H}_2$ and $\mathcal{H}_3$ is negligible:

$$|\mathbf{Adv}_{\mathcal{A}}^{\mathcal{H}_3}(\lambda) - \mathbf{Adv}_{\mathcal{A}}^{\mathcal{H}_2}(\lambda)| \leq \mathbf{Adv}_{C}^{\text{DLOG}}(\lambda)$$

where C is an adversary for $DLOG$ with run-time defined as in the lemma.

At this point, we split our argument into two separate cases. One in which $\mathcal{A}$ never queries H on $(u \| u^{sk_{id}^i} \| w)$ for any index $i \in [\gamma]$ which we call **Case1**, and the other where $\mathcal{A}$ does (**Case2**). We will start with the case in which the adversary does not make this query, and make the following claim:

*Assuming $\mathcal{A}$ wins and **Case 1** occurs, then $\mathcal{A}$ does not win with non-negligible advantage.*

Proof. First, we construct a series of hybrids $\mathcal{H}_{3+1}^1 \dots \mathcal{H}_{\gamma-n+3}^1$ where for $i \in [\gamma - n]$ hybrid $\mathcal{H}_{i+3}^1$, the hash oracle $H(\cdot)$ is queried on $u \| Z \| w$ for all positions $n + j$ where $1 \leq j \leq i$, with $Z \xleftarrow{\$} \mathbb{G}$ instead of $u \| u^{sk_b^{n+j}} \| w$.

Claim 22. $\forall i \in [\gamma - n - 1]$, $\mathcal{H}_{i+3}^1 = \mathcal{H}_{(i+1)+3}^1$

Since Z is in a prime order group, $\exists e \in \mathbb{Z}_q$ such that $Z = u^e$. Neither sk_0^i or sk_1^i are known to $\mathcal{A}$ and defining Z in this way is equivalent to sampling e uniformly at random from $\mathbb{Z}_q$. Since e has the same distribution as sk_0^i and sk_1^i for $i \in \{n+1, \dots \gamma\}$, these distributions are equivalent to one another.

Now consider the following hybrid, $\mathcal{H}_{\gamma-n+4}^1$, where $\mathcal{H}_{\gamma-n+4}^1$ differs in positions i of the flagged ciphertext $\forall i \in [n]$ and these positions are generated as follows: for index i sample $r_i \xleftarrow{\$} \mathbb{Z}_q$, until $H(u^* \| (u^*)^{r_i} \| w^*) =$

$H(u^*\|(u^*)^{sk_b^i}\|w^*)$. Because H is a random oracle with co-domain $\{0,1\}$,

$$\Pr(H(u^*\|(u^*)^{r_i}\|w^*) = H(u^*\|(u^*)^{sk_b^i}\|w^*) \mid r_i \xleftarrow{\$} \mathbb{Z}_q\} = \frac{1}{2}$$

In expectation, it takes 2 attempts to find such a satisfying r value for a single position i and by linearity of expectations it takes 2^n attempts on average to find satisfying r_i values for all positions. Therefore, it is possible to correctly generate this distribution in time polynomial in λ provided γ is logarithmic in λ.

Claim 23. $\mathcal{H}^1_{\gamma-n+3} \approx \mathcal{H}^1_{\gamma-n+4}$

The only parts of the ciphertext that are dependent on the secret key in $\mathcal{H}_{\gamma-n+3}$ are the keys k_i^* $\forall i \in [n]$ computed from the random oracle H. No other part of the flag is dependent on sk_b^i. However, since H gives the same response when sk_b^i or r_i is used, these two distributions are indistinguishable.
This completes our proof. As the flag C^* is now independent of b, the best strategy that $\mathcal{A}$ can employ is to guess b at random, so $\mathcal{A}$ succeeds with no advantage:

$$\mathbf{Adv}_{\mathcal{A}}^{\mathcal{H}^1_{\gamma-n+4}}(\lambda) = 0$$

We will now look at **Case 2** and make the following claim:

*Assuming $\mathcal{A}$ wins and **Case 2** occurs, $\exists B$ an algorithm which can be used to solve* Gap-DH *with non-negligible advantage.*

Proof. Given an instance $\{g, A = g^\alpha, B = g^\beta\}$ and oracle $\mathbf{DDH}.(\cdot,\ \cdot,\ \cdot)$, the algorithm B interacts with the DA-CCA adversary $\mathcal{A}$. Instead of generating $a \xleftarrow{\$} \mathbb{Z}_q \setminus \{0\}$, B generates pk_0 and pk_1 as

$$pk_0^i = A^{x_i}, pk_1^i = A^{y_i}$$

$\forall i \in [\gamma]$ where $a', x_i, y_i \xleftarrow{\$} \mathbb{Z}_q$. This implicitly defines the secret keys as $x_i \cdot \alpha$ and $y_i \cdot \alpha$. It flips $b \xleftarrow{\$} \{0,1\}$. C^* is constructed as $(u^*, y^*, \{c_i^*\}_{i\in[\gamma]})$ where $u^* = B$, $y^* \xleftarrow{\$} \mathbb{Z}_q$, and for each $i \in [\gamma]$, $k_b^i \xleftarrow{\$} \{0,1\}$, such that $c_i^* = k_b^i \oplus 1$. The random oracle H is then programmed, recording for each $i \in [\gamma]$, $(B, ???, g^{m^*}B^{y^*}, i, b)$ with key output k_b^i.
Oracle H:
on input $(u\|Z\|w)$ check for rows in H of the form $R = (u, ???, w,$
$i, id)$. For each satisfying row, query $\mathbf{DDH}_g(pk_{id}^i,\ u,\ Z)$ to get result τ. If $\tau = 1$, change $R \to R'$ where $R' = (u, Z, w, i, id)$. If no such row R exists in the table or $\tau = 0$, record input $(u, Z, w, ???, ???)$. If there is already an entry in the table, return k otherwise let $k \xleftarrow{\$} \{0,1\}$, record, and return k.
Test oracle $\mathcal{O}$: on input $(p' = \frac{1}{2^s}, id, u, y, \{c_i\}_{i\in[\gamma]})$, query G on $u\|c_1\|\ldots\|c_\gamma$ to get m and then set $w = g^m u^y$. For each $i \in [s]$, if there is a row in H of the form (u, Z, w, i, id) where $Z \neq ???$, return k associated with this input. Otherwise, look for rows in H of the form $R = (u, Z, w, ???, ???)$. If such a row exists, query $\mathbf{DDH}_g(pk_{id}^i,\ u,\ Z)$ to get τ. If $\tau = 1$, update $R \to R'$ where $R' = (u, Z, w, i, id)$ and return k associated with R. If no such row exists, look for a row $(u, ???, w, i, id)$ with output k and return k. If no such row exists, record $(u, ???, w, i, id)$ with output $k \xleftarrow{\$} \{0,1\}$.

Upon receiving b' from $\mathcal{A}$ if $b' = b$, randomly select a row of H, $R = (B, Z, w, i, id)$ where $i \in [\gamma]$. If $id = 0$, send $Z^{(x_i{}^{-1})}$ and if $id = 1$, send $Z^{(y_i{}^{-1})}$.
Analysis:
Since $\mathcal{A}$ needed to query on H to succeed in the experiment, any of these rows is a viable candidate to be an answer to the Gap-DH problem. If there are q_H queries made to H, then the probability of success for B

is $\frac{1}{q_H}$ multiplied by the probability of success for $\mathcal{A}$.

$$\mathbf{Adv}_{\mathcal{A}}^{\mathcal{H}_3'}(\lambda) \leq q_H \cdot \mathbf{Adv}_{B}^{\text{Gap-DH}}(\lambda)$$

□

Lemma 1. Suppose it is possible to distinguish between $\mathcal{H}_2$ and $\mathcal{H}_3$. If $\mathcal{A}$ runs in time τ, $\exists C$ a DLOG adversary who runs in time $\tau + O(3 \cdot g_{\text{OP}})$ where $g_{\text{OP}} = \{\times, +, ^{-1}\}$ are group operations in $\mathbb{G}$ and C has non-negligible advantage. More explicitly,

$$|\mathbf{Adv}_{\mathcal{A}}^{\mathcal{H}_3}(\lambda) - \mathbf{Adv}_{\mathcal{A}}^{\mathcal{H}_2}(\lambda)| \leq \mathbf{Adv}_{C}^{\text{DLOG}}(\lambda)$$

Proof. Define C on input g, Z as follows:

```
Set Flag $= (Z, y^*, \{c_i^*\})$, where $y^*, \{c_i\}_{i\in[\gamma]}$ are computed honestly according to $\mathcal{H}_2$
On input $(\{u, y, \{c_i\}_{i\in[\gamma]}\})$ to $\mathcal{O}$:
    if $g^{G(u\|c_1\|\ldots\|c_\gamma)} \cdot u^y = w^*$ :
        $z = (m - m^*)(y - y^*)^{-1}$
        Return $z$
```

Since $r^* \xleftarrow{\$} \mathbb{Z}_q$ there is negligible chance that $\mathcal{A}$ knows r^* such that $g^{r^*} = u^*$. It must also be true that $m \neq m^*$ (since $\mathcal{H}_1$ prevents this from happening). In addition, $y \neq y^*$ since if $y = y^*$, that would imply $m = m^*$. Therefore the discrete log of Z with respect to g is $(m - m^*)(y - y^*)^{-1} \pmod q$ since $g^m Z^y = g^{m^*} Z^{y^*}$. □

G Restricted FMD with Time Periods

In this section we sketch a variant of the CCA-secure restricted FMD scheme that incorporates the notion of multiple independent detection keys, one per time period. To do this, we must slightly modify the interface for an FMD scheme to enable the notion that detection keys also embed a time epoch, which must be a match with the time epoch used to generate the flag ciphertext. The critical properties such as fuzziness and detection ambiguity must continue to hold both with respect to other users' keys. A time-based Fuzzy Message Detection scheme (tbFMD) shares the same interface for the algorithms (Setup, KeyGen, Test) as a standard FMD scheme, but the remaining algorithms are modified as follows:

Flag$(pk, t) \to C$. On input a public key and a *time epoch* t, this randomized algorithm outputs a flag ciphertext C.

Extract$(sk, t, p) \to dsk$. On input a secret key sk, a time epoch t, and a false positive rate $p \in \mathcal{P}$, this algorithm extracts a detection key dsk, or outputs $\perp$ if $p \notin \mathcal{P}$.

A time epoch is assumed to be an integer that roughly corresponds to some pre-defined period of time. However, in practice an epoch may be encoded into an arbitrary string $\{0,1\}^*$ that contains additional data such as the identity of a server or other auxiliary information. The correctness definition for a tbFMD is identical to the definition for FMD except that the same additional input t must be provided to both the Flag and Test algorithms.

```
tbFMD.KeyGen(1^λ) :
    $x \leftarrow \mathbb{Z}_q$
    $h \leftarrow g^x$
    $sk \leftarrow x$
    $pk \leftarrow h$
    Return $sk, pk$

tbFMD.Extract(sk, t, p = 2^{-n}) :
    $x \leftarrow sk$
    for $i \in [n]$ :
        $f_i \leftarrow F(t||i)^x$
    Return $(f_1, \ldots, f_n)$

tbFMD.Flag(pk, t) :
    $h \leftarrow pk$
    $r \xleftarrow{\$} \mathbb{Z}_q$
    $u \leftarrow g^r$
    $z \xleftarrow{\$} \mathbb{Z}_q$
    $w \leftarrow g^z$
    for $i \in [\gamma]$ :
        $h_i \leftarrow F(t||i)$
        $k_i \leftarrow H(u \| e({h_i}^r, h) \| w)$
        $c_i \leftarrow k_i \oplus 1$
    $m \leftarrow G(u \| c_1 \| \ldots \| c_\gamma)$
    $y \leftarrow \left[(z - m)r^{-1}\right] \pmod{q}$
    Return $(u, y, \{c_i\}_{i \in [\gamma]})$

tbFMD.Test(dsk, (u, y, {c_i}_{i∈[γ]})) :
    $f_1, \ldots f_n \leftarrow dsk$
    $m \leftarrow G(u \| c_1 \| \ldots \| c_\gamma)$
    $w \leftarrow g^m \cdot u^y$
    for $i \in [n]$:
        $k_i \leftarrow H(u \| e(u, f_i) \| w)$
        $b_i \leftarrow k_i \oplus c_i$
    if $b_i = 1, \forall i \in [n]$ :
        Return 1
    else:
        Return 0
```

Figure 6: A CCA-secure efficient tbFMD scheme with restricted values of p ($p \approx 2^{-n}$). The groups $\mathbb{G}, \mathbb{G}_T$ and the parameter γ are assumed to be global constants generated by an (unspecified) setup procedure. $\mathbb{G}, \mathbb{G}_T$ are bilinear groups of prime order q, $F : \{0,1\}^* \to \mathbb{G}$, $H : \mathbb{G} \times \mathbb{G}_T \times \mathbb{G} \to \{0,1\}$, $G : \mathbb{G} \times \{0,1\}^\gamma \to \mathbb{Z}_q$ are hash functions.

A realization from Boneh-Franklin. The Boneh-Franklin IBE scheme [BF01] uses bilinear groups with an efficiently computable pairing. Let $\langle g \rangle = \mathbb{G}$ be a group of prime order q and let $\mathbb{G}_T$ be a *target group*. The pairing is defined as $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$.[12] We also require a new hash function $F : \{0,1\}^* \to \mathbb{G}$ that is modeled as a random oracle.

It is possible to prove that, under appropriate assumptions about the hash function, Boneh-Franklin itself is uniformly ambiguous. One could then consider a generic construction from the Boneh-Franklin scheme. However, this would make it challenging to achieve CCA security and an efficient construction directly. To achieve an efficient CCA-secure tbFMD we instead use the underlying Boneh-Franklin construction directly. Our construction is reminiscent of the CCA-secure construction from §5.2 and we present the details in Figure 6. We leave a security analysis of this scheme to future work.

H Selective Detection-Ambiguity implies Adaptive when $|\mathcal{P}|$ is polynomial

Lemma 2. Suppose $\Pi =$ (KeyGen, Flag, Extract, Test) is a FMD scheme secure against selective DA-ATK where ATK $\in$ {CPA, CCA} and $|\mathcal{P}| = p(\lambda)$ for polynomial p and security parameter λ. Then Π is adaptively secure i.e. secure in the sense of DA-ATK from Definition 4.

Proof. First, assume that this is false and we have a scheme that is secure against the selective variant of DA-ATK but not adaptively secure. We let $\mathcal{B}$ be an attacker for Π in the adaptive DA-ATK game and build from this an attacker $\mathcal{A}$, that results in distinguishably different output when $b = 0$ and when $b = 1$.

Description of $\mathcal{A}$:

1. Choose $p \xleftarrow{\$} \mathcal{P}$. Send this to challenger $\mathcal{C}$ and receive $pk_0, pk_1, dsk_0, dsk_1, C$. Send to $\mathcal{B}$ pk_0, pk_1 and C.
2. Receive from $\mathcal{B}$ the probability p'. If $p' \neq p$, output $B \xleftarrow{\$} \{0,1\}$.
3. If $p' = p$, send to $\mathcal{B}$ the detection keys dsk_0, dsk_1. Let B be the output of $\mathcal{B}$. Output B as well.

We now analyze $\mathcal{A}$. If p is chosen uniformly from $\mathcal{P}$ then we expect to match p' with probability at least $\frac{1}{p(\lambda)}$. In this case, we do not choose B at random but instead receive it from $\mathcal{B}$. Therefore if the difference between $\{\mathsf{DA\text{-}ATK}_0(\Pi, \mathcal{B}, \lambda)\}$ and $\{\mathsf{DA\text{-}ATK}_1(\Pi, \mathcal{B}, \lambda)\}$ is non negligible $\mu(n)$ then the absolute difference between $\mathsf{DA\text{-}ATK}_0$ and $\mathsf{DA\text{-}ATK}_1$ in the non selective game using algorithm $\mathcal{A}$ is at least $\mu(n) \cdot \frac{1}{p(\lambda}$. This

[12]This is known as the symmetric pairing setting. In practice, this may require the existence of an efficiently-computable isomorphism to a related group. These details are discussed in [BF01] and many subsequent works.

is non-negligible and violates the selective security of Π. Therefore, the scheme must also be adaptively secure. □

I Proof of Theorem 8 (Decryption with an incorrect key)

We start by presenting a lemma that applies to all $\mathsf{IND}-\mathsf{CPA}$ secure encryption schemes. This lemma uses a slightly different formulation of the incorrect message decryption experiment where instead of encrypting one message we encrypt according to a distribution $\mathcal{D}$. The following lemma proves that if $\mathsf{IND}-\mathsf{CPA}$ security holds the output distribution of incorrect decryption is always "the same".

Lemma 3. Let PKE be a IND-CPA secure public key encryption scheme over a message space $\mathcal{M}$. Let $\mathcal{D}$ be any probability distribution over $\mathcal{M}$. Consider the following experiment:

```
$\mathsf{ExpDistIncorrect}(\mathsf{PKE}, \lambda, \mathcal{D})$
    $(pk_i, sk_i) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda), \forall i \in \{0,1\}$
    $m \overset{\mathcal{D}}{\leftarrow} \mathcal{M}$
    $C \leftarrow \mathsf{PKE.Enc}(pk_0, m)$
    Output $\mathsf{PKE.Dec}(sk_1, C)$
```

Then $\forall m \in \mathcal{M}$, for all distributions $\mathcal{D}$, $\mathsf{ExpDistIncorrect}(\mathsf{PKE}, \lambda, \mathcal{D})$ is computationally indistinguishable from $\mathsf{ExpDistIncorrect}(\mathsf{PKE}, \lambda, p_m)$ where p_m is the probability distribution over a random variable M with $\Pr(M = m) = 1$ and $\forall \bar{m} \in \mathcal{M} \setminus \{m\}$, $\Pr(M = \bar{m}) = 0$.

Proof. Suppose this is not the case and that there exists some distribution $\mathcal{D}^*$, a message m^*, and a distinguisher $\mathcal{B}$ that can distinguish between the previous experiments with better than negligible advantage. This immediately begets an IND-CPA attacker with related advantage. We describe the attacker briefly below.

Description of adversary $\mathcal{A}$: Receive pk from challenger $\mathcal{C}$. Sample $m_0 \overset{\mathcal{D}^*}{\leftarrow} \mathcal{M}$ and set $m_1 = m^*$. Send the messages to the challenger and receive ciphertext c. Generate a random key pair $\overline{pk}, \overline{sk} \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$ and attempt decryption to get $\overline{m} = \mathsf{PKE.Dec}(\overline{sk}, c)$. Send to $\mathcal{B}$ the decrypted message $\overline{m}$. If $\mathcal{B}$ responds with b' respond with b' to the challenger. The advantage of the adversary in this game is equal to that of the distinguisher $\mathcal{B}$: the distribution given to $\mathcal{B}$ is the same as they would see in the regular experiment as the output of both is an incorrect decryption based either on distribution $\mathcal{D}^*$ or p_{m^*} and thus if $\mathcal{B}$ correctly deduces which distribution was used for encryption so will $\mathcal{A}$. □

We now turn our attention back to the main theorem. Let PKE be an ambiguous encryption scheme in the sense of Definition 7. By this definition, PKE is (at minimum) IND-CPA secure, and Lemma 3 applies. We now assume our conclusion is false: PKE is a $\mathcal{D}$-ambiguous encryption scheme secure under CPA but there exists a message m^* such that $\mathsf{ExpMsgIncorrect}(\mathsf{PKE}, \lambda)$ is not computationally indistinguishable from $\mathcal{D}$. Let one such efficient distinguisher be $\mathcal{B}$. We first make the observation that our original experiment $\mathsf{ExpMsgIncorrect}(\mathsf{PKE}, \lambda)$ with m^* as m is equivalent to
$\mathsf{ExpDistIncorrect}(\mathsf{PKE}, \lambda, p_m^*)$ where p_{m^*} is the probability distribution that always encrypts m^*. We now split our argument into two cases:

1. $\mathcal{B}$ can distinguish between $\mathsf{ExpDistIncorrect}(\mathsf{PKE}, \lambda, p_{m^*})$ and $\mathsf{ExpDistIncorrect}(\mathsf{PKE}, \lambda, \mathcal{D})$
2. $\mathcal{B}$ cannot distinguish between the previous distributions

Case 1 cannot be true because it would violate the $\mathsf{IND}-\mathsf{CPA}$ security of PKE via Lemma 3.

Now assume we are in Case 2 and these two distributions are indistinguishable to $\mathcal{B}$. We now show that $\mathcal{B}$ can be used to distinguish in the $\mathcal{D}$-AMB-CPA security game with non negligible advantage.

Description of distinguisher: Receive $pk_0, pk_1, sk_0, sk_1, c$. Let $\overline{m} = \mathsf{PKE.Dec}(sk_0, c)$. Send $\overline{m}$ to $\mathcal{B}$. If $\mathcal{B}$ responds with b' output b'.

Normally, $\mathcal{B}$ is a distinguisher that distinguishes between the distributions $\mathcal{D}$ and $\mathsf{ExpDistIncorrect}(\mathsf{PKE}, \lambda, p_{m^*})$ with non-negligible advantage. However, the distributions handed to $\mathcal{B}$ by this distinguisher are correctly decrypted ciphertext drawn from $\mathcal{D}$ and $\mathsf{ExpDistIncorrect}(\mathsf{PKE}, \lambda, \mathcal{D})$. The first of these is precisely the distribution $\mathcal{D}$ and the second is computationally indistinguishable from $\mathsf{ExpDistIncorrect}(\mathsf{PKE}, \lambda, p_{m^*})$. Therefore the distributions that $\mathcal{B}$ sees are correctly structured. When $\mathcal{B}$ is correct, b' determines whether or not the decryption was correctly done, allowing the distinguisher to deduce the key that was used for encryption. Thus the above distinguisher has the same non-negligible advantage as $\mathcal{B}$, in contradiction to our assumption, and for all messages $m \in \mathcal{M}$ the distributions $\mathsf{ExpMsgIncorrect}(\mathsf{PKE}, \lambda)$ and $\mathcal{D}$ must be computationally indistinguishable.

J Security of Fractional Probability Construction

J.1 Proof Sketch of Theorem 15

Proof sketch. Our argument consists of a sequence of hybrids ending in a final hybrid $\mathcal{H}_2$ where the flag C is not constructed using the bit b or either pair (mpk_0, msk_0), (mpk_1, msk_1). We first give a description of this final hybrid:

$\mathcal{H}_2$ = The challenger creates two public/private key pairs for the $\mathsf{FracFMD}$ scheme as specified in Figure 4 (i.e. $mpk = \{pk^i_j\}_{i\in\{0,1\},j\in[\gamma]}$ and $(pk^i_j, \_) \leftarrow \mathsf{PKE.KeyGen}()$ $\forall i \in \{0,1\}, j \in [\gamma]$ where PKE is uniform -$\mathsf{AMB\text{-}CPA}$ secure and msk is likewise defined). The challenger then constructs the garbled circuit and labels as follows.

1. Sample one set of labels $\{L^i_j\}_{i\in\{0,1\},j\in[\kappa+\gamma]}$ where each label L is sampled uniformly from $\{0,1\}^\lambda$.
2. Generate a simulated garbled circuit $\tilde{C}$ as follows: For every internal gate g in $\mathbf{C}$, generate the garbled table $(R_1, \dots R_4)$ as:

$$R_i \xleftarrow{\$} \{0,1\}^{\lambda+1}, \forall i \in [4]$$

For every gate g connected to an output wire, generate the garbled table $(R_1, \dots R_4)$ as:

$$R_i \xleftarrow{\$} \{0,1\}, \forall i \in [4]$$

A third unrelated key pair is generated, denoted by (mpk_2, msk_2), and used to encrypt all labels in $\{L^i_j\}$ corresponding to the γ wires for the modulus reduction. This produces ciphertext values $\{c^i_j\}_{i\in\{0,1\},j\in[\gamma]}$. The challenger samples $R \leftarrow \{0,1\}^{\kappa+\gamma}$ then sets $C = (\mathbf{C}, \{c^i_j\}_{i\in\{0,1\},j\in[\gamma]}, \{L_{i,R_i}\}_{i\in[\kappa+\gamma]})$ where $R_1 \dots R_{\kappa+\gamma} \leftarrow \mathrm{bin}(R)$. The challenger then sends C, mpk_0, mpk_1 to $\mathcal{A}$. Once $\mathcal{A}$ sends p, the challenger runs $dsk_0 \leftarrow \mathsf{Extract}(msk_0, p)$, $dsk_1 \leftarrow \mathsf{Extract}(msk_1, p)$, $\mathsf{Test}(dsk_0, C) = z_1$, and $\mathsf{Test}(dsk_1, C) = z_2$. If $z_1 = z_2 = 1$, the challenger sends dsk_0, dsk_1 to the adversary, otherwise it outputs a bit b uniformly. Otherwise, the challenger rewinds the adversary and attempts the game again. If the challenger has repeated $(\frac{N}{M_{\min}})^2$ times where $m_{\min}$ is the smallest probability asked for by $\mathcal{A}$ without finding a C such that $\mathsf{Test}(dsk_0, C) = \mathsf{Test}(dsk_1, C)$, choose b' uniformly at random. We now give a sequence of hybrids to prove this is indistinguishable from the normal $\mathsf{DA\text{-}CPA}$ security game. Let $\mathcal{H}_\mathbf{0}$ be the real security game.

$\mathcal{H}_\mathbf{1}$. Modify game $\mathcal{H}_\mathbf{0}$ as follows: instead of honestly constructing labels $\{L^i_j\}_{i\in\{0,1\},j\in[\gamma]}$, generate (mpk_2, msk_2) $\leftarrow \mathsf{FracFMD.KeyGen}(1^\lambda)$ and calculate $\{c^i_j\}_{i\in\{0,1\},j\in[\gamma]]}$ as $c^i_j = \mathsf{PKE.Enc}(\overline{pk^i_j}, \bar{L}^i_j)$ where $\bar{L}^i_j \xleftarrow{\$} \{0,1\}^{\lambda+1}$ and $mpk_2 = \{\overline{pk^i_j}\}_{i\in\{0,1\},j\in[\gamma]}$. Choose $b \xleftarrow{\$} \{0,1\}$ and calculate the garbled circuit labels as $L^i_j = \mathsf{PKE.Dec}(sk^i_j, c^i_j)$ where $msk_b = \{sk^i_j\}_{i\in\{0,1\},j\in[\gamma]}$. Construct the rest of the circuit correctly: generating the labels corresponding to R uniformly from bit strings of the appropriate length and all tables honestly.

Claim 24. $\mathcal{H}_\mathbf{0} \overset{c}{\approx} \mathcal{H}_\mathbf{1}$ *if* PKE *is secure under uniform-*$\mathsf{AMB\text{-}CPA}$.

We make this argument by leveraging the fact that we are using a uniformly ambiguous public key scheme. The circuit itself is correctly structured w.r.t. the labels that it will be evaluated on, as these incorrect decrypted labels were used to garble the circuit. It is also the case that mpk_2, msk_2 are not within the view of the adversary at all. Indeed the only way in which an adversary could distinguish between $\mathcal{H}_\mathbf{0}$ and $\mathcal{H}_\mathbf{1}$ is if the probability distribution over the following random variable M where M represents incorrect decryption i.e. $M = \mathsf{Dec}(sk_j^i, c_j^i)$ where $c_j^i \leftarrow \mathsf{Enc}(\overline{pk_j^i}, \overline{L_j^i})$, $\overline{L_j^i} \xleftarrow{\$} \{0,1\}^{\lambda+1}$, and $(mpk, msk) \leftarrow \mathsf{FracFMD.KeyGen}(1^\lambda)$ is not indistinguishable from uniform. However, from Theorem 8, this distribution must be computationally indistinguishable from uniform and thus these two hybrids are computationally indistinguishable.

$\mathcal{H}_\mathbf{2}$. This hybrid modifies $\mathcal{H}_\mathbf{1}$ as follows: replace each internal gate in $\tilde{C}$ with a gate table comprising four random bitstrings of the form $R_i \xleftarrow{\$} \{0,1\}^{\lambda+1}$ for each $i \in \{0, \dots, 3\}$. Replace each output gate with a gate table comprising four random bits of the form $R_i \xleftarrow{\$} \{0,1\}$ for each $i \in \{0, \dots, 3\}$.

Claim 25. *$\mathcal{H}_\mathbf{2} \overset{c}{\approx} \mathcal{H}_\mathbf{1}$ if $\overline{\mathsf{Enc}}$ is pseudorandom.*

This argument proceeds via a series of $|I+1|$ sub-hybrids, beginning with the "input gates" (gates connected solely to input wires) and proceeding recursively to the output gates of the circuit. In $\bar{\mathcal{H}}_\mathbf{I+1}$ we replace all output gates of the circuit simultaneously.

$\bar{\mathcal{H}}_\mathbf{0} \dots \bar{\mathcal{H}}_{|\mathbf{I}|+\mathbf{1}}$: Order the gates of the circuit $\mathbf{C}$ so that gates that appear earlier in the ordering are either below or at the same depth as those that appear afterward. In hybrid $\bar{\mathcal{H}}_\mathbf{k}$ for $k \in \{1, \dots, I\}$ where I is the number of non-output gates, we replace the garbled table $T = (R_1, R_2, R_3, R_4)$ corresponding to the non-output gate at index k by choosing row values from the uniform distribution over $\{0,1\}^{\lambda+1}$. In hybrid $\bar{\mathcal{H}}_\mathbf{I+1}$ for each output gate index k replace the garbled table $T = (R_1, R_2, R_3, R_4)$ with new row values $R_i \xleftarrow{\$} \{0,1\}$. In each hybrid, keep track of the probabilities p asked for by $\mathcal{A}$ where the first probability asked for is p'. If p' is smaller than $v(\lambda)$ then output $b' \xleftarrow{\$} \{0,1\}$ and try again. Let $p_{\min}$ be the smallest probability that is not negligible in the security parameter that $\mathcal{A}$ asks for and let i be a counter. Each time $\mathcal{A}$ requests detection keys for some probability p, extract the relevant keys and run Test using dsk_0, dsk_1. If either returns 0 and $i < \frac{1}{p_{\min}^2}$, run the adversary again, regenerating an entirely new circuit. If $i \geq \frac{1}{p_{\min}^2}$ and either detection key fails, output $\perp$. Else let b' be the output of $\mathcal{A}$. Output b'.

Claim 26. *For each $k \in \{1, \dots, I\}, \bar{\mathcal{H}}_\mathbf{k-1} \overset{c}{\approx} \bar{\mathcal{H}}_\mathbf{k}$.*

Rows $j \neq i$: Consider any one of the three rows $j \neq i$, where in $\bar{\mathcal{H}}_\mathbf{k-1}$, $R_j = \overline{\mathsf{Enc}}(L_\alpha^a, L_\beta^b, m)$ and either L_α^a is unknown or L_β^b is unknown to $\mathcal{A}$ since it does not correspond to one on the rows $\mathcal{A}$ can decrypt. Without loss of generality, assume $\mathcal{A}$ does not know L_α^a. Then in $\bar{\mathcal{H}}_\mathbf{k}$ where each R_j has been replaced with a random string, L_α^a is indistinguishable from a random element of $\{0,1\}^{\lambda+1}$ and therefore $\overline{\mathsf{Enc}}(L_\alpha^a, L_\beta^b, m)$ is indistinguishable from a random element of $\{0,1\}^{\lambda+1}$. Thus if it is *not* the case that $\mathcal{H}_{k-1} \overset{c}{\approx} \mathcal{H}_k$, there exists a distinguisher against the underlying PRG used in $\overline{\mathsf{Enc}}$.

Row i: Consider in $\bar{\mathcal{H}}_\mathbf{k-1}$ the row $R_i = \overline{\mathsf{Enc}}(L_\alpha^a, L_\beta^b, m_0)$ where both L_α^a, L_β^b are known to $\mathcal{A}$. Here we more precisely define the row $R_i = \mathsf{PRG}(L_\alpha^a, L_\beta^b) \oplus m_0$ where $m_0 \in_R \{0,1\}^{\lambda+1}$, hence the distribution of R_i in $\bar{\mathcal{H}}_\mathbf{k-1}$ is identical to that of $\bar{\mathcal{H}}_\mathbf{k}$ as they are both uniformly random. Note further that m_0 implicitly defines one label and point-and-permute bit for the gate's output wire. Since all wire labels and point-and-permute bits are drawn from the uniform distribution, the distribution of all labels is identical between $\bar{\mathcal{H}}_\mathbf{k-1}$ and $\bar{\mathcal{H}}_\mathbf{k}$.

We make an argument that all subsequent gate evaluations and output distributions are indistinguishable between $\bar{\mathcal{H}}_\mathbf{k-1}$ and $\bar{\mathcal{H}}_\mathbf{k}$ and moreover that hybrid $\bar{\mathcal{H}}_\mathbf{k}$ runs in expected polynomial time. Consider the gate evaluations of all gates that take as input a label and permute bit which were the result of an incorrect gate evaluation. For these gates, the same argument applies as for the gate at index k. That is, the distribution is uniform - since the output of incorrect evaluation is a uniform value when the encrypted value is also uniform - and this is conveniently the expected distribution, since the output of each gate are input labels for another gate. Of course, this argument does not apply to the affected output gates. In general, this is because the first bit of each label is the actual output of the circuit and therefore NOT random. Recall that for any output row i, $R_i = \mathsf{PRG}(\mathsf{L}^{\mathsf{a}}_{\alpha}, \mathsf{L}^{\mathsf{b}}_{\beta}) \oplus \mathsf{m_0}$ is the value sitting at the row which is then evaluated *incorrectly* as $R_i \oplus \mathsf{PRG}(\hat{\mathsf{L}}^{\mathsf{a}}_{\alpha}, \hat{\mathsf{L}}^{\mathsf{b}}_{\beta})$ where either $\hat{L}^a_\alpha \neq L^a_\alpha$ or $\hat{L}^b_\beta \neq L^b_\beta$. Because at least one of the correct labels is unknown, $\mathsf{PRG}(\mathsf{L}^{\mathsf{a}}_{\alpha}, \mathsf{L}^{\mathsf{b}}_{\beta})$ is uniform by the security of the pseudorandom number generator and thus the output of evaluation is a uniform bit for that gate. Assuming that our circuit originally gave output uniformly in the range $[0, M)$ [see Lemma 4] and the gates which are affected are flipping bits uniformly, when Test returns 1 the overall output of the circuit is still uniform in the range $[0, M)$.

The adversary must ask for non negligible p a non-negligible fraction of the time. If not, it is clear $\mathcal{A}$ does not have non negligible advantage since the challenger will output a uniform bit b except with negligible probability by the security game definition. For the least of these, $p_{\min}$, the expected number of times we have to rewind $\mathcal{A}$ before one detection key results in a 1 is less than or equal to $\frac{1}{p_{\min}}$, by the fuzziness of our scheme. Therefore, by independence, the expected number of retries for constructing a ciphertext C such that $\mathsf{Test}(dsk_0, C) = \mathsf{Test}(dsk_1, C) = 1$ must be less than or equal to $\frac{1}{p^2_{\min}}$. Because $p_{\min}$ is not negligible $\frac{1}{p^2_{\min}}$ is not exponential and the running time of the hybrid is expected polynomial.

Since in $\boldsymbol{\mathcal{H}_2}$ the flag C is not based on the bit b at all, the adversary's advantage must be negligible in $\boldsymbol{\mathcal{H}_2}$ and since $\boldsymbol{\mathcal{H}_2}$ has a distribution which is computationally close to $\boldsymbol{\mathcal{H}_0}$ the adversary must have negligible advantage in the original security game as well. This completes our proof.

□

Lemma 4. Let our security parameter be κ, $f(x, y) = x \pmod{y}$ the mod function where $f : \{0,1\}^{|\mathcal{I}_1|} \times \{0,1\}^{|\mathcal{I}_2|} \to \{0,1\}^{|\mathcal{O}|}$. If $|\mathcal{I}_1| = \kappa + |\mathcal{I}_2|$, then $\forall y \in \{0,1\}^{|\mathcal{I}_2|}$, $\{f(x, y) \mid x \xleftarrow{\$} \{0,1\}^{\mathcal{I}_1}\} \approx D$ where D is the uniform distribution over $[0, y)$.

J.2 Proof Sketch of Lemma 4

Proof sketch. Let $y \in \{0,1\}^{\mathcal{I}_2}$ and $2^{|\mathcal{I}_1|} = qy + b$ for some $0 \leq b < y$ by the division algorithm. For $0 \leq i < b$, the probability that i is output from f is $\frac{q+1}{2^{|\mathcal{I}_1|}}$. For all values i where $b \leq i < y$ the probability is $\frac{q}{2^{|\mathcal{I}_1|}}$. We then have the following inequality: $\frac{q}{2^{|\mathcal{I}_1|}} \leq \frac{q}{2^{|\mathcal{I}_1|} - b} = \frac{1}{y} \leq \frac{q+1}{2^{|\mathcal{I}_1|}}$. As $|\mathcal{I}_1| = \kappa + |\mathcal{I}_2|$ the difference between $\frac{q+1}{2^{|\mathcal{I}_1|}}$ and $\frac{q}{2^{|\mathcal{I}_1|}}$ is $\frac{1}{2^{|\mathcal{I}_1|}}$. Since $\frac{1}{2^{|\mathcal{I}_1|}}$ is negligible in the security parameter κ, the difference between $\frac{1}{y}$ and either of these quantities is also negligible in the security parameter. □